Numerical Methods for Partial Differential Equations

Prof. Ralf Hiptmair, Prof. Christoph Schwab,
Prof. H. Harbrecht, Dr. V. Gradinaru, and Dr. A. Chernov

Draft version December 14, 2010, SVN rev. 30831

(C) Seminar für Angewandte Mathematik, ETH Zürich

# Contents

1 Case Study: A Two-point Boundary Value Problem  
  1.1 Introduction 21  
  1.2 A model problem 24  
  1.2.1 Linear elastic string 24  
  1.2.2 Mass-spring model 31  
  1.2.3 Continuum limit 36  
  1.3 Variational approach 45  
  1.3.1 Virtual work equation 45  
  1.3.2 Regularity requirements 54  
  1.3.3 Differential equation 58  
  1.4 Simplified model 63  
  1.5 Discretization 72
# Second-order Scalar Elliptic Boundary Value Problems

## 2.1 Equilibrium models
- 2.1.1 Taut membrane
- 2.1.2 Electrostatic fields
- 2.1.3 Quadratic minimization problems

## 2.2 Sobolev spaces

## 2.3 Variational formulations
- 2.3.1 Linear variational problems
- 2.3.2 Stability

## 2.4 Equilibrium models: Boundary value problems

## 2.5 Diffusion models (Stationary heat conduction)

## 2.6 Boundary conditions

## 2.7 Characteristics of elliptic boundary value problems

## 2.8 Second-order elliptic variational problems

## 2.9 Essential and natural boundary conditions
3 Finite Element Methods (FEM)

3.1 Galerkin discretization

3.2 Case study: Triangular linear FEM in two dimensions

3.2.1 Triangulations

3.2.2 Linear finite element space

3.2.3 Nodal basis functions

3.2.4 Sparse Galerkin matrix

3.2.5 Computation of Galerkin matrix

3.2.6 Computation of right hand side vector

3.3 Building blocks of general FEM

3.3.1 Meshes

3.3.2 Polynomials

3.3.3 Basis functions

3.4 Lagrangian FEM

3.4.1 Simplicial Lagrangian FEM

3.4.2 Tensor-product Lagrangian FEM

3.5 Implementation of FEM

3.5.1 Mesh file format

3.5.2 Mesh data structures [6, Sect. 1.1]

3.5.3 Assembly [6, Sect. 5]

3.5.4 Local computations and quadrature

3.5.5 Incorporation of essential boundary conditions

3.6 Parametric finite elements

3.6.1 Affine equivalence

3.6.2 Example: Quadrilateral Lagrangian finite elements

3.6.3 Transformation techniques

3.6.4 Boundary approximation

3.7 Linearization
# 4 Finite Differences (FD) and Finite Volume Methods (FV)

## 4.1 Finite differences

## 4.2 Finite volume methods (FVM)

### 4.2.1 Gist of FVM

### 4.2.2 Dual meshes

### 4.2.3 Relationship of finite elements and finite volume methods

# 5 Convergence and Accuracy

## 5.1 Galerkin error estimates

## 5.2 Empirical Convergence of FEM

## 5.3 A priori finite element error estimates

### 5.3.1 Estimates for linear interpolation in 1D

### 5.3.2 Error estimates for linear interpolation in 2D

### 5.3.3 The Sobolev scales

### 5.3.4 Anisotropic interpolation error estimates

### 5.3.5 General approximation error estimates

## 5.4 Elliptic regularity theory

## 5.5 Variational crimes

### 5.5.1 Impact of numerical quadrature

### 5.5.2 Approximation of boundary

## 5.6 Duality techniques

### 5.6.1 Linear output functionals

### 5.6.2 Case study: Boundary flux computation

### 5.6.3 $L^2$-estimates

## 5.7 Discrete maximum principle
6 2nd-Order Linear Evolution Problems

6.1 Parabolic initial-boundary value problems

6.1.1 Heat equation
6.1.2 Spatial variational formulation
6.1.3 Method of lines
6.1.4 Timestepping
  6.1.4.1 Single step methods
  6.1.4.2 Stability
6.1.5 Convergence

6.2 Wave equations

6.2.1 Vibrating membrane
6.2.2 Wave propagation
6.2.3 Method of lines
6.2.4 Timestepping
6.2.5 CFL-condition

7 Convection-Diffusion Problems

7.1 Heat conduction in a fluid

7.1.1 Modelling fluid flow
7.1.2 Heat convection and diffusion
7.1.3 Incompressible fluids
7.1.4 Transient heat conduction
7.2 Stationary convection-diffusion problems

7.2.1 Singular perturbation

7.2.2 Upwinding

7.2.2.1 Upwind quadrature

7.2.2.2 Streamline diffusion

7.3 Transient convection-diffusion BVP

7.3.1 Method of lines

7.3.2 Transport equation

7.3.3 Lagrangian split-step method

7.3.3.1 Split-step timestepping

7.3.3.2 Particle method for advection

7.3.3.3 Particle mesh method

7.3.4 Semi-Lagrangian method

8 Numerical Methods for Conservation Laws

8.1 Conservation laws: Examples

8.1.1 Linear advection

8.1.2 Inviscid gas flow

8.2 Scalar conservation laws in 1D

8.2.1 Integral and differential form

8.2.2 Characteristics

8.2.3 Weak solutions
8.2.4 Jump conditions
8.2.5 Riemann problem
8.2.6 Entropy condition
8.2.7 Properties of entropy solutions

8.3 Conservative finite volume discretization
8.3.1 Semi-discrete conservation form
8.3.2 Discrete conservation property
8.3.3 Numerical flux functions
  8.3.3.1 Central flux
  8.3.3.2 Lax-Friedrichs flux
  8.3.3.3 Upwind flux
  8.3.3.4 Godunov flux
8.3.4 Monotone schemes

8.4 Timestepping
8.4.1 CFL-condition
8.4.2 Linear stability
8.4.3 Convergence

8.5 Higher-order conservative schemes
8.5.1 Piecewise linear reconstruction
8.5.2 Slope limiting
8.5.3 MUSCL scheme

8.6 Outlook: systems of conservation laws
# Finite Elements for the Stokes Equations

## 9.1 Viscous fluid flow

## 9.2 The Stokes equations

### 9.2.1 Constrained variational formulation

### 9.2.2 Saddle point problem

### 9.2.3 Stokes system

## 9.3 Saddle point problems: Galerkin discretization

### 9.3.1 Pressure instability

### 9.3.2 Stable Galerkin discretization

### 9.3.3 Convergence

## 9.4 The Taylor-Hood element

---

# Adaptive Finite Element Discretization

## 10.1 Concept of adaptivity

## 10.2 A priori hp-adaptivity

### 10.2.1 Graded meshes in 1D

### 10.2.2 Triangular graded meshes

### 10.2.3 hp-approximation in 1D

### 10.2.4 hp-adapted Lagrangian finite elements

## 10.3 A posteriori mesh adaptation

### 10.3.1 Equidistribution principle

### 10.3.2 Local mesh refinement

### 10.3.3 Refinement control

## 10.4 A posteriori error estimation

### 10.4.1 Hierarchical error estimator

### 10.4.2 Goal oriented error estimation
11 Multilevel iterative solvers

11.1 Solving finite element linear systems

11.2 Subspace correction

11.2.1 Successive subspace correction algorithm (SSC)

11.2.2 Gauss-Seidel iteration

11.2.3 Hierarchical basis multigrid

11.2.3.1 Hierarchical basis in 1D

11.2.3.2 Hierarchical transformations

11.2.3.3 Hierarchical basis in 2D

11.2.3.4 Convergence

11.3 Multigrid method

11.3.1 Multilevel subspace corrections

11.3.2 Multigrid algorithm

11.3.2.1 Transfer operators

11.3.2.2 Local relaxations

11.3.3 Nested iteration

11.4 Algebraic multigrid

11.5 Multilevel preconditioning

12 Sparse Grids Galerkin Methods

12.1 The curse of dimension

12.2 Hierarchical basis

12.3 Sparse grids

12.4 Approximation on sparse grids

12.5 Sparse grids algorithms
Course history

- Summer semester 04, R. Hiptmair (for RW/CSE undergraduates)
- Winter semester 04/05, C. Schwab (for RW/CSE undergraduates)
- Winter semester 05/06, H. Harbrecht (for RW/CSE undergraduates)
- Winter semester 06/07, C. Schwab (for BSc RW/CSE)
- Autumn semester 07, A. Chernov (for BSc RW/CSE)
- Autumn semester 08, C. Schwab (for BSc RW/CSE)
- Autumn semester 09, V. Gradinaru (for BSc RW/CSE, Subversion Revision: 22844)
- Spring semester 10, R. Hiptmair (for BSc Computer Science)
- Autumn semester 10, R. Hiptmair (for BSc RW/CSE, Subversion Revision: 30025)
Preamble

This lecture is a core course for

- BSc in Computational Science and Engineering (RW/CSE),
- BSc in Computer Science with focus Computational Science.

Main skills to be acquired in this course:

- Ability to *implement* advanced numerical methods for the solution of partial differential equations in MATLAB efficiently
- Ability to *modify* and *adapt* numerical algorithms guided by awareness of their mathematical foundations
- Ability to *select* and *assess* numerical methods in light of the predictions of theory
- Ability to *identify features* of a PDE (= partial differential equation) based model that are relevant for the selection and performance of a numerical algorithm
Ability to understand research publications on theoretical and practical aspects of numerical methods for partial differential equations.

This course ≠ Numerical analysis of PDE (→ mathematics curriculum)

(401-3651-00V Numerical methods for elliptic and parabolic partial differential equations, )

Instruction on how to apply software packages

Reading instructions

This course materials are neither a textbook nor lecture notes. They are meant to be supplemented by explanations given in class.

Some pieces of advice:

- these lecture slides are not designed to be self-contained, but to supplement explanations in class.
this document is not meant for mere reading, but for working with,

turn pages all the time and follow the numerous cross-references,

study the relevant section of the course material when doing homework problems.

What to expect

- The course is difficult and demanding (i.e. ETH level)

- Do not expect to understand everything in class. The average student will
  - understand about one third of the material when attending the lectures,
  - understand another third when making a serious effort to solve the homework problems,
  - hopefully understand the remaining third when studying for the examination after the end of the course.

- Perseverance will be rewarded!
Practical information

Course: 401-3663-00L Numerical Methods for Partial Differential Equations

Lectures: Mon 13-15 HG D 5.2
Wed 10-12 HG D 5.2

Tutorials: Mon 08-10 HG D 5.2
Mon 10-12 HG D 5.2

Lecturer: Prof. Ralf Hiptmair, office: HG G 58.2, e-mail: hiptmair@sam.math.ethz.ch

Assistants:
Eivind Fonn, office: HG J 59, e-mail: eivind.fonn@sam.math.ethz.ch
(2nd year PhD student at Seminar of Applied Mathematics, D-MATH)

Dr. Erhan Turan, office: CAB F 81, e-mail: erhan.turan@inf.ethz.ch
(Postdoctoral researcher at Chair of Computational Science, D-INFK)
Assignments:
- 12 weekly assignment sheets, handed out on Monday, discussed in tutorial classes in the following week.
- “Testat” requirement:
  - Take part in the tutorial classes. The attendance is mandatory: at most two tutorial classes can be skipped during the semester.
  - Take active part during the tutorial. Every student will be asked to present the solution of a few exercises at the blackboard.
  - Work extensively on the exercises and hand in the approaches/solutions. In order to obtain the access to the examination, about 33% of the exercises proposed during the semester should be solved and handed in.

Examination: “Sessionsprüfung”: computer based examination, programming and theoretical tasks

3 hour examination on **Friday, January 28, 2011**

Reporting errors

Please report errors in the electronic lecture notes via a wiki page!


(Password: NPDE, please choose EDIT menu to enter information)

Please supply the following information:

- (sub)section where the error has been found,
- precise location (e.g., after Equation (4), Thm. 2.3.3, etc.). Refrain from giving page numbers,
- brief description of the error.

Online discussion forum

Contribute to the forum
This forum has been set up so that you can post questions on the programming exercises that accompany the course. One of the assistants will look at the entries in this forum and

- write an answer in this forum or
- discuss the question in a consulting session and post an answer later.

A second purpose of this forum is that the assistants can collect FAQs and post answers here.
This course is designed to be rather self-contained and additional study of literature should not be crucial to follow the lecture. These references point to further sources of information, mainly devoted mathematical theory.

Case Study: A Two-point Boundary Value Problem

This chapter offers a brief tour of

- mathematical modelling of a physical system based on variational principles,
- the derivation of differential equations from these variational principles,
- the discretization of the variational problems and/or of the differential equations using various approaches.
1.1 Introduction

The term “partial differential equation” (PDE) usually conjures up formulas like

$$\text{div}(\sqrt{1 + \|\text{grad } u(x)\|^2} \text{grad } u(x)) + v \cdot \text{grad } u = f(x), \quad x \in \Omega \subset \mathbb{R}^d.$$ 

This chapter aims to wean you off this impulse and instil an appreciation that

```
<table>
<thead>
<tr>
<th>a meaningful PDE encodes structural principles</th>
</tr>
</thead>
<tbody>
<tr>
<td>(like equilibrium, conservation, etc.)</td>
</tr>
</tbody>
</table>
```

The design and selection of numerical methods has to take into account these governing principles.
Remark 1.1.1 (Mathematical modelling).

Prerequisite for numerical simulation: Mathematical modelling

Physics  Biology  ...  Economics

Appropriate & necessary simplifications

(PDE based) mathematical model

Necessary simplification:

\[
\left\{ \begin{array}{l}
\text{system} \\
\text{phenomenon}
\end{array} \right\}
\text{described by a few variables/functions in configuration space}

The art of modelling: devise “faithful model”

Essential/relevant traits of \[
\left\{ \begin{array}{l}
\text{system} \\
\text{phenomenon}
\end{array} \right\}
\] \rightarrow \text{structural properties of model}
Remark 1.1.2 ("PDEs" for univariate functions).

The classical concept of a PDE inherently involves functions of several independent variables. However, when one embraces the concept of a PDE as encoding fundamental structural properties of a model, then simple representatives in a univariate setting can be discussed.


ordinary differential equations (ODEs) offer simple specimens of important classes of PDEs!

Thus, in this chapter we examine ODEs that are related to the important class of elliptic PDEs.

Note: The developments below cannot live up to standards of mathematical rigour, because what has deliberately omitted is the discussion of the functional analytic framework (function space theory) required for a complete statement of, for instance, minimization problems and variational problems.
1.2 A model problem

1.2.1 Linear elastic string

Static mechanical problem:
Deformation of elastic “1D” string (rubber band) under its own weight
Constraint: string pinned at endpoints

Fig. 1
Sought: (Approximation of) “shape” of elastic string
Note: description of a curve in the plane by a mapping $[0, 1] \mapsto \mathbb{R}^2$ requires coordinate system. Of course, the choice of the coordinate system must not affect the shape obtained from the mathematical model, a property called frame indifference.

Current concept of force field $\mathbf{f}$: force "pulls" at elastic string. Alternative: force due to a potential. Gravitational force (i.e., a constant force field) allows both descriptions.
Remark 1.2.2 (Parametrization of a curve). → [29, Sect. 7.4]

We consider a curve in $\mathbb{R}^2$ $u : [0, 1] \mapsto \mathbb{R}^2$

$$u \in (C^0([0, 1]))^2$$

connected curve

Geometric intuition: $u(\xi)$ moves along the curve as $\xi$ increases from 0 to 1.

notation: $C^k([a, b]) \doteq k$-times continuously differentiable functions on $[a, b] \subset \mathbb{R}$, see [29, Sect. 5.4] $$(C^k([a, b]))^2 \doteq k$-times continuously differentiable curves $u : [a, b] \mapsto \mathbb{R}^2$, that is, if $u = (u_1, u_2)$, then $u_1, u_2 \in C^k([a, b])$. 

Terminology: $[0, 1] \doteq$ parameter domain, notation $\Omega$
Interpretation of curve parameter $\xi$: “virtual time”

\[ \| u' \| \doteq \text{“speed” with which curve is traversed} \]

\[ \int_0^1 \| u'(\xi) \| \, d\xi \doteq \text{length of curve} \]

notation: $\| \cdot \| \doteq \text{Euclidean norm of a vector } \in \mathbb{R}^n$

parametrization is supposed to be \emph{locally injective}:

\[ \forall \xi \in ]0, 1[ : \exists \epsilon > 0 : \forall \eta, |\eta - \xi| < \epsilon : u(\eta) \neq u(\xi) . \]

For $u \in (C^1([0, 1]))^2$ we expect $u'(\xi) \neq 0$ for all $0 \leq \xi \leq 1$

notation: $' \doteq \text{derivative w.r.t. curve parameter, here } \xi$
**Remark 1.2.3** (Material coordinate).

Interpretation of curve parameter $\xi$:

$\xi$: unique identifier for each infinitesimal section of the string, a *label* for each “material point” on the string

$\xi \hat{=} \text{material coordinate, unrelated to “position in space” (}= \text{physical coordinate)},$

$\xi$ has no physical dimension $\Rightarrow$ ‘ does not affect dimension.

**Remark 1.2.4** (Non-dimensional equations).

By fixing reference values for the basic physical units occurring in a model (“scaling”), one can switch to a non-dimensional form of the model equations.

In the case of the elastic string model the basic units are
• unit of length $1\text{m}$,
• unit of force $1\text{N}$.

Thus, non-dimensional equations arise from fixing a reference length $\ell_0$ and a reference force $f_0$.

Below, following a (bad) habit of mathematicians, physical units will be routinely dropped, which tacitly assumes a priori scaling.

Quantities that have to be specified to allow the unique determination of a configuration in a mathematical model are called problem data/parameters. In the elastic string model the problem parameters are

- the boundary conditions (1.2.1),
- the force field $f : [0, 1] \mapsto \mathbb{R}^2, [f] = 1\text{N}$, $f(\xi) \overset{\Delta}{=} \text{force “pulling at” a material point } \xi$.

Special case: gravitational force $f(\xi) := -g\rho(\xi)(0)\bigg|_1$, $0 \leq \xi \leq 1$, $g = 9.81\text{m s}^{-2}$ with density $\rho : [0, 1] \mapsto \mathbb{R}^+, [\rho] = \text{kg}$,
1.2.2 Mass-spring model

Idea: model string as a system of many simple components that interact in simple ways

\[\begin{align*}
    m_1 & \leftrightarrow R \\
    m_2 & \leftrightarrow R \\
    m_3 & \leftrightarrow R \\
    m_4 & \leftrightarrow R \\
    m_5 & \leftrightarrow R \\
    m_6 & \leftrightarrow R \\
    m_7 & \leftrightarrow R \\
    m_8 & \leftrightarrow R
\end{align*}\]

Assumption: linear springs \(\leftrightarrow\) Hooke's law
Force

\[
F(l) = \kappa \left( \frac{l}{l_0} - 1 \right)
\]  
(relative elongation) .  

(1.2.5)

\(\kappa \triangleq \) spring constant (stiffness), \([\kappa] = 1\text{N}, \kappa > 0\),  
\(l_0 \triangleq \) equilibrium length of (relaxed) spring.

\begin{itemize}
  \item elastic energy stored in linear spring at length \(l > 0\)
  \[
  E_{\text{el}} = \int_{l_0}^{l} F(\tau) \, d\tau = \frac{1}{2} \frac{\kappa}{l_0} (l - l_0)^2 , \quad [E_{\text{el}}] = 1\text{J} . 
  \]  
  (1.2.6)
\end{itemize}

Configuration space for mass-spring model:

\[
\mathbf{u}^i \in \mathbb{R}^2 \triangleq \text{position of } i\text{-th mass point, } i = 1, \ldots, n
\]

\[
\triangleright \quad \text{finite-dimensional configuration space } \ = \ (\mathbb{R}^2)^n
\]
Models, for which configurations can be described by means of finitely many real numbers are called \textit{discrete}. Hence, the mass-spring model is a \textit{discrete model}, see Sect. 1.5.

Total elastic energy of mass-spring model in configuration \((u^1, \ldots, u^n) \in (\mathbb{R}^2)^n\):

\[
(1.2.6) \quad \Rightarrow \quad J^{(n)}_{\text{el}} = J^{(n)}_{\text{el}}(u^1, \ldots, u^n) := \frac{1}{2} \sum_{i=0}^{n} \frac{\kappa_i}{l_i} \left( \| u^{i+1} - u^i \| - l_i \right)^2 ,
\]

where \(u^0 := (a_u, a_w)\), \(u^{n+1} := (b_u, b_w)\) (pinning positions (1.2.1)),
\(\kappa_i \hat{=} \text{spring constant of } i\)-th spring, \(i = 0, \ldots, n\),
\(l_i > 0 \hat{=} \text{equilibrium length of } i\)-th spring.

Total “gravitational” energy of mass-spring model in configuration \((u^1, \ldots, u^n)\) due to external force field:

\[
J^{(n)}_f = J^{(n)}_f(u^1, \ldots, u^n) := - \sum_{i=1}^{n} f^i \cdot u^i ,
\]

where \(f^i \hat{=} \text{force acting on } i\)-th mass, \(i = 1, \ldots, n\).

\(R.\, Hiptmair\)
\(C.\, Schwab,\)
\(H.\, Harbrecht\)
\(V.\, Gradinaru\)
\(A.\, Chernov\)
SAM, ETHZ

1.2 Numerical Methods for PDEs
notation: \( \mathbf{u} \cdot \mathbf{v} := \mathbf{u}^T \mathbf{v} = \sum_{j=1}^{n} u_j v_j \) is the inner product of vectors in \( \mathbb{R}^n \).

Known from classical mechanics, static case: equilibrium principle

systems attains configuration(s) of minimal (potential) energy

\[
J^{(n)} := J_{\text{el}}^{(n)} + J_{\text{f}}^{(n)}
\]

equilibrium configuration \( \mathbf{u}_1^*, \ldots, \mathbf{u}_n^* \) of mass-spring system solves

\[
(\mathbf{u}_1^*, \ldots, \mathbf{u}_n^*) = \arg\min_{(\mathbf{u}_1, \ldots, \mathbf{u}_n) \in \mathbb{R}^{2n}} J^{(n)}(\mathbf{u}_1, \ldots, \mathbf{u}_n).
\] (1.2.9)
Plot of $J^{(1)}(u^1)$

Mass-spring system with only one point mass
(non-dimensional $l_1 = l_2 = 1$, $\kappa_1 = \kappa_2 = 1$,
$u^0 = (0, 0)$, $u^2 = (1, 0, 2)$, $f^1 = (0, 0)$)

Note: solutions of (1.2.9) need not be unique!

To see this, consider the case $L := \sum_{i=0}^{n} l_i > \|u^{n+1} - u^0\|$ and $f \equiv 0$ (slack ensemble of springs without external forcing). In this situation many crooked arrangements of the masses will have zero total potential energy.
1.2.3 Continuum limit

Heuristics: elastic string = spring-mass system with “infinitely many infinitesimal masses” and “infinitesimally short” springs.

Policy: consider sequence \((SMM_n)_{n \in \mathbb{N}}\) of spring-mass systems with \(n\) masses, identify material coordinate \((\rightarrow \text{Rem. 1.2.3})\) of point masses, choose system parameters with meaningful limits, derive expressions for energies as \(n \rightarrow \infty\), use them to define the “continuous elastic string model”.

\(<\) minimal energy configuration of a mass spring system for variable \(L\).

\((n = 10, \text{non-dimensional } \kappa_i = 1, l_i = L/n, i = 1, \ldots, 10)\)
Assumption: equal equilibrium lengths of all springs \( l_i = \frac{L}{n+1}, \) \( L > 0, \)
\( L \triangleq \) equilibrium length of elastic string: \( L = \sum_i l_i, [L] = 1m. \)

Equilibrium configuration of mass-spring system

(Non-dimensional \( l_i = \frac{L}{n+1}, \kappa_i = 1, m_i = \frac{1}{n}, f_i = \frac{1}{n}(\begin{array}{c} 0 \\ -1 \end{array}), L = 1, n \) varying)
In the spring-mass model each spring has its own stiffness $\kappa_i$ and every mass point its own force $f^i$ acting on it. When considering the “limit” of a sequence of spring-mass models, we have to detach stiffness and force from springs and masses and attach them to material points, cf. Rem. 1.2.3. In other words stiffness $\kappa_i$ and force $f^i$ have to be induced by a stiffness function $\kappa(\xi)$ and force function $f(\xi)$. This linkage has to be done in a way to allow for a meaningful limit $n \to \infty$ for the potential energy.
“Limit-compatible” system parameters: \( \xi_{i+1/2} := \frac{1}{2}(\xi_{i+1}^{(n)} + \xi_{i}^{(n)}) \)

- \( \kappa_i = \kappa(\xi_{i+1/2}^{(n)}) \) with integrable stiffness function \( \kappa : [0, 1] \mapsto \mathbb{R}^+ \),

- \( f^i = \int_{\xi_{i-1/2}^{(n)}}^{\xi_{i+1/2}^{(n)}} f(\xi) \, d\xi \) “lumped force”, integrable force field \( f : [0, 1] \mapsto \mathbb{R}^2 \)

energies, see\(^{(1.2.7), (1.2.8)}\)

\[
J_{el}^{(n)}(u) = \frac{1}{2} \sum_{i=0}^{n} \frac{n + 1}{L} \kappa(\xi_{i+1/2}^{(n)}) \left( \left\| u(\xi_{i+1}^{(n)}) - u(\xi_{i}^{(n)}) \right\| - \frac{L}{n + 1} \right)^2 ,
\]

\[
J_f^{(n)}(u) = - \sum_{i=1}^{n} \int_{\xi_{i-1/2}^{(n)}}^{\xi_{i+1/2}^{(n)}} f(\xi) \, d\xi \cdot u(\xi_{i}^{(n)}) .
\]

1.2 Numerical Methods for PDEs

R. Hiptmair
C. Schwab,
H. Harbrecht
V. Gradinaru
A. Chernov
SAM, ETHZ

p. 39
Assumption: \( u \in (C^2([0,1]))^2 \) (twice continuously differentiable)

1. Simple limit for potential energy due to external force:

\[
J_f(u) = \lim_{n \to \infty} J_f^{(n)}(u) = \lim_{n \to \infty} \sum_{i=1}^{n} \frac{\xi_{i+1/2}^{(n)} - \xi_{i-1/2}^{(n)}}{2} \int_{\xi_{i-1/2}^{(n)}}^{\xi_{i+1/2}^{(n)}} f(\xi) \, d\xi \cdot u(\xi^{(n)}_i) = - \int_0^1 f(\xi) \cdot u(\xi) \, d\xi .
\]  \hspace{1cm} (1.2.12)

2. Limit of elastic energy:
Tool: Taylor expansion: for \( \mathbf{u} = \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} \in C^2 \) with derivative \( \mathbf{u}' \), \( 1 \gg \eta \to 0 \)

\[
\|\mathbf{u}(\xi + \eta) - \mathbf{u}(\xi - \eta)\| = \sqrt{(u_1(\xi + \eta) - u_1(\xi - \eta))^2 + (u_2(\xi + \eta) - u_2(\xi - \eta))^2}
\]

\[
= \sqrt{(2u'_1(\xi)\eta + O(\eta^3))^2 + (2u'_2(\xi)\eta + O(\eta^3))^2}
\]

\[
= 2\eta \| \mathbf{u}(\xi)' \| \sqrt{1 + O(\eta^2)} = 2\eta \| \mathbf{u}'(\xi) \| + O(\eta^2).
\] (1.2.13)

Apply this to (1.2.10) with \( \eta = \frac{1}{2n+1} \) for \( n \to \infty \)

\[
J^{(n)}_{el}(\mathbf{u}) = \frac{1}{2} \sum_{i=0}^{n} \frac{n+1}{L} \kappa(\xi_{i+1}/2) \left( \frac{1}{n+1} \| \mathbf{u}'(\xi_{i+1}/2) \| + O\left( \frac{1}{(n+1)^2} \right) - \frac{L}{n+1} \right)^2
\]

\[
= \frac{1}{2Ln+1} \sum_{i=0}^{n} \kappa(\xi_{i+1}/2) \left( \| \mathbf{u}'(\xi_{i+1}/2) \| + O\left( \frac{1}{n+1} \right) - L \right)^2.
\] (1.2.14)

Consideration: integral as limit of Riemann sums, see [29, Sect. 6.2]:

\[
q \in C^0([0, 1]): \lim_{n \to \infty} \frac{1}{n+1} \sum_{j=0}^{n} q\left( \frac{j+1/2}{n+1} \right) = \int_0^1 q(\xi) \, d\xi.
\] (1.2.15)
\[ J_{el}(u) = \lim_{n \to \infty} J_{el}^{(n)}(u) = \frac{1}{2L} \int_{0}^{1} \kappa(\xi) \left( \|u'(\xi)\| - L \right)^2 \, d\xi. \] (1.2.16)

Equilibrium condition for limit model (minimal total potential energy):

\[
\begin{align*}
\mathbf{u}_* &= \text{argmin}_{u \in (C^1([0,1]))^2} \int_{0}^{1} \frac{\kappa(\xi)}{2L} \left( \|u'(\xi)\| - L \right)^2 - f(\xi) \cdot u(\xi) \, d\xi \\
&= J(u)
\end{align*}
\] (1.2.17)

This is the total potential energy functional, \([J] = 1J\) = a minimization problem in a function space!

**Example 1.2.18** (Tense string without external forcing).
Setting:
- no external force: \( f \equiv 0 \)
- homogeneous string: \( \kappa = \kappa_0 = \text{const} \)
- tense string: \( L < \| u(0) - u(1) \| \) (\( \geq \) positive elastic energy)

(1.2.17) \( \Leftrightarrow \) \( u_* = \arg\min_{u \in \mathcal{C}^1([0,1])} \frac{\kappa_0}{2L} \int_0^1 \left( \| u'(\xi) \| - L \right)^2 \, d\xi \) .  \( (1.2.19) \)

Note: in (1.2.19) \( u \) enters \( J \) only through \( u' \)!

Constraint on \( u' \): by triangle inequality for integrals, see [29, Sect. 6.3]

\[ \ell := \| u(1) - u(0) \| = \left\| \int_0^1 u'(\xi) \, d\xi \right\| \leq \int_0^1 \| u'(\xi) \| \, d\xi . \]  \( (1.2.20) \)
Consider related minimization problem

\[
\begin{align*}
 w_\ast &= \operatorname*{arg\,min}_w \left\{ \frac{\kappa_0}{2L} \int_0^1 (w - L)^2 \, d\xi : \quad w \in (C^0([0, 1]))^2 , \quad \int_0^1 w(\xi) \, d\xi \geq \ell \right\} .
\end{align*}
\]

(1.2.21)

\[
\Rightarrow \quad \text{unique solution} \quad w_\ast(\xi) = \ell \quad (\text{constant solution})
\]

\[ \|u'(\xi)\| = \ell \] and the boundary conditions (1.2.1) are satisfied for the straight line solution of (1.2.19)

\[
\begin{align*}
 u_\ast(\xi) &= (1 - \xi) u(0) + \xi u(1) .
\end{align*}
\]

It is exactly the “straight string” solution that physical intuition suggests.
1.3 Variational approach

We face the task of minimizing a functional over an $\infty$-dimensional function space. In this section necessary conditions for the minimizer will formally be derived in the form of variational equations. This idea is one of the cornerstone of a branch of analysis called calculus of variations.

We will not dip into this theory, but perform manipulations at a formal level. Yet, all considerations below can be justified rigorously.

1.3.1 Virtual work equation
notation: \( C^k_0([0, 1]) := \{ v \in C^k([0, 1]): v(0) = v(1) = 0 \}, \ k \in \mathbb{N}_0 \)

Main “idea of calculus of variations”:

\[
\begin{align*}
& u_* \text{ solves } (1.2.17) \quad \Rightarrow \quad J(u_*) \leq J(u_* + tv) \quad \forall t \in \mathbb{R}, \ v \in (C^2_0([0, 1]))^2. \quad (1.3.1) \\
& \varphi(t) := J(u_* + tv) \text{ has global minimum for } t = 0 \\
& \text{If } \varphi \text{ differentiable, then } \frac{d\varphi}{dt}(0) = 0
\end{align*}
\]

Note: \( v(0) = v(1) = 0 \), because we must not tamper with the pinning conditions (1.2.1).

Rule: Variation \( v \) must vanish where argument function \( u \) is fixed.

Computation of \( \frac{d\varphi}{dt}(0) \) for \( J \) from (1.2.17) amounts to computing a “configurational derivative” in direction \( v \).
We pursue a separate treatment of energy contributions (This also demonstrates a simple formal approach to computing configurational derivatives.):

1. Potential energy \((1.2.12)\) due to external force:

\[
\lim_{t \to 0} \frac{J_f(u_* + t\mathbf{v}) - J_f(u_*)}{t} = - \lim_{t \to 0} \frac{1}{t} \int_0^1 f(\xi) \cdot t\mathbf{v}(\xi) \, d\xi = - \int_0^1 f(\xi) \cdot \mathbf{v}(\xi) \, d\xi. \tag{1.3.2}
\]

2. Elastic energy \((1.2.16)\): more difficult, tool: Taylor expansion \([29, \text{Sect. 5.5}]\)

Analogous to \((1.2.13)\), \(\mathbf{x} \in \mathbb{R}^2 \setminus \{0\}, \mathbf{h} \in \mathbb{R}^2\), for \(\mathbb{R} \ni t \to 0\)

\[
\|\mathbf{x} + t\mathbf{h}\| = \sqrt{(x_1 + th_1)^2 + (x_2 + th_2)^2} = \sqrt{\|\mathbf{x}\|^2 + 2t\mathbf{x} \cdot \mathbf{h} + t^2 \|\mathbf{h}\|^2}
\]

\[
= \|\mathbf{x}\| \sqrt{1 + 2t \frac{\mathbf{x} \cdot \mathbf{h}}{\|\mathbf{x}\|^2} + t^2 \|\mathbf{h}\|^2} = \|\mathbf{x}\| + t \frac{\mathbf{x} \cdot \mathbf{h}}{\|\mathbf{x}\|} + O(t^2), \tag{1.3.3}
\]

where we used the truncated Taylor series for \(\sqrt{1+x}\)

\[
\sqrt{1 + \delta} = 1 + \frac{1}{2} \delta + O(\delta^2) \quad \text{for} \quad \delta \to 0. \tag{1.3.4}
\]

R. Hiptmair
C. Schwab,
H. Harbrecht
V. Gradinaru
A. Chernov
SAM, ETHZ

1.3
p. 47
Use (1.3.3) in the perturbation analysis for the elastic energy:

\[
(\|u'(\xi) + tv'(\xi)\| - L)^2 = \left( \|u'(\xi)\| + t \frac{u'(\xi) \cdot v'(\xi)}{\|u'(\xi)\|} + O(t^2) - L \right)^2
\]

\[
= (\|u'(\xi)\| - L)^2 + 2t(\|u'(\xi)\| - L) \frac{u'(\xi) \cdot v'(\xi)}{\|u'(\xi)\|} + O(t^2) .
\]

\[
J_{el}(u + tv) - J_{el}(u) = \frac{t}{L} \int_0^1 \kappa(\xi) (\|u'(\xi)\| - L) \frac{u'(\xi) \cdot v'(\xi)}{\|u'(\xi)\|} + O(t^2) \, d\xi . \quad (1.3.5)
\]

\[
\lim_{t \to 0} \frac{J_{el}(u_* + tv) - J_{el}(u_*)}{t} = \int_0^1 \frac{\kappa(\xi)}{L} (\|u'(\xi)\| - L) \frac{u'(\xi) \cdot v'(\xi)}{\|u'(\xi)\|} \, d\xi . \quad (1.3.6)
\]

Here we take for granted \( \|u'(\xi)\| \neq 0 \), which is an essential property of a meaningful parameterization of the elastic string, see Rem. 1.2.2.
Necessary condition for \( u_\ast \) solving (1.2.17)

\[
\int_0^1 \frac{\kappa(\xi)}{L} \left( \|u_\ast'(\xi)\| - L \right) \frac{u_\ast'(\xi) \cdot v'(\xi)}{\|u_\ast'(\xi)\|} - f(\xi) \cdot v(\xi) \, d\xi = 0 \quad \forall v \in (C^2_0([0, 1]))^2.
\] (1.3.7)

This is a non-linear variational equation on domain \( \Omega = [0, 1] \)

**Remark 1.3.8 (Differentiating a functional on a space of curves).**

For a \( C^2 \)-function \( F : \mathbb{R}^d \times \mathbb{R}^d \mapsto \mathbb{R}, \ d \in \mathbb{N} \), consider the functional

\[
J : (C^1_{pw}([0, 1]))^d \mapsto \mathbb{R} , \quad J(u) := \int_0^1 F(u'(\xi), u(\xi)) \, d\xi.
\]

Use the multidimensional Taylor’s formula \([29, \text{Satz 7.5.2}]\)

\[
F(u + \delta u, v + \delta v) = F(u, v) + D_1 F(u, v) \delta u + D_2 F(u, v) \delta v + O(\|\delta u\|^2 + \|\delta v\|^2) .
\] (1.3.9)

Here, \( D_1 F \) and \( D_2 F \) are the partial derivatives of \( F \) w.r.t the first and second vector argument, respectively. These are row vectors.

\[
J(u + tv) = J(u) + t \int_0^1 D_1 F(u'(\xi), u(\xi)) v'(\xi) + D_2 F(u'(\xi), u(\xi)) v(\xi) \, d\xi + O(t^2) .
\] (1.3.10)

“directional derivative” \( (D_u J)(u)(v) \)
The derivatives $u', v'$ are just regular 1D derivatives w.r.t. the parameter $\xi$. They yield column vectors. Hence, we deal with a scalar integrand.

**Remark 1.3.11 (Virtual work principle).**

In statics, the derivation of variational equations from energy minimization (equilibrium principle, see (1.2.9)) is known as the method of **virtual work**: Small admissible changes of the equilibrium configuration of the system invariably entail active work.

**Remark 1.3.12 (Non-linear variational equation).**

Now we unravel the structure behind the non-linear variational problem (1.3.7).

Recall essential terminology from linear algebra:
Definition 1.3.13 ((Bi-)linear forms).

Given an \( \mathbb{R} \)-vector space \( V \), a **linear form (linear functional)** \( \ell \) is a mapping \( f : V \mapsto \mathbb{R} \) that satisfies

\[
\ell(\alpha u + \beta v) = \alpha \ell(u) + \beta \ell(v) \quad \forall u, v \in V, \forall \alpha, \beta \in \mathbb{R}.
\]

A **bilinear form** \( a \) on \( V \) is a mapping \( a : V \times V \mapsto \mathbb{R} \), for which

\[
a(\alpha_1 v_1 + \beta_1 u_1, \alpha_2 v_2 + \beta_2 u_2) = \\
= \alpha_1 \alpha_2 a(v_1, v_2) + \alpha_1 \beta_2 a(v_1, u_2) + \beta_1 \alpha_2 a(u_1, v_2) + \beta_1 \beta_2 a(u_1, u_2)
\]

for all \( u_i, v_i \in V, \alpha_i, \beta_i \in \mathbb{R}, i = 1, 2 \).

\( \Rightarrow \) notation: \( a, b, \ldots \) \( \hat{=} \) bilinear forms

In the case of (1.3.7) we make a very important observation, namely that, keeping \( u_* \) fixed, the left hand side is a **linear functional** (linear form) in the **test function** \( v \):
Structure of

\[
\int_0^1 \frac{\kappa(\xi)}{L} \left( \frac{u'_*(\xi)}{\|u'_*(\xi)\|} - L \right) \frac{u'_*(\xi) \cdot v'_*(\xi)}{\|u'_*(\xi)\|} - f(\xi) \cdot v(\xi) \, d\xi = 0 \quad \forall v \in (C^2_0([0, 1]))^2:
\]

▷ abstract non-linear variational equation

\[
\begin{align*}
\forall v \in V_0 \hat{=}
\end{align*}
\]

\[
V_0 \hat{=} \text{(real) vector space of functions},
\]

\[
V \hat{=} \text{affine space of functions: } V = u_0 + V_0, \text{ with offset function } u_0 \in V,
\]

\[
\ell \hat{=} \text{a linear mapping } V_0 \mapsto \mathbb{R}, \text{ a linear form},
\]

\[
a \hat{=} \text{a mapping } V \times V_0 \mapsto \mathbb{R}, \text{ linear in the second argument}, \text{ that is}
\]

\[
a(u; \alpha v + \beta w) = \alpha a(u; v) + \beta a(u; w) \quad \forall u \in V, v, w \in V_0, \alpha, \beta \in \mathbb{R}.
\]

Terminology related to variational problem \((1.3.14)\):

\[
V \text{ is called trial space}
\]

\[
V_0 \text{ is called test space}
\]
Explanation of terminology:

- **trial space** $\hat{V}$ = the function space in which we seek the solution
- **test space** $\hat{V}$ = the space of eligible test functions $v$ in a variational problem like (1.3.14)

The two spaces need not be the same: $V \neq V_0$ is common and already indicated by the notation. For many variational problem, which are not examined in this course, they may even comprise functions with different smoothness properties.

In concrete terms (for elastic string continuum model):

- $V_0 := (C^2_0([0, 1]))^2$,
- $V := \{ u \in (C^2([0, 1]))^2 : u(0) = \begin{pmatrix} a \\ u_a \end{pmatrix}, u(1) = \begin{pmatrix} b \\ u_b \end{pmatrix} \}$
  
  $= [\xi \mapsto (1 - \xi)u(0) + \xi u(1)] + V_0$
  $=: u_0$ (1.3.16)

- $\ell(v) := \int_0^1 f(\xi) \cdot v(\xi) \, d\xi$, (1.3.17)
Thus, for variational problem (1.3.7) arising from the elastic string model we find the common pattern $V = V_0 + u_0$, that is, the trial space is an **affine space**, arising from the test space $V_0$ by adding an offset function $u_0$.

If $V = V_0 + u_0$, then there is a way to recast (1.3.14) as a variational problem with the same trial and test space $V_0$:

Rewriting (1.3.14) using the offset function $u_0 \in V$:

\[
(1.3.14) \quad \Rightarrow \quad w \in V_0: \quad a(u_0 + w; v) = \ell(v) \quad \forall v \in V_0 \quad \text{and} \quad u = u_0 + w. \tag{1.3.19}
\]

### 1.3.2 Regularity requirements
Issue: The derivation of the continuum models (1.2.17) (→ Sect. 1.2.3) and (1.3.7) was based on the assumption \( u \in (C^2([0, 1]))^2 \).

Is \( u \in (C^2([0, 1]))^2 \) required to render the minimization problem (1.2.17)/variational problem 1.2.3) meaningful?

We will find that curves with less smoothness can still yield relevant solutions of (1.2.17)/(1.3.7).

Obvious (→ Rem. 1.2.2):
\[
\begin{align*}
\mathbf{u} = \text{argmin}_{\mathbf{u} \in (C^0([0, 1]))^2} & \\
\left[ \frac{1}{2L} \int_{0}^{1} \frac{\kappa(\xi)}{2L} (\| \mathbf{u}'(\xi) \| - L)^2 - \mathbf{f}(\xi) \cdot \mathbf{u}(\xi) \, d\xi \right].
\end{align*}
\]

Observe: \( J(\mathbf{u}) \) from (1.2.17), \( \mathbf{a}, \ell \) from (1.3.18) well defined for merely \textit{continuous, piecewise continuously differentiable} functions \( \mathbf{u}, \mathbf{v} : [0, 1] \mapsto \mathbb{R}^2 \),

\( \mathbf{u}' \) will be piecewise continuous and can be integrated.

\( \text{mere integrability of } \kappa, \mathbf{f} \text{ sufficient.} \)
notation: \( C^k_{pw}([a, b]) \) \(\cong\) globally \( C^{k-1} \) and piecewise \( k \)-times continuously differentiable functions on \([a, b] \subset \mathbb{R}: \) for each \( v \in C^k_{pw}([a, b]) \) there is a finite partition \( \{a = \tau_0 < \tau_1 < \cdots < \tau_m = b\} \) such that \( v|_{\tau_{i-1}, \tau_i} \) can be extended to a function \( \in C^k([\tau_{i-1}, \tau_i]) \). \( C^0_{pw}([a, b]) \) \(\cong\) piecewise continuous functions with only a finite number of discontinuities.

**Example 1.3.20** (Non-smooth external forcing).

Setting: \( \kappa = \text{const} \) (homogeneous string)
\[ u_2/f(\xi) \]

\[ \bullet (a, u_a) \]

force

\[ u_1/\xi \]

\[ b/1 \]

\[ a/0 \]

\[ u_2/f(\xi) \]

\[ \bullet (b, u_b) \]

\[ u_1/\xi \]

\[ b/1 \]

\[ a/0 \]

\[ \checkmark \text{ discontinuous } f \]

\[ \checkmark \text{ point force } f(x) = \delta(\xi - \xi_0)_{1}^{0} \]

\[ u_\ast \not\in (C^2([0, 1]))^2 \text{ physically meaningful:} \]

\[ u_\ast \in (C^1([0, 1]))^2 \text{ for discontinuous } f \]

merely \[ u_\ast \in (C^0([0, 1]))^2 \text{ for point force concentrated in } \xi_0 \]: \text{kink at } \xi_0!
1.3.3 Differential equation

Consider non-linear variational equation (1.3.7):

\[
\int_0^1 \frac{\kappa(\xi)}{L} \left( \|u'(\xi)\| - L \right) \frac{u'(\xi) \cdot v'(\xi)}{\|u'(\xi)\|} - f(\xi) \cdot v(\xi) \, d\xi = 0 \quad \forall v \in (C_{0,pw}^1([0, 1]))^2 .
\]  

Assumption: \( u \in (C^2([0, 1]))^2 \) \& \( \kappa \in C^1([0, 1]) \) \& \( f \in (C^0([0, 1]))^2 \)  

Recall: integration by parts formula [29, Satz 6.1.2]:

\[
\int_0^1 u(\xi)v'(\xi) \, d\xi = - \int_0^1 u'(\xi)v(\xi) + (u(1)v(1) - u(0)v(0)) \quad \forall u, v \in C_{pw}^1([0, 1]) .
\]
Apply to elastic energy contribution in (1.3.7):

\[
\int_0^1 \left( \frac{\kappa(\xi)}{L} \left( \|u'(\xi)\| - L \frac{u'(\xi)}{\|u'(\xi)\|} \right) \right) \cdot v'(\xi) - f(\xi) \cdot v(\xi) \, d\xi
\]

\[
= \int_0^1 \left\{ -\frac{d}{d\xi} \left( \frac{\kappa(\xi)}{L} \left( \|u'(\xi)\| - L \frac{u'(\xi)}{\|u'(\xi)\|} \right) \right) - f(\xi) \right\} \cdot v(\xi) \, d\xi.
\]

Note:

\[v(0) = v(1) = 0 \Rightarrow \text{boundary terms vanish!}\]

(1.3.7) \Rightarrow \int_0^1 \left\{ -\frac{d}{d\xi} \left( \frac{\kappa(\xi)}{L} \left( \|u'(\xi)\| - L \frac{u'(\xi)}{\|u'(\xi)\|} \right) \right) - f(\xi) \right\} \cdot v(\xi) \, d\xi = 0

\[\forall v \in (C^1_0([0, 1]))^2 \}

\vspace{1cm}

R. Hiptmair
C. Schwab,
H. Harbrecht
V. Gradinaru
A. Chernov
SAM, ETHZ

1.3
p. 59
Lemma 1.3.23 (fundamental lemma of the calculus of variations). Let $f \in C^0_{pw}([a, b])$, $-\infty < a < b < \infty$, satisfy
\[
\int_a^b f(\xi)v(\xi) \, d\xi = 0 \quad \forall v \in C^k([a, b]), \, v(a) = v(b) = 0.
\]
for some $k \in \mathbb{N}_0$. This implies $f \equiv 0$.

Ass. (1.3.21) & (1.3.7) \quad \Rightarrow \quad \frac{d}{d\xi} \left( \frac{\kappa(\xi)}{L} \left( \|u'(\xi)\| - L \right) \frac{u'(\xi)}{\|u'(\xi)\|} \right) = f(\xi) \quad 0 \leq \xi \leq 1.

If $\kappa \in C^1$, $f \in C^0$, then a $C^2$-minimizer of $J$ a $C^2$-solution of (1.3.7) solve the 2nd-order ODE
\[
-\frac{d}{d\xi} \left( \frac{\kappa(\xi)}{L} \left( \|u\| - L \right) \frac{u'}{\|u\|} \right) = f \quad \text{on } [0; 1].
\]

ODE (1.3.24) + boundary conditions (1.2.1) = two-point boundary value problem (on domain $\Omega = [0, 1]$).
Minimization problem
(1.2.17)
\[ u_* = \arg\min_{v \in V} J(v) \]

Variational problem
(1.3.7)
\[ a(u; v) = f(v) \quad \forall v \]

Two-point BVP

\[ F(u, u', u'') = f, \]
\[ u(0), u(1) \text{ fixed}. \]

1: equivalence ("\(\iff\)") holds if minimization problem has unique solution

2: meaningful two-point BVP stipulates extra regularity (smoothness) of \(u\), see Rem. [1.3.25].

Terminology:

\[ \{ \text{minimization problem (1.2.17)} \} \]
\[ \{ \text{variational problem (1.3.7)} \} \]

is called the weak form of the string model,

Two-point boundary value problem (1.3.24), (1.2.1) is called the strong form of the string model.

A solution \(u\) of (1.3.24), for which all occurring derivatives are continuous is called a classical solution of the two-point BVP.
Remark 1.3.25 (Extra regularity requirements). \[ \rightarrow \text{Ex. 1.3.20} \]

Minimization problem (1.2.17):
- \( \kappa, f \) integrable,
- \( u \) piecewise \( C^1 \)

Variational problem (1.3.7):
- \( \kappa, f \) integrable,
- \( u \) piecewise \( C^1 \)

Two-point BVP:
- \( \kappa \in C^1([0, 1]) \),
- \( f \in (C^0([0, 1]))^2 \),
- \( u \in (C^2([0, 1]))^2 \).

formulation as a classical two-point BVP imposes (unduly) restrictive smoothness on solution and coefficient functions.

Lemma 1.3.26 (Classical solutions are weak solutions).

For \( \tilde{\kappa} \in C^1([0, 1]) \), any classical solution of (1.3.24) also solves (1.3.7).

Proof. (“Derivation of (1.3.24) reversed”)

R. Hiptmair
C. Schwab,
H. Harbrecht,
V. Gradinaru,
A. Chernov
SAM, ETHZ

1.3 Numerical Methods for PDEs
Multiply (1.3.24) with $v \in C^1_{0,pw}([0, 1])$ and integrate over $[0, 1]$. The push a derivative onto $v$ by using (1.3.22).

### 1.4 Simplified model

Setting: taut string

$$L \ll \|u(0) - u(1)\|.$$  

**expected:**

$$\|u'_*(\xi)\| \gg L \text{ for all } 0 \leq \xi \leq 1 \text{ for solution } u_* \text{ of } (1.2.17)$$

“Intuitive asymptotics”:

- renormalize stiffness $\kappa \to \tilde{\kappa} := \frac{\kappa}{L}$, $[\tilde{\kappa}] = \text{Nm}^{-1}$
- suppress equilibrium length: $L = 0$ in (1.2.17).
Simplified equilibrium model:

\[ \tilde{u}_* = \arg\min_{u \in (C^1_{pw}([0,1]))^2} \int_0^1 \frac{1}{2} \tilde{\kappa}(\xi) \| u'(\xi) \|^2 - f(\xi) \cdot u(\xi) \, d\xi. \]

\[ =: \tilde{J}(u) \]  \hfill (1.4.2)

\[ = \text{a quadratic minimization problem in a function space}! \]

**Remark 1.4.3 (Quadratic minimization problem).**

The functional (\(=\) mapping from a function space to \(\mathbb{R}\)) \(\tilde{J}\) from (1.4.2) has the structure

\[ \tilde{J}(u) = \frac{1}{2} a(u, u) - \ell(u), \]

with a symmetric **bilinear form** \(a : V \times V \mapsto \mathbb{R}\) and a **linear form** \(\ell : V \mapsto \mathbb{R}\).

Minimization problems for functionals of this form have been dubbed **quadratic minimization problems**.
Variational problem corresponding to (1.4.2): use
\[
\|x + th\|^2 = \|x\| + 2tx \cdot h + t^2 \|h\|^2 = \|x\| + 2tx \cdot h + O(t^2) .
\]

\[
\lim_{t \to 0} \frac{\tilde{J}(u + tv) - \tilde{J}(u)}{t} = \int_0^1 \tilde{\kappa}(\xi) u'(\xi) \cdot v'(\xi) - f(\xi) \cdot v(\xi) \, d\xi = 0 , \quad v \in (C^1_{pw,0([0, 1]))^2 .
\]

Variational equation satisfied by solution \(\tilde{u}_*\) of (1.4.2):
\[
\int_0^1 \tilde{\kappa}(\xi) u'_*(\xi) \cdot v'(\xi) - f(\xi) \cdot v(\xi) \, d\xi = 0 \quad \forall v \in (C^1_{pw,0([0, 1]))^2 . \quad (1.4.4)
\]

Remark 1.4.5 (Linear variational problems). \(\rightarrow \) Rem. 1.3.12

(1.4.4) has the structure (1.3.14)
\[
u \in V: \quad a(u, v) = l(v) \quad \forall v \in V_0 , \quad (1.4.6)
\]
where now

\[ a : V_0 \times V_0 \mapsto \mathbb{R} \] is a \textbf{bilinear form} (\rightarrow \text{Def. 1.3.13}), that is, linear in \textit{both} arguments.

In general, \textit{quadratic} minimization problems give rise to \textit{linear} variational problems.

This can be confirmed by an elementary computation:

\[ J(u) = \frac{1}{2}a(u, u) - \ell(u) \]

\[ \lim_{t \to 0} \frac{J(u + tv) - J(u)}{t} = \lim_{t \to 0} \frac{ta(u, v) + \frac{1}{2}t^2a(v, v) - t\ell(v)}{t} = a(u, v) - \ell(v), \]

where the bilinearity of \( a \) and the linearity of \( \ell \) was crucial, see Rem. 1.4.3.
Corresponding two-point boundary value problem: by integration by parts, see (1.3.22),

\[
\int_0^1 \tilde{\kappa}(\xi) u'_*(\xi) \cdot v'(\xi) - f(\xi) \cdot v(\xi) \, d\xi = \int_0^1 \left\{ -\frac{d}{d\xi} \left( \tilde{\kappa}(\xi) \frac{d}{d\xi} u(\xi) \right) - f(\xi) \right\} \cdot v(\xi) \, d\xi
\]

\[\forall v \in (C_{pw,0}^1([0, 1]))^2.\]

Then use Lemma 1.3.23.

If \( \kappa \in C^1, f \in C^0 \), then a \( C^2 \)-solution of (1.4.4) solves the two-point BVP

\[
-\frac{d}{d\xi} \left( \tilde{\kappa}(\xi) \frac{du}{d\xi}(\xi) \right) = f(\xi), \quad 0 \leq \xi \leq 1,
\]

\[
u(0) = \begin{pmatrix} a \\ u_a \end{pmatrix}, \quad \nu(1) = \begin{pmatrix} b \\ u_b \end{pmatrix}.
\]

(1.4.7)

Special setting: “gravitational force” \( f(\xi) = -g(\xi) e_2 \)
(1.4.2) decouples into two minimization problems for the components of \( \mathbf{u} \! \)

\[
\tilde{u}_1,* = \arg \min_{u \in C^1_{pw}([0,1]), u(0)=a, u(1)=b} \frac{1}{2} \int_0^1 \tilde{\kappa}(\xi)(u'(\xi))^2 \, d\xi ,
\]

\[
(1.4.2) \Rightarrow \quad \tilde{u}_2,* = \arg \min_{u \in C^1_{pw}([0,1]), u(0)=u_a, u(1)=u_b} \int_0^1 \frac{1}{2} \tilde{\kappa}(\xi)(u'(\xi))^2 + g(\xi)u(\xi) \, d\xi .
\]

The minimization problem for \( \tilde{u}_1,* \) has a closed-form solution:

\[
\tilde{u}_1,*(\xi) = a + \frac{b - a}{\int_0^1 \tilde{\kappa}^{-1}(\tau) \, d\tau} \int_0^\xi \tilde{\kappa}^{-1}(\tau) \, d\tau , \quad 0 \leq \xi \leq 1 . \tag{1.4.9}
\]

This solution can easily be found by converting the minimization problem to a 2-point boundary value problem as was done above, cf. (1.4.4), (1.4.7).

The minimization problem for \( \tilde{u}_2,* \) leads to the linear variational problem, cf. (1.4.4)

\[
\tilde{u}_2,* \in C^1_{pw}([0, 1]), \quad \tilde{u}_2,*(0) = u_a, \tilde{u}_2,*(1) = u_b : \quad \int_0^1 \tilde{\kappa}(\xi)\tilde{u}_2,*(\xi)v'(\xi) \, d\xi = - \int_0^1 g(\xi)v(\xi) \, d\xi \quad \forall v \in C^1_{0,pw}([0, 1]) . \tag{1.4.10}
\]
Remark 1.4.11 (Graph description of string shape).

Focus: situation with vertical gravitational force, see (1.4.9), (1.4.10)

Describe shape of string through graph of displacement function \( \hat{u}_* = \hat{u}_*(x) \), \( \hat{u} : [a, b] \mapsto \mathbb{R} \)
(physical units \([\hat{u}] = 1\text{m}\)).

\[ u_*(a) = u_a \quad , \quad u_*(b) = u_b . \]  \( (1.4.12) \)

\[ \hat{u}(x) = \tilde{u}_2,*(\Phi^{-1}(x)) \quad \text{with} \quad \Phi(\xi) := \tilde{u}_1,*(\xi) . \]  \( (1.4.13) \)

Here \( \tilde{u}_1,*(\xi) \), \( \tilde{u}_1,*(\xi) \) are the components of the curve description of the equilibrium shape of the string, see Sect. 1.2.1:

\[ u_*(\xi) = \begin{pmatrix} \tilde{u}_1,*(\xi) \\ \tilde{u}_2,*(\xi) \end{pmatrix} , \quad 0 \leq \xi \leq 1 . \]
Of course, the graph description is possible only for special string shapes. It also hinges on the choice of suitable coordinates.

Note: \( \xi \mapsto \Phi(\xi) \) is monotone, \( \Phi'(\xi) \neq 0 \) for all \( 0 \leq \xi \leq 1 \), \( \Phi(0) = a \), \( \Phi(1) = b \).

By chain rule [29, Thm. 5.1.3]:

\[
v(\xi) = \hat{v}(\Phi(\xi)) \quad \Rightarrow \quad v'(\xi) = \frac{d\hat{v}}{dx}(x) \Phi'(\xi), \quad x := \Phi(\xi) .
\]  

(1.4.14)

Recall: transformation formula for integrals in one dimension (substitution rule, \( x := \Phi(\xi) \), “\( dx = \Phi'(\xi) d\xi \)”: 

\[
q \in C^0_{pw}([0, 1]): \quad \int_0^1 q(\xi) \, d\xi = \int_a^b \hat{q}(x) \frac{1}{\Phi'(\Phi^{-1}(x))} \, dx , \quad \hat{q}(x) := g(\Phi^{-1}(x)) . \quad (1.4.15)
\]

\[ \leftarrow (1.4.14) \& (1.4.15) \]
Variational problem of taut string problem described as a function of spatial coordinate:

\[
\int_0^1 \tilde{\kappa}(\xi) \tilde{u}_{2,*}(\xi) v'(\xi) \, d\xi = \int_a^b \tilde{\kappa}(\Phi^{-1}(x)) \Phi'(\xi) \frac{d\hat{u}}{dx}(x) \Phi'(\xi) \frac{d\hat{v}}{dx}(x) \frac{1}{|\Phi'(\xi)|} \, dx \\
= \int_a^b \overbrace{\tilde{\kappa}(\Phi^{-1}(x))|\Phi'(\Phi^{-1}(x))|}^{=:\tilde{\sigma}(x)} \frac{d\hat{u}}{dx}(x) \frac{d\hat{v}}{dx}(x) \, dx ,
\]

\[
- \int_0^1 g(\xi) v(\xi) \, d\xi = - \int_a^b \frac{g(\Phi^{-1}(x))}{|\Phi'(\Phi^{-1}(x))|} \hat{v}(x) \, dx .
\]

\[
- \int_0^1 g(\xi) v(\xi) \, d\xi = - \int_a^b \frac{g(\Phi^{-1}(x))}{|\Phi'(\Phi^{-1}(x))|} \hat{v}(x) \, dx .
\]

**Linear variational problem in physical space coordinate on spatial domain \( \Omega = [a, b] \):**

\[
\hat{u}_* \in C^1_{pw}([a, b]), \quad \hat{u}_*(a) = u_a, \quad \hat{u}_*(b) = u_b \\
\int_a^b \tilde{\sigma}(x) \frac{d\hat{u}_*}{dx}(x) \frac{d\hat{v}}{dx}(x) \, dx = - \int_a^b \hat{g}(x) \hat{v}(x) \, dx \quad \forall \hat{v} \in C^1_{0, pw}([a, b]) .
\]

\[(1.4.16)\]
(assuming $\hat{\sigma} \in C^1([a, b])$) Two-point BVP

$$\text{(1.4.16)} \Rightarrow \left\{ \begin{array}{l}
\frac{d}{dx} \left( \hat{\sigma}(x) \frac{d\hat{u}_*(x)}{dx} \right) = \hat{g}(x), \quad a \leq x \leq b, \\
\hat{u}_*(a) = u_a, \quad \hat{u}_*(b) = u_b.
\end{array} \right. \quad \text{(1.4.17)}$$

1.5 Discretization

Goal: “computation” of a/the solution $u : [0, 1] \mapsto \mathbb{R}^2$ of

\begin{align*}
&\text{minimization problem (1.2.17)} \\
&\text{variational problem (1.3.7)} \\
&\text{two-point BVP (1.3.24) & (1.2.1)}
\end{align*}

a function: infinite amount of information, see [18, Rem. 3.1.3].

Well, just provide a **formula** for $u$ (analytic solution): 🙁 in general elusive for the above problems.
Only option: **Numerical algorithm** ➔ **approximate solution**

Finitely many floating point operations

- Numerical algorithms can only operate on discrete models

**Continuous (PDE) model**

(“∞-dimensional”)

**Discretization** ➔ **Discrete model**

(“finitely many unknowns”)

- As small as possible (only a few unknowns)
- As accurate as possible (good approximation (*)
- As faithful as possible (structure preserving)

(*) needs a measure for quality of a solution, usually a norm of the error, error = difference of exact/analytic and approximate solution.
Remark 1.5.1 (“Physics based” discretization).

Mass-spring model (→ Sect. 1.2.2) = discretization of the minimization problem (1.2.17) describing the elastic string.

This discretization may be called “physics based”, because it is inspired by the (physical) context of the model.

Note: Other approaches to discretization discussed below will lead to equations resembling the mass-spring model, see 1.5.1.2.

This section will present a few strategies on how to derive discrete models for the problem of computing the shape of an elastic string. The different approaches start from different formulations, some target the minimization problem (1.2.17), or, equivalently, the variational problem (1.3.7), while others tackle the ODE (1.3.24) together with the boundary conditions (1.2.1).
**Remark 1.5.2 (Timestepping for ODEs).**

For initial value problems for ODEs, whose solutions are functions, too, we also face the problem of discretization: timestepping methods compute a finite number of approximate values of the solutions at discrete instances in time, see [18, Ch. 12].

**Remark 1.5.3 (Coefficients/data in procedural form).**

For the elastic string model (→ Sect. 1.2.3) the stiffness $\kappa(\xi)$, and force field $f$ may not be available in closed form (as formulas).

Instead they are usually given in **procedural form**:

```matlab
function k = kappa(xi);
function f = force(xi);
```
because they may be obtained

- as results of another computation,
- by interpolation from a table.

viable discretizations must be able to deal with data in procedural form!

### 1.5.1 Galerkin discretization

Simple idea of first step of Galerkin discretization

\[
\begin{align*}
\text{In} & \quad \text{minimization problem, e.g., (1.2.17)} \\
\leftrightarrow & \quad \text{variational problem, e.g. (1.3.7)} \\
\text{replace function space } V/V_0 \text{ with finite dimensional subspace } V_N/V_{N,0}
\end{align*}
\]
Note that a subscript tag $N$ distinguishes “discrete functions/quantities”, that is, functions/operators etc. that are associated with a finite dimensional space. In some contexts, $N$ will also be an integer designating the dimension of a finite dimensional space.

Formal presentation: $V, V_0 :$ (affine) function spaces, $\dim V_0 = \infty,$ $V_N, V_{N,0} :$ subspaces $V_N \subset V, V_{N,0} \subset V_0,$ $N := \dim V_{N,0}, \dim V_N < \infty.$

Galerkin discretization of minimization problem for functional $J : V \mapsto \mathbb{R}$:

Continuous minimization problem

$$u = \arg\min_{v \in V} J(v) . \quad (1.5.4)$$

Discrete minimization problem

$$u_N = \arg\min_{v_N \in V_N} J(v_N) . \quad (1.5.5)$$

Galerkin discretization of abstract (non-linear) variational problem (1.3.14), see Rem. 1.3.12
Continuous variational problem

\[ u \in V : \quad a(u; v) = \ell(v) \quad \forall v \in V_0. \tag{1.5.6} \]

Discrete variational problem

\[ u_N \in V_N : \quad a(u_N; v_N) = \ell(v_N) \quad \forall v_N \in V_{N,0}. \tag{1.5.7} \]

Galerkin disc.

Terminology:

\( u_N \in V_N \) satisfying (1.5.5)/(1.5.7) is called a [Galerkin solution](#) of (9.2.21)/(9.3.62).

\( V_N \) is called the [Galerkin trial space](#), \( V_{N,0} \) is the [Galerkin test space](#).

**Remark 1.5.8 (Relationship between discrete minimization problem and discrete variational problem).**

In Sect. 1.3.1 we discovered the equivalence

\[ \text{Continuous minimization problem} \quad (9.2.21) \quad \iff \quad \text{Continuous variational problem} \quad (9.3.62) \]
Now it seems that we have \textit{two different} strategies for Galerkin discretization:

1. Galerkin discretization via the discrete minimization problem (1.5.5),
2. Galerkin discretization based on the discrete variational problem (1.5.7).

However, \textbf{the above equivalence extends to the discrete problems!}

More precisely, we have the \textit{commuting relationship}:

\[
\begin{align*}
\text{minimization problem} & \quad \overset{\text{Galerkin discretization}}{\longrightarrow} \quad \text{discrete minimization problem} \\
\Downarrow \quad \text{Configurational derivative in } V_0 & \quad \Downarrow \quad \text{Configurational derivative in } V_{N,0} \\
\text{variational problem} & \quad \overset{\text{Galerkin discretization}}{\longrightarrow} \quad \text{discrete variational problem}.
\end{align*}
\]

The commuting diagram means that the same discrete variational problem is obtained no matter whether

1. the minimization problem is first restricted to a finite dimensional subspace and the result is converted into a variational problem according to the recipe of Sect. 1.3.1.
2. or whether the variational problem derived from the minimization problem is restricted to the subspace.

To see this, understand that the manipulations of Sect. 1.3.1 can be carried out for infinite and finite dimensional function spaces alike.

Remark 1.5.10 (Offset functions and Galerkin discretization).

Often: \( V = u_0 + V_0 \), with *offset function* \( u_0 \rightarrow \text{Rem. 1.3.19, (1.3.16)} \)

If \( u_0 \) is sufficiently simple, we may choose a trial space \( V_N = u_0 + V_{N,0} \)

\( \geq \) Discrete variational problem analogous to (1.3.19)

\[ w_N \in V_{N,0}: \quad a(u_0 + w_N; v_N) = \ell(v_N) \quad \forall v_N \in V_{N,0} \quad \Rightarrow \quad u_N := w_N + u_0 . \] (1.5.11)

In the case of a linear variational problem (\( \rightarrow \) Rem. 1.4.5), that is, a *bilinear form* \( a \), we have

\[ a(w_N, v_N) = \ell(v_N) - a(u_0, v_N) \quad \forall v_N \in V_{N,0} . \] (1.5.12)
Below we will always make the assumption \( V = u_0 + V_0 \).

However, a computer is clueless about a concept like “finite dimensional subspace”. What it can process are arrays of floating point numbers.

Idea:

1. Choose basis \( \mathcal{B}_N = \{ b_1^N, \ldots, b_N^N \} \) of \( V_{N,0} \): \( V_{N,0} = \text{Span} \{ \mathcal{B}_N \} \)
2. Insert basis representation into minimization problem (1.5.5)

\[
v_N \in V_{N,0} \Rightarrow v_N = u_0 + \nu_1 b_1^N + \cdots + \nu_N b_N^N, \quad \nu_i \in \mathbb{R}, \quad (1.5.13)
\]

and variational equation (1.5.7)

\[
v_N \in V_{N,0} \Rightarrow v_N = \nu_1 b_1^N + \cdots + \nu_N b_N^N, \quad \nu_i \in \mathbb{R}, \quad (1.5.14)
\]

\[
u_N \in V_N \Rightarrow u_N = u_0 + \mu_1 b_1^N + \cdots + \mu_N b_N^N, \quad \mu_i \in \mathbb{R}. \quad (1.5.15)
\]
**Remark 1.5.16 (Ordered basis of test space).**

Once we have chosen a basis $\mathcal{B}$ and ordered it, as already indicated in the notation above, the test space $V_{N,0}$ can be identified with $\mathbb{R}^N$: a coefficient vector $\vec{\mu} = (\mu_1, \ldots, \mu_N)^T \in \mathbb{R}^N$ provides a unique characterization of a function $u \in V_{N,0}$ (basis property)

$$u = \sum_{j=1}^{N} \mu_j b_j^N.$$ 

---

**Discrete minimization problem**

$$u_N = \arg\min_{v_N \in V} J(v_N). \quad (1.5.5)$$

**Minimization problem on $\mathbb{R}^N$**

$$\vec{\mu} = \arg\min_{\vec{\nu} \in \mathbb{R}^N} F(\vec{\nu}), \quad (1.5.17)$$

$$F(\vec{\nu}) := J(u_0 + \nu_1 b_1^N + \cdots + \nu_N b_N^N).$$
amenable to classical optimization techniques

notation: \( \vec{\nu}, \vec{\mu} \triangleq \text{vectors of coefficients} \ (\nu_i)_{i=1}^N, (\mu_i)_{i=1}^N, \) in basis representation of functions \( v_N, u_N \in V_N \) according to (1.5.13).

Discrete variational problem

\[
\begin{align*}
\text{System of equations} \\
\quad u_N \in V_N: \quad a(u_N; v_N) = \ell(v_N) \\
\forall v_N \in V_{N,0} . \quad (1.5.7)
\end{align*}
\]

\[
\begin{align*}
\text{Basis representation} \\
a(u_0 + \sum_{j=1}^N \mu_j b_j^N; b_k^N) = \ell(b_k^N) \\
\forall k = 1, \ldots, N . \quad (1.5.18)
\end{align*}
\]

use techniques for linear/non-linear systems of equations, see [18, Ch. 2], [18, Ch. 4].

The choice of the basis \( \mathcal{B} \) has no impact on the (set of) Galerkin solutions of (1.5.7)!
Below, we apply Galerkin approaches to

- (1.4.16) as an example for the treatment of a linear variational problem:

\[ u \in C^1_{pw}([a, b]), \quad u(a) = u_a, \ u(b) = u_b, \quad \int_a^b \sigma(x) \frac{du}{dx}(x) \frac{dv}{dx}(x) \, dx = - \int_a^b g(x)v(x) \, dx \quad \forall v \in C^1_{0, pw}([a, b]). \]

(1.4.16)

Here:
- spatial domain \( \Omega = [a, b] \), linear offset function \( u_0(x) = \frac{b-x}{b-a} u_a + \frac{x-a}{b-a} u_b \),
- function space \( V_0 = C^1_{0, pw}([a, b]). \)

- (1.3.7) to demonstrate its use in the case of a non-linear variational equation:

\[ u \in C^1_{pw}([0, 1]), \quad \int_0^1 \frac{\kappa(\xi)}{L} \left( \|u'(\xi)\| - L \right) \frac{u'(\xi) \cdot v'(\xi)}{\|u'(\xi)\|} - f(\xi) \cdot v(\xi) \, d\xi = 0 \]

\( \forall v \in (C^1_{0, pw}([0, 1]))^2. \)  

(1.3.7)

Here:
- parameter domain \( \Omega = [0, 1] \), linear offset function \( u_0(\xi) = \xi u(0) + (1 - \xi) u(1), \)
- function space \( V_0 = (C^1_{0, pw}([a, b]))^2. \)
1.5.1.1 Spectral Galerkin scheme

A simple function space (widely used for interpolation, see [18, Ch. 9], and approximation, see [18, Sec. 9.1]): for interval $\Omega \subset \mathbb{R}$

$$V_{N,0} = \mathcal{P}_p(\mathbb{R}) \cap C^0_0(\Omega)$$

$\hat{=} \text{space of univariate polynomials of degree } \leq p \text{ vanishing at endpoints of } \Omega,$

$$N := \dim V_N = p - 1$$

(1.5.19)

Obvious: choice (1.5.19) guarantees $V_N \subset C^1_{pw,0}(\Omega)$ (even $V_{N,0} \subset C^\infty(\Omega)$)

Please note that $V_{N,0}$ is a space of global polynomials on $\Omega$.

Example 1.5.20 (Spectral Galerkin discretization of linear variational problem).
Targeted: linear variational problem (1.4.16) with

- \( a = 0, \ b = 1 \) ➞ domain \( \Omega = [0, 1] \),
- constant coefficient function \( \sigma \equiv 1 \),
- load \( g(x) = -4\pi (\cos(2\pi x^2) - 4\pi x^2 \sin(2\pi x^2)) \),
- boundary values \( u_a = u_b = 0 \).

Concrete variational problem

\[
\int_0^1 \frac{du}{dx}(x) \frac{dv}{dx}(x) \, dx = - \int_0^1 g(x)v(x) \, dx \quad \forall v \in C^1_{0,pw}([0,1]) \, . \tag{1.5.21}
\]

\[
\Rightarrow \quad u(x) = \sin(2\pi x^2), \quad 0 < x < 1 \, .
\]

because \( \frac{d^2u}{dx^2}(x) = g(x) \).
Polynomial spectral Galerkin discretization, degree \( p \in \{4, 5, 6\} \).

Plots of approximate/exact solutions

Remark 1.5.22 (Choice of basis for polynomial spectral Galerkin methods).

Sought: (ordered) basis of \( V_{N,0} \) := \( C^1_0([-1, 1]) \cap P_p(\mathbb{R}) \)

“Tempting”: monomial-type basis

\[ V_{N,0} = \text{Span} \left\{ 1 - x^2, x(1 - x^2), x^2(1 - x^2), \ldots, x^{p-2}(1 - x^2) \right\} \]  \hspace{1cm} (1.5.23)
Monomial basis polynomials

Beware: ill-conditioned!

→ Ex. 1.5.46 below

"Popular": integrated Legendre polynomials

\[ V_{N,0} = \text{Span} \left\{ x \mapsto M_n(x) := \int_{-1}^{x} P_n(\tau) \, d\tau, \quad n = 1, \ldots, p - 1 \right\}, \quad (1.5.24) \]

where \( P_n \) = \( n \)-th Legendre polynomial.
Definition 1.5.25 (Legendre polynomials). → [18, Def. 10.4.12]

The $n$-th Legendre polynomial $P_n$, $n \in \mathbb{N}_0$, is defined by (Rodriguez formula)

$$P_n(x) := \frac{1}{n!2^n} \frac{d^n}{dx^n}[(x^2 - 1)^n].$$
Legendre polynomials $P_0, \ldots, P_5$

\[
\begin{align*}
P_0(x) &= 1, \\
P_1(x) &= x, \\
P_2(x) &= \frac{3}{2}x^2 - \frac{1}{2}, \\
P_3(x) &= \frac{5}{2}x^3 - \frac{3}{2}x, \\
P_4(x) &= \frac{35}{8}x^4 - \frac{15}{4}x^2 + \frac{3}{8}.
\end{align*}
\]

Some facts about Legendre polynomials:

- **Symmetry:**

  \[ P_n \text{ is } \begin{cases} \text{even} & \text{for even } n \text{ odd } n, \\
  \text{odd} & \text{for odd } n \text{ odd } n, \end{cases} \quad P_n(1) = 1, \quad P_n(-1) = (-1)^n. \quad (1.5.26) \]

- **Orthogonality**

  \[
  \int_{-1}^{1} P_n(x)P_m(x)dx = \begin{cases} 
  \frac{2}{2n+1}, & \text{if } m = n, \\
  0, & \text{else.}
  \end{cases} \quad (1.5.27)
\]
3-term recursion

\[ P_{n+1}(x) := \frac{2n+1}{n+1} x P_n(t) - \frac{n}{n+1} P_{n-1}(x) \quad , \quad P_0 := 1 \quad , \quad P_1(x) := x . \quad (1.5.28) \]

This formula paves the way for the efficient evaluation of all Legendre polynomials at many (quadrature) points, see Code [1.5.28].

Code 1.5.29: Computation of Legendre polynomials based on 3-term recursion (1.5.28)

```matlab
function V = legendre(n, x)
% Computes values of Legendre polynomials up to degree n
% in the points x_j passed in the row vector x.
% Exploits the 3-term recursion (1.5.28) for Legendre polynomials
V = ones(size(x)); V = [V; x];
for j=1:n-1
    V = [V; ((2*j+1)/(j+1)).*x.*V(end,:)-j/(j+1)*V(end-1,:)];
end
```

Representation of derivatives and primitives, cf. Code [1.5.31]:

\[ P_n(x) = \left( \frac{d}{dx} P_{n+1}(x) - \frac{d}{dx} P_{n-1}(x) \right) / (2n+1) \quad , \quad n \in \mathbb{N} \quad , \quad (1.5.30) \]

\[ M_n(x) = \frac{1}{2n+1} (P_{n+1}(x) - P_{n-1}(x)) \quad \text{and} \quad \frac{dM_n}{dx} = P_n . \quad (1.5.31) \]
Code 1.5.32: Computation of (integrated) Legendre polynomials using (1.5.28) and (1.5.31)

```
function [V,M] = intlegpol(n,x)
% Computes values of the first n+1 Legendre polynomials \( P_n \) (returned in
% matrix V) and the first n−1 integrated Legendre polynomials
% \( M_n \) (returned in matrix M) in the points \( x_j \) passed in the
% row vector x. Uses the recursion formulas (1.5.28) and
% (1.5.31)
V = ones(size(x)); V = [V; x];
for j=1:n-1, V = [V; ((2*j+1)/(j+1)) * x .* V(end,:); -j/(j+1) * V(end-1,:)];
end
M = diag(1./(2*(1:n-1)+1)) * (V(3:n+1,:) - V(1:n-1,:));
```

Remark 1.5.33 (Transformation of basis functions).

On a “general domain \( \Omega = [a,b] \)”, we obtain the basis function by a so-called affine transformation of the basis functions on \([0,1]\), cf. [18, Rem. 10.1.3], e.g., in the case of integrated Legendre polynomials as basis functions on \( \Omega = [a,b] \) we use the basis functions

\[
b^i_N(x) = M_i \left( 2 \frac{x-a}{b-a} - 1 \right), \quad a \leq x \leq b. \tag{1.5.34}
\]
Note the effect of this transformation on the derivative (chain rule!):

$$\frac{db^i_N}{dx}(x) = \frac{dM_i}{dx} \left( \frac{2x - a}{b - a} - 1 \right) \cdot \frac{2}{b - a} = P_i \left( \frac{2x - a}{b - a} - 1 \right) \cdot \frac{2}{b - a} .$$  \hspace{1cm} (1.5.35)

\[\blacktriangledown\]

\textit{Remark} 1.5.36 (Spectral Galerkin discretization with quadrature).

Consider the linear variational problem, \textit{cf.} (1.4.16),

$$u \in C^1_{0,pw}([a, b]): \quad \int_a^b \sigma(x) \frac{du}{dx}(x) \frac{dv}{dx}(x) \, dx = \int_a^b g(x) v(x) \, dx \quad \forall v \in C^1_{0,pw}([a, b]) .$$ \hspace{1cm} (1.5.37)

Assume: \ \sigma, \ g \ \text{only given in procedural form, see Rem.} \ [1.5.3].

\[\blacktriangledown\]

Analytic evaluation of integrals becomes impossible even if \( u, v \) polynomials!
Only remaining option: Numerical quadrature, see [18, Ch. 10]

Replace integral with $m$-point quadrature formula on $[a, b], m \in \mathbb{N}$ → [18, Sect. 10.1]:

$$
\int_{a}^{b} f(t) \, dt \approx Q_m(f) := \sum_{j=1}^{m} \omega_j^m f(\zeta_j^m).
$$

(1.5.38)

$\omega_j^m$: quadrature weights, $\zeta_j^m$: quadrature nodes $\in [a, b]$.

(1.5.37) ➤ discrete variational problem with quadrature:

$$
u_N \in V_N: \sum_{j=1}^{m} \omega_j^m \sigma(\zeta_j^m) \frac{du_N}{dx}(\zeta_j^m) \frac{dv_N}{dx}(\zeta_j^m) = \sum_{j=1}^{m} \omega_j^m g(\zeta_j^m) v(\zeta_j^m) \quad \forall v \in V_N.
$$

(1.5.39)

Popular (global) quadrature formulas: Gauss quadrature → [18, Sect. 10.4]

Important: Accuracy of quadrature formula and computational cost (no. $m$ of quadrature nodes) have to be balanced.
Remark 1.5.40 (Implementation of spectral Galerkin discretization for linear 2nd-order two-point BVP).

Setting:

- linear variational problem \((1.5.37)\) \(\implies u_0 = 0\),
- coefficients \(\sigma, g\) in procedural form, see Rem. 1.5.3,
- approximation of integrals by \(p\)-point Gaussian quadrature formula,
- polynomial spectral Galerkin discretization, degree \(\leq p, p \geq 2\),
- basis \(\mathcal{B}\): integrated Legendre polynomials, see \((1.5.24)\):

\[
V_{N,0} = \text{Span}\{M_n, n = 1, \ldots, p - 1\}, \quad M_n \overset{\text{def}}{=} \text{integrated Legendre polynomials}.
\]

Trial expression, cf. \((1.5.13)\)

\[
u_N = \mu_1 M_1 + \mu_2 M_2 + \cdots + \mu_N M_N, \quad \mu_i \in \mathbb{R}, \quad N := p - 1.
\]
Note: by definition \( \frac{d}{dx} M_n = P_n \).

From (1.5.39) with (1.5.40)

\[
\sum_{j=1}^{m} \omega_j^m \sigma(\zeta_j^m) \sum_{l=1}^{N} \mu_l P_l(\zeta_j^m) P_k(\zeta_j^m) = \sum_{j=1}^{m} \omega_j^m g(\zeta_j^m) M_k(\zeta_j^m) , \quad k = 1, \ldots, N .
\]

\[
\equiv \varphi_k
\]

\[
\sum_{l=1}^{N} \left( \sum_{j=1}^{m} \omega_j^m \sigma(\zeta_j^m) P_l(\zeta_j^m) P_k(\zeta_j^m) \right) \mu_l = \varphi_k , \quad k = 1, \ldots, N .
\]

A linear system of equations!

\[
\begin{align*}
(A)_{kl} &:= \sum_{j=1}^{m} \omega_j^m \sigma(\zeta_j^m) P_l(\zeta_j^m) P_k(\zeta_j^m) , \quad k, l = 1, \ldots, N , \\
\bar{\mu} &:= (\mu_l)_{l=1}^{N} \in \mathbb{R}^N , \quad \bar{\varphi} = (\varphi_k)_{k=1}^{N} \in \mathbb{R}^N .
\end{align*}
\]

The Galerkin discretization of a linear variational problem leads to a linear system of equations.
Code 1.5.44: Polynomial spectral Galerkin solution of \eqref{1.5.37}

1  function \( u = \text{lin2pbvpspecgalquad}(\sigma, g, N, x) \)
2  \% Polynomial spectral Galerkin discretization of linear 2nd-order two-point BVP
3  \% \(-d/dx(\sigma(x)d/dx) = g(x), \ u(0) = u(1) = 0\)
4  \% on \( \Omega = [0,1] \). Trial space of dimension \( N \).
5  \% Values of approximate solution in points \( x_j \) are returned in the row vector \( u \)
6  \m = N+1; \quad \% Number of quadrature nodes
7  [\text{zeta}, \text{w}] = \text{gaussquad}(m); \quad \% Obtain Gauss quadrature nodes w.r.t \([-1,1]\]
8  \% Compute values of (integrated) Legendre polynomials at Gauss nodes
9  \[V, M\] = \text{intlegpol}(N+1, \text{zeta’});
10  \omega = \text{w’}.*\sigma((\text{zeta’}+1)/2) \times 2; \quad \% Modified quadrature weights
11  A = V(2:N+1,:) \times \text{diag}(\omega) \times V(2:N+1,:)’; \quad \% Assemble Galerkin matrix
12  \phi = M \times (0.5 \times \text{w’} \times g((\text{zeta’}+1)/2)’); \quad \% Assemble right hand side vector
13  \mu = A \backslash \phi; \quad \% Solve linear system
14  \% Compute values of integrated Legendre polynomials at output points
15  \[V, M\] = \text{intlegpol}(N+1, 2 \times x - 1); \; u = \mu’ \times M;

Code 1.5.45: MATLAB driver script creating plots of Ex. 1.5.20

1  \% MATLAB script: Driver routine for polynomial spectral Galerkin discretization of linear 2nd-order
2  \% two-point boundary value problem
3  \text{clear all;}
4  \% Coefficient functions (function handles, see MATLAB help)
5  \sigma = @(x) \text{ones(size}(x));
6  g = @(x) \text{-4*}\pi\times(\cos(2*\pi\times x.\times 2)\text{-4*}\pi\times x.\times 2.*\sin(2*\pi\times x.\times 2));
Example 1.5.46 (Conditioning of spectral Galerkin system matrices).
Finally we can provide a rationale for preferring integrated Legendre polynomials to plain monomials for polynomial spectral Galerkin discretization: the argument is based on condition number of the system matrix from (1.5.43).

Linear variational problem (1.5.21) with bilinear form

\[ a(u, v) = \int_0^1 \frac{du}{dx}(x) \frac{dv}{dx}(x) \, dx, \quad u, v \in C^1_{0,pw}([0, 1]). \]

Choice of basis functions for Galerkin trial/test space \( V_{N,0} := P_p(\mathbb{R}) \cap C^0_0([0, 1]) \): monomial basis (1.5.23), integrated Legendre polynomials (1.5.24).
Monitored: condition number (w.r.t. Euclidean matrix norm $\rightarrow [18, \text{Def. 2.5.26}]$) of Galerkin matrices

Recall that a condition number of $10^m$ involves a loss of $m$ digits w.r.t. the precision guaranteed for the right hand side of the linear system. Thus, using the monomial basis for $N > 10$ may no longer produce reliable results.
**Example 1.5.47** (Implementation of spectral Galerkin discretization for elastic string problem).

Targetted:  
*non-linear* variational equation on domain $\Omega = [0, 1]$

\[
\int_{0}^{1} \frac{\kappa(\xi)}{L} \left( 1 - \frac{L}{\|u'(\xi)\|} \right) u'(\xi) \cdot v'(\xi) - f(\xi) \cdot v(\xi) \, d\xi = 0 \quad \forall v \in (C_{0, pw}^1([0, 1]))^2 .
\]  

Data $\kappa, f$ given in procedural form, see Rem. [1.5.3].

Spectral Galerkin discretization, basis $\mathcal{B} = \{M_n\}_{n=1}^{K}, K \in \mathbb{N}$, consists of integrated Legendre polynomials, see (1.5.24) \(\Rightarrow\) basis representation, cf. (1.5.40)

\[
u_N(\xi) = u(0)(1 - \xi) + u(1)\xi + \left( \begin{array}{c} \mu_1 \\ \mu K+1 \end{array} \right) M_1(\xi) + \cdots + \left( \begin{array}{c} \mu K \\ \mu 2K \end{array} \right) M_K(\xi) .
\]  

Approximate evaluation of integrals by $m$-point Gaussian quadrature on $[0, 1]$, $m := K + 1$ below: nodes $\zeta_j$, weights $\omega_j$, $j = 1, \ldots, m$. 

R. Hiptmair  
C. Schwab,  
H. Harbrecht  
V. Gradinaru  
A. Chernov  
SAM, ETHZ  

1.5  
p. 101
In analogy to (1.5.41) we arrive at the non-linear system of equations: \( M'_k = P_k \)

\[
\sum_{j=1}^{m} s_j (b - a) \sum_{l=1}^{K} \mu_l P_l(\zeta_j) \cdot P_k(\zeta_j) = \sum_{j=1}^{m} \omega_j f_1(\zeta_j) \cdot M_k(\zeta_j), \quad k = 1, \ldots, K,
\]

\[
\sum_{j=1}^{m} s_j (u_b - u_a) \sum_{l=1}^{K} \mu_{K+l} P_l(\zeta_j) \cdot P_k(\zeta_j) = \sum_{j=1}^{m} \omega_j f_2(\zeta_j) \cdot M_k(\zeta_j), \quad k = 1, \ldots, K,
\]

with \( s_j := \omega_j \kappa(\zeta_j) \left( \frac{1}{L} - \frac{1}{\|u'_N(\zeta_j)\|} \right) \) \( (s = s(\vec{\mu})) \).

Rewrite in compact form:

\[
\begin{pmatrix}
R(\vec{\mu}) & 0 \\
0 & R(\vec{\mu})
\end{pmatrix}
\begin{pmatrix}
\vec{\mu}
\end{pmatrix}
=
\begin{pmatrix}
\vec{\varphi}_1 \\
\vec{\varphi}_2
\end{pmatrix},
\]

\[
(\vec{\varphi}_1)_k = \sum_{j=1}^{m} \omega_j f_1(\zeta_j) \cdot M_k(\zeta_j) - (b - a) \sum_{j=1}^{m} s_j P_k(\zeta_j), \quad (1.5.50)
\]

\[
(\vec{\varphi}_1)_k = \sum_{j=1}^{m} \omega_j f_2(\zeta_j) \cdot M_k(\zeta_j) - (u_b - u_a) \sum_{j=1}^{m} s_j P_k(\zeta_j).
\]
Iterative solution of (1.5.49) by fixed point iteration:

Initial guess $\vec{\mu}^{(0)} \in \mathbb{R}^N$; $k = 0$;

repeat
  $k \leftarrow k + 1$;
  Solve the linear system of equations
  \[
  \begin{pmatrix}
  \mathbf{R}(\vec{\mu}^{(k-1)}) & 0 \\
  0 & \mathbf{R}(\vec{\mu}^{(k-1)})
  \end{pmatrix}
  \vec{\mu}^{(k)} = \begin{pmatrix} \vec{\varphi}_1 \\ \vec{\varphi}_2 \end{pmatrix};
  \]

until $\|\vec{\mu}^{(k)} - \vec{\mu}^{(k-1)}\| \leq \text{tol} \cdot \|\vec{\mu}^{(k)}\|$

Code 1.5.51: Polynomial spectral Galerkin discretization of elastic string variational problem

```matlab
function [vu,figsol] = stringspecgal(kappa,f,L,u0,u1,K,xi,tol)

% Solving the non-linear variational problem (1.3.7) for the elastic string by
% means of polynomial
% spectral Galerkin discretization based on K integratted Legendre polynomials.
% Approximate evaluation of integrals by means of Gaussian quadrature.
% kappa, f are handles of type @(xi) providing the coefficient function
% kappa and the force field f. The column vectors u0 and u1 pass the
% pinning points. M is the number of mesh cells, tol specifies the tolerance
% for the
% fixed point iteration. return value: 2x length(xi)-matrix of node
% positions for curve parameter values passed in the row vector xi.
if ( nargin < 8), tol = 1E-2; end
```
m = K+1; % Number of quadrature nodes
[zeta, w] = gaussquad(m); % Obtain Gauss quadrature nodes w.r.t [-1,1]
% Compute values of (integrated) Legendre polynomials at Gauss nodes and evaluation points
[V, M] = intlegpol(K+1, zeta');
[Vx, Mx] = intlegpol(K+1, 2 * xi - 1); Mx = [1 - xi; Mx; xi]; %
% Compute right hand side based on m-point Gaussian quadrature on [0,1].
force = f((zeta' + 1)/2); phi = M * (0.5 * [w'; w']. * force)';
sv = kappa((zeta' + 1)/2); % Values of coefficient function \( \kappa \) at Gauss points in [0,1].
% mu is an 2 \times (K+2)-matrix, containing the vectorial basis expansion coefficients
% of \( u_N \). The first and last column are contributions of the two functions
% \( \xi \mapsto (1 - \xi) \) and \( \xi \mapsto \xi \), which represent the offset function.
% Initial guess for fixed point iteration: straight string
mu = [u0, zeros(2, K), u1];
figsol = figure; hold on;
for k=1:100 % loop for fixed point iteration, maximum 100 iterations
% Plot shape of string
vu = mu * Mx; plot(vu(1,:), vu(2,:), '--g'); drawnow;
title(sprintf('K = %d, iteration #%d', K, k));
xlabel('{\bf x_1}'); ylabel('{\bf x_2}');
% Compute values of derivatives of \( u_N \) and \( \|u_N'\| \) at Gauss points
up = mu(:, 2:K) * V(2:K, :) + repmat(u1 - u0, 1, m);
lup = sqrt(up(1,:).^2 + up(2,:).^2);
```matlab
s = 0.5*(w').*sv.*(1/L - 1./lup);  \% Initialization of s_j

\% Modification of right hand side due to offset function
phi1 = phi(:,1) + (2*(u1(1)-u0(1))'*V(2:K+1,:)'*s');
phi2 = phi(:,2) + (2*(u1(2)-u0(2))'*V(2:K+1,:)'*s');

\% Assemble K × K-matrix blocks R of linear system
R = 4*V(2:K+1,:) * diag(s) * V(2:K+1,:);' 
mu_new = [u0,[(R\phi1)';(R\phi2)'],u1];

\% Check simple termination criterion for fixed point iteration.
if (norm(mu_new - mu,'fro') < tol*norm(mu_new,'fro')/K)
    vu = mu*Mx; fig = plot(vu(1,:),vu(2,:),'r--');
    legend(fig,'spectral Galerkin solution','location','southeast'); break; end
mu = mu_new;
end
```

**Example 1.5.52 (Spectral Galerkin discretization for elastic string simulation).**

Test of polynomial spectral Galerkin method for elastic string problem, algorithm of Ex. 1.5.47, Code 1.5

with

- pinning positions \( u(0) = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \ u(1) = \begin{pmatrix} 1 \\ 0.2 \end{pmatrix} \),
1.5.1.2 Linear finite elements

Two ways to approximate functions by polynomials:
The spectral polynomial Galerkin approach presented in Sect. [1.5.1.1] relies on global polynomials. Now let us examine the use of piecewise polynomials.

Preliminaries: piecewise polynomials have to be defined w.r.t. partitioning of the domain $\Omega \subset \mathbb{R}$

$\Omega = [a, b]$ equipped with nodes $(M \in \mathbb{N})$

$X := \{a = x_0 < x_1 < \cdots < x_{M-1} < x_M = b\}$.

- mesh/grid

$M := \{[x_{j-1}, x_j]: 1 \leq j \leq M\}.$

Special case: equidistant mesh: $x_j := a + jh, \quad h := \frac{b - a}{M}$.

$[x_{j-1}, x_j], j = 1, \ldots, M, \triangleq$ cells of $M$, cell size $h_j := |x_j - x_{j-1}|, j = 1, \ldots, M$ meshwidth $h_M := \max_{j} |x_j - x_{j-1}|$. 

[18, Ch. 9] → piecewise polynomials

[18, Ch. ??]
Recall from Sect. 1.3.2: merely continuous, piecewise $C^1$ trial and test functions provide valid trial/test functions!

\[ V_N = S^0_{1,0}(\mathcal{M}) := \left\{ v \in C^0([0, 1]): v|_{[x_{i-1}, x_i]} \text{ linear, } i = 1, \ldots, M, v(a) = v(b) = 0 \right\} \]

Simplest choice for test space

\[ N := \dim V_N = M - 1 \]

Choice of (ordered) basis $\mathfrak{B}_N$ of $V_N$?

1D “tent functions”

\[ \mathfrak{B} = \{ b^1_N, \ldots, b^{M-1}_N \} \]

\[ b^j_N(x_i) = \delta_{ij} := \begin{cases} 1, & \text{if } i = j \\ 0, & \text{if } i \neq j \end{cases} \]
\[
\frac{d b^j_N}{dx}(x) = \begin{cases} 
\frac{1}{h_j}, & \text{if } x_{j-1} \leq x \leq x_j, \\
-\frac{1}{h_{j+1}}, & \text{if } x_j < x \leq x_{j+1}, \\
0 & \text{elsewhere. (piecewise derivative!)}
\end{cases}
\] (1.5.55)

**Remark 1.5.56 (Benefit of variational formulation of BVPs).**

The possibility of using simple piecewise linear trial and test functions is a clear benefit of the variational formulation that can accommodate merely piecewise continuously differentiable functions, see Sect. 1.3.2.

Below, in Sect. 1.5.2 we will learn about a method that targets the strong form of the 2-point BVP and, thus, has to impose more regularity on the trial functions.
simplest case: linear variational problem with constant stiffness coefficient

\[ u \in C^1_{0,pw}([a, b]): \quad \int_a^b \frac{du}{dx}(x) \frac{dv}{dx}(x) \, dx = \int_a^b g(x)v(x) \, dx \quad \forall v \in C^1_{0,pw}([a, b]) . \]

Discrete variational problem with \( u_N = \mu_1 b_N^1 + \cdots + \mu_N b_N^N \): 

\[ \int_a^b \sum_{l=1}^N \mu_l \frac{db_l^1}{dx}(x) \frac{db_k^N}{dx}(x) \, dx = \int_a^b g(x)b_k^N(x) \, dx \quad k = 1, \ldots, N . \]

\[ \sum_{l=1}^N \left( \int_a^b \frac{db_l^1}{dx}(x) \frac{db_k^N}{dx}(x) \, dx \right) \mu_l = \int_a^b g(x)b_k^N(x) \, dx , k = 1, \ldots, N . \]

\[ (A)_{kl} := \int_a^b \frac{db_l^1}{dx}(x) \frac{db_k^N}{dx}(x) \, dx , \quad k, l = 1, \ldots, N , \]

\[ \bar{\mu} = (\mu_l)_{l=1}^N \in \mathbb{R}^N , \quad \bar{\varphi} = (\varphi_k)_{k=1}^N \in \mathbb{R}^N . \]

A linear system of equations, cf. Rem. \[1.5.40\]!
The system matrix \( \mathbf{A} = (a_{ij}) \in \mathbb{R}^{M-1,M-1} \), \( a_{ij} := \int_a^b \frac{db^j_N(x)}{dx} \frac{db^i_N(x)}{dx} \, dx \), \( 1 \leq i, j \leq N \)

piecewise derivatives

r.h.s. vector \( \vec{\varphi} \in \mathbb{R}^{M-1} \), \( \varphi_k := \int_a^b g(x)b^k_N(x) \, dx \), \( k = 1, \ldots, N \).

The detailed computations start with the evident fact that

\[ |i - j| \geq 2 \implies \frac{db^j_N(x)}{dx} \cdot \frac{db^i_N(x)}{dx} = 0 \quad \forall x \in [a, b], \]

because there is no overlap of the supports of the two basis functions.

**Definition 1.5.57 (Support of a function).**

The support of a function \( f : \Omega \mapsto \mathbb{R} \) is defined as

\[ \text{supp}(f) := \{ \mathbf{x} \in \Omega : f(\mathbf{x}) \neq 0 \}. \]

In addition, we use that the gradients of the tent functions are piecewise constant, see (1.5.55).
\[ \int_0^1 \frac{db_j^N}{dx}(x) \frac{db_i^N}{dx}(x) \, dx = \begin{cases} 
0, & \text{if } |i - j| \geq 2 \\
-\frac{1}{h_{i+1}}, & \text{if } j = i + 1 \\
-\frac{1}{h_i}, & \text{if } j = i - 1 \\
\frac{1}{h_i} + \frac{1}{h_{i+1}}, & \text{if } 1 \leq i = j \leq M - 1 
\end{cases} \]

\[ A \text{ symmetric, positive definite and tridiagonal:} \]

\[ A = \begin{pmatrix} 
\frac{1}{h_1} + \frac{1}{h_2} & -\frac{1}{h_2} & 0 & 0 \\
-\frac{1}{h_2} & \frac{1}{h_2} + \frac{1}{h_3} & -\frac{1}{h_3} & 0 \\
0 & \ldots & \ldots & \ldots \\
0 & 0 & \ldots & -\frac{1}{h_{M-1}} \\
0 & 0 & \ldots & \frac{1}{h_{M-1}} + \frac{1}{h_M} 
\end{pmatrix} \]

\[ (1.5.58) \]

\[ h_j := |x_j - x_{j-1}| \quad \text{local meshwidth, cell size} \]

Mandatory: computation of right hand side vector by numerical quadrature
Natural choice: piecewise polynomial trial/test spaces $\leftrightarrow$ composite quadrature rule

e.g, composite trapezoidal rule: $\varphi_k = \int_0^1 g(x)b_N^k(x) \, dx \approx \frac{1}{2}(h_k + h_{k+1})g(x_k), \quad 1 \leq k \leq N.$

(1.5.59)

For equidistant mesh with uniform cell size $h > 0$ we arrive at the linear system of equations:

$$
\begin{pmatrix}
2 & -1 & 0 \\
-1 & 2 & -1 \\
0 & \ddots & \ddots & \ddots \\
\end{pmatrix}
\begin{pmatrix}
\mu_1 \\
\vdots \\
\mu_N \\
\end{pmatrix}
= h
\begin{pmatrix}
g(x_1) \\
\vdots \\
g(x_N) \\
\end{pmatrix}
$$

(1.5.60)

\text{\textbf{2}} case: linear variational problem with variable stiffness, cf. (1.4.16)

$$
u \in C^1_{0,pw}([a, b]): \quad \int_a^b \sigma(x) \frac{d}{dx}(x) \frac{dv}{dx}(x) \, dx = \int_a^b g(x)v(x) \, dx \quad \forall v \in C^1_{0,pw}([a, b]).$$
Discrete variational problem with \( u_N = \mu_1 b_1^N + \cdots + \mu_N b_N^N \):

\[
\int_a^b \sigma(x) \sum_{l=1}^N \mu_l \frac{d b_l^N(x)}{dx} \frac{d b_k^N(x)}{dx} \, dx = \int_a^b g(x) b_k^N(x) \, dx \quad k = 1, \ldots, N .
\]

(1.5.61)

Here: numerical quadrature required for both integrals

Choice: composite midpoint rule for left hand side integral \( \rightarrow [18, \text{Sect. 10.3}] \)

\[
\int_a^b f(x) \, dx \approx \sum_{j=1}^M h_j f(m_j) , \quad m_j := \frac{1}{2}(x_j + x_{j-1}) .
\]

(1.5.62)

composite trapezoidal rule \([18, \text{Eq. 10.3.3}]\) for right hand side integral, see (1.5.59).

Assumption: \( \sigma \in C^0_{pw}([a, b]) \) with jumps only at grid nodes \( x_j \)

(1.5.61)
\[
\sum_{l=1}^{N} \left( \sum_{j=1}^{M} h_j \sigma(m_j) \frac{db_N^l}{dx}(m_j) \frac{db_N^k}{dx}(m_j) \right) \mu_l = \frac{1}{2} (h_{k+1} + h_k) g(x_k), \quad k = 1, \ldots, N,
\]

\[
\Rightarrow \quad A \bar{\mu} = \bar{\varphi}.
\]

Resulting linear system of equations equidistant mesh with uniform cell size \( h > 0 \)

\[
\begin{pmatrix}
\sigma_1 + \sigma_2 & -\sigma_2 & 0 \\
-\sigma_2 & \sigma_2 + \sigma_3 & -\sigma_3 \\
0 & \cdots & \cdots
\end{pmatrix}
\begin{pmatrix}
\frac{1}{h}
\end{pmatrix}
\begin{pmatrix}
\mu_1 \\
\vdots \\
\mu_N
\end{pmatrix}
\begin{pmatrix}
g(x_1) \\
\vdots \\
g(x_N)
\end{pmatrix}
\]

(1.5.63)

with \( \sigma_j = \sigma(m_j), \quad j = 1, \ldots, m. \)

Remark 1.5.64 (Offset function for finite element Galerkin discretization).
In the case of general boundary conditions

\[ u(a) = u_a, \quad u(b) = u_b \]

use piecewise linear offset function

\[
\begin{align*}
    u_0(x) &= \begin{cases} 
        u_a(1 - \frac{x-a}{h_1}), & \text{if } a \leq x \leq x_1, \\
        u_b(1 - \frac{b-x}{h_M}), & \text{if } x_{M-1} \leq x \leq b, \\
        0, & \text{elsewhere.}
    \end{cases}
\end{align*}
\]

Benefits of this choice of offset function:

- \( u_0 \) is a simple function (since p.w. linear),
- \( u_0 \) is locally supported: contributions from \( u_0 \) will alter only first and last component of right hand side vector.
**Example 1.5.66** (Linear finite element Galerkin discretization for elastic string model).

**Targeted:** *non-linear* variational equation on domain \( \Omega = [0, 1] \)

\[
\int_0^1 \frac{\kappa(\xi)}{L} \left( 1 - \frac{L}{\|u'(\xi)\|} \right) u'(\xi) \cdot v'(\xi) - f(\xi) \cdot v(\xi) \, d\xi = 0 \quad \forall v \in (C_0^{1,pw}([0, 1]))^2. \tag{1.3.7}
\]

- Data \( \kappa, f \) given in procedural form, see Rem. 1.5.3.
- **Trial space** \( V_{N,0} = (S_{1,0}^0(M))^2 \) on equidistant mesh \( M \), meshwidth \( h := \frac{1}{M} \).
- **Basis:** 1D tent functions, *lexikographic ordering*
  \[
  \mathcal{B} = \left\{ \left( \begin{array}{c} b_1^1 N \\ 0 \end{array} \right), \left( \begin{array}{c} b_2^2 N \\ 0 \end{array} \right), \ldots, \left( \begin{array}{c} b_{M-1}^{M-1} N \\ 0 \end{array} \right), \left( \begin{array}{c} 0 \\ b_1^1 N \end{array} \right), \left( \begin{array}{c} 0 \\ b_2^2 N \end{array} \right), \ldots, \left( \begin{array}{c} 0 \\ b_{M-1}^{M-1} N \end{array} \right) \right\}.
  \]
- Evaluation of right hand side by composite trapezoidal rule (1.5.59).
- Evaluation left hand side by composite midpoint rule (1.5.62).

**Preliminary consideration:** the derivative of

\[
\mathbf{u}_N := \mathbf{u}_0 + \mu_1 \left( \begin{array}{c} b_1^1 N \\ 0 \end{array} \right) + \cdots + \mu_{M-1} \left( \begin{array}{c} b_{M-1}^{M-1} N \\ 0 \end{array} \right) + \mu_M \left( \begin{array}{c} 0 \\ b_1^1 N \end{array} \right) + \cdots + \mu_{2M-2} \left( \begin{array}{c} 0 \\ b_{M-1}^{M-1} N \end{array} \right). \tag{1.5.67}
\]
with offset function according to Rem. 1.5.64 is piecewise constant on $\mathcal{M}$:

$$s_j(\vec{\mu}) := u'_N(\xi) = \frac{u(x_j) - u(x_{j-1})}{h}$$

in $]x_{j-1}, x_j[$:

$$s_j(\vec{\mu}) = \frac{1}{h} \begin{cases} 
\begin{pmatrix} 
\mu_j - \mu_{j-1} \\
\mu_{j+M-1} - \mu_{j+M-2} \\
\mu_1 - u(0) \\
u(1) - (\mu_{M-1}) \\
\end{pmatrix}, & \text{if } 2 \leq j \leq M - 1 , \\
(\mu_1 - u(0)), & \text{if } j = 1 , \\
u(1) - (\mu_{M-1}), & \text{if } j = M .
\end{cases}$$

Set:

$$r_j = r_j(\vec{\mu}) := \frac{1}{L} \kappa(m_j) \left( 1 - \frac{L}{L \cdot s_j(\vec{\mu})} \right)$$

Single row non-linear system of equations arising from Galerkin finite element discretization:

row 1: \[(r_1 + r_2)\mu_1 - r_2\mu_2 = hf_1(h) + r_1a , \]

row $j$: \[- r_j\mu_j + (r_j + r_{j+1})\mu_{j+1} - r_{j+1}\mu_{j+2} = f_1(jh) , \quad 2 \leq j < M - 1 , \]

row $M - 1$: \[- r_{M-1}\mu_{M-2} + (r_{M-1} + r_M)\mu_{M-1} = hf_1((M - 1)h) + r_Mb , \]

row $M$: \[(r_1 + r_2(\vec{\mu}))\mu_M - r_2\mu_{M+1} = hf_2(h) + r_1ua , \]

row $j$: \[- r_j\mu_{j+M-1} + (r_j + r_{j+1})\mu_{j+M} - r_{j+1}\mu_{j+M+1} = f_2(jh) , \quad 2 \leq j < M - 1 , \]

row $M - 1$: \[- r_{M-1}\mu_{2M-3} + (r_{M-1} + r_M)\mu_{2M-2} = hf_2((M - 1)h) + r_Mub . \]
Here the dependence $r_j = r_j(\vec{\mu})$ has been suppressed to simplify the notation.

Please study the derivation of (1.5.63) in order to understand how (1.5.69)-(1.5.74) arise.

These equations can be written in a more compact form:

\[(1.5.69)-(1.5.74) \iff \begin{pmatrix} R(\vec{\mu}) & 0 \\ 0 & R(\vec{\mu}) \end{pmatrix} \vec{\mu} = \begin{pmatrix} \vec{\varphi}_1 \\ \vec{\varphi}_2 \end{pmatrix}.\] (1.5.75)

with

\[
R(\vec{\mu}) := \begin{pmatrix}
  r_1 + r_2 & -r_2 & 0 & 0 \\
  -r_2 & r_2 + r_3 & -r_3 & 0 \\
  0 & \ldots & \ldots & 0 \\
  \ldots & \ldots & \ldots & 0 \\
  0 & -r_{M-2} & r_{M-2} + r_{M-1} & -r_{M-1} \\
  0 & 0 & -r_{M-1} & r_{M-1} + r_M
\end{pmatrix} \in \mathbb{R}^{M-1,M-1},
\]

\[
(\vec{\varphi}_1)_j := h f_1(h_j), \quad (\vec{\varphi}_2)_j := h f_2(h_j), \quad j = 1, \ldots, M-1.
\]

Iterative solution of (1.5.75) by fixed point iteration, see Ex. 1.5.47.
Initial guess \( \vec{\mu}^{(0)} \in \mathbb{R}^N; \quad k = 0; \)

repeat

\( k \leftarrow k + 1; \)

Solve the linear system of equations

\[
\begin{pmatrix}
R(\vec{\mu}^{(k-1)}) & 0 \\
0 & R(\vec{\mu}^{(k-1)})
\end{pmatrix}
\begin{pmatrix}
\vec{\mu}^{(k)} \\
\vec{\phi}_2
\end{pmatrix}
= \begin{pmatrix}
\vec{\phi}_1 \\
\vec{\phi}_2
\end{pmatrix};
\]

until \( \|\vec{\mu}^{(k)} - \vec{\mu}^{(k-1)}\| \leq \text{tol} \cdot \|\vec{\mu}^{(k)}\| \)

Code 1.5.76: Linear finite element discretization of elastic string variational problem

```matlab
function [vu,Jrec,figsol,figerg] = stringlinfem(kappa,f,L,u0,u1,M,tol)

% Solving the non-linear variational problem (1.3.7) for the elastic string by
% means of piecewise
% linear finite elements on an equidistant mesh with M - 1 interior nodes.
% kappa, f are handles of type @(xi) providing the coefficient function
% \kappa and the force field f. u0 and u1 pass the pinning points.
% M is the number of mesh cells, tol specifies the tolerance for the fixed
% point
% iteration. return value: 2 \times (M + 1)-matrix of node positions

if (nargin < 7), tol = 1E-2; end

h = 1/M; % meshwidth

phi = h*f(h*(1:M-1)); % Right hand side vector

% Initial guess: straight string, condition L > \|u(0) - u(1)\|.
if (L >= norm(u1-u0)), error('String must be tense'); end
```
vu_new = u0*(1-((0:1/M:1)))+u1*(0:1/M:1);
% Meaning of components of vu: vu(1,2:M) ↔ μ_{1,...,μ_{M-1}}, vu(2,2:M) ↔ μ_M,...,μ_{2M-2}.
figsol = figure; Jrec = []; hold on;
for k=1:100 % loop for fixed point iteration, maximum 100 iterations
  vu = vu_new;
  plot vu(1,:),vu(2,:),'-g'); drawnow;
  title(sprintf('M = %d, iteration #%d',M,k));
  xlabel('{\bf x_1}'); ylabel('{\bf x_2}');
  d = (vu(:,2:end) - vu(:,1:end-1))/h;
  s = sqrt(d(1,:).^2 + d(2,:).^2);
  r = kappa(h*((1:M)-0.5))*(1/L - 1./s)/h;
  Jel = h/(2*L)*kappa(h*((1:M)-0.5))*((s-L).^2)';
  Jf = - (phi(1,:)*vu(1,2:M)'+phi(2,:)*vu(2,2:M)');
  Jrec = [Jrec; k, Jel, Jf, Jel+Jf];
  phi1 = phi(1,:); phi1(1) = phi1(1) + r(1)*u0(1); phi1(M-1) = phi1(M-1) + r(M)*u1(1);
  phi2 = phi(2,:); phi2(1) = phi2(1) + r(1)*u0(2); phi2(M-1) = phi2(M-1) + r(M)*u1(2);
% Solve linear system and compute new iterate
vu_new = [u0, [(R\phi1')';(R\phi2')'],u1];
% Check simple termination criterion for fixed point iteration.
if (norm(vu_new - vu,'fro') < tol*norm(vu_new,'fro')/M)
    plot(vu(1,:),vu(2,:),'r-'); break; end
end
%
% Plot of total potential energy in the course of the iteration
figerg = figure('name','total potential energy');
title(sprintf('elastic string, M = %d',M));
plot(Jrec(:,1),Jrec(:,4),'m-','...');
    Jrec(:,1),Jrec(:,2),'b-+','...';
    Jrec(:,1),Jrec(:,3),'g-^');
xlabel('
\bf no. of iteration step}'); ylabel('energy');
legend('total potential energy','elastic energy','energy in force field','location','east');

Example 1.5.77 (Elastic string shape by finite element discretization).

- Linear finite element discretization of (1.3.7), see Ex. 1.5.66, Code 1.5.75.
- $\kappa \equiv 1$, $L = 0.5$, $u(0) = (00)^T$, $u(1) = (10.2)^T$
gravitational force field \( f(\xi) = -\binom{0}{2} \).

Piecewise linear finite element solution of (1.3.7), equidistant meshes with \( M \) cells, \( M = 5, 10, 20 \).

Convergence of fixed point iteration (\( M = 20 \)): 
1.5.2 Collocation

Targetted: Two-point BVP = ODE $\mathcal{L}(u) = f$  +  boundary conditions
Note: In contrast to the Galerkin approach, collocation techniques do not tackle the weak form of a boundary value problem, but rather the “classical”/strong form.

Idea:
1. seek solution in finite-dimensional trial space \( V_{N,0} \), \( N := \dim V_{N,0} < \infty \)
2. pick collocation nodes \( \mathcal{N} := \{x_1, \ldots, x_N\} \subset \Omega \) such that \( x \)

\[
\begin{align*}
\text{“point evaluation”} & \quad \begin{cases} 
V_{N,0} \mapsto \mathbb{R}^N \\
v \mapsto (v(x_j))_{j=1}^N
\end{cases} 
\end{align*}
\]  

(1.5.78)

is a bijective linear mapping.

Collocation conditions: \( u_N \in V_N: \quad \mathcal{L}(u_N)(x_j) = f(x_j), \quad j = 1, \ldots, N \).

(1.5.79)

3. choose ordered basis \( \mathfrak{B} = \{b_1^N, \ldots, b_N^N\} \) of \( V_{N,0} \) & plug basis representation

\[
u_N = u_0 + \mu_1 b_1^N + \cdots + \mu_N b_N^N \quad (u_0 \text{ is offset function, cf. Rem. 1.5.10})
\]

into collocation conditions (1.5.79)

\[
\tilde{\mu} = (\mu_i)_{i=1}^N: \quad \mathcal{L}(u_0 + \mu_1 b_1^N + \cdots + \mu_N b_N^N)(x_j) = f(x_j), \quad j = 1, \ldots, N.
\]  

(1.5.80)
In general: (1.5.80) is a non-linear system of equation \((N\) equations for \(N\) unknowns \(\mu_1, \ldots, \mu_N\)).

Note: bijectivity of point evaluation (1.5.78) \(\Rightarrow\) \(#\{\text{nodes}\} = \dim V_{N,0}\).

Below: detailed discussion for \textit{linear} two point boundary value problem

\[
\mathcal{L}(u) := -\frac{d}{dx} \left( \sigma(x) \frac{du}{dx}(x) \right) = g(x) , \quad a \leq x \leq b ,
\]

(1.5.81) on domain \(\Omega = [a, b]\), related to variational problem (1.4.16).
Remark 1.5.82 (Smoothness requirements for collocation trial space).

For two-point BVP (1.5.81) consider space \( V_{N,0} := S_{1,0}^0(\mathcal{M}) \) of \( \mathcal{M} \)-piecewise linear finite element functions. \( \rightarrow \) Sect. 1.5.1.2

Note: \( v_N \in S_{1,0}^0(\mathcal{M}) \) is not differentiable in nodes \( x_j \) of the mesh.

- Natural choice collocation points = nodes of the mesh is not possible!
- (because for \( v_N \in S_{1,0}^0(\mathcal{M}) \) the function \( \mathcal{L}(v_N) \) is discontinuous in the nodes of the mesh)

Assuming \( \sigma \in C^1([a,b]) \) global continuity of \( \mathcal{L}(v_N) \) entails \( V_{N,0} \subset C^2([a,b]) \), cf. Sect. 1.5.2.2.

Remark 1.5.83 (Collocation: smoothness requirements for coefficients).

For 2-point BVP (1.5.81): \( \sigma \) must be differentiable in collocation nodes, with known values \( \frac{d\sigma}{dx}(x_j) \), \( j = 1, \ldots, N \), in the sense of Rem. 1.5.3: extra difficulty when \( \sigma \) given in procedural form.
1.5.2.1 Spectral collocation

Focus: *linear* two point boundary value problem (1.5.81)

**trial space** for polynomial spectral collocation:

\[ V_{N,0} = \mathcal{P}_p(\mathbb{R}) \cap C^2_0([a, b]) , \quad p \geq 2. \]  

(1.5.84)

= polynomials of degree \(\leq p\), vanishing at endpoints of domain, \( N := \dim V_{N,0} = p - 1 \).

> same trial space as for polynomial spectral Galerkin approach, see Sect. 1.5.1.1.

Discussion: polynomial spectral collocation for two-point BVP (1.5.81)

**offset function** \( u_0(x) := \frac{b-x}{b-a} u_a + \frac{x-a}{b-a} u_b \).
Basis \( \mathcal{B} := \{ b_j^N := M_j \} \) consisting of integrated Legendre polynomials, see (1.5.24).

Note: \( \mathcal{L} \) from (1.5.81) is a linear differential operator!

Terminology: A differential operator is a mapping on a function space involving only values of the function argument and some of its derivatives in the same point.

A differential operator \( \mathcal{L} \) is linear, if

\[
\mathcal{L}(\alpha u + \beta v) = \alpha \mathcal{L}(u) + \beta \mathcal{L}(v) \quad \forall \alpha, \beta \in \mathbb{R}, \forall \text{functions } u, v
\]  

(1.5.85)

(1.5.80) \[ \sum_{l=1}^{N} \mathcal{L}(b_N^l)(x_k)\mu_l = f(x_k) - \mathcal{L}(u_0)(x_k) , \quad k = 1, \ldots, N . \]  

(1.5.86)

\( A \vec{\mu} = \vec{\varphi} \), \hspace{1cm} \( A_{k,l} := \mathcal{L}(b_N^l)(x_k) , \quad k, l \in \{1, \ldots, N\} \), \hspace{1cm} \( \varphi_k := f(x_k) - \mathcal{L}(u_0)(x_k) , \quad k \in \{1, \ldots, N\} \).  

(1.5.87)

An \( N \times N \) linear system of equations
For BVPs featuring \textit{linear} differential operators, collocation invariably leads to a \textit{linear} system of equations for the unknown coefficients of the basis representation of the collocation solution.

\textit{Remark} 1.5.88 (Bases for polynomial polynomial spectral collocation).

Same choices as for spectral Galerkin methods, see Rem. 1.5.22.

\textit{Remark} 1.5.89 (Collocation points for polynomial spectral collocation).

Rule of thumb (without further explanation, see [16]):

choose \textcolor{blue}{collocation points} \( x_j, \ j = 1, \ldots, N \) such that the induced Lagrangian interpolation operator (\( \rightarrow [18, \text{Thm. 3.3.8}] \)) has a small \( \infty \)-norm, see [18, Lemma 3.3.17].

Popular choice (due to [18, Eq. 9.2.15]): \textcolor{blue}{Chebychev nodes}

\[
x_k := a + \frac{1}{2}(b - a) \left( \cos\left(\frac{2k - 1}{2N} \pi\right) + 1\right), \quad k = 1, \ldots, N.
\] (1.5.90)
Code 1.5.91: Computation of derivatives of Legendre polynomials using (1.5.30)

```matlab
function [V,M,D] = dilegpol(n,x)
% Computes values of the first $n+1$ Legendre polynomials (returned in matrix V)
% the first $n-1$ integrated Legendre polynomials (returned in matrix M), and the
% first $n+1$ first derivatives of Legendre polynomials in the points $x_j$ passed
% in the row vector x.
% Uses the recursion formulas (1.5.28) and (1.5.24)
V = ones(size(x)); V = [V; x];
% recursion (1.5.28) for Legendre polynomials
for j=1:n-1, V = [V; ((2*j+1)/(j+1)). * x.*V(end,:)) - j/(j+1)*V(end-1,:)]; end
% Formula (1.5.24) for integrated Legendre polynomials
M = diag(1./((2*(1:n-1)+1)) .* (V(3:n+1,:) - V(1:n-1,:));
% Recursion formula (1.5.30) for derivatives of Legendre polynomials
if (nargout < 3)
D = [zeros(size(x)); ones(size(x))];
for j=1:n-1, D = [D;((2*j+1)*V(j+1,:)+D(j,:))]; end
end
```

Code 1.5.92: Spectral collocation for linear 2nd-order two-point BVP

```matlab
function u = linspeccol(g,N,x)
% Polynomial spectral collocation discretization of linear 2nd-order two-point BVP
% $-\frac{d^2u}{dx^2} = g(x), \ u(0) = u(1) = 0$
% on $\Omega = [0,1]$. Trial space of dimension $N$, collocation in Chebychev nodes.
```
Values of approximate solution in points $x_j$ are returned in the row vector $u$

```matlab
% Chebychev nodes, see (1.5.90)

\[ cn = \cos((2*(1:N)-1) \times \frac{\pi}{(2*N)}); \]

[V,M,D] = dilegpol(N+1,cn); % Obtain values of (2nd derivatives) of $M_m$

% Solve collocation system
\[ mu = (-4*D(2:N+1,:))' \backslash (g(0.5*(cn+1))'); \]

% Compute values of integrated Legendre polynomials at output points
\[ [V,M] = dilegpol(N+1,2*x-1); u = mu' * M; \]
```

**Example 1.5.93** (Polynomial spectral collocation for 2-point BVP).

Setting of Ex. 1.5.20, spectral polynomial collocation, on $N = 5, 7, 10$, basis from integrated Legendre polynomials, plot of solution $u_N$. 

1.5.2.2 Spline collocation
Analogous to Sect. 1.5.1.2: now collocation based on piecewise polynomials

Rem. 1.5.82 ➢ for BVP (1.5.81) smoothness $V_{N,0} \subset C^2([a, b])$ is required.

Which piecewise polynomial spaces offer this kind of smoothness?

Recall [18, Def. 3.7.1], cf. [18, Sect. 3.7.1]:

**Definition 1.5.94** (Cubic spline).
$s : [a, b] \mapsto \mathbb{R}$ is a cubic spline function w.r.t. the node set $\mathcal{T} := \{a = x_0 < x_1 < x_2 < \ldots < x_{M-1} < x_M = b\}$, if

(i) $s \in C^2([a, b])$ (twice continuously differentiable),

(ii) $s|_{x_{j-1},x_j} \in \mathcal{P}_3(\mathbb{R})$ (piecewise cubic polynomial)

notation: $S_{3,\mathcal{T}} \triangleq$ vector space of cubic splines on node set $\mathcal{T}$
Known:
\[ \dim \mathcal{S}_{3,T} = \#T + 2 = M + 3 \]

Trial space for collocation for 2-point BVP (1.5.81)

natural cubic splines: \( V_{N,0} := \left\{ s \in \mathcal{S}_{3,T} : \begin{array}{l} s''(a) = s''(b) = 0, \\ s(a) = s(b) = 0 \end{array} \right\} \Rightarrow \dim N := V_N = M - 1 \)

Choice of collocation nodes:

collocation nodes for cubic spline collocation = spline nodes \( x_j : \mathcal{N} = \mathcal{T} \)

Example 1.5.95 (Cubic spline collocation discretization of 2-point BVP).
Setting of Ex. 1.5.20

Cubic spline collocation with equidistant nodes, \( M = 5, 7, 12 \)

Solution \( u_N \)

![Graph showing solution with different values of M](image)

1.5.3 Finite differences

Focus: 2nd-order linear two-point BVP

\[
\mathcal{L}(u) := -\frac{d}{dx} \left( \sigma(x) \frac{du}{dx}(x) \right) = g(x), \quad a \leq x \leq b, \tag{1.5.81}
\]
$u(a) = u_a$, $u(b) = u_b$,

Idea:

Replace derivatives $\longrightarrow$ difference quotients (in finitely many special points = nodes of a mesh)

E.g. $\frac{d^2u}{dx^2}(x) \approx \frac{u(x + h) - 2u(x) + u(x - h)}{h^2}$, $h > 0$ “small”. (1.5.96)

Setting as in Sect. 1.5.1.2:

$\Omega = [a, b]$ equipped with nodes $(M \in \mathbb{N})$

$\mathcal{X} := \{a = x_0 < x_1 < \cdots < x_{M-1} < x_M = b\}$.

Special case:

Equidistant mesh: $x_j := a + jh$, $h := \frac{b-a}{M}$.

$[x_{j-1}, x_j], j = 1, \ldots, M$, $\equiv$ cells of $\mathcal{M}$, cell size $h_j := |x_j - x_{j-1}|$, $j = 1, \ldots, M$.

Meshwidth $h_{\mathcal{M}} := \max_j |x_j - x_{j-1}|$.
replacement of outer derivative \((x_{j-1/2} = \frac{1}{2}(x_j + x_{j-1})):\)

\[
\frac{d}{dx} \left( \sigma(x) \frac{du}{dx}(x) \right) \bigg|_{x=x_j} \approx \frac{2}{h_{j-1} + h_j} \left( \sigma(x_{j+1/2}) \frac{du}{dx}(x_{j+1/2}) - \sigma(x_{j-1/2}) \frac{du}{dx}(x_{j-1/2}) \right)
\]

replacement of inner derivative, e.g.,

\[
\frac{du}{dx}(x_{j+1/2}) \approx \frac{u(x_{j+1}) - u(x_j)}{h_j}
\]

\[
- \frac{d}{dx} \left( \sigma(x) \frac{du}{dx}(x) \right) \bigg|_{x=x_j} = \frac{\sigma(x_{j-1/2}) \frac{u(x_j) - u(x_{j-1})}{h_{j-1}} - \sigma(x_{j+1/2}) \frac{u(x_{j+1}) - u(x_j)}{h_j}}{\frac{1}{2}(h_{j-1} + h_j)}.
\]

On equidistant mesh, uniform meshwidth \(h_j = h > 0, j = 1, \ldots, M:\)

\[
- \frac{d}{dx} \left( \sigma(x) \frac{du}{dx}(x) \right) \bigg|_{x=x_j} = \frac{1}{h^2} \left( -\sigma(x_{j+1/2})u(x_{j+1}) + (\sigma(x_{j+1/2}) + \sigma(x_{j-1/2}))u(x_j) - \sigma(x_{j-1/2})u(x_{j-1}) \right).
\]

Unknowns in finite difference method:

\[
\mu_l = u(x_l), \quad l = 1, \ldots, M - 1
\]
\[- \frac{d}{dx} \left( \sigma(x) \frac{du}{dx}(x) \right) = g(x), \ a \leq x \leq b. \]

\[ -\sigma(x_{j+1/2})\mu_{j+1} + (\sigma(x_{j+1/2}) + \sigma(x_{j-1/2}))\mu_j - \sigma(x_{j-1/2})\mu_{j-1} = g(x_j), \ j = 1, \ldots, M - 1. \]  

\[ (A)_{jl} = h^{-2} \begin{cases} 
0, & \text{if } |j - l| > 1, \\
-\sigma(x_{j+1/2}), & \text{if } j = l - 1, \\
\sigma(x_{j-1/2}) + \sigma(x_{j-1/2}), & \text{if } j = l, \\
-\sigma(x_{l+1/2}), & \text{if } l = j - 1.
\end{cases} \]  

An \((M - 1) \times (M - 1)\) linear system of equations.
(Up to scaling with $h$) the finite difference approach and the linear finite element Galerkin scheme (→ Sect. 1.5.1.2) yield the same system matrix for the BVP (1.5.81) and its associated variational problem (1.4.16), cf. (1.5.100) and (1.5.63).

### 1.6 Convergence

For elastic string model (1.2.17)/(1.3.7), taut string model in graph description (1.4.16) with exact solution $u : [0, 1] \mapsto \mathbb{R}^2$ or $u : [a, b] \mapsto \mathbb{R}$, respectively:

<table>
<thead>
<tr>
<th>Discretization schemes</th>
<th>Approximate solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Galerkin approach, Sect. 1.5.1)</td>
<td>$u_N : [0, 1] \mapsto \mathbb{R}^2$</td>
</tr>
<tr>
<td>collocation methods, Sect. 1.5.2)</td>
<td>$u_N : [a, b] \mapsto \mathbb{R}$</td>
</tr>
</tbody>
</table>

(functions $\in V_N$)

Desirable: approximation $u_N$ “close to” exact solution $u$: rigorous meaning? 

How to measure discretization error $u - u_N$?
Remark 1.6.1 (Grid functions).

Note: for finite differences (→ Sect. 1.5.3) we get no solution function, only grid function $\mathcal{X} \mapsto \mathbb{R}$ ("point values")

reconstruction of a function through postprocessing, e.g., linear interpolation

Remark 1.6.2.

We encountered the issues of convergence of approximate solutions before:

- Numerical quadrature [18, Ch. 10]: study of asymptotic behavior of quadrature error
- Numerical integration [18, Ch. 12]: discretization error of single step methods
1.6.1 Norms on function spaces

Tools for measuring discretization errors: norms on function spaces/grid function spaces

Reminder → [18, Sect. 2.5.1]

Definition 1.6.3 (Norm).

A norm $\| \cdot \|_V$ on an $\mathbb{R}$-vector space $V$ is a mapping $\| \cdot \|_V : V \mapsto \mathbb{R}^+_0$, such that

1. (definiteness) $\| v \|_V = 0 \iff v = 0 \quad \forall v \in V$ (N1)
2. (homogeneity) $\| \lambda v \|_V = |\lambda| \| v \|_V \quad \forall \lambda \in \mathbb{R}, \forall v \in V$ (N2)
3. (triangle inequality) $\| w + v \|_V \leq \| w \|_V + \| v \|_V \quad \forall w, v \in V$ (N3)
Definition 1.6.4 (Supremum norm).

The supremum norm of an (essentially) bounded function \( u : \Omega \mapsto \mathbb{R}^n \) is defined as

\[
\| u \|_{\infty} = \| u \|_{L^\infty(\Omega)} := \sup_{x \in \Omega} \| u(x) \|, \quad u \in (L^\infty(\Omega))^n.
\] (1.6.5)

- \( L^\infty(\Omega) \) denotes the vector space of essentially bounded functions. It is the instance for \( p = \infty \) of an \( L^p \)-space.

- The notation \( \| \cdot \|_{\infty} \) hints at the relationship between the supremum norm of functions and the maximum norm for vectors in \( \mathbb{R}^n \).

- For \( n = 1 \) the Euclidean vector norm in the definition reduces to the modulus \( |u(x)| \).

- The norm \( \| u - u_N \|_{L^\infty(\Omega)} \) measures the maximum distance of the function values of \( u \) and \( u_N \).
Definition 1.6.6 (Mean square norm/\(L^2\)-norm).

For a function \(u \in (C^0_{pw}(\Omega))^n\) the mean square norm/\(L^2\)-norm is given by

\[
\|u\|_0 \left( \|u\|_{L^2(\Omega)} \right) := \left( \int_{\Omega} \|u(x)\|^2 \, dx \right)^{1/2}, \quad u \in (L^2(\Omega))^n.
\]

\(L^2(\Omega)\) designates the vector space of square integrable functions, another \(L^p\)-space (for \(p = 2\)) and a Hilbert space.

The “0” in the notation \(\|\cdot\|_0\) refers to the absence of derivatives in the definition of the norm.

Obviously, the \(L^2\)-norm is weaker than the supremum norm:

\[
\|v\|_{L^2([a,b])} \leq \sqrt{b - a} \|v\|_{L^\infty([a,b])} \quad \forall v \in C^0_{pw}([a, b]).
\]

In particular, the \(L^2\)-norm of the discretization error may be small despite large deviations of \(u_N\) from \(u\), provided that these deviations are very much localized.
**Relevant error norms** suggested by application context/physics!

**Remark 1.6.7 (Energy norm).**

We consider the model for a homogeneous taut string in physical space, see (1.4.16), with associated total potential energy functional

\[ J(u) := \int_a^b \frac{1}{2} \left( \frac{du}{dx} (x) \right)^2 + \hat{g}(x) u(x) \, dx , \quad u \in C^1_{0,pw}([a,b]) , \tag{1.6.8} \]

where, for the sake of simplicity, we assume \( u_a = u_b = 0 \).

A manifestly relevant error quantity of interest is the deviation of energies

\[ E_J := |J(u) - J(u_N)| . \]

We adopt the concise notations introduced for abstract (linear) variational problems in Rems. 1.3.12, 1.4.5:

\[ a(u,v) := \int_a^b \frac{du}{dx} (x) \frac{dv}{dx} (x) \, dx , \]

\[ J(u) = \frac{1}{2} a(u,u) - \ell(u) , \]

\[ \ell(u) := \int_a^b \hat{g}(x) u(x) \, dx . \]
where \(a\) is a bilinear form, see Def. 1.3.13.

Assumption: \(u_N \in V_{N,0} \doteq \text{Galerkin solution based on discrete trial space } V_{N,0} \subset V_0.\)

\[
\begin{align*}
a(u,v) = \ell(v) & \quad \forall v \in V_0 := C^1_{0,pw}([a,b]), \\
a(u_N,v_N) = \ell(v_N) & \quad \forall v_N \in V_{N,0} \subset V_0.
\end{align*}
\]

We can use the defining variational equations for \(u\) and \(u_N\) to express

\[
J(u) - J(u_N) = -\frac{1}{2}(a(u,u) - a(u_N,u_N)) \overset{(*)}{=} -\frac{1}{2}a(u + u_N, u - u_N).
\]

\((*)\): a straightforward consequence of the bilinearity of \(a\), see Def. 1.3.13, c.f. \(a^2 - b^2 = (a+b)(a-b)\) for \(a, b \in \mathbb{R}\).

Concretely,

\[
|J(u) - J(u_N)| = \frac{1}{2} \left| \int_a^b \frac{d}{dx}(u + u_N) \cdot \frac{d}{dx}(u - u_N) \, dx \right| \overset{(*)}{\leq} \frac{1}{2} \left( \int_a^b \left| \frac{d}{dx}(u + u_N) \right|^2 \, dx \right)^{1/2} \left( \int_a^b \left| \frac{d}{dx}(u - u_N) \right|^2 \, dx \right)^{1/2}.
\]

\((*)\): due to Cauchy-Schwarz inequality (2.2.15)
Definition 1.6.12 ($H^1$-seminorm). \(\rightarrow\) Def. 2.2.12

For a function \(u \in C^1_{pw}([a, b])\) the \(H^1\)-seminorm reads

\[
|u|_{H^1([a,b])}^2 := \int_a^b \left| \frac{du}{dx}(x) \right|^2 \, dx.
\] (1.6.13)

\(\cdot\) is merely a semi-norm, because it only satisfies norm axioms (N2) and (N3), but fails to be definite: \(|\cdot|_{H^1([a,b])} = 0\) for constant functions.

In the setting of the homogeneous taut string model, we have

\[
|u|_{H^1([a,b])}^2 = a(u, u) \quad \Rightarrow \quad |\cdot|_{H^1([a,b])} \text{ is called the energy norm for the model.}
\]

More explanations in Sect. 2.1.3.

On \(C^1_{0,pw}([a, b])\) the semi-norm \(|\cdot|_{H^1([a,b])}\) is a genuine norm \(\rightarrow\) Def. 1.6.3. See proof of Thm. 2.2.16.
From (1.6.11)

\[ \| u - u_N \|_{H^1(\Omega)} \leq \epsilon \implies |J(u) - J(u_N)| \leq |u + u_N|_{H^1(\Omega)} |u - u_N|_{H^1(\Omega)} \]

(1.6.14)

\[ \leq (2|u|_{H^1(\Omega)} + \epsilon) \epsilon. \]

estimate of the energy norm of the discretization error paves the way for bounding the energy deviation.

Remark 1.6.15 (Norms on grid function spaces).

To measure the discretization error for finite difference schemes (→ Sect. 1.5.3) one may resort to mesh dependent norms

\[ \text{(discrete) } l^2\text{-norm} : \quad \| \bar{\mu} \|_{l^2(\mathcal{X})}^2 := \sum_{j=0}^{M} \frac{1}{2}(h_j + h_{j+1})|\mu_j|^2, \quad (1.6.16) \]
(under convention $h_0 := 0$, $h_{M+1} := 0$),

(discrete) maximum norm  :  $\|\vec{\mu}\|_{l_{\infty}(\mathcal{X})} := \max_{j=0,\ldots,M} |\mu_j|$ .

(1.6.17)

\textbf{Remark 1.6.18} (Approximate computation of norms).

Standard \textit{testing} of implementations of numerical methods for 2-point BVP: Examine norm of discretization error for test cases with (analytically) known exact solution $u$.

Even for numerical methods computing $u_N \in V_N \subset V$ (Galerkin methods $\rightarrow$ Sect. 1.5.1, collocation methods $\rightarrow$ Sect. 1.5.2):

usually exact computation of $\|u - u_N\|$ is impossible/very difficult.

Option: approximate evaluation of norm $\|u - u_N\|$
supremum norm $\|\cdot\|_\infty$: approximation by sampling on discrete point set.

$L^2$-norm, energy norm: numerical quadrature

(Gauss quadrature for spectral schemes, composite quadrature for mesh based schemes)

Error introduced by approximation of norm must be smaller than discretization error

(⇒ use “overkill” quadrature/sampling, cost does not matter much in testing).

1.6.2 Algebraic and exponential convergence

Crucial: convergence is an asymptotic notion!

sequence of discrete models $\Rightarrow$ sequence of approximate solutions $(u_N^{(i)})_{i \in \mathbb{N}}$

$\Rightarrow$ study sequence $(\|u_N^{(i)} - u\|)_{i \in \mathbb{N}}$

created by variation of a discretization parameter:
Discretization parameters:

- **meshwidth** $h > 0$ for finite differences (→ Sect. 1.5.3), p.w. linear finite elements (→ Sect. 1.5.1.2), spline collocation (→ Sect. 1.5.2.2)
- **polynomial degree** for spectral collocation (→ Sect. 1.5.2.1), spectral Galerkin discretization (→ Sect. 1.5.1.1)

**Example 1.6.19** (Numerical studies of convergence).

Focus: Linear 2-point boundary value problem $-\frac{d^2u}{dx^2} = g(x)$, $u(0) = u(1) = 0$ on $\Omega = ]0, 1[$.

Variational form (1.5.21),

exact solution $u(x) = \sin(2\pi x^2)$ (→ setting of Ex. 1.5.20)

① finite difference discretization on equidistant mesh, meshwidth $h > 0$ (→ Sect. 1.5.3)
What is plotted are the discrete versions of the $L^2$-norm and supremum norm, see Rem. 1.6.15.

The energy norm of the error was computed according to the formula

$$\text{energy norm}(\text{error})^2 := \sum_{j=1}^{M} h_j \left| \frac{\mu_j - \mu_{j-1}}{h_j} - \frac{du}{dx}(x_{j-1/2}) \right|^2.$$ 

Spectral collocation, polynomial degree $p \in \mathbb{N}$ → Sect. 1.5.2.1
Monitored: supremum norm (1.6.5), $L^2$-norm (1.6.6) of discretization error $u - u_N$ (approximated by “overkill” Gaussian quadrature with $10^4$ nodes)

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{fig36.png}
\caption{Spectral collocation, Chebychev points}
\end{figure}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{fig37.png}
\caption{Spectral collocation, Chebychev points}
\end{figure}

Spline collocation on equidistant mesh, meshwidth $h > 0$ (→ Sect. 1.5.2.2)

Monitored: supremum norm (1.6.5), $L^2$-norm (1.6.6) of $u - u_N$ (approximated by sampling on fine grid with $10^4$ points)
Spectral Galerkin based on degree $p \in \mathbb{N}$ polynomials  \(\rightarrow\) Sect. [1.5.1.1]

Monitored: supremum norm (1.6.5), $L^2$-norm (1.6.6) of discretization error $u - u_N$ (approximated by trapezoidal rule on fine grid with $10^4$ points)
Observation:

- ‘Empiric convergence” in all cases
- different qualitative behavior (of norm of discretization error)

How to compare different discretizations?
Unified view: Study $\|u - u_N\|$ as function of number $N$ of unknowns (degrees of freedom) measure for costs incurred by method

Definition 1.6.20 (Convergence rate). → [18, Sect. 9.1], [18, Eq. 9.1.3]

\[
\|u - u_N\| = O(N^{-\alpha}), \alpha > 0 \quad \iff \quad \text{algebraic convergence with rate } \alpha
\]

\[
\|u - u_N\| = O(\exp(-\gamma N^\delta)), \text{ with } \gamma, \delta > 0 \quad \iff \quad \text{exponential convergence}
\]

\[\exists N_0 > 0, \exists C > 0 \text{ independent of } N \quad \text{such that } |f(N)| \leq Cg(N) \text{ for } N > N_0. \quad (1.6.21)\]
Linear plot of qualitative convergence behavior: algebraic/exponential convergence rates

Exponential convergence will always win (asymptotically)
Log-linear plot of decrease of discretization error for algebraic/exponential convergence rates
Log-log plot of decrease of discretization error for algebraic/exponential convergence rates
Remark 1.6.22 (Exploring convergence experimentally). → [18, Rem. 9.1.4]

How to determine qualitative asymptotic convergence from raw norms of discretization error?

Given: data tuples \((N_i, \epsilon_i), i = 1, 2, 3, \ldots\), \(N_i \doteq \) problem sizes, \(\epsilon_i \doteq \) error norms

1. Conjecture: \textbf{algebraic convergence}: \(\epsilon_i \approx CN_i^{-\alpha}\)

\[\log(\epsilon_i) \approx \log(C) - \alpha \log N_i\]  
(affine linear in log-log scale).

➤ linear regression on data \((\log N_i, \log \epsilon_i), i = 1, 2, 3, \ldots\) to determine rate \(\alpha\).

2. Conjecture: \textbf{exponential convergence}: \(\epsilon_i \approx C \exp(-\gamma N_i^\delta)\)

\[\log \epsilon_i \approx \log(C) - \gamma N_i^\delta.\]

➤ non-linear least squares fit (→ [18, Sect. 7.5]) to determine \(\delta\):

\[(c, \gamma, \delta) = \text{argmin} \left\{ \sum_i |\log \epsilon_i - c + \gamma N_i^\delta|^2 \right\},\]

residual ↔ validity of conjecture. This can be done by a short MATLAB code (→ exercise)
Example 1.6.23 (Asymptotic nature of convergence).

\[ -\frac{d^2 u}{dx^2} = g(x), \quad u(0) = u(1) = 0, \]
\[ \Omega = ]0, 1[, \]
\[ u(x) = \sin(50\pi x^2) \]

1. finite difference discretization on equidistant mesh, meshwidth \( h > 0 \) (→ Sect. 1.5.3)

2. Spectral Galerkin based on degree \( p \in \mathbb{N} \) polynomials → Sect. 1.5.1.1

Evaluations as in Ex. 1.6.19
Observation: for $h \to 0$, $p \to \infty$, algebraic convergence of the finite difference solution, and exponential convergence of the spectral Galerkin solution emerge. This is the “typical” asymptotic behavior of the discretization error norms for these discretization methods.

However, the onset of asymptotic convergence occurs only for rather small meshwidth or large $p$, respectively, beyond thresholds that may never be reached in a computation. During a long pre-asymptotic phase the error is hardly reduced when increasing the resolution of the discretization.
Example 1.6.24 (Convergence and smoothness of solution).

\[ \Omega = ]0, 1[ \text{ (for finite differences)}, \quad \Omega = ] - 1, 1[ \text{ (for spectral Galerkin)}, \]

exact solution of 2-point BVP for ODE
\[ -\frac{d^2 u}{dx^2} = g(x), \]

\[ u(x) = \begin{cases} 
\frac{3}{4} - x^2, & \text{if } |x| < \frac{1}{2}, \\
1 - |x|, & \text{if } |x| \geq \frac{1}{2}.
\end{cases} \quad \leftrightarrow \quad g(x) = \begin{cases} 
2, & \text{if } |x| < \frac{1}{2}, \\
0, & \text{elsewhere}.
\end{cases} \]
Observations:

- no more exponential convergence of spectral Galerkin
- FD: different rate of algebraic convergence for even/odd $M$

Qualitative asymptotic convergence also depends on data!
Preface

The previous chapter discussed the transformation of a minimization problem on a function space via a variational problem to a differential equation. To begin with, in Sect. 2.1–Sect. 2.4, this chapter revisits this theme for models that naturally rely on function spaces over domains in two and three spatial dimensions. Thus the transformation leads to genuine partial differential equations.

Sect. 2.2 ventures into the realm of Sobolev spaces, which provide the framework for rigorous mathematical investigation of variational equations. However, we will approach Sobolev spaces as “spaces
of physically meaningful solutions” or “spaces of solutions with finite energy”. From this perspective dealing with Sobolev spaces will be reduced to dealing with their norms.

In Sect. 2.5, we change tack and consider a physical phenomenon (heat conduction) where modelling naturally leads to partial differential equations. On this occasion, we embark on a general discussion of boundary conditions in Sect. 2.6.

Then the fundamental class of second-order elliptic boundary value problems is introduced. Appealing to “intuitive knowledge” about the physical systems underlying the models, key properties of their solutions are presented in Sect. 2.7.

In Sect. 2.6 in the context of stationary heat conduction we introduce the whole range of standard boundary conditions for 2nd-order elliptic boundary value problems. Their discussion in variational context will be resumed in Sect. 2.9.

Supplementary and further reading:

R. Hiptmair
C. Schwab, H. Harbrecht
V. Gradinaru
A. Chernov
SAM, ETHZ

Numerical Methods for PDEs

p. 166
An excellent *mathematical* introduction to partial differential equations is Evans’ book [12]. Chapter 2 gives a very good idea about fundamental properties of various simple PDEs. Chapters 6 and 7 fit the scope of this course chapter, but go way beyond it in terms of mathematical depth.

**Remark 2.0.1 (Boundary value problems (BVPs)).**

The traditional concept of a boundary value problem for a partial differential equation:

---

**Boundary value problem** (BVP)

Given a partial differential operator $\mathcal{L}$, a domain $\Omega \subset \mathbb{R}^d$, a boundary differential operator $\mathcal{B}$, boundary data $g$, and a source term $f$, seek a function $u : \Omega \mapsto \mathbb{R}^n$ such that

\[
\mathcal{L}(u) = f \quad \text{in} \quad \Omega ,
\]

\[
\mathcal{B}(u) = g \quad \text{on part of (or all) boundary } \partial \Omega .
\]
Terminology: boundary value problem is **scalar** \( \iff n = 1 \) (in this case the unknown is a real valued function)

What does elliptic mean?

Mathematical theory of PDEs distinguishes three main classes of boundary value problems (BVPs) for partial differential equations (PDE):

- **Elliptic BVPs** (”equilibrium problems”, as discussed in Sects. \([1.2.3, 2.1.1, 2.1.2]\)
- **Parabolic initial boundary value problems** (IBVPs) (”evolution towards equilibrium")
- **Hyperbolic IBVPs**, among them wave propagation problems and conservation laws (”transport/propagation")

The rigorous mathematical definition is complicated and often fails to reveal fundamental properties of, e.g., solutions that are intuitively clear against the backdrop of the physics modelled by a certain PDE. Further discussion of classification in \([3, \S 1]\) and \([15, \text{Ch. 1}]\).
Structural properties of a BPV inherited from the modelled system are more important than formal mathematical classification.

2.1 Equilibrium models

We only consider stationary systems. Then, frequently, see Sect. 1.2.2

\[ \text{equilibrium} = \text{minimal energy configuration of a system} \]

Example: elastic string model of Sect. 1.2 (minimization of energy functional \( J(u) \), see (1.2.17))
Now we study minimization problems for energy functional on spaces of functions $\Omega \mapsto \mathbb{R}$, where $\Omega \subset \mathbb{R}^d$ is a bounded (spatial) domain and $d = 2, 3$.

### 2.1.1 Taut membrane

Recall: energy functional for pinned taut string under gravitational load $\hat{g}$, see (1.4.8), in terms of displacement, see Fig. 17:

$$J(u) := \frac{1}{2} \int_a^b \sigma(x) \left| \frac{du}{dx}(x) \right|^2 - \hat{g}(x)u(x) \, dx , \quad u \in C^1_{pw}([a, b]) , \quad u(a) = u_a , \quad u(b) = u_b .$$

"2D generalization" of an elastic string \(\Rightarrow\) elastic membrane.
Taut drum membranes
Shape of membrane

Graph of $u : \Omega \mapsto \mathbb{R}$

“membrane” on spatial domain $\Omega = [0, 1]^2$

(--- --- ÷ boundary data)

Remark 2.1.1 (Spatial domains).
General assumptions on spatial domains $\Omega \subset \mathbb{R}^d$:

- $d = 1, 2, 3 \equiv \text{“dimension” of domain}$
- $\Omega$ is bounded

\[ \text{diam}(\Omega) := \sup \{ \| x - y \| : x, y \in \Omega \} < \infty, \]

- $\Omega$ has piecewise smooth boundary $\partial \Omega$

Pinning conditions (boundary conditions), cf. (1.2.1), (1.4.12):

\[
\begin{align*}
    u(x) &= g(x) \quad x \in \partial \Omega \\
    \text{fix} \quad \Leftrightarrow \quad u|_{\partial \Omega} &= g \quad \text{on } \partial \Omega.
\end{align*}
\]

for some $g \in C^0(\partial \Omega)$.  \hspace{1cm} (2.1.2)

\begin{itemize}
    \item notation: $\partial \Omega \equiv \text{boundary of } \Omega$
\end{itemize}
(2.1.2) means that the displacement of the membrane over $\partial \Omega$ is provided by a prescribed continuous function $g : \partial \Omega \mapsto \mathbb{R}$: the membrane is clamped into a rigid frame.

Intuition: $g$ has to be continuous, unless the membrane is to be torn!

(Further discussion in Rem. 2.9.4)

configuration space

$$V = \left\{ \text{continuous functions } u \in C^0(\Omega), \right. \left. \text{with } u|_{\partial \Omega} = g. \right\}$$

Think of the membrane as a grid of taut strings. Together with Rem. 1.4.11 this justifies the following expression for its total potential energy.

Potential energy of a taut membrane (described by $u \in C^0(\Omega)$) under vertical loading:

$$J_M(u) := \int_{\Omega} \frac{1}{2} \sigma(x) \| \text{grad } u \|^2 - f(x) u(x) \, dx,$$  \hfill (2.1.3)
Recall the definition of the gradient of a function $F : \Omega \subset \mathbb{R}^d \mapsto \mathbb{R}, F(x) = F(x_1, \ldots, x_d)$, see [29, Kap. 7], [18, Eq. 5.1.8]:

$$\text{grad } F(x) := \begin{pmatrix} \frac{\partial F}{\partial x_1} \\ \vdots \\ \frac{\partial F}{\partial x_d} \end{pmatrix}.$$ 

Note: the gradient at $x$ is a column vector of first partial derivatives, read $\text{grad } F(x)$ as $(\text{grad } F)(x)$; $\text{grad } F$ is a vector valued function $\Omega \mapsto \mathbb{R}^d$.

Also in use (but not in this course) is the “$\nabla$-notation”: $\nabla F(x) := \text{grad } F(x)$.

Note that

$$\sigma(x) \|\text{grad } u\|^2 = \sigma(x_1, x_2) \left| \frac{\partial u}{\partial x_1}(x_1, x_2) \right|^2 + \sigma(x_1, x_2) \left| \frac{\partial u}{\partial x_2}(x_1, x_2) \right|^2,$$

which justifies calling the taut membrane a “two-dimensional string under tension”.
with \( u : \Omega \mapsto \mathbb{R} \equiv \text{displacement function} \), see Fig. 47, \([u] = m\),
\( f : \Omega \mapsto \mathbb{R} \equiv \text{force density (pressure)} \), \([f] = \text{N m}^{-2}\),
\( \sigma : \Omega \mapsto \mathbb{R}^+ \equiv \text{stiffness} \), \([\sigma] = J\).

Displacement of taut membrane in equilibrium achieves minimal potential energy, cf. \((1.2.17)\)

\[
  u^* = \arg\min_{u \in V} J_M(u) .
\]  

Remark 2.1.5 (Minimal regularity of membrane displacement).

Smoothness required for \( u, f \) to render \( J_M(u) \) from \((2.1.3)\) meaningful, cf. Sect. 1.3.2:

- \( u \in C^1_{pw}(\Omega) \) is sufficient for displacement \( u \),
- \( \sigma, f \in C^0_{pw}(\Omega) \) already allows integration.
2.1.2 Electrostatic fields

- metal body in metal box
- prescribed voltage drop body—box

Sought: electric field $\mathbf{E} : \Omega \mapsto \mathbb{R}^3$ in $\Omega \subset \mathbb{R}^3$

$\Omega \hat{=} \text{blue region}$

Fig. 49

R. Hiptmair
C. Schwab,
H. Harbrecht
V. Gradinaru
A. Chernov
SAM, ETHZ
From Maxwell's equations, static case:

\[ E = -\nabla u \]  

(2.1.6)

where \( u : \Omega \mapsto \mathbb{R} \) \( \hat{=} \) electric (scalar) potential, \[ [u] = 1 \text{V} \]

Electric potential in technical device

Electromagnetic field energy: (electrostatic setting)

\[ J_E(E) = \frac{1}{2} \int_{\Omega} (\epsilon(x)E(x)) \cdot E(x) \, dx = \frac{1}{2} \int_{\Omega} (\epsilon(x) \nabla u(x)) \cdot \nabla u(x) \, dx \]  

(2.1.7)

where \( \epsilon : \Omega \mapsto \mathbb{R}^{3,3} \) \( \hat{=} \) dielectric tensor, \( \epsilon(x) \) symmetric, \[ [\epsilon] = \frac{A_s}{\sqrt{m}}. \]
• Symmetry of the dielectric tensor can always be assumed: if $\epsilon(x)$ was not symmetric, then replacing it with $\frac{1}{2}(\epsilon(x)^T + \epsilon(x))$ will yield exactly the same field energy.

• In terms of partial derivatives and tensor components $\epsilon(x) = (\epsilon_{ij})_{i,j=1}^3$ we have

$$\left(\epsilon(x) \text{grad } u(x)\right) \cdot \text{grad } u(x) = \sum_{i=1}^3 \sum_{j=1}^3 \epsilon_{ij}(x) \frac{\partial u}{\partial x_i}(x) \frac{\partial u}{\partial x_j}(x).$$

Fundamental property of dielectric tensor (for “normal” materials):

$$\exists 0 < \epsilon^- \leq \epsilon^+ < \infty: \quad \epsilon^- \|z\|^2 \leq (\epsilon(x)z) \cdot z \leq \epsilon^+ \|z\|^2 \quad \forall z \in \mathbb{R}^3, \forall x \in \Omega. \quad (2.1.8)$$

Terminology:  \( (2.1.8) \) \( \iff \) \( \epsilon \) is bounded and \textit{uniformly positive definite}
Definition 2.1.11 (Uniformly positive (definite) tensor field).

An matrix-valued function $A : \Omega \mapsto \mathbb{R}^{n,n}$, $n \in \mathbb{N}$, is called **uniformly positive definite**, if

$$
\exists \alpha^->0: (A(x)z) \cdot z \geq \alpha^- \|z\|^2 \quad \forall z \in \mathbb{R}^n
$$

(2.1.12)

for almost all $x \in \Omega$, that is, only with the exception of a set of volume zero.

If $A(x)$ is symmetric, then we have the equivalence, cf. [18, Rem. 5.1.20],

$$
(2.1.12) \iff A(x) \text{ s.p.d. (} \rightarrow [18, \text{Def. 2.7.9}]) \quad \text{and} \quad \lambda_{\min}(A(x)) \geq \alpha^- .
$$

What is the set/space $V$ of admissible electric scalar potentials?

Recall: in electrostatics surfaces of conducting bodies are **equipotential surfaces**.
In the situation of Fig. 49:

**Boundary conditions**

\[
\begin{align*}
    u &= 0 \quad \text{on } \Gamma_0 , \\
    u &= U_0 \quad \text{on } \Gamma_1 .
\end{align*}
\]  

(2.1.13)

\[
V = \left\{ u \in C^1_{pw}(\Omega) , u \text{ satisfies (2.1.13)} \right\} .
\]

to render \( J_E(u) \) well defined, cf. Sect. 1.3.2.

Below, the notation \( u = U \) will designate the boundary conditions (2.1.13).

**Equilibrium condition** in electrostatic setting: minimal electromagnetic field energy

\[
\begin{align*}
    u_* &= \text{argmin}_{u \in V} J_E(u) .
\end{align*}
\]  

(2.1.14)
2.1.3 Quadratic minimization problems

Structure of minimization problems (equilibrium problems) encountered above:

\[
\text{Sect. 2.1.1} \quad u^* = \text{argmin}_{u \in C^1_{pw}(\Omega), u=g \text{ on } \partial \Omega} \frac{1}{2} \int_{\Omega} \sigma(x) \| \nabla u(x) \|^2 - f(x) u(x) \, dx ,
\]
\[=: J_M(u), \text{ see (2.1.3)} \]

\[
\text{Sect. 2.1.2} \quad u^* = \text{argmin}_{u \in C^1_{pw}(\Omega), u=U \text{ on } \partial \Omega} \frac{1}{2} \int_{\Omega} \epsilon(x) \nabla u(x) \cdot \nabla u(x) \, dx .
\]
\[=: J_E(u), \text{ see (2.1.7)} \]

Evidently, (2.1.15) and (2.1.16) share a common structure. It is the same structure we have already come across in the minimization problem (1.4.2) for the taut string model in Sect. 1.4.
Definition 2.1.17 (Quadratic functional).

A quadratic functional on a real vector space $V_0$ is a mapping $J : V_0 \mapsto \mathbb{R}$ of the form

$$J(u) := \frac{1}{2} a(u, u) + \ell(u) + c, \quad u \in V_0,$$

(2.1.18)

where $a : V_0 \times V_0 \mapsto \mathbb{R}$ is a symmetric \underline{bilinear form} (→ Def. 1.3.13), $\ell : V_0 \mapsto \mathbb{R}$ a \underline{linear form}, and $c \in \mathbb{R}$.

Recall: A \underline{bilinear form} $a : V_0 \times V_0 \mapsto \mathbb{R}$ is symmetric, if

$$a(u, v) = a(v, u) \quad \forall u, v \in V_0.$$  \hspace{1cm} (2.1.19)

If $V_0 = \mathbb{R}^N$ (finite-dimensional case), then a quadratic functional has the general representation

$$J(u) = u^T A u + b^T u + c, \quad A = A^T \in \mathbb{R}^{N,N}, \quad b \in \mathbb{R}^N, \quad c \in \mathbb{R}.$$  \hspace{1cm} (2.1.20)

Reminder: quadratic functionals of this forms occur in derivation of steepest descent and conjugate gradient methods for linear systems of equations, see [18, Sect. 5.1.1].
**Definition 2.1.21** (Quadratic minimization problem).

A minimization problem

\[ w_* = \arg\min_{w \in V_0} J(w) \]

is called a **quadratic minimization problem**, if \( J \) is a quadratic functional on a real vector space \( V_0 \).

Hey, both (2.1.15) and (2.1.16) are no genuine quadratic minimization problems, because they are posed over affine spaces (\( = \) “vector space + offset function”, cf. (1.3.14))!

“Offset function trick”, c.f. (1.3.19), resolves the mismatch: for quadratic form \( J \) from (2.1.18)

\[
J(u + u_0) = \frac{1}{2} a(u + u_0, u + u_0) + \ell(u + u_0) + c
= \frac{1}{2} a(u, u) + a(u, u_0) + \ell(u) + \frac{1}{2} a(u_0, u_0) + \ell(u_0) + c
=: \tilde{J}(u),
\]

due to the bilinearity of \( a \) and the linearity of \( \ell \).

\[ \arg\min_{u \in u_0 + V_0} J(u) = u_0 + \arg\min_{w \in V_0} J(w + u_0) = u_0 + \arg\min_{w \in V_0} \tilde{J}(w). \quad (2.1.22) \]
For a discussion of quadratic functionals on $\mathbb{R}^n \to [18, \text{Sect. 5.1.1}]$.

Both (2.1.15) and (2.1.16) involve quadratic functionals. To see this apply the “offset function trick” from (2.1.22) in this concrete case: write $u = u_0 + w$ with an offset function $u_0$ that satisfies the boundary conditions and $w \in C^1_{0,\text{pw}}(\Omega)$, cf. (1.3.19).

(2.1.15) $\Rightarrow$ quadratic minimization problem $\rightarrow$ Def. 2.1.21 with, cf. (2.1.18),

$$a(w, v) = \int_{\Omega} \sigma(x)\nabla w(x) \cdot \nabla v(x) \, dx, \quad \ell(v) := a(u_0, v) - \int_{\Omega} f(x)v(x) \, dx.$$  \hfill (2.1.23)

(2.1.16) $\Rightarrow$ quadratic minimization problem $\rightarrow$ Def. 2.1.21 with, cf. (2.1.18),

$$a(w, v) = \int_{\Omega} \nabla w(x)^T \epsilon(x)\nabla v(x) \, dx, \quad \ell(v) := a(u_0, v).$$  \hfill (2.1.24)

In both cases: $V_0 = C^1_{0,\text{pw}}(\Omega)$.
Can we conclude existence and uniqueness of solutions of the minimization problems (2.1.15) and (2.1.16)?

Let us first tackle the issue of **uniqueness**:

### Definition 2.1.25 (Positive definite bilinear form).

A (symmetric) **bilinear form** $a : V_0 \times V_0 \mapsto \mathbb{R}$ on a real vector space $V_0$ is **positive definite**, if

$$u \in V_0 \setminus \{0\} \iff a(u, u) > 0.$$ 

For the special case $V_0 = \mathbb{R}^n$ any matrix $A \in \mathbb{R}^{n,n}$ induces a bilinear form via

$$a(u, v) := u^T Av = (Av) \cdot u, \quad u, v \in \mathbb{R}^n.$$  

(2.1.26)

This connects the concept of a symmetric positive definite bilinear form to the more familiar concept of s.p.d. matrices ($\rightarrow$ [18, Def. 2.7.9])

$$A \text{ s.p.d. } \iff a \text{ from (2.1.26) is symmetric, positive definite.}$$
Definition 2.1.27 (Energy norm). cf. [18, Def. 5.1.1]

A symmetric positive definite bilinear form \( a : V_0 \times V_0 \mapsto \mathbb{R} \) (→ Def. 2.1.25) induces the energy norm

\[ \| u \|_a := (a(u, u))^{1/2}. \]

Origin of the term “energy norm” is clear from the connection with potential energy (e.g., in membrane model and in the case of electrostatic fields, see (2.1.23), (2.1.24)), see above.

Next, we have to verify the norm axioms \((N1)\), \((N2)\), and \((N3)\) from Def. 1.6.3:

- \((N1)\) is immediate from Def. 2.1.25,
- \((N2)\) follows from bilinearity of \( a \),
- \((N3)\) is a consequence of the Cauchy-Schwarz inequality: for any symmetric positive definite bilinear form

\[ |a(u, v)| \leq (a(u, u))^{1/2}(a(v, v))^{1/2}. \quad (2.1.28) \]
Example 2.1.29 (Quadratic functionals with positive definite bilinear form in 2D).

Analogy between quadratic functionals with positive definite bilinear form and parabolas:

\[ J(v) = \frac{1}{2}a(v, v) - \ell(v) \]

\[ f(x) = \frac{1}{2}ax^2 + bx \]

with \( a > 0 \)!

graph of quadratic functional \( \mathbb{R}^2 \mapsto \mathbb{R} \)

\[ \text{Fig. 52} \]
Theorem 2.1.30 (Uniqueness of solutions of quadratic minimization problems).

If the bilinear form \( a : V_0 \times V_0 \mapsto \mathbb{R} \) is positive definite (\( \rightarrow \) Def. 2.1.25), then any solution of

\[
    u_* = \arg\min_{u \in V_0} J(u) \quad , \quad J(u) = \frac{1}{2} a(u, u) + \ell(u) + c ,
\]

is unique for any linear form \( \ell : V_0 \mapsto \mathbb{R} \).

Proof. As in the proof of [18, Lemma 5.1.3], straightforward computations show

\[
    J(u) - J(u_*) = \frac{1}{2} \| u - u_* \|_a^2 .
\]

The assertion of the theorem follows from norm axiom (N1), which holds for the energy norm. \( \square \)

Under the assumptions of the theorem, the quadratic functional \( J \) is convex, which is easily seen by considering the second derivative of the function

\[
    \varphi(t) := J(u + tv) \quad \Rightarrow \quad \ddot{\varphi}(t) = a(v, v) > 0 \quad , \text{if } v \neq 0 .
\]
Is \( a(u, v) := \int_{\Omega} (\epsilon(x) \nabla u(x)) \cdot \nabla v(x) \, dx \) positive definite on \( V_0 := C^{1}_{0,pw}(\Omega) \)?

1: Since \( \epsilon \) bounded and uniformly positive definite (\( \rightarrow \) Def. 2.1.11, (2.1.8))

\[
\epsilon^- \int_{\Omega} \|\nabla u(x)\|^2 \, dx \leq a(u, u) \leq \epsilon^+ \int_{\Omega} \|\nabla u(x)\|^2 \, dx \quad \forall u .
\]

Hence, it is sufficient to examine the simpler bilinear form

\[
d(u, v) := \int_{\Omega} \nabla u(x) \cdot \nabla v(x) \, dx , \quad u, v \in C^{1}_{0,pw}(\Omega) .
\]

2: Obviously

\[
d(u, u) = 0 \quad \Rightarrow \quad \nabla u = 0 \quad \Rightarrow \quad u \equiv \text{const in } \Omega
\]

Observe:

\[
u = 0 \text{ on } \partial \Omega \quad \Rightarrow \quad u = 0
\]

Zero boundary conditions are essential; otherwise one could add constants to the arguments of \( a \) without changing its value.
What about existence?

In a finite dimensional setting this is not a moot point, see Fig. 52 for a “visual proof”.

However, infinite dimensional spaces hold a lot of surprises and existence of solutions of quadratic minimization problems becomes a subtle issue, even if the bilinear form is positive definite.

**Example 2.1.33** (Non-existence of solutions of positive definite quadratic minimization problem).

We consider the quadratic functional

$$J(u) := \int_0^1 \frac{1}{2} u^2(x) - u(x) \, dx = \frac{1}{2} \int_0^1 (u(\xi) - 1)^2 - 1 \, d\xi ,$$

on the space

$$V_0 := C_{0,pw}^0([0,1])$$

It fits the abstract form from Def. 2.1.17 with

$$a(u,v) = \int_0^1 u(x)v(x) \, dx \quad , \quad \ell(v) = \int_0^1 v(x) \, dx .$$
The function \( \varphi(\xi) = \frac{1}{2} \xi^2 - \xi = \frac{1}{2} \xi(1 - 2\xi) = \frac{1}{2}(\xi - 1)^2 - \frac{1}{2} \) has a global minimum at \( \xi = 1 \) and 
\[ \varphi(\xi) - \varphi(1) = \frac{1}{2}(\xi - 1)^2. \]

\[ |\eta - 1| > |\xi - 1| \implies \varphi(\eta) > \varphi(\xi). \]

Assume that \( u \in V_0 \) is a global minimizer of \( J \). Then
\[ w(x) := \min\{1, 2 \max\{u(x), 0\}\}, \]
\[ 0 \leq x \leq 1, \]
is another function \( \in C^0_{0,pw}([0, 1]) \), which satisfies
\[ u(x) \neq 1 \Rightarrow |w(x) - 1| < |u(x) - 1| \]
\[ \Rightarrow J(w) < J(u) ! \]

Hence, whenever we think we have found a minimizer \( \in C^0_{0,pw}([0, 1]) \), the formula provides another eligible function for which the value of the functional is even smaller!

The problem in this example seems to be that we have chosen “too small” a function space, c.f. Sect. 2.2 below.
2.2 Sobolev spaces

Mathematical theory is much concerned about proving existence of suitably defined solutions for minimization problems. As demonstrated in Ex. 2.1.33 this can encounter profound problems.

In this section we will learn about a class of abstract function spaces that has been devised to deal with the question of existence of solutions of quadratic minimization problems like (2.1.15) and (2.1.16). We can only catch of glimps of the considerations; thorough investigation is done in the mathemtical field of functional analysis.
Consider a quadratic minimization problem (→ Def. 2.1.21) for a quadratic functional (→ Def. 2.1.17)

\[ J : V_0 \rightarrow \mathbb{R} \ , \ J(u) = \frac{1}{2} a(u, u) + \ell(u) + c \ , \]

based on a symmetric positive definite (s.p.d.) bilinear form \( a \) → Def. 2.1.25.

It is clear that \( J(V_0) \) is bounded from below, if

\[ \exists C > 0: |\ell(u)| \leq C \|u\|_a \ \forall u \in V_0 \ , \]

where \( \|\cdot\|_a \) is the energy norm induced by \( a \), see Def. 2.1.27:

\[ J(u) = \frac{1}{2} a(u, u) - \ell(u) \geq \frac{1}{2} \|u\|_a^2 - C \|u\|_a \geq -\frac{1}{2} C^2 . \]

Remark: In mathematical terms (2.2.1) means that \( \ell \) is continuous w.r.t. \( \|\cdot\|_a \)

Under these conditions, the quadratic minimization problem for \( J \) should have a (unique, due to Thm. 2.1.30) solution, if it is considered on a space that is “large enough”.

R. Hiptmair
C. Schwab,
H. Harbrecht
V. Gradinaru
A. Chernov
SAM, ETHZ

2.2
p. 194
Idea: for a quadratic minimization problem (→ Def. 2.1.21) with

- symmetric **positive definite** (s.p.d.) bilinear form \(a\),
- a linear form \(\ell\) that is continuous w.r.t. \(\|\cdot\|_a\), see (2.2.1),

posed over a function space follow the advice:

consider it on the largest space of functions for which \(a\) still makes sense!

(and which complies with boundary conditions)

Choose "\(V_0 := \{\text{functions } v \text{ on } \Omega: a(v, v) < \infty\}\)"

**Example 2.2.2** (Space of square integrable functions). → Ex. 2.1.33

Quadratic functional (related to \(J\) from Ex. 2.1.33):

\[
J(u) := \int_{\Omega} \frac{1}{2}|u(x)|^2 - u(x) \, dx.
\]  
\(u \in C^0_{pw}(\Omega)\) \(\square\)  

(2.2.3)
We follow the above recipe, which suggests to choose

\[ V_0 := \{ v : \Omega \mapsto \mathbb{R} \text{ integrable: } \int_{\Omega} |v(x)|^2 \, dx < \infty \} \tag{2.2.4} \]

**Definition 2.2.5 (Space \( L^2(\Omega) \)).**

The function space defined in (2.2.4) is the space of square-integrable functions on \( \Omega \) and denoted by \( L^2(\Omega) \).

It is a normed space with norm

\[ \|v\|_0 := \|v\|_{L^2(\Omega)} := \left( \int_{\Omega} |v(x)|^2 \, dx \right)^{1/2}. \]

Notation:

\[ L^2(\Omega) \leftarrow \text{superscript "2", because square in the definition of norm } \| \cdot \|_0 \]

Note: obviously \( C^{0}_{pw}(\Omega) \subset L^2(\Omega) \).
Remark 2.2.6 (Boundary conditions and $L^2(\Omega)$).

Ex. 2.1.33 vs. Ex. 2.2.2: Crying foul! (boundary conditions $u(0) = u(1) = 0$ in Ex. 2.1.33, but none in Ex. 2.2.2!)

Consider $u \in C^0([0, 1])$ and try to impose boundary values $u_0, u_1 \in \mathbb{R}$ by “altering” $u$:

$$\tilde{u}(x) = \begin{cases} 
  u(x) + (1 - nx)(u_0 - u(0)) & \text{for } 0 \leq x \leq \frac{1}{n}, \\
  u(x) & \text{for } \frac{1}{n} < x < 1 - \frac{1}{n}, \\
  u(x) - n(1 - \frac{1}{n} - x)(u_1 - u(1)) & \text{for } 1 - \frac{1}{n} < x \leq 1.
\end{cases}$$

$$\tilde{u}(0) = u_0, \quad \tilde{u}(1) = u_1, \quad \|\tilde{u} - u\|_{L^2([0,1])} = \frac{1}{3n}(u_0 + u_1 - u(0) - u(1)) \to 0 \quad \text{for } n \to \infty.$$ 

Tiny perturbations of a function $u \in L^2([0, 1])$ (in terms of changing its $L^2$-norm) can make it attain any value at $x = 0$ and $x = 1$.

Boundary conditions cannot be imposed in $L^2(\Omega)$!
Remark 2.2.7 (Quadratic minimization problems on Hilbert spaces).

On the function space \( V_0 = L^2(\Omega) \) the quadratic minimization problem for the quadratic functional from (2.2.3) can be shown to possess a solution. Instrumental in the proof is the fact that \( L^2(\Omega) \) is a Hilbert space, that is, a complete normed space.

This theory is beyond the scope of this course. For more explanations see [12, Ch. 5 and Sect. 6.2].

Now consider a quadratic minimization problem for the functional, c.f. (2.1.15),

\[
J(u) := \int_{\Omega} \frac{1}{2} \| \text{grad } u \|^2 - f(x)u(x) \, dx \quad (u \in C^{1}_{0,\text{pw}}(\Omega))
\] (2.2.8)

What is the natural function space for this minimization problem? Again, we follow the above recipe, which suggests that we choose...
\( V_0 := \{ v : \Omega \rightarrow \mathbb{R} \text{ integrable: } v = 0 \text{ on } \partial \Omega, \int_{\Omega} |\text{grad} \, v(x)|^2 \, dx < \infty \} \tag{2.2.9} \)

**Definition 2.2.10** (Sobolev space \( H^1_0(\Omega) \)).

The space defined in (2.2.9) is the **Sobolev space** \( H^1_0(\Omega) \) with norm

\[
|v|_{H^1_0(\Omega)} := \left( \int_{\Omega} \|\text{grad} \, v\|^2 \, dx \right)^{1/2}.
\]

Notation:
- Superscript “1”, because first derivatives occur in norm
- Subscript “0”, because zero on \( \partial \Omega \)

Note: \(|\cdot|_{H^1(\Omega)}\) is the **energy norm** (→ Def. 2.1.27) associated with the bilinear form in the quadratic functional \( J \) from (2.2.8), cf. (2.1.18).

☛ See Rem. [1.6.7](#) for a discussion of the relevance of the energy norm.
Remark 2.2.11 (Boundary conditions in $H^1_0(\Omega)$).

Rem. 2.2.6 explained why imposing boundary conditions on functions in $L^2(\Omega)$ does not make sense.

Yet, in (2.2.9) zero boundary conditions are required for $v$!

Discussion parallel to Rem. 2.2.6, but now with the norm $|\cdot|_{H^1(\Omega)}$ in mind: Consider $u \in C^1([0, 1])$ and try to impose boundary values $u_0, u_1 \in \mathbb{R}$ by “altering” $u$:

$$
\tilde{u}(x) = \begin{cases} 
  u(x) + (1 - nx)(u_0 - u(0)) & \text{for } 0 \leq x \leq \frac{1}{n}, \\
  u(x) & \text{for } \frac{1}{n} < x < 1 - \frac{1}{n}, \\
  u(x) - n(1 - \frac{1}{n} - x)(u_1 - u(1)) & \text{for } 1 - \frac{1}{n} < x \leq 1.
\end{cases}
$$

$\tilde{u}(0) = u_0$, $\tilde{u}(1) = u_1$, BUT $|\tilde{u} - u|_{H^1([0, 1])}^2 = n(u_0 + u_1 - u(0) - u(1)) \to \infty$ for $n \to \infty$

Enforcing boundary values at $x = 0$ and $x = 1$ cannot be done without significantly changing the “energy” of the function.
However, the solutions of the quadratic minimization problems (2.1.15), (2.1.16) are to satisfy non-zero boundary conditions. They are sought in a larger Sobolev space, which arises from $H^1_0(\Omega)$ by dispensing with the requirement “$v = 0$ on $\partial \Omega$”.

**Definition 2.2.12 (Sobolev space $H^1(\Omega)$).**

*The Sobolev space*

$$H^1(\Omega) := \{v : \Omega \mapsto \mathbb{R} \text{ integrable: } \int_\Omega |\text{grad } v(x)|^2 \, dx < \infty \}$$

is a normed function space with norm

$$\|v\|_{H^1(\Omega)}^2 := \|v\|_0^2 + |v|_{H^1(\Omega)}^2.$$

$H^1(\Omega)$ is the “maximal function space” on which both $J_M$ and $J_E$ from (2.1.15), (2.1.16) are defined.
Remark 2.2.13 ($|\cdot|_{H^1(\Omega)}$-seminorm).

Note that $|\cdot|_{H^1(\Omega)}$ alone is no longer a norm on $H^1(\Omega)$, because for $v \equiv \text{const}$ obviously $|v|_{H^1(\Omega)} = 0$, which violates (NT).

△

In the introduction to this section we saw that a quadratic functional with s.p.d. bilinear form $a$ is bounded from below, if its linear form $\ell$ satisfies the continuity (2.2.1). Now, we discuss this for the quadratic functional $J$ from (2.2.8) in lieu of $J_M$ and $J_E$.

The quadratic functional $J$ from (2.2.8) involves the linear form

$$\ell(u) := \int_{\Omega} f(\mathbf{x}) u(\mathbf{x}) \, d\mathbf{x}. \quad (2.2.14)$$

$f \triangleq \text{load function} \Rightarrow f \in C^0_{pw}(\Omega)$ should be admitted.
Crucial question: Is $\ell$ from (2.2.14) continuous on $H^1_0(\Omega)$?

$c.f.$ (2.2.1)

$\exists C > 0: |\ell(u)| \leq C|u|_{H^1(\Omega)} \quad \forall u \in H^1_0(\Omega)$.

To begin with, we use the Cauchy-Schwarz inequality (2.1.28) for integrals, which implies

$$
|\ell(u)| = \left| \int_\Omega f(x)u(x)\,dx \right| \leq \left( \int_\Omega |f(x)|^2\,dx \right)^{1/2} \left( \int_\Omega |u(x)|^2\,dx \right)^{1/2} = \|f\|_0 \|u\|_0. \quad (2.2.15)
$$

This reduces the problem to bounding $\|u\|_0$ in terms of $|u|_{H^1(\Omega)}$.

**Theorem 2.2.16 (First Poincaré-Friedrichs inequality).**

*If $\Omega \subset \mathbb{R}^d$, $d \in \mathbb{N}$, is bounded, then*

$$
\|u\|_0 \leq \text{diam}(\Omega) \|\text{grad } u\|_0 \quad \forall u \in H^1_0(\Omega).
$$
Proof. The proof employs a powerful technique in the theoretical treatment of function spaces: exploit density of smooth functions (which, by itself, is a deep result).

It boils down to the insight:

In order to establish inequalities between continuous functionals on Sobolev spaces of functions on $\Omega$ it often suffices to show the target inequality for smooth functions in $C_0^\infty(\Omega)$ or $C^\infty(\Omega)$, respectively.

notation: $C_0^\infty(\Omega) \doteq$ smooth functions with (compact) support (→ Def. 1.5.57) inside $\Omega$

In the concrete case (note the zero boundary values inherent in the definition of $H_0^1(\Omega)$) we have to establish the first Poincaré-Friedrichs inequality for functions $u \in C_0^\infty(\Omega)$ only.

For the sake of simplicity the proof is elaborated for $d = 1$, $\Omega = [0, 1]$. It merely employs elementary results from calculus throughout, namely the Cauchy-Schwarz inequality (2.2.15) and the fundamental theorem of calculus [29, Satz 6.3.4], see (2.4.1):

$$\forall u \in C_0^\infty([0, 1]): \quad u(x) = u(0) + \int_0^x \frac{du}{dx}(\tau) \, d\tau , \quad 0 \leq x \leq 1 .$$
∥u∥_0^2 = \int_0^1 \left| \int_0^x \frac{du}{dx}(\tau) \, d\tau \right|^2 \, dx \leq \int_0^1 \left( \int_0^x d\tau \cdot \int_0^x \left| \frac{du}{dx}(\tau) \right|^2 \, d\tau \right) \, dx \leq \left\| \frac{du}{dx} \right\|_0^2.

Taking the square root finished the proof in 1D. \qed

The elementary proof in higher dimensions can be found in [15, Sect. 6.2.2] and in even greater generality in [12, Sect. 5.6.1].

If \( f \in L^2(\Omega) \), then \( \ell(u) = \int_\Omega fu \, dx \) is a continuous linear functional on \( H^1_0(\Omega) \).

Here “continuity” has to be read as

\[ \exists C > 0 : \quad |\ell(u)| \leq C|u|_{H^1(\Omega)} \quad \forall u \in H^1_0(\Omega), \]  

(2.2.1)
Most concrete results about Sobolev spaces boil down to relationships between their norms. The spaces themselves remain intangible, but the norms are very concrete and can be computed and manipulated as demonstrated above.

Do not be afraid of Sobolev spaces!
It is only the norms that matter for us, the ‘spaces” are irrelevant!

Sobolev spaces = “concept of convenience”: the minimization problem seeks its own function space.

Minimization problem

\[ u = \arg \min_{v: \Omega \to \mathbb{R}} J(v) \]

“Maxmimal” function space on which \( J \) is defined
(Sobolev space)
Then, why do you bother me with these uncanny “Sobolev spaces” after all?

- Anyone involved in CSE must be able to understand mathematical publications on numerical methods for PDEs, those regularly resort to the concept of Sobolev spaces to express their findings.
- The statement that a function belongs to a certain Sobolev space can be regarded as a concise way of describing quite a few of its essential properties.

Let us elucidate the second point:
Theorem 2.2.17. Compatibility conditions for piecewise smooth functions in $H^1(\Omega)$

Let $\Omega$ be partitioned into sub-domains $\Omega_1$ and $\Omega_2$. A function that is continuously differentiable in both sub-domains and continuous up to their boundary, belongs to $H^1(\Omega)$, if and only if $u$ is continuous on $\Omega$.

The proof of this theorem requires the notion of weak derivatives that will not be introduced in this course.

Example 2.2.18 (Piecewise linear functions (not) in $H^1_0([0,1])$).
We conclude from Thm. 2.2.17:

\[ u \in H^1_0([0, 1]) \]

\[ u \not\in H^1_0([0, 1]) \]

From Thm. 2.2.17 we conclude

\[ C^1_{pw}([a, b]) \subset H^1([a, b]) \quad \text{and} \quad C^1_{0,pw}([a, b]) \subset H^1_0([a, b]) \]
Thm. 2.2.17 provides a simple recipe for computing the norm $|u|_{H^1(\Omega)}$ of a piecewise $C^1$-function that is continuous in all of $\Omega$.

**Corollary 2.2.19** ($H^1$-norm of piecewise smooth functions).

*Under the assumptions of Thm. 2.2.17 we have for a continuous, piecewise smooth function $u \in C^0(\Omega)$*

$$|u|^2_{H^1(\Omega)} = |u|^2_{H^1(\Omega_1)} + |u|^2_{H^1(\Omega_2)} = \int_{\Omega_1} |\nabla u(\mathbf{x})|^2 \, d\mathbf{x} + \int_{\Omega_2} |\nabla u(\mathbf{x})|^2 \, d\mathbf{x}.$$  

Actually, this is not new, see Sect. 1.3.2: earlier we already evaluated the elastic energy functionals (1.2.16), (1.4.2) for functions in $C^1_{pw}([0, 1])$ by “piecewise differentiation” followed by integration of the resulting discontinuous function.

**Example 2.2.20** (Non-differentiable function in $H^1_0([0, 1])$).
\( d = 1, \Omega = ]0, 1[ : \)

“Tent function”  
\[
\begin{align*}
  u(x) &= \begin{cases} 
    2x & \text{for } 0 < x < \frac{1}{2}, \\
    2(1-x) & \text{for } \frac{1}{2} < x < 1.
  \end{cases}
\end{align*}
\]

Compute  
\[
|u|_{H^1(\Omega)}^2 = \int_0^1 |u'(x)|^2 \, dx = 4 < \infty.
\]

Example for a \( u \in H^1_0(]0, 1[), \) which is not globally differentiable.

Recall: we cheerfully computed the derivative of a piecewise smooth function already in Sect. 1.5.1.2 when differentiating the basis functions, \( \text{cf. (1.5.55)} \). Now this “reckless” computations have found their rigorous justification.
If you are still feeling uneasy when dealing with Sobolev spaces, do not hesitate to think of the following replacements

\[ L^2(\Omega) \to C^0_{pw}(\Omega), \quad H^1_0(\Omega) \to C^1_{0,pw}(\Omega). \]

2.3 Variational formulations

2.3.1 Linear variational problems
Recall: derivation of variational formulation (1.4.4) from taut string minimization problem (1.4.2) in Sect. 1.4.

No surprise: (2.1.15) & (2.1.16) are amenable to the same approach:

Calculus of variations → Sect. 1.3.1: “Directional derivative” of $J_E$:

$$J_E(u + tv) - J_E(u) = \frac{1}{2} \int_{\Omega} (\epsilon(x) \nabla (u + tv)) \cdot \nabla (u + tv) \, dx$$

$$- \frac{1}{2} \int_{\Omega} (\epsilon(x) \nabla u) \cdot \nabla u \, dx$$

$$= \frac{1}{2} \int_{\Omega} (\epsilon(x) \nabla u) \cdot \nabla u + 2t(\epsilon(x) \nabla u) \cdot \nabla v +$$

$$+ t^2(\epsilon(x) \nabla v) \cdot \nabla v - (\epsilon(x) \nabla u) \cdot \nabla u \, dx$$

$$= t \int_{\Omega} (\epsilon(x) \nabla u) \cdot \nabla v \, dx + O(t^2) \quad \text{for } t \to 0 .$$

\((\ast):\) due to the symmetry of $\epsilon(x)$: $$(\epsilon \nabla u) \cdot \nabla v = (\epsilon \nabla v) \cdot \nabla u !$$

$$\lim_{t \to 0} \frac{J_E(u + tv) - J_E(u)}{t} = \int_{\Omega} (\epsilon(x) \nabla u(x)) \cdot \nabla v(x) \, dx ,$$

for perturbation functions $v \in H^1_0(\Omega)$, see Def. 2.2.10.
The requirement $v = 0$ on $\partial \Omega$ reflects the fact that we may not perturb $u$ on the boundary, lest the prescribed boundary values be violated.

As explained in Sect. 1.3.1 (“idea of calculus of variations”), this leads to the following variational problem equivalent to (2.1.16)

$$u \in H^1(\Omega), \quad u = U \text{ on } \partial \Omega, \quad \int_{\Omega} (\epsilon(x) \text{grad } u(x)) \cdot \text{grad } v(x) \, dx = 0 \quad \forall v \in H^1_0(\Omega). \quad (2.3.1)$$

For the membrane problem (2.1.15) we arrive at

$$u \in H^1(\Omega), \quad u = g \text{ on } \partial \Omega, \quad \int_{\Omega} \sigma(x) \text{grad } u(x) \cdot \text{grad } v(x) \, dx = \int_{\Omega} f(x)v(x) \quad \forall v \in H^1_0(\Omega). \quad (2.3.2)$$

Both, (2.3.1) and (2.3.2) have a common structure, expressed in the following variational problem:
Variational formulation of 2nd-order elliptic (Dirichlet) minimization problems:
\[
\begin{align*}
u &\in H^1(\Omega), \\
u &= g \text{ on } \partial\Omega.
\end{align*}
\]
\[
\int_\Omega (\alpha(x) \text{grad } u(x)) \cdot \text{grad } v(x) \, dx = \int_\Omega f(x)v(x) \, dx \quad \forall v \in H^1_0(\Omega). 
\]
(2.3.3)

Symmetric uniformly positive definite material tensor \( \alpha : \Omega \mapsto \mathbb{R}^{d,d} \)

The attribute “Dirichlet” refers to a setting, in which the function \( u \) is prescribed on the entire boundary.

Some more explanations and terminology:

- \( \Omega \subset \mathbb{R}^d, d = 2, 3 \) (spatial) domain, bounded, piecewise smooth boundary
- \( g \in C^0(\partial\Omega) \) ≡ boundary values (Dirichlet data)
- \( f \in C^0_{pw}(\Omega) \) ≡ loading function, source function
- \( \alpha : \Omega \mapsto \mathbb{R}^{d,d} \) ≡ material tensor, stiffness function, diffusion coefficient (uniformly positive definite, bounded → Def. 2.1.11)
\[ \exists 0 < \alpha^- \leq \alpha^+ : \alpha^- \|z\|^2 \leq (\alpha(x)z) \cdot z \leq \alpha^+ \|z\|^2 \quad \forall z \in \mathbb{R}^d , \quad (2.3.4) \]

for almost all \( x \in \Omega \).

Rewriting (2.3.3), using offset function \( u_0 \) with \( u_0 = g \) on \( \partial \Omega \), cf. (2.1.22),

\[ w \in H^1_0(\Omega) : \int_{\Omega} (\alpha(x) \nabla w(x)) \cdot \nabla v(x) \, dx \]

\[ = \int_{\Omega} f(x)v(x) - (\alpha(x) \nabla u_0(x)) \cdot \nabla v(x) \, dx \quad \forall v \in H^1_0(\Omega) . \quad (2.3.5) \]

\[ \text{(2.3.5) is a linear variational problem, see Rem. 1.4.5} \]

We can lift the above discussion to an abstract level, cf. discussion after (1.4.6):

Variational formulation of a quadratic minimization problem (→ Def. 2.1.21)

\[ J(u) := \frac{1}{2}a(u, u) + \ell(u) + c \quad \Rightarrow \quad J(u + tv) = J(u) + t(a(u, v) + \ell(v)) + \frac{1}{2}t^2a(v, v) , \]

for all \( u, v \in V_0 \).
For a quadratic functional (\(\rightarrow\) Def. 2.1.21) on real vector space \(V_0\)
\[
\lim_{t \to 0} \frac{J(u + tv) - J(u)}{t} = a(u, v) + \ell(v). \quad (2.3.6)
\]

Linear variational problem (\(\rightarrow\) Rem. 1.4.5) arising from quadratic minimization problem for functional \(J(u) := \frac{1}{2}a(u, u) + \ell(u) + c:\)

\[
\text{Concretely, for (2.3.5): } V_0 = H^1_0(\Omega) \text{ and }
\]
\[
a(u, v) = \int_{\Omega} (\alpha(x) \text{grad } w(x)) \cdot \text{grad } v(x) \, dx , \quad (2.3.8)
\]
\[
\ell(v) = -\int_{\Omega} f(x)v(x) + (\alpha(x) \text{grad } u_0(x)) \cdot \text{grad } v(x) \, dx . \quad (2.3.9)
\]
2.3.2 Stability

Notion of stability for a (linear) variational problem (2.3.7):

Lipschitz continuity of (linear) mapping data $\ell \mapsto$ solution $w$

Is there/what is a constant $C_{\text{stab}}>0$ such that

$$\|w\|_X \leq C_{\text{stab}} \|\ell\|_Y,$$

where $w$ solves (2.3.7),

with suitable/relevant norms $\|\cdot\|_X$, $\|\cdot\|_Y$? These norms will be suggested by the modelling background. Their choice will determine existence and value of $C_{\text{stab}}$.

Remark 2.3.11 (Sensitivity of linear variational problems).

Recall a notion introduced in [18, Sect. 2.5.5]:

Sensitivity of a problem (for given data) gauges impact of small perturbations of the data on the result.
Remember: “Problem” = mapping from data space to solution space, see [18, Sect. 2.5.2].

Here, we define the “problem” as the mapping

\[
\begin{align*}
\{ \text{linear forms on } V_0 \} & \mapsto V_0 \\
\ell & \mapsto w \in V_0: \quad a(w, v) = -\ell(v) \quad \forall v \in V_0.
\end{align*}
\] (2.3.12)

Undesirable: “sensitive dependence of solution on data”, that is small (in the norm of the data space) perturbations of \( \ell \) translate into huge (in the norm of the solution space) or even “infinite” perturbations of the solution. In this case of an “ill-posed problem” inevitable data errors (e.g., due to non-exact measurements) will thwart any attempt to compute an “accurate” (in the norm of the solution space) solution.

Desirable: Lipschitz continuity of problem map with small Lipschitz constant (well-posed problem).

Note: the problem map (2.3.12) is linear and its Lipschitz constant is given by the smallest value for \( C_{\text{stab}} \) in (2.3.10).
Consider the particular choice (2.3.8).

How to choose the norms $\| \cdot \|_X$ (on solution space) and $\| \cdot \|_Y$ (on data space)?

**Norm on solution space:** *energy norm: $\| \cdot \|_a$

**Norm on r.h.s:** Mean square norm ($L^2$-norm, $\rightarrow$ Def. 2.2.5) for $f$, $H^1$-semi-norm ($\rightarrow$ Def. 2.2.12) for $u_0$

What will be the impact of a perturbation of $\ell$, if we use these norms?

First use the Cauchy-Schwarz inequality (2.2.15) and the uniform positivity ($\rightarrow$ Def. 2.1.11) of $\alpha$, see (2.3.4):

$$|\ell(v)| \leq \| f \|_0 \| v \|_0 + \alpha^+ \| \text{grad } u_0 \|_0 \| \text{grad } v \|_0$$

$$\leq \left( \| f \|_0^2 + (\alpha^+)^2 \| \text{grad } u_0 \|_0^2 \right)^{1/2} \| v \|_{H^1(\Omega)} \quad \forall v \in H^1(\Omega). \quad (2.3.13)$$
Next, we appeal to the lower estimate in (2.3.4) and the first Poincaré-Friedrichs inequality of Thm. 2.2.16:

\[ \|v\|_{H^1(\Omega)} \leq \sqrt{1 + \text{diam}^2(\Omega)} |v|_{H^1(\Omega)} \leq \sqrt{\frac{1 + \text{diam}^2(\Omega)}{\alpha^{-}}} \|v\|_a . \]  

(2.3.14)

Combine (2.3.13) and (2.3.14),

\[ |\mathcal{L}(v)| \leq \left( \|f\|_0^2 + (\alpha^+)^2 |u_0|^2_{H^1(\Omega)} \right)^{1/2} \sqrt{\frac{1 + \text{diam}^2(\Omega)}{\alpha^{-}}} \|v\|_a . \]

This enters the estimate for the perturbation of the solution:

\[ a(w, v) = -\mathcal{L}(v) \quad \forall v \in V_0 , \]
\[ a(w + \delta w, v) = -(\mathcal{L} + \delta \mathcal{L})(v) \quad \forall v \in V_0 . \]

\[ a \text{ bilinear} \]

\[ \Rightarrow \]

\[ \|\delta w\|_a = \sqrt{a(\delta w, \delta w)} = \sqrt{\|\delta \mathcal{L}(\delta w)\|} \leq (K(\delta f, \delta u_0) \|\delta w\|_a)^{1/2} , \]

\[ \Rightarrow \]

\[ \|\delta w\|_a \leq K(\delta f, \delta u_0) . \]

As in Rem. 1.6.7 for associated quadratic energy functional \( J \):

\[ |J(w + \delta w) - J(w)| = \frac{1}{2} |a(2w + \delta w, \delta w)| \leq \frac{1}{2} \|2w + \delta w\|_a \|\delta w\|_a . \]  

(2.3.15)
Perturbation estimates in energy norm directly translate into perturbation estimates for the equilibrium energy!

**Remark 2.3.16 (Needle loading).**

Now we inspect a striking manifestation of instability for a 2nd-order elliptic variational problem caused by a right hand side functional that fails to satisfy (2.2.1).

Consider the taut membrane model, see Sect. 2.1.1 for details, (2.1.15) for the related minimization problem, and (2.3.2) for the associated variational equation.

Let us assume that a needle is poked at the membrane: loading by a force $f$ “concentrated in a point $y$”, often denoted by $f = \delta_y$, $y \in \Omega$, where $\delta$ is the so-called Dirac delta function (delta distribution).

In the variational formulation this can be taken into account as follows ($u_{|\partial\Omega} = 0$, $\sigma \equiv 1$ is assumed):

$$ u \in H^1_0(\Omega): \int_{\Omega} \text{grad} \, u(x) \cdot \text{grad} \, v(x) \, dx = \underbrace{v(y)}_{=: \ell(v)} \quad \forall v \in H^1_0(\Omega) . \quad (2.3.17) $$
Recall the discussion of Sect. 2.2: is the linear functional $\ell$ on the right hand side continuous w.r.t. the $H^1_0(\Omega)$-norm (= energy norm, see Def. 2.1.27) in the sense of (2.2.1)?

Consider the function $v(\mathbf{x}) = \log |\log \|\mathbf{x}\||$, $\mathbf{x} \neq 0$, on $\Omega = \{ \mathbf{x} \in \mathbb{R}^2 : \|\mathbf{x}\| < \frac{1}{2} \}$.

First, we express this function in polar coordinates $(r, \varphi)$

$$x_1 = r \cos \varphi, \quad x_2 = r \sin \varphi \quad \Rightarrow \quad v(r, \varphi) = \log |\log r|.$$  

(2.3.18)

Then we recall the expression for the gradient in polar coordinates

$$\nabla v(r, \varphi) = \frac{\partial v}{\partial r}(r, \varphi)e_r + \frac{1}{r} \frac{\partial v}{\partial \varphi}(r, \varphi)e_\varphi,$$  

(2.3.19)

where $e_r$ and $e_\varphi$ are orthogonal unit vectors in the polar coordinate directions.
Also recall integration in polar coordinates, see [29, Bsp. 8.5.3]:

\[
\int \Omega v(x) \, dx = \int_{0}^{2\pi} \int_{0}^{1/2} v(r, \phi) r \, d\phi \, dr .
\]

Using these formulas we try to compute \(|v|_{H^1(\Omega)}\),

\[
\int \Omega \|\text{grad} \, v(x)\|^2 \, dx = \int_{0}^{2\pi} \int_{0}^{1/2} \left\| -\frac{1}{\log r} e_r \right\|^2 r \, d\phi \, dr = 2\pi \int_{0}^{1/2} \frac{1}{\log^2 r} \cdot \frac{1}{r} \, dr
\]

\[
= \left[-\frac{1}{\log r}\right]_0^{1/2} = \frac{1}{\log 2} < \infty ,
\]

because the improper integral exists. This means that \(v\) has “finite elastic energy”, that is \(v \in H^1(\Omega)\), see Def. 2.2.12.

On the other hand, \(v(0) = \infty\) !

\[H^1(\Omega)\] contains unbounded functions!
Corollary 2.3.21 (Point evaluation on $H^1(\Omega)$).

The point evaluation $v \mapsto v(y), y \in \Omega$ is not a continuous linear form on $H^1(\Omega)$.

This is the mathematics behind the observation that a needle can easily prick a taut membrane: a point load leads to configurations with “infinite elastic energy”.

Another implication of Cor. 2.3.21:
The **quadratic functional** $J(u) := \int_\Omega \|\text{grad } u\|^2 \, dx - u(y), \ y \in \Omega$

is *not* bounded from below on $H^1_0(\Omega)$!

Thus, it is clear that the attempt to minimize $J$ will run into difficulties. Yet, this is the quadratic functional underlying the variational problem (2.3.17).

### 2.4 Equilibrium models: Boundary value problems
Recall the derivation of an ODE from a variational problem on a 1D domain (interval) in Sect. 1.3.3:

Tool: Integration by parts \((1.3.22)\)

This section elucidates how to extend this approach to domains \(\Omega \subset \mathbb{R}^d, d \geq 1\) (usually \(d = 2, 3\)).

Crucial issue: Integration by parts in higher dimensions?

Remember the origin of integration by parts: fundamental theorem of calculus [29, Satz 6.3.4]: for \(F \in C^1_{\text{pw}}([a, b]), a, b \in \mathbb{R}\),

\[
\int_a^b F'(x) \, dx = F(b) - F(a) ,
\]

(2.4.1)

where \(^'\) stands for differentiation w.r.t \(x\). This formula is combined with the product rule [29, Satz 5.2.1 (ii)]

\[
F(x) = f(x) \cdot g(x) \Rightarrow F'(x) = f'(x)g(x) + f(x)g'(x) .
\]

(2.4.2)

\[
\int_a^b f'(x)g(x) + f(x)g'(x) \, dx = f(b)g(b) - f(a)g(a) ,
\]

which amounts to (1.3.22).
There is a **product rule** in higher dimensions, see [29, Sect. 7.2]

**Lemma 2.4.3** (General product rule).

*For all \( \mathbf{j} \in (C^1(\Omega))^d \), \( v \in C^1(\Omega) \) holds*

\[
\text{div}(\mathbf{j}v) = v \text{div} \mathbf{j} + \mathbf{j} \cdot \text{grad} v .
\] (2.4.4)

An important **differential operator**, see [29, Def. 8.8.1]:

**divergence** of a \( C^1 \)-vector field \( \mathbf{j} = (f_1, \ldots, f_d)^T : \Omega \mapsto \mathbb{R}^d \)

\[
\text{div} \mathbf{j}(\mathbf{x}) := \frac{\partial f_1}{\partial x_1}(\mathbf{x}) + \cdots + \frac{\partial f_d}{\partial x_d}(\mathbf{x}) , \quad \mathbf{x} \in \Omega .
\]

A truly fundamental result from differential geometry provides a multidimensional analogue of the fundamental theorem of calculus:
Theorem 2.4.5 (Gauss’ theorem). \( \rightarrow [29, \text{Sect. 8.8}] \)

With \( n : \partial \Omega \mapsto \mathbb{R}^d \) denoting the exterior unit normal vectorfield on \( \partial \Omega \) and \( dS \) indicating integration over a surface, we have

\[
\int_\Omega \text{div} \, j(x) \, dx = \int_{\partial \Omega} j(x) \cdot n(x) \, dS(x) \quad \forall j \in (C^1_{pw}(\Omega))^d .
\]  

(2.4.6)

Note: In (2.4.6) integration again allows to relax smoothness requirements, cf. Sect. [1.3.2].

Theorem 2.4.7 (Green’s first formula).

For all vector fields \( j \in (C^1_{pw}(\Omega))^d \) and functions \( v \in C^1_{pw}(\Omega) \) holds

\[
\int_\Omega j \cdot \text{grad} \, v \, dx = - \int_\Omega \text{div} \, j \, v \, dx + \int_{\partial \Omega} j \cdot n \, v \, dS .
\]  

(2.4.8)
Note that the dependence on the integration variable \( x \) is suppressed in the formula (2.4.8) to achieve a more compact notation. The first Green formula could also have been written as

\[
\int_{\Omega} \mathbf{j}(x) \cdot (\nabla v)(x) \, dx = - \int_{\Omega} (\nabla \cdot \mathbf{j})(x) v(x) \, dx + \int_{\partial \Omega} \mathbf{j}(x) \cdot \mathbf{n}(x) v(x) \, dS(x). \tag{2.4.8}
\]

**Proof.** (of Thm. 2.4.7) Straightforward from Lemma 2.4.3 and Thm. 2.4.5.

Now we apply Green’s first formula to the variational problem (2.3.3), which covers the membrane model and electrostatics:

The role of \( \mathbf{j} \) in (2.4.8) is played by the vector field \( \alpha \nabla u : \Omega \rightarrow \mathbb{R}^d \).

\[
\int_{\Omega} \alpha(x) \nabla u(x) \cdot \nabla v(x) \, dx = : \mathbf{j}(x)
\]

\[\text{SAM, ETHZ}\]
\[- \int_{\Omega} \text{div}(\alpha(x) \text{grad} u(x)) \, v(x) \, dx + \int_{\partial \Omega} (\alpha(x) \text{grad} u(x)) \cdot n(x) \, v(x) \, dS(x) = 0, \]

since \( v|_{\partial \Omega} = 0 \).

Equation (2.3.3)

\[ - \int_{\Omega} \text{div}(\alpha(x) \text{grad} u(x)) \, v(x) \, dx \]

\[ + \int_{\partial \Omega} (\alpha(x) \text{grad} u(x)) \cdot n(x) \, v(x) \, dS(x) \]

\[ = 0, \quad \text{since} \quad v|_{\partial \Omega} = 0 \]

\[ = \int_{\Omega} f(x) v(x) \, dx \quad \forall v \in C^1_{0,\text{pw}}(\Omega), \]

where we have to assume that \( u, \alpha \) are sufficiently smooth: \( \alpha \text{grad} u \in C^1_{\text{pw}}(\Omega) \).

\[
\int_{\Omega} \left( \text{div}(\alpha(x) \text{grad} u(x)) + f(x) \right) \, v(x) \, dx = 0 \quad \forall v \in C^1_{0,\text{pw}}(\Omega). 
\]
Now we can invoke the multidimensional analogue of the fundamental lemma of the calculus of variations, see Lemma 1.3.23.

**Lemma 2.4.10** (Fundamental lemma of calculus of variations in higher dimensions).

If \( f \in L^2(\Omega) \) satisfies

\[
\int_{\Omega} f(x)v(x) \, dx = 0 \quad \forall v \in C^\infty_0(\Omega),
\]

then \( f \equiv 0 \) can be concluded.

\( \alpha \) \( \text{grad} \) \( u \in C^1_{pw}(\Omega) \)

**Partial differential equations (PDE)**

\[
- \text{div}(\alpha(x) \text{grad} u) = f \quad \text{in} \ \Omega.
\]
Again, for the sake of brevity, dependence \( \nabla u = \nabla u(x), f = f(x) \) is not made explicit in the PDE in (2.4.11).

**Remark 2.4.12 (Laplace operator).**

If \( \alpha \) agrees with a positive *constant*, by rescaling of (2.5.6) we can achieve

\[
-\Delta u = f \quad \text{in} \; \Omega .
\]  

\( \Delta = \text{div} \circ \nabla = \frac{\partial^2}{\partial x_1^2} + \frac{\partial^2}{\partial x_2^2} + \frac{\partial^2}{\partial x_3^2} \) = Laplace operator

(2.4.13) is called *Poisson equation*, \( \Delta u = 0 \) in \( \Omega \) is called *Laplace equation*
Finally:

\[ - \text{div}(\alpha(x) \text{grad} u) = f \quad \text{in } \Omega, \quad u = g \quad \text{on } \partial \Omega. \quad (2.4.14) \]

\[ (2.4.14) = \text{second-order elliptic BVP with Dirichlet boundary conditions} \]

Short name for BVPs of the type (2.4.14): “Dirichlet problem”

**Remark 2.4.15 (Extra smoothness requirement for PDE formulation).**

Same situation as in Sect. 1.3.3, cf. Assumption (1.3.21):

Transition from variational equation to PDE requires extra assumptions on smoothness of solution and coefficients.
Remark 2.4.16 (Membrane with free boundary values).

(Graph description of membrane shape by $u : \Omega \mapsto \mathbb{R}$, see Sect. 2.1.1)

Now: membrane clamped only on a part $\Gamma_0 \subset \partial \Omega$ of its edge.

--- : prescribed boundary values here ($\Gamma_0$)

--- : “free boundary”

Figure 56:

Configuration space

$V := \{ u \in H^1(\Omega) : u|_{\Gamma_0} = g \} \rightarrow$ Def. 2.2.12

Total potential energy as in (2.1.3):

$$J_M(u) := \int_\Omega \frac{1}{2} \sigma(x) \|\text{grad} \ u\|^2 - f(x) u(x) \, dx.$$ (2.1.3)
test space in variational formulation

\[ V_0 := \{ u \in H^1(\Omega) : u|_{\Gamma_0} = 0 \} \]

(Remember: test space comprises “admissible perturbations” of configurations, cf. Sect. 1.3.1)

Variational formulation, c.f. (2.3.2)

\[
\begin{align*}
\quad u &\in H^1(\Omega), & \quad &\int_{\Omega} \sigma(x) \nabla u(x) \cdot \nabla v(x) \, dx = \int_{\Omega} f(x) v(x) \quad \forall v \in V_0. \\
\quad u = g \text{ on } \Gamma_0 &
\end{align*}
\]

Application of Green’s first formula (2.4.8) to (2.4.17) leads to

\[
\begin{align*}
- \int_{\Omega} &\left( \text{div}(\sigma(x) \nabla u(x)) + f(x) \right) v(x) \, dx \\
+ \int_{\partial \Omega \setminus \Gamma_0} &\left( (\sigma(x) \nabla u(x)) \cdot n(x) \right) v(x) \, dS(x) \quad \forall v \in V_0.
\end{align*}
\]

Note that, unlike above, the boundary integral term cannot be dropped entirely, because \( v \neq 0 \) on \( \partial \Omega \setminus \Gamma_0 \).

Assumption (→ Rem. 2.4.15): extra smoothness \( u \in C_{pw}^2(\Omega), \sigma \in C_{pw}^1(\Omega) \)
How to deal with the boundary term?

Idea: ① First restrict test function $v$ to $C^\infty_0(\Omega)$

$\Rightarrow$ Boundary term vanishes!

Then, apply Lemma \ref{lemma:2.4.10}

\[
\text{div}(\sigma(x) \nabla u(x)) + f(x) = 0 \quad \text{in } \Omega.
\] (2.4.19)

② Then test with generic $v \in V_0$, while making use of (2.4.19):

\[
\int_{\partial\Omega \setminus \Gamma_0} ((\sigma(x) \nabla u(x)) \cdot n(x)) v(x) \, dS(x) = 0 \quad \forall v \in V_0.
\]

Lemma \ref{lemma:2.4.10} on $\partial\Omega \setminus \Gamma_0$

\[
(\sigma(x) \nabla u(x)) \cdot n(x) = 0 \quad \text{on } \partial\Omega \setminus \Gamma_0.
\] (2.4.20)

When removing pinning conditions on $\partial\Omega \setminus \Gamma_0$ the equilibrium conditions imply the (homogeneous) **Neumann boundary conditions** $(\sigma(x) \nabla u(x)) \cdot n(x) = 0$ on $\partial\Omega \setminus \Gamma_0$. 
Boundary value problem for membrane clamped at $\Gamma_0 \subset \partial \Omega$

$$- \text{div}(\sigma(x) \text{grad } u) = f \quad \text{in } \Omega, \quad u = g \quad \text{on } \Gamma_0, \quad (\sigma(x) \text{grad } u) \cdot n = 0 \quad \text{on } \partial \Omega \setminus \Gamma_0.$$  \hspace{1cm} (2.4.21)

(2.4.21) = Second-order elliptic BVP with Neumann boundary conditions on $\partial \Omega \setminus \Gamma_0$

Short name for BVPs of the type (2.4.21): “Mixed Neumann–Dirichlet problem”

2.5 Diffusion models (Stationary heat conduction)

Now we look at a class of physical phenomena, for which models are based on two building blocks

1. a conservation principle (of mass, energy, etc.),
2. a potential driven flux of the conserved quantity.

Mathematical modelling for these phenomena naturally involves partial differential equations in the first steps, which are supplemented with boundary conditions. Hence, second-order elliptic boundary value problems arise first, while variational formulations are deduced from them, thus reversing the order of steps followed for equilibrium models in Sects. 2.1–2.4.

In order to keep the presentation concrete, the discussion will target heat conduction, about which everybody should have a sound “intuitive grasp”.

notation: $\Omega \subset \mathbb{R}^3$: bounded open region occupied by solid object ($\hat{\Omega} = \Omega \rightarrow$ computational domain)

Fundamental concept: heat flux, modelled by vector field $\mathbf{j} : \Omega \mapsto \mathbb{R}^3$
Heat flux = power flux: \[ \mathbf{j} = \frac{W}{m^2} \]

Vector field \( \mathbf{j} : \Omega := [0, 1]^2 \rightarrow \mathbb{R}^3 \)

Normal vector \( \mathbf{n} \)

Total heat flux through oriented surface \( \Sigma \subset \mathbb{R}^3 \)

Power \( P_{\Sigma} = \int_{\Sigma} \mathbf{j} \cdot \mathbf{n} \, dS \). \( (2.5.1) \)

\( P_{\Sigma} \) ([\( P_{\Sigma} \] = 1W): directed total power flowing through the oriented surface \( \Sigma \) per unit time. Note that the sign of \( P_{\Sigma} \) will change when flipping the normal of \( \Sigma \)!
Conservation of energy

\[ \int_{\partial V} \mathbf{j} \cdot \mathbf{n} \, dS = \int_{V} f \, dx \quad \text{for all “control volumes” } V. \quad (2.5.2) \]

- Power flux through surface of \( V \)
- Heat production inside \( V \)

\( f = \text{heat source/sink} \left( [f] = \frac{W}{m^3} \right) \), \( f = f(x) \) and \( f \) can be discontinuous \( (f \in C^0_{pw}(\Omega)) \)

Intuition:
- Heat flows from hot zones to cold zones
- The larger the temperature difference, the stronger the heat flow

**Experimental evidence** supports this intuition and, for many materials, yields the following quantitative relationship:

**Fourier’s law**

\[ \mathbf{j}(x) = -\kappa(x) \nabla u(x), \quad x \in \Omega. \quad (2.5.3) \]
Meaning of quantities:

\[ j = \text{heat flux} \quad ([j] = 1 \text{W/m}^2) \]
\[ u = \text{temperature} \quad ([u] = 1 \text{K}) \]
\[ \kappa = \text{heat conductivity} \quad ([\kappa] = 1 \text{W/Km}) \]

(2.5.3) \Rightarrow \text{Heat flow from hot to cold regions linearly proportional to gradient of temperature}

Some facts about the heat conductivity:

\[ \kappa: \quad \bullet \kappa = \kappa(x) \text{ for non-homogeneous materials (spatially varying heat conductivity)} \]
\[ \bullet \kappa \text{ can even be discontinuous for composite materials} \]
\[ \bullet \kappa \text{ may be } \mathbb{R}^{3,3}-\text{valued (heat conductivity tensor)} \]

The most general form of the heat conductivity (tensor) enjoys the very same properties as the dielectric tensor introduced in Sect. 2.1.2:
From thermodynamic principles, cf. (2.1.8):

\[
\exists \kappa^-, \kappa^+ > 0: \quad 0 < \kappa^- \leq \kappa(x) \leq \kappa^+ < \infty \quad \text{for almost all } x \in \Omega .
\] (2.5.4)

Terminology: (2.5.4) \(\iff\) \(\kappa\) is bounded and uniformly positive, see Def. 2.1.11.

From 2.5.2 by Gauss’ theorem Thm. 2.4.5

\[
\int_V \text{div } j(x) \, dx = \int_V f(x) \, dx \quad \text{for all “control volumes” } \ V \subset \Omega .
\]

Now appeal to another version of the fundamental lemma of the calculus of variations, see Lemma 2.4.10, this time sporting piecewise constant test functions.

Local form of energy conservation:

\[
\text{div } j = f \quad \text{in } \Omega .
\] (2.5.5)
Combine equations (2.5.5) & (2.5.3)

\[ j = -\kappa(x) \text{grad} \, u \quad (2.5.3) \]

\[ + \quad \text{div} \, j = f \quad (2.5.5) \]

\[ - \text{div}(\kappa(x) \text{grad} \, u) = f \quad \text{in} \, \Omega \quad (2.5.6) \]

*Linear* scalar second order elliptic PDE (for unknown temperature \( u \))

2.6 Boundary conditions
In the examples from Sects. 2.1.1, 2.1.2 we fixed the value of the unknown function \( u : \Omega \mapsto \mathbb{R} \) on the boundary \( \partial \Omega \): Dirichlet boundary conditions in (2.4.14).

Exception: free edge of taut membrane, see Rem. 2.4.16: Neumann boundary conditions in (2.4.21).

In this section we resume the discussion of boundary conditions and examine them for stationary heat conduction, see previous section. This has the advantage that for this everyday physical phenomenon boundary conditions have a very clear intuitive meaning.

Boundary conditions on surface/boundary \( \partial \Omega \) of \( \Omega \):

(i) Temperature \( u \) is fixed: with \( g : \partial \Omega \mapsto \mathbb{R} \) prescribed

\[
  u = g \quad \text{on} \quad \partial \Omega .
\]  

\[ (2.6.1) \]

Dirichlet boundary conditions

(ii) Heat flux \( j \) through \( \partial \Omega \) is fixed: with \( h : \partial \Omega \mapsto \mathbb{R} \) prescribed (\( n : \partial \Omega \mapsto \mathbb{R}^3 \) exterior unit normal vectorfield) on \( \partial \Omega \)

\[
  j \cdot n = -h \quad \text{on} \quad \partial \Omega .
\]

\[ (2.6.2) \]
(iii) Heat flux through \( \partial \Omega \) depends on (local) temperature: with increasing function \( \Psi : \mathbb{R} \rightarrow \mathbb{R} \)

\[
\mathbf{j} \cdot \mathbf{n} = \Psi(u) \quad \text{on} \quad \partial \Omega
\]  

(2.6.3)

radiation boundary conditions

**Example 2.6.4 (Convective cooling (simple model)).**

Heat is carried away from the surface of the body by a fluid at bulk temperature \( u_0 \). A crude model assumes that the heat flux depends *linearly* on the temperature difference between the surface of \( \Omega \) and the bulk temperature of the fluid.

\[
\mathbf{j} \cdot \mathbf{n} = q(u - u_0) \quad \text{on} \quad \partial \Omega \quad \text{where} \quad 0 < q^- \leq q(x) \leq q^+ < \infty \quad \text{for almost all} \quad x \in \partial \Omega.
\]
Example 2.6.5 (Radiative cooling (simple model)).

A hot body emits electromagnetic radiation (blackbody emission), which drains thermal energy. The radiative energy loss is roughly proportional to the 4th power of the temperature difference between the surface temperature of the body and the ambient temperature.

\[ \mathbf{j} \cdot \mathbf{n} = \alpha |u - u_0|(u - u_0)^3 \quad \text{on} \quad \partial \Omega, \quad \text{with} \quad \alpha > 0 \]

→ Non-linear boundary condition

Terminology: If \( g = 0 \) or \( h = 0 \) → homogeneous Dirichlet or Neumann boundary conditions

Remark 2.6.6 (Mixed boundary conditions).

Different boundary conditions can be prescribed on different parts of \( \partial \Omega \) (→ mixed boundary conditions, cf. Rem. 2.4.16)
Example 2.6.7 ("Wrapped rock on a stove").

- Non-homogeneous Dirichlet boundary conditions on $\Gamma_D \subset \partial \Omega$
- Homogeneous Neumann boundary conditions on $\Gamma_N \subset \partial \Omega$
- Convective cooling boundary conditions on $\Gamma_R \subset \partial \Omega$

Partition: $\partial \Omega = \overline{\Gamma_D} \cup \overline{\Gamma_N} \cup \overline{\Gamma_R}$, $\Gamma_D, \Gamma_N, \Gamma_R$ mutually disjoint

\[ -\text{div}(\kappa(x) \nabla u) = f + \text{boundary conditions} \Rightarrow \text{elliptic boundary value problem (BVP)} \]

For second order elliptic boundary value problems exactly one boundary condition is needed on any part of $\partial \Omega$.

\[ \nabla \cdot (\kappa(x) \nabla u) = f \]

Remark 2.6.8 (Linear BVP).

Observe that the solution mapping \( \begin{pmatrix} f \\ g \end{pmatrix} \mapsto u \) for (2.5.6), (2.6.1) is linear.

This means that if $u_i$ solves the Dirichlet problem with source function $f_i$ and Dirichlet data $g_i$, $i = 1, 2$, then $u_1 + u_2$ solves (2.5.6) & (2.6.1) for source $f_1 + f_2$ and boundary values $g_1 + g_2$. 

R. Hiptmair
C. Schwab, H. Harbrecht
V. Gradinaru
A. Chernov
SAM, ETHZ
2.7 Characteristics of elliptic boundary value problems

Qualitative insights gained from heat conduction model:

- **continuity**: the temperature $u$ must be continuous (jump in $u \rightarrow j = \infty$).

- normal component of $j$ across surfaces inside $\Omega$ must be continuous (jump in $j \cdot n \rightarrow$ heat source $f$ of infinite intensity).

- **interior smoothness** of $u$: $u$ smooth where $f$ and $D$ smooth.

- **non-locality**: local alterations in $f, g, h$ affect $u$ everywhere in $\Omega$.

- **quasi-locality**: If local changes in $f, g, h$ confined to $\Omega' \subset \Omega$, their effects decay away from $\Omega'$. 
maximum principle: (in the absence of heat sources extremal temperatures are on the boundary)

if \( f \equiv 0 \), then
\[
\inf_{y \in \partial \Omega} u(y) \leq u(x) \leq \sup_{y \in \partial \Omega} u(y) \quad \text{for all } x \in \Omega
\]

Typical features of solutions of elliptic boundary value problems

Example 2.7.1 (Scalar elliptic boundary value problem in one space dimension).

Poisson equation \( \rightarrow (2.4.13) \) in 1D:
\[-u'' = f\]

\( f \) discontinuous, piecewise \( C^0 \) \( \Rightarrow u \in C^1 \), piecewise \( C^2 \)

Example 2.7.2 (Smoothness of solution of scalar elliptic boundary value problem).

\[-\Delta u = f(x) \quad \text{in } \Omega := ]0, 1[^2, \quad u = 0 \quad \text{on } \partial \Omega, \quad (2.7.3)\]

\( f(x) := \text{sign}(\sin(2\pi k_1 x_1) \sin(2\pi k_2 x_2)), \quad x \in \Omega, \quad k_1, k_2 \in \mathbb{N} \).

Approximate solution computed by means of linear Lagrangian finite elements + lumping
(\( \rightarrow \) Ch. [3], details in Sect. [3.2, 3.5.4])
Source term $f(x)$, $k_1 = k_2 = 2$

“Smooth” $u$ despite “rough” $f$!

Example 2.7.4 (Quasi-locality of solution of scalar elliptic boundary value problem).

$$-\Delta u = f_\delta(x) \quad \text{in } \Omega := [0,1]^2, \quad u = 0 \quad \text{on } \partial \Omega,$$ 

(2.7.5)
\[ f_\delta(x) = \begin{cases} 
\delta^{-2}, & \text{if } \|x - (1/2)\| \leq \delta, \\
0, & \text{elsewhere.}
\end{cases} \quad \delta > 0. \]
2.8 Second-order elliptic variational problems

In Ch. 1 and Sects. 2.1–2.4 we pursued the derivation:

Minimization problem (e.g., (2.1.4), (2.1.14)) ➢ Variational problem (e.g., (2.3.1), (2.3.2)) ➢ BVP for PDE (e.g., (2.4.14), (2.4.21))

Now we are proceeding in the opposite direction:

PDE (e.g., (2.5.6)) + boundary conditions (e.g., (2.6.1), (2.6.2), (2.6.3)) ➢ variational problem

Formal approach:

STEP 1: \textit{test PDE with smooth functions}

(do not test, where the solution is known, e.g., on the boundary)
STEP 2: \textit{integrate over domain}

STEP 3: \textit{perform integration by parts}

(e.g. by using Green's first formula, Thm. 2.4.7)

STEP 4: \texttt{[optional] incorporate boundary conditions into boundary terms}

Example 2.8.1 (Variational formulation for heat conduction with Dirichlet boundary conditions).

BVP: \[- \text{div}(\kappa(x) \text{grad } u) = f \quad \text{in } \Omega, \quad u = g \quad \text{on } \partial \Omega.\] (2.8.2)

STEP 1 & 2:

\text{test with } v \in C_0^\infty(\Omega)

\[\Rightarrow - \int_{\Omega} \text{div}(\kappa(x) \text{grad } u) v \, dx = \int_{\Omega} f v \, dx.\] (2.8.3)

Note: \(v|_{\partial \Omega} = 0\) for test function, because \(u\) already fixed on \(\partial \Omega\).
STEP 3: use Green's formula from Thm. 2.4.7 on \( \Omega \subset \mathbb{R}^d \) (multidimensional integration by parts): Apply (2.4.8) to (2.8.3) with \( j := \kappa(x) \text{grad} u \):

\[
\int_{\Omega} \kappa(x) \text{grad} u \cdot \text{grad} v \, dx - \int_{\partial \Omega} \kappa(x) \text{grad} u \cdot n \, v \, dS = \int_{\Omega} f v \, dx \quad \forall v \in C_0^\infty(\Omega). 
\]

This gives the variational formulation after we switch to “maximal admissible function spaces” (Sobolev spaces, see Sect. 2.2, as spaces of functions with finite energy)

Variational form of (2.8.2): seek

\[
\begin{align*}
&u \in H^1(\Omega) : \quad \int_{\Omega} \kappa(x) \text{grad} u \cdot \text{grad} v \, dx = \int_{\Omega} f v \, dx \quad \forall v \in H_0^1(\Omega). 
\end{align*}
\]

(2.8.4)

Example 2.8.5 (Variational formulation: heat conduction with general radiation boundary conditions).
BVP: \(- \text{div}(\kappa(x) \text{grad } u) = f \) in \(\Omega\), \(- \kappa(x) \text{grad } u \cdot n = \Psi(u)\) on \(\partial \Omega\). \hspace{1cm} (2.8.6)

STEP 1 & 2: \(u|_{\partial \Omega}\) not fixed \(\Rightarrow\) test with \(v \in C^\infty(\overline{\Omega})\)

\[
- \int_\Omega \text{div}(\kappa(x) \text{grad } u) \, v \, d\mathbf{x} = \int_\Omega f v \, d\mathbf{x} \quad \forall v \in C^\infty(\overline{\Omega}).
\]

STEP 3 & 4: apply Green’s first formula (2.4.8) and incorporate boundary conditions:

\[
\int_\Omega \kappa(x) \text{grad } u \cdot \text{grad } v \, d\mathbf{x} - \int_{\partial \Omega} \kappa(x) \text{grad } u \cdot n \, v \, dS = \int_\Omega f v \, d\mathbf{x} \quad \forall v \in C^\infty(\overline{\Omega}).
\]

\(-=\Psi(u)\) (STEP 4)

**Variational formulation** of (2.8.6): seek

\(u \in H^1(\Omega)\): \[
\int_\Omega \kappa(x) \text{grad } u \cdot \text{grad } v \, d\mathbf{x} + \int_{\partial \Omega} \Psi(u) \, v \, dS = \int_\Omega f v \, d\mathbf{x} \quad \forall v \in H^1(\Omega).
\]

(2.8.7)
Theorem 2.8.8. If $\kappa \in C^1(\Omega)$, classical solutions $u \in C^2(\Omega)$ of the boundary value problems (2.8.2) and (2.8.6) also solve the associated variational problems.

Proof. Apply Theorem 2.4.7 as in the derivation of the weak formulations.

Example 2.8.9 (Variational formulation for Neumann problem).

2nd-order elliptic (inhomogeneous) Neumann problem

BVP: 

$$
- \text{div}(\kappa(x) \text{grad } u) = f \quad \text{in } \Omega, \\
\kappa(x) \text{grad } u \cdot n = h(x) \quad \text{on } \partial \Omega. 
$$

(2.8.10)

We confront Neumann boundary conditions (2.6.2) (prescribed heat flux) on the whole boundary.

Variational formulation derived as in Ex. 2.8.5, with $\Psi(u) = -h$.

$$
u \in H^1(\Omega): \quad \int_{\Omega} \kappa(x) \text{grad } u \cdot \text{grad } v \, dx - \int_{\partial \Omega} h v \, dS = \int_{\Omega} f v \, dx \quad \forall v \in H^1(\Omega). 
$$

(2.8.11)
Observation: when we test (2.8.7) with $v \equiv 1$ 

$$- \int_{\partial \Omega} h \, dS = \int_{\Omega} f \, dx$$

(2.8.12)

This is a compatibility condition for the existence of (variational) solutions of the Neumann problem!

Interpretation of (2.8.12) against the backdrop of the stationary heat conduction model:

conservation of energy $\rightarrow$ (2.5.2): Heat generated inside $\Omega$ ($\leftrightarrow f$) must be offset by heat flux through $\partial \Omega$ ($\rightarrow h$).

Remark 2.8.13 (Uniqueness of solutions of Neumann problem).
Observation: if compatibility condition (2.8.12) holds true, then

\[ v \in H^1(\Omega) \text{ solves (2.8.7)} \iff v + \gamma \text{ solves (2.8.7)} \quad \forall \gamma \in \mathbb{R}, \]

we say, “the solution is unique only up to constants”.

Complementary observation: \( a(u, v) := \int_\Omega \kappa(x) \text{grad} u \cdot \text{grad} v \, dx \) is not s.p.d (→ Def. 2.1.25) on \( H^1(\Omega) \).

Idea: Restore uniqueness of solutions by enforcing average temperature to be zero \( \int_\Omega u(x) \, dx = 0 \).

This amounts to posing the variational problem (2.8.7) over the constrained function space

\[ H^1_\ast(\Omega) := \{ v \in H^1(\Omega) : \int_\Omega v(x) \, dx = 0 \}. \quad (2.8.14) \]

The norm on \( H^1_\ast(\Omega) \) is the same as on \( H^1_0(\Omega) \), see Def. 2.2.12. Obviously (why ?), the norm property (N1) is satisfied. These arguments also show that \( a \) is s.p.d (→ Def. 2.1.25) on \( H^1_\ast(\Omega) \), cf. Thm. 2.9.6.
Variational formulation of Neumann problem:

\[ u \in H_1^*(\Omega): \int_{\Omega} \kappa(x) \nabla u \cdot \nabla v \, dx = \int_{\Omega} f v \, dx + \int_{\partial \Omega} h v \, dS \quad \forall v \in H_1^*(\Omega). \quad (2.8.15) \]

2.9 Essential and natural boundary conditions

Synopsis:

- 2nd-order elliptic Dirichlet problem:

\[ -\text{div}(\alpha(x) \nabla u) = f \quad \text{in} \ \Omega, \quad u = g \quad \text{on} \ \partial \Omega. \quad (2.4.14) \]
with variational formulation
\[ u \in H^1(\Omega) : \int_{\Omega} (\alpha(x) \text{grad} \, u(x)) \cdot \text{grad} \, v(x) \, dx = \int_{\Omega} f(x) \, v(x) \, dx \quad \forall v \in H^1_0(\Omega). \quad (2.3.3) \]

\[ \text{2nd-order elliptic Neumann problem:} \]
\[ -\text{div}(\alpha(x) \text{grad} \, u) = f \quad \text{in} \ \Omega, \quad (\alpha(x) \text{grad} \, u) \cdot n = -h \quad \text{on} \ \partial \Omega. \quad (2.9.1) \]

with variational formulation
\[ u \in H^1_*(\Omega) : \int_{\Omega} \alpha(x) \text{grad} \, u \cdot \text{grad} \, v \, dx = \int_{\Omega} f v \, dx + \int_{\partial \Omega} h v \, ds \quad \forall v \in H^1_*(\Omega). \quad (2.8.15) \]

\[ \text{2nd-order elliptic mixed Neumann-Dirichlet problem, see Rem. 2.4.16:} \]
\[ -\text{div}(\alpha(x) \text{grad} \, u) = f \quad \text{in} \ \Omega, \quad u = g \quad \text{on} \ \Gamma_0 \subset \partial \Omega, \quad (\alpha(x) \text{grad} \, u) \cdot n = -h \quad \text{on} \ \partial \Omega \setminus \Gamma_0. \quad (2.9.2) \]

with variational formulation
\[ u \in H^1(\Omega) : \int_{\Omega} (\alpha(x) \text{grad} \, u(x)) \cdot \text{grad} \, v(x) \, dx = \int_{\Omega} f(x) v(x) \, dx + \int_{\partial \Omega \setminus \Gamma_0} h v \, ds \quad (2.9.3) \]

\[ \text{for all} \ v \in H^1(\Omega) \ \text{with} \ v|_{\Gamma_0} = 0. \]
In the variational formulations of 2nd-order elliptic BVPs of Sect. 2.8:

**Dirichlet boundary conditions** are *directly imposed* on trial space and (in homogeneous form) on test space.

Terminology: *essential boundary conditions*

**Neumann boundary conditions** are enforced *only* through the variational equation.

Terminology: *natural boundary conditions*

The attribute “natural” has been coined, because Neumann boundary conditions “naturally” emerge when removing constraints on the boundary, as we have seen for the partially free membrane of Rem. 2.4.16.

*Remark 2.9.4 (Admissible Dirichlet data).*
Requirement for “Dirichlet data” $g : \partial \Omega \mapsto \mathbb{R}$ in (2.4.14):

there is $u \in H^1(\Omega)$ such that $u|_{\partial \Omega} = g$

Analogous to Thm. 2.2.17:

If $g : \partial \Omega \mapsto \mathbb{R}$ is piecewise continuously differentiable (and bounded with bounded piecewise derivatives), then it can be extended to an $u_0 \in H^1(\Omega)$, if and only if it is continuous on $\partial \Omega$.

Bottom line: Dirichlet boundary values have to be continuous

This is also stipulated by physical insight, e.g. in the case of the taut membrane model of Sect. 2.1.1: discontinuous displacement on $\partial \Omega$ would entail ripping apart the membrane.
Remark 2.9.5 (Admissible Neumann data).

In the variational problem (2.8.15) Neumann data \( h : \partial \Omega \mapsto \mathbb{R} \) enter through the linear form on the right hand side

\[
\ell(v) := \int_{\Omega} f(\mathbf{x}) v(\mathbf{x}) \, d\mathbf{x} + \int_{\partial \Omega} h(\mathbf{x}) v(\mathbf{x}) \, dS(\mathbf{x}) .
\]

Remember the discussion in the beginning of Sect. 2.2, also Rem. 2.3.16: we have to establish that \( \ell \) is continuous on \( H^1_*(\Omega) \) defined in (2.8.14). This is sufficient, because the coefficient function \( \kappa \) is uniformly positive and bounded, see (2.5.4). Thus, the energy \( \| \cdot \|_a \) associated with the bilinear form

\[
a(u, v) = \int_{\Omega} \kappa(\mathbf{x}) \text{grad} u \cdot \text{grad} v \, d\mathbf{x}
\]

can be bounded from above and below by \( \| \cdot \|_{H^1(\Omega)} \), cf. the estimate (2.3.14).

Theorem 2.9.6. Second Poincaré-Friedrichs inequality]

If \( \Omega \subset \mathbb{R}^d \), \( d \in \mathbb{N} \), is bounded, then

\[
\exists C = C(\Omega) > 0: \quad \| u \|_0 \leq C \text{ diam}(\Omega) \| \text{grad} u \|_0 \quad \forall u \in H^1_*(\Omega) .
\]
notation: \( C = C(\Omega) \) indicates that the constant \( C \) may depend on the shape of the domain \( \Omega \).

**Proof.** (for \( d = 1, \Omega = [0, 1] \) only, technically difficult in higher dimensions, see [5, Thm. 1.6.6])

As in the proof of Thm. 2.2.16, we employ a density argument and assume that \( u \) is sufficiently smooth, \( u \in C^1([0, 1]) \).

By the fundamental theorem of calculus (2.4.1)

\[
  u(x) = u(y) + \int_y^x \frac{du(\tau)}{dx} \, d\tau, \quad 0 \leq x, y \leq 1.
\]

Then use the Cauchy-Schwarz inequality (2.2.15)

\[
  u(x)^2 \leq \int_0^1 \int_y^x 1 \, d\tau \, dy \int_0^x \int_y^x \left| \frac{du(\tau)}{dx} \right|^2 \, d\tau \, dy \leq \int_0^1 \left| \frac{du}{dx}(\tau) \right|^2 \, d\tau.
\]
Integrate over Ω yields the estimate

\[ \|u\|_0^2 = \int_0^1 u^2(x) \, dx \leq \int_0^1 \left| \frac{du}{dx}(\tau) \right|^2 \, d\tau = |u|_{H^1(\Omega)}^2. \]  

By (2.2.15), Thm. 2.9.6 implies the continuity of the first term in ℓ.

Continuity of the boundary contribution to ℓ hinges on a trace theorem

**Theorem 2.9.7** (Multiplicative trace inequality).

\[ \exists C = C(\Omega) > 0: \quad \|u\|_{L^2(\partial\Omega)}^2 \leq C \|u\|_{L^2(\Omega)} \cdot \|u\|_{H^1(\Omega)} \quad \forall u \in H^1(\Omega). \]

**Proof.** (for \( d = 1, \Omega = [0, 1] \) only, technically difficult in higher dimensions)

As in the proof of Thms. 2.2.16, 2.9.6, we employ a density argument and assume that \( u \) is sufficiently smooth, \( u \in C^1([0, 1]) \).
By the fundamental theorem of calculus (2.4.1):

\[ u(1)^2 = \int_0^1 \frac{dw}{d\xi}(x) \, dx , \quad \text{with} \quad w(\xi) := \xi u^2(\xi) , \]

\[ u(1)^2 = \int_0^1 u^2(x) + 2u(x)\frac{du}{dx}(x)x \, dx . \]

Then use the Cauchy-Schwarz inequality (2.2.15)

\[ u(1)^2 \leq \int_0^1 u^2(x) \, dx + 2 \int_0^1 |x| |u(x)| \left| \frac{du}{dx}(x) \right| \, dx \leq \|u\|^2_0 + 2 \|u\|_0 \left\| \frac{du}{dx} \right\|_0 . \]

A similar estimate holds for \( u(0)^2 \).

Now we can combine

- the Cauchy-Schwarz inequality (2.2.15) on \( \partial \Omega \),
the 2nd Poincaré-Friedrichs inequality of Thm. 2.9.6,
the multiplicative trace inequality of Thm. 2.9.7:

\[ \int_{\partial \Omega} hv \, dS \leq \|h\|_{L^2(\partial \Omega)} \|v\|_{L^2(\partial \Omega)} \leq \|h\|_{L^2(\partial \Omega)} \|v\|_{H^1(\Omega)} \leq \|h\|_{L^2(\partial \Omega)} |v|_{H^1(\Omega)} \quad \forall v \in H^1_*(\Omega). \]

\( h \in L^2(\partial \Omega) \) provides valid Neumann data for the 2nd order elliptic BVP (2.9.1).

In particular Neumann data \( h \) can be \textit{discontinuous}.  

R. Hiptmair
C. Schwab,
H. Harbrecht
V. Gradinaru
A. Chernov
SAM, ETHZ

2.9
p. 268
Finite Element Methods (FEM)

In this chapter:

Problem: linear scalar second-order elliptic boundary value problem $\rightarrow$ Ch. 2
Perspective: variational interpretation in Sobolev spaces $\rightarrow$ Sect. 2.8
Objective: algorithm for the computation of an approximate numerical solution

Preface

Sect. 1.5.1 introduced the fundamental ideas of the Galerkin discretization of variational problems, or, equivalently, of minimization problems, posed over function spaces. A key ingredient are suitably chosen finite-dimensional trial and test spaces, equipped with ordered bases.
In Sect. 1.5.1.2 the abstract approach was discussed for two-point boundary value problems and the concrete case of piecewise linear trial and test spaces, built upon a partition (mesh/grid) of the interval (domain). In this context the locally supported tent functions lent themselves as natural basis functions.

This chapter is devoted to extending the linear finite element method in 1D to

- 2nd-order linear variational problems on bounded spatial domains $\Omega$ in two and three dimensions,
- piecewise polynomial trial/test functions of higher degree.

The leap from $d = 1$ to $d = 2$ will encounter additional difficulties and many new aspects will emerge. This chapter will elaborate on them and present policies how to tackle them.

Throughout, we will restrict ourselves to linear 2nd-order elliptic variational problems on spatial domains $\Omega \in \mathbb{R}^d$, $d = 2, 3$, with the properties listed in Rem. 2.1.1.

**2nd-order elliptic Dirichlet problem:**

$$u \in H^1(\Omega), \quad \int_{\Omega} (\alpha(x) \text{grad} u(x)) \cdot \text{grad} v(x) \, dx = \int_{\Omega} f(x)v(x) \, dx \quad \forall v \in H^1_0(\Omega), \quad (2.3.3)$$
with continuous (→ Rem. 2.9.4) Dirichlet data \( g \in C^0(\partial \Omega) \).

2nd-order elliptic Neumann problems:

\[
\int_{\Omega} (\alpha(x) \, \text{grad} \, u) \cdot \text{grad} \, v \, dx = \int_{\Omega} f v \, dx + \int_{\partial \Omega} h v \, ds \quad \forall v \in H^1_*(\Omega),
\]

with piecewise continuous (→ Rem. 2.9.5) Neumann data \( h \in C^0_{pw}(\partial \Omega) \) that satisfy the compatibility condition (2.8.12).

A simpler version with homogeneous Neumann data and reaction term:

\[
\int_{\Omega} \alpha(x) \, \text{grad} \, u \cdot \text{grad} \, v + c(x) u v \, dx = \int_{\Omega} f v \, dx \quad \forall v \in H^1(\Omega),
\]

with reaction coefficient \( c : \Omega \mapsto \mathbb{R}^+ \), \( c \in C^0_{pw}(\Omega) \). Note that no compatibility conditions is required in this case.

Rem. 1.5.3 still applies: all functions (coefficient \( \alpha \), source function \( f \), Dirichlet data \( g \)) may be given only in procedural form. Recall the discussion of the consequences in Rem. 1.5.36 and Sect. 1.5.1.2.
3.1 Galerkin discretization

Recall the concept of “discretization”, see Sect. 1.5:

Not a moot point: any computer can only handle a finite amount of information (reals)

Variational boundary value problem $\Rightarrow$ System of a finite number of equations for (real) unknowns

Targetted: **linear variational problem** (1.4.6)

\[ u \in V_0: \quad a(u, v) = \ell(v) \quad \forall v \in V_0 , \quad (3.1.1) \]

$V_0 \triangleq$ vector space (Hilbert space) (usually a Sobolev space $\rightarrow$ Sect. 2.2) with norm $\| \cdot \|_V$,

\[ a(\cdot, \cdot) \triangleq \text{bilinear form}, \text{ continuous on } V_0, \text{ which means} \]

\[ \exists C > 0: \quad |a(u, v)| \leq C \| u \|_V \| v \|_V \quad \forall u, v \in V . \quad (3.1.2) \]
\[ \ell \ \hat{=} \ \text{continuous linear form in the sense of, cf. (2.2.1)}, \]
\[ \exists C > 0: \ |\ell(v)| \leq C \|v\|_V \quad \forall v \in V_0. \tag{3.1.3} \]

If \( a \) is symmetric and positive definite (\( \rightarrow \) Def. 2.1.25), we may choose \( \|\cdot\|_V := \|\cdot\|_a \), "energy norm"; see Def. 2.1.27. Continuity of \( a \) w.r.t. \( \|\cdot\|_a \) is clear.

Recall from Sect. 1.5.1:

**Idea of Galerkin discretization**

Replace \( V_0 \) in (3.1.1) with a finite dimensional subspace. 
\( (V_0,N \subset V_0 \text{ called Galerkin (or discrete) trial space/test space}) \)
Twofold nature of symbol “$N$”:

- $N = \text{formal index, tagging “discrete entities” (} \rightarrow \text{“finite amount of information”)}$
- $N = \dim V_{N,0} = \text{dimension of Galerkin trial/test space}$

Discrete variational problem, *cf. (1.5.7)*,

$$u_N \in V_{0,N}: \quad a(u_N, v_N) = \ell(v_N) \quad \forall v_N \in V_{0,N}.$$  \hspace{1cm} (3.1.4)
Theorem 3.1.5 (Existence and uniqueness of solutions of discrete variational problems).

If the \textit{bilinear form} \( a : V_0 \times V_0 \mapsto \mathbb{R} \) is symmetric and positive definite (\( \rightarrow \) Def. 2.1.25) and the \textit{linear form} \( \ell : V_0 \mapsto \mathbb{R} \) is continuous in the sense of

\[
\exists C_\ell > 0: \quad |\ell(u)| \leq C_\ell \|u\|_a \quad \forall u \in V_0,
\]

then the discrete variational problem has a unique \textit{Galerkin solution} \( u_N \in V_{0,N} \) that satisfies the stability estimate (\( \rightarrow \) Sect. 2.3.2)

\[
\|u_N\|_a \leq C_\ell . \tag{3.1.6}
\]

Proof. Uniqueness of \( u_N \) is clear:

\[
a(u_N, v_N) = \ell(v_N) \quad \forall v_N \in V_{0,N} \quad \Rightarrow \quad a(u_N - w_N, v_N) = 0 \quad \forall v_N \in V_{0,N}
\]

\[
v_N := u_N - w_N \in V_{0,N} \quad \Rightarrow \quad \|u_N - w_N\|_a = 0 \quad \Rightarrow \quad u_N - w_N = 0 .
\]

The discrete linear variational problem (3.1.4) is set in the \textit{finite-dimensional} space \( V_{0,N} \). Thus, uniqueness of solutions is equivalent to existence of solutions (\( \rightarrow \) linear algebra).

If you do not like this abstract argument, wait and see the equivalence of (3.1.4) with a linear system.
of equations. It will turn out that under the assumptions of the theorem, the resulting system matrix will be symmetric and positive definite in the sense of [18, Def. 2.7.9].

The estimate (3.1.6) is immediate from setting $v_N := u_N$ in (3.1.4)

$$|a(u_N, u_N)| = |\ell(u_N)| \leq C_{\ell} (a(u_N, u_N))^{1/2} .$$

Recall from Sect. 1.5.1:

2nd step of Galerkin discretization:

Introduce (ordered) basis $\mathcal{B}_N$ of $V_{0,N}$:

$$\mathcal{B}_N := \{b_1^N, \ldots, b_N^N\} \subset V_N , \quad V_N = \text{Span} \{\mathcal{B}_N\} , \quad N := \dim(V_N) .$$

Unique basis representations:

$$u_N = \mu_1 b_1^N + \cdots + \mu_N b_N^N , \quad \mu_i \in \mathbb{R} ,$$

$$v_N = \nu_1 b_1^N + \cdots + \nu_N b_N^N , \quad \nu_i \in \mathbb{R}$$

plug into (3.1.4).
Of course, there are infinitely many ways to choose the basis $\mathcal{B}_N$. Below we will study the impact of different choices.

What follows repeats the derivation of (1.5.18) and, in particular, (1.5.43).

\begin{equation}
\begin{aligned}
\forall v_N \in V_{0,N} : \quad a(u_N, v_N) &= \ell(v_N) \quad \forall v_N \in V_{0,N} .
\end{aligned}
\end{equation}

\begin{align*}
\sum_{k=1}^{N} \sum_{j=1}^{N} \mu_k \nu_j a(b_N^k, b_N^j) &= \sum_{j=1}^{N} \nu_j \ell(b_N^j) \quad \forall \nu_1, \ldots, \nu_N \in \mathbb{R} ,
\end{align*}

\begin{align*}
\sum_{j=1}^{N} \nu_j \left( \sum_{k=1}^{N} \mu_k a(b_N^k, b_N^j) - \ell(b_N^j) \right) &= 0 \quad \forall \nu_1, \ldots, \nu_N \in \mathbb{R} ,
\end{align*}
\[
\sum_{k=1}^{N} \mu_k a(b_N^k, b_N^j) = \ell(b_N^j) \quad \text{for } j = 1, \ldots, N.
\]

\[
[\vec{\mu} = (\mu_1, \ldots, \mu_N)^\top \in \mathbb{R}^N]
\]

\[
\mathbf{A} \vec{\mu} = \vec{\varphi}, \text{ with } \mathbf{A} = \left( a(b_N^k, b_N^j) \right)_{j,k=1}^{N,N} \in \mathbb{R}^{N,N}, \quad \vec{\varphi} = \left( \ell(b_N^j) \right)_{j=1}^{N}.
\]

A linear system of equations

---

**Linear discrete variational problem**

\( u_N \in V_{0,N} : \quad a(u_N, v_N) = \ell(v_N) \quad \forall v_N \in V_{0,N} \)

**Choosing basis \( \mathcal{B}_N \)**

**Linear system of equations**

\( \mathbf{A} \vec{\mu} = \vec{\varphi} \)

**Galerkin matrix:**

\[
\mathbf{A} = \left( a(b_N^k, b_N^j) \right)_{j,k=1}^{N,N} \in \mathbb{R}^{N,N},
\]

**Right hand side vector:**

\[
\vec{\varphi} = \left( \ell(b_N^j) \right)_{j=1}^{N} \in \mathbb{R}^N,
\]

**Coefficient vector:**

\[
\vec{\mu} = (\mu_1, \ldots, \mu_N)^\top \in \mathbb{R}^N,
\]

**Recovery of solution:**

\[
u_N = \sum_{k=1}^{N} \mu_k b_N^k.
\]
(Legacy) terminology for FEM: 

- **Galerkin matrix** = stiffness matrix
- **Right hand side vector** = load vector
- Galerkin matrix for \((u, v) \mapsto \int_\Omega uv \, dx\) = mass matrix

**Corollary 3.1.7.** \((3.1.4)\) has unique solution \iff \(A\) nonsingular

**Remark 3.1.8 (Impact of choice of basis).**

Choice of \(B_N\) in theory does not affect \(u_N\) \(\Rightarrow\) No impact on discretization error!

But: Key properties (e.g., conditioning) of matrix \(A\) crucially depend on basis \(B_N\)!
Lemma 3.1.9. Consider \((3.1.4)\) and two bases of \(V_{0,N}\),

\[
\mathfrak{B}_N := \{b_1^N, \ldots, b_N^N\}, \quad \mathfrak{B}_N := \{b_1^N, \ldots, b_N^N\},
\]
related by

\[
b_N^j = \sum_{k=1}^{N} s_{jk} b_N^k \quad \text{with} \quad S = (s_{jk})_{j,k=1}^{N} \in \mathbb{K}^{N,N} \text{ regular}.
\]

Galerkin matrices \(A, A \in \mathbb{K}^{N,N}\), right hand side vectors \(\vec{\varphi}, \vec{\varphi} \in \mathbb{K}^{N}\), and coefficient vectors \(\vec{\mu}, \vec{\mu} \in \mathbb{R}^{N}\), respectively, satisfy

\[
A = SAS^T, \quad \vec{\varphi} = S \vec{\varphi}, \quad \vec{\mu} = S^{-T} \vec{\mu}.
\] (3.1.10)

Proof. Make use of the bilinearity of \(a\):

\[
A_{lm} = a\left(b_N^m, b_N^l\right) = \sum_{k=1}^{N} \sum_{j=1}^{N} s_{mk} a\left(b_N^k, b_N^j\right) s_{lj} = \sum_{k=1}^{N} \left( \sum_{j=1}^{N} s_{lj} A_{jk} \right) s_{mk} = (SAS^T)_{lm},
\]
Reminder of linear algebra:

Definition 3.1.11 (Congruent matrices).

Two matrices $A \in \mathbb{K}^{N,N}$, $B \in \mathbb{K}^{N,N}$, $N \in \mathbb{N}$, are called congruent, if there is a regular matrix $S \in \mathbb{K}^{N,N}$ such that $B = SAS^H$.

Equivalence relation on square matrices

Lemma 3.1.12. Matrix property invariant under congruence $\iff$ Property of Galerkin matrix invariant under change of basis $\mathfrak{B}_N$

Matrix properties invariant under congruence:

- regularity $\rightarrow$ [18, Def. 2.0.5]
- symmetry
- positive definiteness $\rightarrow$ [18, Def. 2.7.9]
3.2 Case study: Triangular linear FEM in two dimensions

This section elaborates how to extend the linear finite element Galerkin discretization of Sect. 1.5.1.2 to two dimensions. Familiarity with the 1D setting is essential for understanding the current section.

Initial focus: well-posed 2nd-order linear variational problem posed on $H^1(\Omega)$ (→ Def. 2.2.12)

Example: Neuman problem with homogeneous Neumann data and reaction term

\[
 u \in H^1(\Omega): \quad \int_{\Omega} \alpha(x) \text{grad} u \cdot \text{grad} v + c(x) uv \, dx = \int_{\Omega} f v \, dx \forall v \in H^1(\Omega), \quad (3.0.1)
\]

\[
 \updownarrow \quad \text{see Sect. 2.4}
\]

BVP: \quad \begin{align*}
 - \text{div}(\alpha(x) \text{grad} u) + c(x) u &= f & \text{in } \Omega, \\
 \text{grad} u \cdot n &= 0 & \text{on } \partial \Omega.
\end{align*}

R. Hiptmair
C. Schwab
H. Harbrecht
V. Gradinaru
A. Chernov
SAM, ETHZ
Assumptions on \( \Omega \subset \mathbb{R}^2 \), see Rem. 2.1.1:

\[ \Omega \text{ is a polygon} \]

polygon with 10 corners

By default, the domain \( \Omega \) is assumed to be an \textit{open} set, that is, \( x \in \Omega \) implies \( x \notin \partial \Omega \)!

### 3.2.1 Triangulations
What is the 2D counterpart of mesh/grid $\mathcal{M}$ from Sect. (1.5.1.2)?

**Triangulation $\mathcal{M}$ of $\Omega$:**

(i) $\mathcal{M} = \{K_i\}_{i=1}^M$, $M \in \mathbb{N}$, $K_i \supseteq$ open triangle

(ii) disjoint interiors: $i \neq j \Rightarrow K_i \cap K_i = \emptyset$

(iii) tiling property: $\bigcup_{i=1}^M \overline{K}_i = \overline{\Omega}$

(iv) intersection $\overline{K}_i \cap \overline{K}_j$, $i \neq j$,

is

- either $\emptyset$

- or an edge of both triangles

- or a vertex of both triangles

**notation:** $\supseteq$ a subset of $\mathbb{R}^d$ together with its boundary ("closure")

**Parlance:** vertices of triangles $=$ nodes of mesh ($= \text{set } \mathcal{V}(\mathcal{M})$)

Fig. 62
3.2.2 Linear finite element space

A mesh that does not comply with the property (iv) from above.
Next goal: generalize the spline space $S^0_1(\mathcal{M}) \subset H^1([a, b])$ of piecewise linear functions on a 1D grid $\mathcal{M}$, see Fig. 25, that was used as Galerkin trial/test space in 1D in Sect. 1.5.1.2

$$V_{0,N} = S^0_{1,0}(\mathcal{M}) := \left\{ v \in C^0([0, 1]): v|_{[x_{i-1}, x_i]} \text{ linear, } i = 1, \ldots, M, v(a) = v(b) = 0 \right\}.$$  

Grid/mesh cells:  
- intervals $[x_{i-1}, x_i]$, $i = 1, \ldots, M$  
- triangles $K_i$, $i = 1, \ldots, M$

Linear functions:
- $x \in \mathbb{R} \mapsto \alpha + \beta \cdot x$, $\alpha, \beta \in \mathbb{R}$  
- $\bm{x} \in \mathbb{R}^2 \mapsto \alpha + \beta \cdot \bm{x}$, $\alpha \in \mathbb{R}$, $\beta \in \mathbb{R}^2$

$$V_{0,N} = S^0_1(\mathcal{M}) := \left\{ v \in C^0(\Omega): \forall K \in \mathcal{M}: v|_K(\bm{x}) = \alpha_K + \beta_K \cdot \bm{x}, \alpha_K \in \mathbb{R}, \beta_K \in \mathbb{R}^2, \bm{x} \in K \right\} \subset H^1(\Omega)$$

see Thm. 2.2.17
Functions of the form \( \mathbf{x} \mapsto \alpha_K + \beta_K \cdot \mathbf{x}, \alpha_K \in \mathbb{R}, \beta_K \in \mathbb{R}^2 \) are called (affine) linear.

\[ S^0_1(\mathcal{M}) \]

- notation: \( S^0_1(\mathcal{M}) \) of continuous functions, cf. \( C^0(\Omega) \)
- locally 1st degree polynomials
- scalar functions

\( \triangleright \) continuous piecewise affine linear function \( \in S^0_1(\mathcal{M}) \) on a triangular mesh \( \mathcal{M} \)
Remark 3.2.1 (Piecewise gradient).

Thm. 2.2.17 \[ S_{1}^{0}(\mathcal{M}) \subset H^{1}(\Omega) \]

\[ \Rightarrow \quad \text{for } u_{N} \in S_{1}^{0}(\mathcal{M}) \text{ the gradient } \nabla u_{N} \text{ can be computed on each triangle as piecewise constant function, cf. Ex. 2.2.20.} \]

(On \( K \in \mathcal{M} \): \( \nabla(\alpha_{K} + \beta_{K} \cdot x) = \beta_{K} \))

△

3.2.3 Nodal basis functions

Next goal: generalization of “tent functions”, see (1.5.53).

Recall condition (1.5.54), which defines a tent function in the space \( S_{1}^{0}(\mathcal{M}) \). This approach carries over to 2D.
Idea: define (?) basis function $b^x_N$, $x \in \mathcal{V}(M)$, by

$$
\begin{align*}
    b^x_N(y) &= \begin{cases} 
        1, & \text{if } y = x, \\
        0, & \text{if } y \in \mathcal{V}(M) \setminus \{x\}
    \end{cases}
\end{align*}
$$

(3.2.2)

Is this possible?

**Reasoning:**

- There is exactly one plane through three non-collinear points in $\mathbb{R}^3$. The graph of a linear function $\mathbb{R}^2 \mapsto \mathbb{R}$ is a plane.

- On a triangle $K$ with vertices $a^1, a^2, a^3$: (affine) linear $q : K \mapsto \mathbb{R}$ uniquely determined by values $q(a^i)$.

- $v_N \in S^0_1(M)$ uniquely determined by $\{v_N(x), x \text{ node of } M\}$!

- $\dim S^0_1(M) = \#\mathcal{V}(M)$ \quad ($\mathcal{V}(M) = \text{set of nodes (}=\text{vertices of triangles}) \text{ of } M$)
Writing \( V(\mathcal{M}) = \{x^1, \ldots, x^N\} \), the nodal basis \( \mathcal{B}_N := \{b^1_N, \ldots, b^N_N\} \) of \( S^0_1(\mathcal{M}) \) is defined by the conditions

\[
b^i_N(x^j) = \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{else,} \end{cases} \quad i, j \in \{1, \ldots, N\}.
\]

(Ordering (↔ numbering) of nodes assumed!)

Piecewise linear nodal basis function

("hat function"/ "tent function")

\[
u_N = \sum_{i=1}^{N} \mu_i b^i_N \in S^0_1(\mathcal{M})
\]

Coefficient \( \mu_j \) = "nodal value" of \( u_N \) at \( j \)-th node of \( \mathcal{M} \)

\[
u_N(x^j) = \mu_j
\]
Remark 3.2.4 (Linear finite element space for homogeneous Dirichlet problem).

Recall that the Dirichlet problem with homogeneous boundary conditions $u|_{\partial\Omega} = 0$ is posed on the Sobolev space $H^1_0(\Omega)$ (→ Def. 2.2.10), see (2.3.3), Ex. 2.8.1.

Galerkin space for homogeneous Dirichlet b.c.: $V_{0,N} = S^0_{1,0}(\mathcal{M}) := S^0_1(\mathcal{M}) \cap H^1_0(\Omega)$

Notation:

$S^0_{1,0}(\mathcal{M})$ zero on $\partial \Omega$, cf. $H^1_0(\Omega)$

$S^0_{1,0}(\mathcal{M}) = \text{Span} \{ b^j_N : \mathbf{x}^j \in \Omega \text{ (interior node !)} \}$

$\dim S^0_{1,0}(\mathcal{M}) = \# \{ \mathbf{x} \in \mathcal{V}(\mathcal{M}) : \mathbf{x} \notin \partial \Omega \}$
“Location” of nodal basis functions:
(mesh $\mathcal{M}$ → Fig. 149)

- $\bullet$ → nodal basis functions of $S_1^0(\mathcal{M})$
- $\bullet$ → nodal basis functions of $S_{1,0}^0(\mathcal{M})$

Bottom line: the Galerkin trial/test space contained in $H^1_0(\Omega)$ is obtained by dropping all “tent functions” that do not vanish on $\partial\Omega$ from the basis.
3.2.4 Sparse Galerkin matrix

Now: \( a \) \( \widehat{=} \) any (symmetric) bilinear form occurring in a linear 2nd-order variational problem, most general form

\[
a(u,v) := \int_\Omega (\alpha(x) \nabla u) \cdot \nabla v + c(x) uv \, dx + \int_{\partial \Omega} h v \, ds, \quad u,v \in H^1(\Omega).
\]

(3.2.5)

\( b_N^j \) \( \widehat{=} \) nodal basis function associated with vertex \( x^j \) of triangulation \( M \) of \( \Omega \), see Sect. 3.2.3.

Note: \( a \) symmetric \( \Rightarrow \) symmetric Galerkin matrix

Now we study the sparsity (\( \rightarrow [18, \text{Sect. 2.6}] \)) of the Galerkin matrix \( A := \left( a(b_N^j, b_N^i) \right)_{i,j=1}^N \in \mathbb{R}^{N,N} \), \( N := \dim S_0^1(M) = \# \mathcal{V}(M) \), see Sect. 3.1.

The consideration are fairly parallel to those that made us understand that the Galerkin matrix for the 1D case was tridiagonal, see (1.5.58).
Nodes $\mathbf{x}^i, \mathbf{x}^j \in \mathcal{V}(\mathcal{M})$ not connected by an edge $\iff$ $\text{Vol}(\text{supp}(b^i_N) \cap \text{supp}(b^j_N)) = 0 \implies (A)_{ij} = 0$. 
Lemma 3.2.6 (Sparsity of Galerkin matrix).

\[ \exists C = C(\text{topology of } \Omega): \# \{(i, j) \in \{1, \ldots, N\}^2 : (A)_{ij} \neq 0 \} \leq 7 \cdot N + C. \]

Proof. Euler's formula (http://en.wikipedia.org/wiki/Euler_characteristic)

\[ \#M - \#E(M) + \#V(M) = \chi_{\Omega}, \quad \chi_{\Omega} = \text{Euler characteristic of } \Omega. \]

Note that \( \chi_{\Omega} \) is a topological invariant (alternating sum of Betti numbers).

By combinatorial considerations (traverse edges and count triangles):

\[ 2 \cdot \#E_I(M) + \#E_B(M) = 3 \cdot \#M, \]

where \( E_I(M), E_B(M) \) stand for the sets of interior and boundary edges of \( M \), respectively.

\[ \#E_I(M) + 2\#E_B(M) = 3(\#V(M) - \chi_{\Omega}). \]

Then use

\[ N = \#V(M), \quad \text{nnz}(A) \leq N + 2 \cdot \#E(M) \leq 7 \cdot \#V(M) - 6\chi_{\Omega}. \]
Recall from [18, Def. 2.6.1]:

**Notion 3.2.7 (Sparse matrix).** \( A \in \mathbb{K}^{m,n}, m, n \in \mathbb{N}, \) is **sparse**, if

\[
\text{nnz}(A) := \#\{(i, j) \in \{1, \ldots, m\} \times \{1, \ldots, n\} : a_{ij} \neq 0\} \ll mn.
\]

Sloppy parlance: matrix **sparse** \( \iff \) “almost all” entries = 0 /“only a few percent of” entries \( \neq \) 0

---

**Example 3.2.8 (Sparse Galerkin matrices).**

\( M \) = triangular mesh, \( V_{0,N} = \mathcal{S}^0_{1,0}(\mathcal{M}) \), homogeneous **Dirichlet boundary conditions**, linear 2nd-order scalar elliptic differential operator.

Galerkin discretization of a 2nd-order linear variational problems utilizing the **nodal basis** of \( \mathcal{S}^0_1(\mathcal{M})/\mathcal{S}^0_{1,0}(\mathcal{M}) \)
leads to sparse linear systems of equations.
Triangular mesh $\mathcal{M}$

Resulting sparsity pattern of Galerkin matrix

Recall: visualization of sparsity pattern by means of MATLAB `spy`-command.
3.2.5 Computation of Galerkin matrix

For sake of simplicity consider

\[
a(u, v) := \int_{\Omega} \nabla u \cdot \nabla v \, dx, \quad u, v \in H^1_0(\Omega).
\]

and Galerkin discretization based on

- triangular mesh, see Sect. 3.2.1,
- discrete trial/test space \( S_{1,0}(\mathcal{M}) \subset H^1_0(\Omega) \),
- nodal basis \( \mathcal{B}_N = \{ b^i_N \} \) according to (3.2.2).

\[
(A)_{i,j} = a(b^j_N, b^i_N) = \int_{\Omega} \nabla b^j_N \cdot \nabla b^i_N \, dx
\]

Sect. 3.2.4: we need only study the cases, where \( x^i, x^j \in \mathcal{V}(\mathcal{M}) \)

1. are connected by an edge of the triangulation,
2. coincide.
Idea: “Assembly”
(add up cell contributions)

\[(A)_{ij} = \int_{K_1} \text{grad } b_N^j|_{K_1} \cdot \text{grad } b_N^i|_{K_1} \, dx + \int_{K_2} \text{grad } b_N^j|_{K_2} \cdot \text{grad } b_N^i|_{K_2} \, dx\]

Zero in on single triangle \(K \in \mathcal{M}\):

\[a_K(b_N^j, b_N^i) := \int_{K} \text{grad } b_N^j|_{K} \cdot \text{grad } b_N^i|_{K} \, dx, \quad x^i, x^j \text{ vertices of } K.\]  

(3.2.9)

Use analytic representation for \(b_N^j|_{K}\):

if \(a_1, a_2, a_3\) vertices of \(K\), \(\lambda_i := b_N^j|_{K}\), \(a^i = x^j\)

\(i \leftrightarrow \text{local vertex number}, \ j \leftrightarrow \text{global node number}\)
The functions $\lambda_1, \lambda_2, \lambda_3$ on the triangle $K$ are also known as \textit{barycentric coordinate functions}. They provide the nonzero restrictions of \textit{tent functions} to triangles, see Fig. 87.
\[ \lambda_1(x) = \frac{1}{2|K|} \left( x - a^2 \right) \cdot \left( a_2^3 - a_1^3 \right) = -\frac{|e_1|}{2|K|} (x - a^2) \cdot n^1, \]

\[ \lambda_2(x) = \frac{1}{2|K|} \left( x - a^3 \right) \cdot \left( a_3^2 - a_1^2 \right) = -\frac{|e_2|}{2|K|} (x - a^3) \cdot n^2, \]

\[ \lambda_3(x) = \frac{1}{2|K|} \left( x - a^1 \right) \cdot \left( a_1^2 - a_2^2 \right) = -\frac{|e_3|}{2|K|} (x - a^1) \cdot n^3. \]

\[(e_i = \text{edge opposite vertex } a^i, \text{see Figure for numbering scheme}>\)

From the distance formula for a point w.r.t to a line given in Hesse normal form:

\[ (a^i - a^j) \cdot n_i = \text{dist}(a^i; e_i) = h_i (h_i \hat{=} \text{height}) \text{ and } 2|K| = |e_i|h_i \Rightarrow \lambda_i(a^i) = 1. \]

This shows that the \( \lambda_i \) really provide the restrictions of p.w. linear nodal basis functions (tent functions) of \( S^0_1(\mathcal{M}) \) to triangle \( K \), because they are clearly (affine) linear and comply with (3.2.2).
\[
\text{grad } \lambda_1 = \frac{1}{2|K|} \left( \frac{a_2^2 - a_3^3}{a_1^2 - a_3^3} \right), \quad \text{grad } \lambda_2 = \frac{1}{2|K|} \left( \frac{a_3^3 - a_1^1}{a_1^1 - a_3^1} \right), \quad \text{grad } \lambda_3 = \frac{1}{2|K|} \left( \frac{a_1^1 - a_2^2}{a_1^1 - a_3^1} \right).
\]

\[
\left( \int_K \text{grad } \lambda_i \cdot \text{grad } \lambda_j \, dx \right)_{i,j=1}^3 = \text{element (stiffness) matrix } A_K
\]

\[
= \frac{1}{2} \begin{pmatrix}
\cot \omega_3 + \cot \omega_2 & - \cot \omega_3 & - \cot \omega_2 \\
- \cot \omega_3 & \cot \omega_3 + \cot \omega_1 & - \cot \omega_1 \\
- \cot \omega_2 & - \cot \omega_1 & \cot \omega_2 + \cot \omega_1
\end{pmatrix}.
\] (3.2.10)

The local numbering and naming conventions are displayed in Fig. 110.

Derivation of (3.2.10), see also [19, Lemma 3.47]: obviously, because the gradients \( \text{grad } \lambda_i \) are constant on \( K \),

\[
a(\lambda_i, \lambda_j) = \int_K \text{grad } \lambda_i \cdot \text{grad } \lambda_j \, dx = \frac{1}{4|K|} |e_i| |e_j| n_i \cdot n_j.
\]
Then use: \( \mathbf{n}_i \cdot \mathbf{n}_j = \cos(\pi - \omega_k) = -\cos \omega_k, \quad (i \neq j) \)
\( |K| = \frac{1}{2} |e_i||e_j| \sin \omega_k, \quad (i \neq j). \)

Case \( i = j \) employs a trick: 
\[
\sum_{i=1}^{3} \lambda_i = 1 \quad \Rightarrow \quad \sum_{i=1}^{3} a(\lambda_i, \lambda_j) = 0.
\]

**Remark 3.2.11** (Scaling of entries of element matrix for \(-\Delta\)).

(3.2.10): \( \mathbf{A}_K \) does not depend on the “size” of triangle \( K \)!

(more precisely, element matrices are equal for similar triangles)

This can be seen by the following reasoning:

- Obviously translation and rotation of \( K \) does not change. \( \mathbf{A}_K \)
- **Scaling** of \( K \) by a factor \( \rho > 0 \) has the following effect that
  - the area \( |K| \) is scaled by \( \rho^2 \),
the gradients $\nabla \lambda_i$ are scaled by $\rho^{-1}$ (the barycentric coordinate functions $\lambda_i$ become steeper when the triangle shrinks in size.).

Both effects just offset in $a_K$ from (3.2.9) such that $A_K$ remains invariant under scaling.

\[\begin{align*}
\text{“Assembly” of } (A)_{ij} \text{ starts from the sum} \\
(A)_{ij} &= \int_{K_1} \nabla b_N|_{K_1} \cdot \nabla b_N|_{K_1} dx + \int_{K_2} \nabla b_N|_{K_2} \cdot \nabla b_N|_{K_2} dx.
\end{align*}\]

\[\triangle\]

\[\begin{align*}
(A)_{ij} \text{ can be obtained by summing respective } (*) \text{ entries of the elements matrices of the elements adjacent to the edge connecting } x^i \text{ and } x^j
\end{align*}\]

\[\text{(*)}: \text{ watch correspondence of local and global vertex numbers!}\]
\[(A)_{ij}\] by summing entries of two element matrices

"Assembly" of diagonal entry \((A)_{ii}\): summing corresponding diagonal entries of element matrices belonging to triangles adjacent to node \(x^i\).
Remark 3.2.12 (Assembly algorithm for linear Lagrangian finite elements).

Assume:
- numbering of nodal basis functions $\leftrightarrow$ numbering of mesh vertices $\in V(\mathcal{M})$
- numbering of triangles (cells) of mesh $\mathcal{M} = \{K_1, \ldots, K_M\}$, $M := \#\mathcal{M}$.
Data structure: \( \text{idx} \in \mathbb{N}^{\#M,3} \): local→global index mapping array

\[
\text{idx}(k,l) = \text{global number of vertex } l \text{ of } k\text{-th cell}
\]

\[
x^{\text{idx}(k,l)} = a^l \quad \text{when } a^1, a^2, a^3 \text{ are the vertices of } K_k.
\]  

(3.2.13)

Code 3.2.14: Assembly of finite element Galerkin matrix for linear finite elements \( S_1^0(M) \)

1. \( A = \texttt{zeros}(N,N) ; \quad \% \ N = \#V(M) \)
2. \( \textbf{for } i=1:M \quad \% \ M = \#M \)
3. \( A_k = \text{getElementMatrix}(i) ; \quad \% \text{Compute } 3 \times 3\text{ element matrix, see (3.2.10)} \)
4. \( A(\text{idx}(i,:),\text{idx}(i,:)) = A(\text{idx}(i,:),\text{idx}(i,:)) + A_k; \)
5. \( \textbf{end} \)

Note: ☞ Homogeneous Dirichlet boundary conditions not taken into account in Code 3.2.13
☞ Regard Code 3.2.13 as “MATLAB pseudo-code”: in actual implementation \( A \) must be initialized as sparse matrix, see Rem. 3.5.18

Computational effort \( = O(\#M) \)
3.2.6 Computation of right hand side vector

We consider the linear form (right hand side of linear variational problem), see (2.3.3), (3.0.1):

\[ \ell(v) := \int_\Omega f(x) v(x) \, dx , \quad v \in H^1(\Omega) , \quad f \in L^2(\Omega) . \]

Recall formula for right hand side vector

\[ (\tilde{\varphi})_j = \ell(b^j_N) = \int_\Omega f(x) b^j_N(x) \, dx , \quad j = 1, \ldots, N . \]  \hspace{1cm} (3.2.15)
Idea: “Assembly”

\[
(\bar{\phi})_j = \sum_{l=1}^{N_j} \int_{K_l} f(x) \, b^j_{N|K_l}(x) \, dx ,
\]

where \( K_1, \ldots, K_{N_j} \) \( \hat{=} \) triangles adjacent to node \( x^j \).

(Integration confined to \( \text{supp}(b^j_N) \)!) 

\[\text{Fig. 76}\]

Zero in on single triangle \( K \in \mathcal{M} \):

\[
\ell_K(b^j_N) := \int_{K} f(x) \, b^j_{N|K}(x) \, dx , \quad x^j \text{ vertex of } K .
\]

(3.2.16)

Rem. [1.5.3]: \( f : \Omega \mapsto \mathbb{R} \) given in procedural form

\[
\text{function } y = f(x)
\]

Mandatory: use of numerical quadrature for approximate evaluation of \( \ell_K(b^j_N) \), cf. (1.5.59).
1D setting of Sect. 1.5.1.2: use of composite quadrature rules based on low Gauss/Newton-Cotes quadrature formulas on the cells $[x_{j-1}, x_j]$ of the grid, e.g. composite trapezoidal rule (1.5.59).

What is the 2D counterpart of the composite trapezoidal rule?

Recall:

trapezoidal rule [18, Eq. 10.2.7] integrates linear interpolant of integrand based on endpoint values
Idea:

2D trapezoidal rule

for triangle $K$ with vertices $a^1, a^2, a^3$

$$\int_K f(x) dx \approx \frac{|K|}{3} (f(a^1) + f(a^2) + f(a^3)) . \quad (3.2.17)$$

$\hat{\int}$ integration of linear interpolant $\sum_{i=1}^3 f(a^i) \lambda_i$ of $f$.

element (load) vector: $\vec{\phi}_K := \left( \ell_K(b_N^{j(i)}) \right)_{i=1}^3 = \frac{|K|}{3} \begin{pmatrix} f(a^1) \\ f(a^2) \\ f(a^3) \end{pmatrix},$

where $x^j(i) = a^i, i = 1, 2, 3$ (global node number $\leftrightarrow$ local vertex number).

As above in Fig. 75: “Assembly” of $(\vec{\phi})_j$ by summing up contributions from element vectors of triangles adjacent to $x^j$.

$$(\vec{\phi})_j = \sum_{l=1}^{N_j} \ell_K(b_N^j|K_l) = \sum_{l=1}^{N_j} (\vec{\phi})_{i(l,j)} = f(x^j) \cdot \frac{1}{3} \sum_{l=1}^{N_j} |K_l| , \quad (3.2.18)$$

where $i(l, j)$ is the local vertex index of the node $x^j$ (global index $j$) in the triangle $K_l$. 

R. Hiptmair
C. Schwab,
H. Harbrecht
V. Gradinaru
A. Chernov
SAM, ETHZ

3.2 Numerical Methods for PDEs

p. 311
Remark 3.2.19 (Assembly of right hand side vector for linear finite elements). → Rem. 3.2.12
Code 3.2.20: Assembly of right hand side vector for linear finite elements, see (3.2.18)

```matlab
phi = zeros(N,1); % N = ℏV(M)
for i=1:M % M = ℏM
    phiK = getElementVector(i); % Compute element (load) vector ∈ ℝ³
    phi(idx(i,:)) = phi(idx(i,:)) + phiK; % idx according to (3.2.13)
end
```

3.3 Building blocks of general FEM

The previous section explored the details of a simple finite element discretization of 2nd-order elliptic variational problems. Yet, it already introduced *key features and components* that distinguish the finite element approach to the discretization of linear boundary value problems for partial differential equations:
a focus on the variational formulation of a boundary value problem → Sect. 2.8,

a partitioning of the computational domain $\Omega$ by means of a mesh $\mathcal{M}$ (→ Sect. 3.2.1)

the use of Galerin trial and test spaces based on piecewise polynomials w.r.t. $\mathcal{M}$ (→ Sect. 3.2.2),

the use of locally supported basis functions for the assembly of the resulting linear system of equations (→ Sect. 3.2.3).

In this section a more abstract point of view is adopted and the components of a finite element method for scalar 2nd-order elliptic boundary value problems will be discussed in greater generality. However, prior perusal of Sect. 3.2 is strongly recommended.

3.3.1 Meshes

First main ingredient of FEM: triangulation/mesh of $\Omega$ → Sect. 3.2.1
Definition 3.3.1. A mesh (or triangulation) of $\Omega \subset \mathbb{R}^d$ is a finite collection $\{K_i\}_{i=1}^M$, $M \in \mathbb{N}$, of open non-degenerate (curvilinear) polygons ($d = 2$)/polyhedra ($d = 3$) such that

(A) $\overline{\Omega} = \bigcup \{\overline{K}_i, i = 1, \ldots, M\}$,
(B) $K_i \cap K_j = \emptyset \iff i \neq j$,
(C) for all $i, j \in \{1, \ldots, M\}, i \neq j$, the intersection $\overline{K}_i \cap \overline{K}_j$ is either empty or a vertex, edge, or face of both $K_i$ and $K_j$.

Terminology: Given mesh $\mathcal{M} := \{K_i\}_{i=1}^M$: $K_i$ called cell or element.
Vertices of a mesh $\rightarrow$ nodes (set $\mathcal{V}(\mathcal{M})$)
Types of meshes:

- Triangular mesh in 2D
  ![Triangular mesh in 2D](Fig. 79)

- Quadrilateral mesh in 2D
  ![Quadrilateral mesh in 2D](Fig. 80)

- 2D hybrid mesh comprising
  - triangles
  - quadrilaterals
  - curvilinear cells (at $\partial\Omega$)

- Tetrahedral meshes in 3D (created with NETGEN):

R. Hiptmair, C. Schwab, H. Harbrecht, V. Gradinaru, A. Chernov
SAM, ETHZ

3.3 Numerical Methods for PDEs
Tensor product mesh = grid

in 2D: \( a = x_0 < x_1 < \ldots < x_n = b \),
\( c = y_0 < y_1 < \ldots < y_m = d \).

\[
M = \{ [x_{i-1}, x_i] \times [y_{j-1}, y_j] : 1 \leq i \leq n, 1 \leq j \leq m \}.
\] (3.3.2)

Restricted to tensor product domains

Fig. 82

R. Hiptmair
C. Schwab,
H. Harbrecht
V. Gradinaru
A. Chernov
SAM, ETHZ

p. 317
If \( [C] \) does not hold

Triangular non-conforming mesh
(with hanging nodes)

\( K_i \cap K_j \) is only part of an edge/face for at most one of the adjacent cells.

(However, conforming if degenerate quadrilaterals admitted)

Terminology:

\textbf{Simplicial mesh} = triangular mesh in 2D
tetrahedral mesh in 3D

\section*{3.3.2 Polynomials}
Second main ingredient of FEM:
In FEM: Galerkin trial\-test space comprise \textit{locally polynomial} functions on $\Omega$

Clear: polynomials of degree $\leq p$, $p \in \mathbb{N}_0$, in 1D (univariate) polynomials, see (1.5.19)

$$\mathcal{P}_p(\mathbb{R}) := \{ x \mapsto c_0 + c_1 x + c_2 x^2 + \ldots c_p x^p \}.$$ 

In higher dimensions this concept allows various generalizations, one given in the following definition, one given in Def. 3.3.7.

\begin{definition}[Multivariate polynomials]
Space of \textit{multivariate} ($d$-variate) polynomials of (total) degree $p \in \mathbb{N}_0$:

$$\mathcal{P}_p(\mathbb{R}^d) := \{ \mathbf{x} \in \mathbb{R}^d \mapsto \sum_{\alpha \in \mathbb{N}_0^d, |\alpha| \leq p} c_\alpha \mathbf{x}^\alpha, \; c_\alpha \in \mathbb{R} \}.$$ 
\end{definition}

Def. 3.3.3 relies on multi-index notation:

$$\alpha = (\alpha_1, \ldots, \alpha_d): \quad \mathbf{x}^\alpha := x_1^{\alpha_1} \cdot \ldots \cdot x_d^{\alpha_d}, \quad \text{(3.3.4)}$$
\[ |\alpha| = \alpha_1 + \alpha_2 + \cdots + \alpha_d. \quad (3.3.5) \]

Special case:

\[ \text{d = 2: } \mathcal{P}_p(\mathbb{R}^2) = \left\{ \sum_{\alpha_1,\alpha_2 \geq 0, \alpha_1+\alpha_2 \leq p} c_{\alpha_1,\alpha_2} x_1^{\alpha_1} x_2^{\alpha_2}, \ c_{\alpha_1,\alpha_2} \in \mathbb{R} \right\}. \]

Example:

\[ \mathcal{P}_2(\mathbb{R}^2) = \text{Span} \left\{ 1, x_1, x_2, x_1^2, x_2^2, x_1 x_2 \right\} \]

**Lemma 3.3.6** (Dimension of spaces of polynomials).

\[ \dim \mathcal{P}_p(\mathbb{R}^d) = \binom{d+p}{p} \text{ for all } p \in \mathbb{N}_0, d \in \mathbb{N} \]

**Proof.** Distribute \( p \) “powers” to the \( d \) independent variables or discard them \( \triangleright d + 1 \) bins.

Combinatorial model: number of different linear arrangements of \( p \) identical items and \( d \) separators

\[ = \binom{d+p}{p} \]

\[ \square \]
Leading order \[
\dim \mathcal{P}_p(\mathbb{R}^d) = O(p^d)
\]

**Definition 3.3.7 (Tensor product polynomials).**

Space of tensor product polynomials of degree \( p \in \mathbb{N} \) in each coordinate direction

\[
\mathcal{Q}_p(\mathbb{R}^d) := \{ x \mapsto p_1(x_1) \cdots p_d(x_d), \ p_i \in \mathcal{P}_p(\mathbb{R}), i = 1, \ldots, d \}.
\]

**Example:**

\[
\mathcal{Q}_2(\mathbb{R}^2) = \text{Span} \left\{ 1, x_1, x_2, x_1 x_2, x_1^2, x_2^2, x_1 x_2^2, x_1^2 x_2, x_1 x_2^2, x_2^2 \right\}
\]

**Lemma 3.3.8 (Dimension of spaces of tensor product polynomials).**

\[
\dim \mathcal{Q}_p(\mathbb{R}^d) = (p + 1)^d \quad \text{for all} \quad p \in \mathbb{N}_0, \ d \in \mathbb{N}
\]
3.3.3 Basis functions

Third main ingredient of FEM: locally supported basis functions
(see Sect. 3.1 for role of bases in Galerkin discretization)

Basis functions $b_1^N, \ldots, b_N^N$ for a finite element trial/test space $V_{0,N}$ built on a mesh $M$ satisfy:

(a) $\mathcal{B}_N := \{b_1^N, \ldots, b_N^N\}$ is basis of $V_{0,N} \quad \Rightarrow \quad N = \dim V_{0,N},$

(b) each $b_i^N$ is associated with a single cell/edge/face/vertex of $M,$

(c) $\text{supp}(b_i^N) = \bigcup \{K: K \in M, p \subset \overline{K}\},$ if $b_i^N$ associated with cell/edge/face/vertex $p.$
Finite element terminology: \( b^i_N = \text{global shape functions/global basis functions} \)

Mesh \( \mathcal{M} \) + global shape functions \( \rightarrow \) complete description of finite element space

**Example 3.3.9** (Supports of global shape functions in 1D). \( \rightarrow \) Sect. 1.5.1.2

\( \Omega = ]a, b[ \equiv \text{interval} \)

Equidistant mesh

\[ \mathcal{M} := \{ ]x_{j-1}, x_j[, \ j = 1, \ldots, M \} , \]

\[ x_j := a + hj, \ h := (b - a)/M, \ M \in \mathbb{N} . \]

**Example 3.3.10** (Supports of global shape functions on triangular mesh).
Requirement (c) implies that global finite element basis functions are *locally supported*.

What is the rationale for this requirement?
Consider a generic bilinear form $a$ arising from a linear scalar 2nd-order elliptic BVP, see (3.2.5): it involves integration over $\Omega/\partial\Omega$ of products of (derivatives of) basis functions. Thus the integrand for $a(b^i_N, b^j_N)$ vanishes outside the overlap of the supports of $b^i_N$ and $b^j_N$.

**Galerkin matrix** $A \in \mathbb{R}^{N \times N}$ with $(A)_{ij} := a(b^i_N, b^j_N)$, $i, j = 1, \ldots, N$ satisfies $a_{ij} \neq 0$ only if $b^i_N$ and $b^j_N$ associated with vertices/faces/edges(cells) adjacent to common cell.

Finite element stiffness matrices are **sparse** (→ Notion 3.2.7)

Global shape functions $\rightarrow$ Restriction to element $\rightarrow$ local shape functions (3.3.11)
**Definition 3.3.12** (Local shape functions).

Given finite element function space on mesh $\mathcal{M}$ with global shape functions $b^j_N$, $i = 1, \ldots, N$:

$$\{b^j_N|_K, \ K \subset \text{supp}(b^j_N)\} = \text{set of local shape functions on } K \in \mathcal{M}.$$ 

Local shape functions $b^1_K, \ldots, b^Q_K$, $Q = Q(K) \in \mathbb{N}$ also associated with vertices/edges/faces/interior of $K$.

**Example 3.3.13** (Local shape functions for $S^0_1(\mathcal{M})$ in 2D). → Sect. 3.2.3

Global basis function for $S^0_1(\mathcal{M})$

On “unit triangle” $K$ with vertices

$$a^1 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \ a^2 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \ a^3 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

$$b^1_K(x) = 1 - x_1 - x_2,$$

Local shape functions:

$$b^2_K(x) = x_1,$$

$$b^3_K(x) = x_2.$$
These are the barycentric coordinate functions $\lambda_1$, $\lambda_2$, $\lambda_3$ introduced in Sect. 3.2.5

3.4 Lagrangian FEM
Taken for granted: finite element mesh $\mathcal{M}$ according to Def. 3.3.1.

Goal: construction of finite element spaces and global shape functions of higher polynomials degrees.

Lagrangian finite element spaces provide spaces $V_{0,N}$ of $\mathcal{M}$-piecewise polynomials that fulfill

$$V_{N,0} \subset C^0(\Omega) \quad \text{Thm. 2.2.17} \quad \Rightarrow \quad V_{N,0} \subset H^1(\Omega).$$

Notation:

(Lagrangian FE spaces)

$$S^0_p(\mathcal{M})$$

continuous functions, cf. $C^0(\Omega)$

locally polynomials of degree $p$, e.g. $\mathcal{P}_p(\mathbb{R}^d)$

3.4.1 Simplicial Lagrangian FEM
\( \mathcal{M} \): Simplicial mesh, consisting of triangles in 2D, tetrahedra in 3D.

Now we generalize \( S^0_1(\mathcal{M})/S^0_{1,0}(\mathcal{M}) \) from Sect. 3.2 to higher polynomial degree \( p \in \mathbb{N}_0 \).

**Definition 3.4.1** (Simplicial Lagrangian finite element spaces).

The space of \( p \)-th degree Lagrangian finite element functions on simplicial mesh \( \mathcal{M} \)

\[
S^0_p(\mathcal{M}) := \{ v \in C^0(\Omega) : v|_K \in \mathcal{P}_p(K) \quad \forall K \in \mathcal{M} \}.
\]

Def. 3.4.1 merely describes the space of trial/test functions used in a Lagrangian finite element method on a Simplicial mesh. A crucial ingredient is still missing (→ Sect. 3.3.3): the global shape functions still need to be specified. This is done by generalizing (3.2.2) based on sets of special interpolation nodes.

**Example 3.4.2** (Triangular quadratic Lagrangian finite elements).
interpolation nodes

\[ \mathcal{N} := \mathcal{V}(\mathcal{M}) \cup \{ \text{midpoints of edges} \}, \]
\[ \mathcal{N} = \{ \mathbf{p}_1, \ldots, \mathbf{p}_N \}. \]

Nodal basis functions \( b_j^N \), \( j = 1, \ldots, N \) defined by, cf. (3.2.2)

\[ b_j^N(\mathbf{p}_i) = \begin{cases} 1, & \text{if } i = j, \\ 0, & \text{else.} \end{cases} \quad (3.4.3) \]

A “definition” like (3.4.3) is cheap, but it may be pointless, in case no such functions \( b_j^N \) exist. To establish their existence, we first study the case of a single triangle \( K \).

We have to show that there is a basis of \( \mathcal{P}_2(\mathbb{R}^2) \) that satisfies (3.4.3) in the case of a mesh consisting of a single triangle \( \mathcal{M} = \{ K \} \).

A first simple consistency check: does the number of interpolation nodes \( \# \mathcal{N} \) for \( \mathcal{M} = \{ K \} \) agree with \( \dim \mathcal{P}_2(\mathbb{R}^2) = 6 \)? Yes, it does!
Local shape functions (barycentric coordinate representation)

\[ b^1_K = (2\lambda_1 - 1)\lambda_1, \]
\[ b^2_K = (2\lambda_2 - 1)\lambda_2, \]
\[ b^3_K = (2\lambda_3 - 1)\lambda_3, \]
\[ b^4_K = 4\lambda_1\lambda_2, \]
\[ b^5_K = 4\lambda_2\lambda_3, \]
\[ b^6_K = 4\lambda_1\lambda_3. \]

(3.4.4)

To see the validity of the formulas (3.4.4), note that

- \( \lambda_i(a^j) = 1 \) and \( \lambda_i(a^j) = 0 \), if \( i \neq j \), where \( a^1, a^2, a^3 \) are the vertices of the triangle \( K \),
- \( \lambda_1(m^{12}) = \lambda_1(m^{13}) = \frac{1}{2} \), where \( m^{ij} = \frac{1}{2}(a^i + a^j) \) denotes the midpoint of the edge connecting \( a^i \) and \( a^j \),
- each barycentric coordinate function \( \lambda_i \) is affine linear such that \( \lambda_i\lambda_j \in P_2(\mathbb{R}^2) \).
Selected local shape functions:

So far we have seen that *local shape functions* can be found that satisfy (3.4.3).

**Issue:** can the local shape functions from (3.4.4) be “stiched together” across interelement edges such that they yield a *continuous* global basis function? (Remember that Thm. 2.2.17 demands global continuity in order to obtain a subspace of $H^1(\Omega)$.)
The restriction of a quadratic polynomial to an edge is an \textit{univariate} quadratic polynomial.

Fixing its value in three points, the midpoint of the edge and the endpoints, \textit{uniquely} fixes this polynomial.

The local shape functions associated with the same \underline{interpolation node} “from left and right” agree on the edge.

> continuity!
(3.4.3): this function attains value $= 1$ at a vertex ($\bullet$) and vanishes at the midpoints ($\bullet$) of the edges of adjacent triangles, as well as at any other vertex.

**Example 3.4.5** (Interpolation nodes for cubic and quartic Lagrangian FE in 2D).
3.4.2 Tensor-product Lagrangian FEM
Now we consider tensor product meshes (grids), see (3.3.2), Fig. 82, for a 2D example.

**Example 3.4.6 (Bilinear Lagrangian finite elements).**

Sought: generalization of 1D piecewise linear finite element functions from Sect. 1.5.1.2, see Fig. 25, to 2D tensor product grid $\mathcal{M}$.

Tensor product structure of $\mathcal{M}$ ➤ tensor product construction of FE space

This is best elucidated by a tensor product construction of basis functions:

$$
\begin{align*}
  b_{N,x}^j(x) & : 1D \text{ tent function on } \mathcal{M}_x = \{ [x_{j-1}, x_j], \ j = 1, \ldots, n \} \\
  b_{N,y}^l(y) & : 1D \text{ tent function on } \mathcal{M}_y = \{ [y_{j-1}, y_j], \ j = 1, \ldots, n \}
\end{align*}
$$

2D tensor product “tent function” associated with node $p$:

$$
 b_N^p(\boldsymbol{x}) = b_{N,x}^j(x_1) \cdot b_{N,y}^l(x_2) , \text{ where } p = (x_j, y_l)^T .
$$

(3.4.7)
\[ b_{N,x}^{J}(x) \]

\[ b_{N,y}^{L}(y) \]

\[ b_{N}^{P}(x) \]

\[ \text{2D tensor product tent function} \]

No pyramid!

Basis functions *associated* \( \rightarrow \) Sect. 3.3.3, condition \([c]\) with nodes of \( \mathcal{M} \),
Tensor product construction ➤ bilinear local shape functions, e.g. on \( K = ]0, 1[^2 \)

\[
\begin{align*}
    b_1^K(x) &= (1 - x_1)(1 - x_2), \\
    b_2^K(x) &= x_1(1 - x_2), \\
    b_3^K(x) &= x_1 x_2, \\
    b_4^K(x) &= (1 - x_1)x_2.
\end{align*}
\] (3.4.8)

Fig. 100
Bilinear local shape functions on unit square $K$:

$$\text{Span} \left\{ b_{K}^{1}, b_{K}^{2}, b_{K}^{3}, b_{K}^{4} \right\} = \mathcal{Q}_{1}(\mathbb{R}^2).$$

Bilinear Lagrangian finite element space on 2D tensor product mesh $\mathcal{M}$:

$$S_{1}^{0}(\mathcal{M}) := \left\{ v \in C^{0}(\Omega) : v|_{K} \in \mathcal{Q}_{1}(\mathbb{R}^2) \ \forall K \in \mathcal{M} \right\} .$$

(3.4.9)
The following is a natural generalization of (3.4.9) to higher degree local tensor product polynomials, see Def. 3.3.7:

**Definition 3.4.10** (Tensor product Langrangian finite element spaces).

Space of $p$-th degree Lagrangian finite element functions on tensor product mesh $\mathcal{M}$

$$S^0_\mathcal{M} := \{ v \in C^0(\Omega) : v|_K \in \mathcal{Q}_p(K) \ \forall K \in \mathcal{M} \}.$$ 

Terminology: $S^0_\mathcal{M} = \text{multilinear finite elements}$ ($p = 1, d = 2 = \text{bilinear finite elements}$)

Remaining issue: definition of global basis functions (global shape functions)

Policy: use of **interpolation nodes** as in Sect. 3.4.1, see Ex. 3.4.2.

**Example 3.4.11** (Quadratic tensor product Lagrangian finite elements).

Consider case $p = 2, d = 2$ of Def. 3.4.10:
Interpolation nodes for $S^0_2(\mathcal{M})$

$$\mathcal{N} = \mathcal{V}(\mathcal{M}) \cup \{\text{midpoints of edges}\}.$$  

Note: number of interpolation nodes belonging to one cell is

$$9 = \dim Q_2(\mathbb{R}^2).$$

Choice of interpolation nodes for tensor product Lagrangian finite elements:

Global basis functions defined analogously to (3.4.3).
Remark 3.4.12 (Imposing homogeneous Dirichlet boundary conditions).

What is a global basis for $S^0_p(M) \cap H^1_0(\Omega)$, where $M$ is either a simplicial mesh or a tensor product mesh?

We proceed analogous to Rem. 3.2.4: recall that global basis functions are defined via interpolation nodes $p^j$, $j = 1, \ldots, N$, see (3.4.3).

$$S^0_{p,0}(M) := S^0_p(M) \cap H^1_0(\Omega) = \text{Span} \left\{ b^j_N : p^j \in \Omega \text{ (interior node)} \right\}. \tag{3.4.13}$$
Remark 3.4.14 ((Bi)-linear Lagrangian finite elements on hybrid meshes).

\( \mathcal{M} \): 2D hybrid mesh comprising triangles & rectangles

Idea: use
- linear functions \((\rightarrow \text{Def. 3.3.3, } p = 1)\) on triangular cells,
- bi-linear functions \((\rightarrow \text{Def. 3.4.10, } p = 1)\) on rectangles.

\[
S_1^0(\mathcal{M}) = \left\{ v \in H^1(\Omega) : v|_K \in \begin{cases} \mathcal{P}_1(\mathbb{R}^2), & \text{if } K \in \mathcal{M} \text{ is triangle}, \\ \mathcal{Q}_1(\mathbb{R}^2), & \text{if } K \in \mathcal{M} \text{ is rectangle} \end{cases} \right\}.
\] (3.4.15)

Two issues arise:

1. Does the prescription (3.4.15) yield a large enough space? (Note that \( v \in H^1(\Omega) \Rightarrow S_1^0(\mathcal{M}) \subset C^0(\Omega) \), see Thm. 2.2.17, but continuity might enforce too many constraints.)
2. Does the space from (3.4.15) allow for locally supported basis functions associated with nodes of the mesh?

We will give a positive answer to both question by constructing the basis functions:

Define global shape functions \( b^j_N \) according to (3.2.3)

This makes sense, because

- linear/bi-linear functions on \( K \) are uniquely determined by their values in the vertices,
- the restrictions to an edge of \( K \) of the local linear and bi-linear shape functions are both \textit{linear} univariate functions, see Figs. 72, 101.

Fixing vertex values for \( v_N \in S^0_1(\mathcal{M}) \) uniquely determines \( v \) on all edges of \( \mathcal{M} \) already, thus, \textit{ensuring global continuity}, which is necessary due to Thm. 2.2.17.
Remark 3.4.16 (Lagrangian finite elements on hybrid meshes).

\( \mathcal{M} \): 2D hybrid mesh comprising triangles & rectangles

- Matching interpolation nodes on edges of triangles and rectangles
- Glueing of local shape functions on triangles and rectangles possible

... global interpolation nodes for \( p = 2 \)
This section discusses algorithmic details of Galerkin finite element discretization of 2nd-order elliptic variational problems for spatial dimension $d = 2, 3$ on bounded polygonal/polyhedral domains $\Omega \subset \mathbb{R}^d$.

The presentation matches the LehrFEM finite element MATLAB library, parts of which will be made available for participants of the course. A detailed documentation is available from [6].

The guiding principle behind the implementation of finite element codes is

> to rely on *local* computations as much as possible!

This is made possible by the *local supports* of the *global basis functions*, see Sect. [3.3.3], Ex. [3.3.10].

### 3.5.1 Mesh file format

Data flow in (most) finite element software packages:
Here “ направленное ” designates passing of information, which is usually done by writing and reading files to and from hard disk. This requires particular file formats.

*Example* 3.5.1 (Triangular mesh: file format).
File format for storing triangular mesh (of polygonal domain):

```
# Two-dimensional simplicial mesh

1  ξ₁  η₁  # Coordinates of first node
2  ξ₂  η₂  # Coordinates of second node
...
N  ξₙ  ηₙ  # Coordinates of N-th node

1  n₁¹  n₂¹  n₃¹  X₁  # Indices of nodes of first triangle
2  n₁²  n₂²  n₃²  X₂  # Indices of nodes of second triangle
...
M  n₁ᴹ  n₂ᴹ  n₃ᴹ  Xᴹ  # Indices of nodes of M-th triangle

Xᵢ, i = 1, ..., M → extra information (e.g. material properties in triangle #ᵢ).
```
Optional: additional information about edges (on $\partial \Omega$):

$$K \in \mathbb{N} \quad \# \text{Number of edges on } \partial \Omega$$

$$n_1^1 \ n_2^1 \ Y_1 \quad \# \text{Indices of endpoints of first edge}$$

$$n_1^2 \ n_2^2 \ Y_2 \quad \# \text{Indices of endpoints of second edge} \quad (3.5.3)$$

$$\vdots$$

$$n_1^K \ n_2^K \ Y_K \quad \# \text{Indices of endpoints of } K\text{-th edge}$$

$Y_k$, $k = 1, \ldots, K \rightarrow$ extra information

**Example 3.5.4** (Mesh file format for MATLAB code “LehrFEM”).

**Vertex coordinate file:**

<table>
<thead>
<tr>
<th>% List of vertices</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 +0.000000e+00 -1.000000e+00</td>
</tr>
<tr>
<td>2 +1.000000e+00 +0.000000e+00</td>
</tr>
<tr>
<td>3 +0.000000e+00 +1.000000e+00</td>
</tr>
<tr>
<td>4 -1.000000e+00 +0.000000e+00</td>
</tr>
<tr>
<td>5 +0.000000e+00 +0.000000e+00</td>
</tr>
</tbody>
</table>

**Cell information file:**

<table>
<thead>
<tr>
<th>% List of elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 1 2 5</td>
</tr>
<tr>
<td>2 2 3 5</td>
</tr>
<tr>
<td>3 3 4 5</td>
</tr>
<tr>
<td>4 4 1 5</td>
</tr>
</tbody>
</table>
Loading a mesh

```matlab
m = load_Mesh('Coord_Circ.dat','Elem_Circ.dat');
plot_Mesh(m,'apts');
```

Option flags:

'a': with axes

'p': vertex labels on

't': cell labels on

's': caption/title on

For details see [6, Sect. 1.3.1], [6, Sect. 1.3.2].
How to create a mesh?

→ **Mesh generation** (beyond scope of this course)

→ [http://www.andrew.cmu.edu/user/sowen/mesh.html](http://www.andrew.cmu.edu/user/sowen/mesh.html)

Free software:
- **DistMesh** (MATLAB, used in “LehrFEM”, see [6, Sect. 1.2])
- **NETGEN** (industrial strength open source mesh generator)
- **Triangle** (easy to use 2D mesh generator)
- **TETGEN** (Tetrahedral mesh generation)

*Example 3.5.5* (Mesh generation in LehrFEM).

Algorithm & details → [24], more explanations in [6, Sect. 1.2].
MATLAB-CODE: mesh generation for circular domain

```matlab
BBOX = [-1 -1; 1 1];
H0 = 0.1;
DHD = @(x) sqrt(x(:,1).^2+x(:,2).^2)-1;
HHANDLE = @(x) ones(size(x,1),1);
Mesh = init_Mesh(BBOX,H0,DHD,...
    HHANDLE,[],1);
save_Mesh(Mesh,'Coordinates.dat',...
    'Elements.dat');
```

Bounding box

Largest reasonable edge length

Signed distance function \( \varphi(x) \):

(distance from \( \partial \Omega \), \( \varphi(x) < 0 \) \iff \( x \in \Omega \))

Element size function
(determines local edge length)

3.5.2 Mesh data structures [6, Sect. 1.1]

Issue: internal representation of mesh (\rightarrow Def. 3.3.1) in computer code
mesh data structure must provide:

1. offer unique identification of cells/(faces)/(edges)/vertices
2. represent mesh topology (= incidence relationships of cells/faces/edges/vertices)
3. describe mesh geometry (= location/shape of cells/faces/edges/vertices)
4. allow sequential access to edges/faces of a cell
   \[ \text{\rightarrow traversal of local shape functions\,\,degrees of freedom} \]
5. make possible traversal of cells of the mesh (\text{\rightarrow global numbering})

Focus: \textbf{array oriented data layout} \text{ \hspace{1em} (\rightarrow MATLAB, FORTRAN)}

Notation:
\[ \mathcal{M} = \text{mesh (set of elements)}, \quad \mathcal{V}(\mathcal{M}) = \text{set of nodes (vertices) in } \mathcal{M}, \quad \mathcal{E}(\mathcal{M}) = \text{set of edges in } \mathcal{M} \]

Case: \textit{d-dimensional simplicial triangulation }\mathcal{M}, \quad \textit{minimal data structure (cf. Sect. 3.5.1)}

\rightarrow \text{Coordinates of vertices } \mathcal{V}(\mathcal{M}) : \#\mathcal{V}(\mathcal{M}) \times d\text{-array Coordinates of reals}

\rightarrow \text{Vertex indices for cells: } \#\mathcal{M} \times (d + 1)\text{-array Elements of integers.}
Example 3.5.6 (Arrays storing 2D triangular mesh).

<table>
<thead>
<tr>
<th>$i$</th>
<th>Coordinates</th>
<th>$K_j$</th>
<th>Vertex indices</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-1, 1</td>
<td>1</td>
<td>1 2 9</td>
</tr>
<tr>
<td>2</td>
<td>-1, 0</td>
<td>2</td>
<td>2 5 9</td>
</tr>
<tr>
<td>3</td>
<td>-1, -1</td>
<td>3</td>
<td>5 8 9</td>
</tr>
<tr>
<td>4</td>
<td>0, -1</td>
<td>4</td>
<td>5 7 8</td>
</tr>
<tr>
<td>5</td>
<td>0, 0</td>
<td>5</td>
<td>3 4 2</td>
</tr>
<tr>
<td>6</td>
<td>1, -1</td>
<td>6</td>
<td>4 5 2</td>
</tr>
<tr>
<td>7</td>
<td>1, 0</td>
<td>7</td>
<td>4 7 5</td>
</tr>
<tr>
<td>8</td>
<td>1, 1</td>
<td>8</td>
<td>4 6 7</td>
</tr>
<tr>
<td>9</td>
<td>0, 1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Array Coordinates

This information is enough for efficient assembly of finite element Galerkin matrices/right hand side vectors for (bi-)linear Lagrangian finite elements, see Rem. 3.2.12, 3.2.19.
Global shape functions associated with edges/faces ➾ extra information required!

Optional extra information:

→ Edge connecting vertices: $\#V(M) \times \#V(M)$ symmetric sparse integer matrix $I_\mathcal{E}$

$$(I_\mathcal{E})_{ij} := \begin{cases} 0, & \text{if vertex } \#i \text{ not linked to } \#j \\ e_{ij}, & \text{if edge connecting } \#i \text{ and } \#j \end{cases}$$

here $e_{ij}$ is the unique edge number $\in \{1, 2, \ldots, \#\mathcal{E}(M)\}$

→ End points of the edges: $\#\mathcal{E}(M) \times 2$ array of integer (= vertex indices of end points).

→ Cell adjacent to edges: $\#\mathcal{E}(M) \times 2$ array of integers (=cell indices)

(one cell index =0 if edge is on $\partial\Omega$)

**Example 3.5.7** (Extended MATLAB mesh data structure). ➾ [6, Sect. 1.1]

```
mesh = add_Edge2Elem(add_Edges(init_Mesh(BBOX,H0,DHD,HHANDLE,[[],1])))
```

(init_Mesh ➔ Ex. 3.5.5)
vertex coordinates, see Ex. 3.5.4

vertex indices of triangles, see Ex. 3.5.4

coordinates in Coordinates array

indices of endpoints

sparse integer matrix:

entry \((i,j)\) = edge index, if \(\neq 0\)

•

•

•

indices of adjacent cells in Elements array

indices of adjacent cells in Elements array

local indices of edges w.r.t. adjacent cells

Notation: \(\mathcal{E}(\mathcal{M})\) \(\doteq\) edges of 2D mesh

How to number \(\leftrightarrow\) order

local shape functions

global shape functions

Elements, Edges arrays \(\Rightarrow\) ordering of vertices of cells/endpoints of edges

Arrays (of vertices, cells, edges) \(\Rightarrow\) array indices \(\Rightarrow\) numbering of global shape functions
Remark 3.5.8. Second option: C++/JAVA-style object oriented data layout

Nodes, cells of $\mathcal{M}$ $\leftrightarrow$ dynamically allocated objects (instances of classes Node, Cell)

```cpp
class Node {
    private:
        double x, y;
        ID id;
    public:
        Node(double x, double y, ID id=0);
        Point getCoords(void) const;
        ID getId(void) const;
};

class Cell {
    private:
        const vector<Node*> vertices;
        ID id;
    public:
        Cell(const vector<Node*> &vertices, ID id=0);
        int NoNodes(void) const;
        const Node &getNode(int) const;
        ID getId(void) const;
};

class BdFace {
    private:
        const vector<Node*> vertices;
        BdCond bdcond;
    public:
        BdFace(const vector<Node*> &vertices);
        int NoNodes(void) const;
        const Node &getNode(int) const;
        BdCond getBdCond(void) const;
};

class Mesh {
    private:
    list<Node> nodes;
    list<Cell> cells;
    list<BdFace> bdfaces;
    public:
        Mesh(istream &file);
        virtual Mesh(void);
        const list<Node> &Nodes(void) const;
        const list<Cell> &Cells(void) const;
        const list<BdFace> &BdFaces(void) const;
};
```
ID getId() \(\rightarrow\) provides unique identifier for each node/cell.

Distinguish:  
- local objects (\(\rightarrow\) classes Node, Cell, BdFace)  
- global objects ("mesh management" class Mesh, see below)

3.5.3 Assembly [6, Sect. 5]

"Assembly" = term used for computing entries of stiffness matrix/right hand side vector (load vector) in a finite element context.

From the dictionary: "Assemble" = to fit together all the separate parts of sth.
Aspects of assembly for linear Lagrangian finite elements \((V_{0,N} = S^0_{1,0}(\mathcal{M}))\) were discussed in Sects. 3.2.5, 3.2.6. (Refresh yourself on these sections in case you cannot remember the main ideas behind building the Galerkin matrix and right hand side vector.)

We consider a discrete variational problem \((V_{0,N} = \text{FE space}, \dim V_{0,N} = N \in \mathbb{N}, \text{see } (3.1.4))\)

\[
u_N \in V_{0,N}: \quad a(u_N, v_N) = \ell(v_N) \quad \forall v_N \in V_{0,N}.
\]  

To be computed (see also Sect. 3.2.5, Sect. 3.2.6):

- **Galerkin matrix** (stiffness matrix):
  \[A = \left( a(b^j_N, b^i_N) \right)_{i,j=1}^{N} \in \mathbb{R}^{N,N}\]

- **r.h.s. vector** (load vector):
  \[
  \vec{\phi} := \left( \ell(b^i_N) \right)_{i=1}^{N} \in \mathbb{R}^{N}
  \]

both can be written in terms of local cell contributions, since usually

\[
a(u, v) = \sum_{K \in \mathcal{M}} a_K(u|_K, v|_K), \quad \ell(v) = \sum_{K \in \mathcal{M}} \ell_K(v|_K).
\]  

(3.5.9)
Example: bilinear forms/linear forms arising from 2nd-order elliptic BVPs, e.g., (2.9.1), (2.9.2), (2.9.3), can be localized in straightforward fashion by restricting integration to mesh cells (→ Rem. 3.2.1): for \( u, v \in H^1(\Omega) \)

\[
a(u, v) := \int_{\Omega} \alpha(x) \nabla u \cdot \nabla v \, dx = \sum_{K \in \mathcal{M}} \int_K \alpha(x) \nabla u \cdot \nabla v \, dx \quad =: a_K(u|_K, v|_K), \tag{3.5.10}
\]

\[
\ell(v) := \int_{\Omega} fv \, dx = \sum_{K \in \mathcal{M}} \int_K fv \, dx \quad =: \ell_K(v|_K). \tag{3.5.11}
\]

Recall (3.3.11): Restrictions of global shape functions to cells = local shape functions
Definition 3.5.12 (Element (stiffness) matrix and element (load) vector).

Given local shape functions \( \{b^1_K, \ldots, b^Q_K\} \), \( Q \in \mathbb{N} \), we call

- element (stiffness) matrix 
  \[
  A_K := \left( a_K(b^i_K, b^j_K) \right)_{i,j=1}^Q \in \mathbb{R}^{Q \times Q},
  \]

- element (load) vector 
  \[
  \varphi_K := \left( f_K(b^i_K) \right)_{i=1}^Q \in \mathbb{R}^Q.
  \]

Note: Here \( Q \), the number of local shape functions on element \( K \in \mathcal{M} \), is independent of \( K \). In general, we could also have \( Q = Q_K \) when we blend several element types in one mesh, see Rem. 3.4.14.

<table>
<thead>
<tr>
<th>Type of FE space</th>
<th>( Q )</th>
</tr>
</thead>
<tbody>
<tr>
<td>degree ( p ) Lagrangian FE on \textit{triangular} mesh</td>
<td>( \dim \mathcal{P}_p(\mathbb{R}^2) = \frac{1}{2}(p + 1)(p + 2) )</td>
</tr>
<tr>
<td>degree ( p ) Lagrangian FE on \textit{tetrahedral} mesh</td>
<td>( \dim \mathcal{P}_p(\mathbb{R}^3) = \frac{1}{6}(p + 1)(p + 2)(p + 3) )</td>
</tr>
<tr>
<td>degree ( p ) Lagrangian FE on \textit{tensor product} mesh in 2D</td>
<td>( \dim Q_p(\mathbb{R}^2) = (p + 1)^2 )</td>
</tr>
</tbody>
</table>
Again scrutinize Figs. 74, 75 and the accompanying remarks in Sect. 3.2.5. We learn that in the special setting of this section

- the entries of the finite element Galerkin matrix can be obtained by summing corresponding entries of some element matrices,
- this corresponding entry of an element matrices is determined by the unique association of a local basis function to a global basis function.

These insights are formalized in the next theorem.
Theorem 3.5.13. The stiffness matrix and load vector can be obtained from their cell counterparts by

\[
A = \sum_{K} T_{K}^{T} A_{K} T_{K}, \quad \bar{\phi} = \sum_{K} T_{K}^{T} \bar{\phi}_{K},
\]

with the index mapping matrices ("T-matrices") \( T_{K} \in \mathbb{R}^{Q,N} \), defined by

\[
(T_{K})_{ij} := \begin{cases} 
1, & \text{if } (b_{N}^{j}|_{K} = b_{K}^{i}), \\
0, & \text{otherwise.}
\end{cases} \quad 1 \leq i \leq Q, 1 \leq j \leq N.
\]

Note: Every T-matrix has exactly one non-vanishing entry per row.

Proof. (of Thm. 3.5.13)

\[
(A)_{ij} = a(b_{N}^{j}, b_{N}^{i}) = \sum_{K \in \mathcal{M}} a_{K}(b_{N}^{j}|_{K}, b_{N}^{i}|_{K}) = \sum_{K \in \mathcal{M}, \text{supp}(b_{N}^{j}) \cap K \neq \emptyset, \text{supp}(b_{N}^{i}) \cap K \neq \emptyset} a_{K}(b_{K}^{l(j)}, b_{K}^{l(i)}) = \sum_{K \in \mathcal{M}, \text{supp}(b_{N}^{j}) \cap K \neq \emptyset, \text{supp}(b_{N}^{i}) \cap K \neq \emptyset} (A_{K})_{l(i),l(j)}
\]

\( l(i) \in \{1, \ldots, Q\}, 1 \leq i \leq N \) \( \hat{=} \) index of the local shape function corresponding to the global shape function \( b_{N}^{i} \) on \( K \).
By (3.5.15), the indices \( l(i) \) encode the T-matrix according to

\[
(T_K)_{l(i),i} = 1, \quad i = 1, \ldots, N,
\]

where all other entries of \( T_K \) are understood to vanish.

\[
\Rightarrow (A)_{ij} = \sum_{K \in \mathcal{M}, \text{supp}(b^j_N) \cap K \neq \emptyset} \sum_{l=1}^Q \sum_{n=1}^Q (T_K)_{li} (A_K)_{ln} (T_K)_{nj}.
\]

Example 3.5.16 (Assembly for linear Lagrangian finite elements on triangular mesh).
Using the local/global numbering indicated beside

\[ T_{K^*} = \begin{pmatrix}
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0
\end{pmatrix} \]

Cell oriented assembly \[\leftrightarrow (3.5.14) \leftrightarrow A = \sum_K T_K^\top A_K T_K \]

\[ A = \sum_K T_K^\top A_K T_K \begin{cases}
\text{foreach } K \in \mathcal{M} \text{ do } \\
\quad \text{local operations on } K \ (\rightarrow A_K) \text{ and } A = A + T_K^\top A_K T_K \\
\text{endo}\end{cases} \]
Notion: local operations $\hat{=} \quad$ required only data from fixed “neighbourhood” of $K$

computational effort “$O(1)$”: independent of $\#\mathcal{M}$

\[
\text{Computational cost(Assembly of Galerkin matrix } A) = O(\#\mathcal{M})
\]

Cell oriented assembly in LehrFEM [6, Sect. 5.1]

\begin{verbatim}
function A = assemble(Mesh)
    for k = Mesh.Elements'
        idx = ①
        Aloc = ②
        A(idx,idx) = A(idx,idx)+Aloc;
    end
\end{verbatim}

① row vector of index numbers of global shape functions $b_{i_{1}}^{N}, \ldots, b_{i_{Q}}^{N} \in V_{N}$
corresponding to local shape functions $b_{1}^{K}, \ldots, b_{Q}^{K}$:

$\blacktriangleright$ $\text{idx} = (i_{1}, \ldots, i_{Q})$

(encodes index mapping matrix $T_{K}$)

② $Q \times Q$ element stiffness matrix
For Lagrangian FEM of fixed degree $p$ (→ Sect. 3.4):

the total computational effort is of the order $O(\#M) = O(N)$, $N := \dim S^0_p(M)$.

**Example 3.5.17** (Assembly for quadratic Lagrangian FE in MATLAB code).

Setting: FE space $S^0_2(M)$ on triangular mesh $M$ of polygon $\Omega \subset \mathbb{R}^2$, see Ex. 3.4.2

Recall: 6 local shape functions: 3 vertex-associated, 3 edge-associated

 Convention: vertex-associated global shape functions $\rightarrow b^\mathcal{V}(M), \ldots, b^{\#\mathcal{V}(M)}_N$

edge-associated global shape functions $\rightarrow b^{\#\mathcal{V}(M)+1}_N, \ldots, b^{\#\mathcal{V}(M)+\#E(M)}_N$

Local numbering $\rightarrow$

![Diagram of a triangle with local numbering: 1, 2, 3, 4, 5, 6]
function A = assemMat_QFE(Mesh,EHandle,varargin)

nV = size(Mesh.Coordinates,1);
nE = size(Mesh.Elements,1)

I = zeros(36*nE,1); J = I; a = I; offset = 0;
for k =1:nE
    vidx = Mesh.Elements(k,:)
    idx = [vidx,...
        Mesh.Vert2Edge(vidx(1),vidx(2))+nV,...
        Mesh.Vert2Edge(vidx(2),vidx(3))+nV,...
        Mesh.Vert2Edge(vidx(3),vidx(1))+nV];
    Aloc = transpose(EHandle(Mesh.Coordinates(vidx,:),...
        Mesh.ElemFlag(k),varargin{:}));
    Qsq = prod(size(Aloc)); range = offset + 1:Qsq;
    t = idx(ones(length(idx),1),:)' ; I(range) = t(:);
    t = idx(ones(1,length(idx)),:) ; J(range) = t(:);
    a(range) = Aloc(:);
    offset = offset + Qsq;
end
A = sparse(I,J,a);
①: EHandle (function handle) → provides **element stiffness matrix** $A_K \in \mathbb{R}^{6,6}$

②: $I, J, a \hat{=} \text{linear arrays storing } (i, j, (A)_{ij}) \text{ for stiffness matrix } A$.
   Initialized with 0 for the sake of efficiency → Ex. 3.5.18

③: $\text{idx} \hat{=} \text{index mapping vector, see ❶ above}$

④: $\text{Alloc} = A_K \in \mathbb{R}^{6,6}$ (element stiffness matrix → Def. 3.5.12)

⑤: Mesh.ElemFlag(k) marks groups of elements (e.g. to select local coefficient function $\alpha(x)$ in (2.8.4))

⑥: Build **sparse** MATLAB-matrix (→ Def. 3.2.7) from index-entry arrays, see manual entry for MATLAB function **sparse** and [18, Sect. 2.6.2].
Remark 3.5.18 (Efficient implementation of assembly). \[ \rightarrow [18, \text{Sect. 2.6.2}] \]

tic-toc-timing (min of 4v runs), MATLAB V7, Intel Pentium 4 Mobile CPU 1.80GHz, Linux
Computation of element stiffness matrices skipped!

- **Sparse assembly:**
  \[
  A(\text{idx}, \text{idx}) = A(\text{idx}, \text{idx}) + A\text{loc};
  \]

- **Array assembly I:** “growing arrays”
  \[
  I = []; J = []; a = [];
  \]
  
  \[
  t = \text{idx}(:, \text{ones(length(idx),1)})';
  
  I = [I; t(:)];
  
  t = \text{idx}(:, \text{ones(1,length(idx)))};
  
  J = [J; t(:)];
  
  a = [a; A\text{loc}(:)];
  \]

- **Array assembly III**
  \[
  \rightarrow \text{see code fragment above}
  \]

More detailed discussion \[ \rightarrow [31] \text{ and } [18, \text{Sect. 2.6.2}] \].
3.5.4 Local computations and quadrature

We have seen that the (global) Galerkin matrix and right hand side vector are conveniently generated by “assembling” entries of element (stiffness) matrices and element (load) vectors.

Now we study the computation of these local quantities, see also Sect. 3.2.5, 3.2.6.

First option: analytic evaluations

We discuss bilinear form related to $-\Delta$, triangular Lagrangian finite elements of degree $p$, Sect. 3.4.1, Def. 3.4.1:

$$K \text{ triangle: } a_K(u, v) := \int_K \text{grad } u \cdot \text{grad } v \, dx$$

Use **barycentric coordinate** representations of **local shape functions**, in 2D

$$b_K^i = \sum_{\alpha \in \mathbb{N}_0^3, |\alpha| \leq p} \kappa_\alpha \lambda_1^{\alpha_1} \lambda_2^{\alpha_2} \lambda_3^{\alpha_3}, \quad \kappa_\alpha \in \mathbb{R},$$

\hspace{1cm} (3.5.19)
where $\lambda_i$ are the affine linear barycentric coordinate functions (linear shape functions), see Fig. 72.

For the barycentric coordinate representation of the quadratic local shape functions see (3.4.4), for a justification of (3.5.19) consult Rem. 3.6.9.

$$\Rightarrow \quad \text{grad} b_K^i = \sum_{\alpha \in \mathbb{N}_0^3, |\alpha| \leq p} \kappa_{\alpha} \left( \alpha_1 \lambda_1^{\alpha_1-1} \lambda_2^{\alpha_2} \lambda_3^{\alpha_3} \text{grad} \lambda_1 + \alpha_2 \lambda_1^{\alpha_1} \lambda_2^{\alpha_2-1} \lambda_3^{\alpha_3} \text{grad} \lambda_2 + \alpha_3 \lambda_1^{\alpha_1} \lambda_2^{\alpha_2} \lambda_3^{\alpha_3-1} \text{grad} \lambda_3 \right).$$

(3.5.20)

To evaluate

$$\int_K \lambda_1^{\beta_1} \lambda_2^{\beta_2} \lambda_3^{\beta_3} \text{grad} \lambda_i \cdot \text{grad} \lambda_j \, d\mathbf{x}, \quad i, j \in \{1, 2, 3\}, \beta_k \in \mathbb{N}.$$  

(3.5.21)
If $\mathbf{a}^1, \mathbf{a}^2, \mathbf{a}^3$ vertices of $K$ (counterclockwise ordering):

\[
\lambda_1(\mathbf{x}) = \frac{1}{2|K|} \left( \mathbf{x} - \left( \begin{array}{c} a_1^2 \\ a_2^2 \end{array} \right) \right) \cdot \left( \begin{array}{c} a_2^3 - a_2^3 \\ a_1^3 - a_1^3 \end{array} \right),
\]

\[
\lambda_2(\mathbf{x}) = \frac{1}{2|K|} \left( \mathbf{x} - \left( \begin{array}{c} a_1^3 \\ a_2^3 \end{array} \right) \right) \cdot \left( \begin{array}{c} a_3^1 - a_3^1 \\ a_1^1 - a_1^1 \end{array} \right),
\]

\[
\lambda_3(\mathbf{x}) = \frac{1}{2|K|} \left( \mathbf{x} - \left( \begin{array}{c} a_1^1 \\ a_2^1 \end{array} \right) \right) \cdot \left( \begin{array}{c} a_2^2 - a_2^2 \\ a_1^2 - a_1^2 \end{array} \right).
\]

\[
\text{grad } \lambda_1 = \frac{1}{2|K|} \left( \begin{array}{c} a_2^2 - a_2^3 \\ a_1^3 - a_1^3 \end{array} \right), \quad \text{grad } \lambda_2 = \frac{1}{2|K|} \left( \begin{array}{c} a_3^1 - a_3^1 \\ a_1^1 - a_1^1 \end{array} \right), \quad \text{grad } \lambda_3 = \frac{1}{2|K|} \left( \begin{array}{c} a_2^2 - a_2^3 \\ a_1^3 - a_1^3 \end{array} \right).
\]

*(3.5.22)*
Lemma 3.5.23 (Integration of powers of barycentric coordinate functions).
For any non-degenerate \(d\)-simplex \(K\) and \(\alpha_j \in \mathbb{N}, j = 1, \ldots, d + 1,\)
\[
\int_K \lambda_1^{\alpha_1} \cdots \lambda_{d+1}^{\alpha_{d+1}} \, d\mathbf{x} = d|K| \frac{\alpha_1! \alpha_2! \cdots \alpha_{d+1}!}{(\alpha_1 + \alpha_2 + \cdots + \alpha_{d+1} + d)!} \quad \forall \alpha \in \mathbb{N}_{d+1}^0. \tag{3.5.24}
\]

Proof for \(d = 2\)

Step #1: transformation \(K \to \) “unit triangle” \(\hat{K} :=\) convex \(\{(0,0), (1,0), (0,1)\}\),
\[
\Rightarrow \int_K \lambda_1^{\beta_1} \lambda_2^{\beta_2} \lambda_3^{\beta_3} \, d\mathbf{x} = 2|K| \int_0^1 \int_0^{1-\xi_1} \xi_1^{\beta_1} \xi_2^{\beta_2} (1 - \xi_1 - \xi_2)^{\beta_3} \, d\xi_2 \, d\xi_1 \\
= 2|K| \int_0^1 \xi_1^{\beta_1} \int_0^1 (1 - \xi_1)^{\beta_2 + \beta_3 + 1} s^{\beta_2} (1 - s)^{\beta_3} \, ds \, d\xi_1 \\
= 2|K| \int_0^1 \xi_1^{\beta_1} (1 - \xi_1)^{\beta_2 + \beta_3 + 1} \, d\xi_1 \cdot B(\beta_2 + 1, \beta_3 + 1)
\]
\[ = 2\left| \kappa \right| B(\beta_1 + 1, \beta_2 + \beta_3 + 2) \cdot B(\beta_2 + 1, \beta_3 + 1) , \]

\((*) \triangleq \) substitution \( s(1 - \xi_1) = \xi_2, \quad B(\cdot, \cdot) \triangleq \text{Euler's beta function} \)

\[ B(\alpha, \beta) := \int_0^1 t^{\alpha-1} (1 - t)^{\beta-1} \, dt , \quad 0 < \alpha, \beta < \infty . \]

Using \( \Gamma(\alpha + \beta) B(\alpha, \beta) = \Gamma(\alpha) \Gamma(\beta), \, \Gamma \triangleq \text{Gamma function}, \, \Gamma(n) = (n - 1)! , \)

\[ \Rightarrow \int_K \lambda_1^{\beta_1} \lambda_2^{\beta_2} \lambda_3^{\beta_3} \, d\mathbf{x} = 2\left| \kappa \right| \cdot \frac{\Gamma(\beta_1 + 1)\Gamma(\beta_2 + 1)\Gamma(\beta_3 + 1)}{\Gamma(\beta_1 + \beta_2 + \beta_3 + 3)} \quad \square . \]

**Remark.** Alternative: symbolic computing (MAPLE, Mathematica) for local computations

Second option: cell-based quadrature

At this point turn the pages back to (1.5.61) and remember the use of numerical quadrature for computing the Galerkin matrix for the linear finite element method in 1D.

Reminder: numerical quadrature mandatory in the presence of coefficients/source terms in **procedural form** \( \rightarrow \) Rem. [1.5.3].
Local quadrature formula, cf. (3.2.17)

\[
\int_{\Omega} f(x) \, dx \approx \sum_{K \in M} |K| \sum_{l=1}^{P} \omega^K_l f(\zeta^K_l), \quad \zeta^K_l \in K, \, \omega^K_l \in \mathbb{R}, \, \quad P \in \mathbb{N}.
\] (3.5.25)

Terminology:

\[\omega^K_l \rightarrow \text{weights}, \quad \zeta^K_l \rightarrow \text{quadrature nodes}\]

(3.5.25) = \text{P-point local quadrature rule}

**Mandatory**

- for computation of **load vector** \(f\) complicated/only available in procedural form, Rem. 1.5.3,
- for computation of **stiffness matrix**, if \(\alpha = \alpha(x)\) does not permit analytic integration.

Example for local quadrature rule: 2D trapezoidal rule from (3.2.17)

**Guideline [18, Sect. 10.2]:** only quadrature rules with positive weights are numerically stable.
How to gauge the quality of parametric local quadrature rules? \[ \rightarrow [18, \text{Sect. 10.3}] \]

Quality of a parametric local quadrature rule on $K \sim$ maximal degree of polynomials (multivariate $\rightarrow$ Def. 3.3.3, or tensor product $\rightarrow$ Def. 3.3.7) on $K$ integrated exactly by the corresponding quadrature rule on $K$.

Parlance: Quadrature rule exact for $\mathcal{P}_p(\mathbb{R}^d)$ $\Rightarrow$ quadrature rule of order $p + 1$

degree of exactness $p$

How are quadrature rules specified for the many different cells of a finite element mesh?

Remark 3.5.26 (Affine transformation of triangles).
Definition 3.5.27 (Affine (linear) transformation).

Mapping $\Phi : \mathbb{R}^d \mapsto \mathbb{R}^d$ is affine (linear), if $\Phi(x) = Fx + \tau$ with some $F \in \mathbb{R}^{d,d}$, $\tau \in \mathbb{R}^d$.

notation: ‘unit triangle” $\hat{K} := \text{convex} \left\{ \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \right\}$

Lemma 3.5.28 (Affine transformation of triangles).

For any non-degenerate triangle $K \subset \mathbb{R}^2 (|K| > 0)$ there is a unique affine transformation $\Phi_K$, $\Phi_K(\hat{x}) = F_K\hat{x} + \tau_K$ (→ Def. 3.5.27), with $K = \Phi(\hat{K})$. 

R. Hiptmair
C. Schwab,
H. Harbrecht,
V. Gradinaru,
A. Chernov
SAM, ETH Zürich
Formula:

\[ K = \text{convex} \left\{ \left( \frac{a_1^1}{a_2^1} \right), \left( \frac{a_1^2}{a_2^2} \right), \left( \frac{a_1^3}{a_2^3} \right) \right\} \Rightarrow \Phi_K(\hat{x}) = \begin{pmatrix} a_1^2 - a_1^1 & a_1^3 - a_1^1 \\ a_2^2 - a_2^1 & a_2^3 - a_2^1 \end{pmatrix} \hat{x} + \begin{pmatrix} a_1^1 \\ a_2^1 \end{pmatrix}. \]  

(3.5.29)

Note that

\[ |K| = \frac{1}{2} |\det F_K|. \]
Remark 3.5.30 (Transformation of local quadrature rules on triangles).

\[ \Phi_K(\hat{x}) := F_K \hat{x} + \tau_K \]  
\[ \hat{=} \]  
affine transformation (\( \rightarrow \) Def. 3.5.27) mapping \( \hat{K} \) to triangle \( K \), see Lemma 3.5.28.

By transformation formula for integrals [29, Satz 8.5.2]

\[ \int_K f(x) \, dx = \int_{\hat{K}} f(\Phi_K(\hat{x})) |\det F_K| \, d\hat{x} . \]  
\[ (3.5.31) \]

\( P \)-point quadrature formula on \( \hat{K} \) \( \rightarrow \) \( P \)-point quadrature formula on \( K \)

\[ \int_{\hat{K}} f(\hat{x}) \, d\hat{x} \approx |\hat{K}| \sum_{l=1}^{P} \omega_{l} f(\hat{\zeta}_{l}) \]  
\[ \int_{\Omega} f(x) \, dx \approx \sum_{K \in \mathcal{M}} |K| \sum_{l=1}^{P} \omega_{l}^{K} f(\zeta_{l}^{K}) \]  
\[ (3.5.32) \]

with \( \omega_{l}^{K} = \hat{\omega}_{l} \), \( \zeta_{l}^{K} = \Phi_K(\hat{\zeta}_{l}) \).
Only quadrature formula (3.5.25) on unit triangle $\hat{K}$ needs to be specified! (The same applies to tetrahedra, where affine mappings for $d = 3$ are used.)

Since the space $\mathcal{P}_p(\mathbb{R}^d)$ is *invariant* under affine mappings,

$$q \in \mathcal{P}_p(\mathbb{R}^d) \implies \hat{x} \mapsto q(\Phi(\hat{x})) \in \mathcal{P}_p(\mathbb{R}^d) \quad \text{for any affine transformation } \Phi,$$

the orders of the quadrature rules on the left and right hand side of (3.5.31) agree.

**Example 3.5.34** (Useful quadrature rules on triangles). → [6, Sect. 3.3.2]

Specification of quadrature rule for “unit triangle” $\hat{K} := \text{convex} \left\{ (0,0), (1,0), (0,1) \right\}$.

Quadrature rules described by pairs $(\hat{\omega}_1, \hat{\zeta}_1), \ldots, (\hat{\omega}_P, \hat{\zeta}_P), P \in \mathbb{N}$. 

---

R. Hiptmair
C. Schwab, H. Harbrecht
V. Gradinaru
A. Chernov
SAM, ETHZ
Quadrature rule of order 2 (exact for $\mathcal{P}_1(\hat{K})$)
\[
\left\{ \left( \frac{1}{3}, \left( \begin{array}{c}
0 \\
0 
\end{array} \right) \right), \left( \frac{1}{3}, \left( \begin{array}{c}
0 \\
1 
\end{array} \right) \right), \left( \frac{1}{3}, \left( \begin{array}{c}
1 \\
0 
\end{array} \right) \right) \right\} .
\] (3.5.35)

Quadrature rule of order 3 (exact for $\mathcal{P}_2(\hat{K})$)
\[
\left\{ \left( \frac{1}{3}, \left( \begin{array}{c}
\frac{1}{2} \\
0 
\end{array} \right) \right), \left( \frac{1}{3}, \left( \begin{array}{c}
0 \\
\frac{1}{2} 
\end{array} \right) \right), \left( \frac{1}{3}, \left( \begin{array}{c}
\frac{1}{2} \\
\frac{1}{2} 
\end{array} \right) \right) \right\} .
\] (3.5.36)

One-point quadrature rule of order 2 (exact for $\mathcal{P}_1(\hat{K})$)
\[
\left\{ \left( 1, \left( \begin{array}{c}
\frac{1}{3} \\
\frac{1}{3} 
\end{array} \right) \right) \right\} .
\] (3.5.37)

Quadrature rule of order 6 (exact for $\mathcal{P}_5(\hat{K})$)
\[
\left\{ \left( \frac{9}{40}, \left( \begin{array}{c}
\frac{1}{3} \\
\frac{1}{3} 
\end{array} \right) \right), \left( \frac{155 + \sqrt{15}}{1200}, \left( \begin{array}{c}
6 + \sqrt{15}/21 \\
6 + \sqrt{15}/21 
\end{array} \right) \right), \left( \frac{155 + \sqrt{15}}{1200}, \left( \begin{array}{c}
9 - 2\sqrt{15}/21 \\
6 + \sqrt{15}/21 
\end{array} \right) \right), \left( \frac{155 + \sqrt{15}}{1200}, \left( \begin{array}{c}
9 - 2\sqrt{15}/21 \\
9 + 2\sqrt{15}/21 
\end{array} \right) \right), \left( \frac{155 + \sqrt{15}}{1200}, \left( \begin{array}{c}
9 - 2\sqrt{15}/21 \\
9 + 2\sqrt{15}/21 
\end{array} \right) \right) \right\} .
\] (3.5.38)
In [10]: quadrature rules up to order $p = 21$ with $P \leq \frac{1}{6}p(p + 1) + 5$

**Remark 3.5.39 (Numerical quadrature in LehrFEM).** → [6, Sect. 3]

Routines return $P$-point quadrature formulas for

\[
\hat{K} = \begin{cases} 
\text{unit triangle} & \text{convex } \left\{ (0, 0), (0, 1), (1, 0) \right\} \\
\text{unit square} & \text{convex } \left\{ (0, 0), (0, 1), (1, 0), (1, 1) \right\}
\end{cases}
\]

for triangular cell,

for rectangular cell,

in MATLAB **structure** `QuadRule` with fields

- `QuadRule.w`: weights $\hat{\omega}_l$ of quadrature rule on $\hat{K}$,
- `QuadRule.x`: coordinates of nodes $\hat{\zeta}_l \in \hat{K}$ of quadrature rule on $\hat{K}$

For triangles: `QuadRule = PnOq()`, $\hat{\omega}_l = n$-point quadrature of order $q$

Location of quadrature nodes $\hat{\zeta}_l$ in unit triangle $\hat{K}$:
Example 3.5.40 (Local quadrature rules on quadrilaterals).

If $K$ quadrilateral $\Rightarrow \widehat{K} := \text{convex} \left\{ \left( \frac{0}{0} \right), \left( \frac{1}{0} \right), \left( \frac{0}{1} \right), \left( \frac{1}{1} \right) \right\}$ (unit square).

On $\widehat{K}$: 

\[ \text{tensor product construction:} \]

If $\{ (\omega_1, \zeta_1), \ldots, (\omega_P, \zeta_P) \}, \ P \in \mathbb{N}, \text{quadrature rule on the interval } [0, 1[, \text{ exact for } \mathcal{P}_P]0, 1[, \text{ then}$

\[ \left\{ \begin{array}{ccc}
(\omega_1^2, (\frac{\zeta_1}{\zeta_1})) & \cdots & (\omega_1 \omega_P, (\frac{\zeta_1}{\zeta_1})) \\
\vdots & & \vdots \\
(\omega_1 \omega_P, (\frac{\zeta_P}{\zeta_1})) & \cdots & (\omega_P^2, (\frac{\zeta_P}{\zeta_P}))
\end{array} \right\} \]
provides a quadrature rule on the unit square $\hat{K}$, exact for $Q_p(\hat{K})$.

Quadrature rules on $[0, 1]$ ($\rightarrow [18$, Ch. 10$)$:
- classical Newton-Cotes formulas (equidistant quadrature nodes).
- Gauss-Legendre quadrature rules, exact for $P_2P([0, 1])$ using only $P$ nodes.
- Gauss-Lobatto quadrature rules: $P$ nodes including $\{0, 1\}$, exact for $P_{2P-1}([0, 1])$.
3.5.5 Incorporation of essential boundary conditions

Recall variational formulation of *non-homogeneous* Dirichlet boundary value problem from Ex. [2.8.1]:

\[
\begin{align*}
\quad & u \in H^1(\Omega) : \quad \int_{\Omega} \kappa(x) \nabla u \cdot \nabla v \, dx = \int_{\Omega} f v \, dx \quad \forall v \in H^1_0(\Omega).
\end{align*}
\]  
\quad (2.8.4)

\[\Downarrow\]

\[\begin{align*}
- \text{div}(\kappa(x) \nabla u) &= f \quad \text{in} \; \Omega, \quad u = g \quad \text{on} \; \partial \Omega,
\end{align*}\]

with (admissible \(\rightarrow\) Rem. [2.9.4]) Dirichlet data \(g \in C^0(\partial \Omega)\).

Recall from Sect. 2.9:  
**Dirichlet b.c.** = essential boundary conditions  
(built into trial space)

Remember offset function technique, see [1.3.19] and Sect. 2.1.3: 

\[
(2.8.4) \quad \Leftrightarrow \quad u = u_0 + w,
\]

\[
\begin{align*}
\quad & w \in H^1_0(\Omega): \quad \int_{\Omega} \kappa(x) \nabla w \cdot \nabla v \, dx = \int_{\Omega} -\kappa(x) \nabla u_0 \cdot \nabla v + f v \, dx \quad \forall v \in H^1_0(\Omega),
\end{align*}
\]  
\quad (3.5.41)
where \( u_0 = g \) on \( \partial \Omega \).

Adapt this to finite element Galerkin discretization by generalizing the 1D example Rem. 1.5.64 to \( d = 2, 3 \):

Remember: we already know finite element subspaces \( V_{0,N} := S_{p,0}(\mathcal{M}) \subset H^1_0(\Omega) \), see Rem. 3.4.12.

Idea from Rem. 1.5.64:

use **offset function** \( u_0 \in V_N := S^0_p(\mathcal{M}) \)

*locally supported near the boundary*:

\[
\text{supp}(u_0) \subset \bigcup \{ K \in \mathcal{M} : \overline{K} \cap \partial \Omega \neq \emptyset \} .
\]  

(3.5.42)
Example 3.5.43 (offset functions for linear Lagrangian FE).
For Dirichlet data $g \in C^0(\partial \Omega)$

$$u_0 = \sum_{x \in V(\mathcal{M}) \cap \partial \Omega} g(x) b^x_N$$

(3.5.44)

$b^x_N \overset{\text{tent function}}{=} \text{tent function associated with node } x \in V(\mathcal{M}), \text{ cf. Sect. 3.2.3. (3.5.44) generalizes (1.5.65) to 2D.}$

**Remark 3.5.45 (Approximate Dirichlet boundary conditions).**

Be aware that for the choice (3.5.44)

$$u_0 \neq g \text{ on } \partial \Omega.$$

Rather, $u_0$ is a *piecewise linear interpolant* of the Dirichlet data $g \in C^0(\partial \Omega)$. Therefore, another *approximation* comes into play when enforcing Dirichlet boundary conditions by means of piecewise
polynomial offset functions.

**Example 3.5.46** (Implementation of non-homogeneous Dirichlet b.c. for linear FE).

Consider (2.8.4) and assume the following ordering of the nodal basis functions, see Fig. 67

\[ \mathcal{B}_0 := \{ b^1_N, \ldots, b^N_N \} \quad \triangleq \quad \text{nodal basis of } S_{1,0}^0(\mathcal{M}), \]

(tent functions associated with interior nodes)

\[ \mathcal{B} := \mathcal{B}_0 \cup \{ b^{N+1}_N, \ldots, b^M_N \} \quad \triangleq \quad \text{nodal basis of } S_1^0(\mathcal{M}) \]

(extra basis functions associated with nodes \( \in \partial \Omega \)).

Note: \( M = \# \mathcal{V}(\mathcal{M}), N = \# \{ \mathbf{x} \in \mathcal{V}(\mathcal{M}), \mathbf{x} \notin \partial \Omega \} \) (no. of interior nodes)

\[ A_0 \in \mathbb{R}^{N,N} \quad \triangleq \quad \text{Galerkin matrix for discrete trial/test space } S_{1,0}^0(\mathcal{M}), \]

\[ A \in \mathbb{R}^{M,M} \quad \triangleq \quad \text{Galerkin matrix for discrete trial/test space } S_1^0(\mathcal{M}). \]
If \( u_0 \in S^0_1(M) \) is chosen according to (3.5.44), then

\[
u_0 \in \text{Span}\left\{ b_{N+1}^N, \ldots, b_M^N \right\} \iff u_0 = \sum_{j=N+1}^M \gamma_j b_j^N,
\]

which means that the coefficient vector \( \vec{\nu} \) of the finite element approximation \( w_N \in S^0_{1,0}(M) \) of \( w \in H^1_0(\Omega) \) from (3.5.41) solves the linear system of equations

\[
A_0 \vec{\nu} = \vec{\phi} - A_{0\partial} \vec{\gamma}.
\]

Non-homogeneous Dirichlet boundary data are taken into account through a modified right hand side vector.

Alternative consideration leading to (3.5.48):

1. First ignore essential boundary conditions and assemble the linear system of equations arising
from the discretization of \( a \) on the (larger) FE space \( S^0_1(M) \):
\[
\begin{pmatrix}
A_0 & A_{0\partial} \\
A_{0\partial}^T & A_{\partial\partial}
\end{pmatrix}
\begin{pmatrix}
\bar{\mu}_0 \\
\bar{\mu}_{\partial}
\end{pmatrix}
= \begin{pmatrix}
\bar{\varphi} \\
\bar{\varphi}_{\partial}
\end{pmatrix}.
\]
(3.5.49)

Here, \( \bar{\mu}_0 \) \( \hat{=} \) coefficients for *interior* basis functions \( b^1_N, \ldots, b^N_N \)
\( \bar{\mu}_{\partial} \) \( \hat{=} \) coefficient for basis functions \( b^{N+1}_N, \ldots, b^M_N \) for basis functions associated with node \( x \in \partial \Omega \).

We realize that the coefficient vector of (3.5.49) is that of a FE approximation of \( u \)

\[ \bar{\mu}_{\partial} \text{ known} = \text{values of } g \text{ at boundary nodes: } \bar{\mu}_{\partial} = \tilde{\gamma} \]

Moving known quantities in (3.5.49) to the right hand side yields (3.5.48).

**Example 3.5.50** (Non-homogeneous Dirichlet boundary conditions in LehrFEM).

**Code 3.5.51: Solving 2nd-order Dirichlet BVP with linear FE in LehrFEM**

```matlab
1 % Initialize constants anf functions
2 NREFS = 5; % Number of red refinement steps
3 F_HANDLE = @f_LShap; % Right hand side source term
4 GD_HANDLE = @g_D_LShap; % Dirichlet boundary data
```
% Load mesh
Mesh = load_Mesh('meshvert.dat','meshel.dat');
Mesh.ElemFlag = ones(size(Mesh.Elements,1),1);
Mesh = add_Edges(Mesh);
Loc = get_BdEdges(Mesh);
Mesh.BdFlags = zeros(size(Mesh.Edges,1),1);
Mesh.BdFlags(Loc) = 1;

% Assemble stiffness matrix and load vector for linear Lagrangian FE
A = assemMat_LFE(Mesh,@STIMA_Lapl_LFE);
phi = assemLoad_LFE(Mesh,P7O6(),F_HANDLE,0,1,2);

% Incorporate Dirichlet boundary data for vertices adjacent to edges
% with carryingflag 1. U contains nodal values for Dirichlet boundary data, FreeDofs contains numbers of interior nodes.
% See Ex. 3.5.46 for further explanations.
[U,FreeDofs] = assemDir_LFE(Mesh,-7,GD_HANDLE);
L = L - A*U;

% Solve the linear system
U(FreeDofs) = A(FreeDofs,FreeDofs)

% Plot solution
plot_LFE(U,Mesh); colorbar;
3.6 Parametric finite elements

2D hybrid mesh $\mathcal{M}$ with (curvilinear) triangles and quadrilaterals

How to build $S^0_1(\mathcal{M})$?
3.6.1 Affine equivalence

Recall Lemma 3.5.28: affine transformation of triangles (3.5.29)

All cells of a triangular mesh are affine images of “unit triangle” $\hat{K}$

“Unit triangle”: $\hat{K} = \left< \left( \begin{array}{c} 0 \\ 0 \end{array} \right), \left( \begin{array}{c} 1 \\ 0 \end{array} \right), \left( \begin{array}{c} 0 \\ 1 \end{array} \right) \right>$

For $K = \text{convex} \left\{ a^1, a^2, a^3 \right\}$:

$F_K = \left( \begin{array}{ccc} a_2^2 - a_1^1 & a_3^3 - a_1^1 \\ a_2^2 - a_2^1 & a_3^3 - a_2^1 \\ a_2^2 - a_2^1 & a_3^3 - a_2^1 \end{array} \right)$, $\tau_K = a^1 \\ 

$\Phi_K(\hat{x}) = F_K \hat{x} + \tau_K$

Fig. 116
Remark 3.6.1 (Pullback of functions).

In a natural way, a transformation of domains induces a transformation of the functions defined on them:

\[
\text{Definition 3.6.2 (Pullback).}
\]

Given domains \( \Omega, \hat{\Omega} \subset \mathbb{R}^d \) and a bijective mapping \( \Phi : \hat{\Omega} \mapsto \Omega \), the pullback \( \Phi^* u : \hat{\Omega} \mapsto \mathbb{R} \) of a function \( u : \Omega \mapsto \mathbb{R} \) is a function on \( \hat{\Omega} \) defined by
\[
(\Phi^* u)(\hat{x}) := u(\Phi(\hat{x})) , \quad \hat{x} \in \hat{\Omega}.
\]

Implicitly, we used the pullback of integrands when defining quadrature rules through transformation, see (3.5.31).

Obviously, the pullback \( \Phi^* \) induces a linear mapping between spaces of functions on \( \Omega \) and \( \hat{\Omega} \), respectively.
In the context of numerical quadrature we made the observation, cf. (3.5.33):

**Lemma 3.6.3** (Preservation of polynomials under affine pullback). If $\Phi : \mathbb{R}^d \mapsto \mathbb{R}^d$ is an affine (linear) transformation ($\mapsto$ Def. 3.5.27), then

\[
\Phi^*(\mathcal{P}_p(\mathbb{R}^d)) = \mathcal{P}_p(\mathbb{R}^d) \quad \text{and} \quad \Phi^*(\mathcal{Q}_p(\mathbb{R}^d)) = \mathcal{Q}_p(\mathbb{R}^d).
\]
In fact, Lemma [3.5.28] reveals another reason for the preference for polynomials in building discrete Galerkin spaces.

**Proof.** (of Lemma [3.5.28])

Since the pullback is linear, we only need to study its action on the (monomial) basis $x \mapsto x^\alpha$, $\alpha \in \mathbb{N}_0^d$ of $P_p(\mathbb{R}^d)$, see Def. [3.3.3] and the explanations on multi-index notation (3.3.4).

Then resort to induction w.r.t. degree $p$. 

$$
\Phi_K^*(x^\alpha) = \Phi_K^*(x_1) \cdot \Phi_K^*(x^{\alpha'}) = \left( \sum_{l=1}^{d} (F)_l \hat{x}_l + \tau_1 \right) \cdot \Phi_K^*(x^{\alpha'}) \in P_p(\mathbb{R}^d),
$$

with $\alpha' := (\alpha_1 - 1, \alpha_2, \ldots, \alpha_d)$, where we assumed $\alpha_1 > 0$. Here, we have used the induction hypothesis to conclude $\Phi_K^*(x^\alpha) \in P_{p-1}(\mathbb{R}^d)$.

A simple observation:

Consider $S_1^0(\mathcal{M})$, triangle $K \in \mathcal{M}$, unit triangle $\hat{K}$, affine mapping $\Phi_K : \hat{K} \mapsto K$. 

• $b_1^K, b_2^K, b_3^K$ (standard) local shape functions on $K$,  
• $\hat{b}_1^i, \hat{b}_2^i, \hat{b}_3^i$ (standard) local shape functions on $\hat{K}$,  
\[ \hat{b}_i = \Phi_K^* b_i^K \iff \hat{b}_i(\hat{x}) = b_i^K(x), \quad x = \Phi_K(\hat{x}) \] (3.6.4)

Of course, we assume that $\Phi_K$ respects the local numbering of the vertices of $\hat{K}$ and $K$.

The proof of (3.6.4) is straightforward: both $\Phi_K^* b_i^K$ (by Lemma 3.6.3) and $\hat{b}_i$ are (affine) linear functions that attain the same values at the vertices of $\hat{K}$. Hence, they have to agree.

Note: (3.6.4) holds true for all simplicial Lagrangian finite element spaces

Proof. (of (3.6.4)) Recall the definition of global shape functions and also local shape functions for $S_p^0(M)$, $p \in \mathbb{N}$, by means of the conditions (3.4.3) at interpolation nodes, see Ex. 3.4.2 for $p = 2$. □
Note: we already used the definition of basis functions through basis functions on the “reference cell” \([0, 1]\) and affine pullback in 1D, see Rem. 1.5.33

Now write \( p^i_K \hat{=} \text{(local) interpolation nodes on triangle } K \), \( \hat{p}^i \hat{=} \text{(local) interpolation nodes on unit triangle } \hat{K} \).

Observe: Assuming a matching numbering \( p^i_K = \Phi_K(\hat{p}^i) \), where \( \Phi_K : \hat{K} \mapsto K \) is the unique affine transformation mapping \( \hat{K} \) onto \( K \), see (3.5.29).

This is clear for \( p = 2 \), because affine transformations take midpoints of edges to midpoints of edges. The same applies to the interpolation nodes for higher degree Lagrangian finite elements defined in Ex. 3.4.5.

The local shape functions \( b^i_K \in P_p(\mathbb{R}^d), \hat{b}^i \in P_p(\mathbb{R}^d), i = 1, \ldots, Q \), are uniquely defined by the interpolation conditions

\[
b^i_K(p^K_j) = \delta_{ij}, \quad \hat{b}^i(\hat{p}^j) = \delta_{ij}.
\]

Together with \( p^K_j = \Phi_K(\hat{p}^j) \) this shows that \( \Phi_K^*b^i_K \) satisfies the interpolation conditions (3.6.5) on \( \hat{K} \) and, thus, has to agree with \( \hat{b}^i \).
Terminology: finite element spaces satisfying (3.6.4) are called **affine equivalent**

**Remark 3.6.6 (Evaluation of local shape functions at quadrature points)**.

We consider Lagrangian finite element spaces on a **simplicial mesh** $\mathcal{M}$.

Recall from Sect. 3.5.4: definition (3.5.32) of local quadrature formulas via “unit simplex”.

In particular: quadrature nodes on $K$:

$$\zeta_l^K = \Phi_K(\hat{\zeta}_l)$$

Def. 3.6.2

$$b^i_K(\zeta^K_l) \overset{\text{Def. 3.6.2}}{=} \Phi^*_K(b^i_K)(\hat{\zeta}_l) \overset{(3.6.4)}{=} \hat{b}^i(\hat{\zeta}_l) \text{ independent of } K \, .$$ (3.6.7)

$$\int_K F(b^i_K, b^j_K) \, d\mathbf{x} \approx |K| \sum_{l=1}^{P} \omega_l F(\hat{b}^i(\zeta_l), \hat{b}^j(\zeta_l))$$, (3.6.8)

for any function $F : \mathbb{R}^2 \mapsto \mathbb{R}$.

➢ Precompute $\hat{b}^i(\zeta_l), i = 1, \ldots, Q, l = 1, \ldots, P$ and store the values in a table!
Remark 3.6.9 (Barycentric representation of local shape functions).

We consider Lagrangian finite element spaces on a simplicial mesh $\mathcal{M}$.

(3.4.4): formulas for local shape functions for $S^0_2(\mathcal{M})$ ($d = 2$) in terms of barycentric coordinate functions $\lambda_i$, $i = 1, 2, 3$. Is this coincidence? NO! Does (3.5.19) hold for any (simplicial) Lagrangian finite element space?

\[ b^i_K(x) \overset{(3.6.4)}{=} (\Phi^{-1}_K)^* \left( \hat{x} \mapsto \hat{b}^i(\hat{x}_1, \hat{x}_2) \right), \]
\[ = \hat{b}^i((\Phi^{-1}_K)^*(\hat{\lambda}_2)(x), (\Phi^{-1}_K)^*(\hat{\lambda}_3)(x)) = \hat{b}^i(\lambda_2(x), \lambda_3(x)) \]

where $\lambda_2(\hat{x}) = \hat{x}_1$, $\lambda_3(\hat{x}) = \hat{x}_2$, $\lambda_1(\hat{x}) = 1 - \hat{x}_1 - \hat{x}_2 \overset{\hat{\bullet}}{=} \text{barycentric coordinate functions on } \hat{K}$, see Ex. 3.3.13.

$\lambda_i \overset{\hat{\bullet}}{=} \text{barycentric coordinate functions on triangle } K$, see Fig. [72],

$\Phi_K \overset{\hat{\bullet}}{=} \text{affine transformation (\rightarrow Def. 3.5.27), } \Phi_K(\hat{K}) = K$, see (3.5.29).

By the chain rule:

\[ \text{grad } b^i_K(x) = \frac{\partial \hat{b}^i}{\partial \hat{x}_1}(\hat{x}) \text{grad } \lambda_2 + \frac{\partial \hat{b}^i}{\partial \hat{x}_2}(\hat{x}) \text{grad } \lambda_3, \quad x = \Phi_K(\hat{x}). \]
This formula is convenient, because \( \text{grad} \lambda_i \equiv \text{const} \), see (3.5.22).

This facilitates the computation of element (stiffness) matrices for 2nd-order elliptic problems in variational form: when using a quadrature formula according to (3.5.32)

\[
\int_K (\alpha(x) \text{grad} b_i^j_K) \cdot \text{grad} b_K^j \, dx \\
\approx |K| \sum_{l=1}^{P_K} \omega_l \left( \begin{array}{c} \frac{\partial b_i^j}{\partial x_1}(\tilde{\zeta}_l) \\ \frac{\partial b_i^j}{\partial x_2}(\tilde{\zeta}_l) \end{array} \right)^T \begin{pmatrix} \text{grad} \lambda_1 \cdot \text{grad} \lambda_1 & \text{grad} \lambda_1 \cdot \text{grad} \lambda_2 \\ \text{grad} \lambda_1 \cdot \text{grad} \lambda_2 & \text{grad} \lambda_2 \cdot \text{grad} \lambda_2 \end{pmatrix} \left( \begin{array}{c} \frac{\partial b_j^i}{\partial x_1}(\tilde{\zeta}_l) \\ \frac{\partial b_j^i}{\partial x_2}(\tilde{\zeta}_l) \end{array} \right)
\]

This is very interesting, because

- the values \( \frac{\partial b_i^j}{\partial x_1}(\tilde{\zeta}_l) \) can be precomputed,
- simple expressions for \( \text{grad} \lambda_i \cdot \text{grad} \lambda_j \) are available, see Sect. 3.2.5.
So far, see Sect. 3.3.3 and (3.3.11), we have adopted the perspective

\[ \text{Restriction to element} \quad \rightarrow \quad \text{local shape functions} \]

Now we reverse this construction

\[ \text{local shape functions} \quad \rightarrow \quad \text{“glueing”} \quad \rightarrow \quad \text{global shape functions} \quad (3.6.10) \]

In fact, when building the global basis functions for quadratic Lagrangian finite elements we already proceeded this way, see Ex. 3.4.2. Fig. 94 lucidly conveys what is meant by “glueing”.

Be aware that the possibility to achieve a continuous global basis function by glueing together local shape function on adjacent cells, entails a judicious choice of the local shape functions.

This section will demonstrate how the policy (3.6.10) together with the formula (3.6.4) will enable us to extend Lagrangian finite element beyond the meshes discussed in Sect. 3.4.
What is \( S_1^0(\mathcal{M}) \)?

Clear: If \( K \) is a rectangle, \( \hat{K} \) the unit square, then there is a unique affine transformation \( \Phi_K \) (\( \rightarrow \) Def. 3.5.27) with \( K = \Phi_K(\hat{K}) \).

In this case (3.6.4) holds for the local shape functions of bilinear Lagrangian finite elements from Ex. 3.4.6 (and all tensor product Lagrangian finite elements introduced in Sect. 3.4.2).

**Idea:**
- **local shape functions** \( \rightarrow \) **global shape functions**

  Build local shape functions by “inverse pullback”

\[
 b^i_K = (\Phi_K^{-1})^* \hat{b}^i ,
\]

where \( \{\hat{b}^i\}_{i=1}^Q \doteq \) set of shape functions on reference element \( \hat{K} \).
What is $\Phi_K$ for a general quadrilateral?

Affine transformations fail to produce general quadrilaterals from a square. They only give parallelograms.

It takes bilinear transformations to obtain a generic quadrilateral from the unit square.
Bilinear transformation of unit square to quadrilateral with vertices $\mathbf{a}^i$, $i = 1, 2, 3, 4$:

$$
\Phi_K(\hat{x}) = (1 - \hat{x}_1)(1 - \hat{x}_2) \mathbf{a}^1 + \hat{x}_1(1 - \hat{x}_2) \mathbf{a}^2 + \hat{x}_1\hat{x}_2 \mathbf{a}^3 + (1 - \hat{x}_1)\hat{x}_2 \mathbf{a}^4. \tag{3.6.12}
$$

The mapping property $\Phi_K(\hat{a}^i) = \mathbf{a}^i$ is evident. In order to see $\Phi_K(\hat{K}) = K$ ($\hat{K}$ = unit square) for (3.6.12), verify that $\Phi_K$ maps all parallels to the coordinate axes to straight lines.

Moreover, a simple computation establishes:

If $\hat{K}$ is the unit square, $\Phi_K : \hat{K} \mapsto K$ a bilinear transformation, and $\hat{b}^i$ the bilinear local shape functions (3.4.8) on $\hat{K}$,

$$
\text{then } (\Phi_K^{-1})^{*}\hat{b}^i \text{ are linear on the edges of } K.
$$
“Glueing” of local shape functions possible

Explanation:

1. Pick a vertex \( \mathbf{x} \in V(\mathcal{M}) \) and consider an adjacent quadrilateral \( K \), on which there is a local shape function \( b^i_K \) such that \( b^i_K(\mathbf{x}) = 1 \) and \( b^i_K \) vanishes on all other vertices of \( K \). This local shape function is obtained by inverse pullback of the \( \tilde{b}^i \) associated with \( \Phi^{-1}_K(\mathbf{x}) \).

2. The same construction can be carried out for another quadrilateral \( \tilde{K} \) that shares the vertex \( \mathbf{x} \) and an edge \( e \) with \( K \). On that quadrilateral we find the local shape function \( b^j_{\tilde{K}} \).

3. Both \( b^i_K|_e \) and \( b^j_{\tilde{K}}|_e \) are linear and attain the same values, that is 0 and 1 at the endpoints \( \mathbf{x} \) and \( \mathbf{y} \) of \( e \), respectively.
Continuity of global shape function (defined by interpolation conditions at nodes)

Remark 3.6.13 (Non-polynomial “bilinear” local shape functions).

The local shape functions $b^i_K$ defined by (3.6.11), where $\Phi_K$ is a bilinear transformation and $\hat b^i$ are the bilinear local shape functions on the unit square, are not polynomial in general.

Visualization of local shape functions on trapezoidal cell $K := \text{convex}\left\{ \left(0\right), \left(\frac{3}{4}\right), \left(\frac{1}{2}\right), \left(\frac{1}{4}\right) \right\}$:
3.6.3 Transformation techniques

“Bilinear” Lagrangian finite elements = a specimen of parametric finite elements
Definition 3.6.14 (Parametric finite elements).

A finite element space on a mesh \( \mathcal{M} \) is called parametric, if there exists a reference element \( \hat{K}, Q \in \mathbb{N} \), and functions \( \hat{b}^i \in C^0(\hat{K}) \), \( i = 1, \ldots, Q \), such that

\[
\forall K \in \mathcal{M}: \exists \text{ bijection } \Phi_K : \hat{K} \mapsto K: \hat{b}^i = \Phi_K^* b_K^i, \quad i = 1, \ldots, Q,
\]

where \( \{b^1_K, \ldots, b^Q_K\} = \text{set of local shape functions on } K \).

This definition takes the possibility of “glueing” for granted: the concept of a local shape function, see (3.3.11), implies the existence of a global shape function with the right continuity properties.

How to implement parametric finite elements?

We consider a generic elliptic 2nd-order variational Dirichlet problem

\[
\begin{align*}
u \in H^1(\Omega), \quad & \int_{\Omega} (\alpha(x) \text{grad } u(x)) \cdot \text{grad } v(x) \, dx = \int_{\Omega} f(x)v(x) \, dx \quad \forall v \in H^1_0(\Omega). \tag{2.3.3}
\end{align*}
\]
Issue: computation of element (stiffness) matrices and element (load) vectors (→ Def. 3.5.12).

Challenge: local shape functions \( b_K^1, \ldots, b_K^Q, K \in \mathcal{M} \), only known implicitly

\[ b_K^i = (\Phi_K^{-1})^* \hat{b}^i \]

\[ \rightarrow \] Known: transformation \( \Phi : \hat{K} \mapsto K, \hat{K} \) reference element, functions \( \hat{b}^1, \ldots, \hat{b}^Q \)

\[ \hat{b}^i = \Phi^* b^i_K, \quad i = 1, \ldots, Q \] (→ pullback, Def. 3.6.2)

Use transformation to \( \hat{K} \) to compute element stiffness matrix \( \mathbf{A}_K \),

element load vector \( \mathbf{\varphi}_K \):

\[ (\mathbf{A}_K)_{ij} = \int_K \alpha(x) \text{grad} \ b_K^j(x) \cdot \text{grad} \ b_K^i(x) \, dx \]

\[ = \int_{\hat{K}} (\Phi^* \alpha)(\hat{x}) (\Phi^* (\text{grad} \ b_K^j))(\hat{x}) \cdot (\Phi^* (\text{grad} \ b_K^i))(\hat{x}) \left| \det D \Phi(\hat{x}) \right| d\hat{x}, \]

\[ (\mathbf{\varphi}_K)_i = \int_K f(x) b_K^i(x) \, dx = \int_{\hat{K}} (\Phi_K^* f)(\hat{x}) \hat{b}_K^i(\hat{x}) \left| \det D \Phi(\hat{x}) \right| d\hat{x}, \]
by transformation formula (for multidimensional integrals, see also (3.5.31)):

$$\int_K f(x) \, dx = \int_{\hat{K}} f(\hat{x}) \left| \det D\Phi(\hat{x}) \right| d\hat{x} \quad \text{for } f : K \mapsto \mathbb{R},$$

(3.6.15)

All integrals have been transformed to the reference element $\hat{K}$, where we apply a quadrature formula (3.5.32).

Needed: values of determinant of Jacobi matrix $D\Phi$ at quadrature nodes $\hat{\zeta}_l$.

Also needed: gradients $\Phi^*(\text{grad } b^i_K)$ at quadrature nodes $\hat{\zeta}_l$!? (Seems to be a problem as $b^i_K$ may be elusive, cf. Rem. 3.6.13!)

**Lemma 3.6.16** (Transformation formula for gradients).

*For differentiable $u : K \mapsto \mathbb{R}$ and any diffeomorphism $\Phi : \hat{K} \mapsto K$ we have*

$$\left( \text{grad}_{\hat{x}}(\Phi^*u) \right)(\hat{x}) = \left( D\Phi(\hat{x}) \right)^T \underbrace{\left( \text{grad}_x u \right)(\Phi(\hat{x}))}_{= \Phi^*(\text{grad } u)(x)} \quad \forall \hat{x} \in \hat{K}.$$

(3.6.17)
Proof: use chain rule for components of the gradient

\[
\frac{\partial \Phi^* u}{\partial \hat{x}_i}(\hat{x}) = \frac{\partial}{\partial \hat{x}_i} u(\Phi(\hat{x})) = \sum_{j=1}^d \frac{\partial u}{\partial x_j}(\Phi(\hat{x})) \frac{\partial \Phi_j}{\partial \hat{x}_i}(\hat{x}) .
\]

Here, \( D\Phi(\hat{x}) \in \mathbb{R}^{d,d} \) is the Jacobian of \( \Phi \) at \( \hat{x} \in \hat{K} \), see [29, Bem. 7.6.1].

Using Lemma 3.6.16 we arrive at:

\[
(A_K)_{ij} = \int_{\hat{K}} \left( \alpha(\Phi(\hat{x}))(D\Phi)^{-T} \, \text{grad} \, \hat{b}^i \right) \cdot \left( (D\Phi)^{-T} \, \text{grad} \, \hat{b}^j \right) | \det D\Phi | \, d\hat{x} . 
\tag{3.6.18}
\]

Note that the argument \( \hat{x} \) is suppressed for some terms in the integrand.

\( \text{notation: } M^{-T} := (M^{-1})^T = (M^T)^{-1} \)
Example 3.6.19 (Transformation techniques for bilinear transformations).

\[ \Phi(\hat{x}) = \begin{pmatrix} \alpha_1 + \beta_1 \hat{x}_1 + \gamma_1 \hat{x}_2 + \delta_1 \hat{x}_1 \hat{x}_2 \\ \alpha_2 + \beta_2 \hat{x}_1 + \gamma_2 \hat{x}_2 + \delta_2 \hat{x}_1 \hat{x}_2 \end{pmatrix}, \quad \alpha_i, \beta_i, \gamma_i, \delta_i \in \mathbb{R}, \]

\[ \Rightarrow D\Phi(\hat{x}) = \begin{pmatrix} \beta_1 + \delta_1 \hat{x}_2 & \gamma_1 + \delta_1 \hat{x}_1 \\ \beta_2 + \delta_2 \hat{x}_2 & \gamma_2 + \delta_2 \hat{x}_1 \end{pmatrix}, \]

\[ \Rightarrow \det(D\Phi(\hat{x})) = \beta_1 \gamma_2 - \beta_2 \gamma_1 + (\beta_1 \delta_2 - \beta_2 \delta_1) \hat{x}_1 + (\delta_1 \gamma_2 - \delta_2 \gamma_1) \hat{x}_2. \]

Both \( D\Phi(\hat{x}) \) and \( \det(D\Phi(\hat{x})) \) are (componentwise) linear in \( x \).

If \( \Phi = \Phi_K \) for a generic quadrilateral \( K \) as in (3.6.12), then the coefficients \( \alpha_i, \beta_i, \gamma_i, \delta_i \) depend on the shape of \( K \) in a straightforward fashion:

\[ \begin{pmatrix} \alpha_1 \\ \alpha_2 \end{pmatrix} = a^1, \quad \begin{pmatrix} \beta_1 \\ \beta_2 \end{pmatrix} = a^2 - a^1, \quad \begin{pmatrix} \gamma_1 \\ \gamma_2 \end{pmatrix} = a^4 - a^1, \quad \begin{pmatrix} \delta_1 \\ \delta_2 \end{pmatrix} = a^3 - a^2 - a^4 + a^1. \]
3.6.4 Boundary approximation

Intuition: Approximating a (smooth) curved boundary $\partial \Omega$ by a polygon/polyhedron will introduce a (sort of) discretization error.

Parametric finite element constructions provide a tool for avoiding polygonal/polyhedral approximation of boundaries.

Here we discuss this for a very simple case of triangular meshes in 2D (more details $\rightarrow$ [5, Sect, 10.2]).
Idea: Piecewise polynomial approximation of boundary (boundary fitting)
(\(\partial \Omega\) locally considered as function over straight edge of an element)

Example: Piecewise quadratic boundary approximation
(Part of \(\partial \Omega\) between \(a_1\) and \(a_2\) approximated by parabola)

Mapping \(\tilde{K} \rightarrow \text{"curved element" } K\):

\[
\tilde{\Phi}_K(\tilde{x}) := \tilde{x} + 4\delta \lambda_1(\tilde{x})\lambda_2(\tilde{x}) n.
\] (3.6.20)

(\(\lambda_i\) barycentric coordinate functions on \(\tilde{K}\), \(n\) normal to \(E_\Gamma\), see Fig. 122)

Note: Essential: \(\delta\) sufficiently small \(\implies\) \(\Phi\) bijective

The complete transformation \(\Phi_K : \tilde{K} \mapsto K\) is obtained by joining an affine transformation (→
Def. 3.5.27) \( \Phi^a_K : \tilde{K} \mapsto \tilde{K}, \ \Phi^a_K(\tilde{x}) := F_K \tilde{x} + \tau_K, \) and \( \tilde{\Phi}_K : \)

\[ \Phi_K = \tilde{\Phi}_K \circ \Phi^a_K. \]

For parabolic boundary fitting:

\[ D\tilde{\Phi}_K = I + 4\delta n \cdot \nabla (\lambda_1 \lambda_2) \in \mathbb{R}^{2,2}, \quad \det(D\tilde{\Phi}_K) = 1 + 4\delta n \cdot \nabla (\lambda_1 \lambda_2). \]

### 3.7 Linearization
So far we have discussed the finite elements for *linear* second-order variational boundary value problems only.

However, as we have learned in Ex. 1.5.66, in 1D the Galerkin approach based on linear finite elements was perfectly capable of dealing with *non-linear* two-point boundary value problems. Indeed the abstract discussion of the Galerkin approach in Sect. 1.5.1 was aimed at general and possibly non-linear variational problems, see (1.5.7), (1.5.18).

It goes without saying that the abstract (and formal) discussion of Sect. 1.5.1 remains true for *non-linear* second-order boundary value problems in variational form.

**Difficult:** Characterization of “spaces of functions with finite energy” (→ Sobolev spaces, Sect. 2.2) for non-linear variational problems.

(Relief!) In this course we do not worry that much about function spaces.
Recall (→ Rem. 1.3.12): Non-linear variational problem

\[ u \in V: \quad a(u; v) = \ell(v) \quad \forall v \in V_0, \]

**V**

- **V** \( V_0 \) \( \hat{} \) test space, (real) vector space (usually a function space, “Sobolev-type” space \( \rightarrow \) Sect. 2.2)
- **V** \( V \) \( \hat{} \) trial space, affine space: usually \( V = u_0 + V_0 \), with offset function \( u_0 \in V \)
- **f** \( f \) \( \hat{} \) a linear mapping \( V_0 \mapsto \mathbb{R} \), a linear form,
- **a** \( a \) \( \hat{} \) a mapping \( V \times V_0 \mapsto \mathbb{R} \), linear in the second argument, that is

\[ a(u; \alpha v + \beta w) = \alpha a(u; v) + \beta a(u; w) \quad \forall u \in V, v, w \in V_0, \alpha, \beta \in \mathbb{R}. \]

**Example 3.7.2** (Heat conduction with radiation boundary conditions).
2nd-order elliptic boundary value problem, cf. (2.5.6) & (2.6.3)

\[- \text{div} (\kappa(x) \text{grad} u) = f \quad \text{in} \ \Omega, \]
\[
\kappa(x) \text{grad} u \cdot n(x) + \Psi(u) = 0 \quad \text{on} \ \partial \Omega.
\]

Variational formulation from Ex. 2.8.5

\[
u \in H^1(\Omega): \quad \int_{\Omega} \kappa(x) \text{grad} u \cdot \text{grad} v \, dx + \int_{\partial \Omega} \Psi(u) v \, dS = \int_{\Omega} f v \, dx \quad \forall v \in H^1(\Omega). \quad (2.8.7)
\]

If \( \Psi : \mathbb{R} \rightarrow \mathbb{R} \) is not an affine linear function, then (2.8.7) represents a non-linear variational problem (1.3.14) with

- trial/test space \( V = V_0 = H^1(\Omega) \) (→ Def. 2.2.12),
- right hand side linear form \( \ell(v) := \int_{\Omega} f v \, dx \),
- \( a(u; v) := \int_{\Omega} \kappa(x) \text{grad} u \cdot \text{grad} v \, dx + \int_{\partial \Omega} \Psi(u) v \, dS. \)

Note that the non-linearity enters only through the boundary term.
Pursuing the policy of Galerkin discretization (choice of discrete spaces and corresponding bases, → Sect. 1.5.1) we can convert (1.3.14) into a non-linear system of equations

\[ a(u_0 + \sum_{j=1}^{N} \mu_j b_j^N; b_k^N) = f(b_k^N) \quad \forall k = 1, \ldots, N. \]  

(1.5.18)

If the left hand side depends smoothly on the unkowns (the corefficients \( \mu_j \) of \( \vec{\mu} \)), then the classical Newton method (→ [18, Sect. 4.4]) to solve it iteratively.

Here, we focus on a different approach that reverses the order of the steps:

1. Linearization of problem (“Newton in function space”),
2. Galerkin discretization of linearized problems.

“Newton in function space”:

Recall idea of Newton’s method [18, Sect. 4.4] for the iterative solution of \( F(x) = 0, \quad F : D \subset \mathbb{R}^N \mapsto \mathbb{R}^N \) smooth:
Idea: local linearization:

Given $\tilde{\xi}^{(k)} \in D \Rightarrow \tilde{\xi}^{(k+1)}$ as zero of affine linear model function

$$F(\tilde{\xi}) \approx \tilde{F}(\tilde{\xi}) := F(\tilde{\xi}^{(k)}) + DF(\tilde{\xi}^{(k)})(\tilde{\xi} - \tilde{\xi}^{(k)}) .$$

Newton iteration:

$$\tilde{\xi}^{(k+1)} := \tilde{\xi}^{(k)} - DF(\tilde{\xi}^{(k)})^{-1}F(\tilde{\xi}^{(k)}) , \quad [ \text{if } DF(\tilde{\xi}^{(k)}) \text{ regular} ] \quad (3.7.3)$$

← apply idea to (1.3.14)

Idea: local linearization:

Given $u^{(k)} \in V \Rightarrow u^{(k+1)}$ from

$$w \in V_0 : a(u^{(k)}; v) + Du a(u^{(k)}; v) w = \ell(v) \quad \forall v \in V_0 ,$$

$$u^{(k+1)} := u^{(k)} + w . \quad (3.7.4)$$

The meaning of $DF(\tilde{\xi}^{(k)})$ in (3.7.3) is clear: it stands for the Jacobian of $F$ evaluated at $\tilde{\xi}^{(k)}$.

But what is the meaning of $Du a(u^{(k)}; v) w$ in (3.7.4)?
Remember the “definition” of the Jacobian (for sufficiently smooth $F$)

$$DF(\vec{\xi})\vec{\mu} = \lim_{t \to 0} \frac{F(\vec{\xi} + t\vec{\mu}) - F(\vec{\xi})}{t}, \quad \vec{\xi} \in D, \vec{\mu} \in \mathbb{R}^N. \quad (3.7.5)$$

➢ try the “definition”

$$D_u a(u^{(k)}; v)w = \lim_{t \to 0} \frac{a(u + tw; v) - a(u; v)}{t}, \quad u^{(k)} \in V, \quad v, w \in V_0. \quad (3.7.6)$$

If $(u, v) \mapsto a(u; v)$ depends smoothly on $u$, then

$$(v, w) \mapsto D_u a(u^{(k)}; v)w \quad \text{is a bilinear form} \quad V_0 \times V_0 \mapsto \mathbb{R}.$$  

**Example 3.7.7** (Derivative of non-linear $u \mapsto a(u; \cdot)$).

Apply formula (3.7.6) to the non-linear boundary term in (2.8.7), that is, here

$$a(u; v) := \int_{\partial \Omega} \Psi(u)v \, dS, \quad u, v \in H^1(\Omega).$$
\[
\mathbf{a}(u + tw; v) - \mathbf{a}(u; v) = \int_{\partial \Omega} (\Psi(u + tw) - \Psi(u))v \, dS, \quad u, v \in H^1(\Omega).
\]

Assume \( \Psi : \mathbb{R} \mapsto \mathbb{R} \) is smooth with derivative \( \Psi' \) and employ Taylor expansion for fixed \( w \in H^1(\Omega) \) and \( t \to 0 \)

\[
\mathbf{a}(u + tw; v) - \mathbf{a}(u; v) = \int_{\partial \Omega} t \Psi'(u)wv \, dS + O(t^2).
\]

\[
\mathbf{D}_u \mathbf{a}(u^{(k)}; v)w = \lim_{t \to 0} \frac{\mathbf{a}(u + tw; v) - \mathbf{a}(u; v)}{t} = \int_{\partial \Omega} \Psi'(u)wv \, dS.
\]

= a bilinear form in \( v, w \) on \( H^1(\Omega) \times H^1(\Omega) \)!

This example also demonstrates how to actually compute \( \mathbf{D}_u \mathbf{a}(u^{(k)}; v)w \)!

**Idea:** Galerkin discretization of the linear variational problem from (3.7.4)

\[
w \in V_0: \quad \mathbf{c}(w, v) = g(v) \quad \forall v \in V_0,
\]

\[
\mathbf{c}(w, v) = \mathbf{D}_u \mathbf{a}(u^{(k)}; v)w, \quad g(v) := \ell(v) - \mathbf{a}(u^{(k)}; v).
\]
Newton-Galerkin iteration for (1.3.14)

Given $u^{(k)}_N \in V^{(k)}_N \Rightarrow u^{(k+1)}_N \in V^{(k+1)}_N$ from

$$w_N \in V^{(k+1)}_{0,N}: \quad D_ua(u^{(k)}_N;v_N)w_N = \ell(v_N) - a(u^{(k)}_N;v_N) \quad \forall v_N \in V^{(k+1)}_{0,N},$$

(3.7.8)

Newton update

Note: different Galerkin trial/test spaces $V^{(k)}_N$, $V^{(k+1)}_{0,N}$ may be used in different steps of the iteration!

(It may enhance efficiency to use Galerkin trial/test spaces of a rather small dimension in the beginning and switch to larger when the iteration is about to converge.)

Warning! If $V^{(k)}_N \neq V^{(k+1)}_N$ you cannot simply add $u^{(k)}_N$ and $w$

Linear projection operator $P^{(k+1)}_N : V^{(k)}_N \mapsto V^{(k+1)}_N$ required in (3.7.8)
Any of the Lagrangian finite element spaces introduced in Sect. 3.4 will supply valid $V_N/V_{0,N}$. Offset functions can be chosen according to the recipes from Sect. 3.5.5.

Important aspect: termination of iteration, see [18, Thm. 4.4.3].

Option: termination based on relative size of Newton update, with $w, u_N^{(k+1)}$ from (3.7.8)

$$\text{STOP, if } \|w\| \leq \tau \|u_N^{(k+1)}\|, \quad (3.7.9)$$

where $\|\cdot\|$ is a relevant norm (e.g., energy norm) on $V_N^{(k+1)}$ and $\tau > 0$ a prescribed relative tolerance.
Finite Differences (FD) and Finite Volume Methods (FV)

Now we examine two approaches to the discretization of scalar linear 2nd-order elliptic BVPs that offer an alternative to finite element Galerkin methods discussed in Ch. 3.

What these methods have in common with (low degree) Lagrangian finite element methods is

- that they rely on meshes (→ Sect. 3.3.1) tiling the computational domain $\Omega$,
- they lead to \textit{sparse} linear systems of equations.
Remark 4.0.1 (Collocation approach on “complicated” domains).

Sect. 1.5.2.2 taught us spline collocation methods. A crucial insight was that collocation methods (see beginning of Sect. 1.5.2 for a presentation of the idea), which target the boundary value problem in ODE/PDE form, have to employ discrete trial spaces comprised of \textit{continuously differentiable} functions, see Rem. 1.5.82.

It is very difficult to construct spaces of piecewise polynomial $C^1$-functions on non-tensor product domains for $d = 2, 3$ and find suitable \textit{collocation nodes}, cf. (1.5.78).

Therefore we skip the discussion of collocation methods for 2nd-order elliptic BVPs on $\Omega \subset \mathbb{R}^d$, $d = 2, 3$. \triangleq
4.1 Finite differences

A finite difference scheme for a 2-point boundary value problem was presented in Sect. 1.5.3, which you are advised to browse again. Its gist was

- to replace the derivatives in the *differential equation* with *difference quotients* connecting approximate values of the solutions *at the nodes of a grid/mesh*.

Recall: Finite differences target the “ODE/PDE-formulation” of the boundary value problem.

Our goal: extension to higher dimensions
2D model problem:

Homogeneous Dirichlet BVP for Laplacian:

$$-\Delta u = -\frac{\partial^2 u}{\partial x_1^2} - \frac{\partial^2 u}{\partial x_2^2} = f \quad \text{in } \Omega := [0, 1]^2,$$

$$u = 0 \quad \text{on } \partial \Omega.$$

Discretization based on

\(\mathcal{M} = \) (triangular) tensor-product grid

(meshwidth \( h = (1 + N)^{-1}, N \in \mathbb{N} \))

lexikographic (line-by-line) ordering of nodes of \(\mathcal{M}\)

finite difference approach to \(-\Delta\): approximation of derivatives by symmetric difference quotients

This is nothing new: we did this in (1.5.97).
\[
\frac{\partial^2 u}{\partial x_1^2} \bigg|_{x=(\xi,\eta)} \approx \frac{u(\xi - h, \eta) - 2u(\xi, \eta) + u(\xi + h, \eta)}{h^2},
\]
\[
\frac{\partial^2 u}{\partial x_2^2} \bigg|_{x=(\xi,\eta)} \approx \frac{u(\xi, \eta - h) - 2u(\xi, \eta) + u(\xi, \eta + h)}{h^2}.
\]
\[
-\Delta u \bigg|_{x=(\xi,\eta)} \approx \frac{1}{h^2}(4u(\xi, \eta) - u(\xi - h, \eta) - u(\xi + h, \eta) - u(\xi, \eta - h) - u(\xi, \eta + h)).
\]

Use this approximation at grid point \( p = (ih, jh) \). This will connect the five point values \( u(ih, jh), u((i - 1)h, jh), u((i + 1)h, jh), u(ih, (j - 1)h), u(ih, (j + 1)h) \).

Approximations \( \mu_{i,j} \) to the point values \( u(ih, jh) \) will be the unknowns of the finite difference method.

Centering the above difference quotients at grid points yields linear relationships between the unknowns:
\[
\frac{1}{h^2}(4u(ih, jh) - u(ih - h, jh) - u(ih + h, jh) - u(ih, jh - h) - u(ih, jh + h)) = f(ih, jh),
\]
\[
\frac{1}{h^2}(4\mu_{i,j} - \mu_{i-1,j} - \mu_{i+1,j} - \mu_{i,j-1} - \mu_{i,j+1}) = f(ih, jh). \quad (4.1.1)
\]

Also this is familiar from the discussion in 1D. Yet, in 1D the association of the point values and of components of the vector \(\vec{\mu}\) of unknowns was straightforward and suggested by the linear ordering of the nodes of the grid. In 2D we have much more freedom.

One option on tensor-product grids is the line-by-line ordering (lexikographic ordering) depicted in Fig. [123]. This allows a simple indexing scheme:

\[
u(p) \leftrightarrow \mu_{i,j} \leftrightarrow \mu_{(j-1)N+i}
\]
\[
\begin{align*}
-\mu(j-2)N+i - \mu(j-1)N+i-1 + \frac{4\mu(j-1)N+i - \mu(j-1)N+i+1 - \mu jN+i}{h^2} = \\
\underbrace{f(\nu h, jh)}_{=\varphi(j-1)N+i}.
\end{align*}
\]

Linear system of \(N^2\) equations \(A\vec{\mu} = \vec{\varphi}\) with \(N^2 \times N^2\) block-tridiagonal Poisson matrix

\[
A := \frac{1}{h^2} \begin{pmatrix}
T & -I & 0 & \cdots & \cdots & 0 \\
-I & T & -I & \ddots & \ddots & \vdots \\
0 & -I & T & -I & \ddots & \vdots \\
\vdots & \ddots & \ddots & \ddots & \ddots & 0 \\
0 & \cdots & \cdots & 0 & -I & T \\
0 & \cdots & \cdots & 0 & -I & T
\end{pmatrix},
\quad
T := \begin{pmatrix}
4 & -1 & 0 & 0 \\
-1 & 4 & -1 & \ddots \\
0 & -1 & 4 & -1 & \ddots \\
\vdots & \ddots & \ddots & \ddots & \ddots \\
0 & \cdots & \cdots & -1 & 4 & -1 \\
0 & \cdots & \cdots & 0 & -1 & 4
\end{pmatrix} \in \mathbb{R}^{N,N}
\]

Band structure of Poisson matrix

The MATLAB command

\[
A = \text{gallery('poisson',n)}
\]

creates a sparse \(n^2 \times n^2\) Poisson matrix.
Already in Sect. 1.5.3 we saw that the linear system of equations popping out of the finite difference discretization of the linear two-point BVP (1.5.81) was the same as that obtained via the linear finite Galerkin approach on the same mesh.

In two dimensions we will also come to this conclusion! So, let us derive the Galerkin matrix and right hand side vector for the 2D model problem on the tensor product mesh depicted in Fig. 123. To begin with we convert it into a triangular mesh $\mathcal{M}$ by splitting each square into two equal triangles by inserting a diagonal (green lines in Fig. 123). On this mesh we use linear Lagrangian finite elements as in Sect. 3.2.

Then we repeat the considerations of Sect. 3.2.

2 Linear Lagrangian finite element Galerkin discretization $\rightarrow$ Sect. 3.2: $V_{0,N} = S_{1,0}^0(\mathcal{M})$

(global shape functions $\hat{=} \text{"tent functions"}$, $\rightarrow$ Fig. 87)
Element stiffness matrix from (3.2.10):

\[
A_K = \frac{1}{2} \begin{pmatrix}
2 & -1 & -1 \\
-1 & 1 & 0 \\
-1 & 0 & 1 \\
\end{pmatrix}.
\]

(← numbering of local shape functions)

Element load vector: use three-point quadrature formula (3.5.35)

\[
\bar{\varphi}_K = \frac{1}{6} h^2 \begin{pmatrix}
f(a^1) \\
f(a^2) \\
f(a^3) \\
\end{pmatrix}.
\]
Local assembly:

← green: local vertex numbers

Contributions to load vector component associated with node \( p \):

- From \( K_1 \) : \( (\vec{\phi}_{K_1})_2 \)
- From \( K_2 \) : \( (\vec{\phi}_{K_2})_3 \)
- From \( K_3 \) : \( (\vec{\phi}_{K_3})_3 \)
- From \( K_4 \) : \( (\vec{\phi}_{K_4})_1 \)
- From \( K_5 \) : \( (\vec{\phi}_{K_5})_1 \)
- From \( K_6 \) : \( (\vec{\phi}_{K_6})_2 \)

\[ \vec{\phi}_p = h^2 f(p) . \]

Assembly of finite element Galerkin matrix from element (stiffness) matrices (→ Sect. 3.5.3):
\[
\frac{1}{2} \begin{pmatrix}
1 & 0 & -1 \\
-1 & 2 & -1 \\
0 & -1 & 1
\end{pmatrix}
\]

\[
\frac{1}{2} \begin{pmatrix}
1 & -1 & 0 \\
-1 & 2 & -1 \\
0 & -1 & 1
\end{pmatrix}
\]
\[ N^2 \times N^2 \text{ linear system of equations } h^2 A \vec{\mu} = h^2 \vec{\varphi}, \ A \doteq \text{Poisson matrix} \] (4.1.3)

Finite element Galerkin schemes

(Most) finite difference schemes ↔ finite element Galerkin schemes with numerical quadrature on structured meshes

Discussion: finite differences vs. finite element Galerkin methods (here focused on 2nd-order linear scalar problems)

- Finite element methods can be used on general triangulations and structured (tensor-product) meshes alike, which delivers superior flexibility in terms of geometry resolution (advantage FEM).
- The correct treatment of all kinds of boundary conditions (→ Sect. 2.6). naturally emerges from the variational formulations in the finite element method (advantage FEM).
- Finite element methods have built-in “safety rails” because there are clear criteria for choosing viable finite element spaces and once this is done, there is no freedom left to go astray (advantage FEM).
Finite element methods are harder to understand (advantage FD, but only with students who have not attended this course!)

Then, why are “finite difference methods” ubiquitous in scientific and engineering simulations?

When people talk “finite differences” they have in mind structured meshes (translation invariant, tensor product structure) and use the term as synonym for “discretization on structured meshes”. The popularity of structured meshes is justified:

- structured meshes allow regular data layout and vectorization, which boost the performance of algorithms on high performance computing hardware.

  → course “Parallel Computing for Scientific Simulations”

- translation invariant PDE operators give rise to simple Galerkin matrices that need not be assembled and stored (recall the 5-point-stencil for $-\Delta$) and support very efficient matrix $\times$ vector operations.

Use structured meshes whenever possible!
4.2 Finite volume methods (FVM)

4.2.1 Gist of FVM

Focus: linear scalar 2nd-order elliptic boundary value problem in 2D (→ Sect. 2.5), homogeneous Dirichlet boundary conditions (→ Sect. 2.6), uniformly positive scalar heat conductivity $\kappa = \kappa(x)$

$$-\text{div}(\kappa(x)\ \text{grad} \ u) = f \quad \text{in} \ \Omega, \quad u = 0 \quad \text{on} \ \partial\Omega.$$  

Finite volume methods for 2nd-order elliptic BVP are inspired by the conservation principle (2.5.2).

$$\int_{\partial V} j \cdot n \ dS = \int_V f \ dx \quad \text{for all “control volumes” } V.$$  \hspace{1cm} (2.5.2)

Physics requires that this holds for all (infinitely many) “control volumes” $V \subset \Omega$.

Since discretization has to lead to a finite number of equations, the idea is to demand that (2.5.2) holds for only a finite number of special control volumes.
First ingredient: (finitely many) control volumes

Concrete choice:

Control volumes =

(polygonal) cells of a mesh $\tilde{\mathcal{M}} = \{C_i\}$
covering computational domain $\Omega$.

Associate cell $C_i \leftrightarrow$ nodal value $\mu_i$

Meaning: $\mu_i \approx u(p_i)$, $p_i$ = “center” of $C_i$

The conservation law (2.5.2) had to be linked to the flux law (2.5.3) in order to give rise to a 2nd-order scalar PDE see (2.5.5)–(2.5.6).
Correspondingly, “heat conservation in control volumes” has to be supplemented by a rule that furnishes the heat flux between two adjacent control volumes.

**Second ingredient:** local numerical fluxes

For two adjacent cells $C_k, C_i$ with common edge $\Gamma_{ik} := C_i \cap \overline{C_k}$.

Numerical flux

$$J_{ik} = \Psi(\mu_i, \mu_k) \approx \int_{\Gamma_{ik}} j \cdot n_{ik} \, dS$$

($\Psi =$ numerical flux function, $j =$ (heat) flux, see (2.5.1), $n_{ik} \hat{=} =$ edge normal.)

Idea: consider balance law on (finitely many !) control volumes $C_i$

$$\int_{\partial C_i} j \cdot n \, dS = \int_{C_i} f \, dx \Rightarrow \sum_{k \in U_i} J_{ik} = \int_{C_i} f \, dx .$$

notation: $U_i := \{ j : C_i \text{ and } C_j \text{ share edge, } C_j \in \widetilde{M} \}$, $p_i =$ node associated with control volume $C_i$. 
System of equations \( \tilde{M} := \# \tilde{M} \) equations, unknowns \( \mu_i \):

\[
\sum_{k \in U_i} \Psi(\mu_i, \mu_k) = \int_{C_i} f \, dx \quad \forall i = 1, \ldots, \tilde{M}.
\] (4.2.1)

Further approximation: 1-point quadrature for approximate evaluation of integral over \( C_i \),

\[
\sum_{k \in U_i} \Psi(\mu_i, \mu_k) = |C_i| f(p_i) \, dx \quad \forall i = 1, \ldots, \tilde{M}.
\] (4.2.2)

Note: homogeneous Dirichlet problem  ➢ only “interior” control volumes in (4.2.2)

### 4.2.2 Dual meshes

Dual meshes are a commonly used technique for the construction of control volumes for FVM, based on conventional FE triangulation \( M \) of \( \Omega \) (\( \rightarrow \) Sect. 3.3.1).
Focus: dual mesh for triangular mesh $\mathcal{M}$ in 2D, $\Omega$ polygon

Popular choice: Voronoi dual mesh

![Diagram with Voronoi dual mesh](image)

Voronoi cells

$$C_i := \{ \mathbf{x} \in \Omega : |\mathbf{x} - \mathbf{p}_i| < |\mathbf{x} - \mathbf{p}_j| \quad \forall j \neq i \}.$$  (4.2.3)

Voronoi dual mesh $\widetilde{\mathcal{M}} := \{ C_i \}_{i=1}^M$

Fig. 127
Construction of Voronoi dual cells:
edges $\rightarrow$ perpendicular bisectors
nodes $\rightarrow$ circumcenters of triangles

straightforward generalization to 3D

Remark 4.2.4 (Geometric obstruction to Voronoi dual meshes).
Obtuse angle $\omega$:

- circumcenter $\notin$ triangle
- $\overline{C_i} \cap \overline{C_j} \neq \emptyset \implies$ nodes $i, j$ connected by edge
- geometric construction breaks down
- connectivity of unknowns hard to determine

**Angle condition** to ensure $\overline{C_i} \cap \overline{C_j} \neq \emptyset \iff$ nodes $i, j$ connected by edge of $\mathcal{M}$:

(i) sum of angles facing interior edge $\leq \pi$,

(ii) angles facing boundary edges $\leq \pi/2$ (for non-Dirichlet boundary conditions).

(i), (ii) characterize **Delaunay triangulations**
Popular choice: Barycentric dual mesh

Dual cells:
edges $\rightarrow$ union of lines connecting barycenters and midpoints of edges of $\mathcal{M}$
nodes $\rightarrow$ barycenters of triangles

No geometric obstructions

4.2.3 Relationship of finite elements and finite volume methods
Hardly surprising, finite volume methods and finite element Galerkin discretizations are closely related. This will be explored in this section for a model problem.

Setting:

- We consider the homogeneous Dirichlet problem for the Laplacian \( \Delta \)
  \[
  -\Delta u = f \quad \text{in } \Omega, \quad u = 0 \quad \text{on } \partial \Omega. \quad (4.2.5)
  \]

- Discretization by finite volume method based on a triangular mesh \( \mathcal{M} \) and on Voronoi dual cells

  \[\rightarrow \text{Fig. 127:}\]

  Assumption: \( \mathcal{M} = \text{Delaunay triangulation of } \Omega \Leftrightarrow \text{angle condition} \)

  Number of control volumes = number of interior nodes of \( \mathcal{M} \)

Still missing: specification of numerical flux function \( \Psi : \mathbb{R}^2 \mapsto \mathbb{R} \) for each dual edge
Idea: obtain **numerical flux** from Fourier’s law (2.5.3) applied to a (sufficiently smooth) $u_N : \Omega \mapsto \mathbb{R}$ *reconstructed* from dual cell values $\mu_i$.

Natural approach, since $\mu_i$ is read as approximation of $u(p_i)$, where the “center” $p_i$ of the dual cell $C_i$ coincides with a node $x_i \in \mathcal{V}(\mathcal{M})$ of the triangular mesh $\mathcal{M}$:

$$u_N = \mathbf{1}_1 \mu := \sum_{i=1}^N \mu_i b_i^N,$$  \hspace{1cm} (4.2.6)

where $N = \#\mathcal{V}(\mathcal{M}) =$ number of dual cells, size of vector $\mu$, $b_i^N \doteq$ nodal basis function (“tent function”) of $S^0_{1,0}(\mathcal{M})$ belonging to the node inside $C_i$.

$u_N \doteq$ piecewise linear interpolant of vertex values $\mu_i$

Note that $u_N$ is not smooth across inner edges of $\mathcal{M}$. However, we do not care when computing $j := \kappa(x) \nabla u_N$, because this flux is *only needed at edges of the dual mesh*, which lie inside triangles of $\mathcal{M}$ (with the exception of single points that are irrelevant for the flux integrals).
vertex values \( \mu_i \) on \( \mathcal{V}(\mathcal{M}) \)

p.w. linear interpolant \( u_N := \mathbb{P}_1 \mu \in S_{1,0}^0(\mathcal{M}) \)

Choice of numerical flux:
\[
J_{ik} := - \int_{\Gamma_{ik}} \text{grad} \, \mathbb{P}_1 \mu \cdot n_{ik} \, dS
\]  

(4.2.7)

\[
\sum_{k \in \mathcal{U}_i} \int_{\Gamma_{ik}} \text{grad} \, \mathbb{P}_1 \mu \cdot n_{ik} \, dS = \sum_{j \in \mathcal{U}_i} \mu_j \left( \sum_{k \in \mathcal{U}_i} \int_{\Gamma_{ik}} \text{grad} \, b^j_N \cdot n_{ik} \, dS \right) = \int_{\mathcal{C}_i} f(x) \, dx
\]

\( \Rightarrow \) \( (4.2.2) \) \( \Leftrightarrow \) one row of finite volume discretization matrix from

\[
= \text{matrix entry } (A)_{ij}
\]
\[ (A)_{ij} = \int_{\partial C_i} \nabla b^j_N \cdot n_i \, dS. \] (4.2.8)

\( n_i \) = exterior unit normal vector to \( \partial C_i \).

Part of the boundary of the control volume \( C_i \):

\[ \Gamma^K_i := \partial C_i \cap K. \]
Now, consider $i \neq j \leftrightarrow$ off-diagonal entries of $A$:

First, we recall that the intersection of the support of the “tent function” $b^j_N$ with $\partial C_i$ is located inside $K_1 \cup K_2$, see Fig. 133.

\[
(A)_{ij} = \int_{\Gamma_i} \nabla b^j_N \cdot n_i \, dS + \int_{\Gamma_i} \nabla b^j_N \cdot n_i \, dS.
\]

Next observe that $\nabla b^j_N$ is piecewise constant, which implies

\[
\text{div} \, \nabla b^j_N = 0 \quad \text{in } K_1, \quad \text{div} \, \nabla b^j_N = 0 \quad \text{in } K_2. \tag{4.2.9}
\]
Now apply Gauss’ theorem Thm. 2.4.5 to the domains $C_i \cap K_1$ and $C_i \cap K_2$ (shaded in figure).

Also use again that $\nabla b^j_N \equiv \text{const}$ on $K_1$ and $K_2$.

\begin{equation}
(A)_{ij} = \frac{1}{2} \int_{e_1} \nabla b^j_N|_{K_1} \cdot n_{e_1} \, dS + \frac{1}{2} \int_{e_{ij}} \nabla b^j_N|_{K_1} \cdot n_{e_{ij}}^1 \, dS \\
+ \frac{1}{2} \int_{e_{ij}} \nabla b^j_N|_{K_2} \cdot n_{e_{ij}}^2 \, dS + \frac{1}{2} \int_{e_2} \nabla b^j_N|_{K_1} \cdot n_{e_2} \, dS \ . \tag{4.2.10}
\end{equation}
On the other hand, an entry of finite element Galerkin matrix $\tilde{A}$ based on linear Lagrangian finite element space $S_1^0(\mathcal{M})$ can be computed as, see Sect. 3.2.5:

$$(\tilde{A})_{ij} = \int_{K_1} \text{grad} b_N^j \cdot \text{grad} b_N^i \, dx + \int_{K_2} \text{grad} b_N^j \cdot \text{grad} b_N^i \, dx.$$ 

Conduct local integration by parts using Green’s first formula from Thm. 2.4.7 and taking into account (4.2.9) and the linearity of the local shape functions:

$$(\tilde{A})_{ij} = \int_{\partial K_1} (\text{grad} b_N^j|_{K_1} \cdot n_1) b_N^i \, dS + \int_{\partial K_2} (\text{grad} b_N^j|_{K_2} \cdot n_2) b_N^i \, dS$$

$$= \frac{1}{2}|e_1| \text{grad} b_N^j|_{K_1} \cdot n_{e_1} + \frac{1}{2}|e_{ij}| \text{grad} b_N^j|_{K_1} \cdot n_{e_{ij}} +$$

$$\frac{1}{2}|e_2| \text{grad} b_N^j|_{K_2} \cdot n_{e_2} + \frac{1}{2}|e_{ij}| \text{grad} b_N^j|_{K_2} \cdot n_{e_{ij}}.$$ 

This is the same value as for $(A)_{ij}$ from (4.2.10)! Similar considerations apply to the diagonal entries $(A)_{ii}$ and $(\tilde{A})_{ii}$.

The finite volume discretization and the finite element Galerkin discretization spawn the same system matrix for the model problem (4.2.5).
Convergence and Accuracy

In this chapter we resume the discussion of Sect. 1.6 of accuracy of a Galerkin solution $u_N$ of a variational boundary value problem.

More precisely, we are going to study convergence, see Rem. 1.6.2

Focus: finite element Galerkin discretization of linear scalar 2nd-order elliptic boundary value problems in 2D, 3D

Prerequisites (what you should know by now):

- Boundary value problems (from equilibrium models, diffusion models): Sects. 2.4, 2.6
- Variational formulation: Sect. 2.8, see also (2.3.3), (2.8.15), (3.0.1)
- Some Sobolev spaces and their norms: Sect. 2.2
- Abstract Galerkin discretization: Sect. 3.1
- Lagrangian finite elements: Sects. 3.4, 3.2
5.1 Galerkin error estimates

Setting: \textit{linear variational problem} (1.4.6) in the form

\[ u \in V_0: \quad a(u, v) = \ell(v) \quad \forall v \in V_0, \quad (3.1.1) \]

- \( V_0 \) = (real) vector space, a space of functions \( \Omega \mapsto \mathbb{R} \) for scalar 2nd-order elliptic variational problems,
- \( a: V_0 \times V_0 \mapsto \mathbb{R} \) = a \textit{bilinear form}, see Def. 1.3.13,
- \( \ell: V_0 \mapsto \mathbb{R} \) = a \textit{linear form}, see Def. 1.3.13,

☞ We want (3.1.1) to be related to a \textit{quadratic minimization problem} (→ Def. 2.1.21):

\begin{center}
\textbf{Assumption 5.1.1.} The bilinear form \( a: V_0 \times V_0 \mapsto \mathbb{R} \) in (3.1.1) is symmetric and positive definite (→ Def. 2.1.25).
\end{center}
a supplies an inner product on $V_0$

a induces energy norm $\| \cdot \|_a$ on $V_0$ ($\rightarrow$ Def. 2.1.27)

We want (3.1.1) to be well posed, see Rem. 2.3.11

Assumption 5.1.2. The right hand side functional $\ell : V_0 \mapsto \mathbb{R}$ from (3.1.1) is continuous w.r.t. to the energy norm ($\rightarrow$ Def. 2.1.27) induced by $a$:

$$\exists C > 0 : |\ell(u)| \leq C \| u \|_a \quad \forall u \in V_0.$$  \hspace{1cm} (2.2.1)

An assumption to appease fastidious mathematicians:

Assumption 5.1.3. $V_0$ equipped with the energy norm $\| \cdot \|_a$ is a Hilbert space, that is, complete.

Theorem 5.1.4 (Existence and uniqueness of solution of linear variational problem).

Under Assumptions 5.1.1–5.1.3 the linear variational problem has a unique solution $u \in V_0$. 

This theorem is also known as the **Riesz representation theorem** for continuous linear functionals.

**Remark 5.1.5** (Well-posed 2nd-order linear elliptic variational problems).

For instance, Assumption 5.1.1 is satisfied for the bilinear form

\[ a(u, v) := \int_{\Omega} (\alpha(x) \nabla u) \cdot \nabla v \, dx, \quad u, v \in H^1_0(\Omega), \tag{5.1.6} \]

and uniformly positive definite (\(\rightarrow\) Def. 2.1.11) coefficient tensor \(\alpha : \Omega \mapsto \mathbb{R}^{d,d}\), see Sect. 2.1.3.

For the right hand side functional

\[ \ell(v) := \int_{\Omega} f(x)v(x) \, dx + \int_{\partial \Omega} h(x)v(x) \, dS, \quad v \in H^1(\Omega), \]

we found in Sect. 2.2, see (2.2.15), and Rem. 2.9.5 that \(f \in L^2(\Omega)\) and \(h \in L^2(\partial \Omega)\) ensures Assumption 5.1.2.

Assumption 5.1.3 for \(a\) from (5.1.6) is a deep result in the theory of Sobolev spaces [12, Sect. 5.2.3, Thm. 2].
Now consider Galerkin discretization of (3.1.1) based on Galerkin trial/test space \( V_{0,N} \subset V_0, \quad N := \dim V_{0,N} < \infty \):

\[
\begin{align*}
  u_N \in V_{0,N}: \quad & a(u_N, v_N) = \ell(v_N) \quad \forall v_N \in V_{0,N}. \\
\end{align*}
\]  

(3.1.4)

Thm. 3.1.5: existence and uniqueness of Galerkin solution \( u_N \in V_{0,N} \)

Goal: bound relevant norm of discretization error \( u - u_N \)

Here: relevant norm = energy norm \( \| \cdot \|_a \)

Why is the energy norm a “relevant norm”?

\( \geq \) Bounds of \( \| u - u_N \|_a \) provide bounds for the error in energy, see Rem. [1.6.7, (1.6.10)]

\[
\begin{align*}
  |J(u) - J(u_N)| = & \frac{1}{2} a(u, u) - a(u_N, u_N) = |\frac{1}{2} a(u + u_N, u - u_N)| \\
\text{(2.1.28)} & \leq \| u - u_N \|_a \cdot \| u + u_N \|_a.
\end{align*}
\]

(No doubt, energy is a key quantity for the solution of an equilibrium problem, which is defined as the minimizer of a potential energy functional.)
Other “relevant norms” were discussed in Sects. 1.6.1, 2.2:

- the mean square norm or $L^2(\Omega)$-norm, see Def. 2.2.5,
- the supremum norm or $L^\infty(\Omega)$-norm, see Def. 1.6.4.

The Galerkin approach allows a remarkably simple bound of the energy norm of the discretization error $u - u_N$:

\[
\begin{align*}
    a(u, v) &= \ell(v) \quad \forall v \in V_0, \quad V_{0,N} \subset V_0 \\
    a(u_N, v_N) &= \ell(v) \quad \forall v_N \in V_{0,N} \implies a(u - u_N, v_N) = 0 \quad \forall v_N \in V_{0,N}.
\end{align*}
\]

Galerkin orthogonality

\[
a(u - u_N, v_N) = 0 \quad \forall v_N \in V_{0,N}. \quad (5.1.7)
\]

[Geometric meaning for inner product $a(\cdot, \cdot) \rightarrow$]
Discretization error $e_N := u - u_N$ “$a(\cdot, \cdot)$-orthogonal” to discrete trial/test space $V_N$

Remark 5.1.8. If $a(\cdot, \cdot)$ is inner product on $V$: “Pythagoras’ theorem” → Fig. 136

\[
\|u - v_N\|_a^2 = \|u - u_N\|_a^2 - \|u_N - v_N\|_a^2.
\] (5.1.9)

\[
(5.1.9) \quad \iff \quad (v_N = 0) \text{ simple formula for computation of energy norm of Galerkin discretization error in numerical experiments with known } u.
\]

\[\Box\]

Theorem 5.1.10 (Cea’s lemma).

Under Assumptions 5.1.1–5.1.3 the energy norm of the Galerkin discretization error satisfies

\[
\|u - u_N\|_a = \inf_{v_N \in V_{0,N}} \|u - v_N\|_a.
\]
Proof. Use bilinearity of $a$ and Galerkin orthogonality (5.1.7): for any $v_N \in V_{0,N}$

$$\|u - u_N\|_a^2 = a(u - u_N, u - u_N) = a(u - v_N, u - u_N) + a(v_N - u_N, u - u_N).$$

Next, use the Cauchy-Schwarz inequality for the inner product $a$:

$$a(u, v) \leq \|u\|_a \|v\|_a \quad \forall u, v \in V_0.$$

$$\|u - u_N\|_a^2 \leq \|u - v_N\|_a \cdot \|u - u_N\|_a,$$

and cancel one factor $\|u - u_N\|_a$.

Optimality of Galerkin solutions:

$$\|u - u_N\|_a = \inf_{\substack{v_N \in V_{0,N} \cap V \subset \mathbb{R}^n}} \|u - v_N\|_a,$$

(5.1.11)

(norm of) discretization error best approximation error
To assess accuracy of Galerkin solution: study capability of $V_{0,N}$ to approximate $u$!

“Monotonicity” of best approximation: consider different trial/test spaces

\[ V_{0,N}, V'_{0,N} \subset V_0, \quad V_{0,N} \subset V'_{0,N} \Rightarrow \inf_{v_N \in V'_{0,N}} \| u - v_N \|_a \leq \inf_{v_N \in V_{0,N}} \| u - v_N \|_a. \]

Enhance accuracy by enlarging ("refining") trial space.

Now return to finite element Galerkin discretization of linear 2nd-order elliptic variational problems.

How to achieve refinement of FE space?

- **h-refinement**: replace $\mathcal{M}$ (underlying $V_{0,N}$) $\rightarrow$ $\mathcal{M}'$ (underlying larger discrete trial space $V'_{0,N'}$)
Example 5.1.12 (regular refinement of triangular mesh in 2D).

Regular refinement of triangle $K$ into four congruent triangles $T_1, T_2, T_3, T_4$.
\textbullet{} \textbf{p-refinement:} replace \( V_{0,N} := S_p^0(\mathcal{M}) \), \( p \in \mathbb{N} \) with \( V'_{0,N} := S_{p+1}^0(\mathcal{M}) \) \( \Rightarrow V_{0,N} \subset V'_{0,N} \)

The extreme case of \( p \)-refinement amounts to the use of \textit{global} polynomials on \( \Omega \) as trial and test functions \( \Rightarrow \) (polynomial) \textbf{spectral Galerkin method}, see Sect. 1.5.1.1.

Combination of h-refinement and p-refinement \( \Rightarrow \) \textbf{OF COURSE} (\textbf{hp-refinement}, [26])

\section*{5.2 Empirical Convergence of FEM}

\textbf{5.2 Empirical Convergence of FEM}
Recall from Sect. 1.6.2:

Crucial: convergence is an *asymptotic notion*!

sequence of discrete models $\Rightarrow$ sequence of approximate solutions $(u_N^{(i)})_{i \in \mathbb{N}}$

$\Rightarrow$ study sequence $(\|u_N^{(i)} - u\|)_{i \in \mathbb{N}}$

created by *variation* of a *discretization parameter*:

In this section some numerical experiments will demonstrate

- meaningful notions of “discretization parameters”,
- qualitative behaviors of the sequence $(\|u_N^{(i)} - u\|)_{i \in \mathbb{N}}$ we may expect,

for Lagrangian finite element discretization of linear scalar 2nd-order elliptic variational problems ($\rightarrow$ Sect. 2.8).

Sequences of discrete models will be generated by either *$h$-refinement* or *$p$-refinement*. 

Model problem: Dirichlet problem for Poisson equation:

$$-\Delta u = f \in L^2(\Omega) \quad \text{in} \quad \Omega, \quad u = g \in C^0(\partial \Omega) \quad \text{on} \quad \partial \Omega. \quad (5.2.1)$$

**Example 5.2.2** (Convergence of linear and quadratic Lagrangian finite elements in energy norm).

Setting: $\Omega = [0, 1]^2$, $f(x_1, x_2) = 2\pi^2 \sin(\pi x_1) \sin(\pi x_2)$, $\mathbf{x} \in \Omega$, $g = 0$

- Smooth solution $u(x, y) = \sin(\pi x) \sin(\pi y)$.

- Galerkin finite element discretization based on triangular meshes and
  - linear Lagrangian finite elements, $V_{0,N} = S^{0}_{1,0}(\mathcal{M}) \subset H^1_0(\Omega)$ (→ Sect. 3.2),
  - quadratic Lagrangian finite elements, $V_{0,N} = S^{0}_{2,0}(\mathcal{M}) \subset H^1_0(\Omega)$ (→ Ex. 3.4.2),
- quadrature rule (3.5.38) for assembly of local load vectors (→ Sect. 3.5.4),

Monitored: $H^1(\Omega)$-semi-norm (→ Def. 2.2.10) of the Galerkin discretization error $u - u_N$

Approximate (*) computation of $|u - u_N|_{H^1(\Omega)}$ on a sequence of meshes (created by successive regular refinement (→ Ex. 5.1.12) of coarse initial mesh)

(*) : use of local quadrature rule (3.5.38) (on current FE mesh)
Focus on asymptotics entails studying a

norm of the discretization error as function of a (real, cardinal) discretization parameter.
The discretization parameter must be linked to the resolution ("capability to approximate generic solution") of the Galerkin trial/test space \( V_{0,N} \). Possible choices are

- \( N := \dim V_{0,N} \) as a measure of the "cost" of a discretization, see Sect. 1.6.2,
- the maximum "size" of mesh cells, expressed by the mesh width \( h_M \) (→ Def. 5.2.3), see below.

**Definition 5.2.3 (Mesh width).**

Given a mesh \( M = \{K\} \), its mesh width \( h_M \) is defined as

\[
h_M := \max\{\diam K : K \in M\} , \quad \diam K := \max\{|p - q| : p, q \in K\} .
\]

This generalizes the concept of "mesh width" introduced in Sect. 1.5.1.2.
Recall type of convergence (algebraic convergence vs. exponential convergence) from Def. 1.6.20 and how to detect them in a numerical experiment by inspecting appropriate graphs, see Rem. 1.6.22.

Observations:
- **Algebraic rates of convergence** in terms of $N$ and $h$
- Quadratic Lagrangian FE converge with double the rate of linear Lagrangian FE

$H^1(\Omega)$-semi-norm of discretization error on unit square ($-\leftrightarrow p = 1$, $-\leftrightarrow p = 2$)
Example 5.2.4 (Convergence of linear and quadratic Lagrangian finite elements in $L^2$-norm).

Setting as above in Ex. 5.2.2, $\Omega = [0, 1]^2$.

Monitored: asymptotics of the $L^2(\Omega)$-semi-norm of the Galerkin discretization error (approximate computation of $\| u - u_N \|_{L^2(\Omega)}$ by means of local quadrature rule (3.5.38) on a sequence of meshes created by successive regular refinement (→ Ex. 5.1.12) of coarse initial mesh).
Observations:

- Linear Lagrangian FE ($p = 1$) \[ \| u - u_N \|_0 = O(h_M^2) = O(N^{-1}) \]
- Quadratic Lagrangian FE ($p = 2$) \[ \| u - u_N \|_0 = O(h_M^3) = O(N^{-1.5}) \]

For the “conversion” of convergence rates with respect to the mesh width $h_M$ and $N := \text{dim } S^0_p(M)$, note that in 2D for Lagrangian finite element spaces with fixed polynomial degree (→ Sect. 3.4) and meshes created by global (that is, carried out everywhere) regular refinement

\[ N = O(h^{-2}_M). \]
**Example 5.2.6** ($h$-convergence of Lagrangian FEM on L-shaped domain).

Setting: Model problem (5.2.1) on $\Omega = \left(-1, 1\right)^2 \setminus \left(0, 1\times] \setminus -1, 0\right]$, exact solution (in polar coordinates)

$$u(r, \varphi) = r^{2/3} \sin\left(\frac{2}{3} \varphi\right), \quad f = 0, \ g = u|_{\partial \Omega}.$$
Note: $\mathbf{grad} \ u$ has a singularity at $0$, that is, \[ \|\mathbf{grad} \ u(0)\| = \infty. \]
– linear Lagrangian finite elements, \( V_{0,N} = S_{1,0}^0(\mathcal{M}) \subset H_0^1(\Omega) \) (\( \rightarrow \) Sect. 3.2),
– quadratic Lagrangian finite elements, \( V_{0,N} = S_{2,0}^0(\mathcal{M}) \subset H_0^1(\Omega) \) (\( \rightarrow \) Ex. 3.4.2),

- linear/quadratic interpolation of Dirichlet data to obtain offset function \( u_0 \in S_{p,0}^0(\mathcal{M}) \), \( p = 1, 2 \), see Sect. 3.5.5, Ex. 3.5.43.

Sequence of meshes created by successive regular refinement (\( \Rightarrow \) Ex. 5.1.12) of coarse initial mesh, see Fig. 141.
Unstructured triangular meshes of $\Omega = ]-1, 1[^2 \setminus [0, 1] \times ]-1, 0[$ (two coarsest specimens)

Approximate computation of $|u - u_N|_{H^1(\Omega)}$ by using local quadrature formula (3.5.38) on FE meshes.
$H^1(\Omega)$-semi-norm of discretization error on “L-shaped” domain ($p \leftrightarrow p = 1$, $- \leftrightarrow p = 2$)

Observations:

- For both $p = 1, 2$: $\|u - u_N\|_1 = O(N^{-1/3})$
- No gain from higher polynomial degree

Conjecture: singularity of $\text{grad } u$ at $x = 0$ seems to foil faster algebraic convergence of quadratic Lagrangian finite element solutions!
Example 5.2.7 (Convergence of Lagrangian FEM for \( p \)-refinement).

Model BVP as in Ex. 5.2.2 \((\Omega = [0, 1]^2)\) and Ex. 5.2.6 (L-shaped domain \( \Omega = [-1, 1]^2 \setminus (0, 1] \times [-1, 0] \)).

Galerkin finite element discretization based on \( S_p^0(M) \), \( p = 1, 2, 3, 5, 6, 7, 8, 9, 10 \), built on a fixed coarse triangular mesh of \( \Omega \).

\( p \)-refinement

Monitored: \( H^1(\Omega) \)-semi-norm (energy norm) and \( L^2(\Omega) \)-norm of discretization error as functions of polynomial degree \( p \) and \( N := \dim S_p^0(M) \).

(Computation of norms by means of local quadrature rule of order 19!. This renders the error in norm computations introduced by numerical quadrature negligible.)

Meaningful discretization parameters for asymptotic study of error norms:
polynomial degree $p$ for Lagrangian finite element space,

$N := \dim V_{0,N}$ as a measure of the “cost” of a discretization, see Sect. 1.6.2.

$\Omega = ]0, 1[^2$: behavior of $|u - u_N|_{H^1(\Omega)}$ for different polynomial degrees.

Lagrangian FEM: $p$-convergence for smooth (analytic) solution
Observation: **exponential convergence** of FE discretization error, *cf.* the behavior of the discretization error of spectral collocation and polynomial spectral Galerkin methods in 1D, Ex. 1.6.19.

Lagrangian FEM: *p*-convergence for solution with singular gradient

**Observation:** Only **algebraic convergence** of FE discretization error!

**The suspect:** “singular behavior” of $\text{grad } u$ at $\mathbf{x} = 0$. 

---

**Numerical Methods for PDEs**

R. Hiptmair
C. Schwab,
H. Harbrecht,
V. Gradinaru,
A. Chernov
SAM, ETHZ

p. 482
5.3 A priori finite element error estimates

We are interested in a priori estimates of norms of the discretization error.

A priori estimate: bounds for error norms available before computing approximate solutions.

A posteriori estimate: bounds for error norms based on an approximate solution already computed.

Results of Sect. 5.1 provide handle on a priori estimate for Galerkin discretization error:

Optimality (5.1.11) of Galerkin solution a priori error estimates
Thm. 5.1.10 ➤

Estimate energy norm of Galerkin discretization error $u - u_N$ by bounding best approximation error for exact solution $u$ in finite element space:

$$
\|u - u_N\|_a \leq \inf_{v_N \in V_{0,N}} \|u - v_N\|_a,
$$

(5.1.11)

(norm of) discretization error

(best approximation error)

How to estimate best approximation error $\inf_{v_N \in V_{0,N}} \|u - v_N\|_V$?

➤ Well, given solution $u$ seek candidate function $w_N \in V_{0,N}$ with

$$
\|u - w_N\|_V \approx \inf_{v_N \in V_N} \|u - v_N\|_V.
$$

Natural choice: $w_N$ by interpolation/averaging of (unknown, but existing) $u$
5.3.1 Estimates for linear interpolation in 1D

Computational domain (→ Sect. 1.4): interval $\Omega = [a, b]$

Given: mesh of $\Omega$ (→ Sect. 1.5.1.2): $\mathcal{M} := \{x_{j-1}, x_j: j = 1, \ldots, M\}, M \in \mathbb{N}$

![Fig. 146](image.png)

Piecewise linear interpolant of $u \in C^0([a, b])$

$$I_1 u \in S^0_1(\mathcal{M}),$$

$$(I_1 u)(x_j) = u(x_j), \quad j = 0, \ldots, M.$$  \hspace{1cm} (5.3.1)

Goal: Bound suitable norm (→ Sect. 1.6.1) of interpolation error $u - I_1 u$

in terms of geometric quantities $\ast$ characterizing $\mathcal{M}$.

$\ast$: A typical such quantity is the **mesh width** $h_\mathcal{M} := \max_j |x_j - x_{j-1}|$

Now we investigate different norms of the interpolation error.
\[ \| u - l_1 u \|_{L^\infty([a,b])} \], see [18, Sect. 9.1] and [18, Sect. 9.4.1]: from [18, Thm. 9.1.7] for \( n = 1 \): for 
\( u \in C^2([a, b]) \)

\[
\max_{x_{j-1} \leq x \leq x_j} u(t) - (l u)(x) = \frac{1}{4} u''(\xi_t)(x_j - x_{j-1})^2, \quad \text{for some } \xi_t \in [x_{j-1}, x_j], \tag{5.3.3}
\]

with local linear interpolant

\[
(l u)(x) = \frac{x - x_{j-1}}{x_j - x_{j-1}} u(x_j) - \frac{x_j - x}{x_j - x_{j-1}} u(x_{j-1}). \tag{5.3.4}
\]

**interpolation error estimate in \( L^\infty([a, b]) \)**

\[
\| u - l_1 u \|_{L^\infty([a,b])} \leq \frac{1}{4} h^2 M \| u'' \|_{L^\infty([a,b])}. \tag{5.3.5}
\]

This is obtained by simply taking the maximum over all *local* norms of the interpolation error.

Recall: supremum norm (maximum norm) from Def. 1.6.4

Now, we also want to study other norms of the interpolation error:

\[
\| u - l_1 u \|_{L^2([a,b])}:
\]
Now all mesh cells contribute to this norm:

\[
\|u - l_1u\|_{L^2([a,b])}^2 = \sum_{j=1}^M \|u - l_1u\|_{L^2([x_{j-1},x_j])}^2 = \sum_{j=1}^M \int_{x_{j-1}}^{x_j} |(u - lu)(x)|^2 \, dx , \quad l_1u \text{ from (5.3.4)} .
\]

> Idea:

*localization*

(Estimate error on individual mesh cells and sum local bounds)

By integrating by parts (1.3.22) twice, for \( u \in C^2([x_{j-1}, x_j]), x \in [x_{j-1}, x_j], \)

\[
\begin{align*}
\int_{x_{j-1}}^{x} \frac{(x_j - x)(\xi - x_{j-1})}{x_j - x_{j-1}} u''(\xi) \, d\xi + \int_{x}^{x_j} \frac{(x - x_{j-1})(x_j - \xi)}{x_j - x_{j-1}} u''(\xi) \, d\xi \\
= \frac{x_j - x}{x_j - x_{j-1}} u(x_{j-1}) + \frac{x - x_{j-1}}{x_j - x_{j-1}} u(x_j) - u(x) .
\end{align*}
\]

(5.3.7)

This is a *representation formula* for the local interpolation error \( |u - u| \) of the form

\[
(|u - u)(x) = \int_{x_{j-1}}^{x_j} G(x, \xi) u''(\xi) \, d\xi .
\]
with \( G(x, \xi) = \begin{cases} 
\frac{(x_j - x)(\xi - x_{j-1})}{x_j - x_{j-1}} & \text{for } x_{j-1} \leq \xi < x, \\
\frac{x_j - x_{j-1}}{(x - x_{j-1})(x_j - \xi)} & \text{for } x \leq \xi \leq x_j. 
\end{cases} \)

which satisfies

\[
|G(x, \xi)| \leq |x_j - x_{j-1}| \quad \Rightarrow \quad \int_{x_{j-1}}^{x_j} G(x, \xi)^2 \, d\xi \leq |x_j - x_{j-1}|^3.
\]

Kernel functions \( G \) for 1D linear interpolation for \( x_{j-1} = 0, \ x_j = 1 \).
\begin{align*}
\int_{x_{j-1}}^{x_j} |u(x) - l(u(x))|^2 \, dx &= \int_{x_{j-1}}^{x_j} \left| \int_{x_{j-1}}^{x_j} G(x, \xi) u''(\xi) \, d\xi \right|^2 \, dx \\
&\leq \int_{x_{j-1}}^{x_j} \left\{ \int_{x_{j-1}}^{x_j} G(x, \xi)^2 \, d\xi \cdot \int_{x_{j-1}}^{x_j} |u''(\xi)|^2 \, d\xi \right\} \, dx ,
\end{align*}

\text{(5.3.8)}

\text{(2.2.15)}
\[
\| u - i u \|^2_{L^2([x_{j-1}, x_j])} = \int_{x_{j-1}}^{x_j} |u(x) - i u(x)|^2 \, dx \leq |x_j - x_{j-1}|^4 \int_{x_{j-1}}^{x_j} |u''(\xi)|^2 \, d\xi .
\]

(5.3.8)

(5.3.9)

Apply this estimate on \([x_{j-1}, x_j]\), sum over all cells of the mesh \(\mathcal{M}\) and take square root.

\[
\| u - i u \|^2_{L^2([a,b])} \leq h^2_M \| u'' \|^2_{L^2([a,b])} .
\]

(5.3.10)

\[
|u - i u|_{H^1([a,b])}.
\]

Differentiate representation formula (5.3.7): for \(x_{j-1} < x < x_j\)

\[
\frac{d}{dx} (i u - u)(x) = \int_{x_{j-1}}^{x_j} -\frac{\xi - x_{j-1}}{x_j - x_{j-1}} u''(\xi) \, d\xi + \int_{x_{j-1}}^{x_j} \frac{x_j - \xi}{x_j - x_{j-1}} u''(\xi) \, d\xi .
\]
\[ \int_{x_{j-1}}^{x_j} \left| \frac{d}{dx} (u - I_1 u)(x) \right|^2 \, dx = \int_{x_{j-1}}^{x_j} \left\| \frac{\partial G}{\partial x}(x, \xi) u''(\xi) \right\|^2 \, d\xi \, dx \]

\[ \leq \int_{x_{j-1}}^{x_j} \left\{ \int_{x_{j-1}}^{x_j} \left| \frac{\partial G}{\partial x}(x, \xi) \right|^2 \, d\xi \cdot \int_{x_{j-1}}^{x_j} |u''(\xi)|^2 \, d\xi \right\} \, dx . \]

\[ |u - I_1 u|^2_{H^1([x_{j-1}, x_j])} \leq (x_j - x_{j-1})^2 \int_{x_{j-1}}^{x_j} |u''(\xi)|^2 \, d\xi . \quad (5.3.11) \]

As above, apply this estimate on \([x_{j-1}, x_j]\), sum over all cells of the mesh \(\mathcal{M}\) and take square root.

\[ (5.3.11) \Rightarrow |u - I_1 u|_{H^1([a,b])} \leq h \mathcal{M} \left\| u'' \right\|_{L^2([a,b])} . \quad (5.3.12) \]
What we learn from this example:

1. We have to rely on smoothness of the interpoland $u$ to obtain bounds for norms of the interpolation error.

2. The bounds involve norms of derivatives of the interpoland.

3. For smooth $u$ we find algebraic convergence (→ Def. 1.6.20) of norms of the interpolation error in terms of mesh width $h_M \to 0$. 
5.3.2 Error estimates for linear interpolation in 2D

Given:

- polygonal domain \( \Omega \subset \mathbb{R}^2 \)
- triangular mesh \( M \) of \( \Omega \) (\( \rightarrow \) Def. 3.3.1)

Sect 5.3.1 introduced piecewise linear interpolation on a mesh/grid in 1D. The next definition gives the natural 2D counterpart on a triangular mesh, which is closely related to the piecewise linear reconstruction (interpolation) operator from (4.2.6), see Figs. 131, 132.
**Definition 5.3.13** (Linear interpolation in 2D).

The linear interpolation operator \( I_1 : C^0(\overline{\Omega}) \mapsto S_1^0(\mathcal{M}) \) is defined by

\[
I_1 u \in S_1^0(\mathcal{M}), \quad I_1 u(p) = u(p) \quad \forall p \in \mathcal{V}(\mathcal{M}).
\]

Recalling the definition of the nodal basis \( \mathcal{B} = \{ b^p_N : p \in \mathcal{V}(\mathcal{M}) \} \) of \( S_1^0(\mathcal{M}) \) from (3.2.3), where \( b^p_N \) is the “tent function” associated with node \( p \), an equivalent definition is, cf. (3.5.44),

\[
I_1 u = \sum_{p \in \mathcal{V}(\mathcal{M})} u(p) b^p_N, \quad u \in C^0(\overline{\Omega}).
\]

**Task:** For “sufficiently smooth” \( u : \Omega \mapsto \mathbb{R} \) (\( \leftrightarrow u \in C^\infty(\overline{\Omega}) \) to begin with) estimate the interpolation error norm \( \| u - I_1 u \|_{H^1(\Omega)} \).

**Idea:**

- **Localization**
  - \( I_1 \) local: first, estimate \( \| u - I_1 u \|^2_{H^1(K)}, \ K \in \mathcal{M}, \)
  - then, global estimate via summation as in Sect. 5.3.1.
- Focus on single triangle \( K \in \mathcal{M} \).
Crucial for localization to work: linear interpolation operator \( I_1 : C^0(\bar{\Omega}) \to S_1^0(\mathcal{M}) \) can be defined purely locally by

\[
l_1u|_K = u(a^1)_1 + u(a^2)_2 + u(a^3)_3 ,
\]

(5.3.15)

for each triangle \( K \in \mathcal{M} \) with vertices \( a^1, a^2, a^3 \) (\( \lambda_k \equiv \) barycentric coordinate functions = local shape functions for \( S_1^0(\mathcal{M}) \)), see Fig. 72).

Next step, cf. (5.3.7): representation formula for local interpolation error.

\[
u \in C^2(\bar{K}): \text{ by mean value formula } \forall x \in K,\]

\[
u(a^j) = \nu(x) + \text{grad} \nu(x) \cdot (a^j - x) + \int_{0}^{1} (a^j - x)^T D^2u(x + \xi (a^j - x))(a^j - x)(1 - \xi) \, d\xi ,
\]

(5.3.16)

\[
D^2u(x) = \begin{pmatrix}
\frac{\partial^2 u}{\partial x_1^2}(x) & \frac{\partial^2 u}{\partial x_1 \partial x_2}(x) \\
\frac{\partial^2 u}{\partial x_2 \partial x_1}(x) & \frac{\partial^2 u}{\partial x_2^2}(x)
\end{pmatrix} \equiv \text{Hessian}.
\]

The formula (5.3.16) is easily verified by applying integration by parts

\[
f(b) - f(a) = \left[ \xi f'(\xi) \right]_a^b - \int_a^b \xi f''(\xi) \, d\xi = f'(a)(b - a) + \int_a^b (b - \xi) f''(\xi) \, d\xi.
\]
to the function \( \phi(t) = u(ta^j + (1 - t)x) \) with \( a = 0, b = 1 \).

Next, use (5.3.16) to replace \( u(a^j) \) in the formula (5.3.15) for local linear interpolation. Also use the identities for the barycentric coordinate functions

\[
\sum_{j=1}^{3} \lambda_j(x) = 1 \quad , \quad x = \sum_{j=1}^{3} a^j \lambda_j(x) . \tag{5.3.17}
\]

\[
l_1u(x) = \sum_{j=1}^{3} u(a^j) \lambda_j(x) = u(x) \cdot \sum_{j=1}^{3} \lambda_j(x) + \text{grad } u(x) \cdot \sum_{j=1}^{3} (a^j - x) \lambda_j(x) + R(x) ,
\]

with \( R(x) := \sum_{j=1}^{3} \left( \int_{0}^{1} (a^j - x)^T D^2 u(x + \xi(a^j - x))(a^j - x)(1 - \xi) \, d\xi \right) \lambda_j(x) . \tag{5.3.18}\)

Again, as in the case of (5.3.7) for 1D linear interpolation we have arrived at an integral representation formula for the local interpolation error:

\[
(u - l_1u)(x) = \sum_{j=1}^{3} \left( \int_{0}^{1} (a^j - x)^T D^2 u(x + \xi(a^j - x))(a^j - x)(1 - \xi) \, d\xi \right) \lambda_j(x) . \tag{5.3.19}\]
Together with the triangle inequality, the trivial bound $|\lambda_j| \leq 1$ yields

$$
\|u - I_1 u\|_{L^2(K)} \leq \sum_{j=1}^{3} \left( \int_K \left( \int_0^1 (\alpha^j - x)^T D^2 u(x + \xi(\alpha^j - x)) (\alpha^j - x)(1 - \xi) \, d\xi \right)^2 \, dx \right)^{1/2}.
$$

To estimate an expression of the form

$$
\int_K \left( \int_0^1 (\alpha^j - x)^T D^2 u(x + \xi(\alpha^j - x)) (\alpha^j - x)(1 - \xi) \, d\xi \right)^2 \, dx,
$$

we may assume, without loss of generality, that $\alpha^j = 0$.

Task: estimate terms (where 0 is a vertex of $K$!)

$$
\int_K \left( \int_0^1 x^T D^2 u((1 - \xi)x) x (1 - \xi) \, d\xi \right)^2 \, dx = \int_K \left( \int_0^1 x^T D^2 u(\xi x)x \, \xi \, d\xi \right)^2 \, dx.
$$

Denote $\gamma \triangleq$ angle of $K$ at vertex 0,

$h \triangleq$ length of longest edge of $K$. 

$K$ is contained in the sector

\[ S := \{ \mathbf{x} = (r \cos \varphi, r \sin \varphi) : 0 \leq r < h, 0 \leq \varphi \leq \gamma \} \]

**Lemma 5.3.21.** For any $\psi \in L^2(S)$

\[
\frac{1}{S} \left( \int_{0}^{1} \left( \int_{0}^{r} |\mathbf{y}|^2 \psi(\tau \mathbf{y}) \tau \, d\tau \right)^2 \, d\mathbf{y} \right) \leq \frac{h^4}{8} \| \psi \|^2_{L^2(S)}.
\]

Using polar coordinates $(r, \varphi)$, $\hat{\mathbf{s}}_{\varphi} = (\cos \varphi, \sin \varphi)$, see [29, Bsp. 8.5.3], and Cauchy-Schwarz inequality (2.2.15):

\[
\frac{1}{S} \left( \int_{0}^{1} \left( \int_{0}^{r} |\mathbf{y}|^2 \psi(\tau \mathbf{y}) \tau \, d\tau \right)^2 \, d\mathbf{y} \right) = \frac{\gamma}{0} \int_{0}^{h} \left( \int_{0}^{1} r^2 \psi(\tau \hat{\mathbf{s}}_{\varphi}) \tau \, d\tau \right)^2 r \, dr \, d\varphi
\]

\[
= \frac{\gamma}{0} \int_{0}^{h} \left( \int_{0}^{r} \psi(\sigma \hat{\mathbf{s}}_{\varphi}) \sigma \, d\sigma \right)^2 r \, dr \, d\varphi \leq \frac{\gamma}{0} \int_{0}^{h} \int_{0}^{r} \psi^2(\sigma \hat{\mathbf{s}}_{\varphi}) \sigma \, d\sigma \cdot \int_{0}^{r} \sigma \, d\sigma \, r \, dr \, d\varphi
\]

R. Hiptmair
C. Schwab
H. Harbrecht
V. Gradinaru
A. Chernov
SAM, ETHZ

5.3
p. 498
\[
\begin{align*}
\leq \frac{1}{2} \int_0^h \int_0^h \psi^2(\sigma \mathcal{S}_\phi) \sigma \, d\sigma \, d\phi \cdot \int_0^h r^3 \, dr.
\end{align*}
\]

Use \(|z^\top A y| \leq \|A\|_F |z| |y|\), \(A \in \mathbb{R}^{n,n}\), \(z, y \in \mathbb{R}^n\), and then apply Lemma 5.3.21 with \(y := x - a^j\), \(\tau = 1 - \xi\).

\[
\|u - l_1 u\|_{L^2(K)}^2 \leq \frac{3}{8} h_K^4 \left\| D^2 u \right\|_{L^2(K)}^2, \tag{5.3.22}
\]

with Frobenius matrix norm \(\left\| D^2 u(x) \right\|_F^2 := \sum_{i,j=1}^2 \left| \frac{\partial^2 u}{\partial x_i \partial x_j}(x) \right|^2\)

size of triangle \(h_K := \text{diam } K := \max\{|p - q|: p, q \in K\}\)

Estimate for gradient: from (5.3.16) we infer the local integral representation formula, which can also be obtained by taking the gradient of (5.3.19).

\[
\text{grad} l_1 u(x) = u(x) \sum_{j=1}^3 \text{grad} \lambda_j(x) + \sum_{j=1}^3 (a^j - x)^\top \text{grad} \lambda_j(x) \cdot \text{grad} u(x) + G(x),
\]

with \(G(x) := \sum_{j=1}^3 \left( \int_0^1 (a^j - x)^\top D^2 u(x + \xi(a^j - x))(a^j - x)(1 - \xi) \, d\xi \right) \text{grad} \lambda_j(x)\).
Note that $\text{grad} \sum_{j=1}^{3} \lambda_j(x) = \text{grad} 1 = 0$ and

$$\sum_{j=1}^{3} \text{grad} \lambda_j(x)(a^j - x)^\top = \sum_{j=1}^{3} \text{grad} \lambda_j(x)(a^j)^\top = \text{grad}(\sum_{j=1}^{3} \lambda_j(x)a^j) = \text{grad} x = I.$$  

(3.5.22) \quad \Rightarrow \quad \boxed{\left| \text{grad} \lambda_j(x) \right| \leq \frac{h_K}{2|K|}, \quad x \in K.} \quad (5.3.23)

$$\|\text{grad}(u - l_1 u)\|_{L^2(K)}^2 \leq \frac{h_K^2}{4|K|^2} \|R\|_{L^2(K)}^2 \leq \frac{3}{8} \frac{h_K^6}{4|K|^2} \left\|D^2 u\right\|_{F}^2 \leq L^2(K). \quad (5.3.24)$$

Summary of *local* interpolation error estimates for linear interpolation according to Def. 5.3.13:
Lemma 5.3.25 (Local interpolation error estimates for 2D linear interpolation).

For any triangle $K$ and $u \in C^2(K)$ the following holds

$$
\| u - l_1 u \|_{L^2(K)}^2 \leq \frac{3}{8} h_K^4 \| D^2 u \|_F^2,
$$
(5.3.22)

$$
\| \text{grad}(u - l_1 u) \|_{L^2(K)}^2 \leq \frac{3}{24} \frac{h_K^6}{|K|^2} \| D^2 u \|_F^2.
$$
(5.3.24)

New aspect compared to Sect. 5.3.1: shape of $K$ enters error bounds of Lemma 5.3.25.

We aim to extract this shape dependence from the bounds.
Definition 5.3.26 (Shape regularity measures).

For a simplex $K \in \mathbb{R}^d$ we define its shape regularity measure as the ratio

$$\rho_K := h_K^d : |K|,$$

and the shape regularity measure of a simplicial mesh $\mathcal{M} = \{K\}$

$$\rho_{\mathcal{M}} := \max_{K \in \mathcal{M}} \rho_K.$$

Important: The shape regularity measure $\rho_K$ is an invariant of a similarity class of triangles.

(≡ if a triangle is transformed by scaling, rotation, and translation, the shape regularity measure does not change)

Sloppily speaking, $\rho_K$ depends only on the shape, not the size of $K$.

For triangle $K$: $\rho_K$ large $\iff$ $K$ “distorted” $\iff$ $K$ has small angles.
The shape regularity measure $\rho_M$ is often used to gauge the *quality* of meshes produced by mesh generators.

Final step: we add up the local estimates from Lemma 5.3.25 over all triangles of the mesh and take the square root.
Theorem 5.3.27 (Error estimate for piecewise linear interpolation).

For any $u \in C^2(\bar{\Omega})$

$$\| u - I_1 u \|_{L^2(\Omega)} \leq \sqrt{\frac{3}{8}} h^2_M \| D^2 u \|_{L^2(\Omega)} ,$$

$$\| \text{grad}(u - I_1 u) \|_{L^2(\Omega)} \leq \sqrt{\frac{3}{24}} \rho M h M \| D^2 u \|_{L^2(\Omega)} .$$

Remark 5.3.28 (Energy norm and $H^1(\Omega)$-norm).

Objection! Well, Cea’s lemma Thm. 5.1.10 refers to the energy norm, but Thm. 5.3.27 provides estimates in $H^1(\Omega)$-norm only!

For uniformly positive definite (→ Def. 2.1.11) and bounded coefficient tensor $\alpha : \Omega \mapsto \mathbb{R}^{d,d}$, cf. (2.1.8),

$$\exists 0 < \alpha^- < \alpha^+ : \alpha^- \| z \|^2 \leq z^T \alpha(x) z \leq \alpha^+ \| z \|^2 \quad \forall z \in \mathbb{R}^d , x \in \Omega ,$$

and the energy norm (→ Def. 2.1.27) induced by

$$a(u, v) := \int_{\Omega} (\alpha(x) \text{grad} u) \cdot \text{grad} v \, dx , \quad u, v \in H^1_0(\Omega) , \quad (5.1.6)$$
we immediately find the **equivalence** (\( \equiv \) two-sided uniform estimate)

\[
\sqrt{\alpha^-} |v|_{H^1(\Omega)} \leq \|v\|_a \leq \sqrt{\alpha^+} |v|_{H^1(\Omega)}.
\]

(5.3.29)

Thus, interpolation error estimates in \( |\cdot|_{H^1(\Omega)} \) immediately translate into estimates in terms of the energy norm.

### 5.3.3 The Sobolev scales

Bounds in Thm. 5.3.27 involve \( \|D^2 u\|_{F^2(\Omega)} \) measures smoothness of \( u \)

 normals of this type are a tool to measure the **smoothness** of functions (that usually are solutions of elliptic BVP):
**Definition 5.3.30** (Higher order Sobolev spaces/norms).

The $m$-th order Sobolev norm, $m \in \mathbb{N}_0$, for $u : \Omega \subset \mathbb{R}^d \mapsto \mathbb{R}$ (sufficiently smooth) is defined by

$$
\|u\|^2_{H^m(\Omega)} := \sum_{k=0}^{m} \sum_{\alpha \in \mathbb{N}^d, |\alpha| = k} \int_{\Omega} |D^\alpha u|^2 \, dx,
$$

where $D^\alpha u := \frac{\partial |\alpha| u}{\partial x_1^{\alpha_1} \cdots \partial x_d^{\alpha_d}}$.

**Sobolev space** $H^m(\Omega) := \{v : \Omega \mapsto \mathbb{R} : \|v\|_{H^m(\Omega)} < \infty\}$.

Recall: multiindex notation (3.3.4), (3.3.5)

Gripe (→ Sect. 2.2): Don’t bother me with these Sobolev spaces!

Response: Well, these concepts are pervasive in the numerical analysis literature and you have to be familiar with them.
Reassuring: Again, it is only the norms $\|u\|_{H^m(\Omega)}$ that matter for us!

Now, we have come across an additional purpose of Sobolev spaces and their norms:

provide framework for variational formulation of elliptic BVP

($\rightarrow$ Sect. 2.2)

Sobolev spaces

provide norms $\|\cdot\|_{H^m(\Omega)}$ that measure smoothness of functions

Sobolev scale:

$$\ldots \subset H^3(\Omega) \subset H^2(\Omega) \subset H^1(\Omega) \subset L^2(\Omega)$$

Observation: bounds in Thm. 5.3.27 = “principal parts” of Sobolev norms, that is, the parts containing the highest partial derivatives.
**Definition 5.3.31** (Higher order Sobolev semi-norms).

The \( m \)-th order Sobolev semi-norm, \( m \in \mathbb{N} \), for sufficiently smooth \( u : \Omega \rightarrow \mathbb{R} \) is defined by

\[
|u|_{H^m(\Omega)}^2 := \sum_{\alpha \in \mathbb{N}^d, |\alpha| = m} \int_{\Omega} |D^\alpha u|^2 \, dx.
\]

Elementary observation: \( |p|_{H^m(\Omega)} = 0 \iff p \in \mathcal{P}_{m-1}(\mathbb{R}^d) \)

By density arguments we can rewrite the interpolation error estimates of Thm. 5.3.27 in terms of Sobolev semi-norms:

**Corollary 5.3.32** (Error estimate for piecewise linear interpolation in 2D).

Under the assumptions/with notations of Thm. 5.3.27

\[
\|u - I_1 u\|_{L^2(\Omega)} \leq \sqrt{\frac{3}{8}} h^2_M |u|_{H^2(\Omega)}, \quad \forall u \in H^2(\Omega).
\]

\[
|u - I_1 u|_{H^1(\Omega)} \leq \sqrt{\frac{3}{24}} \rho_M h_M^2 |u|_{H^2(\Omega)},
\]
Remark 5.3.33 (Continuity of interpolation operators).

Apply \( \Delta \)-inequality to estimates of Cor. 5.3.32:

\[
\| I_1 u \|_{L^2(\Omega)} \leq \| u \|_{L^2(\Omega)} + \sqrt{\frac{3}{8}} h_M^2 \| u \|_{H^2(\Omega)} \leq 2 \| u \|_{H^2(\Omega)},
\]

(5.3.34)

if lengths are scaled such that \( h_M \leq 1 \). Estimate (5.3.34) means that \( I_1 : H^2(\Omega) \mapsto L^2(\Omega) \) is a continuous linear mapping.

The same conclusion could have been drawn from the following fundamental result:

\[m > \frac{d}{2} \Rightarrow H^m(\Omega) \subset C^0(\overline{\Omega}) \land \exists C = C(\Omega) > 0: \| u \|_{\infty} \leq C \| u \|_{H^m(\Omega)} \forall u \in H^m(\Omega).\]

\[\text{Theorem 5.3.35 (Sobolev embedding theorem).}\]

On the other hand \( I_1 : H^1(\Omega) \mapsto L^2(\Omega) \) is not continuous, as we learn from Rem. 2.3.16.
5.3.4 Anisotropic interpolation error estimates

Triangular cells with “bad shape regularity” ($\rho_K$ “large”): very small/large angles:

![Diagram of triangles with bad shape regularity]

The estimates of Lemma 5.3.25 might suggest that we face huge local interpolation errors, once $\rho_K$ becomes large.

**Issue:** are the estimates of Lemma 5.3.25 sharp?

We will try to find this out experimentally by computing the best possible constants in the estimates

$$
\|u - I_1 u\|_{L^2(K)} \leq C_{K,2} h_K^2 \|u\|_{H^2(K)}, \quad \|u - I_1 u\|_{H^1(K)} \leq C_K h_K \|u\|_{H^2(K)}.
$$

**Note:** Merely translating, rotating, or scaling $K$ does not affect the constants $C_{K,2}$ and $C_K$. Therefore, we can restrict ourselves to “canonical triangles”. Every general triangle can be mapped to one of these by translating, rotating, and scaling.
\[ C_{K,2} := \sup_{u \in H^2(K) \setminus \{0\}} \frac{\|u - I_1 u\|_{L^2(K)}}{\|u\|_{H^2(K)}}, \quad C_K := \sup_{u \in H^2(k) \setminus \{0\}} \frac{\|u - I_1 u\|_{H^1(K)}}{\|u\|_{H^2(K)}}, \]

on triangle \( K := \text{convex} \left\{ \left( \frac{0}{0} \right), \left( \frac{1}{0} \right), \left( \frac{p_x}{p_y} \right) \right\} \).

Sampling the space of “canonical” triangles (modulo similarity)

\[ 0 \leq p_x, p_y \leq 1. \]

+ Numerical computation of \( C_K, C_{K,2} \)

implementation by A. Inci (spectral polynomial Galerkin method)
triangle $K := \text{convex } \{(0,0),(1,0),(\frac{1}{2},h)\}, \ h > 0,$

$u(x,y) = x(1-x), \ 0 < x < 1.$

The interpolant becomes steeper and steeper as $h \to 0$:

\[ \|u\|_{H^2(K)}^2 = \frac{3031}{1440} h, \quad \|u - I_1 u\|_{H^1(K)}^2 = \frac{29}{2880} h + \frac{1}{12} h + \frac{1}{32} h^{-1}, \quad \|u - I_1 u\|_{L^2(K)}^2 = \frac{29}{2889} h \]
\[
\frac{\| u - l_1u \|^2_{H^1(K)}}{\| u \|^2_{H^2(K)}} \geq \frac{269}{6062} + \frac{45}{3031} h^{-2}, \quad \frac{\| u - l_1u \|^2_{L^2(K)}}{\| u \|^2_{H^2(K)}} = \frac{29}{6062}.
\]

**Example 5.3.36** (Good accuracy on “bad” meshes).

\(\Omega = ]0, 1[^2, \ u(x_1, x_2) = \sin(\pi x_1) \sin(\pi x_2), \ BVP -\Delta u = f, \ u|_{\partial \Omega} = 0,\) finite element Galerkin discretization on triangular meshes, \(V_N = S_{1,0}^0(M)\).

- meshes created by random distortion of tensor product grids
2D triangular mesh

# Vertices : 41,     # Elements : 64,     # Edges : 56

Fig. 157

2D triangular mesh

# Vertices : 145,     # Elements : 256,     # Edges : 203

Fig. 158
Monitored: for different mesh resolutions, $H^1(\Omega)$-seminorm of discretization error as function of smallest/largest angle in the mesh.

Observation: Accuracy does not suffer much from distorted elements!
Example 5.3.37 (Gap between interpolation error and best approximation error).

Ex. 5.3.36 raises doubts whether the interpolation error can be trusted to provide good, that is, reasonably sharp bounds for the best approximation error.

In this example we will see that

\[ \inf_{v_N \in S_p^0(\mathcal{M})} \| u - v_N \|_1 \ll \| u - I_p u \|_{H^1(\Omega)} \] is possible!

Elementary cell of “bad mesh” \( \mathcal{M}_{\text{bad}} \)

Elementary cell of “good mesh” \( \mathcal{M}_{\text{good}} \)
On “bad” mesh: \[ \sup_{u \in H^2(\Omega)} \frac{\|u - I_1 u\|_{H^1(\Omega)}}{\|u\|_{H^2(\Omega)}} \to \infty \text{ as } h/\delta \to \infty, \]

On “good” mesh: \[ \sup_{u \in H^2(\Omega)} \frac{\|u - I_1 u\|_{H^1(\Omega)}}{\|u\|_{H^2(\Omega)}} \text{ uniformly bounded in } h/\delta. \]

Yet, \[ \inf_{v_N \in S^0_1(M_{\text{bad}})} \|u - v_N\|_{H^1(\Omega)} \leq \inf_{v_N \in S^0_1(M_{\text{good}})} \|u - v_N\|_{H^1(\Omega)} \quad \forall u \in H^2(\Omega). \]

### 5.3.5 General approximation error estimates

In Sect. 5.3.2, we only examined the behavior of norms of the interpolation error for piecewise linear interpolation into \( S^0_1(M) \), that is, the case of Lagrangian finite elements of degree \( p = 1 \).
However, Ex. 5.2.2 sent the clear message that quadratic Lagrangian finite elements achieve faster convergence of the energy norm of the Galerkin discretization error, see Fig. 137, 138.

On the other hand quadratic finite elements could not deliver faster convergence in Ex. 5.2.6.

In this section we learn about theoretical results that shed light on these observations and extend the results of Sect. 5.3.2.

**Remark 5.3.38** ($L^\infty$ interpolation error estimate in 1D).

The faster convergence of quadratic Lagrangian FE in Ex. 5.2.2 does not come as a surprise: recall the estimate from [18, Eq. 9.4.6]:

$$\|u - I_p u\|_{L^\infty([a,b])} \leq \frac{h^{p+1}}{(p+1)!} \|u^{(p+1)}\|_{L^\infty([a,b])} \quad \forall u \in C^{p+1}([a, b]),$$

where $I_p u$ is the $\mathcal{M}$-piecewise polynomial interpolant of $u$ of local degree $p$. It generalizes (5.3.5).
\[
\| u - l_p u \|_{L^\infty([a,b])} = O(h^{p+1})
\]

**Remark 5.3.39** (Local interpolation onto higher degree Lagrangian finite element spaces).

**\( M \): triangular/tetrahedral/quadrilateral/hybrid mesh of domain \( \Omega \) (→ Sect. 3.3.1)**

Recall (→ Sect. 3.4): nodal basis functions of \( p \)-th degree Lagrangian finite element space \( S^0_p(M) \) defined via interpolation nodes, cf. (3.4.3).

Set of interpolation nodes: \( N = \{ p_1, \ldots, p_N \} \subset \overline{\Omega} \), \( N = \dim S^0_p(M) \).

**General nodal Lagrangian interpolation operator**

\[
l_p : \begin{cases} 
C^0(\overline{\Omega}) &\to S^0_p(M) \\
u &\mapsto l_p(u) := \sum_{l=1}^N u(p_l) b^l_N
\end{cases}
\]
where $b^l_N$ are the nodal basis functions.

$$(3.4.3) \Rightarrow I_p(u)(p_l) = u(p_l), \quad l = 1, \ldots, N \quad \text{(Interpolation!)}.$$ 

By virtue of the location of the interpolation nodes, see Ex. 3.4.2, Ex. 3.4.5, and Fig. 103, the nodal interpolation operators are purely local:

$$\forall K \in \mathcal{M}: \quad I_p u|_K = \sum_{i=1}^{Q} u(q^K_i) b^K_i,$$

(5.3.40)

$q^K_i, \ i = 1, \ldots, Q = $ local interpolation nodes in cell $K \in \mathcal{M}$, see Ex. 3.4.2, Ex. 3.4.5, and Fig. 103.

$b^K_i, \ i = 1, \ldots, Q = $ local shape functions: $b^K_i(q^K_j) = \delta_{ij}$.

\textit{Example 5.3.41 (Piecewise quadratic interpolation).} $\rightarrow$ Ex. 3.4.2
triangle \( K = \text{convex}\{a^1, a^2, a^3\}, p = 2 \)

\[\Rightarrow\] local quadratic interpolation:

\[ |_{2}u|_{K} = -\sum_{i=1}^{3} \lambda_i (1 - 2\lambda_i) u(a^i) + \sum_{1\leq i < j \leq 3} 4\lambda_i \lambda_j u(\frac{1}{2}(a^i + a^j)) . \]

local shape functions, see (3.4.4)

The following theorem summarized best approximation results for affine equivalent Lagrangian FE spaces \( S_0^p(\mathcal{M}) \) (\( \rightarrow \) Sect. 3.4) on mesh \( \mathcal{M} \) of a bounded polygonal/polyhedral domain \( \Omega \subset \mathbb{R}^d \). It is the result of many years of research in approximation theory, see [26, Sect. 3.3], [1].
**Theorem 5.3.42** (Best approximation error estimates for Lagrangian finite elements).

Let \( \Omega \subset \mathbb{R}^d \), \( d = 1, 2, 3 \), be a bounded polygonal/polyhedral domain equipped with a mesh \( \mathcal{M} \) consisting of simplices or parallelepipeds. Then, for each \( k \in \mathbb{N} \), there is a constant \( C > 0 \) depending only on \( k \) and the shape regularity measure \( \rho_M \) such that

\[
\inf_{v_N \in \mathcal{S}_p^0(\mathcal{M})} \| u - v_N \|_{H^1(\Omega)} \leq C \left( \frac{h_M}{p} \right)^{\min\{p+1,k\}-1} \| u \|_{H^k(\Omega)} \quad \forall u \in H^k(\Omega). \quad (5.3.43)
\]

This theorem is a typical example of finite element analysis results that you can find in the literature. It is important to know what kind of information can be gleaned from statements like that of Thm. 5.3.42.

**Remark 5.3.44** ("Generic constants").

A statement like (5.3.43) is typical of a priori error estimates in the numerical analysis literature, which often come in the form

\[
\| u - u_N \|_X \leq C \cdot \text{"discretization parameter"} \cdot \| u \|_Y,
\]

where \( C > 0 \) is not specified precisely or only claimed to exist (though, in principle, they could be computed),

- \( C \) must neither depend on the exact solution \( u \) nor the discrete solution \( u_N \),
- the possible dependence of \( C \) on problem parameters or discretization parameters has
Such constants $C > 0$ are known as **generic constants**. Customarily, different generic constants are even denoted by the same symbol (“$C$” is most common).

**Remark 5.3.45** (Nature of a priori estimates). → Sect. 1.6.2

Cea’s lemma, Thm. **5.1.10** $\Rightarrow$ Thm. **5.3.42** implies a priori estimates of the energy norm of the finite element Galerkin discretization error (see also Rem. **5.3.28**) of the form

$$
\|u - u_N\|_a \leq C \left( \frac{h_M}{p} \right)^{\min\{p+1,k\}-1} \|u\|_{H^k(\Omega)},
$$  

(5.3.46)

where $u$ is the exact solution of the discretized 2nd-order elliptic boundary value problem.

(5.3.46) does not give concrete information about $\|u - u_N\|_a$, because

- we do not know the value of the “generic constant” $C > 0$, see Rem. **5.3.44**,
- as $u$ is unknown, a bound for $\|u\|_{H^k(\Omega)}$ may not be available.
A priori error estimates like (5.3.46) exhibit only the \textit{trend} of the (norm of) the discretization error as discretization parameters $h_M$ (mesh width), $p$ (polynomial degree) are varied.

\begin{remark}
Remark 5.3.47. The estimate of Thm. 5.3.42 is \textit{sharp}: the powers of $h_M$ and $p$ cannot be increased.
\end{remark}

What do Thm. 5.3.42, (5.3.46), tell us about the \textit{efficiency} of a Lagrangian finite element Galerkin discretization of a 2nd-order elliptic BVP?

\textbf{Question 5.3.48.} What \textit{computational effort} buys us what \textit{error} (measured in \textit{energy norm})?
Bad luck (→ Rem. 5.3.45): actual error norm remains elusive! Therefore, rephrase the question so that it fits the available information about the effect of changing discretization parameters on the error:

**Question 5.3.49.** What increase in computational effort buys us a prescribed decrease of the (energy norm of the) error?

The answer to this question offers an a priori gauge of the **asymptotic efficiency** of a discretization method.

Convention: computational effort $\approx$ number of unknowns $N = \dim S_p^0(\mathcal{M})$ (problem size)

Framework: family $\mathcal{M}$ of simplicial meshes of domain $\Omega \subset \mathbb{R}^d$, $d = 1, 2, 3$, created by **global regular refinement** of a single initial mesh
Global regular refinement of a simplicial mesh (→ Ex. 5.1.12)

- avoids greater distortion of “child cells” w.r.t. their parents,
- spawns meshes with fairly uniform size $h_K$ of cells.

$$\exists C > 0: \quad \rho_M \leq C, \quad \forall M \in \mathcal{M}. \quad \exists C > 0: \quad \max \{h_K/h_{K'}, \ K, K' \in \mathcal{M}\} \leq C, \quad \forall M \in \mathcal{M}. \,$$

Now, for meshes $\in \mathcal{M}$, we investigate “$N$-dependence”, $N = \dim S_p^0(M)$, of energy norm of finite element discretization error:

Counting argument $\quad N = \dim S_p^0(M) \approx p^d h_\mathcal{M}^{-d} \Rightarrow \frac{h_\mathcal{M}}{p} \approx N^{-1/d}. \quad (5.3.50)$

dimensions of local spaces, Lemma 3.3.6 $\sim \# \mathcal{M} \sim \# \mathcal{V}(\mathcal{M}), \mathcal{E}(\mathcal{M})$ etc.

Notation: $\approx \hat{=} \Rightarrow$ equivalence up to constants only depending on $\gamma$ (in $\mathcal{M}_\gamma$), $\Omega$
Example 5.3.51 (Dimensions of Lagrangian finite element spaces on triangular meshes).

\[ d = 2: \] for triangular meshes \( \mathcal{M} \), by Lemma 3.3.6

\[
\dim S^0_p(\mathcal{M}) = \#\{\text{nodes}(\mathcal{M})\} + \#\{\text{edges}(\mathcal{M})\} (p - 1) + \#\mathcal{M} \frac{1}{2}(p - 1)(p - 2) .
\]

1 basis function per vertex

\( p - 1 \) basis functions per edge

\( \frac{1}{2}(p - 1)(p - 2) \) “interior” basis functions

Geometric considerations: the number of triangles sharing a vertex can be bounded in terms of \( \rho_\mathcal{M} \), because \( \rho_\mathcal{M} \) implies a lower bound for the smallest angles of the triangular cells.

\[ \exists C = C(\rho_\mathcal{M}): \#\{K_j \in \mathcal{M}: \overline{K}_i \cap \overline{K}_j \neq \emptyset\} \leq C \quad (i = 1, 2, \ldots, \#\mathcal{M}) . \]

If every vertex belongs only to a small number of triangles, the number \( \#\{\text{nodes}(\mathcal{M})\} \) can be bounded by \( C \cdot \#\mathcal{M} \), where \( C > 0 \) will depend on \( \rho_\mathcal{M} \) only. The same applies to the edges.

\[ \#\{\text{nodes}(\mathcal{M})\}, \#\{\text{edges}(\mathcal{M})\} \approx \#\mathcal{M} . \]

\[ \dim S^0_p(\mathcal{M}) \approx (\#\mathcal{M})p^2 , \quad (5.3.52) \]
with constants hidden in \( \approx \) depending on \( \rho_M \) only.

Now, we merge (5.3.46) and (5.3.50):

\[
\inf_{v_N \in S_p^0(\mathcal{M})} \| u - v_N \|_{H^1(\Omega)} \leq C N^{-\frac{\min\{p,k-1\}}{d}} \| u \|_{H^k(\Omega)},
\]

(5.3.53)

with \( C > 0 \) depending only on \( d, p, k, \) and \( \rho_M \).

\[ (5.3.53) \rightarrow \text{algebraic convergence} \ (\rightarrow \text{Def. 1.6.20}) \text{ in problem size} \]

\[
(\text{rate} \quad \frac{\min\{p,k-1\}}{d})
\]

We observe that

- the rate of convergence is limited by the polynomial degree \( p \) of the Lagragian FEM,
the rate of convergence is limited by the smoothness of the exact solution $u$, measured by means of the Sobolev index $k$, see Sect. 5.3.3.

the rate of convergence will be worse for $d = 3$ than for $d = 2$, the effect being more pronounced for small $k$ or $p$.

Answer to Question 5.3.49:

Assumption: a priori error estimate (5.3.53) is sharp

$$\exists C = C(u, \ldots) > 0: \text{error norm}(N) \approx CN^{-\min\{p,k-1\}/d} \quad \forall M \in M.$$ 

$$\frac{\text{error norm}(N_1)}{\text{error norm}(N_2)} \approx \left(\frac{N_1}{N_2}\right)^{\frac{\min\{p,k-1\}}{d}}.$$ 

Reduction of (the energy norm of) the error by a factor $\rho > 1$ requires increase of the problem size by factor $\rho^{\min\{p,k-1\}/d}$.
Discussion: Solution $u \in H^k(\Omega) \gg$ optimal asymptotic efficiency for $p = k - 1$

Remark 5.3.54 (General asymptotic estimates).
Recall (→ Sect. 1.6.2): convergence is an asymptotic notion

Now we deduce asymptotic estimates for the best approximation errors from Thm. 5.3.42, and (5.3.53), in particular, for the case \( N \to \infty \):

- **h-refinement:** \( p \) fixed, \( h_M \to 0 \) for \( M \in \mathbb{M} \):
  
  \[
  \inf_{v_N \in S^0_p(M)} \| u - v_N \|_1 = O(N^{-p/d}) \]  
  \[
  \Rightarrow \quad \text{algebraic convergence w.r.t. } N
  \]

  \( \begin{array}{c}
  p \leq k - 1 \\
  k \leq p + 1
  \end{array} \)

- **p-refinement:** \( M \in \mathbb{M} \) fixed, \( p \to \infty \):
  
  \[
  \inf_{v_N \in S^0_p(M)} \| u - v_N \|_1 = O(N^{-(k-1)/d}) \]  
  \[
  \Rightarrow \quad \text{large } p
  \]

  \( \begin{array}{c}
  p \text{ large}
  \end{array} \)

Note: for very smooth solution \( u \), i.e. \( k \gg 1 \), polynomial degree \( p \) limits speed of convergence.
5.4 Elliptic regularity theory

Crudely speaking, in Sect. 5.3.5 we saw that the asymptotic behavior of the Lagrangian finite element Galerkin discretization error (for 2nd-order elliptic BVPs) can be predicted provided that

- we use families of meshes, whose cells have rather uniform size and whose shape regularity measure is uniformly bounded,
- we have an idea about the smoothness of the exact solution $u$, that is, we know $u \in H^k(\Omega)$ for a (maximal) $k$, see Thm. 5.3.42.
Knowledge about the mesh can be taken for granted, but how can we guess the smoothness of the (unknown!) exact solution $u$?

A (partial) answer is given in this section.

Focus: Scalar 2nd-order elliptic BVP with homogeneous Dirichlet boundary conditions

$$-	ext{div} (\sigma(x) \nabla u) = f \quad \text{in} \quad \Omega, \quad u = g \quad \text{on} \quad \partial \Omega.$$ 

To begin with, we summarize the available information:

- **Known**: $u$ solves BVP
- **Information** about coefficient $\sigma$, domain $\Omega$, source function $f$, boundary data $g$

$u$ will belong to a certain class of functions (e.g. subspace $S \subset V$)
Example 5.4.1 (Elliptic lifting result in 1D).

\( d = 1, \Omega = [0, 1[, \) coefficient \( \sigma \equiv 1 \), homogeneous Dirichlet boundary conditions:

\[
u'' = f, \quad u(0) = u(1) = 0.
\]

Obvious:

\[
f \in H^k(\Omega) \Rightarrow u \in H^{k+2}(\Omega) \quad \text{(a lifting theorem)}\]

Can this be generalized to higher dimensions \( d > 1 \)?

Partly so:
Theorem 5.4.2 (Smooth elliptic lifting theorem).
If $\partial \Omega$ is $C^\infty$-smooth, i.e. possesses a local parameterization by $C^\infty$-functions, and $\sigma \in C^\infty(\overline{\Omega})$, then, for any $k \in \mathbb{N}$,

$$
\begin{align*}
  u &\in H^1_0(\Omega) \quad \text{and} \quad -\text{div}(\sigma \text{grad} u) \in H^k(\Omega) \\
  u &\in H^1(\Omega), \quad -\text{div}(\sigma \text{grad} u) \in H^k(\Omega) \quad \text{and} \quad \text{grad} u \cdot n = 0 \quad \text{on} \quad \partial \Omega \\
  \Rightarrow \quad u &\in H^{k+2}(\Omega).
\end{align*}
$$

In addition, for such $u$ there is $C = C(k, \Omega, \sigma)$ such that

$$
\|u\|_{H^{k+2}(\Omega)} \leq C\|\text{div}(\sigma \text{grad} u)\|_{H^k(\Omega)}.
$$
What about non-smooth $\partial \Omega$?

These are very common in engineering applications (“CAD-geometries”).

polygonal domain with corners $c^i$

How will the corners affect the smoothness of solutions of

$$u \in H^1_0(\Omega): \quad \Delta u = f \in C^\infty(\overline{\Omega})?$$

**Example 5.4.3 (Corner singular functions).**
corner singular function

\[ u_s(r, \varphi) = r \frac{\pi}{\omega} \sin \left( \frac{\pi}{\omega} \varphi \right), \quad (5.4.4) \]

\[ r \geq 0, \quad 0 \leq \varphi \leq \omega. \]

(in local polar coordinates)

\[ u_s = 0 \text{ on } \partial \Omega \text{ locally at } c! \]

Straightforward computation:

\[ \Delta u_s = 0 \quad \text{in} \quad \Omega! \]

To see this recall: \( \Delta \) in polar coordinates:

\[ \Delta u = \frac{1}{r} \frac{\partial}{\partial r} \left( r \frac{\partial u}{\partial r} \right) + \frac{1}{r^2} \frac{\partial^2 u}{\partial \varphi^2}. \quad (5.4.5) \]

\[ \Delta u_s(r, \varphi) = \frac{1}{r} \frac{\partial}{\partial r} \left( r \frac{\pi}{\omega} \frac{\pi}{\omega - 1} \sin \left( \frac{\pi}{\omega} \varphi \right) \right) + \frac{1}{r^2} \frac{\pi}{\omega} \frac{\partial}{\partial \varphi} \cos \left( \frac{\pi}{\omega} \varphi \right) \frac{\pi}{\omega} \]

\[ = \left( \frac{\pi}{\omega} \right)^2 r \frac{\pi}{\omega - 2} \sin \left( \frac{\pi}{\omega} \varphi \right) - \left( \frac{\pi}{\omega} \right)^2 r \frac{\pi}{\omega - 2} \sin \left( \frac{\pi}{\omega} \varphi \right) = 0. \]
What is “singular” about these functions? Plot them for $\omega = \frac{3\pi}{2}$, cf. Ex. 5.2.6

- $u_\omega$ for $\omega = \frac{3\pi}{2}$ (Fig. 164)
- $\|\text{grad } u_\omega\|$ for $\omega = \frac{3\pi}{2}$ (Fig. 165)
Recall gradient (2.3.19) in polar coordinates

\[
\text{grad } u = \frac{\partial u}{\partial r} e_r + \frac{1}{r} \frac{\partial u}{\partial \varphi} e_\varphi .
\] (2.3.19)

\[
\Rightarrow \quad \text{grad } u_s(r, \varphi) = \frac{\pi}{\omega} r^{\frac{\pi}{\omega} - 1} \left( \sin\left(\frac{\pi}{\omega}\varphi\right) e_r + \cos\left(\frac{\pi}{\omega}\varphi\right) e_\varphi \right).
\]

\( \omega > \pi \) (“re-entrant corner”) \( \Rightarrow \) “\( \text{grad } u_s(0) = \infty \)”

How does this “blow-up” of the gradient affect the Sobolev regularity (that is, the smoothness as expressed through “\( u_s \in H^k(\Omega) \)” of the corner singular function \( u_s \)？

We try to compute \( \|u\|_{H^2(D)} \), with (in polar coordinates, see Fig. 163)

\[
D := \{(r, \varphi) : 0 < r < 1, \ 0 < \varphi < \omega\} .
\]

By tedious computations we find

\[
\omega > \pi \quad \Rightarrow \quad \int_D \left\| D^2 u_s(r, \varphi) \right\|^2_F \ r d(r, \varphi) = \infty .
\]

Def. 5.3.30 \( \Rightarrow \) \[
\{ \omega > \pi \quad \Rightarrow \quad u_s \notin H^2(D) \ \}.
\]
Bad news: With the exception of concocted examples, corner singular functions like (5.4.4) will be present in the solution of linear scalar 2nd-order elliptic BVP on polygonal domains!

The meaning of “being present” is elucidated in the following theorem:
Theorem 5.4.6 (Corner singular function decomposition).

Let \( \Omega \subset \mathbb{R}^2 \) be a polygon with \( J \) corners \( c^i \). Denote the polar coordinates in the corner \( c^i \) by \((r_i, \varphi_i)\) and the inner angle at the corner \( c^i \) by \( \omega_i \). Additionally, let \( f \in H^l(\Omega) \) with \( l \in \mathbb{N}_0 \) and \( l \neq \lambda_{ik} - 1 \), where the \( \lambda_{ik} \) are given by the singular exponents

\[
\lambda_{ik} = \frac{k\pi}{\omega_i} \quad \text{for } k \in \mathbb{N}.
\] (5.4.7)

Then \( u \in H^1_0(\Omega) \) with \(-\Delta u = f\) in \( \Omega \) can be decomposed

\[
u = u^0 + \sum_{i=1}^{J} \psi(r_i) \sum_{\lambda_{ik}<l+1} \kappa_{ik} s_{ik}(r_i, \varphi_i), \quad \kappa_{ik} \in \mathbb{R},
\] (5.4.8)

with regular part \( u^0 \in H^{l+2}(\Omega) \), cut-off functions \( \psi \in C^\infty(\mathbb{R}^+) \) (\( \psi \equiv 1 \) in a neighborhood of \( 0 \)), and corner singular functions

\[
\lambda_{ik} \notin \mathbb{N}: \quad s_{ik}(r, \varphi) = r^{\lambda_{ik}} \sin(\lambda_{ik}\varphi),
\]
\[
\lambda_{ik} \in \mathbb{N}: \quad s_{ik}(r, \varphi) = r^{\lambda_{ik}}(\ln r) \sin(\lambda_{ik}\varphi).
\] (5.4.9)
$\Omega \subset \mathbb{R}^2$ has re-entrant corners $\Rightarrow$ if $u$ solves $\Delta u = f$ in $\Omega$, $u = 0$ on $\partial \Omega$, then $u \not\in H^2(\Omega)$ in general.

**Theorem 5.4.10** (Elliptic lifting theorem on convex domains).

If $\Omega \subset \mathbb{R}^d$ convex, $u \in H^1_0(\Omega)$, $\Delta u \in L^2(\Omega)$ $\Rightarrow$ $u \in H^2(\Omega)$.

Terminology: if conclusion of Thm. 5.4.10 true $\rightarrow$ Dirichlet problem 2-regular.

Similar lifting theorems also hold for Neumann BVPs, BVPs with smooth coefficients.

**Remark 5.4.11** (Causes for non-smoothness of solutions of elliptic BVPs).

Causes for poor Sobolev regularity of solution $u$ of BVPs for $-\text{div}(\sigma(x)\text{grad }u) = f$:
• Corner of $\partial \Omega$, see above
• Discontinuities of $\sigma$  
  $\rightarrow$  singular functions at “material corners”
• Mixed boundary conditions
• Non-smooth source function $f$

5.5 Variational crimes
Variational crime \(= \) replacing (exact) discrete (linear) variational problem

\[
    u_N \in V_{0,N}: \quad a(u_N, v_N) = f(v_N) \quad \forall v_N \in V_{0,N},
\]

with perturbed variational problem

\[
    \tilde{u}_N \in V_{0,N}: \quad a_N(\tilde{u}_N, v_N) = f_N(v_N) \quad \forall v_N \in V_{0,N}.
\]

\[
\blacktriangleright \quad \text{perturbation of Galerkin solution } u_N \quad \Rightarrow \quad \text{perturbed solution } \tilde{u}_N \in V_{0,N}
\]

Approximations \(a_N(\cdot, \cdot) \approx a(\cdot, \cdot), \ f_N(\cdot) \approx f(\cdot)\) due to

- use of numerical quadrature \(\rightarrow\) Sect. 3.5.4,
- approximation of boundary \(\partial \Omega\) \(\rightarrow\) Sect. 3.6.4.

We are all sinners! Variational crimes are \emph{inevitable} in practical FEM, recall Rem. 1.5.3!

Which “variational petty crimes” can be tolerated?
5.5.1 Impact of numerical quadrature

Model problem: on polygonal/polyhedral $\Omega \subset \mathbb{R}^d$:

$$u \in H^1_0(\Omega): \quad a(u, v) := \int_{\Omega} \sigma(x) \nabla u \cdot \nabla v \, dx = f(v) := \int_{\Omega} fv \, dx \quad (5.5.3)$$

Assumptions:

- $\sigma$ satisfies (2.5.4)
- $\sigma \in C^0(\overline{\Omega})$, $f \in C^0(\overline{\Omega})$

- Galerkin finite element discretization, $V_N := S^0_p(\mathcal{M})$ on simplicial mesh $\mathcal{M}$

- Approximate evaluation of $a(u_N, v_N)$, $f(v_N)$ by a fixed stable local numerical quadrature rule ($\rightarrow$ Sect. 3.5.4)

➤ perturbed bilinear form $a_N$, right hand side $f_N$ (see (5.5.1))
Focus: \textit{h-refinement} (key discretization parameter is the mesh width $h_{\mathcal{M}}$)

\textit{Example 5.5.4} (Impact of numerical quadrature on finite element discretization error).

$\Omega = [0, 1]^2$, $\sigma \equiv 1$, $f(x, y) = 2\pi^2 \sin(\pi x) \sin(\pi y)$, $(x, y)^T \in \Omega$

➢ solution $u(x, y) = \sin(\pi x) \sin(\pi y)$, $g = 0$.

Details of numerical experiment:

- \textit{Quadratic} Lagrangian FE ($V_N = S_2^0(\mathcal{M})$) on triangular meshes $\mathcal{M}$, obtained by regular refinement
- “Exact” evaluation of bilinear form by very high order quadrature
- $f_N$ from one point quadrature rule (3.5.37) of order 2
$H^1(\Omega)$-norm of discretization error on unit square ($\leftrightarrow$ rule (3.5.37), $\leftrightarrow$ rule (3.5.38))

Observation: Use of quadrature rule of order 2 $\Rightarrow$ Algebraic rate of convergence (w.r.t. $N$) drops from $\alpha = 1$ to $\alpha = 1/2$!
Finite element theory [8, Ch. 4,§4.1] tells us that the Guideline 5.5.2 can be met, if the local numerical quadrature rule has sufficiently high order. The quantitative results can be condensed into the following rules of thumb:

\[ \| u - u_N \|_1 = O(h^p_M) \text{ at best} \quad \Rightarrow \quad \text{Quadrature rule of order } 2p - 1 \text{ sufficient for } f_N. \]

\[ \| u - u_N \|_1 = O(h^p_M) \text{ at best} \quad \Rightarrow \quad \text{Quadrature rule of order } 2p - 1 \text{ sufficient for } a_N. \]

5.5.2 Approximation of boundary

We focus on 2nd-order scalar linear variational problems as in the previous section.
Example 5.5.5 (Impact of linear boundary fitting on FE convergence).

Setting: \( \Omega := B_1(0) := \{ x \in \mathbb{R}^2 : |x| < 1 \} \), \( u(r, \varphi) = \cos(r \pi / 2) \) (polar coordinates)

\[
\Rightarrow f = \frac{\pi}{2r} \sin(r \pi / 2) + \frac{\pi}{2} \cos(r \pi / 2)
\]

- Sequences of unstructured triangular meshes \( \mathcal{M} \) obtained by \textbf{regular refinement} (of coarse mesh with 4 triangles) + linear boundary fitting.
- Galerkin FE discretization based on \( V_N := S_{1,0}^0(\mathcal{M}) \) or \( V_N := S_{2,0}^0(\mathcal{M}) \).
- Recorded: approximate norm \( |u - u_N|_{1,\Omega_h} \), evaluated using numerical quadrature rule (3.5.38).

(FE solution extended beyond the domain covered by \( \mathcal{M} \) (“mesh interior”) to \( \Omega \) (“full domain”) by means of polynomial extrapolation.)
Linearly boundary fitted unstructured triangular meshes of $\Omega = B_1(0)$.

Fig. 168

Fig. 169
Discretization errors with respect to $H^1$ semi-norm

Mesh width [log] vs. Discretization error [log]

QFE error on full domain
LFE error on full domain
LFE error on mesh interior

Fig. 170

$H^1(\Omega)$-norm of discretization error on unit ball ($\leftrightarrow p = 1$, $\leftrightarrow p = 2$)

Theoretical guideline:

If $V_{0,N} = S_p^0(\mathcal{M})$, use boundary fitting with polynomials of degree $p$. 

5.6 Duality techniques

5.6.1 Linear output functionals

Adopt abstract setting of Sect. 5.1:

**linear variational problem** (1.4.6) in the form

\[ u \in V_0: \quad a(u, v) = \ell(v) \quad \forall v \in V_0 , \]  

\[ \text{(3.1.1)} \]

- \( V_0 \) = (real) vector space, a space of functions \( \Omega \mapsto \mathbb{R} \) for scalar 2nd-order elliptic variational problems, usually “energy space” \( H^1(\Omega)/H^1_0(\Omega) \), see Sect. 2.2

- \( a : V_0 \times V_0 \mapsto \mathbb{R} \) = a **bilinear form**, see Def. 1.3.13,

- \( \ell : V_0 \mapsto \mathbb{R} \) = a **linear form**, see Def. 1.3.13,
Assumptions 5.1.1, 5.1.2, 5.1.3 are supposed to hold \(\Rightarrow\) existence, uniqueness, and stability of solution \(u\) by Thm. 5.1.4.

(Examples of 2nd-order linear BVPs discussed in Rem. 5.1.5, Sect. 2.8)

Galerkin discretization using \(V_{0,N} \subset V_0\) \(\Rightarrow\) discrete variational problem

\[
    u_N \in V_{0,N}: \quad a(u_N, v_N) = f(v_N) \quad \forall v_N \in V_{0,N}.
\]

New twist: we are interested mainly/only in the number \(F(u)\), where \(F: V_0 \mapsto \mathbb{R}\) is an output functional.

Mathematical terminology: functional \(\doteq\) mapping from a function space into \(\mathbb{R}\)

**Example 5.6.1 (Output functionals).**

Some output functionals for solutions of PDEs commonly encountered in applications:
• mean values, see Ex. 5.6.4 below
• total heat flux through a surface (for heat conduction model \(\rightarrow\) Sect. 2.5), see Ex. 5.6.13 below
• total surface charge of a conducting body (for electrostatics \(\rightarrow\) Sect. 2.1.2)
• total heat production (Ohmic losses) by stationary currents
• total force on a charged conductor (for electrostatics \(\rightarrow\) Sect. 2.1.2)
• lift and drag in computational fluid dynamics (aircraft simulation)
• and many more . . .

We consider output functionals with special properties, which are rather common in practice:

**Assumption 5.6.2** (Linearity of output functional).

The output functional \(F\) is a [linear form](Def. 1.3.13) on \(V_0\).
To put the next assumption into context, please recall Ass. 5.1.2 and Rem. 2.3.11.

**Assumption 5.6.3** (Continuity of output functional).

The output functional is **continuous** w.r.t. the **energy norm** in the sense that

\[ \exists C_f > 0: \ |F(v)| \leq C_f \|v\|_a \quad \forall v \in V_0. \]

Now consider Galerkin discretization of (3.1.1) based on Galerkin trial/test space \( V_{0,N} \subset V_0 \), \( N := \dim V_{0,N} < \infty \) ➞ discrete variational problem

\[ u_N \in V_{0,N}: \ a(u_N, v_N) = \ell(v_N) \quad \forall v_N \in V_{0,N}. \quad (3.1.4) \]

What would you dare to sell as an approximation of \( F(u) \)? Of course, . . .

Galerkin solution \( u_N \in V_{0,N} \) ➞ approximate output value \( F(u_N) \)
How accurate is $F(u_N)$, that is, how big is the output error $|F(u) - F(u_N)|$?

Linearity (→ Ass. 5.6.2) and continuity Ass. 5.6.3 conspire to furnish a very simple estimate

$$|F(u) - F(u_N)| \leq C_f \|u - u_N\|_a.$$ 

A priori estimates for $\|u - u_N\|_a$ ⇒ estimates for $|F(u) - F(u_N)|$

Hence, Thm. 5.3.42 immediately tells us the asymptotic convergence of linear and continuous output functionals defined for solutions of 2nd-order scalar elliptic BVPs and Lagrangian finite element discretization.

**Example 5.6.4 (Approximation of mean temperature).**

Heat conduction model (→ Sect. 2.5), scaled heat conductivity $\kappa \equiv 1$, on domain $\Omega = ]0, 1[^2$, fixed temperature $u = 0$ on $\partial \Omega$:

$$-\Delta u = f \quad \text{in } \Omega \quad , \quad u = 0 \quad \text{on } \partial \Omega .$$

Heat source function $f(x, y) = 2\pi^2 \sin(\pi x) \sin(\pi y)$, $(x, y)^T \in \Omega$

⇒ solution $u(x, y) = \sin(\pi x) \sin(\pi y)$.
mean temperature \[ F(u) = \frac{1}{|\Omega|} \int_{\Omega} u \, dx. \]

Details of finite element Galerkin discretization:

- Sequence of triangular meshes \( \mathcal{M} \) created by regular refinement.
- Galerkin discretization: \( V_{0,N} := S_{1,0}^0(\mathcal{M}) \) (linear Lagrangian finite elements \( \rightarrow \) Sect. 3.2).
- Quadrature rule (3.5.38) of order 6 for assembly of right hand side vector (more than sufficiently accurate \( \rightarrow \) guidelines from Sect. 5.5.1).

Expected: algebraic convergence in \( h_{\mathcal{M}} \) with rate 1 of approximate mean temperature
Observation: Mean value converges twice as fast as expected: algebraic convergence $O(h_\mathcal{M}^2)$!
Theorem 5.6.5 (Duality estimate for linear functional output).

Define the dual solution $g_F \in V_0$ to $F$ as solution of

$$g_F \in V_0: \quad a(v, g_F) = F(v) \quad \forall v \in V_0.$$  

Then

$$|F(u) - F(u_N)| \leq \|u - u_N\|_a \inf_{v_N \in V_{0,N}} \|g_F - v_N\|_a. \quad (5.6.6)$$

Proof. For any $v_N \in V_{0,N}$:

$$F(u) - F(u_N) = a(u - u_N, g_F) \overset{(*)}{=} a(u - u_N, g_F - v_N) \leq \|u - u_N\|_a \|g_F - v_N\|_a.$$

$(*) \leftarrow$ by Galerkin orthogonality $(5.1.7)$.

If $g_F$ can be approximated well in $V_{0,N}$, then the output error can converge $\to 0$ (much) faster than $\|u - u_N\|_a$.

Example 5.6.7 (Approximation of mean temperature cnt’d).  $\rightarrow$ Ex. 5.6.4
The mean temperature functional (5.6.6) is obviously linear $\to$ Ass. 5.6.2.

By the Cauchy-Schwarz inequality (2.2.15) it clearly satisfies Ass. 5.6.3 even with $\|\cdot\|_a = \|\cdot\|_{L^2(\Omega)}$, let alone for $\|\cdot\|_a = |\cdot|_{H^1(\Omega)}$ on $H^1_0(\Omega)$.

What is $g_F \in H^1_0(\Omega)$ in this case? By Thm. 5.6.5 it is the solution of the variational problem

$$\int_\Omega \nabla g_F \cdot \nabla v \, dx = F(v) = \frac{1}{|\Omega|} \int_\Omega v \, dx \quad \forall v \in H^1_0(\Omega).$$

The associated 2nd-order BVP reads

$$-\Delta g_F = \frac{1}{|\Omega|} \quad \text{in } \Omega, \quad g_F = 0 \quad \text{on } \partial \Omega.$$

Now recall the elliptic lifting theory Thm. 5.4.10 for convex domains: since $\Omega = ]0, 1[^2$ is convex, we conclude $g_F \in H^2(\Omega)$.

By interpolation estimate of Thm. 5.3.27 ($i_1 \hat{=} \text{linear interpolation onto } S^0_1(\mathcal{M})$)

$$\inf_{v_N \in S^0_1(\mathcal{M})} |g_F - v_N|_{H^1(\Omega)} \leq |g_F - i_1 g_F|_{H^1(\Omega)} \leq C h_{\mathcal{M}} |g_F|_{H^2(\Omega)},$$

where $C > 0$ may depend on $\Omega$ and the shape regularity measure ($\to$ Def. 5.3.26) of $\mathcal{M}$. 
Plug this into the duality estimate (5.6.6) of Thm. 5.6.5 and note that $u \in H^2(\Omega)$ by virtue of Thm. 5.4.10 and $f \in L^2(\Omega)$:

$$|F(u) - F(u_N)| \leq C h_M \cdot |u - u_N|_{H^1(\Omega)} \leq C h_M^2,$$

where the “generic constant” $C > 0$ depends only on $\Omega, u, \rho_M$.

Again, by the elliptic lifting theory Thm. 5.4.10 we infer that $u \in H^2(\Omega)$ holds for this example since $f \in L^2(\Omega)$. 

5.6.2 Case study: Boundary flux computation

Model problem (process engineering):

Long pipe carrying turbulent flow of coolant (water)

\[ \Omega \subset \mathbb{R}^2 : \text{cross-section of pipe} \]
\[ \kappa : \text{(scaled) heat conductivity of pipe material (assumed homogeneous, } \kappa = \text{const)} \]

Assumption: Constant temperatures \( u_o, u_i \) at outer/inner wall \( \Gamma_o, \Gamma_i \) of pipe

Task: Compute heat flow pipe \( \rightarrow \) water

Mathematical model: elliptic boundary value for stationary heat conduction (→ Sect. 2.5)

\[
- \text{div}(\kappa \text{grad } u) = 0 \quad \text{in } \Omega, \quad u = u_x \quad \text{on } \Gamma_x, \; x \in \{i, o\}.
\]  
(5.6.8)

Heat flux through \( \Gamma_i \):

\[
J(u) := \int_{\Gamma_i} \kappa \text{grad } u \cdot n \, dS.
\]  
(5.6.9)

Relate to abstract framework:

\[
(5.6.8) \cong (3.1.1), \quad V_0 \cong H^1_0(\Omega) \quad (\rightarrow \text{Sect. 2.8})
\]
(Actually, \( u \in H^1(\Omega) \), but by means of \textit{offset functions} we can switch to the variational space \( H^1_0(\Omega) \), see Sects. 2.1.3, 3.5.5.)

Numerical method: finite element computation of heat conduction in pipe
(e.g. linear Lagrangian finite element Galerkin discretization, Sect. 3.2)

Expectation: \textcolor{red}{Algebraic convergence} \( |J(u) - J(u_N)| = O(h^2_M) \) for regular \( h \)-refinement

This expectation is based on the analogy to Ex. 5.6.4 (Approximation of mean temperature), where duality estimates yielded \( O(h^2_M) \) convergence of the mean temperature error in the case of Galerkin discretization by means of linear Lagrangian finite elements on a sequence of meshes obtained by regular refinement. Now, it seems, we can follow the same reasoning.

\begin{example}

\textbf{Example 5.6.10 (Computation of heat flux).}

\end{example}

\begin{itemize}
  \item Setting: model problem “heat flux pipe to water”, see (5.6.8) and Fig. 172.
  \item Linear output functional from (5.6.9)
\end{itemize}
Domain \( \Omega = B_{R_0}(0) \setminus B_{R_i}(0) := \{ x \in \mathbb{R}^2 : R_i < |x| < R_o \} \) with \( R_o = 1 \) and \( R_i = 1/2 \)

Dirichlet boundary data \( u_i = 60^\circ C \) on \( \Gamma_i \), \( u_o = 10^\circ C \) on \( \Gamma_o \), heat source \( f \equiv 0 \), heat conductivity \( \kappa \equiv 1 \).

Exact solution: \( u(r, \varphi) = C_1 \ln(r) + C_2 \),

with \( C_1 := (u_o - u_i)/(\ln R_i - \ln R_o) \), \( C_2 := (\ln R_o u_i - \ln R_i u_o)/(\ln R_i - \ln R_o) \).

Details of linear Lagrangian finite element Galerkin discretization:

- Sequences of unstructured triangular meshes \( \mathcal{M} \) obtained by regular refinement of coarse mesh (from grid generator).
- Galerkin FE discretization based on \( V_{0,N} := S^0_{1,0}(\mathcal{M}) \).
- Approximate evaluation of \( a(u_N, v_N), f(v_N) \) by six point quadrature rule (3.5.38) (“overkill quadrature”, see Sect. 5.5.1)
- Approximate evaluation of \( J(u_N) \) by 4 point Gauss-Legendre quadrature rule on boundary edges of \( \mathcal{M} \).
- Linear boundary approximation (circle replaced by polygon).
- Recorded: errors \( |J - J(u_N)| \) on sequence of meshes.
Unstructured triangular meshes for $\Omega = B_1(0) \setminus B_{1/2}(0)$ (two coarsest specimens).
Observation:

**Algebraic convergence** of output error for $J$ from (5.6.9) *only with rate 1* (in mesh width $h_M$)!

(This is not the fault of the piecewise linear boundary approximation, which is sufficient when using piecewise linear Lagrangian finite elements, see Sect. 5.5.2.)

Why was our expectation mistaken?

Suspicion: the output functional $J$ fails to meet requirements of duality estimates of Thm. 5.6.5.
boundary flux functional $J$ from (5.6.9) is not continuous on $H^1(\Omega)$!

**Example 5.6.11 (Non-continuity of boundary flux functional).**

Idea: find $u \in H^1(\Omega)$, for which “$J(u) = \infty$”,

*cf.* investigation of non-continuity of point evaluation functional on $H^1(\Omega)$ → Rem. 2.3.16.

On $\Omega = \{x \in \mathbb{R}^2: \|x\| < 1\}$ (unit disk) consider

$$u(x) = (1 - \|x\|)^\alpha =: g(\|x\|), \quad \frac{1}{2} < \alpha < 1,$$

and the boundary flux functional (5.6.9) on $\partial \Omega$.

☞ On the one hand, using the expression (2.3.19) for the gradient in polar coordinates,

$$J_0(v) = \int_{\partial \Omega} \frac{\partial u(x)}{\partial r}(x) \, dS(x) = 2\pi \alpha (1 - r)^{\alpha - 1} \bigg|_{r=1} \approx \infty.$$

R. Hiptmair
C. Schwab,
H. Harbrecht
V. Gradinaru
A. Chernov
SAM, ETHZ

5.6 Numerical Methods for PDEs

p. 569
On the other hand, straightforward computation of improper integral using (2.3.20):

\[ |u|^2_{H^1(\Omega)} = \int_{\Omega} \|\nabla u(x)\|^2 \, dx = 2\pi \int_0^1 |g'(r)|^2 r \, dr = 2\pi \alpha^2 \int_0^1 (1 - r)^{2\alpha - 2} r \, dr \]

\[ = 2\pi \alpha^2 \int_0^1 s^{2\alpha - 2}(1 - s) \, ds = 2\pi \alpha \left[ \frac{s^{2\alpha - 1}}{2\alpha - 1} - \frac{s^{2\alpha}}{2\alpha} \right]_{s=0}^{s=1} = 2\pi \frac{1}{2\alpha - 1} < \infty . \]

Def. 2.2.12

\[ u \in H^1(\Omega) \quad (u \in C^0(\overline{\Omega}) \text{ and } u \in C^\infty(\Omega \setminus \{0\}) !) . \]

Ex. 5.6.11 ➝ Thm. 5.6.5 cannot be applied

(Potentially) poor convergence of flux obtained from straightforward evaluation of \( J(u_N) \) for FE solution \( u_N \in S^0_{1,0}(\mathcal{M}) ! \)

 Apparently there is no remedy, because the boundary flux functional (5.6.9) seems to be enforced on
us by the problem: we are not allowed to tinker with it, are we?

Trick:

use fixed cut-off function $\psi \in C^0(\overline{\Omega}) \cap H^1(\Omega)$, $\psi \equiv 1$ on $\Gamma_i$, $\psi|_{\Gamma_o} = 0$

$$
\int_{\Gamma_i} \kappa \, \text{grad} \, u \cdot n \, dS = \int_{\Gamma_i} (\kappa \, \text{grad} \, u \cdot n) \, \psi \, dS = \int_{\Omega} \left( \text{div}(\kappa \, \text{grad} \, u) \, \psi + \kappa \, \text{grad} \, u \cdot \text{grad} \, \psi \right) \, dx = 0
$$

\[
\text{use } J^*(u) := \int_{\Omega} \kappa \, \text{grad} \, u \cdot \text{grad} \, \psi \, dx \, . \quad (5.6.12)
\]

Obviously (*): $J^* : H^1(\Omega) \mapsto \mathbb{R}$ continuous & $J^*(u) = J(u)$ for solution of (5.6.8)

(*): By the Cauchy-Schwarz inequality (2.2.15), since $\kappa = \text{const}$,

$$
|J^*(u)| \leq \kappa \| \text{grad} \, u \|_{L^2(\Omega)} \| \text{grad} \, \psi \|_{L^2(\Omega)} \leq C |u|_{H^1(\Omega)} ,
$$

with $C := \kappa \| \text{grad} \, \psi \|_{L^2(\Omega)}$, which is a constant independent of $u$, as $\psi$ is a fixed function.
Objection: You cannot just tamper with the output functional of a problem just because you do not like it!

Retort: Of course, one can replace the output function $J$ with another one $J^*$ as long as

$$J(u) = J^*(u) \quad \text{for the exact solution } u \text{ of the BVP},$$

because the objective is not to “evaluate $J$”, but to obtain an approximation for $J(u)$!

Example 5.6.13 (Computation of heat flux cnt’d). → Ex. 5.6.13

Further details on flux evaluation:

- Galerkin FE discretization based on $V_{0,N} := S_{1,0}^0(\mathcal{M})$ or $V_{0,N} := S_{2,0}^0(\mathcal{M})$.
- Approximate evaluation of $J^*(u_N)$ by six point quadrature rule (3.5.38) (“overkill quadrature”, see Sect. 5.5.1)
- Cut-off function with linear decay in radial direction
- Recorded: errors $|J - J(u_N)|$ and $|J - J^*(u_N)|$. 
Convergence of $|J(u) - J(u_N)|$ and $|J(u) - J^*(u_N)|$ for linear Lagrangian finite element discretization.

Additional observations:

- Algebraic convergence $|J(u) - J^*(u_N)| = O(h_M^2)$ (rate 2 !) for alternative output functional $J^*$ from (5.6.12).
- Dramatically reduced output error!
Remark 5.6.14 (Finding continuous replacement functionals).

Now you will ask: How can we find good (continuous) replacement functionals, if we are confronted with an unbounded output functional on the energy space?

Unfortunately, there is no recipe, and sometimes it does not seem to be possible to find a suitable $J^*$ at all, for instance in the case of point evaluation, cf. Rem. 2.3.16.

Good news: another opportunity to show off how smart you are!

5.6.3 $L^2$-estimates
So far we have only studied the energy norm ($\leftrightarrow H^1(\Omega)$-norm, see Rem. 5.3.28) of the finite element discretization error for 2nd-order elliptic BVP.

The reason was the handy tool of Cea’s lemma Thm. 5.1.10.

What about error estimates in other “relevant norms”, e.g.,

- in the mean square norm or $L^2(\Omega)$-norm, see Def. 2.2.5,
- in the supremum norm or $L^\infty(\Omega)$-norm, see Def. 1.6.4?

In this section we tackle $\|u - u_N\|_{L^2(\Omega)}$. We largely reuse the abstract framework of Sect. 5.6.1: linear variational problem (3.1.1) with exact solution $u \in V_0$, Galerkin finite element solution $u_N \in V_{0,N}$, see p. 554, and the special framework of linear 2nd-order elliptic BVPs, see Rem. 5.1.5: concretely,

$$a(u, v) := \int_\Omega \kappa(x) \text{grad} u \cdot \text{grad} v \, dx, \quad u, v \in H^1_0(\Omega).$$

Example 5.6.15 ($L^2$-convergence of FE solutions). → Ex. 5.2.4
Setting: \( \Omega = [0, 1]^2 \), \( D \equiv 1 \), \( f(x, y) = 2\pi^2 \sin(\pi x) \sin(\pi y) \), \( (x, y)^T \in \Omega \).

\[ u(x, y) = \sin(\pi x) \sin(\pi y). \]

- Sequence of triangular meshes \( \mathcal{M} \), created by regular refinement.
- FE Galerkin discretization based on \( S^0_{1,0}(\mathcal{M}) \) or \( S^0_2(\mathcal{M}) \).
- Quadrature rule (3.5.38) for assembly of local load vectors (\( \rightarrow \) Sect. 3.5.4).
- Approximate \( L^2(\Omega) \)-norm by means of quadrature rule (3.5.38).
Observations:  
1. Linear Lagrangian FE ($p = 1$) \[ \| u - u_N \|_0 = O(N^{-1}) \]
2. Quadratic Lagrangian FE ($p = 2$) \[ \| u - u_N \|_0 = O(N^{-1.5}) \]
Remark 5.6.16 ($L^2$ interpolation error).

Recall the interpolation error estimate of Thm. 5.3.27

$$\| u - I_1 u \|_{L^2(\Omega)} = O(h^2_M) \quad \text{vs.} \quad \| u - I_1 u \|_{H^1(\Omega)} = O(h_M),$$

on a family of meshes with uniformly bounded shape regularity measure.

Higher rate of algebraic convergence of the interpolation error when measured in the weaker $L^2(\Omega)$-norm compared to the stronger $H^1(\Omega)$-norm.

Therefore a similar observation in the case of the finite element approximation error is not so surprising.

Now we supply a rigorous underpinning and explanation of the behavior of $\| u - u_N \|_{L^2(\Omega)}$ that we have observed and expect.
Idea: Consider special continuous linear output functional

\[ J(v) := \int_{\Omega} v \cdot (u - u_N) \, dx \]

This functional is highly relevant for \( L^2 \)-estimates, because

\[ F(u) - F(u_N) = \| u - u_N \|_{L^2(\Omega)}^2 \]

➤ estimates for the output error will provide bounds for \( \| u - u_N \|_{L^2(\Omega)} \)

Note: Both \( u \) and \( u_N \) are fixed functions \( \in H^1(\Omega) \)

➤ Linearity of \( J \) (→ Ass. 5.6.2) is obvious.

➤ Continuity \( J : H^1_0(\Omega) \rightarrow \mathbb{R} \) (→ Ass. 5.6.3) is clear, use Cauchy-Schwarz inequality (2.2.15).

Duality estimate of Thm. 5.6.5 can be applied:
\[ F(u) - F(u_N) = \|u - u_N\|_{L^2(\Omega)}^2 \leq C|u - u_N|_{H^1(\Omega)} \inf_{v_N \in V_{0,N}} |g_F - v_N|_{H^1(\Omega)}, \]

(5.6.17)

where \( C > 0 \) may depend only on \( \kappa \), and the dual solution \( g_F \in H^1_0(\Omega) \) satisfies

\[
 a(g_F, v) = F(v) \quad \forall v \in V_0 \quad \Leftrightarrow \quad \iint_{\Omega} \kappa(\mathbf{x}) \text{grad } g_F \cdot \text{grad } v \, d\mathbf{x} = \int_{\Omega} v(u - u_N) \, d\mathbf{x} \quad \forall v \in H^1_0(\Omega)
\]

\[ -\text{div}(\kappa(\mathbf{x}) \text{grad } g_F) = u - u_N \quad \text{in } \Omega, \quad g_F = 0 \quad \text{on } \partial \Omega. \] (5.6.18)

**Assumption 5.6.19** (2-regularity of homogeneous Dirichlet problem).
We assume that the homogeneous Dirichlet problem with coefficient \( \kappa \) is 2-regular on \( \Omega \): There is \( C > 0 \), which depends on \( \Omega \) only such that

\[
 u \in H^1_0(\Omega) \quad \text{div}(\kappa(\mathbf{x}) \text{grad } u) \in L^2(\Omega) \quad \Rightarrow \quad u \in H^2(\Omega) \quad \text{and} \quad |u|_{H^2(\Omega)} \leq C \|\text{div}(\kappa(\mathbf{x}) \text{grad } u)\|_{L^2(\Omega)}.
\]
By the elliptic lifting theorem for convex domains Thm. 5.4.10 we know

\[ \Omega \text{ convex} \implies \text{Ass. 5.6.19} \text{ is satisfied.} \]

Ass. 5.6.19 in conjunction with (5.6.18) yields

\[ |g_F|_{H^2(\Omega)} \leq C \|u - u_N\|_{L^2(\Omega)}, \tag{5.6.20} \]

where \(C > 0\) depends only on \(\Omega\).

Now we can appeal to the general best approximation theorem for Lagrangian finite element spaces Thm. 5.3.42:

\[ \inf_{v_N \in S^0_p(M)} |g_F - v_N|_{H^1(\Omega)} \leq C \frac{h_M}{p} |g_F|_{H^2(\Omega)} \leq C \frac{h_M}{p} \|u - u_N\|_{L^2(\Omega)}, \tag{5.6.21} \]

where the “generic constants” \(C > 0\) depend only on \(\Omega\) and the shape regularity measure \(\rho_M\) (Def. 5.3.26).

Combine (5.6.17) and (5.6.21) and cancel one power of \(\|u - u_N\|_{L^2(\Omega)}\):

With \(C > 0\) depending only on \(\Omega, \kappa\), and the shape regularity measure \(\rho_M\) we conclude
Ass. 5.6.19 \Rightarrow \| u - u_N \|_{L^2(\Omega)} \leq C \frac{hM}{p} \| u - u_N \|_{H^1(\Omega)}.

for $h$-refinement: gain of one factor $O(hM)$ (vs. $H^1(\Omega)$-estimates)

Is it important to assume $2$-regularity, Ass. 5.6.19 or merely a technical requirement of the theoretical approach?

*Example 5.6.22* ($L^2$-estimates on non-convex domain). *cf. Ex. 5.2.6*

Setting: $\Omega = ]-1, 1[^2 \setminus [0, 1[ \times ]-1, 0[), D \equiv 1, u(r, \varphi) = r^{2/3} \sin(2/3\varphi)$ (polar coordinates)

$\triangleright f = 0$, Dirichlet data $g = u|_{\partial \Omega}$.

Finite element Galerkin discretization and evaluations as in Ex. 5.6.15.
$L^2(\Omega)$-norm of discretization error on “L-shaped” domain ($\rightarrow p = 1$, $\leftarrow p = 2$)

Observation: For both $(p = 1, 2)$ $\blacksquare$ algebraic convergence $\|u - u_N\|_0 = O(N^{-2/3})$

Comparison with Ex. 5.2.6: for both linear and quadratic Lagrangian FEM

$\|u - u_N\|_{L^2(\Omega)} = O(N^{-2/3}) \iff \|u - u_N\|_{H^1(\Omega)} = O(N^{-1/3})$, 
that is, we again observe a doubling of the rate of convergence for the weaker norm.

No gain through the use of quadratic FEM, because of limited smoothness of both $u$ and dual solution $g_F$. For both the solution and the dual solution the gradient will have a singularity at $0$.

5.7 Discrete maximum principle

So far we have investigated the accuracy of finite element Galerkin solutions: we studied relevant norms $\|u - u_N\|$ of the discretization error.

Now new perspective: structure preservation by FEM
To what extent does the finite element solution $u_N$ inherit key structural properties of the solution $u$ of a 2nd-order scalar elliptic BVP?

This issue will be discussed for a special structural property of the solution of the linear 2nd-order elliptic BVP (inhomogeneous Dirichlet problem) in variational form (→ Sect. 2.8)

$$u \in \tilde{g} + H^1_0(\Omega): \quad a(u, v) := \int_\Omega \kappa \nabla u \cdot \nabla v \, dx = \int_\Omega f v \, dx \quad \forall v \in H^1_0(\Omega). \quad (5.7.1)$$

where $\tilde{g} \triangleq \text{offset function},$ extension of Dirichlet data $g \in C^0(\partial \Omega),$ see Sect. 2.3.1, (2.3.5),

$\kappa \triangleq \text{bounded and uniformly positive definite diffusion coefficient, see (2.5.4).}$

(5.7.1) $\Longleftrightarrow$ BVP (PDE-form)

$$- \text{div}(\kappa(\vec{x}) \nabla u) = f \quad \text{in } \Omega, \quad u = g \quad \text{on } \partial \Omega.$$  

Recall (→ Sect. 2.5): (5.7.1) models stationary temperature distribution in body, when temperature on its surface is prescribed by $g.$
Intuition: In the absence of heat sources maximal and minimal temperature attained on surface.

In the presence of a heat source \( f \geq 0 \) the temperature minimum will be attained on surface \( \partial \Omega \).

If \( f \leq 0 \) (heat sink), then the maximal temperature will be attained on the surface.

In fact this is a theorem, cf. Sect. [2.7].

**Theorem 5.7.2** (Maximum principle for 2nd-order elliptic BVP).
For \( u \in C^0(\Omega) \cap H^1(\Omega) \) holds the maximum principle

\[
- \text{div}(\kappa(\mathbf{x}) \text{grad} \ u) \geq 0 \implies \min_{x \in \partial \Omega} u(x) = \min_{x \in \Omega} u(x),
\]

\[
- \text{div}(\kappa(\mathbf{x}) \text{grad} \ u) \leq 0 \implies \max_{x \in \partial \Omega} u(x) = \max_{x \in \Omega} u(x).
\]
$\Delta u = 0$

Maximum/minimum on $\partial \Omega$

Proof. (for the case $-\text{div}(\kappa(\mathbf{x}) \text{grad} u) = 0$)

Sect. 2.1.3 $u$ solves quadratic minimization problem

$$u = \arg\min_{v \in H^1(\Omega)} \int_{\Omega} \kappa(\mathbf{x}) \|\text{grad} v(\mathbf{x})\|^2 \, d\mathbf{x} \quad \text{subject to} \quad v = g \text{ on } \partial \Omega \Omega.$$
If $u$ had a global maximum at $x^*$ in the interior of $\Omega$, that is

$$\exists \delta > 0: \quad u(x^*) \geq \max_{x \in \partial \Omega} u(x) + \delta.$$ 

Now “chop off” the maximum and define

$$w(x) := \min\{u(x), u(x^*) - \delta\}, \quad x \in \Omega.$$
Obviously, \( w \in C^0(\Omega) \), and as a continuous function which is piecewise in \( H^1 \) the function \( w \) will
also belong to $H^1(\Omega)$. However

$$\int_{\Omega} \kappa(x) \| \text{grad} \, w(x) \|^2 \, dx < \int_{\Omega} \kappa(x) \| \text{grad} \, u(x) \|^2 \, dx,$$

which contradicts the definition of $u$ as the global minimizer of the quadratic energy functional. \[ \square \]

Now we consider a finite element Galerkin discretization of (5.7.1) by means of linear Lagrangian finite elements (→ Sect. 3.4), using offset functions supported near $\partial \Omega$ as explained in Sect. 3.5.5.

$\Rightarrow$ finite element Galerkin solution $u_N \in S_1^0(\mathcal{M}) \subset C^0(\overline{\Omega})$

**Issue:** does $u_N$ satisfy a **maximum principle**, that is, can we conclude

$$f \geq 0 \implies \min_{x \in \partial \Omega} u_N(x) = \min_{x \in \Omega} u_N(x),$$

$$f \leq 0 \implies \max_{x \in \partial \Omega} u_N(x) = \max_{x \in \Omega} u_N(x)?$$

**Example 5.7.4** (Maximum principle for finite difference discretization).
Recall from Sect. 4.1: finite difference discretization of

\[-\Delta u = 0 \quad \text{in } \Omega := ]0, 1[^2, \quad u = g \quad \text{on } \partial \Omega,
\]
on an $M \times M$ tensor product mesh

\[M := \{[(i - 1)h, ih] \times [(j - 1)h, jh], \ 1 \leq i, j \leq M\}, \quad M \in \mathbb{N}.
\]

Unkowns in the finite difference method: $\mu_{ij} \approx u((ih, jh)^T), \ 1 \leq i, j \leq M - 1.$

Unkowns are solutions of a linear system of equations, see (4.1.2)

\[
\frac{1}{h^2}(4\mu_{i,j} - \mu_{i-1,j} - \mu_{i+1,j} - \mu_{i,j-1} - \mu_{i,j+1}) = 0, \quad 1 \leq i, j \leq M - 1,
\]

where values corresponding to points on the boundary are gleamed from $g$:

\[
\mu_{0,j} := g(0, hj), \quad \mu_{M,j} := g(1, hj), \quad \mu_{i,0} := g(hi, 0), \quad \mu_{i,M} := g(hi, 1), \quad 1 \leq i, j < M.
\]
The finite difference solution \((\mu_{i,j})_{1 \leq i,j < M}\) will attain its maximal value somewhere:

\[ \exists n, m \in \{1, \ldots, M - 1\}: \quad \mu_{n,m} = \mu_{\max} := \max_{0 \leq i,j \leq M} \mu_{i,j}. \]
Assume: \((nh, mh)^T\) in the interior \(\iff 1 \leq n, m < M\)

Be aware of the following two facts:

\[
\mu_{n-1,m}, \mu_{n+1,m}, \mu_{n,m-1}, \mu_{n,m+1} \leq \mu_{n,m},
\]
\[
\mu_{n,m} = \frac{1}{4}(\mu_{n-1,m} + \mu_{n+1,m} + \mu_{n,m-1} + \mu_{n,m+1}) \quad \text{(average!)}.
\]

\[
\Downarrow \leftarrow \text{"averaging argument"}
\]
\[
\mu_{n-1,m} = \mu_{n+1,m} = \mu_{n,m-1} = \mu_{n,m+1} = \mu_{n,m}!\]

(5.7.6) (5.7.7) (5.7.8)

The same argument can now target the neighboring grid points \(((n-1)h, mh)^T, ((n+1)h, mh)^T, (nh, (m-1)h)^T, (nh, (m+1)h)^T\). By induction we find:

\[
\mu_{i,j} = \mu_{\max} \quad \forall 0 \leq i, j \leq M,
\]

that is, the finite difference solution has to be \textit{constant}!

The finite difference solution can attain its maximum in the interior only in the case of constant boundary data \(g\)!

Maximum principle satisfied for \(f = 0\)!
Now we try to generalize the considerations of the previous example to the discretization by means of linear Lagrangian finite elements on a triangular mesh (of a polygonal domain $\Omega \subset \mathbb{R}^2$) see Sect. 3.2.

$\widetilde{A} \in \mathbb{R}^{M \times M} \doteq S_1^0(\mathcal{M})$ - Galerkin matrix for $a$ from (5.7.1) ($M := \#\mathcal{V}(\mathcal{M})$)

Row of this matrix connects all values $\mu_j = u_N(x^j)$ of Galerkin solution $u_N \in S_1^0(\mathcal{M})$ according to

$$(\widetilde{A})_{ii}\mu_i + \sum_{j \neq i} (\widetilde{A})_{ij}\mu_j = (\varphi)_i, \quad x^i \text{ interior node},$$

where $\mu_j := g(x^j)$ for $x^j \in \partial\Omega$.

The above averaging argument from Ex. 5.7.4 carries over, if the entries of $\widetilde{A}$ satisfy the following conditions:
\begin{itemize}
\item \((\tilde{A})_{ii} > 0\) (positive diagonal), \hspace{1cm} (5.7.9)
\item \((\tilde{A})_{ij} \leq 0\) for \(j \neq i\) (non-positive off-diagonal entries), (5.7.10)
\item \(\sum_j (\tilde{A})_{ij} = 0\), if \(x^i\) is interior node. \hspace{1cm} (5.7.11)
\end{itemize}

(Recall [18, Def. 2.7.7]: matrix \(\tilde{A}\) satisfying (5.7.9)–(5.7.11) is \textbf{diagonally dominant}.)

\begin{itemize}
\item \textbf{averaging argument} \(\Rightarrow\quad u_N(x^i) = \max_{y \in \mathcal{V}(\mathcal{M})} u_N(y)\) can only hold for an interior node \(x^i\), if \(\mu_N = \text{const.}\).
\item \textbf{Since} \(u_N \in \mathcal{S}_1^0(\mathcal{M})\) \text{attains its extremal values at nodes of the mesh, the maximum principles holds for it in the case} \(f = 0\) \text{provided that (5.7.9)–(5.7.11)} \text{are satisfied.}
\end{itemize}

More general case \(f \leq 0\):

\(\phi_i = \int_{\Omega} f(x)b_{N}^i(x) \, dx \leq 0\), since \(b_{N}^i \geq 0\).

Then the averaging argument again rules out the existence of an interior maximum for an non-constant solution. The case \(f \geq 0\) follows similarly.
When will \((5.7.9)-(5.7.11)\) hold for \(S^0_1(\mathcal{M})\)-Galnerkin matrix?

First consider \(\kappa \equiv 1\), \(\iff -\Delta u = f\)
(The linear finite element discretization of this BVP was scrutinized in Sect. 3.2)

From formula \((3.2.10)\) for element matrix & assembly, see Fig. 74:

\[
(\tilde{A})_{ij} = -\cot \alpha - \cot \beta = -\frac{\sin(\alpha + \beta)}{\sin \alpha \sin \beta}.
\]

\[
(\tilde{A})_{ij} \leq 0 \iff \alpha + \beta < \pi.
\]

Moreover

\[
\sum_{x \in \mathcal{V}(\mathcal{M})} b_N^x = 1 \iff \sum_j (\tilde{A})_{ij} = 0 \quad (\iff (5.7.11)).
\]

The condition \((5.7.9) \iff (\tilde{A})_{ii} > 0\) is straightforward.
**Theorem 5.7.12** (Maximum principle for linear FE solution of Poisson equation).

The linear finite element solution of

\[-\Delta u = 0 \quad \text{in } \Omega \subset \mathbb{R}^2, \quad u = g \quad \text{on } \partial \Omega,\]

on a triangular mesh $\mathcal{M}$ satisfies the **maximum principle** (5.7.3), if $\mathcal{M}$ is a Delaunay triangulation.

**Remark 5.7.13** (Maximum principle for linear FE for 2nd-order elliptic BVPs).

For $S_1^0(\mathcal{M})$-Galerkin discretization of (5.7.1) on triangular mesh, the conditions (5.7.9)–(5.7.11) are fulfilled,

if all angles of triangles of $\mathcal{M} \leq \frac{\pi}{2}$.  

△
Remark 5.7.14 (Maximum principle for higher order Lagrangian FEM).

Even when using $p$-degree Lagrangian finite elements with nodal basis functions associated with interpolation nodes, see Sect. 3.4.1, the discrete maximum principle will fail to hold on any mesh for $p > 1$. \[\triangle\]
Now we study scalar linear partial differential equations for which one coordinate direction is special and identified with time and denoted by the independent variable $t$. The other coordinates are regarded as spatial coordinates and designated by $\mathbf{x} = (x_1, \ldots, x_d)^T$.

solution will be a “function of time and space”: $u = u(\mathbf{x}, t)$

The domain for such PDEs will have tensor product structure (tensor product of spatial domain and a bounded time interval):
Computational domain:
\[ \tilde{\Omega} := \Omega \times [0, T] \subset \mathbb{R}^{d+1} \]

- **Space-time cylinder**

\[ \Omega \subset \mathbb{R}^d \triangleq \text{spatial domain} \] (satisfying assumptions of Sect. 2.1.1)

\[ T > 0 \triangleq \text{final time} \]

On \( \Omega \times \{0\} \rightarrow \text{initial conditions} \),
on \( \partial \Omega \times [0, T] \rightarrow \text{(spatial) boundary conditions} \).

**PDE for** \( u(x, t) \)

+ initial conditions  + boundary conditions

= evolution problem
Note: No boundary conditions on $\Omega \times \{T\}$ ("final conditions") are prescribed: time is supposed to have a “direction” that governs the flow of information in the evolution problem.

Evolution problems (on bounded spatial domains) are also known as initial-boundary value problems (IBVP).

Remark 6.0.1 (Initial time).

Why do we always pick initial time $t = 0$?

The modelled physical systems will usually be time-invariant, so that we are free to shift time. Remember the analogous situation with autonomous ODE, see [18, Sect. 12.1].
6.1 Parabolic initial-boundary value problems

6.1.1 Heat equation

Sect. 2.5 treated *stationary* heat conduction: no change of temperature with time (temporal equilibrium)

Now we consider the evolution of a temperature distribution $u = u(x, t)$.

\[ \Omega \subset \mathbb{R}^d \] space occupied by solid body (bounded spatial computational domain),
\[ \kappa = \kappa(x) \] (spatially varying) heat conductivity ($[\kappa] = \frac{W}{Km}$),
\[ T > 0 \] final time for “observation period” $[0, T]$,
\[ u_0 : \Omega \mapsto \mathbb{R} \] initial temperature distribution in $\Omega$,
\[ g : \partial \Omega \times [0, T] \mapsto \mathbb{R} \] surface temperature, varying in space and time: $g = g(x, t)$,
\[ f : \Omega \times [0, T] \mapsto \mathbb{R} \] time-dependent heat source/sink ($[f] = \frac{W}{m^2}$): $f = f(x, t)$.

Goal: derive PDE governing *transient* heat conduction.
Conservation of energy:

\[
\frac{d}{dt} \int_V \rho u \, d\mathbf{x} + \int_{\partial V} \mathbf{j} \cdot \mathbf{n} \, dS = \int_V f \, d\mathbf{x}
\]

for all “control volumes” \( V \) (6.1.1)

energy stored in \( V \)  
power flux through \( \partial V \)  
heat generation in \( V \)

\( \rho = \rho(\mathbf{x}) \): (spatially varying) heat capacity \([\rho] = JK^{-1}\), uniformly positive, cf. (2.5.4).

As in Sect. 2.5, now apply Gauss’ Theorem Thm. 2.4.5 to the power flux integral in (6.1.1). This converts the surface integral to a volume integral over \( \text{div} \, \mathbf{j} \) and we get

\[
\frac{d}{dt} \int_V \rho u \, d\mathbf{x} + \int_V \text{div} \, \mathbf{j} \, d\mathbf{x} = \int_V f \, d\mathbf{x}
\]

for all “control volumes” \( V \)

Now appeal to another version of the fundamental lemma of the calculus of variations, see Lemma [2.4.10], this time involving piecewise constant test functions.

Local form of energy balance law (Heat equation)

\[
\frac{\partial}{\partial t}(\rho u)(\mathbf{x}, t) + (\text{div}_\mathbf{x} \mathbf{j})(\mathbf{x}, t) = f(\mathbf{x}, t) \quad \text{in} \quad \tilde{\Omega}.
\]

(6.1.2)
The heat flux is linked to temperature variations by Fourier's law:

$$j(x) = -\kappa(x) \text{grad } u(x) , \quad x \in \Omega. \quad \text{(2.5.3)}$$

From here we let all differential operators like \text{grad} and \text{div} act on the spatial independent variable $x$. As earlier, the independent variables $x$ and $t$ will be omitted frequently. Watch out!

Now, plug (2.5.3) into (6.1.2).

$$\frac{\partial}{\partial t}(\rho u) - \text{div}(\kappa(x) \text{grad } u) = f \quad \text{in } \tilde{\Omega} := \Omega \times [0, T[. \quad \text{(6.1.3)}$$

+ Dirichlet boundary conditions (fixed surface temperature) on $\partial\Omega \times ]0, T[$:

$$u(x, t) = g(x, t) \quad \text{for } (x, t) \in \partial\Omega \times ]0, T[. \quad \text{(6.1.4)}$$

+ initial conditions for $t = 0$:

$$u(x, 0) = u_0(x) \quad \text{for all } x \in \Omega. \quad \text{(6.1.5)}$$
Terminology: (6.1.2) & (6.1.4) & (6.1.5) is a specimen of a 2nd-order parabolic initial-boundary value problem.

Remark 6.1.6 (Compatible boundary and initial data).

Natural regularity requirements for Dirichlet data $g$:

$g$ continuous in time and space

Natural compatibility requirement at initial time and $u_0 \in C^0(\bar{\Omega})$

$g(x, 0) = u_0(x) \quad \forall x \in \partial \Omega$. 

△
Remark 6.1.7 (Boundary conditions for 2nd-order parabolic IBVPs).

Physical intuition for transient heat conduction:

On \( \partial \Omega \times ]0, T[ \) we can impose any of the boundary conditions discussed in Sect. 2.6:

- **Dirichlet boundary conditions** \( u(x, t) = g(x, t) \), see (6.1.4) (fixed surface temperature),
- **Neumann boundary conditions** \( j(x, t) \cdot n = -h(x, t) \) (fixed heat flux through surface),
- **radiation boundary conditions** \( j(x, t) \cdot n = \Psi(u(x, t)) \),

and any combination of these as discussed in Ex. 2.6.7, yet, *only one* of them at any part of \( \partial \Omega \times ]0, T[ \), see Rem. 2.6.6.

### 6.1.2 Spatial variational formulation
Now we study the linear 2nd-order parabolic initial-boundary value problem with pure Dirichlet boundary conditions, introduced in the preceding section:

$$\frac{d}{dt}(\rho u) - \operatorname{div}(\kappa(x) \nabla u) = f \quad \text{in} \quad \tilde{\Omega} := \Omega \times [0, T],$$  \hspace{1cm} (6.1.3)

$$u(x, t) = g(x, t) \quad \text{for} \quad (x, t) \in \partial \Omega \times [0, T],$$  \hspace{1cm} (6.1.4)

$$u(x, 0) = u_0(x) \quad \text{for all} \quad x \in \Omega.$$  \hspace{1cm} (6.1.5)

Assume: Homogeneous Dirichlet boundary conditions $g = 0$

The general case can be reduced to this by using the offset function trick, see Sect. 3.5.5, and solve the parabolic initial-boundary value problem for $w(x, t) := u(x, t) - \tilde{g}(x, t)$, where $\tilde{g}(\cdot, t)$ is an extension of the Dirichlet data $g$ to $\tilde{\Omega}$. Then $w$ will satisfy homogeneous Dirichlet boundary conditions and solve an evolution equation with a modified source function $\tilde{f}(x, t)$.

Now we pursue the formal derivation of the \textit{spatial} variational formulation of (6.1.3)–(6.1.4).

The steps completely mirror those discussed in Sect. 2.8

\begin{align*}
\text{STEP 1:} & \quad \text{test PDE with functions } v \in H^1_0(\Omega) \\
& \text{(do not test, where the solution is known, that is, on the boundary } \partial \Omega )
\end{align*}
Note: test function does *not depend on time*: $v = v(x)$!

**STEP 2:** *integrate over domain* $\Omega$

**STEP 3:** *perform integration by parts in space*

(by using Green's first formula, Thm. 2.4.7)

**STEP 4:** [optional] *incorporate boundary conditions into boundary terms*

For the concrete PDE (6.1.3) and boundary conditions (6.1.4) refer to Ex. 2.8.1, for more general boundary conditions to Ex. 2.8.5.
Spatial variational form of (6.1.3)–(6.1.4): seek \( t \in ]0, T[ \mapsto u(t) \in H^1_0(\Omega) \)

\[
\int_\Omega \rho(x) \dot{u}(t)v \, dx + \int_\Omega \kappa(x) \nabla u(t) \cdot \nabla v \, dx = \int_\Omega f(x, t)v(x) \, dx \quad \forall v \in H^1_0(\Omega),
\]

(6.1.8)

\( u(0) = u_0 \in H^1_0(\Omega) \).

(6.1.9)

Be aware: \( u(t) \) = function space (\( H^1_0(\Omega) \))-valued function on \( ]0, T[ \).

Also note that \( \nabla \) acts on the spatial independent variables that are suppressed in the notation \( u(t) \).

\( \nabla \) Notation: \( \dot{u}(t) = \frac{\partial u}{\partial t}(t) \) = (partial) derivative w.r.t. time.

Shorthand notation (with obvious correspondences):

\[
t \in ]0, T[ \mapsto u(t) \in V_0 : \quad \begin{cases} 
m(\dot{u}(t), v) + a(u(t), v) = \ell(t)(v) & \forall v \in V_0, \\
u(0) = u_0 \in V_0. \end{cases}
\]
Again, here $\ell(t)$ denotes a linear form valued function on $]0, T]$. 

Concretely:

$$m(u, v) := \int_{\Omega} \rho(x) \dot{u}(t) v \, dx, \quad u, v \in H^1_0(\Omega),$$

$$a(u, v) := \int_{\Omega} \kappa(x) \text{grad} u(t) \cdot \text{grad} v \, dx, \quad u, v \in H^1_0(\Omega),$$

$$\ell(t)(v) := \int_{\Omega} f(x, t) v(x) \, dx, \quad v \in H^1_0(\Omega).$$

Note that both $m$ and $a$ are symmetric, positive definite bilinear forms ($\rightarrow$ Def. 2.1.25).

Equivalent formulation, since the bilinear form $m$ does not depend on time:
\[ t \in ]0, T[ \mapsto u(t) \in V_0 : \begin{cases} \frac{d}{dt} m(u(t), v) + a(u(t), v) = \ell(t)(v) & \forall v \in V_0, \\ u(0) = u_0 \in V_0. \end{cases} \] (6.1.11)

Now we are concerned with the stability of parabolic evolution problems: We investigate whether \( \|u\|_{H^1(\Omega)} \) stays bounded for all times in the case \( f \equiv 0 \).

For the sake of simplicity: consider \( \rho \equiv 1 \) and \( \kappa \equiv 1 \)
(General case is not more difficult, because both \( \rho \) and \( \kappa \) are bounded and uniformly positive, see (2.5.4).)

By the first Poincaré-Friedrichs inequality Thm. 2.2.16

\[ \exists \gamma > 0: \ |v|^2_{H^1(\Omega)} \geq \gamma \|v\|^2_{L^2(\Omega)} \quad \forall v \in H^1_0(\Omega). \] (6.1.12)

In fact, Thm. 2.2.16 reveals \( \gamma = \text{diam}(\Omega)^{-2} \).
Remark 6.1.13 (Differentiating bilinear forms with time-dependent arguments).

Consider (temporally) smooth $u : [0, T] \mapsto V_0$, $v : [0, T] \mapsto V_0$ and a symmetric bilinear form $b : V_0 \times V_0 \mapsto \mathbb{R}$.

What is $\frac{d}{dt} b(u(t), v(t))$?

Formal Taylor expansion:

$$b(u(t + \tau), v(t + \tau)) = b(u(t) + \dot{u}(t)\tau + O(\tau^2), v(t) + \dot{v}(t)\tau + O(\tau^2))$$

$$= b(u(t), v(t)) + \tau(b(\dot{u}(t), v(t)) + b(u(t), \dot{v}(t))) + O(\tau^2).$$

This is a general product rule, see [18, Eq. 4.4.5].
Lemma 6.1.14 (Decay of solutions of parabolic evolutions).
For \( \rho \equiv 1, \kappa \equiv 1, \) and \( f \equiv 0 \) the solution \( u(t) \) of (6.1.8) satisfies
\[
\|u(t)\|_{L^2(\Omega)} \leq e^{-\gamma t} \|u_0\|_{L^2(\Omega)}, \quad |u(t)|_{H^1(\Omega)} \leq e^{-\gamma t} |u_0|_{H^1(\Omega)} \quad \forall t \in ]0, T[.
\]

Proof. Multiply the solution of the parabolic IBVP with an exponential weight function:
\[
w(t) := \exp(\gamma t) u(t) \in H^1_0(\Omega) \quad \Rightarrow \quad \dot{w} := \frac{dw}{dt}(t) = \gamma w(t) + \exp(\gamma t) \frac{du}{dt}(t),
\]
solves the parabolic IBVP
\[
m(\dot{w}, v) + \tilde{a}(w, v) = 0 \quad \forall v \in V, \quad w(0) = u_0,
\]
with \( \tilde{a}(w, v) = a(w, v) - \gamma m(w, v) \), \( \gamma \) from (6.1.12). To see this, use that \( u(t) \) solves (6.1.11) with \( f \equiv 0 \) (elementary calculation).

Note: \( (6.1.12) \quad \Rightarrow \quad \tilde{a}(v, v) \geq 0 \quad \forall v \in V \)

Exponential decay of \( \|\cdot\|_{L^2(\Omega)} \)-norm of solution:
\[
\frac{d}{dt} \frac{1}{2} \|w\|_{L^2(\Omega)}^2 = \frac{d}{dt} \frac{1}{2} m(w, w) = m(\dot{w}, w) = -\tilde{a}(w, w) \leq 0
\]
This confirms that $t \mapsto \|w\|_{L^2(\Omega)}(t)$ is a decreasing function, which involves

\[(6.1.17) \quad \Rightarrow \quad \|w(t)\|_{L^2(\Omega)} \leq \|w(0)\|_{L^2(\Omega)},\]

and the first assertion of the Lemma is evident. Next, we verify the exponential decay of $|\cdot|_{H^1(\Omega)}$-norm of solution by a similar trick:

\[
\frac{1}{2} \frac{d}{dt} \|w\|^2_{\tilde{a}} = \tilde{a}\left(\frac{d}{dt}w, w\right) = -m\left(\frac{d}{dt}w, \frac{d}{dt}w\right) \leq 0 \quad \Rightarrow \quad \|w(t)\|_{\tilde{a}} \leq \|w(0)\|_{\tilde{a}},
\]

\[
|w(t)|^2_{H^1(\Omega)} \leq |w(0)|^2_{H^1(\Omega)} - \gamma\left(\|w(0)\|_{L^2(\Omega)}^2 - \|w(t)\|_{L^2(\Omega)}^2\right) \geq 0 \quad \text{by (6.1.17)}.
\]

Exponential decrease of energy during parabolic evolution without excitation

(“Parabolic evolutions dissipate energy”)

6.1.3 Method of lines
Idea: Apply Galerkin discretization (→ Sect. 3.1) to abstract linear parabolic variational problem (6.1.11).

\[ t \in ]0, T[ \mapsto u(t) \in V_0 : \begin{cases}
 m(\dot{u}(t), v) + a(u(t), v) = \ell(t)(v) & \forall v \in V_0, \\
 u(0) = u_0 \in V_0 .
\end{cases} \tag{6.1.11} \]

1st step: replace \( V_0 \) with a finite dimensional subspace \( V_{0,N}, N := \text{dim} V_{0,N} < \infty \)

\[ t \in ]0, T[ \mapsto u(t) \in V_{0,N} : \begin{cases}
 m(\dot{u}_N(t), v_N) + a(u_N(t), v_N) = \ell(t)(v_N) & \forall v_N \in V_{0,N} , \\
 u_N(0) = \text{projection/interpolant of } u_0 \text{ in } V_{0,N} .
\end{cases} \tag{6.1.18} \]

2nd step: introduce (ordered) basis \( \mathcal{B}_N := \{b_1^N, \ldots, b_N^N\} \) of \( V_{0,N} \)

\[ (6.1.18) \quad \Rightarrow \quad \begin{cases}
 M \left\{ \frac{d}{dt} \tilde{\mu}(t) \right\} + A \tilde{\mu}(t) = \varphi(t) & \text{for } 0 < t < T , \\
 \tilde{\mu}(0) = \tilde{\mu}_0 .
\end{cases} \tag{6.1.19} \]
s.p.d. stiffness matrix $A \in \mathbb{R}^{N,N}$, $(A)_{ij} := a(b^j_N, b^i_N)$ (independent of time),

s.p.d. mass matrix $M \in \mathbb{R}^{N,N}$, $(M)_{ij} := m(b^j_N, b^i_N)$ (independent of time),

source (load) vector $\bar{\phi}(t) \in \mathbb{R}^N$, $(\bar{\phi}(t))_i := \ell(t)(b^i_N)$ (time-dependent),

$\bar{\mu}_0 \triangleq$ coefficient vector of a projection of $u_0$ onto $V_{0,N}$.

For the concrete linear parabolic evolution problem (6.1.8)–(6.1.9) and spatial finite element discretization based on a finite element trial/test space $V_{0,N} \subset H^1(\Omega)$ we can compute

- the mass matrix $M$ as the Galerkin matrix for the bilinear form $(u, v) \mapsto \int_\Omega \rho(x) uv \, dx$, $u, v \in L^2(\Omega)$,

- the stiffness matrix $A$ as Galerkin matrix arising from the bilinear form $(u, v) \mapsto \int_\Omega \kappa(x) \text{grad} \, u \cdot \text{grad} \, v \, dx$, $u, v \in H^1(\Omega)$.

The calculations are explained in Sects. 3.5.3 and 3.5.4 and may involve numerical quadrature.
(6.1.19) is an ordinary differential equation (ODE) for \( t \mapsto \vec{\mu}(t) \in \mathbb{R}^N \)

Conversion (6.1.11) \( \rightarrow \) (6.1.19) through Galerkin discretization in space only is known as the method of lines.

(6.1.19) \( \hat{=} \) A semi-discrete evolution problem

Discretized in space \( \leftrightarrow \) but still continuous in time

Remark 6.1.20 (Spatial discretization options).

Beside the Galerkin approach any other method for spatial discretization of 2nd-order elliptic BVPs can be used in the context of the method of lines: the matrices \( A, M \) may also be generated by finite differences (\( \rightarrow \) Sect. 4.1), finite volume methods (\( \rightarrow \) Sect. 4.2), or collocation methods (\( \rightarrow \) Sect. 1.5.2).
6.1.4 Timestepping

For implementation we need a fully discrete evolution problem. This requires additional discretization in time:

\[
\text{semi-discrete evolution problem (6.1.19) } \quad + \text{ timestepping } \quad \rightarrow \quad \text{fully discrete evolution problem}
\]

Benefit of method of lines: we can apply already known integrators for initial value problems for ODEs to (6.1.19).

First, refresh central concepts from numerical integration of initial value problems for ODEs, see [18, Ch. 12], [18, Ch. 13]:

- single step methods of order \( p \), see [18, Def. 12.2.17] and [18, Thm. 12.3],
- explicit and implicit Runge-Kutta single step methods, see [18, Sect. 12.4], [18, Sect. ??], encoded by Butcher scheme [18, Eq. 12.4.10], [18, Eq 13.3.8].
- the notion of a stiff problem (\( \rightarrow \) [18, Notion 13.2.25]),
- the definition of the stability function of a single step method, see [18, Thm. 13.3.9],
- the concept of L-stability [18, Def 13.3.11] and how to verify it for Runge-Kutta methods.
6.1.4.1 Single step methods

Recall: single step methods (→ [18, Def. 12.2.17])

- are based on a temporal mesh \( \{0 = t_0 < t_1 < \ldots < t_{M-1} < t_M := T \} \) (with local timestep size \( \tau_j = t_j - t_{j-1} \)),
- compute sequence \( (\bar{\mu}^{(j)})_{j=0}^M \) of approximations \( \bar{\mu}^{(j)} \approx \mu(t_j) \) to the solution of (6.1.19) at the nodes of the temporal mesh according to

\[
\bar{\mu}^{(j)} := \Psi^{t_{j-1}, t_j}(\bar{\mu}^{(j-1)}) := \Psi(t_{j-1}, t_j, \bar{\mu}^{(j-1)}), \quad j = 1, \ldots, M,
\]

where \( \Psi \) is the discrete evolution defining the single step method, see [18, Def. 12.2.17].

Example 6.1.21 (Euler timestepping). → [18, Sect. 12.2]
We target the initial value problem

\[
M \left\{ \frac{d}{dt} \tilde{\mu}(t) \right\} + A \tilde{\mu}(t) = \vec{\phi}(t) \quad \text{for } 0 < t < T, \tag{6.1.19}
\]

\[
\tilde{\mu}(0) = \tilde{\mu}_0.
\]

Explicit Euler method [18, Eq. 12.2.4]: replace \( \frac{d}{dt} \) in (6.1.19) with forward difference quotient, see [18, Rem. 12.2.5]:

\[
M \tilde{\mu}^{(j)} = M \tilde{\mu}^{(j-1)} - \tau_j (A \tilde{\mu}^{(j-1)} - \vec{\phi}(t_{j-1})) , \quad j = 1, \ldots, M - 1 . \tag{6.1.22}
\]

Implicit Euler method [18, Eq. 12.2.10]: replace \( \frac{d}{dt} \) in (6.1.19) with backward difference quotient

\[
M \tilde{\mu}^{(j)} = M \tilde{\mu}^{(j-1)} - \tau_j (A \tilde{\mu}^{(j)} - \vec{\phi}(t_j)) , \quad j = 1, \ldots, M - 1 . \tag{6.1.23}
\]

Note that both (6.1.22) and (6.1.23) require the solution of a linear system of equations in each step

\[
(6.1.22): \quad \tilde{\mu}^{(j)} = \tilde{\mu}^{(j-1)} + \tau_j M^{-1}(\vec{\phi}(t_{j-1}) - A \tilde{\mu}^{(j-1)}) ,
\]

\[
(6.1.23): \quad \tilde{\mu}^{(j)} = (\tau_j A + M)^{-1} \left( M \mu^{(j-1)} + \tau_j \vec{\phi}(t_j) \right) .
\]

Recall [18, Sect. 12.3]: both Euler method are of first order. \(\diamond\)
Example 6.1.24 (Crank-Nicolson timestepping).

Crank-Nicolson method = implicit midpoint rule: replace \( \frac{d}{dt} \) in (6.1.19) with symmetric difference quotient and average right hand side:

\[
M \left\{ \frac{d}{dt} \tilde{\mu}(t) \right\} + A \tilde{\mu}(t) = \tilde{\varphi}(t)
\]

\[
\downarrow
\]

\[
M \frac{\tilde{\mu}^{(j)} - \tilde{\mu}^{(j-1)}}{\tau} = -\frac{1}{2} A \left( \tilde{\mu}^{(j)} + \tilde{\mu}^{(j-1)} \right) + \frac{1}{2} (\tilde{\varphi}(t_j) + \tilde{\varphi}(t_{j-1})).
\] (6.1.25)

This yields a method that is 2nd-order consistent.

Generalization of Euler methods:

Runge-Kutta single step methods → [18, Sect. 12.4], [18, Sect. 13.3]
**Definition 6.1.26 (General Runge-Kutta method).** \[ [18, \text{Def. 13.3.7}] \]

For coefficients \( b_i, a_{ij} \in \mathbb{R}, c_i := \sum_{j=1}^{s} a_{ij}, i, j = 1, \ldots, s, s \in \mathbb{N} \), the discrete evolution \( \Psi^{s,t} \)

of an \( s \)-stage Runge-Kutta single step method (RK-SSM) for the ODE \( \dot{y} = f(t, y) \), is defined by

\[
\begin{align*}
    k_i & := f(t + c_i \tau, y + \tau \sum_{j=1}^{s} a_{ij} k_j), \quad i = 1, \ldots, s, \\
    \Psi^{t,t+\tau} y & := y + \tau \sum_{i=1}^{s} b_i k_i.
\end{align*}
\]

The \( k_i \in \mathbb{R}^d \) are called increments.

**Shorthand notation for \( s \)-stage Runge-Kutta methods:**

**Butcher scheme** \[ [18, \text{Eq. 13.3.8}] \]

\[
\begin{bmatrix}
    c_1 & a_{11} & a_{12} & \cdots & \cdots & a_{1s} \\
    c_2 & a_{21} & \ddots & & & a_{2s} \\
    \vdots & \vdots & & \ddots & & \vdots \\
    c_s & a_{s1} & \ddots & & & a_{ss} \\
    b_1 & b_2 & \cdots & \cdots & \cdots & b_s
\end{bmatrix}
\]

, \( c, b \in \mathbb{R}^s \), \( \mathcal{A} \in \mathbb{R}^{s \times s} \) \hspace{1cm} (6.1.27)

**Concretely for linear parabolic evolution:** application of \( s \)-stage Runge-Kutta method to

\[
M \left\{ \frac{d}{dt} \tilde{\mu}(t) \right\} + A \tilde{\mu}(t) = \tilde{\varphi}(t) \quad \Leftrightarrow \quad \dot{\tilde{\mu}} = \underbrace{M^{-1} (\tilde{\varphi}(t) - A \tilde{\mu}(t))}_{=f(t, \tilde{\mu})}.
\]

(6.1.19)
Then simply plug this into the formulas of Def. 6.1.26.

**Timestepping scheme for (6.1.19):** compute \( \vec{\mu}^{(j+1)} \) from \( \vec{\mu}^{(j)} \) through

\[
\vec{\kappa}_i \in \mathbb{R}^N: \quad M \vec{\kappa}_i + \sum_{m=1}^{s} \tau a_{im} A \vec{\kappa}_m = \bar{\varphi}(t_j + c_i \tau) - A \vec{\mu}^{(j)}, \quad i = 1, \ldots, s, \tag{6.1.28}
\]

\[
\vec{\mu}^{(j+1)} = \vec{\mu}^{(j)} + \tau \sum_{m=1}^{s} \vec{\kappa}_m b_m. \tag{6.1.29}
\]

**Note:** For an implicit RK-method (6.1.28) is a linear system of equations of size \( Ns \).

### 6.1.4.2 Stability

**Example 6.1.30 (Convergence of Euler timestepping).**

Parabolic evolution problem in one spatial dimension (IBVP):

\[
\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} \quad \text{in } [0, 1] \times [0, 1], \tag{6.1.31}
\]
\[ u(t, 0) = u(t, 1) = 0 \quad \text{for} \ 0 \leq t \leq 1 , \quad u(0, x) = \sin(\pi x) \quad \text{for} \ 0 < x < 1 . \]  

(6.1.32)

exact solution \[ u(t, x) = \exp(-\pi^2 t) \sin(\pi x) . \]  

(6.1.33)

Spatial finite element Galerkin discretization by means of linear finite elements \( (V_0, N = S^0_{1,0}(\mathcal{M})) \) on equidistant mesh \( \mathcal{M} \) with meshwidth \( h := \frac{1}{N} \) \( \rightarrow \) Sect. 1.5.1.2.

\( u_{N,0} := l_1 u_0 \) by linear interpolation on \( \mathcal{M} \), see Sect. 5.3.1.

Timestepping by explicit and implicit Euler method (6.1.22), (6.1.23) with uniform timestep \( \tau := \frac{1}{M} \).

Galerkin matrices, see (1.5.60):

\[
A = \frac{1}{h} \begin{pmatrix}
2 & -1 & 0 & 0 \\
-1 & 2 & -1 & 0 \\
0 & 0 & \ddots & \ddots \\
\end{pmatrix}, \quad M = \frac{h}{6} \begin{pmatrix}
4 & 1 & 0 & 0 \\
1 & 4 & 1 & 0 \\
0 & \ddots & \ddots & \ddots \\
\end{pmatrix}.
\]
function [errex, errimp] = sinevl(N, M, u)
  % Solve fully discrete two-point parabolic evolution problem \( (6.1.31) \)
  % in \([0,1] \times [0,1]\). Use both explicit and implicit Euler method for timestepping
  % N: number of spatial grid cells
  % M: number of timesteps
  % u: handle of type @(t,x) to exact solution

  if (nargin < 3), u = @(t,x) (exp(-pi^2)*t).*sin(pi*x)); end % Exact solution

  h = 1/N; tau = 1/M; % Spatial and temporal meshwidth
  x = h:h:1-h; % Spatial grid, interior points

  % Finite element stiffness and mass matrix
  Amat = gallery('tridiag',N-1,-1,2,-1)/h;
  Mmat = h/6*gallery('tridiag',N-1,1,4,1);
  Xmat = Mmat+tau*Amat;

  mu0 = u(0,x)'; % Discrete initial value
  mui = mu0; mue = mu0;

  % Timestepping
erre = 0; erri = 0;

for k=1:M
    mue = mue - tau*(Mmat\(Amat\*mue)); % explicit Euler step
    mui = Xmat\(Mmat\*mui); % implicit Euler step
    utk = u(k*tau,x)';
    erre = erre + norm(mue-utk)^2; % Computation of error norm
    erri = erri + norm(mui-utk)^2;
end

errex = sqrt(erre*h*tau);
errimp = sqrt(erri*h*tau);

Evaluation of approximate space-time $L^2$-norm of the discretization error:

$$
\text{err}^2 := h\tau \cdot \sum_{j=1}^{M} \sum_{i=1}^{N-1} |u(t_j, x_i) - \mu_{i}^{(j)}|^2 .
$$ (6.1.35)

Error norm for explicit Euler timestepping:
<table>
<thead>
<tr>
<th>N \ M</th>
<th>50</th>
<th>100</th>
<th>200</th>
<th>400</th>
<th>800</th>
<th>1600</th>
<th>3200</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Inf</td>
<td>0.009479</td>
<td>0.006523</td>
<td>0.005080</td>
<td>0.004366</td>
<td>0.004011</td>
<td>0.003834</td>
</tr>
<tr>
<td>10</td>
<td>Inf</td>
<td>Inf</td>
<td>Inf</td>
<td>Inf</td>
<td>0.001623</td>
<td>0.001272</td>
<td>0.001097</td>
</tr>
<tr>
<td>20</td>
<td>Inf</td>
<td>Inf</td>
<td>Inf</td>
<td>Inf</td>
<td>Inf</td>
<td>Inf</td>
<td>0.000405</td>
</tr>
<tr>
<td>40</td>
<td>Inf</td>
<td>Inf</td>
<td>Inf</td>
<td>Inf</td>
<td>Inf</td>
<td>Inf</td>
<td>Inf</td>
</tr>
<tr>
<td>80</td>
<td>Inf</td>
<td>Inf</td>
<td>Inf</td>
<td>Inf</td>
<td>Inf</td>
<td>Inf</td>
<td>Inf</td>
</tr>
<tr>
<td>160</td>
<td>Inf</td>
<td>Inf</td>
<td>Inf</td>
<td>Inf</td>
<td>Inf</td>
<td>Inf</td>
<td>Inf</td>
</tr>
<tr>
<td>320</td>
<td>Inf</td>
<td>Inf</td>
<td>Inf</td>
<td>Inf</td>
<td>Inf</td>
<td>Inf</td>
<td>Inf</td>
</tr>
</tbody>
</table>

Error norm for implicit Euler timestepping:

<table>
<thead>
<tr>
<th>N \ M</th>
<th>50</th>
<th>100</th>
<th>200</th>
<th>400</th>
<th>800</th>
<th>1600</th>
<th>3200</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>0.007025</td>
<td>0.001828</td>
<td>0.000876</td>
<td>0.002257</td>
<td>0.002955</td>
<td>0.003306</td>
<td>0.003482</td>
</tr>
<tr>
<td>10</td>
<td>0.009641</td>
<td>0.004500</td>
<td>0.001826</td>
<td>0.000461</td>
<td>0.000228</td>
<td>0.000575</td>
<td>0.000749</td>
</tr>
<tr>
<td>20</td>
<td>0.010303</td>
<td>0.005175</td>
<td>0.002509</td>
<td>0.001149</td>
<td>0.000461</td>
<td>0.000116</td>
<td>0.000058</td>
</tr>
<tr>
<td>40</td>
<td>0.010469</td>
<td>0.005345</td>
<td>0.002681</td>
<td>0.001321</td>
<td>0.000634</td>
<td>0.000289</td>
<td>0.000116</td>
</tr>
<tr>
<td>80</td>
<td>0.010511</td>
<td>0.005387</td>
<td>0.002724</td>
<td>0.001364</td>
<td>0.000677</td>
<td>0.000332</td>
<td>0.000159</td>
</tr>
<tr>
<td>160</td>
<td>0.010521</td>
<td>0.005398</td>
<td>0.002734</td>
<td>0.001375</td>
<td>0.000688</td>
<td>0.000343</td>
<td>0.000170</td>
</tr>
<tr>
<td>320</td>
<td>0.010524</td>
<td>0.005400</td>
<td>0.002737</td>
<td>0.001378</td>
<td>0.000691</td>
<td>0.000346</td>
<td>0.000172</td>
</tr>
</tbody>
</table>
Explicit Euler timestepping: we observe a glaring *instability* (exponential blow-up) in case of *large timestep combined with fine mesh*.

Implicit Euler timestepping: no blow-up at any combination of spatial and temporal mesh width.

\textit{Example} 6.1.36 \texttt{(ode45} for discrete parabolic evolution\texttt{)}.

Same IBVP and spatial discretization as in Ex. \texttt{6.1.30}.

Adaptive Runge-Kutta timestepping by MATLAB standard integrator \texttt{ode45}.

Monitored:
- Number of timesteps as a function on spatial meshwidth $h$,
- discrete $L^2$-error (6.1.35).
function [Nsteps,err] = peode45(N,tol,u)
% Solving fully discrete two-point parabolic evolution problem (6.1.31) in \([0,1] \times [0,1]\) by means of adaptiv MATLAB standard Runge-Kutta integrator.
if (nargin < 3), u = @(t,x) \(\exp(-((\pi^2)*t)\ast\sin(\pi\ast x))\); end  % Exact solution

% Finite element stiffness and mass matrix, see Sect. 1.5.1.2
h = 1/N; % spatial meshwidth
Amat = gallery ('tridiag',N-1,-1,2,-1)/h;
Mmat = h/6\ast gallery ('tridiag',N-1,1,4,1);
x = h:h:1-h;  % Spatial grid, interior points

mu0 = u(0,x)'; % Discrete initial value
fun = @(t,muv) -(Mmat\(Amat\ast muv));  % right hand side of ODE

opts = odeset ('reltol',tol,'abstol',0.01\ast tol);
[t,mu] = ode45 (fun,[0,1],mu0,opts);

Nsteps = length(t);
[T,X] = meshgrid(t,x); err = norm(mu'-u(T,X),'fro');
Observations:

- **ode45**: dramatic increase of no. of timesteps for $h_M \to 0$ without gain in accuracy.
- Implicit Euler achieves better accuracy with only 100 equidistant timesteps!
This reminds us of the **stiff initial value problems** studied in [18, Thm. 13.2]:

**Notion 6.1.38** (Stiff IVP). \[ \rightarrow \text{[18, Notion 13.2.25]} \]

An initial value problem for an ODE is called **stiff**, if stability imposes much tighter timestep constraints on explicit single step methods than the accuracy requirements.

Admittedly, this is a fuzzy notion. Yet, it cannot be fleshed out on the abstract level, but has to be discussed for concrete evolution problem, which is done next.

Let us try to understand, why semi-discrete parabolic evolutions (6.1.19) arising from the method of lines lead to stiff initial value problems.

**Technique:**   **Diagonalization**, cf. [18, Eq. 13.2.10]
Recall the concept of a “square root” $M^{1/2}$ of an s.p.d. matrix $M$, see [18, Sect. 5.3])

\[ A, M \text{ symmetric positive definite } \Rightarrow \ M^{-1/2}AM^{-1/2} \text{ symmetric positive definite } . \]

[18, Cor. 6.1.9] \[ \exists \text{ orthogonal } T \in \mathbb{R}^{N,N}: \ T^\top M^{-1/2}AM^{-1/2}T = D := \text{diag}(\lambda_1, \ldots, \lambda_N) , \]

where the $\lambda_i > 0$ are generalized eigenvalues for $A\tilde{\xi} = \lambda M\tilde{\xi}$ ➤ $\lambda_i \geq \gamma$ for all $i$ ($\gamma$ is the constant introduced in (6.1.12)).

- Transformation ("diagonalization") of (6.1.19) based on substitution $\tilde{\eta} := T^\top M^{1/2} \tilde{\mu}$:

\[ (6.1.19) \quad \tilde{\eta} := T^\top M^{1/2} \tilde{\mu} \quad \Rightarrow \quad \frac{d}{dt}\tilde{\eta}(t) + D\tilde{\eta} = T^\top M^{-1/2}\varphi(t) . \] (6.1.39)

➤ Since $D$ is diagonal, (6.1.39) amounts to $N$ decoupled scalar ODEs (for eigencomponents $\eta_i$ of $\tilde{\mu}$).

Note:

for $\varphi \equiv 0, \lambda > 0$: \[ \eta_i(t) = \exp(-\lambda_i t)\eta_i(0) \to 0 \quad \text{for } t \to \infty \]

As in [18, Thm. 13.2.12] this transformation can be applied to the explicit Euler timestepping (6.1.22) (for $\varphi \equiv 0$, uniform timestep $\tau > 0$)

\[ \tilde{\mu}^{(j)} = \tilde{\mu}^{(j-1)} - \tau M^{-1}A\tilde{\mu}^{(j-1)} \quad \Rightarrow \quad \tilde{\eta}^{(j)} = \tilde{\eta}^{(j-1)} - \tau D\tilde{\eta}^{(j-1)} , \]
that is, the decoupling of eigencomponents carries over to the explicit Euler method: for $i = 1, \ldots, N$

$$\eta_i^{(j)} = \eta_i^{(j-1)} - \tau \lambda_i \eta_i^{(j-1)} \Rightarrow \eta_i^{(j)} = (1 - \tau \lambda_i)^j \eta_i^{(0)}.$$  \hfill (6.1.40)

\[|1 - \tau \lambda_i| < 1 \iff \lim_{j \to \infty} \eta_i^{(j)} = 0. \]  \hfill (6.1.41)

The condition $|1 - \tau \lambda_i| < 1$ enforces a timestep size constraint: $\tau < \frac{2}{\lambda_i}$ \hfill (6.1.42)

in order to achieve the qualitatively correct behavior $\lim_{j \to \infty} \eta_i^{(j)} = 0$ and to avoid blow-up $\lim_{j \to \infty} |\eta_i^{(j)}| = \infty$: the timestep size constraint \hfill (6.1.42) is necessary only for the sake of stability (not in order to guarantee a prescribed accuracy).

This accounts to the observed blow-ups in Ex. 6.1.30. On the other hand, adaptive stepsize control \hfill [18, Sect. 12.5] manages to ensure the timestep constraint, but the expense of prohibitively small timesteps that render the method grossly inefficient, if some of the $\lambda_i$ are large.

The next numerical demonstrations and Lemma show that $\lambda_{\text{max}} := \max_i \lambda_i$ will inevitably become huge for finite element discretization on fine meshes.
Example 6.1.43 (Behavior of generalized eigenvalues of \( A\vec{\mu} = \lambda M\vec{\mu} \)).

Bilinear forms associated with parabolic IBVP and homogeneous Dirichlet boundary conditions

\[
a(u, v) = \int_{\Omega} \nabla u \cdot \nabla v \, dx, \quad m(u, v) = \int_{\Omega} u(x)v(x) \, dx, \quad u, v \in H^1_0(\Omega).
\]

Linear finite element Galerkin discretization, see Sect. 1.5.1.2 for 1D, and Sect. 3.2 for 2D.

Numerical experiments in 1D & 2D:

- \( \Omega = ]0, 1[ \), equidistant meshes \( \rightarrow \) Ex. 6.1.30
- “disk domain” \( \Omega = \{ x \in \mathbb{R}^2 : \| x \| < 1 \} \), sequence of regularly refined meshes.

Monitored: largest and smallest generalized eigenvalue
% LehrFEM MATLAB script for computing Dirichlet eigenvalues of Laplacian
% on a unit disc domain.

GD_HANDLE = @(x,varargin)zeros(size(x,1),1); % Zero Dirichlet data
H0 = [.25 .2 .1 .05 .02 .01 0.005]'; % target mesh widths
NRef = length(H0); % Number of refinement steps

% Variables for mesh widths and eigenvalues
M_W = zeros(NRef,1); lmax = M_W; lmin = M_W;

% Main refinement loop
for iter = 1:NRef

% Set parameters for mesh
C = [0 0]; % Center of circle
R = 1; % Radius of circle
BBOX = [-1 -1; 1 1]; % Bounding box
DHANDLE = @dist_circ; % Signed distance function
HHANDLE = @h_uniform; % Element size function
FIXEDPOS = []; % Fixed boundary vertices of the mesh
DISP = 0; % Display flag
% Mesh generation
Mesh =
    init_Mesh(BBOX,H0(iter),DHANDLE,HHANDLE,FIXEDPOS,DISP,C,R);
Mesh = add_Edges(Mesh); % Provide edge information
Loc = get_BdEdges(Mesh); % Obtain indices of edges on ∂Ω
Mesh.BdFlags = zeros(size(Mesh.Edges,1),1);
Mesh.BdFlags(Loc) = -1; % Flag boundary edges
Mesh.ElemFlag = zeros(size(Mesh.Elements,1),1);
M_W(iter) = get_MeshWidth(Mesh); % Get mesh width

fprintf('Mesh on level %i: %i elements, h =
    %f\n',iter,size(Mesh,1),M_W(iter));

% Assemble stiffness matrix and mass matrix
A = assemMat_LFE(Mesh,@STIMA_Lapl_LFE,P7O6());
M = assemMat_LFE(Mesh,@MASS_LFE,P7O6());

% Incorporate Dirichlet boundary data (nothing to do here)
[U,FreeNodes] = assemDir_LFE(Mesh,-1,GD_HANDLE);
A = A(FreeNodes,FreeNodes);
M = M(FreeNodes,FreeNodes);

% Use MATLAB’s built-in eigs-function to compute the
% extremal eigenvalues, see [18, Sect. 6.4].
NEigen = 6;
d = eigs(A,M,NEigen,'sm'); lmin(iter) = min(d);
d = eigs(A,M,NEigen,'lm'); lmax(iter) = max(d);
end

figure; plot(M_W,lmin,'b--',M_W,lmax,'r-*'); grid on;
set(gca,'XScale','log','YScale','log','XDir','reverse');
title('
bf Eigenvalues of Laplacian on unit disc');
xlabel('{\bf mesh width h}','fontsize',14);
ylabel('{\bf generalized eigenvalues}','fontsize',14);
legend('{\lambda_{\min}}','{\lambda_{\max}}','Location','NorthWest');
p = polyfit(log(M_W),log(lmax),1);
add_Slope(gca,'east',p(1));

print -depsc2 '.../.../.../Slides/NPDEPics/geneigdisklfe.eps';
Observation:

- $\lambda_{\text{min}} := \min_i \lambda_i$ does hardly depend on the mesh width.
- $\lambda_{\text{max}} := \max_i \lambda_i$ displays a $O(h^{-2})$ growth as $h_M \to 0$.
Remark 6.1.45 (Spectrum of elliptic operators).

The observation made in Ex. 6.1.43 is not surprising!

To understand why, let us translate the generalized eigenproblem “back to the ODE/PDE level”:

\[ A \vec{\mu} = \lambda M \vec{\mu} \tag{6.1.46} \]

\[ u_N \in V_{0,N}: \quad a(u_N, v_N) = \lambda m(u_N, v_N) \quad \forall v_N \in V_{0,N}. \]

\[ u \in H^1_0(\Omega): \quad \int_{\Omega} \nabla u \cdot \nabla v \, dx = \lambda \int_{\Omega} u \cdot v \, dx \quad \forall v \in H^1_0(\Omega). \tag{6.1.47} \]

which is a so-called elliptic eigenvalue problem.
It is easily solved in 1D on $\Omega = ]0, 1[$:

$$
(6.1.47) \Rightarrow \frac{d^2 u}{dx^2}(x) = \lambda u(x), \quad 0 < x < 1, \quad u(0) = u(1) = 0.
$$

$$
\Rightarrow u_k(x) = \sin(k\pi x) \iff \lambda_k = (\pi k)^2, \quad k \in \mathbb{N}.
$$

Note that we find an infinite number of eigenfunctions and eigenvalues, parameterized by $k \in \mathbb{N}$. The eigenvalues tend to $\infty$ for $k \to \infty$:

$$
\lambda_k = O(k^2) \quad \text{for} \quad k \to \infty.
$$

Of course, (6.1.46) can have a finite number of eigenvectors only. Crudely speaking, they correspond to those eigenfunctions $u_k(x) = \sin(k\pi x)$ that can be resolved by the mesh (if $u_k$ “oscillates too much”, then it cannot be represented on a grid). These are the first $N$ so that we find in 1D for an equidistant mesh

$$
\lambda_{\max} = O(N^2) = O(h^{-2}_M).
$$

This is heuristics, but the following Lemma will a precise statement.
Lemma 6.1.48 (Behavior of of generalized eigenvalues).

Let $\mathcal{M}$ be a simplicial mesh and $A$, $M$ denote the Galerkin matrices for the bilinear forms $a(u, v) = \int_\Omega \text{grad } u \cdot \text{grad } v \, dx$ and $m(u, v) = \int_\Omega u(x)v(x) \, dx$, respectively, and $V_{0,N} := S^0_{p,0}(\mathcal{M})$. Then the smallest and largest generalized eigenvalues of $A\vec{\mu} = \lambda M\vec{\mu}$, denoted by $\lambda_{\min}$ and $\lambda_{\max}$, satisfy

$$\frac{1}{\text{diam}(\Omega)^2} \leq \lambda_{\min} \leq C, \quad \lambda_{\max} \geq Ch^{-2}_{\mathcal{M}},$$

where the “generic constants” ($\rightarrow$ Rem. 5.3.44) depend only on the polynomial degree $p$ and the shape regularity measure $\rho_{\mathcal{M}}$.

Proof. (partial) We rely on the Courant-Fischer min-max theorem [18, Thm. 6.3.35] that, among other consequences, expresses the boundaries of the spectrum of a symmetric matrix through the extrema of its Rayleigh quotient

$$T = T^T \in \mathbb{R}^{N,N} \quad \Rightarrow \quad \lambda_{\min}(T) = \min_{\vec{\xi} \in \mathbb{R}^N \setminus \{0\}} \frac{\vec{\xi}^T T \vec{\xi}}{\vec{\xi}^T \vec{\xi}}, \quad \lambda_{\max}(T) = \max_{\vec{\xi} \in \mathbb{R}^N \setminus \{0\}} \frac{\vec{\xi}^T T \vec{\xi}}{\vec{\xi}^T \vec{\xi}}.$$
Apply this to the generalized eigenvalue problem

\[ A \vec{\mu} = \lambda M \vec{\mu} \quad \Rightarrow \quad \vec{\zeta} = M^{1/2} \vec{\mu} \quad \underbrace{M^{-1/2} A M^{-1/2}}_{= : T} \vec{\zeta} = \lambda \vec{\zeta} . \]

\[ \lambda_{\text{min}} = \min_{\vec{\mu} \neq 0} \frac{\vec{\mu}^T A \vec{\mu}}{\vec{\mu}^T M \vec{\mu}} , \quad \lambda_{\text{max}} = \max_{\vec{\mu} \neq 0} \frac{\vec{\mu}^T A \vec{\mu}}{\vec{\mu}^T M \vec{\mu}} . \] (6.1.49)

As a consequence we only have to find bounds for the extrema of a generalized Rayleigh quotient, cf. [18, Eq. 6.3.33]. This generalized Rayleigh quotient can be expressed as

\[ \frac{\vec{\mu}^T A \vec{\mu}}{\vec{\mu}^T M \vec{\mu}} = \frac{a(u_N, u_N)}{m(u_N, u_N)} , \quad \vec{\mu} \doteq \text{coefficient vector for } u_N . \] (6.1.50)

Now we discuss a lower bound for \( \lambda_{\text{max}} \), which can be obtained by inserting a suitable candidate function into (6.1.50).

Discussion for special setting: \( V_{0,N} = S_1^0(\mathcal{M}) \) on triangular mesh \( \mathcal{M} \)

Candidate function: “tent function” \( u_N = b_N^{i} \) (\( \rightarrow \) Sect. 3.2.3) for some node \( x^i \in V(\mathcal{M}) \) of the mesh!

By elementary computations as in Sect. 3.2.5 we find

\[ a(b_N^{i}, b_N^{i}) \approx C , \quad m(b_N^{i}, b_N^{i}) \leq C \max_{K \in \mathcal{U}(x^i)} h_K^2 , \] (6.1.51)
where the generic constants $C > 0$ depend on the shape regularity measure $\rho_M$ only.

\[(6.1.49) \& (6.1.51) \Rightarrow \lambda_{\text{max}} \geq C h_M^{-2} \]

Lemma 6.1.48 \hspace{1cm} \text{timestep constraint (6.1.42) unacceptable to semi-discrete parabolic evolutions!}

From [18, Sect. 13.3] we already know that some \textit{implicit} single step methods are not affected by stability induced timestep constraints.

Recall [18, Ex. 13.3.1]: apply diagonalization technique, see (6.1.39), to implicit Euler timestepping with uniform timestep $\tau > 0$

\[
\bar{\mu}^{(j)} = \bar{\mu}^{(j-1)} - \tau M^{-1} A \bar{\mu}^{(j)}
\]

\[
\bar{\eta} := T^T M^{1/2} \bar{\mu}
\]

\[
\bar{\eta}^{(j)} = \bar{\eta}^{(j-1)} - \tau D \bar{\eta}^{(j)} ,
\]
that is, the decoupling of eigencomponents carries over to the implicit Euler method: for \( i = 1, \ldots, N \)

\[ \eta_i^{(j)} = \eta_i^{(j-1)} - \tau \lambda_i \eta_i^{(j)} \Rightarrow \eta_i^{(j)} = \left( \frac{1}{1 + \tau \lambda_i} \right)^j \eta_i^{(0)} . \]  

(6.1.52)

\[ \left| \frac{1}{1 + \tau \lambda_i} \right| < 1 \quad \text{and} \quad \lambda_i > 0 \Rightarrow \lim_{j \to \infty} \eta_i^{(j)} = 0 \quad \forall \tau > 0 . \]  

(6.1.53)

This **diagonalization trick** can be applied to general Runge-Kutta single step methods (RKSSM, \( \rightarrow \) Def. 6.1.26). Loosely speaking, the following diagram commutes

\[
\begin{align*}
\mathbf{M} \frac{d}{dt} \bar{\mu} + \mathbf{A} \mu &= 0 & \text{transformation} & \bar{\eta} = \mathbf{T}^T \mathbf{M}^{1/2} \bar{\mu} & \frac{d}{dt} \eta_i = -\lambda_i \eta_i , \ i = 1, \ldots, N \\
\text{RK-SSM} & \downarrow & & \downarrow & \text{RK-SSM} \\
\bar{\mu}^{(j)} = \Psi^\tau \bar{\mu}^{(j-1)} & \text{transformation} & \bar{\eta} = \mathbf{T}^T \mathbf{M}^{1/2} \bar{\mu} & \bar{\eta}_i^{(j)} = \tilde{\Psi}^\tau \bar{\eta}_i^{(j-1)} , \ i = 1, \ldots, N .
\end{align*}
\]

(6.1.54)

The bottom line is

that we have to study the behavior of the RK-SSM **only** for linear scalar ODEs \( \dot{y} = -\lambda y, \ \lambda > 0 \).

This is the gist of the **model problem analysis** discussed in [18, Sect. 13.3].
There we saw that everything boils down to inspecting the modulus of a rational stability function on $\mathbb{C}$, see [18, Thm. 13.3.9]. This gave rise to the concept of $L$-stability, see [18, Def. 13.3.11]. Here, we will not delve into a study of stability functions.

**Necessary** condition for suitability of a single step method for semi-discrete parabolic evolution problem (6.1.19) (“method of lines”):

The discrete evolution $\Psi_T^\lambda : \mathbb{R} \mapsto \mathbb{R}$ of the single step method applied to the scalar ODE $\dot{y} = -\lambda y$ satisfies

$$\lambda > 0 \Rightarrow \lim_{j \to \infty} (\Psi_T^\lambda)^j y_0 = 0 \quad \forall y_0 \in \mathbb{R}, \quad \forall \tau > 0 . \quad (6.1.55)$$

**Definition 6.1.56** (L($\pi$)-stability).

A single step method satisfying (6.1.55) is called $L(\pi)$-stable.

**Example 6.1.57** (L($\pi$)-stable Runge-Kutta single step methods).

Simplest example: implicit Euler time-stepping (6.1.23).
Some commonly used higher order methods, specified through their Butcher schemes, see (6.1.27):

\[
\begin{array}{c|ccc}
1 & 5 & -1 \\
3 & 3 & 1 \\
4 & \frac{1}{4} & \frac{1}{4}
\end{array}
\]

(6.1.58) \quad \text{RADAU-3 scheme (order 3)}

\[
\begin{array}{c|ccc}
\lambda & \lambda & 0 \\
1 & 1 - \lambda & \lambda \\
1 & 1 - \lambda & \lambda
\end{array}
\]

(6.1.59) \quad \lambda := 1 - \frac{1}{2}\sqrt{2}, \quad \text{SDIRK-2 scheme (order 2)}

More examples → [18, Ex. 13.3.20]

6.1.5 Convergence
Why should one prefer complicated implicit $L(\pi)$-stable Runge-Kutta single step methods (→ Ex. 6.1.57) to the simple implicit Euler method?

Silly question! Because these methods deliver “better accuracy”!

However, we need some clearer idea of what is meant by this. To this end, we now study the dependence of (a norm of) the discretization error for a parabolic IBVP on the parameters of the spatial and temporal discretization.

**Example 6.1.60 (Convergence of fully discrete timestepping in one spatial dimension).**

\[
\frac{d}{dt}u - u'' = f(t, x) \text{ on } ]0, 1[ \times ]0, 1[ \\
\text{exact solution } u(x, t) = (1 + t^2)e^{-\pi^2 t} \sin(\pi x), \text{ source term accordingly} \\
\text{Linear finite element Galerkin discretization equidistant mesh, see Sect. 1.5.1.2, } V_{0,N} = S^0_{1,0}(\mathcal{M}), \\
\text{piecewise linear spatial approximation of source term } f(x, t) \\
\text{implicit Euler timestepping} (\rightarrow \text{Ex. 6.1.21}) \text{ with uniform timestep } \tau > 0
\]

Monitored: error norm \[
\left( \tau \sum_{j=1}^{M} |u - u_N(\tau j)|^2_{H^1(\Omega)} \right)^{1/2}.
\]
The norms $|u - u_N(\tau_j)|_{H^1(\Omega)}$ were approximated by high order local quadrature rules, whose impact can be neglected.

$\langle h_M$- and $\tau$-dependence of error norm

Observation:

$\tau$ small: error norm $\approx h_M$

$h_M$ small: error norm $\approx \tau$

The error seems to behave like

$$\text{error norm} \approx C_1 h_M + C_2 \tau. \quad (6.1.61)$$

Recall from Sect. 5.3.5, Thm. 5.1.10, Thm. 5.3.42:

energy norm of spatial finite element discretization error $O(h_M)$ for $h_M \to 0$.
Since the implicit Euler method is \textit{first order consistent} we expect

\begin{equation}
\text{temporal timestepping error } O(\tau)
\end{equation}

\[6.1.61\] \Rightarrow \text{conjecture: total error is \textbf{sum} of spatial and temporal discretization error.}

From Fig. 188 we draw the compelling conclusion:

\begin{itemize}
  \item for big mesh width $h_M$ (spatial error dominates) further reduction of timestep size $\tau$ is useless,
  \item if timestep $\tau$ is large (temporal error dominates), refinement of the finite element space does not yield a reduction of the total error.
\end{itemize}
Example 6.1.62 (Higher order timestepping for 1D heat equation).

- same IBVP as in Ex. 6.1.60
- spatial discretization on equidistant grid, very small meshwidth \( h = 0.5 \cdot 10^{-4} \), \( V_N = S^0_{1,0}(\mathcal{M}) \)

Various timestepping methods
(\(\Rightarrow\) different orders of consistency)

- implicit Euler timestepping (6.1.23), first order
- Crank-Nicolson-method (6.1.25), order 2
- SDIRK-2 timestepping (\(\rightarrow\) Ex. 6.1.57), order 2
- Gauss-Radau-Runge-Kutta collocation methods with \( s \) stages, order \( 2s - 1 \)

Note: all methods \( L(\pi) \)-stable (\(\rightarrow\) Def. 6.1.56), except for Crank-Nicolson-method.

![Discretization error for heat equation](image)
Monitored: $\max_j \left\| u(t_j) - u_N^{(j)} \right\|_{L^2([0,1])}$ (evaluated by high order quadrature)

Assume that

- the solution of the parabolic IBVP \((6.1.3)-(6.1.5)\) is “sufficiently smooth”;
- its spatial Galerkin finite element discretization relies on degree \(p\) Lagrangian finite elements (→ Sect. 3.4) on uniformly shape-regular families of meshes;
- timestepping is based on an \(L(\pi)\)-stable single step method of order \(q\) with uniform timestep \(\tau > 0\).

Then we can expect an asymptotic behavior of the total discretization error according to

\[
\left(\tau \sum_{j=1}^{M} |u - u_N(\tau j)|^2_{H^1(\Omega)}\right)^{\frac{1}{2}} \leq C\left(h_M^p + \tau^q\right),
\tag{6.1.64}
\]

where \(C > 0\) must not depend on \(h_M, \tau\).

This has been dubbed a “meta-theorem”, because quite a few technical assumptions on the exact solution and the methods have been omitted in its statement. Therefore it is not a mathematically rigorous statement of facts. More details in [19].
A message contained in (6.1.64):

\[
\text{total discretization error} = \text{spatial error} + \text{temporal error}
\]

Rem. 5.3.45 still applies: (6.1.64) does not give information about actual error, but only about the trend of the error, when discretization parameters \( h_M \) and \( \tau \) are varied.

Nevertheless, as in the case of the a priori error estimates of Sect. 5.3.5, we can draw conclusions about optimal refinement strategies in order to achieve prescribed error reduction.

As in Sect. 5.3.5 we make the assumption that the estimates (6.1.64) are sharp for all contributions to the total error and that the constants are the same (!)

\[
\begin{align*}
\text{contribution of spatial error} & \approx Ch_M^p, \quad h_M \overset{\Delta}{=} \text{mesh width (} \rightarrow \text{Def. 5.2.3)} , \\
\text{contribution of temporal error} & \approx C\tau^q, \quad \tau \overset{\Delta}{=} \text{timestep size .}
\end{align*}
\]

(6.1.65)

This suggests the following change of \( h_M, \tau \) in order to achieve error reduction by a factor of \( \rho > 1 \):

\[
\begin{align*}
\text{reduce mesh width by factor} & \quad \rho^{1/p} \quad \text{(6.1.65)} \\
\text{reduce timestep by factor} & \quad \rho^{1/q} \quad \Rightarrow \text{error reduction by} \quad \rho > 1 \quad .
\end{align*}
\]
Guideline: spatial and temporal resolution have to be adjusted in tandem

Remark 6.1.67 (Potential inefficiency of conditionally stable single step methods).

Terminology: A timestepping scheme is labelled conditionally stable, if blow-up can be avoided by using sufficient small timesteps (timestep constraint).

Now we can answer the question, why a stability induced timestep constraint like

$$\tau \leq O(h^{-2})$$

(6.1.68)

can render a single step method grossly inefficient for integrating semi-discrete parabolic IBVPs.

\[
\begin{align*}
\text{in order to reduce the error by a fixed factor } \rho \text{ one has to reduce both timestep and meshwidth by some other fixed factors (asymptotically). More concretely, for the timestep } \tau: \\
\tau \text{ requires reduction of } \tau \text{ by a factor } \rho^{1/q} \\
\text{stability entails reduction of } \tau \text{ by a factor } (\rho^{1/p})^2 = \rho^{2/p}.
\end{align*}
\]
\[
\frac{1}{q} < \frac{2}{p} \quad \Rightarrow \quad \text{stability enforces smaller timestep than required by accuracy}
\]

\[
\Rightarrow \quad \text{timestepping is \textit{inefficient}!}
\]

Faced with conditional stability (6.1.68), then for the sake of efficiency use \textit{high-order spatial discretization} combined with \textit{low order timestepping}.

However, this may not be easy to achieve

- because high-order timestepping is much simpler than high-order spatial discretization,
- because limited spatial smoothness of exact solution (\(\rightarrow\) results of Sect. 5.4 apply!) may impose a limit on \(q\) in (6.1.64).

Concretely: 5th-order \texttt{ode45} timestepping \((q = 5)\) \(\Rightarrow\) use degree-10 Lagrangian FEM!
Lemma 6.1.14 teaches that in the absence of time-dependent sources the rate of change of temperature will decay exponentially in the case of heat conduction.

Now we will encounter a class of evolution problems where temporal and spatial fluctuations will not be damped and will persist for good:

This will be the class of linear conservative wave propagation problems

As before these initial-boundary value problems (IBVP) will be posed on a space time cylinder $\tilde{\Omega} := \Omega \times [0, T] \subset \mathbb{R}^{d+1}$ (→ Fig. 183), where $\Omega \subset \mathbb{R}^d$, $d = 2, 3$, is a bounded spatial domain as introduced in the context of elliptic boundary value problems, see Sect. 2.1.1.

The unknown will be a function $u = (x, t) : \tilde{\Omega} \mapsto \mathbb{R}$. 

\[ \text{6.2 Wave equations} \]
6.2.1 Vibrating membrane

Recall:

- Tense string model (→ Sect. 1.4), shape of string described by continuous displacement function
  \[ u : [a, b] \mapsto \mathbb{R}, \ u \in H^1([a, b]). \]

- Taut membrane model (→ Sect. 2.1.1), shape of membrane given by displacement function
  \[ u : \Omega \mapsto \mathbb{R}, \ u \in H^1(\Omega), \] over base domain \( \Omega \subset \mathbb{R}^2 \).
In Sect. 2.1.3 we introduced the general variational formulation: with Dirichlet data (elevation of frame) given by $g \in C^0(\partial\Omega)$,

$$V := \{ v \in H^1(\Omega) : v_{|\partial\Omega} = g \}$$

we seek
\[ u \in V: \quad \int_{\Omega} \sigma(x) \, \text{grad} \, u \cdot \text{grad} \, v \, dx = \int_{\Omega} f(x) v(x) \, dx, \quad \forall v \in H_0^1(\Omega), \quad (6.2.1) \]

where \( f : \Omega \mapsto \mathbb{R} \equiv \text{density of vertical force}, \)

\[ \sigma : \Omega \mapsto \mathbb{R} \equiv \text{uniformly positive stiffness coefficient (characteristic of material of the membrane)}. \]

Now we switch to a \textit{dynamic setting}: we allow variation of displacement with time, \( u = u(x,t) \), the membrane is allowed to vibrate.

Recall (secondary school): \textit{Newton's second law of motion} (law of inertia)

\[ F = m \, a \]  \hspace{1cm} (6.2.2)

\[ \text{force} = \text{mass} \cdot \text{acceleration} \] \hspace{1cm} (6.2.3)

Apply this in a local version (stated for densities) to membrane

\[ \text{force density} \quad f(x,t) = \rho(x) \cdot \frac{\partial^2 u}{\partial t^2}(x,t), \] \hspace{1cm} (6.2.4)
where

- \( \rho : \Omega \rightarrow \mathbb{R}^+ \) \( \Rightarrow \) uniformly positive mass density of membrane, \([\rho] = \text{kg m}^{-2}\),
- \( \ddot{u} := \frac{\partial^2 u}{\partial t^2} \) \( \Rightarrow \) vertical acceleration (second temporal derivative of position).

Now, we assume that the force \( f \) in (2.3.2) is due to inertia forces only and express these using (6.2.4):

\[
(2.3.2) \quad \int \sigma(x) \nabla u(x, t) \cdot \nabla v(x) \, dx = - \int \rho(x) \cdot \frac{\partial^2 u}{\partial t^2}(x, t) \, dx \quad \forall v \in H^1_0(\Omega).
\]

Why the “−”-sign? Because, here the inertia force enters as a reaction force.

Linear wave equation in variational form (Dirichlet boundary conditions):

\[
\int_{\Omega} \rho(x) \cdot \frac{\partial^2 u}{\partial t^2}(x, t) \, dx + \int_{\Omega} \sigma(x) \nabla u(x, t) \cdot \nabla v(x) \, dx = 0 \quad \forall v \in H^1_0(\Omega)
\]

(6.2.5)

\[
u \in V(t): \quad m(\ddot{u}, v) + a(u, v) = 0 \quad \forall v \in V_0.
\]

(6.2.6)
where

\[ V(t) := \{ v : 0 < t < T \} \]

\[ V(t) := \{ v : 0 < t < T \} \]

(6.2.5)

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]

\[ \{ v : 0 < t < T \} \]
The wave equations (6.2.8), (6.2.9) have to be supplemented by

- **spatial boundary conditions**: \( v(\mathbf{x}, t) = g(\mathbf{x}, t) \) for \( \mathbf{x} \in \partial \Omega, \ 0 < t < T \),
- **two initial conditions**

\[
  u(\mathbf{x}, 0) = u_0(\mathbf{x}) \quad , \quad \frac{\partial u}{\partial t}(\mathbf{x}, 0) = v_0 \quad \text{for} \quad \mathbf{x} \in \Omega,
\]

with initial data \( u_0, v_0 \in H^1(\Omega) \), satisfying the compatibility conditions \( u_0(\mathbf{x}) = g(\mathbf{x}, 0) \) for \( \mathbf{x} \in \partial \Omega \).

**6.2.8** & boundary conditions & initial conditions = hyperbolic evolution problem

Hey, why do we need **two** initial conditions in contrast to the heat equation?

Remember that

- **(6.2.8)** is a **second-order equation** also in time (whereas the heat equation is merely first-order),
• for second order ODEs $\ddot{y} = f(y)$ we need two initial conditions

$y(0) = y_0 \quad \text{and} \quad \dot{y}(0) = v_0$, \hspace{1cm} (6.2.10)

in order to get a well-posed initial value problem, see [18, Rem. 12.1.15].

The physical meaning of the initial conditions (6.2.10) in the case of the membrane model is

• $u_0 \hat{=} \text{initial displacement of membrane, } u_0 \in H^1(\Omega) \text{ “continuous”,}$

• $v_0 \hat{=} \text{initial vertical velocity of membrane.}$

\textbf{Remark 6.2.11 (Boundary conditions for wave equation).}

Rem. [6.1.7] also applies to the wave equation (6.2.8):
On $\partial \Omega \times [0, T]$ we can impose any of the boundary conditions discussed in Sect. 2.6:

- **Dirichlet boundary conditions** $u(x, t) = g(x, t)$ (membrane attached to frame),
- **Neumann boundary conditions** $j(x, t) \cdot n = 0$ (free boundary, Rem. 2.4.16)
- **radiation boundary conditions** $j(x, t) \cdot n = \Psi(u(x, t))$,

and any combination of these as discussed in Ex. 2.6.7, yet, *only one* of them at any part of $\partial \Omega \times [0, T]$, see Rem. 2.6.6.

**Remark** 6.2.12 (Wave equation as first order system in time).

Usual procedure [18, Rem. 12.1.15]: higher-order ODE can be converted into first-order ODEs by introducing derivatives as additional solution components. This approach also works for the second-order (in time) wave equation (6.2.8):
Additional unknown: velocity \( v(\mathbf{x}, t) = \frac{\partial u}{\partial t}(\mathbf{x}, t) \)

\[
\frac{\partial^2 u}{\partial t^2} - \text{div}(\sigma(\mathbf{x}) \text{grad } u) = 0 \quad \begin{cases} 
\dot{u} = v, \\
\dot{v} = \text{div}(\sigma(\mathbf{x}) \text{grad } u) 
\end{cases} \quad \text{in } \tilde{\Omega} \quad (6.2.13)
\]

with initial conditions

\[
u(\mathbf{x}, 0) = u_0(\mathbf{x}) \quad , \quad \dot{v}(\mathbf{x}, 0) = v_0(\mathbf{x}) \quad \text{for } \mathbf{x} \in \Omega . \quad (6.2.14)\]

### 6.2.2 Wave propagation

Constant coefficient wave equation for \( d = 1, \Omega = \mathbb{R} \) (“Cauchy problem”)

\[
c > 0: \quad \frac{\partial^2 u}{\partial t^2} - c^2 \frac{\partial^2 u}{\partial x^2} = 0 \quad , \quad u(x, 0) = u_0(x) , \quad \frac{\partial u}{\partial t}(x, 0) = v_0(x) , \quad x \in \mathbb{R} . \quad (6.2.15)\]
Change of variables: $\xi = x + ct, \ \tau = x - ct$: $\tilde{u}(\xi, \tau) := u\left(\frac{\xi + \tau}{2c}, \frac{\xi - \tau}{2c}\right)$. Applying the chain rule we immediately see

$$u \text{ satisfies (6.2.15)} \implies \frac{\partial^2 \tilde{u}}{\partial \xi \partial \tau} = 0 \implies \tilde{u}(\xi, \tau) = F(\xi) + G(\tau),$$

for any $F, G \in C^2(\mathbb{R})$!

Matching initial data

$$u(x, t) = \frac{1}{2}(u_0(x + ct) + u_0(x - ct)) + \frac{1}{2} \int_{x-ct}^{x+ct} v_0(s) \, ds \quad (6.2.16)$$

(6.2.16) = d’Alembert solution of Cauchy problem (6.2.15).
Consider $d = 1$, initial-boundary value problem (6.2.15) for wave equation:

$$c > 0: \quad \frac{\partial^2 u}{\partial t^2} - c^2 \frac{\partial^2 u}{\partial x^2} = 0 \quad , \quad u(x, 0) = u_0(x) \quad , \quad \frac{\partial u}{\partial t}(x, 0) = v_0(x) \quad , \quad x \in \mathbb{R} . \quad (6.2.15)$$
Intuitive: from D’Alembert formula (6.2.16)

Domain of dependence: the value of the solution in \((\bar{x}, \bar{t})\) (●) will depend only on data in the yellow triangle in Fig. 192.
Domain of influence: initial data in $I_0$ will be relevant for the solution only in the yellow triangle in Fig. 193.

**Theorem 6.2.18** (Domain of dependence for isotropic wave equation). Let $u : \tilde{\Omega} \mapsto \mathbb{R}$ be a (classical) solution of $\frac{\partial^2 u}{\partial t^2} - c\Delta u = 0$. Then

$$\left( |x - x_0| \geq R \Rightarrow u(x, 0) = 0, \frac{\partial u}{\partial t}(x, 0) = 0 \right) \Rightarrow u(x, t) = 0, \text{ if } |x - x_0| \geq R + ct.$$
Now, we examine this for the model problem

\[ u \in H^1_0(\Omega): \int_\Omega \rho(x) \cdot \frac{\partial^2 u}{\partial t^2} v \, dx + \int_\Omega \sigma(x) \nabla u \cdot \nabla v \, dx = 0 \quad \forall v \in H^1_0(\Omega) \]  

(6.2.19)

\[ u \in V_0: \quad m(\ddot{u}, v) + a(u, v) = 0 \quad \forall v \in V_0. \]  

(6.2.20)

Here we do not include the case of non-homogeneous spatial Dirichlet boundary conditions through an affine trial space. This can always be taken into account by offset functions, see the remark after (6.1.5).
Theorem 6.2.21 (Energy conservation in wave propagation).
If \( u : \tilde{\Omega} \mapsto \mathbb{R} \) solves \( (6.2.20) \), then
\[
\frac{1}{2}m \left( \frac{\partial u}{\partial t}, \frac{\partial u}{\partial t} \right) + \frac{1}{2}a(u, u) \equiv \text{const}.
\]

**Proof.** A “formal proof” boils down to a straightforward application of the product rule (\( \rightarrow \) Rem. 6.1.13) together with the symmetry of the bilinear forms \( m \) and \( a \).

Introduce the **total energy**
\[
E(t) := \frac{1}{2}m \left( \frac{\partial u}{\partial t}, \frac{\partial u}{\partial t} \right) + \frac{1}{2}a(u, u).
\]

Then
\[
\frac{dE}{dt}(t) = m(\ddot{u}, \dot{u}) + a(\dot{u}, u) = 0 \quad \text{for solution } u \text{ of } (6.2.20),
\]
because this is what we conclude from \( (6.2.20) \) for the special test function \( v(\bm{x}) = \dot{u}(\bm{x}, t) \) for any \( t \in ]0, T[ \). \( \square \)
The method of lines approach to the wave equation \((6.2.19), (6.2.20)\) is exactly the same as for the heat equation, see Sect. 6.1.3.

Idea: Apply Galerkin discretization (→ Sect. 3.1) to abstract linear parabolic variational problem \((6.1.11)\).

\[
\begin{align*}
    t \in [0, T] \mapsto u(t) & \in V_0 : \\
    & \begin{cases} \\
    m\left(\frac{d^2u}{dt^2}(t), v\right) + a(u(t), v) = 0 & \forall v \in V_0, \\
    u(0) = u_0 \in V_0, & \frac{du}{dt}(0) = v_0 \in V_0. \\
    \end{cases}
\end{align*}
\]

\((6.2.22)\)

1st step: replace \(V_0\) with a finite dimensional subspace \(V_{0,N}, N := \dim V_{0,N} < \infty\)

Discrete hyperbolic evolution problem

\[
\begin{align*}
    t \in [0, T] \mapsto u(t) & \in V_{0,N} : \\
    & \begin{cases} \\
    m\left(\frac{d^2u_N}{dt^2}(t), v_N\right) + a(u_N(t), v_N) = 0 & \forall v_N \in V_{0,N}, \\
    u_N(0) = \text{projection/interpolant of } u_0 \text{ in } V_{0,N}, \\
    \frac{du_N}{dt}(0) = \text{projection/interpolant of } v_0 \text{ in } V_{0,N}. \\
    \end{cases}
\end{align*}
\]

\((6.2.23)\)
2nd step: introduce an ordered basis $\mathcal{B}_N := \{b^1_N, \ldots, b^N_N\}$ of $V_{0,N}$

\[
\left\{\begin{array}{c}
M \left\{ \frac{d^2}{dt^2} \vec{\mu}(t) \right\} + A \vec{\mu}(t) = 0 \\
\vec{\mu}(0) = \vec{\mu}_0, \quad \frac{d\vec{\mu}}{dt}(0) = \vec{\nu}_0.
\end{array}\right. \quad \text{(6.2.23) \Rightarrow (6.2.24)}
\]

- **s.p.d. stiffness matrix** $A \in \mathbb{R}^{N,N}$, $(A)_{ij} := a(b^j_N, b^i_N)$ (independent of time),
- **s.p.d. mass matrix** $M \in \mathbb{R}^{N,N}$, $(M)_{ij} := m(b^j_N, b^i_N)$ (independent of time),
- **source (load) vector** $\vec{\varphi}(t) \in \mathbb{R}^N$, $(\vec{\varphi}(t))_i := \ell(t)(b^i_N)$ (time-dependent),
- $\vec{\mu}_0 \doteq$ coefficient vector of a projection of $u_0$ onto $V_{0,N}$,
- $\vec{\nu}_0 \doteq$ coefficient vector of a projection of $v_0$ onto $V_{0,N}$. 

R. Hiptmair, C. Schwab, H. Harbrecht, V. Gradinaru, A. Chernov
SAM, ETHZ
Remark 6.2.25 (First-order semidiscrete hyperbolic evolution problem).

Completely analogous to Rem. 6.2.12:

\[ M \left\{ \frac{d^2}{dt^2} \vec{\mu}(t) \right\} + A \vec{\mu}(t) = 0 \]

\[ \left\{ \begin{array}{l}
\frac{d}{dt} \vec{\mu}(t) = \vec{\nu}(t), \\
M \frac{d}{dt} \vec{\nu}(t) = -A \vec{\mu}(t),
\end{array} \right. \quad 0 < t < T. \quad (6.2.26) \]

with initial conditions

\[ \vec{\mu}(0) = \vec{\mu}_0, \quad \vec{\nu}(0) = \vec{\nu}_0. \quad (6.2.27) \]
6.2.4 Timestepping

The method of lines approach gives us the semi-discrete hyperbolic evolution problem = 2nd-order ODE:

\[ M \left\{ \frac{d^2 \mu(t)}{dt^2} \right\} + A \mu(t) = 0 \, , \, \mu(0) = \bar{\mu}_0 \, , \, \frac{d\mu}{dt}(0) = \bar{\eta}_0 . \]  \hspace{1cm} (6.2.28)

Key features of (6.2.28) ⇒ to be respected “approximately” by timestepping:

- **reversibility**: (6.2.28) invariant under time-reversal \( t \leftarrow -t \)

- **energy conservation**, *cf.* Thm. 6.2.21:

\[ E_N(t) := \frac{1}{2} \frac{d\mu}{dt} \cdot M \frac{d\mu}{dt} + \frac{1}{2} \mu \cdot A \mu = \text{const} \]

*Example* 6.2.29 (Euler timestepping for 1st-order form of semi-discrete wave equation).
Model problem: wave propagation on a square membrane

$$\frac{\partial^2 u}{\partial t^2} - \Delta u = 0 \quad \text{on} \quad ]0, 1[^2 \times ]0, 1[, $$

$$u(x, t) = 0 \quad \text{on} \quad \partial \Omega \times ]0, T[, $$

$$u(x, 0) = u_0(x), \quad \frac{\partial u}{\partial t}(x, 0) = 0. $$

Initial data $u_0(x) = \max\{0, \frac{1}{5} - \|x\|\}, \quad v_0(x) = 0$,

$\mathcal{M} \hat{=} \text{"structured triangular tensor product mesh"}$, see Fig. 123, $n$ squares in each direction,

linear finite element space $V_{N,0} = S^0_{1,0}(\mathcal{M}), \quad N := \dim S^0_{1,0}(\mathcal{M}) = (n - 1)^2$,

All local computations (→ Sect. 3.5.4) rely on 3-point vertex based local quadrature formula “2D trapezoidal rule” (3.2.17). More explanations will be given in Rem. 6.2.34 below.
\( A = N \times N \) Poisson matrix, see (4.1.3), scaled with \( h := n^{-1} \).

- mass matrix \( M = hI \), thanks to quadrature formula, see Rem. 6.2.34.

Timestepping: implicit and explicit Euler method (→ Ex. 6.1.21, [18, Sect. 12.2]) for 1st-order ODE (6.2.26), timestep \( \tau > 0 \):

\[
\begin{align*}
\vec{\mu}(j) - \vec{\mu}(j-1) &= \tau \vec{\nu}(j-1), \\
M(\vec{\nu}(j) - \vec{\nu}(j-1)) &= -\tau A \vec{\mu}(j-1).
\end{align*}
\]

explicit Euler

\[
\begin{align*}
\vec{\mu}(j) - \vec{\mu}(j-1) &= \tau \vec{\nu}(j), \\
M(\vec{\nu}(j) - \vec{\nu}(j-1)) &= -\tau A \vec{\mu}(j).
\end{align*}
\]

implicit Euler

Monitored: behavior of (discrete) kinetic, potential, and total energy

\[
E_{\text{kin}}^{(j)} = (\vec{\nu}(j))^T M \vec{\nu}(j), \quad E_{\text{pot}}^{(j)} = (\vec{\mu}(j))^T A \vec{\mu}(j), \quad j = 0, 1, \ldots
\]

Explicit Euler timestepping:
Implicit Euler timestepping:
Observation: neither method conserves energy,

- explicit Euler timestepping $\Rightarrow$ steady increase of total energy
- implicit Euler timestepping $\Rightarrow$ steady decrease of total energy
Euler methods violate energy conservation!

(The same is true of all explicit Runge-Kutta methods, which lead to an increase of the total energy over time, and \(L(\pi)\)-stable implicit Runge-Kutta method, which make the total energy decay.)

Let us try another simple idea for the 2nd-order ODE (6.2.24):

Replace \(\frac{d^2}{dt^2} \vec{\mu} \) with symmetric difference quotient (1.5.96)

\[
M \left\{ \frac{d^2}{dt^2} \vec{\mu}(t) \right\} + A \vec{\mu}(t) = 0 \tag{6.2.28}
\]

\[
M \frac{\vec{\mu}(j+1) - 2\vec{\mu}(j) + \vec{\mu}(j-1)}{\tau^2} = -A \vec{\mu}(j), \quad j = 0, 1, \ldots \tag{6.2.30}
\]

This is a two-step method, the Störmer scheme/explicit trapezoidal rule
By Taylor expansion: 

Störmer scheme is a 2nd-order method

However, from where do we get $\vec{\mu}^{(-1)}$? Two-step methods need to be kick-started by a special initial step: This is constructed by approximating the second initial condition by a symmetric difference quotient:

$$\frac{d}{dt} \vec{\mu}(0) = \vec{\nu}_0 \quad \Rightarrow \quad \frac{\vec{\mu}(1) - \vec{\mu}^{(-1)}}{2\tau} = \vec{\nu}_0.$$ (6.2.31)

Example 6.2.32 (Leapfrog timestepping).

For the semi-discrete wave equation we again consider the explicit trapezoidal rule (Störmer scheme):

$$M \frac{\vec{\mu}^{(j+1)} - 2\vec{\mu}(j) + \vec{\mu}^{(j-1)}}{\tau^2} = -A \vec{\mu}(j), \quad j = 1, \ldots.$$ (6.2.30)

Inspired by Rem. 6.2.25 we introduce the auxiliary variable

$$\vec{\nu}^{(j+1/2)} := \frac{\vec{\mu}^{(j+1)} - \vec{\mu}^{(j)}}{\tau},$$
which can be read as an approximation of the velocity \( v := \dot{u} \).

This leads to a timestepping scheme, which is \textit{algebraically equivalent} to the explicit trapezoidal rule:

\[
\begin{align*}
M \frac{\vec{\nu}(j+\frac{1}{2}) - \vec{\nu}(j-\frac{1}{2})}{\tau} &= -A \vec{\mu}(j), \\
\frac{\vec{\mu}(j+1) - \vec{\mu}(j)}{\tau} &= \vec{\nu}(j+\frac{1}{2}), \\
+ \text{ initial step } \vec{\nu}(-\frac{1}{2}) + \vec{\nu}(\frac{1}{2}) &= 2\vec{\nu}_0.
\end{align*}
\]  

(6.2.33)

\[ 1 \times \text{evaluation } A \times \text{vector}, \]
\[ 1 \times \text{solution of linear system for } M \]
Remark 6.2.34 (Mass lumping).

Required in each step of leapfrog timestepping: solution of linear system of equations with (large sparse) system matrix \( \mathbf{M} \in \mathbb{R}^{N,N} \) ➢ expensive!

Trick for (bi-)linear finite element Galerkin discretization: \( V_{0,N} \subset S_1^0(\mathcal{M}) \):

use *vertex based local quadrature rule*

(e.g. “2D trapezoidal rule” (3.2.17) on triangular mesh)

\[
\int_K f(\mathbf{x}) \, d\mathbf{x} \approx \frac{|K|}{\#V(K)} \sum_{\mathbf{p} \in V(K)} f(\mathbf{p}), \quad V(K) := \text{set of vertices of } K.
\]

(For a comprehensive discussion of local quadrature rules see Sect. 3.5.4)

Mass matrix \( \mathbf{M} \) will become a *diagonal* matrix (due to defining equation (3.2.3) for nodal basis functions, which are associated with nodes of the mesh).

This so-called mass lumping trick was used in the finite element discretization of Ex. 6.2.29.
Example 6.2.35 (Energy conservation for leapfrog).

Model problem and discretization as in Ex. 6.2.29.

Leapfrog timestepping with constant timestep size $\tau = 0.01$

Code 6.2.36: Computing behavior of energies for Störmer timestepping

```matlab
function lfen(n,m)
% leapfrog timestepping for 2D wave equation, computation of energies
% n: spatial resolution (no. of cells in one direction)
% m: number of timesteps

% Assemble stiffness matrix, see Sect. 4.1, (4.1.3)
N = (n-1)^2; h = 1/n; A = gallery('poisson',n-1)/(h*h);

% initial displacement $u_0(x) = \max\{0,\frac{1}{5} - \|x\|\}$
[X,Y] = meshgrid(0:h:1,0:h:1);
U0 = 0.2-sqrt((X-0.5).^2+(Y-0.5).^2);
```
\texttt{U0} (\texttt{find} \ (U0 < 0)) = 0.0;
\texttt{u0} = \texttt{reshape} \ (U0(2: end-1, 2: end-1), N, 1);
\texttt{v0} = \texttt{zeros} \ (N, 1); \ % \ initial \ velocity

\% \ loop \ for \ Störmer \ timestepping, \ see \ (6.2.30)
\texttt{tau} = 1/m; \ % \ uniform \ timestep \ size
\texttt{u} = \texttt{u0} + \texttt{tau} * \texttt{v0} - 0.5 * \texttt{tau}^2 * \texttt{A} * \texttt{u0}; \ % \ special \ initial \ step
\texttt{u\_old} = \texttt{u0};
[\texttt{pen}, \texttt{ken}] = \texttt{geten} \ (A, \texttt{tau}, \texttt{u0}, \texttt{u}); \ % \ compute \ potential \ and \ kinetic \ energy
\texttt{E} = [0.5 * \texttt{tau}, \texttt{pen}, \texttt{ken}, \texttt{pen}+\texttt{ken}];
\texttt{for} \ k=1:m-1
\texttt{u\_new} = -(\texttt{tau}^2) * (\texttt{A} * \texttt{u}) + 2 * \texttt{u} - \texttt{u\_old};
[\texttt{pen}, \texttt{ken}] = \texttt{geten} \ (A, \texttt{tau}, \texttt{u}, \texttt{u\_new});
\texttt{E} = [\texttt{E}; (k+0.5) * \texttt{tau}, \texttt{pen}, \texttt{ken}, \texttt{pen}+\texttt{ken}];
\texttt{u\_old} = \texttt{u}; \texttt{u} = \texttt{u\_new};
\texttt{end}

\texttt{figure} \ (\texttt{\'name\'}\texttt{\',\'Leapfrog \ energies\'});
\texttt{plot} \ (\texttt{E}(:, 1), \texttt{E}(:, 3), \texttt{\'r-\'}, \texttt{E}(:, 1), \texttt{E}(:, 2), \texttt{\'b-\'}, \texttt{E}(:, 1), \texttt{E}(:, 4), \texttt{\'m-\'});
\texttt{xlabel} \ ('{\bf \textit{time}} \ t', \texttt{\'fontsize\'}, 14);
\texttt{ylabel} \ ('{\bf \textit{energies}}', \texttt{\'fontsize\'}, 14);
\texttt{legend} \ ('{\bf \textit{kinetic energy}}', '{\bf \textit{potential energy}}', '{\bf \textit{total energy}}', \texttt{\'location\'}, \texttt{\'south\'});
Code 6.2.37: Computing potential and kinetic energy for Störmer timestepping

```matlab
function [pen, ken] = geten(A, ts, u_old, u_new)

% Compute the current approximate potential and kinetic energies for u_old
% and u_new from Sörmer timestepping
%
% E_{kin}^{(j)} = \tau^{-2}(\vec{\mu}^{(j)} - \vec{\mu}^{(j-1)})^T M (\vec{\mu}^{(j)} - \vec{\mu}^{(j-1)}) ,
% E_{pot}^{(j)} = \frac{1}{4}(\vec{\mu}^{(j)} + \vec{\mu}^{(j-1)})^T A (\vec{\mu}^{(j)} + \vec{\mu}^{(j-1)}) , \ j = 0, 1, ...

meanv = 0.5*(u_old+u_new); pen = dot(meanv, A*meanv); % potential energy

dtemp = (u_new-u_old)/ts; ken = dot(dtemp, dtemp); % kinetic energy
```

6.2.5 CFL-condition

*Example 6.2.38* (Blow-up for leapfrog timestepping).


Observation:
Leapfrog suffers a blow-up: exponential increase of energies!
A similar behavior is observed with the explicit Euler scheme for the semi-discrete heat equation, in case the timestep constraint is violated, see Sect. 6.1.4.2.

> (as in Sect. 6.1.4.2) Stability analysis of leapfrog timestepping based on diagonalization:

\[ \exists \text{ orthogonal } T \in \mathbb{R}^{N,N}: \quad T^\top M^{-1/2} A M^{-1/2} T = D := \text{diag}(\lambda_1, \ldots, \lambda_N). \]

where the \( \lambda_i > 0 \) are generalized eigenvalues for \( A\vec{\xi} = \lambda M\vec{\xi} \) \( \Rightarrow \lambda_i \geq \gamma \) for all \( i \) (\( \gamma \) is the constant introduced in (6.1.12)).
Next, apply transformation $\tilde{\eta} := T^T M^{1/2} \tilde{\mu}$ to the 2-step formulation (6.2.30)

\[
(6.2.30) \quad \tilde{\eta} = T^T M^{1/2} \tilde{\mu} \quad \Rightarrow \quad \tilde{\eta}^{(j+1)} - 2\tilde{\eta}^{(j)} + \tilde{\eta}^{(j-1)} = -\tau^2 D \tilde{\eta}^{(j)}
\]

Again, we have achieved a complete decoupling of the time-stepping for the eigencomponents.

\[
\eta^{(j+1)}_i - 2\eta^{(j)}_i + \eta^{(j-1)}_i = -\tau^2 \lambda_i \eta^{(j)}_i, \quad i = 1, \ldots, N, \quad j = 1, 2, \ldots \quad (6.2.39)
\]

In fact, (6.2.39) is what we end up with then applying Størmer's scheme to the scalar linear 2nd-order ODE $\ddot{\eta}_i = -\lambda_i \eta_i$. In a sense, the commuting diagram (6.1.54) remains true for 2-step methods and second-order ODEs.

(6.2.39) is a linear two-step recurrence formula for the sequences $(\eta^{(j)}_i)_j$.

Try:

\[
\eta^{(j)}_i = \xi^j \quad \text{for some} \quad \xi \in \mathbb{C} \setminus \{0\}
\]

Plug this into (6.2.39)

\[
\Rightarrow \quad \xi^2 - 2\xi + 1 = -\tau^2 \lambda_i \xi \quad \Leftrightarrow \quad \xi^2 - (2 - \tau^2 \lambda_i)\xi + 1 = 0.
\]

\[
\Rightarrow \quad \text{two solutions} \quad \xi_\pm = \frac{1}{2} \left( 2 - \tau^2 \lambda_i \pm \sqrt{(2 - \tau^2 \lambda_i)^2 - 4} \right).
\]
We can get a blow-up of some solutions of (6.2.39), if $|\xi_+| > 1$ of $|\xi_-| > 1$. From secondary school we know Vieta’s formula

$$\xi_+ \cdot \xi_- = 1 \quad \Rightarrow \quad \begin{cases} \xi_+ \in \mathbb{R} \quad \text{and} \quad \xi_+ \neq \xi_- \quad \Rightarrow \quad |\xi_+| > 1 \text{ or } |\xi_-| > 1 \\ \xi_- = \xi^* \quad \Rightarrow \quad |\xi_-| = |\xi_+| = 1 \end{cases},$$

where $\xi^*$ designates complex conjugation. So the recurrence (6.2.39) has only bounded solution, if and only if

$$\text{discriminant} \quad D := (2 - \tau^2 \lambda_i)^2 - 4 \leq 0 \quad \Leftrightarrow \quad \tau < \frac{2}{\sqrt{\lambda_i}}. \quad (6.2.40)$$

$\leftrightarrow$ \quad \textit{stability induced timestep constraint for leapfrog timestepping}

Special setting: spatial finite element Galerkin discretization based on fixed degree Lagrangian finite element spaces ($\rightarrow$ Sect. 3.4), meshes created by uniform regular refinement.

Under these conditions a generalization of Lemma 6.1.48 shows

\textit{Stability} of leapfrog timestepping entails $\tau \leq O(h_M)$ for $h_M \to 0$. 

This is known as the **Courant-Friedrichs-Lewy (CFL) condition**.

**Remark 6.2.41 (Geometric interpretation of CFL condition in 1D).**

**Setting:**

- 1D wave equation, (spatial) boundary conditions ignored ("Cauchy problem"),

\[
c > 0: \quad \frac{\partial^2 u}{\partial t^2} - c^2 \frac{\partial^2 u}{\partial x^2} = 0, \quad u(x, 0) = u_0(x), \quad \frac{\partial u}{\partial t}(x, 0) = v_0(x), \quad x \in \mathbb{R}.
\]  \(6.2.15\)

- Linear finite element Galerkin discretization on equidistant spatial mesh \(\mathcal{M} := \{[x_{j-1}, x_j]: j \in \mathbb{Z}\}\), \(x_j := hj\) (meshwidth \(h\)), see Sect. 1.5.1.2.

- **Mass lumping** for computation of mass matrix, which will become \(h \cdot I\), see Rem 6.2.34.

- Timestepping by Sörmer scheme (6.2.30) with constant timestep \(\tau > 0\).
Since the method is a two-step method, information from time-slices $t_k$ and $t_k-1$ is needed.

Below: yellow region $\hat{\Delta}$ domain of dependence (d.o.d.) of $(\bar{x}, \bar{t})$
Fig. 202.

\( c\tau < h \): numerical domain of dependence (marked \( \cdots \) contained in d.o.d.
\( \Rightarrow \) CFL-condition met

Fig. 203.

\( c\tau > h \): numerical domain of dependence (marked \( \cdots \) not contained in d.o.d.
\( \Rightarrow \) CFL-condition violated

\( \bullet \Rightarrow \) coarse grid, \( \blacksquare \Rightarrow \) fine grid, \( \square \Rightarrow \) d.o.d.

\( \triangleleft \) 1D consideration:

sequence of \textbf{equidistant} space-time grids of \( \tilde{\Omega} \) with

\[ \tau = \gamma h \quad (\tau/h = \text{meshwidth in time/space}) \]

If \( \gamma > \text{CFL-constraint} \) (here \( \gamma > c^{-1} \)), then

analytical domain \( \not\subset \) numerical domain of dependence
initial data $u_0$ outside numerical domain of dependence cannot influence approximation at grid point $(\bar{x}, \bar{t})$ on any mesh  ➤ no convergence!

CFL-condition $\iff$ analytical domain of dependence $\subset$ numerical domain of dependence

Will the CFL-condition thwart the efficient use of leapfrog, see Rem. 6.1.67?

To this end we need an idea about the convergence of the solutions of the fully discrete method:
“Meta-theorem” 6.2.42 (Convergence of fully discrete solutions of the wave equation).

Assume that

- the solution of the IBVP for the wave equation (6.2.19) is “sufficiently smooth”,
- its spatial Galerkin finite element discretization relies on degree \( p \) Lagrangian finite elements (∴ Sect. [3.4]) on uniformly shape-regular families of meshes,
- timestepping is based on the leapfrog method (6.2.33) with uniform timestep \( \tau > 0 \).

Then we can expect an asymptotic behavior of the total discretization error according to

\[
\left( \tau \sum_{j=1}^{M} \| u - u_N(\tau j) \|_{H^1(\Omega)}^2 \right)^{\frac{1}{2}} \leq C ( h^p_M + \tau^2 ), \tag{6.2.43}
\]

\[
\left( \tau \sum_{j=1}^{M} \| u - u_N(\tau j) \|_{L^2(\Omega)}^2 \right)^{\frac{1}{2}} \leq C ( h^{p+1}_M + \tau^2 ), \tag{6.2.44}
\]

where \( C > 0 \) must not depend on \( h_M, \tau \).

"expect": unless lack of regularity of the solution \( u \) interferes, cf. 5.4, Rem. 5.4.11.
As in the case of Metatheorem 6.1.63 (nothing new!) we find:

\[
\text{total discretization error} = \text{spatial error} + \text{temporal error}
\]

Rem. 5.3.45 still applies: (6.2.43) does not give information about actual error, but only about the trend of the error, when discretization parameters \( h_M \) and \( \tau \) are varied.

Nevertheless, as in the case of the a priori error estimates of Sect. 5.3.5, we can draw conclusions about optimal refinement strategies in order to achieve prescribed error reduction.

As in Sect. 5.3.5 we make the assumption that the estimates (6.2.43) are sharp for all contributions to the total error and that the constants are the same (!)

\[
\begin{align*}
\text{contribution of spatial (energy) error} & \approx Ch_M^p, \quad h_M = \text{mesh width} \quad (\rightarrow \text{Def. 5.2.3}), \\
\text{contribution of temporal error} & \approx C\tau^2, \quad \tau = \text{timestep size}.
\end{align*}
\]

(6.2.45)

This suggests the following change of \( h_M, \tau \) in order to achieve error reduction by a factor of \( \rho > 1 \):

\[
\begin{align*}
\text{reduce mesh width by factor} & \quad \rho^{1/p} \quad (6.1.65) \\
\text{reduce timestep by factor} & \quad \rho^{1/2} \quad \Rightarrow \quad (\text{energy}) \text{ error reduction by } \rho > 1. \quad (6.2.46)
\end{align*}
\]
Guideline: spatial and temporal resolution have to be adjusted in tandem

Parallel zu Rem. 6.1.67 we may wonder whether the timestep constraint \( \tau < O(h_M) \) (asymptotically) enforces small timesteps not required for accuracy:

When interested in error in energy norm (\( \leftrightarrow H^1(\Omega) \)-norm):

Only for \( p = 1 \) (linear Lagrangian finite elements) the requirement \( \tau < O(h_M) \) stipulates the use of a smaller timestep than accuracy balancing according to (6.2.46).

When interested in \( L^2(\Omega) \)-norm:

No undue timestep constraint enforced by CFL-condition for any \((h\text{-version})\) of Lagrangian finite element Galerkin discretization.
The leapfrog timestep constraint \( \tau \leq O(h_M) \) does not compromise (asymptotic) efficiency, if \( p \geq 2 \) (\( p \) denotes degree of spatial Lagrangian finite elements).
7 Convection-Diffusion Problems

Supplementary and further reading:

[25] offers comprehensive about theory and algorithms for the numerical approximation of singularly perturbed problems of the type treated in this chapter.

7.1 Heat conduction in a fluid
$\Omega \subset \mathbb{R}^d \triangleq \text{bounded computational domain}, \ d = 1, 2, 3$

To begin with we want to develop a mathematical model for stationary fluid flow, for instance, the steady streaming of water.
7.1.1 Modelling fluid flow

Flow field:
\[ \mathbf{v} : \Omega \mapsto \mathbb{R}^d \]

Assumption:
\[ \mathbf{v} \text{ is continuous, } \mathbf{v} \in (C^0(\overline{\Omega}))^d \]

In fact, we will require that \( \mathbf{v} \) is uniformly Lipschitz continuous, but this is a mere technical assumption.

Clearly: \( \mathbf{v}(\mathbf{x}) \) \( \hat{=} \) fluid velocity at point \( \mathbf{x} \in \Omega \)

\( \triangleright \) \( \mathbf{v} \) corresponds to a velocity field!
Given a flow field $v \in (C^0(\overline{\Omega}))^d$ we can consider the autonomous initial value problems

$$\frac{d}{dt}y = v(y), \quad y(0) = x_0.$$  

(7.1.1)

Its solution $t \mapsto y(t)$ defines the path travelled by a particle carried along by the fluid, a particle trajectory, also called a streamline.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{fig206}
\caption{Particle trajectories (streamlines) in flow field of Fig. 205.}
\end{figure}

(\ast \hat{=} \text{initial particle positions})
A flow field induces a transformation (mapping) of space! to explain this, let us temporarily make the assumption that

the flow does neither enter nor leave \( \Omega \),
(this applies to fluid flow in a close container)

which can be modelled by

\[
\mathbf{v}(x) \cdot \mathbf{n}(x) = 0 \quad \forall x \in \partial \Omega ,
\]

that is, the flow is always parallel to the boundary of \( \Omega \): all particle trajectories stay inside \( \Omega \).

Now we fix some “time of interest” \( t > 0 \).

> mapping \( \Phi^t : \left\{ \begin{array}{c} \Omega \rightarrow \Omega \\ x_0 \rightarrow y(t) \end{array} \right\} , \quad t \mapsto y(t) \) solution of IVP (7.1.1),

(7.1.3)
is well-defined mapping of $\Omega$ to itself, the flow map. Obviously, it satisfies

$$\Phi^0 x_0 = x_0 \quad \forall x_0 \in \Omega .$$

(7.1.4)

In [18, Def. 12.1.27] the more general concept of an evolution operator was introduced, which agrees with the flow map in the current setting.

**Fig. 208**
flow field $\mathbf{v} : \Omega \mapsto \mathbb{R}^2$

**Fig. 209**
snapshots of $\Phi^t(V)$ for control volume $V$
\( \Phi^\tau(V) \) \( \hat{=} \) volume occupied at time \( t = \tau \) by particles that occupied \( V \subset \Omega \) at time \( t = 0 \).

### 7.1.2 Heat convection and diffusion

\[ u : \Omega \mapsto \mathbb{R} \] stationary temperature distribution in fluid \textit{moving} according to a stationary flow field

\[ v : \Omega \mapsto \mathbb{R}^d \]

We adapt the considerations of Sect. [2.5] that led to the stationary heat equation. Recall

**Conservation of energy**

\[
\int_{\partial V} \mathbf{j} \cdot \mathbf{n} \, dS = \int_V f \, d\mathbf{x} \quad \text{for all "control volumes" } V. \tag{2.5.2}
\]

- power flux through surface of \( V \)
- heat production inside \( V \)
From 2.5.2 by Gauss’ theorem Thm. 2.4.5

\[ \int_V \text{div} \mathbf{j}(x) \, dx = \int_V f(x) \, dx \quad \text{for all “control volumes” } V \subset \Omega. \]

Now appeal to another version of the fundamental lemma of the calculus of variations, see Lemma 2.4.10, this time sporting piecewise constant test functions.

\[ \text{local form of energy conservation:} \]

\[ \text{div} \mathbf{j} = f \quad \text{in } \Omega. \quad (2.5.5) \]

However, in a moving fluid a power flux through a fixed surface is already caused by the sheer fluid flow carrying along thermal energy. This is reflected in a modified Fourier’s law (2.5.3):
Fourier’s law in moving fluid

\[ j(x) = -\kappa \nabla u(x) + v(x) \rho u(x) , \quad x \in \Omega . \tag{7.1.5} \]

**diffusive heat flux** (due to spatial variation of temperature)

**convective heat flux** (due to fluid flow)

\( \kappa > 0 \) $\equiv$ heat conductivity \( ([\kappa] = 1 \text{W/Km}) \), \( \rho > 0 \) $\equiv$ heat capacity \( ([\rho] = \frac{\text{J}}{\text{Km}^3}) \), both assumed to be constant (in contrast to the models of Sect. 2.5 and Sect. 6.1.1).

Combine equations (2.5.5) & (7.1.5):

\[
\text{div} \ j = f + j(x) = -\kappa \nabla u(x) + v(x) \rho u(x)
\]

\[
- \text{div}(\kappa \nabla u) + \text{div}(\rho v(x)u) = f \quad \text{in} \quad \Omega . \tag{7.1.6}
\]
**Linear scalar convection-diffusion equation** (for unknown temperature $u$)

\[ - \text{div}(\kappa \, \text{grad} \, u) + \div(\rho \mathbf{v}(\mathbf{x})u) = f. \]

**Terminology**:
- **diffusive term** (2nd-order)
- **convective term** (1st-order)

The 2nd-order elliptic PDE (7.1.6) has to be supplemented with exactly one **boundary condition** on any part of $\partial \Omega$, see Sect. 2.6, Ex. 2.6.7. This can be any of the (“elliptic”) boundary conditions introduced in Sect. 2.6:

- **Dirichlet boundary conditions**: $u = g \in C^0(\partial \Omega)$ on $\partial \Omega$ (fixed surface temperature),
- **Neumann boundary conditions**: $\mathbf{j} \cdot \mathbf{n} = -h$ on $\partial \Omega$ (fixed heat flux),
- **(non-linear) radiation boundary conditions**: $\mathbf{j} \cdot \mathbf{n} = \Psi(u)$ on $\partial \Omega$ (temperature dependent heat flux, radiative heat flux).

**Guideline**: Required boundary conditions determined by highest-order term.
7.1.3 Incompressible fluids

For the sake of simplicity we will mainly consider incompressible fluids.

**Definition 7.1.7** (Incompressible flow field).

A fluid flow is called incompressible, if the associated flow map \( \Phi^t \) is volume preserving,

\[
|\Phi^t(V)| = |\Phi^0(V)| \quad \text{for all sufficiently small} \ t > 0, \ \text{for all control volumes} \ V.
\]

Can incompressibility be read off the velocity field \( \mathbf{v} \) of the flow?

To investigate this issue, again assume the “no flow through the boundary condition” (7.1.2) and recall that the flowmap \( \Phi^t \) from (7.1.3) satisfies

\[
\frac{\partial}{\partial t} \Phi(t, x) = \mathbf{v}(\Phi(t, x)), \quad x \in \Omega, \ t > 0.
\]
Here, in order to make clear the dependence on independent variables, time occurs as an argument of $\Phi$ in brackets, on par with $x$.

Next, formal differentiation w.r.t. $x$ and change of order of differentiation yields a differential equation for the Jacobian $D_x \Phi^t$,

$$\frac{\partial}{\partial t} (D_x \Phi)(t, x) = Dv(\Phi(t, x))(D_x \Phi)(t, x). \quad (7.1.9)$$

Second strand of thought: apply transformation formula for integrals (3.5.31), [29, Satz 8.5.2]: for fixed $t > 0$

$$|\Phi(t, V)| = \int_{\Phi(t,V)} 1 \, dx = \int_V |\det(D_x \Phi)(t, \hat{x})| \, d\hat{x}. \quad (7.1.10)$$

Volume preservation by the flow map is equivalent to

$$t \mapsto |\Phi(t, V)| = \text{const.} \iff \frac{d}{dt} |\Phi(t, V)| = 0,$$

for any control volume $V \subset \Omega$.
Theorem 7.1.11 (Differentiation formula for determinants).

Let $S : I \subset \mathbb{R} \mapsto \mathbb{R}^{n,n}$ be a smooth matrix-valued function. If $S(t_0)$ is regular for some $t_0 \in I$, then

$$
\frac{d}{dt} (\det \circ S)(t_0) = \det(S(t_0)) \operatorname{tr}(\frac{dS}{dt}(t_0)S^{-1}(t_0)).
$$

because the divergence of a vector field $\mathbf{v}$ is just the trace of its Jacobian $D\mathbf{v}$! From (7.1.4) we know that for small $t > 0$ the Jacobian $D_x \Phi(t, \hat{x})$ will be close to $I$ and, therefore, $\det(D_x \Phi)(t, \hat{x}) \neq 0$ for $t \approx 0$. Thus, for small $t > 0$ we conclude

$$
\frac{d}{dt} |\Phi(t, V)| = 0 \iff \operatorname{div} \mathbf{v}(\Phi(t, \hat{x})) = 0 \quad \forall \hat{x} \in V.
$$

Since this is to hold for any control volume $V$, the final equivalence is

$$
\frac{d}{dt} |\Phi(t, V)| = 0 \quad \forall \text{control volumes } V \iff \operatorname{div} \mathbf{v} = 0 \quad \text{in } \Omega.
$$
**Theorem 7.1.12** (Divergence-free velocity fields for incompressible flows).

A stationary fluid flow in $\Omega$ is incompressible ($\rightarrow$ Def. 7.1.7), if and only if its associated velocity field $\mathbf{v}$ satisfies $\text{div} \, \mathbf{v} = 0$ everywhere in $\Omega$.

In the sequel we make the **assumption**:

$$\text{div} \, \mathbf{v} = \sum_{j=1}^{d} \frac{\partial v_j}{\partial x_j} = 0 .$$

(Note: for $d = 1$ this boils down to $\frac{dv}{dx} = 0$ and implies $\mathbf{v} = \text{const}$.)

Then we can use the product rule in higher dimensions of Lemma 2.4.3:

$$\text{div}(\rho \mathbf{v} \, \mathbf{u}) = \rho (\mathbf{u} \cdot \text{div} \, \mathbf{v} + \mathbf{v} \cdot \text{grad} \, \mathbf{u}) = \rho \mathbf{v} \cdot \text{grad} \, \mathbf{u} .$$

(7.1.13)

Thus, we can rewrite the scalar convection-diffusion equation (7.1.6) for an incompressible flow field

$$-\text{div}(\kappa \text{grad} \, \mathbf{u}) + \text{div}(\rho \mathbf{v} (x) \mathbf{u}) = f \quad \text{in} \quad \Omega$$
\( \text{div} \, \mathbf{v} = 0 \)

\[-\kappa \Delta u + \rho \mathbf{v} \cdot \text{grad} \, u = f \quad \text{in} \quad \Omega \, . \quad (7.1.14)\]

When carried along by the flow of an incompressible fluid, the temperature cannot be increased by local compression, the effect that you can witness when pumping air. Hence, only sources/sinks can lead to local extrema of the temperature.

Now recall the discussion of the physical intuition behind the maximum principle of Thm. 5.7.2. These considerations still apply to stationary heat flow in a moving incompressible fluid.

**Theorem 7.1.15** (Maximum principle for scalar 2nd-order convection diffusion equations). \([12, 6.4.1, \text{Thm. I}]\)

Let \( \mathbf{v} : \Omega \mapsto \mathbb{R}^d \) be a continuously differentiable vector field. Then there holds the maximum principle

\[-\Delta u + \mathbf{v} \cdot \text{grad} \, u \geq 0 \implies \min_{x \in \partial \Omega} u(x) = \min_{x \in \Omega} u(x) , \]

\[-\Delta u + \mathbf{v} \cdot \text{grad} \, u \leq 0 \implies \max_{x \in \partial \Omega} u(x) = \max_{x \in \Omega} u(x) . \]
7.1.4 Transient heat conduction

In Sect. 6.1.1 we generalized the laws of stationary heat conduction derived in Sect. 2.5 to time-dependent temperature distributions \( u = u(\mathbf{x}, t) \) sought on a space-time cylinder \( \tilde{\Omega} := \Omega \times ]0, T[ \). The same ideas apply to heat conduction in a fluid:

- Start from energy balance law (6.1.1) and convert it into local form (6.1.2).
- Combine it with the extended Fourier's law (7.1.5).

\[
\frac{\partial}{\partial t}(\rho u) - \text{div}(\kappa \text{grad} u) + \text{div}(\rho \mathbf{v}(\mathbf{x}, t)u) = f(\mathbf{x}, t) \quad \text{in} \quad \tilde{\Omega} := \Omega \times ]0, T[. \tag{7.1.16}
\]

For details and notations refer to Sect. 6.1.1.

This PDE has to be supplemented with
• boundary conditions (as in the stationary case, see Sect. 2.6),
• initial conditions (same as for pure diffusion, see Sect. 6.1.1).

Under the assumption $\text{div}_x \mathbf{v}(x, t) = 0$ of incompressibility (→ Def. 7.1.7 and Thm. 7.1.12) (7.1.16) is equivalent to, cf. (7.1.13),

$$\frac{\partial}{\partial t}(\rho u) - \kappa \Delta u + \rho \mathbf{v}(x, t) \cdot \nabla u = f(x, t) \quad \text{in } \tilde{\Omega} := \Omega \times ]0, T[. \quad (7.1.17)$$

7.2 Stationary convection-diffusion problems
Model problem, cf. (7.1.14), modelling stationary heat flow in an incompressible fluid with prescribed temperature at “walls of the container” (↔ Dirichlet boundary conditions).

\[-\kappa \Delta u + \rho \mathbf{v}(\mathbf{x}) \cdot \nabla u = f \quad \text{in } \Omega , \quad u = 0 \quad \text{on } \partial \Omega .\]

Perform scaling \( \hat{=} \) choice of physical units: makes equation non-dimensional by fixing “reference length”, “reference time interval”, “reference temperature”, “reference power”.

A suitable choice of physical units leads to rescaled physical constants \( \kappa \rightarrow \epsilon, \rho \rightarrow 1, \| \mathbf{v} \|_{L^\infty(\Omega)} \rightarrow 1 \).

After scaling we deal with the non-dimensional boundary value problem

\[-\epsilon \Delta u + \mathbf{v}(\mathbf{x}) \cdot \nabla u = f \quad \text{in } \Omega , \quad u = 0 \quad \text{on } \partial \Omega , \quad (7.2.1)\]

with \( \epsilon > 0, \| \mathbf{v} \|_{L^\infty(\Omega)} = 1, \text{div } \mathbf{v} = 0 \rightarrow \text{incompressible fluid, see Def. [7.1.7].} \)
Remark 7.2.2 (Variational formulation for convection-diffusion BVP).

Standard “4-step approach” of Sect. 2.8 can be directly applied to BVP (7.2.1) with one new twist:

Do not use integration by parts (Green’s formula, Thm. 2.4.7) on convective terms!

Variational formulation for BVP (7.2.1):

\[ u \in H^1_0(\Omega): \quad \epsilon \int_{\Omega} \nabla u \cdot \nabla v \, dx + \int_{\Omega} (v \cdot \nabla u) v \, dx = \int_{\Omega} f(x) \, dx \quad \forall v \in H^1_0(\Omega). \]

\[ \Rightarrow \text{ a linear variational problem, see Sect. 2.3.1.} \]

Obvious: \( a \) is not symmetric, see (2.1.19).

\( \Rightarrow \) a does not induce an energy norm (\( \rightarrow \) Def. 2.1.27)

As replacement for the energy norm use \( H^1(\Omega) \)-(semi)norm (\( \rightarrow \) Def. 2.2.10)
In this case we have to make sure that $a$ fits the chosen norm in the sense that

$$\exists C > 0: \ |a(u, v)| \leq C |u|_{H^1(\Omega)} |v|_{H^1(\Omega)} \quad \forall u, v \in H^1_0(\Omega). \quad (7.2.3)$$

Terminology: $(7.2.3)$ $\text{a is continuous on } H^1(\Omega)$, cf. $(3.1.2)$. 

By Cauchy-Schwarz inequality for integrals $(2.1.28)$:

$$|a(u, v)| \leq \|v\|_{L^\infty(\Omega)} |u|_{H^1(\Omega)} \|v\|_{L^2(\Omega)} \leq \text{diam}(\Omega) \|v\|_{L^\infty(\Omega)} |u|_{H^1(\Omega)} |v|_{H^1(\Omega)} \quad \forall u, v \in H^1(\Omega),$$

which confirms $(7.2.3)$.

Surprise: $a$ is positive definite (→ Def. 2.1.25), because

$$\int_\Omega (v \cdot \text{grad } u) u \, dx = \int_\Omega (v u) \cdot \text{grad } u \, dx$$

Green’s formula

$$= - \int_\Omega \text{div}(v u) u \, dx + \int_{\partial \Omega} u^2 v \cdot n \, dS$$

$(2.4.4)$ & $\text{div } v = 0

\int_\Omega (v \cdot \text{grad } u) u \, dx.$
\[ a(u, u) = \epsilon \int_{\Omega} \| \nabla u \|^2 \, dx > 0 \quad \forall u \in H^1_0(\Omega) \setminus \{0\} \quad (7.2.4) \]

From this and (7.2.3) we conclude existence and uniqueness of solutions of the BVP (7.2.1) in the Sobolev space \( H^1_0(\Omega) \).

### 7.2.1 Singular perturbation

Setting: fast-moving fluid \( \leftrightarrow \) convection dominates diffusion \( \leftrightarrow \) \( \epsilon \ll 1 \) in (7.2.1)
Example 7.2.5 (1D convection-diffusion boundary value problem).

\[ -\varepsilon \frac{d^2 u_\varepsilon}{dx^2} + \frac{du_\varepsilon}{dx} = 1 \quad \text{in } \Omega, \]

\[ u_\varepsilon(0) = 0, \quad u_\varepsilon(1) = 0, \]

\[ u_\varepsilon(x) = x + \frac{\exp(-x/\varepsilon) - 1}{1 - \exp(-1/\varepsilon)}. \]

For \( \varepsilon \ll 1 \):

boundary layer at \( x = 1 \)

Pointwise limit:

\[ \lim_{\varepsilon \to 0} u_\varepsilon(x) \to x \quad \forall 0 < x < 1. \]

"Limit problem": ignore diffusion \( \Rightarrow \) set \( \varepsilon = 0 \)

\[ \varepsilon = 0 \]

\[ \textbf{v}(x) \cdot \nabla u = f(x) \quad \text{in } \Omega. \]
Case $d = 1$ ($\Omega = ]0, 1[; v = \pm 1$)

\[ (7.2.6) \quad d=1 \quad \pm \frac{du}{dx}(x) = f(x) \quad \Rightarrow \quad u(x) = \int f \, dx + C. \quad (7.2.7) \]

What about this constant $C$?

If $v = 1$ $\leftrightarrow$ fluid flows “from left to right”, so we should integrate the source from 0 to $x$:

\[ u(x) = u(0) + \int_0^x f(s) \, ds = \int_0^x f(s) \, ds, \quad (7.2.8) \]

because $u(0) = 0$ by the boundary condition $u = 0$ on $\partial \Omega$. If $v = -1$ we start the integration at $x = 1$. Note that this makes the maximum principle of Thm. 7.1.15 hold.

For $d > 1$ we can solve (7.2.6) by the method of characteristics:

To motivate it, be aware that (7.2.6) describes pure transport of a temperature distribution in the velocity field $v$, that is, the heat/temperature is just carried along particle trajectories and changes only under the influence of heat sources/sinks along that trajectory.
Denote by $u$ the solution of (7.2.6) and recall the differential equation (7.1.1) for a particle trajectory

$$\frac{dy}{dt}(t) = v(y(t)) \quad , \quad y(0) = x_0 . \quad (7.1.1)$$

$$\frac{du(y(t))}{dt} = \nabla u(y(t)) \cdot \frac{dy}{dt}(t) = \nabla u \cdot v(y(t)) \quad \equiv \quad f(y(t)). \quad (7.2.6)$$

➢ Compute $u(y(t))$ by integrating source $f$ along particle trajectory!

$$u(y(t)) = u(x_0) + \int_0^t f(y(s)) \, ds \quad (7.2.9)$$

Taking the cue from $d = 1$ we choose $x_0$ as “the point on the boundary where the particle enters $\Omega$”. These points form the part of the boundary through which the flow enters $\Omega$, the inflow boundary

$$\Gamma_{\text{in}} := \{ x \in \partial \Omega : v(x) \cdot n(x) < 0 \} . \quad (7.2.10)$$

Its complement in $\partial \Omega$ contains the outflow boundary

$$\Gamma_{\text{out}} := \{ x \in \partial \Omega : v(x) \cdot n(x) > 0 \} . \quad (7.2.11)$$

**Remark 7.2.12 (Streamlines).**
In the case of closed streamlines the stationary pure transport problem fails to have a unique solution: on a closed streamline $u$ can attain “any” value, because there is no boundary value to fix $u$. 

Fig. 211
Return to case $d = 1$. In general solution $u(x)$ from (7.2.7) will not satisfy the boundary condition $u(1) = 0$! Also for $u(x)$ from (7.2.9) the homogeneous boundary conditions may be violated where the particle trajectory leaves $\Omega$!

In the limit case $\epsilon = 0$ not all boundary conditions of (7.2.1) can be satisfied.

**Notion 7.2.13** (Singularly perturbed problem).

A boundary value problem depending on parameter $\epsilon \approx \epsilon_0$ is called singularly perturbed, if the limit problem for $\epsilon \to \epsilon_0$ is not compatible with the boundary conditions.

Especially in the case of 2nd-order elliptic boundary value problems:

Singular perturbation = 1st-order terms become dominant for $\epsilon \to \epsilon_0$
In mathematical terms, singular perturbation for boundary values for PDEs is defined as a change of type of the PDE for $\epsilon = 0$: in the case of (7.2.1) the type changes from elliptic to hyperbolic, see Rem. 2.0.1.

7.2.2 Upwinding

Focus: linear finite element Galerkin discretization for 1D model problem, cf. Ex. 7.2.5

\[-\epsilon \frac{d^2 u}{dx^2} + \frac{du}{dx} = f(x) \quad \text{in } \Omega , \quad u(0) = 0 \quad , \quad u(1) = 0 . \quad (7.2.14)\]

Variational formulation, see Rem. 7.2.2:

\[u \in H^1_0(0, 1): \quad \epsilon \int_{0}^{1} \frac{du}{dx}(x) \frac{dv}{dx}(x) \text{dx} + \int_{0}^{1} \frac{du}{dx}(x) v(x) \text{dx} = \int_{0}^{1} f(x)v(x) \text{dx} \quad \forall v \in H^1_0(0, 1) . \]

\[=:a(u,v) \quad \text{=}:=l(v)\]
As in Sect. 1.5.1.2: use equidistant mesh $\mathcal{M}$ (mesh width $h > 0$), composite trapezoidal rule (1.5.59) for right hand side linear form, standard “tent function basis”, see (1.5.53).

linear system of equations for coefficients $\mu_i$, $i = 1, \ldots, M - 1$, providing approximations for point values $u(ih)$ of exact solution $u$.

\[
\left(-\frac{\epsilon}{h} - \frac{1}{2}\right) \mu_{i-1} + \frac{2\epsilon}{h} \mu_i + \left(-\frac{\epsilon}{h} + \frac{1}{2}\right) \mu_{i+1} = hf(ih), \quad i = 1, \ldots, M - 1, \quad (7.2.15)
\]

where the homogeneous Dirichlet boundary conditions are taken into account by setting $\mu_0 = \mu_M = 0$.

Remark 7.2.16 (Finite differences for convection-diffusion equation in 1D).

As in Sect. 1.5.3 on the finite difference in 1D, we can also obtain (7.2.15) by replacing the derivatives
by suitable difference quotients:

\[-\epsilon \frac{d^2 u}{dx^2} + \frac{du}{dx} = f(x)\]

\[\epsilon \left( \frac{-\mu_{i+1} + 2\mu_i - \mu_{i-1}}{h^2} \right) + \frac{\mu_{i+1} - \mu_{i-1}}{2h} = f(ih).\]

**Example 7.2.17** (Linear FE discretization of 1D convection-diffusion problem).

- Model boundary value problem *(7.2.14)*
- linear finite element Galerkin discretization as described above
- As in Ex. *7.2.5*: \( f \equiv 1 \)
For very small $\epsilon$: spurious oscillations of linear FE Galerkin solution.
In order to understand this observation, study the linear finite element Galerkin discretization in the limit case $\epsilon = 0$

\begin{equation}
\epsilon = 0 \quad \mu_{i+1} - \mu_{i-1} = 2hf(i\epsilon), \quad i = 1, \ldots, M. \tag{7.2.15}
\end{equation}

Linear system of equations with singular system matrix!

For $\epsilon > 0$ the Galerkin matrix will always be regular due to (7.2.4), but the linear relationship (7.2.18) will become more and more dominant as $\epsilon > 0$ becomes smaller and smaller. In particular, (7.2.18) sends the message that values at even and odd numbered nodes will become decoupled, which accounts for the oscillations.

Desired: robust discretization of (7.2.14)

= discretization that produces qualitatively correct solutions for any $\epsilon > 0$

(\ast): “qualitatively correct”, e.g., satisfaction of maximum principle, Thm. 7.1.15

Guideline:
Numerical methods for singularly perturbed problems must “work” for the limit problem $u' = f$ on an equidistant mesh of $\Omega := [0, 1]$?

What is a meaningful scheme for limit problem $u' = f$ on an equidistant mesh of $\Omega := [0, 1]$?

Explicit Euler method: $\mu_{i+1} - \mu_i = hf(\xi_i) \ i = 0, \ldots, N$,

Implicit Euler method: $\mu_{i+1} - \mu_i = hf(\xi_{i+1}) \ i = 0, \ldots, N$.

Use **one-sided difference quotients** for discretization of convective term!

Which type? (Explicit or implicit Euler?)

Linear system arising from **use of backward difference quotient**

$$\frac{du}{dx}|_{x=x_i} = \frac{\mu_i - \mu_{i-1}}{h} :$$

$$\left(-\frac{\epsilon}{h} - 1\right)\mu_{i-1} + \left(\frac{2\epsilon}{h} + 1\right)\mu_i + \frac{-\epsilon}{h}\mu_{i+1} = hf(ih), \ i = 1, \ldots, M - 1, \quad (7.2.19)$$

Linear system arising from **use of forward difference quotient**

$$\frac{du}{dx}|_{x=x_i} = \frac{\mu_{i+1} - \mu_i}{h} :$$

$$-\frac{\epsilon}{h}\mu_{i-1} + \left(\frac{2\epsilon}{h} - 1\right)\mu_i + \left(-\frac{\epsilon}{h} + 1\right)\mu_{i+1} = hf(ih), \ i = 1, \ldots, M - 1, \quad (7.2.20)$$
Example 7.2.21 (One-sided difference approximation of convective terms).

Model problem of Ex. 7.2.17, discretizations (7.2.19) and (7.2.20).

Only the discretization of $\frac{du}{dx}$ based on the backward difference quotient generates qualitatively correct (piecewise linear) discrete solutions (a “good method”).
If the forward difference quotient is used, the discrete solutions may violate the maximum principle of Thm. 7.1.15 (a “bad method”).

How can we tell a good method from a bad method by merely examining the system matrix?

Heuristic criterion for $\epsilon \to 0$-robust stability of nodal finite element Galerkin discretization/finite difference discretization of \textit{singularly perturbed} scalar linear convection-diffusion BVP (7.2.1) (with Dirichlet b.c.):

(Linearly interpolated) discrete solution satisfies maximum principle (5.7.3).

$\upharpoonright$ System matrix complies with sign-conditions (5.7.9)−(5.7.11).
Nodal finite element Galerkin discretization $\hat{=} \mu_i$ of Galerkin solution $u_N \in V_N$ double as point values of $u_N$ at interpolation nodes. This is satisfied for Lagrangian finite element methods ($\rightarrow$ Sect. 3.4) when standard nodal basis functions according to (3.4.3) are used.

Recall the sign-conditions (5.7.9)–(5.7.11) for the system matrix $A$ arising from nodal finite element Galerkin discretization or finite difference discretization:

- (5.7.9): positive diagonal entries, $\begin{align*}(A)_{ii} &> 0, \end{align*}$
- (5.7.10): non-positive off-diagonal entries, $\begin{align*}(A)_{ij} &\leq 0, \text{ if } i \neq j, \end{align*}$
- “(5.7.11)”: diagonal dominance, $\begin{align*}\sum_j (A)_{ij} &\geq 0. \end{align*}$

These conditions are met for *equidistant meshes in 1D*

- for the standard $S_1^0(\mathcal{M})$-Galerkin discretization (7.2.15), provided that $|\epsilon h^{-1}| \geq \frac{1}{2}$,
- when using *backward* difference quotients for the convective term (7.2.19) for any choice of $\epsilon \geq 0$, $h > 0$,
- when using *forward* difference quotients for the convective term (7.2.20), provided that $|\epsilon h^{-1}| \geq 1$. 

R. Hiptmair
C. Schwab,
H. Harbrecht
V. Gradinaru
A. Chernov
SAM, ETHZ

Numerical Methods for PDEs
Only the use of a *backward* difference quotient for the convective term guarantees the (discrete) maximum principle in an $\epsilon \to 0$-robust fashion!

**Terminology:** Approximation of $\frac{du}{dx}$ by *backward* difference quotients $\equiv$ upwinding

**Example 7.2.22** (Spurious Galerkin solution for 2D convection-diffusion BVP).

- Triangle domain $\Omega = \{(x, y) : 0 \leq x \leq 1, -x \leq y \leq x\}$.
- Velocity $v(x) = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ \begin{equation} \text{(7.2.1)} \end{equation} becomes $-\epsilon \Delta u + u_x = 1$.
- Exact solution: $u_\epsilon(x_1, x_2) = x - \frac{1}{1-e^{-1/\epsilon}}(e^{-(1-x_1)/\epsilon} - e^{-1/\epsilon})$, Dirichlet boundary conditions set accordingly.
- Standard Galerkin discretization by means of linear finite elements on sequence of triangular mesh created by regular refinement.
2D triangular mesh

Coarse initial mesh

# Vertices : 15,      # Elements : 16,      # Edges : 30

Exact solution

(\epsilon = 10^{-10})

Standard Galerkin solution on $x_2 = 0$-line

As expected:

spurious oscillations mar Galerkin solution

Difficulty observed in 1D also haunts discretization in higher dimensions.
7.2.2.1 Upwind quadrature

Revisit 1D model problem

\[-\epsilon \frac{d^2u}{dx^2} + \frac{du}{dx} = f(x) \quad \text{in } \Omega, \quad u(0) = 0, \quad u(1) = 0,\]

(7.2.14)

with variational formulation, see Rem. 7.2.2:

\[u \in H^1_0(0,1): \quad \epsilon \int_0^1 \frac{du}{dx}(x) \frac{du}{dx}(x) \, dx + \int_0^1 \frac{du}{dx}(x) v(x) \, dx = \int_0^1 f(x)v(x) \, dx \quad \forall v \in H^1_0(0,1).\]

Linear finite element Galerkin discretization on equidistant mesh \(\mathcal{M}\) with \(M\) cells, meshwidth \(h = \frac{1}{M}\).

cf. Sect. 1.5.1.2.
We opt for the composite trapezoidal rule

\[ \int_{0}^{1} \psi(x) \, dx \approx h \sum_{j=1}^{M-1} \psi(jh) , \quad \text{for} \quad \psi \in C^0([0, 1]), \psi(0) = \psi(1) = 0 . \]

for evaluation of convective term in bilinear form \( a \):

\[ \int_{0}^{1} \frac{du_N}{dx}(x) v_N(x) \, dx \approx h \sum_{j=1}^{M-1} \frac{du_N}{dx}(jh) v(hj) , \quad v_N \in S^0_{1,0}(M) . \]  

(7.2.23)

Note: this is not a valid formula, because \( \frac{du_N}{dx}(jh) \) is ambiguous, since \( \frac{du_N}{dx} \) is discontinuous at nodes of the mesh for \( u_N \in S^0_{1,0}(M) \)!

Up to now we resolved this ambiguity by the policy of local quadrature, see Sect. 3.5.4: quadrature rule applied locally on each cell with all information taken from that cell.

However: Convection transports information in the direction of \( \mathbf{v} \)!
Idea:
Use *upstream* information to evaluate $\frac{du_N}{dx}(jh)$ in (7.2.23)

$$
\frac{du_N}{dx}(jh) := \lim_{\delta \to 0} \frac{du_N}{dx}(jh - \delta) = \frac{du_N}{dx} |_{x_{j-1},x_j}.
$$

$\triangleq$ upwind quadrature

Upwind quadrature yields the following contribution of the discretized convective term to the linear system using the basis expansion $u_N = \sum_{l=1}^{M-1} \mu_l b_N^l$ into *locally supported* nodal basis functions ("tent functions")

$$
\int_0^1 \sum_{l=1}^{M-1} \mu_l \frac{db_N^l}{dx}(x) b_N^l(x) \, dx \approx h \frac{\mu_i - \mu_{i-1}}{h},
$$

where we used
• \( b_N^i(jh) = \delta_{ij} \), see (1.5.54),

• \( \left. \frac{d u_N}{d x} \right|_{x_j-1,x_j} = \frac{\mu_i - \mu_{i-1}}{h} \) from (1.5.55).

Linear system from upwind quadrature:

\[
\left( -\frac{\epsilon}{h} - 1 \right) \mu_{i-1} + \left( \frac{2\epsilon}{h} + 1 \right) \mu_i + -\frac{\epsilon}{h} \mu_{i+1} = h f(ih), \quad i = 1, \ldots, M - 1,
\]

which is the same as that obtained from a backward finite difference discretization of \( \frac{d u}{d x} \).

The idea of upwind quadrature can be generalized to \( d > 1 \): we consider \( d = 2 \) and linear Lagrangian finite element Galerkin discretization on triangular meshes, see Sect. 3.2.

Approximation of contribution of convective terms to bilinear form by means of \textit{global trapezoidal rule}:
\[
\int_{\Omega} (\mathbf{v} \cdot \text{grad} \ u) v \, dx \approx \sum_{p \in N(M)} \left( \frac{1}{3} \sum_{K \in U_p} |K| \right) \cdot \mathbf{v}(p) \cdot \text{grad} \ u(p) \cdot v(p).
\] (7.2.24)

ambiguous for \( u \in S^0_1(M) \)!

\[\text{notation: } U_p := \{ K \in M : p \in \overline{K} \}\]

\# Fix the ambiguous value of \( \mathbf{v}(p) \cdot \text{grad} \ u_N(p) \), \( u_N \in S^0_1(M) \), by taking the gradient from the triangle upstream to the node \( p \):
Idea: Use upstream/upwind information to evaluate $\nabla u_N(p)$ in (7.2.24)

$$v(p) \cdot \nabla u_N(p) := \lim_{\delta \to 0} v(p) \cdot \nabla u_N(p - \delta v(p)).$$  \hspace{1cm} (7.2.25)

$\hat{=} \text{general upwind quadrature}$

Note: By (7.1.1) the vector $v(p)$ supplies the direction of the streamline through $p$. Hence, $-v(p)$ is the direction from which information is “carried into $p$” by the flow.
Contribution of convective term to the $i$-th linear of the final linear system of equations (test function = tent function $b^i_N$)

$$\left( \frac{1}{3} \sum_{K \in U_i} |K| \right) \mathbf{v}(x^i) \cdot \text{grad} u_N |K_u$$

where $K_u$ is the upstream triangle of $p$. $\triangleright$

Using the expressions for the gradients of barycentric coordinate functions from Sect. 3.2.5 and the nodal basis expansion of $u_N$, we obtain for the convective contribution to the $i$-th line of the final linear system

$$\frac{U_i}{2|K_u|} \left( \begin{array}{c} - \|x^j - x^k\| n^i \cdot \mathbf{v}(x^i) \mu_i - \|x^i - x^j\| n^k \cdot \mathbf{v}(x^i) \mu_k - \|x^i - x^k\| n^j \cdot \mathbf{v}(x^i) \mu_j \end{array} \right)$$

$\leftrightarrow$ diagonal entry
By the very definition of the upstream triangle $K_u$ we find
\[
 n^i \cdot v(x^i) \leq 0, \quad n^k \cdot v(x^i) \geq 0, \quad n^j \cdot v(x^i) \geq 0.
\]

> sign conditions (5.7.9), (5.7.10) are satisfied, (5.7.11) is obvious.

**Example 7.2.26 (Upwind quadrature discretization).**

\[
\Omega = [0, 1]^2
\]
\[
-\epsilon \Delta u + \left( \begin{array}{c} 1 \\ 1 \end{array} \right) \cdot \text{grad} u = 0
\]

Dirichlet boundary conditions: $u(x, y) = 1$ for $x > y$ and $u(x, y) = 0$ for $x \leq y$

Limiting case ($\epsilon \to 0$): $u(x, y) = 1$ for $x > y$ and $u(x, y) = 0$ for $x \leq y$

layer along the diagonal from $\left( \begin{array}{c} 0 \\ 1 \end{array} \right)$ to $\left( \begin{array}{c} 1 \\ 0 \end{array} \right)$ in the limit $\epsilon \to 0$

linear finite element upwind quadrature discretization on triangular mesh.

Monitored: discrete solutions along diagonal from $\left( \begin{array}{c} 0 \\ 1 \end{array} \right)$ to $\left( \begin{array}{c} 1 \\ 0 \end{array} \right)$ for $\epsilon = 10^{-10}$. 
Upwind quadrature scheme respects maximum principle, whereas the standard Galerkin solution is rendered useless by spurious oscillations.
7.2.2.2 Streamline diffusion

We take another look at the 1D upwind discretization of (7.2.14) and view it from a different perspective.

1D upwind (finite difference) discretization of (7.2.14):

\[
\begin{align*}
\left(-\frac{\epsilon}{h} - 1\right) \mu_{i-1} + \left(\frac{2\epsilon}{h} + 1\right) \mu_i - \frac{\epsilon}{h} \mu_{i+1} &= hf(ih) \quad i = 1, \ldots, M - 1 .
\end{align*}
\]

\[
\begin{align*}
\epsilon + h/2 \quad &\frac{-\mu_{i-1} + 2\mu_i - \mu_{i+1}}{h^2} + \frac{-\mu_{i-1} + \mu_{i+1}}{2h} = f(ih) ,
\end{align*}
\]

for \( i = 1, \ldots, M - 1 \).

Upwinding = \( h \)-dependent enhancement of diffusive term

artificial diffusion/viscosity
We also observe that the upwinding strategy just adds the \textit{minimal amount of diffusion} to make the resulting system matrix comply with the conditions (5.7.9)–(5.7.11), which ensure that the discrete solution satisfies the maximum principle.

\textbf{Issue:} \quad How to extend the trick of adding artificial diffusion to $d > 1$ ?

Well, just add an extra $h$-dependent multiple of $-\Delta$! Let’s try.

\textit{Example} 7.2.27 (Effect of added diffusion).

Convection-diffusion boundary value problem ((7.2.1) with $\mathbf{v} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$)

$$-\epsilon \Delta u + \frac{\partial u}{\partial x_1} = 0 \quad \text{in } \Omega = ]0, 1[^2, \quad u = g \quad \text{on } \partial \Omega.$$ 

Here, Dirichlet data $g(\mathbf{x}) = 1 - 2|\mathbf{x}_2 - \frac{1}{2}|$.

Thus, for $\epsilon \approx 0$ we expect $u \approx g$, because the Dirichlet data are just transported in $x_1$-direction and there are no boundary layers.
Stronger diffusion leads to “smearing” of features that the flow field transports into the interior of the domain.

(Too much) artificial diffusion  $\Rightarrow$ smearing of internal layers

(We are no longer solving the right problem!)
Remark 7.2.28 (Internal layers).

Pure transport problem:
\[ \mathbf{v} \cdot \nabla u = 0 \quad \text{in } \Omega, \]
where \( \Omega = ]0, 1[^2, \mathbf{v} = \begin{pmatrix} 2 \\ 1 \end{pmatrix}, \epsilon = 10^{-4}, \)

Dirchlet b.c. that can only partly be fulfilled: \( u = 1 \) on \( \{ x_1 = 0 \} \cup \{ x_2 = 1 \} \), \( u = 0 \) on \( \{ x_1 = 1 \} \cup \{ x_2 = 0 \} \)

Solution of pure transport problem with discontinuous boundary data

- displays a discontinuity across the streamline emanating from the point of discontinuity on \( \partial \Omega \),
- is \textit{smooth along streamlines}. 

Qualitative solution of

\[-\delta \Delta + \mathbf{v} \cdot \text{grad} \, u = 0 \quad \text{in } \Omega,\]

with \(\delta > 0\), the same boundary data

Smearing of internal layer!

We would also find a boundary layer which is omitted in the figure.

Heuristics: If the solution is smooth along streamlines, then adding diffusion in the direction of streamlines cannot do much harm.

What does “diffusion in a direction” mean?
Think of a generalized Fourier's law (2.5.3) for \( d = 2 \), e.g.,

\[
\mathbf{j}(\mathbf{x}) = -\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \nabla u(\mathbf{x}) .
\]

This means, only a temperature variation in \( x_1 \)-direction triggers a heat flow.

Diffusion in a direction \( \mathbf{v} \in \mathbb{R}^2 \)

\[
\mathbf{j}(\mathbf{x}) = -\mathbf{v} \mathbf{v}^T \nabla u(\mathbf{x}) \tag{7.2.29}
\]

Such an extended Fourier's law is an example of anisotropic diffusion.

Anisotropic diffusion can simply be taken into account in variational formulations and Galerkin discretization by replacing the heat conductivity \( \kappa \)/stiffness \( \sigma \) with a symmetric, positive (semi-)definite matrix, the diffusion tensor.

Idea: **Anisotropic artificial diffusion** in streamline direction

On cell \( K \) replace:

\[
\epsilon \leftarrow \epsilon \mathbf{I} + \delta_K \hat{\mathbf{v}}_K \hat{\mathbf{v}}_K^T \in \mathbb{R}^{2 \times 2}.
\]

\( \hat{\mathbf{v}}_K \triangleq \) local velocity (e.g., obtained by averaging)

\( \delta_K > 0 \triangleq \) method parameter controlling the strength of anisotropic diffusion
This idea underlies the so-called **streamline-diffusion method**.

Thus, (for the model problem) Galerkin discretization may target the variational problem

\[
\int_{\Omega} \left( \epsilon I + \delta_K v_K v_K^T \right) \nabla u \cdot \nabla v + v(x) \cdot \nabla u v \, dx = \int_{\Omega} f v \, dx \quad \forall v \in H^1_0(\Omega). \tag{7.2.30}
\]

This tampering affects the solution \( u \)

(solution of (7.2.30) \( \neq \) solution of (7.2.1))

Desirable: Maintain **consistency** of variational problem!

**Definition 7.2.31** (Consistent modifications of variational problems).

A variational problem is called a **consistent modification** of another, if both possess the same (unique) solution(s).
Note: the variational crimes investigated in Sect. 5.5 represent non-consistent modifications.

Ensuring consistency for streamline-upwind variational problem:

Idea: Add anistropic diffusion through a residual term that vanishes for the exact solution $u$

streamline-upwind variational problem: given mesh $\mathcal{M}$ seek $u \in H^1_0(\Omega) \cap H^2(\mathcal{M})$

$$
\int_{\Omega} \epsilon \nabla u \cdot \nabla v + v(x) \cdot \nabla u v \, dx
+ \sum_{K \in \mathcal{M}} \delta_K \int_K (-\epsilon \Delta u + v \cdot \nabla u - f) \cdot (v \cdot \nabla v) \, dx = \int_{\Omega} fv \, dx \quad \forall v \in H^1_0(\Omega). \tag{7.2.32}
$$

Note that enhanced smoothness of $u$, namely in addition $u \in H^2(K)$ for all $K \in \mathcal{M}$, is required to render (7.2.32) meaningful ($\rightarrow$ Sobolev space $H^2(\mathcal{M})$).
Note: in the case of Galerkin discretization based on $V_{N,0} = S_1^0(\mathcal{M})$, we find $\Delta u_N = 0$ in each $K \in \mathcal{M}$.

For Galerkin discretization of (7.2.32) by means of linear Lagrangian finite elements, the local control parameters $\delta_K$ are usually chosen according to the rule

$$
\delta_K := \begin{cases} 
\epsilon^{-1} h_K^2, & \text{if } \frac{||v||_{K,\infty} h_K}{2\epsilon} \leq 1, \\
h_K, & \text{if } \frac{||v||_{K,\infty} h_K}{2\epsilon} > 1.
\end{cases}
$$

which is suggested by theoretical investigations and practical experience.

**Example 7.2.33** (Streamline-diffusion discretization).

Exactly the same setting as in Ex. 7.2.26 with the upwind quadrature approach replaced with the streamline diffusion method.

**Code 7.2.34**: Assembling SUPG stabilization part of element matrix in LehrFEM

```matlab
function Aloc = STIMASUPGLFE(Vertices, flag, QuadRule, VHandle, a, d1, d2, varargin)
```
% ALOC = STIMA_SUPG_LFE(VERTICES) provides the extra terms for SUPG stabilization to be
% added to the Galerkin element matrix for linear finite elements

% VERTICES is 3-by-2 matrix specifying the vertices of the current element
% in a row wise orientation.

% a: diffusivity
% d1 d2: apriori chosen constants for SUPG-modification

% Flag not used, needed for interface to assemMat_LFE

% QUADRULE is a struct, which specifies the Gauss quadrature that is used
% to do the numerical integration:
% W Weights of the Gauss quadrature.
% X Abscissae of the Gauss quadrature.e:

% VHANDLE is function handle for velocity field

% Preallocate memory for element matrix
Aloc = zeros(3,3);

% Analytic computation of entries of element matrix using barycentric
% coordinates, see Sect. 3.2.5
l1x = Vertices(2,2)-Vertices(3,2);
l1y = Vertices(3,1)-Vertices(2,1);
l2x = Vertices(3,2)-Vertices(1,2);
l2y = Vertices(1,1)-Vertices(3,1);
l3x = Vertices(1,2)-Vertices(2,2);
l3y = Vertices(2,1)-Vertices(1,1);

% Compute element mapping
P1 = Vertices(1,:);
P2 = Vertices(2,:);
P3 = Vertices(3,:);

BK = [ P2 - P1 ; P3 - P1 ];  % transpose of transformation matrix

det_BK = abs(det(BK));  % twice the area of the triangle

nPoints = size(QuadRule.w,1);

% Quadrature points in actual element stored as rows of a matrix
x = QuadRule.x*BK + ones(nPoints,1)*P1;

c = VHandle(x,varargin{:});
% Entries of anisotropic diffusion tensor
FHandle = [c(:,1).*c(:,1) c(:,1).*c(:,2) c(:,2).*c(:,1) c(:,2).*c(:,2)];

% Compute local PecletNumber for SUPG control parameter
hK = max([norm(P2-P1), norm(P3-P1), norm(P2-P3)]);

v_infK = max(abs(c(:))); PK = v_infK * hK / (2 * a);

% Apply quadrature rule and fix constant part
w = QuadRule.w; e = sum((FHandle.*[w w w w]), 1);

if (PK <= 1), Aloc = d1 * hK^2 / a * Aloc;
else Aloc=d2*hK*Aloc; end

return
Observations:

- The streamline upwind method does not exactly respect the maximum principle, but offers a better resolution of the internal layer compared with upwind quadrature (Parlance: streamline diffusion method is “less diffusive”).

Example 7.2.35 (Convergence of SUPG and upwind quadrature FEM).

- $\Omega = ]0, 1[^2$, model problem (7.2.1), $\mathbf{v}(\mathbf{x}) = (\frac{2}{3})$, right hand side $\mathbf{f}$ such that
  
  $$u_\varepsilon(x, y) = xy^2 - y^2 e^{\frac{2x-1}{\varepsilon}} - xe^{3\frac{y-1}{\varepsilon}} + e^{2\frac{x-1}{\varepsilon}+3\frac{y-1}{\varepsilon}}.$$

- Finite element discretization, $V_{0,N} = S_1^0(\mathcal{M})$ und sequence of unstructured triangular “uniform” meshes, with
  
  - upwind quadrature stabilization from Sect. 7.2.2.1,
  - SUPG stabilization according to (7.2.32).
Monitored: (Approximate) $L^2(\Omega)$-norm of discretization error (computed with high-order local quadrature)

$u_\epsilon$ for $\epsilon = 1$

Convergence for $\epsilon = 1$

Observation: SUPG stabilization does not affect $O(h_M^2)$-convergence of $\|u - u_N\|_{L^2(\Omega)}$ for $h$-refinement and $h_M \to 0$, whereas upwind quadrature leads to worse $O(h_M)$ convergence of the $L^2$-error norm.
7.3 Transient convection-diffusion BVP

Sect. [7.1.4] introduced the transient heat conduction model in a fluid, whose motion is described by a non-stationary velocity field (→ Sect. [7.1.1]) \( \mathbf{v} : \Omega \times [0,T[ \mapsto \mathbb{R}^d \)

\[
\frac{\partial}{\partial t}(\rho u) - \text{div}(\kappa \text{grad } u) + \text{div}(\rho \mathbf{v}(x,t) u) = f(x,t) \quad \text{in } \tilde{\Omega} := \Omega \times ]0,T[, \quad (7.1.16)
\]

where \( u = u(x,t) : \tilde{\Omega} \mapsto \mathbb{R} \) is the unknown temperature.

Assuming \( \text{div } \mathbf{v}(x,t) = 0 \), as in Sect. [7.2], by scaling we arrive at the model equation for transient convection-diffusion

\[
\frac{\partial u}{\partial t} - \epsilon \Delta u + \mathbf{v}(x,t) \cdot \text{grad } u = f \quad \text{in } \tilde{\Omega} := \Omega \times ]0,T[, \quad (7.3.1)
\]
supplemented with

- **Dirichlet boundary conditions:** \( u(x, t) = g(x, t) \quad \forall x \in \partial \Omega , \quad 0 < t < T \),
- **initial conditions:** \( u(x, 0) = u_0(x) \quad \forall x \in \Omega \).

### 7.3.1 Method of lines

For the solution of IBVP (7.3.1) follow the general policy introduced in Sect. 6.1.3:

1. **Discretization in space on a fixed mesh** \( \Rightarrow \) initial value problem for ODE
2. **Discretization in time** (by suitable numerical integrator \( \Rightarrow \) timestepping)
For instance, in the case of Dirichlet boundary conditions,

\[
\begin{cases}
\frac{\partial u}{\partial t} - \epsilon \Delta u + \mathbf{v}(\mathbf{x}, t) \cdot \nabla u = f & \text{in } \tilde{\Omega} := \Omega \times [0, T], \\
u(\mathbf{x}, t) = g(\mathbf{x}, t) & \forall \mathbf{x} \in \partial \Omega, 0 < t < T, \quad u(\mathbf{x}, 0) = u_0(\mathbf{x}) & \forall \mathbf{x} \in \Omega.
\end{cases}
\]  
(7.3.2)

\[\leftarrow \text{spatial discretization} \]

\[
M \frac{d\tilde{\mu}}{dt}(t) + \epsilon A \tilde{\mu}(t) + B \tilde{\mu}(t) = \varphi(t),
\]  
(7.3.3)

where

- \( \tilde{\mu} = \tilde{\mu}(t) : [0, T] \mapsto \mathbb{R}^N \) \( \cong \) coefficient vector describing approximation \( u_N(t) \) of \( u(\cdot, t) \),

- \( A \in \mathbb{R}^{N,N} \) \( \cong \) s.p.d. matrix of discretized \(-\Delta\), e.g., (finite element) Galerkin matrix,

- \( M \in \mathbb{R}^{N,N} \) \( \cong \) (lumped \( \rightarrow \) Rem. 6.2.34) mass matrix

- \( B \in \mathbb{R}^{N,N} \) \( \cong \) matrix for discretized convective term, e.g., Galerkin matrix, upwind quadrature matrix (\( \rightarrow \) Sect. 7.2.2.1), streamline diffusion matrix (\( \rightarrow \) Sect. 7.2.2.2).
Example 7.3.4 (Implicit Euler method of lines for transient convection-diffusion).

1D convection-diffusion IBVP:

\[
\begin{align*}
\frac{\partial u}{\partial t} - \epsilon \frac{\partial^2 u}{\partial x^2} + \frac{\partial u}{\partial x} &= 0, \\
u(x, 0) &= \max(1 - 3|x - \frac{1}{3}|, 0), \\
u(0) &= \nu(1) = 0.
\end{align*}
\]  

(7.3.5)

Spatial discretization on equidistant mesh with meshwidth \( h = 1/N \):

1. central finite difference scheme, see (7.2.15),
2. upwind finite difference discretization, see (7.2.19),

\( M = hI \) (“lumped” mass matrix, see Rem. 6.2.34),

Temporal discretization with uniform timestep \( \tau > 0 \):

1. implicit Euler method, see (6.1.23),
2. explicit Euler method, see (6.1.22),

Computations with \( \epsilon = 10^{-5} \), implicit Euler discretization, \( h = 0.01 \), \( \tau = 0.00125 \):
Observation:

- Central finite differences display spurious oscillations as in Ex. 7.2.17.
- Upwinding suppresses spurious oscillations, but introduces *spurious damping*.

Computations with $\epsilon = 10^{-5}$, spatial upwind discretization, $h = 0.01$, $\tau = 0.005$: 
Observation: implicit Euler time-stepping causes stronger spurious damping than explicit Euler time-stepping.

However, explicit Euler subject to tight stability induced timestep constraint for larger values of $\epsilon$, see Sect. 6.1.4.2.
Advice for spatial discretization for method of lines approach

Use $\epsilon$-robustly stable spatial discretization of convective term.

**Remark 7.3.6** (Choice of timestepping for m.o.l. for transient convection-diffusion).

If $\epsilon$-robustness for all $\epsilon > 0$ (including $\epsilon > 1$) desired $\Rightarrow$ Arguments of Sect. 6.1.4.2 stipulate use of $L(\pi)$-stable ($\rightarrow$ Def. 6.1.56) timestepping methods (implicit Euler (6.1.23), RADAU-3 (6.1.58), SDIRK-2 (6.1.59))

In the *singly perturbed case* $0 < \epsilon \ll 1$ conditionally stable explicit timestepping is an option, due to a timestep constraint of the form $\tau < O(hM)$, which does not interfere with efficiency, cf. the discussion in Sect. 6.1.5.
7.3.2 Transport equation

Focus on the situation of singular perturbation (→ Def. 7.2.13): \( 0 < \epsilon \ll 1 \)

迦 study limit problem (as in Sect. 7.2.1)

\[
\frac{\partial u}{\partial t} - \epsilon \Delta u + \mathbf{v}(x,t) \cdot \nabla u = f \quad \text{in } \tilde{\Omega} := \Omega \times ]0, T[, \\
\]

\[
\leftarrow \epsilon = 0 \\
\frac{\partial u}{\partial t} + \mathbf{v}(x,t) \cdot \nabla u = f \quad \text{in } \tilde{\Omega} := \Omega \times ]0, T[. \tag{7.3.7}
\]

transport equation
Now: focus on case \( f \equiv 0 \) (no sources)

Let \( u = u(x, t) \) be a \( C^1 \)-solution of

\[
\frac{\partial u}{\partial t} + \mathbf{v}(x, t) \cdot \nabla u = 0 \quad \text{in } \widetilde{\Omega} := \Omega \times ]0, T[.
\]

(7.3.8)

Recall: for the stationary pure transport problem (7.2.6) we found solutions by integrating the source term along streamlines (following the flow direction).

\[ t \mapsto u(y(t), t), \quad \text{where } y(t) \text{ solves } \frac{dy}{dt}(t) = \mathbf{v}(y(t), t), \quad \text{see (7.1.1)} \]

By the chain rule

\[
\frac{d}{dt} u(y(t), t) = \nabla u(y(t), t) \cdot \frac{dy}{dt}(t) + \frac{\partial u}{\partial t}(y(t), t)
\]

(7.3.9)

\[
= \nabla u(y(t), t) \cdot \mathbf{v}(y(t), t) + \frac{\partial u}{\partial t}(y(t), t) \quad (7.3.8)
\]

\[
= 0.
\]

\[
\text{A fluid particle “sees” a constant temperature!}
\]

Remark 7.3.10 (Solution formula for sourceless transport).
Situation: *no inflow/outflow* (e.g., fluid in a container)

\[ \mathbf{v}(\mathbf{x}, t) \cdot \mathbf{n}(\mathbf{x}) = 0 \quad \forall \mathbf{x} \in \partial \Omega, \ 0 < t < T. \] (7.1.2)

▷ all streamlines will “stay inside \( \Omega \)”, flow map \( \Phi^t \) (7.1.3) defined for all times \( t \in \mathbb{R} \).

Initial value problem:

\[ \mathbf{v}(\mathbf{x}, t) \cdot \nabla u = 0 \quad \text{in} \ \tilde{\Omega}, \quad u(\mathbf{x}, 0) = u_0(\mathbf{x}) \quad \forall \mathbf{x} \in \Omega. \]

Exact solution

\[ u(\mathbf{x}, t) = u_0(\mathbf{x}_0(\mathbf{x}, t)), \] (7.3.11)

where \( \mathbf{x}_0(\mathbf{x}, t) \) is the position at time 0 of the fluid particle that is located at \( \mathbf{x} \) at time \( t \).
This solution formula can be generalized to any divergence free velocity field $\mathbf{v} : \Omega \mapsto \mathbb{R}^d$ and $f \neq 0$. The new aspect is that streamlines can \textit{enter} and \textit{leave} the domain $\Omega$. In the former case the solution value is given by a “transported boundary value”:

$$\frac{d}{dt} u(y(t)) = f(y(t), t)$$

\[ u(x, t) = \begin{cases} 
  u_0(x_0) + \int_0^t f(y(s), s) \, ds, & \text{if } y(s) \in \Omega \quad \forall 0 < s < t, \\
  g(y(s_0), s_0) + \int_{s_0}^t f(y(s), s) \, ds, & \text{if } y(s_0) \in \partial \Omega, y(s) \in \Omega \quad \forall s_0 < s < t, 
\end{cases} \tag{7.3.12} \]

where we have assumed \textbf{Dirichlet boundary conditions} on the inflow boundary

$$u(x, t) = g(x, t) \quad \text{for } x \in \Gamma_{\text{in}} := \{ x \in \partial \Omega : \mathbf{v}(x) \cdot \mathbf{n}(x) < 0 \} \quad \text{cf. (7.2.10).}$$
7.3.3 Lagrangian split-step method

Lagrangian discretization schemes for the IBVP (7.3.2) are inspired by insight into the traits of solutions of pure transport problems.

The variant that we are going to study separates the transient convection-diffusion problem into a pure diffusion problem (heat equation → Sect. 6.1.1) and a pure transport problem (7.3.7). This is achieved by means of a particular approach to timestepping.

7.3.3.1 Split-step timestepping

Abstract perspective: consider ODE, whose right hand side is the sum of two (smooth) functions

\[ \dot{y} = g(t, y) + h(t, y) , \quad g, h : \mathbb{R}^m \mapsto \mathbb{R}^m . \]  

(7.3.13)

There is an abstract timestepping scheme that offers great benefits if one commands efficient methods to solve initial value problems for both \( \dot{z} = g(z) \) and \( \dot{w} = h(w) \).
Strang splitting: single step method for (7.3.13), timestep \( \tau := t_j - t_{j-1} > 0 \): compute \( y^{(j)} \approx y(t_j) \) from \( y^{(j-1)} \approx y(t_{j-1}) \) according to

\[
\tilde{y} := z(t_{j-1} + \frac{1}{2} \tau) , \quad \text{where} \quad z(t) \quad \text{solves} \quad \dot{z} = g(t, z), \quad z(t_{j-1}) = y^{(j-1)}, \quad (7.3.14)
\]

\[
\hat{y} := w(t_j) \quad \text{where} \quad w(t) \quad \text{solves} \quad \dot{w} = h(t, w), \quad w(t_{j-1}) = \tilde{y}, \quad (7.3.15)
\]

\[
y^{(j)} := z(t_j) , \quad \text{where} \quad z(t) \quad \text{solves} \quad \dot{z} = g(t, z), \quad z(t_{j-1} + \frac{1}{2} \tau) = \hat{y}. \quad (7.3.16)
\]
One timestep involves three sub-steps:

1. Solve $\dot{z} = g(t, z)$ over time $[t_{j-1}, t_{j-1} + \frac{1}{2} \tau]$ using the result of the previous timestep as initial value $\leftrightarrow (7.3.14)$.

2. Solve $\dot{w} = h(t, w)$ over time $\tau$ using the result of 1 as initial value $\leftrightarrow (7.3.15)$.

3. Solve $\dot{z} = g(t, z)$ over time $[t_{j-1} + \frac{1}{2} \tau, t_j]$ using the result of 2 as initial value $\leftrightarrow (7.3.16)$.

**Theorem 7.3.17** (Order of Strang splitting single step method).

*Assuming exact solution of the initial value problems of the sub-steps, the Strang splitting single step method for *(7.3.13)* is of second order.*

Fig. 233

This applies to Strang splitting timestepping for initial value problems for ODEs. Now we boldly regard *(7.3.2)* as an “*ODE in function space*” for the unknown “function space valued function” $u = u(t)$.
\[ [0, T] \mapsto H^1(\Omega) \]

\[
\frac{du}{dt} = \epsilon \Delta u + f - \mathbf{v} \cdot \nabla u
\]

\[
\dot{y} = g(y) + h(y)
\]

Formally, we arrive at the following “timestepping scheme in function space” on a temporal mesh \( 0 = t_0 < t_1 < \cdots < t_M := T \) for (7.3.1):

Given approximation \( u^{(j-1)} \approx u(t_{j-1}) \),

① Solve (autonomous) IBVP for pure diffusion from \( t_{j-1} \) to \( t_{j-1} + \frac{1}{2}\tau \)

\[
\frac{\partial w}{\partial t} - \epsilon \Delta w = 0 \quad \text{in} \quad \Omega \times ]t_{j-1}, t_{j-1} + \frac{1}{2}\tau[ , \]

(7.3.14) \( \iff \)

\[
w(x, t) = g(x, t_{j-1}) \quad \forall x \in \partial \Omega, \ t_{j-1} < t < t_{j-1} + \frac{1}{2}\tau , \quad (7.3.18)
\]

\[
w(x, t_{j-1}) = u^{(j-1)}(x) \quad \forall x \in \Omega .
\]
② Solve IBVP for *pure transport* (= advection), see Sect. 7.3.2,

\[
\frac{\partial z}{\partial t} + \mathbf{v}(x, t) \cdot \nabla z = f(x, t) \quad \text{in } \Omega \times [t_{j-1}, t_j],
\]

(7.3.15) \iff \begin{align*}
\frac{\partial z}{\partial t} + \mathbf{v}(x, t) \cdot \nabla z &= f(x, t) \quad \text{in } \Omega \times [t_{j-1}, t_j], \\
\frac{\partial z}{\partial t} &= 0 \quad \text{on inflow boundary } \Gamma_{in}, t_{j-1} < t < t_j,
\end{align*}

\begin{align*}
z(x, t) &= g(x, t) \quad \forall x \in \Omega, t_{j-1} < t < t_j, \\
z(x, t_{j-1}) &= w(x, t_{j-1} + \frac{1}{2} \tau) \quad \forall x \in \Omega.
\end{align*}

(7.3.19)

③ Solve IBVP for *pure diffusion* from \(t_{j-1} + \frac{1}{2} \tau\) to \(t_j\)

\[
\frac{\partial w}{\partial t} - \epsilon \Delta w = 0 \quad \text{in } \Omega \times [t_{j-1} + \frac{1}{2} \tau, t_j],
\]

(7.3.16) \iff \begin{align*}
\frac{\partial w}{\partial t} - \epsilon \Delta w &= 0 \quad \text{in } \Omega \times [t_{j-1} + \frac{1}{2} \tau, t_j], \\
w(x, t) &= g(x, t_j) \quad \forall x \in \partial \Omega, \ t_{j-1} + \frac{1}{2} \tau < t < t_j, \\
w(x, t_{j-1} + \frac{1}{2} \tau) &= z(x, t_j) \quad \forall x \in \Omega.
\end{align*}

(7.3.20)

Then set \(u^{(j)}(x) := w(x, t_j), \ x \in \Omega.\)

Efficient “implementation” of Strang splitting time-stepping, if \(g = g(y)\):

combine last sub-step ③ with first sub-step ① of the next timestep
Remark 7.3.21 (Approximate sub-steps for Strang splitting time).

The solutions of the initial value problems in the sub-steps of Strang splitting timestepping may be computed only approximately.

If this is done by one step of a 2nd-order timestepping method in each case, then the resulting approximate Strang splitting timestepping will still be of second order, cf. Thm. 7.3.17.
7.3.3.2 Particle method for advection

Recall the discussion of the IBVP for the pure transport (= advection) equation from Sect. 7.3.2

\[
\frac{\partial u}{\partial t} + \mathbf{v}(x, t) \cdot \nabla u = f \quad \text{in } \tilde{\Omega} := \Omega \times ]0, T[ ,
\]

\[
u(x, t) = g(x, t) \quad \text{on } \Gamma_{\text{in}} \times ]0, T[ ,
\]

\[
u(x, 0) = u_0(x) \quad \text{in } \Omega ,
\]

with inflow boundary

\[
\Gamma_{\text{in}} := \{ x \in \partial \Omega : \mathbf{v}(x) \cdot \mathbf{n}(x) < 0 \} .
\]

Case \( f \equiv 0 \): a travelling fluid particle sees a constant solution, see (7.3.9)

\[
u(x, t) = \begin{cases} 
u_0(x_0) \quad , \text{if } y(s) \in \Omega \quad \forall 0 < s < t , \\ g(y(s_0), s_0) \quad , \text{if } y(s_0) \in \partial \Omega, y(s) \in \Omega \quad \forall s_0 < s < t , \end{cases}
\]

where \( s \mapsto y(s) \) solves the initial value problem \( \frac{dy}{ds}(s) = \mathbf{v}(y(s), s) \), \( y(t) = x \) ("backward particle trajectory").
Case of general $f$, see Rem. 7.3.10: Since $\frac{d}{dt}u(y(t)) = f(y(t), t)$

$u(x, t) = \begin{cases} 
  u_0(x_0) + \int_0^t f(y(s), s) \, ds, & \text{if } y(s) \in \Omega \quad \forall 0 < s < t, \\
  g(y(s_0), s_0) + \int_{s_0}^t f(y(s), s) \, ds, & \text{if } y(s_0) \in \partial\Omega, y(s) \in \Omega \quad \forall s_0 < s < t.
\end{cases}$

(7.3.12)

The solution formula (7.3.12) suggests an approach for solving (7.3.22) approximately.

We first consider the simple situation of no inflow/outflow (e.g., fluid in a container, see Rem. 7.3.10)

$$v(x, t) \cdot n(x) = 0 \quad \forall x \in \partial\Omega, \ 0 < t < T.$$  

(7.1.2)

① Pick suitable interpolation nodes $\{p_i\}_{i=1}^N \subset \Omega$ (initial ‘particle positions”)
② Solve initial value problems (cf. ODE (7.1.1) for particle trajectories)

\[
\dot{y}(t) = \mathbf{v}(y(t), t), \quad y(0) = \mathbf{p}_i, \quad i = 1, \ldots, N,
\]

by means of a suitable single-step method with uniform timestep \( \tau := T/M, M \in \mathbb{N} \).

 sequencies of solution points \( \mathbf{p}_i^{(j)}, j = 0, \ldots, M, i = 1, \ldots, N \)

③ Reconstruct approximation \( u_N^{(j)} \approx u(\cdot, t_j), t_j := j \tau \), by interpolation:

\[
u_N^{(j)}(\mathbf{p}_i^{(j)}) := u_0(\mathbf{p}_i) + \tau \sum_{l=1}^{j-1} f\left(\frac{1}{2}(\mathbf{p}_i^{(l)} + \mathbf{p}_i^{(l-1)}), \frac{1}{2}(t_l + t_{l-1})\right), \quad i = 1, \ldots, N
\]

where the composite midpoint quadrature rule was used to approximate the source integral in (7.3.12).

This method falls into the class of

- **particle methods**, because the interpolation nodes can be regarded fluid particles tracked by the method,
- **Lagrangian methods**, which treat the IBVP in coordinate systems moving with the flow,
- **characteristic methods**, which reconstruct the solution from knowledge about its behavior along streamlines.
For general velocity field \( \mathbf{v} : \Omega \mapsto \mathbb{R}^d \):

- Stop tracking \( i \)-th trajectory as soon as an interpolation nodes \( p_i^{(j)} \) lies outside spatial domain \( \Omega \).
- In each timestep start new trajectories from fixed locations on inflow boundary \( \Gamma_{\text{in}} \) ("particle injection"). These interpolation nodes will carry the boundary value.

**Example 7.3.24** (Point particle method for pure advection).

- **IBVP (7.3.22)** on \( \Omega = [0,1]^2 \), \( T = 2 \), with \( f \equiv 0 \), \( g \equiv 0 \).
- Initial locally supported bump \( u_0(\mathbf{x}) = \max\{0, 1 - 4 \| \mathbf{x} - (1/4) \| \} \).
- Two stationary divergence-free velocity fields
  - \( \mathbf{v}_1(\mathbf{x}) = \begin{pmatrix} -\sin(\pi x_1) \cos(\pi x_2) \\ \cos(\pi x_1) \sin(\pi x_2) \end{pmatrix} \) satisfying (7.1.2),
  - \( \mathbf{v}_2(\mathbf{x}) = \begin{pmatrix} -x_2 \\ x_1 \end{pmatrix} \).
Initial positions of interpolation points on regular tensor product grid with meshwidth $h = \frac{1}{40}$.

Approximation of trajectories by means of explicit trapezoidal rule [18, Eq. 12.4.6] (method of Heun).

velocity field $\mathbf{v}_1$ (circvel)

velocity field $\mathbf{v}_2$ (rotvel)
Code 7.3.25: Confined velocity field

```matlab
function V = circvel(P)
% Circular velocity (divergence free, zero normal component on unit square).
% P: 2xN matrix of point coordinates
% return value: velocity vectors at points in P

v = @(p) [-sin(pi*p(1))*cos(pi*p(2)); sin(pi*p(2))*cos(pi*p(1))];
V = [];
for p=P
    V = [V, v(p)];
end
```

Code 7.3.26: Pass-through velocity field

```matlab
function V = rotvel(P)
% Circular velocity

v = @(p) [-p(2); p(1)];
V = [];
for p=P
    V = [V, v(p)];
end
```
Code 7.3.27: Point particle method for pure advection

```matlab
function partadv(v,u0,g,n,tau,m)

% Point particle method for pure advection problem
% on the unit square
% v: handle to a function returning the velocity field for (an array) of points
% u0: handle to a function returning the initial value u0 for (an array) of points
% g: handle to a function g = g(x) returning the Dirichlet boundary values
% n: h = 1/n is the grid spacing of the initial point distribution
% tau: timestep size, m: number of timesteps, that is, T = m\tau

% Initialize points
h = 1/n; [Xp,Yp] = meshgrid(0:h:1,0:h:1);
P = [reshape(Xp,1,(n+1)^2); reshape(Yp,1,(n+1)^2)];

% Initialize points on the boundary
BP = [(0:h:1);zeros(1,n+1)], [ones(1,n+1);(0:h:1)],...
    [(0:h:1);ones(1,n+1)], [zeros(1,n+1);(0:h:1)];
U = u0(P); % Initial values

% Plot velocity field
hp = 1/10; [Xp,Yp] = meshgrid(0:hp:1,0:hp:1);
Up = zeros(size(Xp)); Vp = zeros(size(Xp));
for i=0:10, for j=0:10
```
\begin{verbatim}
x = v([Xp(i+1,j+1);Yp(i+1,j+1)]);
Up(i+1,j+1) = x(1); Vp(i+1,j+1) = x(2);
end; end

figure ('name','velocity field','renderer','painters');
quiver(Xp,Yp,Up,Vp,'b-'); set(gca,'fontsize',14); hold on;
plot([0 1 1 0 0],[0 0 1 1 0],'k-');
axis([-0.1 1.1 -0.1 1.1]);
xlabel('\textbf{x}_1'); ylabel('\textbf{x}_2');
axis off;

fp = figure ('name','particles','renderer','painters');
fs = figure ('name','solution','renderer','painter');

% Visualize points (interior points in red, boundary points in blue)
figure(fp); plot(P(1,:),P(2,:), 'r+', BP(1,:), BP(2,:), 'b*');
title(sprintf('n = %i, t = %f, \tau = %f, %i points', n, 0, tau, size(P,2)));
drawnow; pause;

% Visualize solution
figure(fs); plotpartsol(P,U); drawnow;
\end{verbatim}
for l=1:m  
% Advect points (explicit trapezoidal rule)
P1 = P + tau/2*v(P); P = P + tau*v(P1);

% Remove points on the boundary or outside the domain
Pnew = []; Unew = []; l = 1;
for p=P
    if ((p(1) > eps) (p(1) < 1-eps) (p(2) > eps) (p(2) < 1-eps))
        Pnew = [Pnew,p]; Unew = [Unew; U(l)];
    end
    l = l+1;
end

% Add points on the boundary (particle injection)
P = [Pnew, BP]; U = [Unew; g(BP)];

% Visualize points
figure(fp); plot(P(1,:),P(2,:), 'r+', BP(1,:),BP(2,:), 'b*');
title(sprintf('n = %i, t = %f, \tau = %f, %i points', n, t, tau, size(P,2)));
drawnow;

% Visualize solution
```
figure(fs); plotpartsol(P,U); drawnow;

   t = t+tau;
end

partadv(@circvel,@initvals,@bdvals,40,0.025,80)
partadv(@rotvel,@initvals,@bdvals,40,0.025,80)
```

### 7.3.3.3 Particle mesh method
The method introduced in the previous section, can be used to tackle the pure advection problem (7.3.19) in the 2nd sub-step of the Strang splitting timestepping.

Issue: How to combine Lagrangian advection with a method for the pure diffusion problem (7.3.18) faced in the other sub-steps of the Strang splitting timestepping?

Idea: two views

“particle temperatures” \( u(p_i^{(j)}) \)

\[ \uparrow \]

Nodal values of finite element function \( u_N^{(j)} \in S_1^0(M) \)

Outline: algorithm for one step of size \( \tau > 0 \) of Strang splitting timestepping for transient convection-diffusion problem

\[
\begin{aligned}
\frac{\partial u}{\partial t} - \epsilon \Delta u + \mathbf{v}(x, t) \cdot \text{grad} u &= f \quad \text{in } \tilde{\Omega} := \Omega \times ]0, T[ , \\
\mathbf{u}(x, t) &= 0 \quad \forall x \in \partial\Omega, \; 0 < t < T , \quad u(x, 0) = u_0(x) \quad \forall x \in \Omega .
\end{aligned}
\] (7.3.2)
Given triangular mesh $\mathcal{M}^{(j-1)}$ of $\Omega$,
- $u_N^{(j-1)} \in S_{1,0}^0(\mathcal{M}^{(j-1)}) \leftrightarrow$ coefficient vector $\vec{\mu}^{(j-1)} \in \mathbb{R}^{N_j-1}$,
approach to solve (7.3.18) by a single step of implicit Euler (6.1.23) (size $\frac{1}{2}\tau$)

$$\vec{\nu} = (M + \frac{1}{2}\tau\epsilon A)^{-1} \vec{\mu}^{(j-1)},$$

where $A \in \mathbb{R}^{N_j-1 \times N_j-1} \triangleq S_{1,0}^0(\mathcal{M})$-Galerkin matrix for $-\Delta$, $M \triangleq$ (possibly lumped) $S_{1,0}^0(\mathcal{M})$-mass matrix.

More advisable to maintain 2nd-order timestepping: 2nd-order $L(\pi)$-stable single step method, e.g., SDIRK-2 (6.1.59).

2 Lagrangian advection step (of size $\tau$) for (7.3.19) with
- initial “particle positions” $p_i$ given by nodes of $\mathcal{M}^{(j-1)}$,
- initial “particle temperatures” given by corresponding coefficients $\nu_i$.

3 Remeshing: advection step has moved nodes to new positions $\tilde{p}_i$ (and, maybe, introduced new nodes by “particle injection”, deleted nodes by “particle removal”).

Create new triangular mesh $\mathcal{M}^{(j)}$ with nodes $\tilde{p}_i$ (+ boundary nodes), $i = 1, \ldots, N_j$
Repeat diffusion step ① starting with \( w_N \in S_{1,0}^0(\mathcal{M}(j)) \) = linear interpolant (→ Def. 5.3.13) of “particle temperatures” on \( \mathcal{M}(j) \).

\[ u_{N}^{(j)} \]

Example 7.3.28 (Delaunay-remeshing in 2D).

Delaunay algorithm for creating a 2D triangular mesh with prescribed nodes:

① Compute Voronoi cells, see (4.2.3) & http://www.qhull.org/.

② Connect two nodes, if their associated Voronoi dual cells have an edge in common.

MATLAB

\[
\text{TRI} = \text{delaunay}(x,y)
\]
Code 7.3.29: Demonstration of Delaunay-remeshing

```matlab
function meshadv(v,n,tau,m)

% Point advaction and remeshing for Lagrangian method
% v: handle to a function returning the velocity field for (an array) of points
% n: h=1/n is the grid spacing of the initial point distribution

% Initialize points
h = 1/n; [Xp,Yp] = meshgrid(0:h:1,0:h:1);
P = [reshape(Xp,1,(n+1)^2); reshape(Yp,1,(n+1)^2)];

% Initialize points on the boundary
BP = [(0:h:1); zeros(1,n+1)], [ones(1,n+1);(0:h:1)],...
     [(0:h:1);ones(1,n+1)], [zeros(1,n+1);(0:h:1)];

% Plot triangulation
fp = figure('name','evolving meshes','renderer','painters');
TRI = delaunay(P(1,:),P(2,:));
plot(P(1,:),P(2,:),'r+'); hold on;
   triplot(TRI,P(1,:),P(2,:),'blue'); hold off;
title(sprintf('n = %i, t = %f, \tau = %f, %i points',n,0,tau,size(P,2)));
drawnow; pause;

for l=1:m
```

R. Hiptmair
C. Schwab,
H. Harbrecht
V. Gradinaru
A. Chernov
SAM, ETHZ
% Advect points (explicit trapezoidal rule)
P1 = P + tau/2*v(P); P = P + tau*v(P1);

% Remove points on the boundary or outside the domain
Pnew = []; l = 1;
for p=P
    if ((p(1) > eps) && (p(1) < 1-eps) && (p(2) > eps) && (p(2) < 1-eps))
        Pnew = [Pnew, p];
    end
    l = l+1;
end
P = [Pnew, BP]; % Add points on the boundary (particle injection)

% Plot triangulation
TRI = delaunay(P(1,:),P(2,:));
plot(P(1,:),P(2,:),’r+’); hold on;
    triplot(TRI,P(1,:),P(2,:),’blue’); hold off;
title(sprintf(’n = %i, t = %f, \tau = %f, %i points’,n,t,tau,size(P,2)));
drawnow;

t = t+tau;
\( \Omega = [0, 1]^2 \), velocity fields like in Ex. 7.3.24. Advection of interpolation nodes by means of explicit trapezoidal rule.

Start animations:

```
meshadv(@circvel, 20, 0.05, 40);
meshadv(@rotvel, 20, 0.05, 40);
```

Example 7.3.30 (Lagrangian method for convection-diffusion in 1D).
Same IBVP as in Ex. 7.3.4

- Linear finite element Galerkin discretization with mass lumping in space
- Strang splitting applied to diffusive and convective terms
- Implicit Euler timestepping for diffusive partial timestep

Code 7.3.31: Lagrangian method for (7.3.5)

```matlab
function lagr(epsilon,N,M)
% This function implements a simple Lagrangian advection scheme for the 1D convection-diffusion
% IBVP \(-\epsilon \frac{d^2u}{dx^2} + \frac{du}{dx} = 0\), \(u(x,0) = \max(1 - 3|x - \frac{1}{3}|,0)\),
% and homogeneous Dirichlet boundary conditions \(u(0) = u(1) = 0\). Timestepping employs Strang splitting
% applied to diffusive and convective spatial operators.
% epsilon: strength of diffusion
% N: number of cells of spatial mesh
% M: number of timesteps
```
% timestep size
h = 1/N; x = 0:h:1; u = max(1-3*abs(x(2:end-1)-1/3),0)’; % Initial value

[Amat,Mmat] = getdeltamat(x); % Obtain stiffness and mass matrix
u = (Mmat+0.5*tau*epsilon*Amat)\(Mmat*u); % Implicit Euler timestep

for j=1:M+1
  % Advection step: shift meshpoints, drop those travelling out of Ω = [0,1], insert new meshpoints from the left. Solution values are just copied.
  xm = x(2:end-1)+tau; % Transport of meshpoints (here: explicit Euler)
  idx = find(xm < 1); % Drop meshpoints beyond x = 1
  x = [0,tau,xm(idx),1]; % Insert new meshpoint at left end of Ω
  u = [0;u(idx)]; % Copy nodal values and feed 0 from left

  % Diffusion partial timestep
  [Amat,Mmat] = getdeltamat(x); % Obtain stiffness and mass matrix on new mesh
  u = (Mmat+tau*epsilon*Amat)\(Mmat*u); % Implicit Euler step
end
\[ \epsilon = 10^{-5}; \]

\[ \epsilon = 0.1; \]

“Reference solution” computed by method of lines, see Ex. [7.3.4], with \( h = 10^{-3}, \tau = 5 \cdot 10^{-5}; \)
Example 7.3.32 (Lagrangian method for convection-diffusion in 2D).

- IBVP (7.3.2) on $\Omega = [0, 1]^2$, $T = 1$,
- Particle mesh method based on Delaunay remeshing, see Ex. 7.3.28, and linear finite element Galerkin discretization for diffusion step.

**Code 7.3.33: Particle mesh method in 2D**

```matlab
function ConvDiffLagr(v, epsilon, u0, n, tau, m)
% Point particle method for convection-diffusion problem on the unit square
% v: handle to a function returning the velocity field for (an array) of points
```
% u0: handle to a function returning the initial value \( u_0 \) for (an array) of points
\[
% \text{n: } h = 1/n \text{ is the grid spacing of the initial point distribution}
\%
% \tau: \text{ timestep size, } m: \text{ number of timesteps, that is, } T = m\tau
\%

% Initialize points
\[ h = 1/n; \ [Xp,Yp] = \text{meshgrid}(0:h:1,0:h:1); \]
\[ P = [\text{reshape}(Xp,1,(n+1)^2); \text{reshape}(Yp,1,(n+1)^2)]; \]
% Initialize points on the boundary
\[ BP = [[[0:h:1]; \text{zeros}(1,n+1)], [\text{ones}(1,n+1);(0:h:1)], ... \]
\[ ((0:h:1); \text{ones}(1,n+1)], [\text{zeros}(1,n+1);(0:h:1)]]; \]
% Construct initial mesh by Delaunay algorithm
\[ TRI = \text{delaunay}(P(1,:),P(2,:)); \]

U = u0(P); % Initial values

fp = figure('name','particles','renderer','painters');
fs = figure('name','solution','renderer','painters');

% Visualize mesh, points (interior points in red, boundary points in blue)
% the piecewise linear approximate solution
\text{figure}(fp); \text{plot}(P(1,:),P(2,:),'r+',BP(1,:),BP(2,:),'m*'); \text{hold on};
\text{triplot}(TRI,P(1,:),P(2,:),'blue'); \text{hold off};
\text{title}(\text{sprintf('n = %i, t = %f, \tau = %f, %i points',n,0,tau,size(P,2))});
drawnow;

\text{figure}(fs); \text{trisurf}(TRI,P(1,:),P(2,:),U');
\text{axis}([0 1 0 1 0 1]); \text{xlabel}('{\bf x_1}');
\text{ylabel}('{\bf x_2}'); \text{zlabel}('{\bf u}');
\text{title}(\text{sprintf('n = %i, t = %f, \tau = %f, %i points',n,0,tau,size(P,2))});
pause;

% Initial diffusion half step (implicit Euler)
[Amat, Mmat] = getGalerkinMatrices(TRI, P(1,:), P(2,:)); % Compute Galerkin matrices

% Isolate indices of interior points
j = 1; intidx = [];
for p=P
    if ((p(1) > eps) (p(1) < 1-eps) (p(2) > eps) (p(2) < 1-eps))
        intidx = [intidx, j];
    end
    j = j+1;
end
Amat = Amat(intidx, intidx); Mmat = Mmat(intidx, intidx);
U(intidx) = (Mmat + 0.5 * epsilon * tau * Amat) \ (Mmat * U(intidx));

% full(Amat), full(Mmat), return;

t = 0;
for l=1:m
    % Advect points (explicit trapezoidal rule)
    P1 = P + tau/2 * v(P); P = P + tau * v(P1);

    % Remove points on the boundary or outside the domain
    Pnew = []; Unew = []; l = 1; j = 0;
    for p=P
        if ((p(1) > eps) (p(1) < 1-eps) (p(2) > eps) (p(2) < 1-eps))
            Pnew = [Pnew, p]; Unew = [Unew; U(l)];
            j = j+1; % Counter for interior points
        end
        l = l+1;
    end
% Add points on the boundary (particle injection)
P = [Pnew, BP];

% Delaunay algorithm for building triangulation
TRI = delaunay(P(1,:),P(2,:));
[Amat,Mmat] = getGalerkinMatrices(TRI,P(1,:),P(2,:)); % Compute Galerkin matrices
Amat = Amat(1:j,1:j); Mmat = Mmat(1:j,1:j);
U = (Mmat+epsilon*tau*Amat)\(Mmat * Unew); % implicit Euler step
U = [U; zeros(size(BP,2),1)]; % zero padding for boundary nodes

% Visualize mesh, points (interior points in red, boundary points in blue)
figure(fp); plot(P(1,:),P(2,:),’r+’,BP(1,:),BP(2,:),’m*’); hold on;
triplot(TRI,P(1,:),P(2,:),’blue’); hold off;
title(sprintf(’n = %i, t = %f, \tau = %f, %i points’,n,t,tau,size(P,2)));
drawnow;

figure(fs); trisurf(TRI,P(1,:),P(2,:),U’);
axis([0 1 0 1 0 1]); xlabel(’{\bf x_1}’);
ylabel(’{\bf x_2}’); zlabel(’{\bf u}’);
title(sprintf(’n = %i, t = %f, \tau = %f, %i points’,n,t,tau,size(P,2)));
t = t+tau;
Advantage of Lagrangian (particle) methods for convection diffusion:

No artificial diffusion required (no “smearing”)
No stability induced timestep constraint

Drawback of Lagrangian (particle) methods for convection diffusion:

Remeshing (may be) expensive and difficult.
Point advection may produce “voids” in point set.

7.3.4 Semi-Lagrangian method
Now we study a family of methods for transient convection-diffusion that takes into account transport along streamlines, but, in contrast to genuine Lagrangian methods, relies on a fixed mesh.

**Definition 7.3.34** (Material derivative).

Given a velocity field \( v : \Omega \times [0, T] \rightarrow \mathbb{R}^d \), the material derivative of a function \( f = f(x, t) \) at \((x, t)\) is

\[
\frac{Df}{Dv}(x, t_0) = \lim_{\tau \to 0} \frac{f(x, t_0) - f(\Phi_{t_0}^{-\tau} x, t_0 - \tau)}{\tau}, \quad x \in \Omega, \ 0 < t_0 < T,
\]

with \( \Phi_{t_0}^t \) the flow map (at time \( t_0 \)) associated with \( v \), that is, cf. (7.1.3), (7.1.4),

\[
\frac{d\Phi_{t_0}^t x}{dt} = v(\Phi_{t_0}^t x, t - t_0) \quad , \quad \Phi^0 x = x.
\]

The material derivative \( \frac{Df}{Dv} \) is the rate of change of \( f \) experienced by a particle carried along by the flow.
because $\Phi^t_{t_0}x$ describes the trajectory of a particle located at $x$ at time $t_0$ ($\leftrightarrow t = 0$).

By a straightforward application of the chain rule for smooth $f$

$$\frac{Df}{Dv}(x, t) = \nabla_x f(x, t) \cdot v(x, t) + \frac{\partial f}{\partial t}(x, t). \quad (7.3.37)$$

The transient convection-diffusion equation can be rewritten as (7.3.1)

$$\begin{align*}
\frac{\partial u}{\partial t} - \epsilon \Delta u + v(x, t) \cdot \nabla u &= f \quad \text{in} \quad \tilde{\Omega} := \Omega \times ]0, T[, \\
D\frac{u}{Dv} - \epsilon \Delta u &= f \quad \text{in} \quad \tilde{\Omega} := \Omega \times ]0, T[. \quad (7.3.38)
\end{align*}$$
Idea: **Backward difference** ("implicit Euler") discretization of material derivative

\[
\begin{align*}
\frac{Du}{Dv}(x,t=(\bar{x},t_0)) & \approx \frac{u(\bar{x},t_0) - u(\Phi_{t_0}^{-\tau}\bar{x},t_0 - \tau)}{\tau}, \\
\text{with timestep } \tau > 0, \text{ where } t \mapsto \Phi^t(\bar{x}) \text{ solves the initial value problem}
\end{align*}
\]

\[
\begin{align*}
\frac{d\Phi^t_{t_0}(\bar{x})}{dt}(t) &= v(\Phi^t_{t_0}(\bar{x}),t_0 + t), \quad \Phi^0_{t_0}(\bar{x}) = \bar{x}.
\end{align*}
\]

**Semi-discretization of** (7.3.38)** in time** (with fixed timestep \(\tau > 0\))

\[
\begin{align*}
\frac{u^{(j)}(x) - u^{(j-1)}(\Phi_{t_j}^{-\tau}\bar{x})}{\tau} = \epsilon \Delta u^{(j)}(x) &= f(x,t_j) \quad \text{in } \Omega, \\
+ \quad \text{boundary conditions at } t = t_j,
\end{align*}
\]

(7.3.40)

where \(u^{(j)} : \Omega \mapsto \mathbb{R}\) is an approximation for \(u(\cdot,t_j), t_j := j\tau, j \in \mathbb{N}\).

Note the difference to the method of lines (→ Sects. 6.1.3, 6.2.3, 7.3.1): in (7.3.40) semidiscretization in time was carried out first, now followed by discretization in space, which reverses the order adopted in the method of lines.
Cast (7.3.40) into variational form according to the recipe of Sect. 2.8 and apply \textit{Galerkin discretization} (here discussed for linear finite elements, homogeneous Dirichlet boundary conditions \( u = 0 \) on \( \partial \Omega \)).

This yields one timestep (size \( \tau \)) for the \textit{semi-Lagrangian method}: the approximation \( u_N^{(j)} \) for \( u(j \tau) \) (equidistant timesteps) is computed from the previous timestep according to

\[
\begin{align*}
  u_N^{(j)} & \in S_{1,0}^0(\mathcal{M}): \\
  \int_{\Omega} \frac{u_N^{(j)}(x) - u_N^{(j-1)}(\Phi_{t_j}^{-\tau} x)}{\tau} v_N(x) \, dx + \epsilon \int_{\Omega} \nabla u_N^{(j)} \cdot \nabla v_N \, dx \\
  = \int_{\Omega} f(x, t_j) v_N(x) \, dx \quad \forall v_N \in S_{1,0}^0(\mathcal{M}). \quad (7.3.41)
\end{align*}
\]

Here, \( \mathcal{M} \) is supposed to be a \textit{fixed} triangular mesh of \( \Omega \).

However, (7.3.41) cannot be implemented: \( x \mapsto u_N^{(j-1)}(\Phi_{t_j}^{-\tau} x) \) is a finite element function that has been “transported with the (reversed) flow” (in the sense of pullback, see Def. 3.6.2).
The pullback $x \mapsto v_N(\Phi_{t_j}^{-T}x)$ of $v_N \in S_{1,0}^0(\mathcal{M})$ is piecewise smooth w.r.t. the mapped mesh drawn with \ldots. Hence, it is not smooth inside the cells of $\mathcal{M}$.

- the transported function may **not** be a finite element function on $\mathcal{M}$,
- the transported function may not even be piecewise smooth on $\mathcal{M}$.
local quadrature for the approximate computation of the integral in (7.3.41) that involves $u_N^{(j-1)}(\Phi_{t_j}^{-\tau} x)$ is not possible, because accurate numerical quadrature requires a (locally) smooth integrand.

Idea: replace $u_N^{(j-1)}(\Phi_{t_j}^{-\tau} x)$ with linear interpolant ($\rightarrow$ Def. 5.3.13)

$I_1 (u_N^{(j-1)} \circ \Phi^{-\tau}_{t_j}) \in S^0_{1,0}(\mathcal{M}),$

approximate $\Phi_{t_j}^{-\tau} x$ by $x - \tau v(x, t_j)$ (explicit Euler).

(“streamline backtracking”)

Implementable version of (7.3.41) (using mass lumping, see Rem. 6.2.34)

$u_N^{(j)} \in S^0_{1,0}(\mathcal{M}): \frac{1}{3} |U_p| (\mu^{(j)}_p - u_N^{(j-1)}(p - \tau v(p, t_j))) + \tau \int_{\Omega} \nabla u_N^{(j)} \cdot \nabla b^p_N \, dx$

$= \frac{1}{3} |U_p| f(p), \quad p \in \mathcal{N}(\mathcal{M}) \cap \Omega, \quad (7.3.42)$
where $\mu_p^{(j)}$ are the nodal values of $u_N^{(j)} \in S^0_{1,0}(\mathcal{M})$ associated with the interior nodes of the mesh $\mathcal{M}$, $b_N^p$ is the “tent function” belonging to node $p$, $|U_p|$ is the sum of the areas of all triangles adjacent to $p$.

**Example 7.3.43** (Semi-Lagrangian method for convection-diffusion in 1D).

Same IBVP as in Ex. **7.3.30**

- Linear finite element Galerkin discretization with mass lumping in space
- Semi-Lagrangian method: 1D version of (7.3.41)
- Explicit Euler streamline backtracking
function semilagr(epsilon,N,M,filename)
% Semi-Lagrangian linear finite element method for transient
% convection-diffusion problem in 1D
if (nargin < 4), filename = 'semilagr'; end

T = 0.5; tau = T/M; tstop = T/4; % timestep size
h = 1/N; x = 0:h:1; u = max(1-3*abs(x(2:end-1)-1/3),0)';
% Initial value \( u(x,0) \)

[Amat,Mmat] = getdeltamat(x);

figure; plot(x,[0;u;0],'r+'); axis([0 1 0 1]);
xlabel('{\bf x}','fontsize',14);
ylabel('{\bf u}','fontsize',14);
title('t=0');
print('-depsc2', sprintf('.../.../.../.../rw/Slides/NPDEPics/%s%i.eps',filename,0));

for j=1:M+1
    xs = x(2:end-1) - tau; % Shifted gridpoints
    uv = [0;u;0]; % Nodal values padded with zero
    % Find intervals to which shifted points belong and perform linear interpolation
xsp = xs/h; xsp(find(xsp < 0)) = 0;
xw = (xsp - floor(xsp))';
uxs = (1-xw).*uv(floor(xsp)+1) + xw.*uv(ceil(xsp)+1);

%Timestep for material derivative
u = (Mmat+tau*epsilon*Amat)
   \(\) (Mmat*uxs);

if (j*tau > tstop)
    figure;
    plot(x,[0;u;0],''r''+'); axis([0 1 0 1]);
xlabel(''{\bf x}'',''fontsize'',14);
ylabel(''{\bf u}'',''fontsize'',14);
title(sprintf(''Semi-Lagrangian method: M = %f, tau = %f,
t=%f'',length(x)-1,tau,j*tau));
tstop = tstop + T/4;
print('-depsc2', sprintf(''.../.../.../.../rw/Slides/NPDEPics/%s%i.ep''
    filename,j));
end

\(\epsilon = 10^{-5}:\)
\(\epsilon = 0.1:\)

“Reference solution” computed by method of lines, see Ex. 7.3.4, with \(h = 10^{-3}, \tau = 5 \cdot 10^{-5}:\)
Example 7.3.45 (Semi-Lagrangian method for convection-diffusion in 2D).

- 2nd-order scalar convection diffusion problem (7.3.2), \( \Omega := ]0, 1[^2, f = 0, g = 0,
- velocity field

\[ \mathbf{v}(\mathbf{x}) := \begin{pmatrix} -\sin(\pi x_1) \cos(\pi x_2) \\ \sin(\pi x_2) \cos(\pi x_1) \end{pmatrix}. \]
- Initial condition: “compactly supported cone shape”

\[ u_0(x) = \max(0, 1-4*\sqrt{((x(:,1)-0.5)^2+(x(:,2)-0.25)^2)}; \]
- semi-Lagrangian finite element Galerkin discretization according to (7.3.41) on regular triangular meshes of square domain \( \Omega \), see Fig. [123].
Example with $\epsilon = 0$:

We observe smearing of initial data due to numerical diffusion inherent in the interpolation step of the semi-Lagrangian method.
Conservation laws describe physical phenomena governed by

- *conservation* laws for certain physical quantities (e.g., mass, momentum, energy, etc.),
- *transport* of conserved physical quantities.

We have already examined problems of this type in connection with transient heat conduction in Sect. 7.1.4. There thermal energy was the conserved quantity and a *prescribed* external velocity field $v$ determined the transport.

A new aspect emerging for general conservation laws is that the transport velocity itself may depend on the conserved quantities themselves, which gives rise to *non-linear models*.
Supplementary and further reading:

[22]: Comprehensive monograph and textbook about so-called finite volume method providing detailed explanations.

[20]: concisely written textbook adopting a mathematical perspective and delving into technical details.

[9]: mathematical monograph about the theory of initial value problems for conservation laws.

8.1 Conservation laws: Examples
Focus: Cauchy problems

Spatial domain $\Omega = \mathbb{R}^d$ (unbounded!)

Cauchy problems are pure initial value problems (no boundary values).

Rationale: ① *Finite speed of propagation* typical of conservation laws

(Potential spatial boundaries will not affect the solution for some time in the case of compactly supported initial data, *cf.* situation for wave equation, where we also examined the Cauchy problem, see (6.2.15).)

② No spatial boundary $\Rightarrow$ need not worry about (spatial) boundary conditions!

(Issue of spatial boundary conditions can be very intricate for conservation laws)
8.1.1 Linear advection

Cauchy problem for linear transport equation (advection equation) → Sect. [7.1.4, (7.1.16)]:

\[ \frac{\partial}{\partial t}(\rho u) + \text{div}(v(x, t)(\rho u)) = f(x, t) \quad \text{in} \quad \Omega := \mathbb{R}^d \times ]0, T[, \quad (8.1.1) \]

\[ u(x, 0) = u_0(x) \quad \text{for all} \quad x \in \mathbb{R}^d \quad \text{(initial conditions).} \quad (8.1.2) \]

\( u = u(x, t) \triangleq \text{temperature}, \rho > 0 \triangleq \text{heat capacity}, v = v(x, t) \triangleq \text{prescribed velocity field.} \)

(8.1.1) = linear scalar conservation law

Conserved quantity: thermal energy (density) \( \rho u \)

(Recall the derivation of (7.1.16) through conservation of energy, cf. (6.1.1).)
Simplified problem: assume constant heat capacity $\rho \equiv 1$, no sources $f \equiv 0$, stationary velocity field $\mathbf{v} = \mathbf{v}(\mathbf{x})$ ➞ rescaled initial value problem written in conserved variables

$$\frac{\partial u}{\partial t} + \text{div}(\mathbf{v}(\mathbf{x})u) = 0 \quad \text{in} \quad \tilde{\Omega} := \mathbb{R}^d \times [0, T],$$

$$u(\mathbf{x}, 0) = u_0(\mathbf{x}) \quad \text{for all} \quad \mathbf{x} \in \mathbb{R}^d \quad \text{(initial conditions)}.$$ 

Convention: differential operator div acts on spatial independent variable only,

$$(\text{div } \mathbf{f})(\mathbf{x}, t) := \frac{\partial f_1}{\partial x_1} + \cdots + \frac{\partial f_d}{\partial x_d}, \quad \mathbf{f}(\mathbf{x}, t) = \begin{pmatrix} f_1(\mathbf{x}, t) \\ \vdots \\ f_d(\mathbf{x}, t) \end{pmatrix}.$$ 

Special case: Constant coefficient linear advection in 1D

- $d = 1$ ➞ $\Omega = \mathbb{R}$,
- constant velocity $\mathbf{v} = \text{const.}$.

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}(vu) = 0 \quad \text{in} \quad \tilde{\Omega} = \mathbb{R} \times [0, T], \quad u(x, 0) = u_0(x) \quad \forall x \in \mathbb{R}.$$ (8.1.5)
This is the 1D version of the transport equation (7.3.7) ➤ solution given by (7.3.11)

\[ u(x, t) = u_0(x - vt), \quad x \in \mathbb{R}, \quad 0 \leq t < T. \]  

(8.1.6)

Solution \( u = u(x, t) \) = initial data “travelling” with velocity \( v \).

Solution formula (8.1.6) makes perfect sense even for **discontinuous** initial data \( u_0 \)!

We should not expect \( u = u(x, t) \) to be differentiable in space or time.

A “weaker” concept of solution is required, see Sect. 8.2.3 below.

This consideration should be familiar: for second order elliptic boundary value problems, for which classical solutions are to be twice continuously differentiable, the concept of a variational solution
made it possible to give a meaning to solutions $\in H^1(\Omega)$ that are merely continuous and piecewise differentiable, see Rem. [1.3.25].

Related to (8.1.6): d’Alembert solution formula (6.2.16) for 1D wave equation (6.2.15).

**Remark 8.1.11 (Boundary conditions for linear advection).**

Recall the discussion in Sects. 7.2.1, 7.3.2, cf. solution formula (7.3.12):

For the scalar linear advection initial boundary value problem

\[
\frac{\partial u}{\partial t} + \text{div}(\mathbf{v}(x, t)u) = f(x, t) \quad \text{in} \quad \tilde{\Omega} := \Omega \times [0, T],
\]

\[
u(x, 0) = u_0(x) \quad \text{for all} \quad x \in \Omega,
\]

on a bounded domain $\Omega \subset \mathbb{R}^d$, **boundary conditions** (e.g., prescribed temperature)

\[
u(x, t) = g(x, t) \quad \text{on} \quad \Gamma_{in}(t) \times [0, T],
\]
can be imposed on the **inflow boundary**

\[ \Gamma_{\text{in}}(t) := \{ x \in \partial \Omega : \mathbf{v}(x, t) \cdot \mathbf{n}(x) < 0 \} , \quad 0 < t < T . \] (8.1.14)

Note: \( \Gamma_{\text{in}} \) can change with time!

Bottom line:

Knowledge of local and current direction of transport needed to impose meaningful boundary conditions!
8.1.2 Inviscid gas flow

Frictionless gas flow in (infinitely) long pipe

Terminology: frictionless $\equiv$ inviscid

Assumption: variation of gas density negligible ("near incompressibility")

motion of fluid driven by inertia $\leftrightarrow$ conservation of linear momentum

We derive a continuum model for inviscid, nearly incompressible fluid in a straight infinitely long pipe $\leftrightarrow \Omega = \mathbb{R}$ (Cauchy problem).

This simple model will be based on conservation of linear momentum, whereas conservation of mass and energy will be neglected (and violated). Hence, the crucial conserved quantity will be the momentum.

Unknown: $u = u(x,t) = \text{momentum density} \sim \text{local velocity} v = v(x,t)$ of fluid
Conserved quantity: (linear) momentum of fluid $u = u(x, t)$

\[ f \sim v \cdot u \] (after scaling: $f(u) = \frac{1}{2} u \cdot u$)  

(“momentum $u$ advected by velocity $u$”)

Conservation of linear momentum ($\sim u$): for all control volumes $V := ]x_0, x_1[ \subset \Omega$:

\[
\int_{x_0}^{x_1} u(x, t_1) - u(x, t_0) \, dx + \int_{t_0}^{t_1} \frac{1}{2} u^2(x_1, t) - \frac{1}{2} u^2(x_0, t) \, dt = 0 \quad \forall 0 < t_0 < t_1 < T .
\]

(8.1.15)

Temporarily assume that $u = u(x, t)$ is smooth in both $x$ and $t$ and set $x_1 = x_0 + h$, $t_1 = t_0 + \tau$.

First approximate the integrals in (8.1.15).

\[
\int_{x_0}^{x_1} u(x, t_1) - u(x, t_0) \, dx = h(u(x_0, t_1) - u(x_0, t_0)) + O(h^2) \quad \text{for} \quad h \to 0 ,
\]
\[
\int_{t_0}^{t_1} \left( \frac{1}{2} u_1^2(x_1, t) - \frac{1}{2} u_0^2(x_0, t) \right) \, dt = \tau \left( \frac{1}{2} u_0^2(x_1, t_0) - \frac{1}{2} u_0^2(x_0, t_0) \right) + O(\tau^2) \quad \text{for} \quad \tau \to 0.
\]

Then employ Taylor expansion for the differences:

\[u(x_0, t_1) - u(x_0, t_0) = \frac{\partial u}{\partial t}(x_0, t_0) \tau + O(\tau^2) \quad \text{for} \quad \tau \to 0,\]

\[\frac{1}{2} u_1^2(x_1, t_0) - \frac{1}{2} u_0^2(x_0, t_0) = \frac{\partial}{\partial x} \left( \frac{1}{2} u_0^2 \right)(x_0, t_0) h + O(h^2) \quad \text{for} \quad h \to 0.\]

Finally, divide by \( h \) and \( \tau \) and take the limit \( \tau \to 0, h \to 0 \):

\[
\frac{\partial u}{\partial t} + \frac{\partial}{\partial x} \left( \frac{1}{2} u^2 \right) = 0 \quad \text{in} \quad \Omega \times ]0, T[. \tag{8.1.16}
\]

\textbf{(8.1.16) = Burgers equation:} a one-dimensional scalar conservation law (without sources)

\textbf{Remark 8.1.17 (Euler equations).}
The above gas model blatantly ignores the fundamental laws of conservation of mass and of energy. These are taken into account in a famous more elaborate model of inviscid fluid flow:

**Euler equations** [7], a more refined model for inviscid gas flow in an infinite pipe

\[
\frac{\partial}{\partial t} \begin{pmatrix} \rho \\ \rho u \\ E \end{pmatrix} + \frac{\partial}{\partial x} \begin{pmatrix} \rho u \\ \rho u^2 + p \\ (E + p)u \end{pmatrix} = 0 \quad \text{in} \quad \mathbb{R} \times [0, T],
\]

\[u(x, 0) = u_0(x) \quad , \quad \rho(x, 0) = \rho_0(x) \quad , \quad E(x, 0) = E_0(x) \quad \text{for} \quad x \in \mathbb{R},\]

where
- \( \rho = \rho(x, t) \) ≡ fluid density, \([\rho] = \text{kg m}^{-1}\),
- \( u = u(x, t) \) ≡ fluid velocity, \([u] = \text{m s}^{-1}\),
- \( p = p(x, t) \) ≡ fluid pressure, \([p] = \text{N}\),
- \( E = E(x, t) \) ≡ total energy density, \([E] = \text{J m}^{-1}\).

+ *state equation* (material specific constitutive equations), e.g., for ideal gas

\[p = (\gamma - 1)(E - \frac{1}{2}\rho u^2), \quad \text{with adiabatic index} \quad 0 < \gamma < 1.\]

Conserved quantities (densities):

\( \rho \leftrightarrow \text{mass density} \quad , \quad \rho u \leftrightarrow \text{momentum density} \quad , \quad E \leftrightarrow \text{energy density}. \)
Underlying physical conservation principles for individual densities:

- **First equation** \( \frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x}(\rho u) = 0 \) \( \leftrightarrow \) **conservation of mass**, 

- **Second equation** \( \frac{\partial (\rho u)}{\partial t} + \frac{\partial}{\partial x}(\rho u^2 + p) = 0 \) \( \leftrightarrow \) **conservation of momentum**, 

- **Third equation** \( \frac{\partial E}{\partial t} + \frac{\partial}{\partial x}((E + p)u) = 0 \) \( \leftrightarrow \) **conservation of energy**.

Euler equations (8.1.18) = non-linear system of conservation laws (in 1D)

As is typical of non-linear systems of conservation laws, the analysis of the Euler equations is intrinsically difficult: hitherto not even existence and uniqueness of solutions for general initial values could be established. Moreover, solutions display a wealth of complicated structures. Therefore, this course is confined to scalar conservation laws, for which there is only one unknown real-valued function of space and time.
8.2 Scalar conservation laws in 1D

8.2.1 Integral and differential form

What we have seen so far (except for Euler’s equations in Rem. 8.1.17)

\[ \frac{\partial u}{\partial t} + \frac{\partial}{\partial x} \left( \frac{1}{2} u^2 \right) = 0 \quad \text{in } \Omega \times ]0, T[ , \quad (8.1.16) \]

Burgers equation:

linear advection:

\[ \frac{\partial}{\partial t} (\rho u) + \text{div}(\mathbf{v}(x, t)(\rho u)) = f(x, t) \quad \text{in } \mathbb{R}^d \times ]0, T[ . \quad (8.1.1) \]

Now, we learn about a class of Cauchy problems to which these two belong. First some notations and terminology:

- \( \Omega \subset \mathbb{R}^d \) \( \hat{\Omega} \) fixed (bounded/unbounded) spatial domain \( (\Omega = \mathbb{R}^d = \text{Cauchy problem}) \)

- Computational domain: \( \hat{\Omega} := \Omega \times ]0, T[ , T > 0 \) final time

- \( U \subset \mathbb{R}^m (m \in \mathbb{N}) \) \( \hat{U} \) phase space (state space) for conserved quantities \( u_i \) (usually \( U = \mathbb{R}^m \))
Our focus below: \( scalar \ case \quad m = 1 \)

Conservation law for transient state distribution

\[ u : \tilde{\Omega} \mapsto U : u = u(x, t), \text{ for } 0 \leq t \leq T \]

\[
\frac{d}{dt} \int_V u \, dx + \int_{\partial V} f(u, x) \cdot n \, dS(x) = \int_V s(u, x, t) \, dx \quad \forall \text{ "control volumes" } V \subset \Omega. \tag{8.2.1}
\]

- **Terminology:**
  - **flux function** \( f : U \times \Omega \mapsto \mathbb{R}^d \)
  - **source function** \( s : U \times \Omega \times ]0, T[ \mapsto \mathbb{R} \) (here usually \( s = 0 \))

- For Burgers equation (8.1.16):
  \( f(u, x) = \frac{1}{2}u^2, \quad s = 0, \)

- For linear advection (8.1.1):
  \( f(u, x) = v(x, t)u, \quad s = f(x, t) \)

(Note: in this case the conserved quantity is actually \( \rho_u \), which was again denoted by \( u \))
(8.2.1) has the same structure as the "conservation of energy law" (6.1.1) for heat conduction.

Conservation of energy:

\[
\frac{d}{dt} \int_V \rho u \, d\mathbf{x} + \int_{\partial V} \mathbf{j} \cdot \mathbf{n} \, dS = \int_V f \, d\mathbf{x}
\]

for all "control volumes" \( V \) (6.1.1)

energy stored in \( V \) \hspace{1cm} \text{power flux through } \partial V \hspace{1cm} \text{heat generation in } V

In this case the heat flux was given by

Fourier's law \( \mathbf{j}(\mathbf{x}) = -\kappa(\mathbf{x}) \nabla u(\mathbf{x}) \), \( \mathbf{x} \in \Omega \), (2.5.3)
or its extended version (7.1.5). In Fourier's law the flux is a \textit{linear} function of \( u \).

Conversely, for the \textit{flux function} \( f : U \times \Omega \mapsto \mathbb{R}^d \) in (8.2.1) we assume

\( f \) only depends on local state \( u \), not on derivatives of \( u \)!

On the other hand we go far beyond Fourier's law, since

\( f \) will in general be a \textit{non-linear} function of \( u \)!
Remark 8.2.3 (Diffusive flux).

Taking into account the relationship with heat “diffusion”, a flux function of the form of Fourier’s law (2.5.3)

\[ f(u) = -\kappa(x) \nabla u, \]

is called a diffusive flux.

Now, integrate (8.2.1) over time period \([t_0, t_1] \subset [0, T]\) and use fundamental theorem of calculus:

**Space-time integral form of (8.2.1), cf. (8.1.15),**

\[
\int_{V} u(x, t_1) \, dx - \int_{V} u(x, t_0) \, dx + \int_{t_0}^{t_1} \int_{\partial V} f(u, x) \cdot n \, dS(x) \, dt = \int_{t_0}^{t_1} \int_{V} s(u, x, t) \, dx \, dt \quad (8.2.4)
\]
for all \( V \subset \Omega, \ 0 < t_0 < t_1 < T, \ n \triangleq \text{exterior unit normal at } \partial V \)

\[ \nabla \cdot \mathbf{f}(u, \mathbf{x}) = s(u, \mathbf{x}, t) \quad \text{in } \tilde{\Omega}. \]  
\[ (8.2.5) \]

\( \nabla \) acting on spatial variable \( \mathbf{x} \) only

\[ u(\mathbf{x}, 0) = u_0(\mathbf{x}), \quad \mathbf{x} \in \Omega \]

Special case \( d = 1 \iff (8.2.5) = \text{one-dimensional scalar conservation law for “density” } u : \tilde{\Omega} \to \mathbb{R} \)

\[ \frac{\partial u}{\partial t}(x, t) + \frac{\partial}{\partial x}(f(u(x, t), x)) = s(u(x, t), x, t) \quad \text{in } ]\alpha, \beta[ \times ]0, T[, \quad \alpha, \beta \in \mathbb{R} \cup \{\pm\infty\}. \]  
\[ (8.2.6) \]
Remark 8.2.7 (Boundary values for conservation laws).

Suitable boundary values on $\partial \Omega \times [0, T]$ → usually tricky question (highly $f$-dependent)

Reason: remember discussion in Rem. 8.1.11, meaningful boundary conditions hinge on knowledge of local (in space and time) transport direction, which, in a non-linear conservation law, will usually depend on the unknown solution $u = u(x, t)$.

Δ

8.2.2 Characteristics

We consider Cauchy problem $(\Omega = \mathbb{R})$ for one-dimensional scalar conservation law (8.2.6):

$$
\begin{align*}
\frac{\partial u}{\partial t} + \frac{\partial}{\partial x} f(u) &= 0 \quad \text{in } \mathbb{R} \times [0, T], \\
u(x, 0) &= u_0(x) \quad \text{in } \mathbb{R}.
\end{align*}
$$

(8.2.9)
Assumption: flux function $f : \mathbb{R} \mapsto \mathbb{R}$ smooth ($f \in C^2$) and convex \cite[Def. 5.5.2]{29}.

Recall \cite[Thm. 5.5.2]{29}: $f$ convex $\Rightarrow$ derivative $f'$ increasing.

**Definition 8.2.10** (Characteristic curve for one-dimensional scalar conservation law).

A curve $\Gamma := (\gamma(\tau), \tau) : [0, T] \mapsto \mathbb{R} \times ]0, T[$ in the $(x, t)$-plane is a characteristic curve of (8.2.9), if

$$\frac{d}{d\tau}\gamma(\tau) = f'(u(\gamma(\tau), \tau)) , \quad 0 \leq \tau \leq T , \quad (8.2.11)$$

where $u$ is a continuously differentiable solution of (8.2.9).
Example 8.2.12 (Characteristics for advection).

Constant linear advection (8.1.5): \( f(u) = vu \)

\[ \gamma(\tau) = v\tau + c, \quad c \in \mathbb{R}. \]

solution (8.1.6) \( u(x,t) = u_0(x - vt) \)

meaningful for any \( u_0 \) ! (cf. Sect. 7.3.2)
This example reveals a close relationship between streamlines (→ Sect. 7.1.1) and characteristic curves. That the latter are a true generalization of the former is also reflected by the following simple observation, which generalizes the considerations in Sect. 7.3.2, (7.3.9).

**Lemma 8.2.13** (Classical solutions and characteristic curves).

Smooth solutions of (8.2.9) are constant along characteristic curves.

**Proof.** Apply chain rule twice, cf. (7.3.9), and use the defining equation (8.2.11) for a characteristic curve:

\[
\frac{d}{d\tau} u(\gamma(\tau), \tau) = \frac{\partial u}{\partial x}(\gamma(\tau), \tau) \frac{d}{d\tau} \gamma(\tau) + \frac{\partial u}{\partial t}(\gamma(\tau), \tau)
\]

(8.2.11)

\[
= \frac{\partial u}{\partial x}(\gamma(\tau), \tau) \cdot f'(u(\gamma(\tau), \tau)) + \frac{\partial u}{\partial t}(\gamma(\tau), \tau)
\]

chain rule

\[
= \left( \frac{\partial}{\partial x} f(u) \right)(\gamma(\tau), \tau) + \frac{\partial u}{\partial t}(\gamma(\tau), \tau) = 0.
\]

notation: \(f' \triangleq \text{derivative of flux function } f : U \subset \mathbb{R} \mapsto \mathbb{R}\)
So, \( u \) is constant on a characteristic curve.

\( \Rightarrow \quad f'(u) \) is constant on a characteristic curve.

(8.2.11) \( \Rightarrow \) slope of characteristic curve is constant!

Characteristic curve through \((x_0, 0)\) = straight line \((x_0 + f'(u_0(x_0))\tau, \tau), 0 \leq \tau \leq T!\)

Implicit solution formula for (8.2.9) \( (f' \) monotone !):

\[
u(x, t) = u_0(x - f'(u(x, t))t) . \tag{8.2.14}
\]

Example 8.2.15 (Breakdown of characteristic solution formula).
for Burger’s equation (8.1.16):

\( f(u) = \frac{1}{2}u^2 \) smooth and strictly convex

\[ f'(u) = u \] (increasing)

\( \text{if} \ u_0 \text{ smooth and decreasing} \)

\[ \text{characteristic curves intersect !} \]

\[ \text{solution formula (8.2.14) becomes invalid} \]
$t < 1.3$: solution by (8.2.14)

the wave breaks: “multivalued solution”

breakdown of classical solutions & Ex. 8.2.12  ➔  new concept of solution of (8.2.9)
8.2.3 Weak solutions

“Space-time Gaussian theorem”

\[ \frac{\partial u}{\partial t} + \frac{\partial}{\partial x} f(u) = 0 \] \hspace{1cm} (8.2.16)

\[ \text{div}(x,t) \begin{pmatrix} f(u) \\ u \end{pmatrix} = 0 \text{ in } \tilde{\Omega}. \] \hspace{1cm} (8.2.17)

∀ “space-time control volumes” \( \tilde{V} \subset \tilde{\Omega} \):

\[ \int_{\partial \tilde{V}} \begin{pmatrix} f(u(\tilde{x})) \\ u(\tilde{x}) \end{pmatrix} \cdot \begin{pmatrix} n_x(\tilde{x}) \\ n_t(\tilde{x}) \end{pmatrix} \, dS(\tilde{x}) = 0, \]

\( \tilde{n} := (n_x, n_t)^T \) \( \hat{=} \) space-time unit normal

(8.2.17) for space-time rectangle \( \tilde{V} = [x_0, x_1] \times [t_0, t_1] \) ➤ integral form of (8.2.16), cf. (8.2.4):

\[ \int_{x_0}^{x_1} u(x, t_1) \, dx - \int_{x_0}^{x_1} u(x, t_0) \, dx = \int_{t_0}^{t_1} f(u(x_0, t)) \, dt - \int_{t_0}^{t_1} f(u(x_1, t)) \, dt. \] \hspace{1cm} (8.2.18)
Still, \((8.2.18)\) encounters problems, if a discontinuity of \(u\) coincides with an edge of the space-time rectangle.

Idea: Obtain weak form of \((8.2.16)\) from \((8.2.17)\) by integration by parts, that is, application of Green's first formula Thm. 2.4.7 in space-time!

**STEP I:** Test \((8.2.17)\) with compactly supported smooth function \(\Phi : \tilde{\Omega} \mapsto \mathbb{R}, \Phi(\cdot, T) = 0\), and integrate over space-time cylinder \(\tilde{\Omega} = \mathbb{R} \times [0, T]\):

\[
(8.2.17) \quad \int_{\tilde{\Omega}} \text{div}(x,t) \left( \frac{f(u)}{u} \right) \Phi(x,t) \, dx \, dt = 0.
\]

**STEP II:** Perform integration by parts using Green's first formula Thm. 2.4.7 on \(\tilde{\Omega}\):

\[
\int_{\tilde{\Omega}} \text{div}(x,t) \left( \frac{f(u)}{u} \right) \Phi(x,t) \, dx \, dt = 0
\]

Thm. 2.4.7 \(\Rightarrow\)

\[
\int_{\tilde{\Omega}} \left( \frac{f(u)}{u} \right) \cdot \text{grad}(x,t) \Phi \, dx \, dt + \int_{-\infty}^{\infty} u(x,0) \Phi(x,0) \, dx = 0,
\]
because \( \partial \Omega = \mathbb{R} \times \{0\} \cup \mathbb{R} \times \{T\} \) with “normals” \( n = \begin{pmatrix} 0 \\ -1 \end{pmatrix} \) (\( t = 0 \) boundary) and \( n = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \) (\( t = T \) boundary), which has to be taken into account in the boundary term in Green’s formula. The “\( t = T \) boundary part” does not enter as \( \Phi(\cdot, T) = 0 \).

Note that \( u(x, 0) \) is fixed by the initial condition: \( u(x, 0) = u_0(x) \).

**Definition 8.2.19** (Weak solution of Cauchy problem for scalar conservation law).

For \( u_0 \in L^\infty(\mathbb{R}), u : \mathbb{R} \times ]0, T[ \to \mathbb{R} \) is a weak solution of the [Cauchy problem](8.2.9), if

\[
\begin{align*}
 u & \in L^\infty(\mathbb{R} \times ]0, T[) \quad \land \quad \int_{-\infty}^{\infty} \int_{0}^{T} \left\{ u \frac{\partial \Phi}{\partial t} + f(u) \frac{\partial \Phi}{\partial x} \right\} \, dt \, dx + \int_{-\infty}^{\infty} u_0(x) \Phi(x, 0) \, dx = 0,
\end{align*}
\]

for all \( \Phi \in C^\infty_0(\mathbb{R} \times [0, T]), \Phi(\cdot, T) = 0 \).

**Remark 8.2.21** (Properties of weak solutions).
By reversing integration by parts, it is easy to see that

\[ u \text{ weak solution of (8.2.9)} \& u \in C^1 \iff u \text{ classical solution of (8.2.9)}. \]

Arguments from mathematical integration theory confirm

\[ u \in L^\infty_{\text{loc}}(\mathbb{R} \times ]0, T[) \text{ weak solution of (8.2.9)} \Rightarrow u \text{ satisfies integral form (8.2.18)} \]

for “almost all” \( x_0 < x_1, 0 < t_0 < t_1 < T \).
8.2.4 Jump conditions

For piecewise smooth vectorfield \( j : \Omega \subset \mathbb{R}^2 \):

“\( \text{div} \ j = 0 \)”

\[ \int_{\partial V} j \cdot n \, dS = 0 \quad \forall \text{ control volumes } V \subset \Omega \]

Necessary condition:

Continuity of normal components
across discontinuities

discontinuous divergence-free vectorfield

To see this, consider a slender tiny rectangle aligned with a line of discontinuity of \( j \). In the absence of normal continuity a net flux through its boundary will result, provided that the rectangle is small enough (“pillbox argument”).
Apply this insight to vectorfield on space-time domain \( \tilde{\Omega} = \mathbb{R} \times [0, T] \):

\[
\frac{\partial u}{\partial t} + \frac{\partial}{\partial x} f(u) = 0 \quad \Leftrightarrow \quad \text{div}_{(x,t)} \left( \begin{pmatrix} f(u) \\ u \end{pmatrix} \right) = 0 \quad \text{in} \quad \tilde{\Omega}.
\]

\( (8.2.17) \)

Normal at \( C^1 \)-curve \( \Gamma := \tau \mapsto (\gamma(\tau), \tau) \) in \( (\gamma(\tau), \tau) \)

\[
\tilde{n} = \frac{1}{\sqrt{1 + |\dot{s}|^2}} \begin{pmatrix} 1 \\ -\dot{s} \end{pmatrix}, \quad \dot{s} := \frac{d\gamma}{d\tau}(\tau) \quad \text{"speed of curve"}.
\]

To see this, recall that the normal is orthogonal to the tangent vector \( (\dot{s}) \) and that in 2D the direction orthogonal to \( (x_1, x_2) \) is given by \( (-x_2, x_1) \).
"normal continuity" of piecewise smooth vectorfield \((f(u), u)^T\)

\[
\left( \frac{1}{\dot{s}} \right) \cdot \begin{bmatrix} [f(u)] \\ [u] \end{bmatrix} = 0 ,
\]

where \([\cdot]\) \(\equiv\) jump across \(\Gamma\).

Terminology: \((8.2.22) = \text{Rankine-Hugoniot (jump) condition, shorthand notation:} \)

\[
\dot{s}(u_l - u_r) = f_l - f_r , \quad \dot{s} := \frac{d\gamma}{d\tau} \quad \text{"propagation speed of discontinuity"} \quad (8.2.23)
\]

Remark 8.2.26 (Discontinuity connecting constant states).
The simplest situation compliant with Rankine-Hugoniot jump condition: \textit{constant states} to the left and right of the curve of discontinuity (8.2.22):

$$u(x,t) = \begin{cases} u_l \in \mathbb{R} , & \text{for } x < \dot{s}t , \\ u_r \in \mathbb{R} , & \text{for } x < \dot{s}t , \end{cases}$$ (8.2.27)

with \textit{constant} speed $\dot{s}$ of discontinuity, according to (8.2.23) given by (for $u_l \neq u_r$)

$$\dot{s} = \frac{f(u_l) - f(u_r)}{u_l - u_r}.$$ 

8.2.5 Riemann problem
Rem. 8.2.26: situation of locally constant states in particularly easy.

Consider: **Cauchy-problem** (8.2.9) for piecewise constant initial data \( u_0 \).

**Definition 8.2.28** (Riemann problem).

\[
 u_0(x) = \begin{cases} 
 u_l \in \mathbb{R}, & \text{if } x < 0, \\
 u_r \in \mathbb{R}, & \text{if } x > 0.
\end{cases} \quad \Rightarrow \quad \text{Riemann problem for (8.2.9)}
\]

Assumption, *cf.* Sect. 8.2.2: flux function \( f : \mathbb{R} \mapsto \mathbb{R} \) smooth & convex

\( f' \) non-decreasing  ➤ pattern of characteristic curves for Riemann problem:
**Definition 8.2.29** (Shock).

If $\Gamma$ is a smooth curve in the $(x, t)$-plane and $u$ a weak solution of (8.2.9), a discontinuity of $u$ across $\Gamma$ is called a shock.

**Rem. 8.2.26**  ➤ shock speed $s$  $\iff$ Rankine-Hugoniot jump conditions:

\[(x_0, t_0) \in \Gamma: \quad \dot{s} = \frac{f(u_l) - f(u_r)}{u_l - u_r}, \quad u_l := \lim_{\epsilon \to 0} u(x_0 - \epsilon, t_0), \quad u_r := \lim_{\epsilon \to 0} u(x_0 + \epsilon, t_0).\]  

\[(8.2.30)\]
Lemma 8.2.31 (Shock solution of Riemann problem).

\[
u(x, t) = \begin{cases} 
  u_l & \text{for } x < \dot{s}t, \\
  u_r & \text{for } x > \dot{s}t,
\end{cases}
\]

\[
\dot{s} := \frac{f(u_l) - f(u_r)}{u_l - u_r}, \quad x \in \mathbb{R}, \ 0 < t < T,
\]

is weak solution of the Riemann problem (→ Def. 8.2.28) for (8.2.9).

Burgers flux \( f(u) = \frac{1}{2}u^2 \), \( u_l > u_r \): characteristic curves impinge on shock
Burgers flux $f(u) = \frac{1}{2}u^2$, $u_l < u_r$: characteristic curves emanate from shock (expansion shock)

**Example 8.2.32** (Vanishing viscosity for Burgers equation).

There is no such material as an “inviscid” fluid in nature, because in any physical system there will be a tiny amount of friction. This leads us to the very general understanding that conservation laws can usually be regarded as limit problems $\epsilon = 0$ for singularly perturbed transport-diffusion problems with an “$\epsilon$-amount” of diffusion.

In 1D, for any $\epsilon > 0$ these transport-diffusion problems will possess a unique smooth solution. Studying its behavior for $\epsilon \to 0$ will tell us, what are “physically meaningful” solutions for the conservation
law. This consideration is called the **vanishing viscosity** method to define solutions for conservation laws.

Here we pursue this idea for Burgers equation, see Sect. 8.1.2.

Viscous Burgers equation:

\[
\frac{\partial u}{\partial t} + \frac{\partial}{\partial x} \left( \frac{1}{2} u^2 \right) = \epsilon \frac{\partial^2 u}{\partial x^2}.
\] (8.2.33)

This is the **dissipative (viscous) term**.

Travelling wave solution of Riemann problem for (8.2.33) via Cole-Hopf transform → [12, Sect. 4.4.1]

\[
u_\epsilon(x, t) = w(x - s t), \quad w(\xi) = u_r + \frac{1}{2}(u_l - u_r)(1 - \tanh \left( \frac{\xi(u_l - u_r)}{4\epsilon} \right)) , \quad s = \frac{1}{2}(u_l + u_r).
\]
\[ u_\epsilon(x, t) = \text{classical solution of (8.2.33) for all } t > 0, \]
\[ x \in \mathbb{R} \text{ (only for } u_l > u_r \text{).} \]

\[ u_l > u_r, \quad t = 0.5 \]

emerging shock for \( \epsilon \to 0 \)

\[ u_\epsilon \to u \text{ from Lemma 8.2.31 in } L^\infty(\mathbb{R}). \]
Highly accurate numerical solution of Riemann problem for (8.2.33)

\[ u_l < u_r \]

no shock as \( \epsilon \to 0 \! \)

\( u_\epsilon \to \) a piecewise linear function!

Let us try to derive a (weak) solution of the homogeneous scalar conservation law (8.2.16) with the structure observed in Ex. 8.2.32.

Idea: conservation law (8.2.16) homogeneous in spatial/temporal derivatives:

\[ \frac{\partial u}{\partial t} + \frac{\partial}{\partial x} f(u) = 0 \quad \text{in} \quad \mathbb{R} \times \mathbb{R}^+ \quad \Rightarrow \quad \frac{\partial u_\lambda}{\partial t} + \frac{\partial}{\partial x} f(u_\lambda) = 0 \quad \text{in} \quad \mathbb{R} \times \mathbb{R}^+ , \]
\( u_\lambda(x, t) := u(\lambda x, \lambda t), \lambda > 0 \). This suggests that we look for solutions of the Riemann problem that are constant on all straight lines in the \( x - t \)-plane that cross \((0, 0)^T\).

try similarity solution:

\[
\begin{align*}
\frac{\partial u}{\partial t} + \frac{\partial}{\partial x} f(u) &= 0 \\
\psi' &\equiv 0 \quad \lor \quad f'(\psi(w)) = w \quad \Leftrightarrow \quad \psi(w) = (f')^{-1}(w).
\end{align*}
\]

\( f' \) strictly monotone!
Lemma 8.2.34. *(Rarefaction solution of Riemann problem)*

If $f \in C^2(\mathbb{R})$ strictly convex, $u_l < u_r$, then

$$u(x, t) := \begin{cases} 
    u_l & \text{for } x < f'(u_l)t, \\
    g\left(\frac{x}{t}\right) & \text{for } f'(u_l) < \frac{x}{t} < f'(u_r), \\
    u_r & \text{for } x > f'(u_r)t,
\end{cases}$$

$g := (f')^{-1}$, is a weak solution of the Riemann problem (→ Def. 8.2.28).

**Proof.** We show that the rarefaction solution is a weak solution according to Def. 8.2.19 for $\Phi \in C_0^\infty(\mathbb{R} \times ]0, T[)$

$$\int_0^T \left\{ \int_{-\infty}^{f'(u_l)t} u_l \frac{\partial \Phi}{\partial t} + f(u_l) \frac{\partial \Phi}{\partial x} \, dx + \int_{f'(u_l)t}^{f'(u_r)t} g\left(\frac{x}{t}\right) \frac{\partial \Phi}{\partial t} + f(g\left(\frac{x}{t}\right)) \frac{\partial \Phi}{\partial x} \, dx + \int_{f'(u_r)t}^\infty u_r \frac{\partial \Phi}{\partial t} + F(u_r) \frac{\partial \Phi}{\partial x} \, dx \right\} \, dt$$

$$= \int_0^T \int_{f'(u_l)t}^{f'(u_r)t} g'\left(\frac{x}{t}\right) \frac{x}{t^2} \Phi - f'(g\left(\frac{x}{t}\right)) \frac{1}{t} g'\left(\frac{x}{t}\right) \Phi \, dx \, dt = 0 ,$$
because \((f' \circ g)(x/t) = x/t\) and by fundamental theorem of calculus.

**Terminology:** solution of Lemma 8.2.34 = rarefaction wave: continuous solution!

Burger flux function \(f(u) = \frac{1}{2}u^2\), \(u_l < u_r\): rarefaction wave solutions

### 8.2.6 Entropy condition
Sect 8.2.5 ➤ Non-uniqueness of weak solutions:

If \( f' \) is increasing and \( u_l < u_r \) both a shock and a rarefaction wave provide valid weak solutions.

Riemann solution (Burgers equation):
- shock
- rarefaction wave

How to select “physically meaningful” = admissible solution?

Vanishing viscosity technique \( \rightarrow \) Ex. 8.2.32: add an “\( \epsilon \)-amount” of diffusion (“friction”) and study solution for \( \epsilon \to 0 \).
However, desirable: simple selection criteria (entropy conditions)

**Definition 8.2.35 (Lax entropy condition).**

\( u \) is the weak solution of (8.2.9), piecewise classical solution in a neighborhood of \( C^2 \)-curve \( \Gamma := (\gamma(\tau), \tau), 0 \leq \tau \leq T \), discontinuous across \( \Gamma \).

\( u \) satisfies the Lax entropy condition in \((x_0, t_0) \in \Gamma\): \( \iff f'(u_l) \geq \dot{s} := \frac{f(u_l) - f(u_r)}{u_l - u_r} > f'(u_r) \).

Characteristic curves must not emanate from shock \( \iff \) no “generation of information”

Parlance: shock satisfying Lax entropy condition = physical shock

Note: \( f' \) increasing \( \Rightarrow \) Def 8.2.35: necessary for physical shock \( u_l > u_r \)
Physically meaningful weak solution of conservation law = **entropy solution**

For *scalar* conservation laws with locally Lipschitz-continuous flux function $f$:

Existence & uniqueness of entropy solutions

**Remark 8.2.36 (General entropy solution for 1D scalar Riemann problem).** → [23]

Entropy solution of Riemann problem (→ Def. 8.2.28) for (8.2.9) with arbitrary $f \in C^1(\mathbb{R})$:

$$u(x, t) = \psi(x/t), \quad \psi(\xi) := \begin{cases} 
\arg\min_{u_l \leq u \leq u_r} (f(u) - \xi u), & \text{if } u_l < u_r, \\
\arg\max_{u_r \leq u \leq u_l} (f(u) - \xi u), & \text{if } u_l \geq u_r.
\end{cases}$$

(8.2.37)

**Example 8.2.38 (Entropy solution of Burgers equation).**
Analytical solution available for Burgers equation \((8.1.16)\) with initial data, see \(\text{[12, Sect. 3.4, Ex. 3]}\)

\[
  u_0(x) = \begin{cases} 
    0 & \text{if } x < 0 \text{ or } x > 1, \\
    1 & \text{if } 0 \leq x \leq 1. 
  \end{cases}
\]
Vector field in $x - t$-plane

\[
\begin{pmatrix}
    f(u(x,t)) \\
    u(x,t)
\end{pmatrix}
\]

for entropy solution $u = u(x,t)$

Observe the normal continuity across the shock: the vector field is tangential to the shock curve.

8.2.7 Properties of entropy solutions
Setting: \( u \in L^\infty(\mathbb{R} \times ]0, T[) \) weak (\( \rightarrow \) Def. 8.2.19) **entropy solution** of **Cauchy problem**

\[
\frac{\partial u}{\partial t} + \frac{\partial}{\partial x} f(u) = 0 \quad \text{in } \mathbb{R} \times ]0, T[, \quad u(x, 0) = u_0(x), \quad x \in \mathbb{R}.
\]  \( (8.2.9) \)

with **flux function** \( f \in C^1(\mathbb{R}) \) (not necessarily convex/concave).

Notation: \( \tilde{u} \in L^\infty(\mathbb{R} \times ]0, T[) \) \( \hat{=} \) entropy solution w.r.t. **initial data** \( \tilde{u}_0 \in L^\infty(\mathbb{R}) \).

**Theorem 8.2.39** (**Comparison principle** for scalar conservation laws).

\( \text{If } u_0 \leq \tilde{u}_0 \text{ a.e. on } \mathbb{R} \Rightarrow u \leq \tilde{u} \text{ a.e. on } \mathbb{R} \times ]0, T[ \)

With obvious consequences:

\( u_0(x) \in [\alpha, \beta] \text{ on } \mathbb{R} \Rightarrow u(x, t) \in [\alpha, \beta] \text{ on } \mathbb{R} \times ]0, T[ \)
\( L^\infty \)-stability (\( \Rightarrow \) no blow-up can occur!)

\[
\forall 0 \leq t \leq T: \|u(\cdot, t)\|_{L^\infty(\mathbb{R})} \leq \|u_0\|_{L^\infty(\mathbb{R})}. \tag{8.2.40}
\]

**Theorem 8.2.41** \((L^1\)-contractivity of evolution for scalar conservation law).

\[
\forall t \in ]0, T[, R > 0: \int_{|x|<R} |u(x, t)| \, dx \leq \int_{|x|<R+\dot{s}t} |u_0(x)| \, dx,
\]

with *maximal speed of propagation*

\[
\dot{s} := \max \{|f'(\xi)|: \inf_{x \in \mathbb{R}} u_0(x) \leq \xi \leq \sup_{x \in \mathbb{R}} u_0(x)\}. \tag{8.2.42}
\]

Thm. 8.2.41 \( \Rightarrow \) finite speed of propagation in conservation law, bounded by \( \dot{s} \) from (8.2.42):

As in the case of the wave equation \( \rightarrow \) Sect. 6.2.2:
maximal domain of dependence of \((\bar{x}, \bar{t}) \in \tilde{\Omega}\)

maximal domain of influence of \(I_0 \subset \mathbb{R}\)

Analogous to Thm. 6.2.18:

**Corollary 8.2.43** (Domain of dependence for scalar conservation law). \(\rightarrow [9, \text{Cor. 6.2.2}]\)

*The value of the entropy solution at \((\bar{x}, \bar{t}) \in \tilde{\Omega}\) depends only on the restriction of the initial data to \(\{x \in \mathbb{R}: |x - \bar{x}| < \dot{s} \bar{t}\}\).*
Another strand of theoretical results asserts that the solution of a 1D scalar conservation law cannot develop oscillations:

\[ u \] solves (8.2.9) \quad \Rightarrow \quad \text{No. of local extrema (in space) of } u(\cdot, t) \text{ decreasing with time}

### 8.3 Conservative finite volume discretization

**Example 8.3.1** (Naive finite difference scheme).

Cauchy problem for Burgers equation (8.1.16) rewritten using product rule:

\[
\frac{\partial u}{\partial t}(x, t) + u(x, t)\frac{\partial u}{\partial x}(x, t) = 0 \quad \text{in } \mathbb{R} \times ]0, T[.
\]
related to advection with velocity \( v(x, t) = u(x, t) \):

\[
\frac{\partial u}{\partial t}(x, t) + u(x, t)\frac{\partial u}{\partial x}(x, t) = 0 \quad \text{in} \quad \mathbb{R} \times ]0, T[.
\]

If \( u_0(x) \geq 0 \), then, by Thm. 8.2.39, \( u(x, t) \geq 0 \) for all \( 0 < t < T \), that is, positive direction of transport throughout.

Heeding guideline from Sect. 7.3.1: use upwind discretization (backward differences) in space!

on (infinite) equidistant grid, meshwidth \( h > 0 \), \( x_j = h j, j \in \mathbb{Z} \), obtain semi-discrete problem for nodal values \( \mu_j = \mu_j(t) \approx u(x_j, t) \)

\[
\frac{\partial u}{\partial t}(x, t) + u(x, t)\frac{\partial u}{\partial x}(x, t) = 0 \quad \text{in} \quad \mathbb{R} \times ]0, T[.
\]

\[
\mu_j(t) + \mu_j \frac{\mu_j - \mu_{j-1}}{h} = 0, \quad j \in \mathbb{Z}, \quad 0 < t < T.
\]  

(8.3.3)

Numerical experiment with Cauchy problem from Ex. 8.2.38, \( h = 0.08 \), integration of (8.3.3) with MATLAB ode45.
Observation from numerical experiment: OK for rarefaction wave, but scheme cannot capture speed of shock correctly!

Analysis: consider $\mu_j(0) = \begin{cases} 1 & \text{if } j < 0, \\ 0 & \text{if } j \geq 0. \end{cases}$

$\hookrightarrow$ **Riemann problem** with $u_0(x) = 1$ for $x < 0 - \epsilon$, $u_0(x) = 0$ for $x > 0 - \epsilon$, $\epsilon \ll 1$.

Entropy solution (for this $u_0$) = travelling shock ($\rightarrow$ Lemma 8.2.31), speed $\dot{s} = \frac{1}{2} > 0$

Numerical solution: $\tilde{\mu}(t) = \tilde{\mu}_0$ for all $t > 0$!

3-point FDM (8.3.3) “converges” to wrong solution!
8.3.1 Semi-discrete conservation form

Objective: spatial semi-discretization of Cauchy problem

\[ \frac{\partial u}{\partial t} + \frac{\partial}{\partial x} f(u) = 0 \quad \text{in} \; \mathbb{R} \times ]0, T[, \quad u(x, 0) = u_0(x), \quad x \in \mathbb{R}. \quad (8.2.9) \]

on (infinite) equidistant spatial mesh with mesh width \( h > 0 \)

\[ \mathcal{M} := \{ x_{j-1}, x_j : x_j := jh, \; j \in \mathbb{Z} \} . \quad (8.3.4) \]

mesh cells and dual cells
Finite volume interpretation of nodal unknowns $\mu_j (\leftrightarrow x_j, j \in \mathbb{Z})$: conserved quantities in dual cells $]x_{j-1/2}, x_{j+1/2}[$, midpoints $x_{j-1/2} := \frac{1}{2}(x_j + x_{j-1})$:

$$
\mu_j(t) \approx \frac{1}{h} \int_{x_{j-1/2}}^{x_{j+1/2}} u(x, t) \, dx.
$$  \hfill (8.3.5)

$$
\bar{\mu}(t) := (\mu_j(t))_{j \in \mathbb{Z}} \in \mathbb{R}^\mathbb{Z} \quad \leftrightarrow \quad u_N(x, t) = \sum_{j \in \mathbb{Z}} \mu_j(t) \chi_{]x_{j-1/2}, x_{j+1/2}[}(x). \quad \hfill (8.3.6)
$$

notation: characteristic function \( \chi_{]x_{j-1/2}, x_{j+1/2}[}(x) = \begin{cases} 
1 & \text{if } x_{j-1/2} < x \leq x_{j+1/2}, \\
0 & \text{elsewhere.}
\end{cases} \)

\( (\mu_j(t))_{j \in \mathbb{Z}} \quad \leftrightarrow \quad \text{piecewise constant approximation } u_N(t) \approx u(\cdot, t) \)

Note: \( u_N(t) \) discontinuous at dual cell boundaries \( x_{j+1/2} \)!

8.3
p. 868

R. Hiptmair
C. Schwab,
H.
Harbrecht
V.
Gradinaru
A. Chernov
SAM, ETHZ
By spatial integration over dual cells, which now play the role of the control volumes in (8.2.1):

\[
\frac{d}{dt} \int_{x_{j-1/2}}^{x_{j+1/2}} u(x, t) \, dx + f(u(x_{j+1/2}, t)) - f(u(x_{j-1/2}, t)) = 0, \quad j \in \mathbb{Z},
\]

(8.3.9)

(8.3.5)

\[
\frac{d\mu_j}{dt}(t) + \frac{1}{h}(f(u_N(x_{j+1/2}, t)) - f(u_N(x_{j-1/2}, t))) = 0, \quad j \in \mathbb{Z}.
\]

(8.3.10)

Problem: jump of \( u_N(t) \Rightarrow \) ambiguity of values \( u_N(x_{j+1/2}, t), u_N(x_{j-1/2}, t) \), as we encountered it in the context of upwind quadrature in Sect. 7.2.2.1.

Abstract “solution”:

Approximation \( f(u_N(x_{j+1/2}, t)) \approx f_{j+1/2}(t) := F(\mu_{j-m_l+1}(t), \ldots, \mu_{j+m_r}(t)), \quad j \in \mathbb{Z}, \)

with numerical flux function \( F : \mathbb{R}^{m_l+m_r} \mapsto \mathbb{R}, \quad m_l, m_r \in \mathbb{N}_0. \)

Note: the same numerical flux function is used for all dual cells!
Finite volume semi-discrete evolution for (8.2.9) in conservation form:

\[
\frac{d\mu_j}{dt}(t) = -\frac{1}{h} \left( F(\mu_{j-m_l+1}(t), \ldots, \mu_{j+m_r}(t)) - F(\mu_{j-m_l}(t), \ldots, \mu_{j+m_r-1}(t)) \right), \quad j \in \mathbb{Z}.
\] (8.3.11)

Numerical flux (function) \( F : \mathbb{R}^{m_l+m_r} \mapsto \mathbb{R} \)

Special case: 2-point numerical flux \((m_l = m_r = 1)\):

\[
F = F(v, w)
\]

\((v \triangleq \text{left state}, w \triangleq \text{right state})\)

\[
\frac{d\mu_j}{dt}(t) = -\frac{1}{h} \left( F(\mu_j(t), \mu_{j+1}(t)) - F(\mu_{j-1}(t), \mu_j(t)) \right), \quad j \in \mathbb{Z}.
\] (8.3.12)

Assumption on numerical flux functions: \( F \) Lipschitz-continuous in each argument.

**Code 8.3.13:** Wrapper code for finite volume evolution with 2-point flux

```matlab
function ufinal = consformevl(a,b,N,u0,T,F)
% finite volume discrete evolution in conservation form with 2-point flux, see (8.3.12)
% Cauchy problem over time [0,T] restricted to finite interval [a,b],
% equidistant mesh with meshwidth N cells, meshwidth \( h := \frac{b-a}{N} \).
% 2-point numerical flux function \( F = F(v,w) \) passed in handle \( F \)

h = (b-a)/N; x = a+0.5*h:h:b-0.5*h; % centers of dual cells
mu0 = h*u0(x)'; % vector \( \vec{\mu}_0 \) of initial cell averages (column vector)
% right hand side function for MATLAB ode solvers
```

\[8.3\]

p. 870
odefun = @(t,mu) (-1/h * fluxdiff(mu,F));

% timestepping by explicit Runge-Kutta method of order 5
options = odeset('abstol',1E-8,'reltol',1E-6,'stats','on');
[t,MU] = ode45(odefun,[0 T],mu0,options);

% 3D graphical output of \( u(x,t) \) over space-time plane
[X,T] = meshgrid(x,t);
figure; surf(X,T,MU/h); colormap(copper);
xlabel('{\bf x}', 'fontsize',14);
ylabel('{\bf t}', 'fontsize',14);
zlabel('{\bf u}', 'fontsize',14);
ufinal = MU(:,end);
end

function fd = fluxdiff(mu,F)
n = length(mu); fd = zeros(n,1);
% constant continuation of data outside \([a,b]\)
fd(1) = F(mu(1),mu(2)) - F(mu(1),mu(1));
for j=2:n-1
    fd(j) = F(mu(j),mu(j+1)) - F(mu(j-1),mu(j)); % see (8.3.12)
end
fd(n) = F(mu(n),mu(n)) - F(mu(n-1),mu(n));
end
8.3.2 Discrete conservation property

An evident first property of finite volume methods in conservation form:

\[ \mu_j(0) = \mu_0 \in \mathbb{R} \quad \forall j \in \mathbb{Z} \quad \Rightarrow \quad \mu_j(t) = \mu_0 \quad \forall j \in \mathbb{Z}, \quad \forall t > 0. \tag{8.3.15} \]

that is, constant solutions are preserved by the method.

A “telescopic sum argument” combined with the interpretation (8.3.6) shows that the conservation form (8.3.11) of the semi-discrete conservation law involves

\[ \frac{d}{dt} \int_{x_{k-1/2}}^{x_{m+1/2}} u_N(x, t) \, dx = h \sum_{l=k}^{m} \frac{d\mu_j}{dt}(t) = -(f_{m+1/2}(t) - f_{k-1/2}(t)) \quad \forall k, m \in \mathbb{Z}. \]
\[
\frac{d}{dt} \int_{x_{k-1/2}}^{x_{m+1/2}} u(x, t) \, dx = - \left( f(u(x_{j+1/2}, t)) - f(u(x_{k-1/2}, t)) \right),
\]

With respect to unions of dual cells and numerical fluxes, the semidiscrete solution \( u_N(t) \) satisfies a balance law of the same structure as a (weak) solution of (8.2.9).

Of course, the numerical flux function \( F \) has to fit the flux function \( f \) of the conservation law:

**Definition 8.3.16 (Consistent numerical flux function).**

A **numerical flux function** \( F : \mathbb{R}^{m_l+m_r} \to \mathbb{R} \) is **consistent** with the **flux function** \( f : \mathbb{R} \to \mathbb{R} \), if

\[
F(u, \ldots, u) = f(u) \quad \forall u \in \mathbb{R}.
\]

Focus: solution of **Riemann problem** (\( \to \) Def. 8.2.28) by finite volume method in conservation form (8.3.11):
Initial data “constant at $\pm \infty$”:
\[ \mu_j(0) = u_l, \quad \mu(-j)(0) = u_r \quad \text{for large} \quad j. \]

Consistency of the numerical flux function implies for large $m \gg 1$
\[ \frac{d}{dt} \int_{-x_m-1/2}^{x_{m+1/2}} u_N(x, t) \, dt = -(F(u_r, \ldots, u_r) - F(u_l, \ldots, u_l)) = -(f(u_r) - f(u_l)). \quad (8.3.18) \]

Exactly the same balance law holds for any weak solutions of the Riemann problem!

Situation: discrete solution $u_N(t)$ decreasing & supposed to approximate a shock
approximate location of shock at time $t$:

$$x_*(t) \in \mathbb{R}: \quad \int_{x_*(t)}^{\infty} u_l - u_N(x, t) \, dx = \int_{x_*(t)}^{\infty} u_N(x, t) - u_r \, dx$$

equality of yellow areas

$$\int_{x_m - 1/2}^{x_{m+1/2}} u_N(x, t) \, dx = (x_*(t) + x_{-m-1/2}) u_l + (x_{m+1/2} - x_*(t)) u_r.$$  

$$(8.3.18) \quad \frac{dx_*}{dt}(t) = \frac{1}{u_l - u_r} \sum_{j \in \mathbb{Z}} \frac{d\mu_j}{dt}(t) = \frac{f(u_l) - f(u_r)}{u_l - u_r} \quad (8.2.23) \quad \equiv \dot{s}.$$  

Conservation form with consistent numerical flux yields correct “discrete shock speed” (not liable to effect of Ex. 8.3.1)
8.3.3 Numerical flux functions

8.3.3.1 Central flux

Example 8.3.19 (Central flux for Burgers equation).

- Cauchy problem for Burgers equation (8.1.16) (flux function \( f(u) = \frac{1}{2}u^2 \)) from Ex. 8.2.38 (“box” initial data)
- Spatial finite volume discretization in conservation form (8.3.11) with central numerical fluxes

\[
F_1(v, w) := \frac{1}{2}(f(v) + f(w)) \quad , \quad F_2(v, w) := f\left(\frac{1}{2}(v + w)\right) .
\] (8.3.20)

Obviously the 2-point numerical fluxes \( F_1 \) and \( F_2 \) are consistent according to Def. 8.3.16. The resulting spatially semi-discrete scheme is given by, see (8.3.12)

\[
F_1: \quad \frac{d\mu_j}{dt}(t) = -\frac{1}{2h}(f(\mu_{j+1}(t)) - f(\mu_{j-1}(t))) ,
\]

\[
F_2: \quad \frac{d\mu_j}{dt}(t) = -\frac{1}{h}(f\left(\frac{1}{2}(\mu_j(t) + \mu_{j+1}(t))\right) - f\left(\frac{1}{2}(\mu_j(t) + \mu_{j-1}(t))\right)) .
\]
timestepping based on adaptive Runge-Kutta method `ode45` of MATLAB

\[
\text{opts = odeset('abstol',1E-7,'reltol',1E-6);}
\]

Fully discrete evolution for central numerical flux $F_1$: $h = 0.03$

![Fig. 270](image.png)
Fully discrete evolution for central numerical flux $F_2$: $h = 0.017$
Observation: massive spurious oscillations utterly pollute numerical solution

Example 8.3.21 (Central flux for linear advection).
Cauchy problem (8.1.5): constant velocity scalar linear advection, \( v = 1 \), flux function \( f(u) = vu \)

\[
\frac{\partial u}{\partial t} + v \frac{\partial u}{\partial x} = 0 \quad \text{in} \quad \tilde{\Omega} = \mathbb{R} \times ]0, T[ , \quad u(x, 0) = u_0(x) \quad \forall x \in \mathbb{R} .
\]

(8.1.5)

Cauchy problem for 1D transport equation (7.3.7)

Finite volume spatial discretization in conservation form (8.3.11) with central numerical fluxes from (8.3.20):

\[
F_1(v, w) := \frac{1}{2} (f(v) + f(w))
\]

\[
F_2(v, w) := f\left(\frac{1}{2}(v + w)\right)
\]

\[
\Rightarrow \quad \frac{d\mu_j(t)}{dt} = -\frac{v}{2h} (\mu_{j+1}(t) - \mu_{j-1}(t)) , \quad j \in \mathbb{Z} .
\]

Spatial semi-discretization using linear finite element Galerkin discretization of convective term, see (7.2.15).

Sect. 7.3.1: this method is prone to spurious oscillations, see Ex. 7.3.4.

This offers an explanation also for its failure for Burgers equation, see Ex. 8.3.19.
8.3.3.2 Lax-Friedrichs flux

Sect. [7.2.2.2]: artificial diffusion cures instability of central difference quotient

\[
\frac{\partial u}{\partial t} + \left( \frac{ch}{2} \right) \frac{-\mu_{j-1} + 2\mu_j - \mu_{j+1}}{h^2} \uparrow + c \frac{\mu_{j+1} - \mu_{j-1}}{2h} \downarrow = 0, \quad j \in \mathbb{Z}.
\]

Can this be rewritten in conservation form (8.3.11)? YES!

\[
\left( \frac{ch}{2} \right) \frac{-\mu_{j-1} + 2\mu_j - \mu_{j+1}}{h^2} + c \frac{\mu_{j+1} - \mu_{j-1}}{2h} = \frac{1}{h} \left( F(\mu_j, \mu_{j+1}) - F(\mu_{j-1}, \mu_j) \right),
\]

with

\[
F(v, w) := \frac{c}{2}(v + w) - \frac{c}{2}(w - v).
\]

Central numerical flux diffusive/viscous numerical flux
Recall from Rem. 8.2.3: the flux function $f(u) = -\frac{\partial u}{\partial x}$ models diffusion. Hence, the diffusive numerical flux amounts to a central finite difference discretization of the partial derivative in space:

$$-\frac{\partial u}{\partial x}(x, t) \bigg|_{x=x_{j+1/2}} \approx -\frac{1}{h}(u(x_{j+1}, t) - u(x_j, t)).$$

How to adapt this to general scalar conservation laws?

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x} f(u) = \frac{\partial u}{\partial t} + f'(u) \frac{\partial u}{\partial x} = 0 \quad (8.3.25)$$

*local* speed of transport $\leftrightarrow c$

(local) Lax-Friedrichs flux

$$F_{LF}(v, w) = \frac{1}{2}(f(v) + f(w)) - \frac{1}{2} \max_{\min\{v,w\} \leq u \leq \max\{v,w\}} |f'(u)|(w-v). \quad (8.3.26)$$

**Example 8.3.27** (Lax-Friedrichs flux for Burgers equation).

same setting and conservative discretization as in Ex. 8.3.19

Numerical flux function: Lax-Friedrichs flux (8.3.26)
Fig. 272

\[ h = 0.066667, t=1.002336 \]

finite volume solution

exact solution

\[ h = 0.066667, t=2.000915 \]

finite volume solution

exact solution

\[ h = 0.066667, t=3.000435 \]

finite volume solution

exact solution

\[ h = 0.066667, t=4.000000 \]

finite volume solution

exact solution

\[ u(t,x) \]

\[ x \]

\[ t \]
Observation: spurious completely suppressed, qualitatively good resolution of both shock and rarification.

Effect of artificial diffusion: smearing of shock, cf. discussion in Ex. 7.2.27.

8.3.3.3 Upwind flux

Another idea for stable spatial discretization of stationary transport in Sect. 7.2.2.1:

“upwinding” = obtain information from where transport brings it

remedy for ambiguity of evaluation of discontinuous gradient in upwind quadrature
Ambiguity also faced in the evaluation of $f(u_N(x_{j+1/2}), t), f(u_N(x_{j-1/2}), t)$, see (8.3.10), which forced us to introduce numerical flux functions in (8.3.11).

(8.3.25): local velocity of transport at $(x, t) \in \tilde{\Omega}$ is given by $f'(u(x, t))$

ambiguous local velocity of transport at discontinuity of $u_N$!

Idea: deduce local velocity of transport from Rankine-Hugoniot jump condition (8.2.23)

local velocity of transport $= \begin{cases} f'(u) & \text{for unique state, } u = u_l = u_r \\ \frac{f(u_r) - f(u_l)}{u_r - u_l} & \text{at discontinuity.} \end{cases}$

$(u_l, u_r \hat{=} \text{states to left and right of discontinuity})$

upwind flux for scalar conservation law with flux function $f$:

$$F_{uw}(v, w) = \begin{cases} f(v) & \text{if } \dot{s} \geq 0 \\ f(w) & \text{if } \dot{s} < 0 \end{cases}, \quad \dot{s} := \begin{cases} \frac{f(w) - f(v)}{w - v} & \text{for } v \neq w \\ f'(v) & \text{for } v = w. \end{cases} \quad (8.3.28)$$

Example 8.3.29 (Upwind flux for Burgers equation).
same setting and conservative discretization as in Ex. 8.3.19

Numerical flux function: upwind flux (8.3.28)

Fig. 273
Example 8.3.30 (Upwind flux and transsonic rarefaction).

Cauchy problem (8.2.9) for Burgers equation (8.1.16), i.e., \( f(u) = \frac{1}{2}u^2 \) and initial data

\[
u_0(x) = \begin{cases} 
-1 & \text{for } x < 0 \text{ or } x > 1 , \\
1 & \text{for } 0 < x < 1 .
\end{cases}
\]
The entropy solution (→ Sect. 8.2.6) of this Cauchy problem features a transsonic rarefaction fan at $x = 0$: this is a rarefaction solution (→ Lemma 8.2.34) whose “edges” move in opposite directions.
Conservative finite volume discretization with upwind flux produces (stationary) *expansion shock* instead of transonic rarefaction!

Sect. 8.2.6: this is a weak solution, but it violates the entropy condition, “non-physical shock”.

*Example* 8.3.31 (Upwind flux: Convergence to *expansion shock*).

- Cauchy problem (8.2.9) for Burgers equation (8.1.16), i.e., $f(u) = \frac{1}{2}u^2$

- $u_0(x) = 1$ for $x > 0$, $u_0(x) = -1$ for $x < 0$
  - entropy solution = rarefaction wave ($\rightarrow$ Lemma 8.2.34)

- FV in conservation form, upwind flux (8.3.28), on equidistant grid, $x_j = (j + \frac{1}{2})h$, meshwidth $h > 0$
initial nodal values \( \mu_j(0) = \begin{cases} -1 & \text{for } j < 0, \\ 1 & \text{for } j \geq 0. \end{cases} \)

Semi-discrete evolution equation:

\[
\frac{d \mu_j}{dt}(t) = -\frac{1}{2h} \begin{cases} \mu_{j+1}^2(t) - \mu_j^2(t) & \text{for } j \geq 0, \\ \mu_j^2(t) - \mu_{j-1}^2(t) & \text{for } j < 0. \end{cases}
\]

\( \mu_j(t) = \mu_j(0) \) for all \( t \) \( \rightarrow \) for \( h \rightarrow 0 \), convergence to entropy violating expansion shock!

finite volume method may converge to non-physical weak solutions!

8.3.3.4 Godunov flux
The upwind flux (8.3.28) is a numerical flux of the form

$$F(v, w) = f(u^\downarrow(v, w))$$  
with an intermediate state  \( u^\downarrow(v, w) \in \mathbb{R} \).

For the upwind flux the intermediate state is not really “intermediate”, but coincides with one of the states \( v, w \) depending on the sign of the “local shock speed” \( \dot{s} := \frac{f(w) - f(v)}{w - v} \).

Idea: obtain suitable intermediate state as

$$u^\downarrow(v, w) = \psi(0), (8.3.32)$$

where \( u(x, t) = \psi(x/t) \) solves the Riemann problem (→ Def. 8.2.28)

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x} f(u) = 0 \ , \ u(x, 0) = \begin{cases} v , & \text{for } x < 0 , \\ w , & \text{for } x \geq 0 . \end{cases} (8.3.33)$$

We focus on \( f : \mathbb{R} \mapsto \mathbb{R} \) strictly convex & smooth (e.g. Burgers equations (8.1.16))

Riemann problem (8.3.33) (→ Def. 8.2.28) has the entropy solution (→ Sect. 8.2.6):

1. If \( v > w \)  
   discontinuous solution, shock  (→ Lemma 8.2.31)

$$u(t, x) = \begin{cases} v & \text{if } x < \dot{s} t , \\ w & \text{if } x > \dot{s} t , \end{cases} \quad \dot{s} = \frac{f(v) - f(w)}{v - w} . (8.3.34)$$
If \( v \leq w \) ➤ continuous solution, rarefaction wave (\( \rightarrow \) Lemma 8.2.34)

\[
u(t, x) = \begin{cases} 
v & \text{if } x < f'(v)t, \\
g(x/t) & \text{if } f'(v) \leq x/t \leq f'(w), \\
w & \text{if } x > f'(w)t, \end{cases}
\]

\( g := (f')^{-1} \). \quad (8.3.35)

➢ All weak solutions of a Riemann problem are of the form \( u(x, t) = \psi(x/t) \) with a suitable function \( \psi \), which is

- piecewise constant with a jump at \( \dot{s} := \frac{f(w)-f(v)}{w-v} \) for a shock solution (8.3.34),

- the continuous function (in the case of strictly convex flux function \( f \))

\[
\psi(\xi) := \begin{cases} 
v & \text{if } \xi < f'(v), \\
(f')^{-1}(\xi) & \text{if } f'(v) < \xi < f'(w), \\
w & \text{if } \xi > f'(v), \end{cases}
\]

provided that \( w > v \) = situation of a rarefaction solution (8.3.35), see Lemma 8.2.34.
\[ u(x,t)/t \]

\[ t \]

\[ x_{j-2} \quad x_{j-1} \quad x_{j} \quad x_{j+1} \quad x_{j+2} \quad x_{j+3} \]

\[ \hat{\infty} \equiv \text{piecewise constant function } u(0,t) \]

\[ \hat{\infty} = \text{shock in } (t,x)\text{-plane} \]

\[ \hat{\infty} = \text{rarefaction wave in } (t,x)\text{-plane} \]

**Fig. 277**

\[ u^+(v,w) = \begin{cases} 
  w & \text{if } v > w \wedge \dot{s} < 0 \text{ (shock 1)}, \\
  v & \text{if } v > w \wedge \dot{s} > 0 \text{ (shock 3)}, \\
  f'(v) & \text{if } v < w \wedge f'(v) > 0 \text{ (rarefaction 4)}, \\
  (f')^{-1}(0) & \text{if } v < w \wedge f'(v) \leq 0 \leq f'(w) \text{ (rarefaction 5)}. 
\end{cases} \]

(8.3.36)

for convex flux function \( f \)
Detailed analysis of (8.3.36):

\[ v > w \text{ (shock case): } f(u^\downarrow(v, w)) = \begin{cases} f(v), & \text{if } \frac{f(w) - f(v)}{w - v} > 0 \iff f(w) < f(v), \\ f(w), & \text{if } \frac{f(w) - f(v)}{w - v} \leq 0 \iff f(w) \geq f(v). \end{cases} \]

\[ f(u^\downarrow(v, w)) = \max\{f(v), f(w)\}. \]
For a convex flux function $f$:

$$v < w \implies f'(v) \leq \frac{f(w) - f(v)}{w - v} \leq f'(w).$$

For $v < w$ (rarefaction case)

$$f(u^+(v, w)) = \begin{cases} 
    f(v), & \text{if } f'(v) > 0, \\
    f(z), & \text{if } f'(v) < 0 < f'(w), \\
    f(w), & \text{if } f'(w) < 0,
\end{cases}$$

where $f'(z) = 0 \iff f$ has a global minimum in $z$.

2-point numerical flux function according to (8.3.32) and (8.3.33): Godunov numerical flux.
Using general Riemann solution (8.2.37): for any flux function

\[
F_{GD}(v, w) = \begin{cases} 
\min_{v \leq u \leq w} f(u), & \text{if } v < w, \\
\max_{w \leq u \leq v} f(u), & \text{if } w \leq v.
\end{cases}
\]

for Burgers equation (8.1.16)

Remark 8.3.38 (Upwind flux and expansion shocks).

\[
F_{uw}(v, w) = F_{GD}(v, w), \text{ except for the case of transsonic rarefaction!}
\]

(transsonic rarefaction = rarefaction fan with edges moving in opposite direction, see Ex. 8.3.30)
What does the upwind flux $F_{uw}(v, w)$ from (8.3.28) yield in the case of transsonic rarefaction?

If $f$ convex, $v < w$, $f'(v) < 0 < f'(w)$,

$$F_{uw}(v, w) = f(\psi(0)),$$

where $u(x, t) = \psi(x/t)$ is a non-physical entropy-condition violating (→ Def. 8.2.35) expansion shock weak solution of (8.3.33).

Upwind flux treats transsonic rarefaction as expansion shock!

Example 8.3.39 (Godunov flux for Burgers equation).

Same setting and conservative discretization as in Ex. 8.3.30.

Numerical flux function: Godunov numerical flux (8.3.37)
Fig. 280

- $h = 0.066667, t = 1.004154$
- $h = 0.066667, t = 2.001356$
- $h = 0.066667, t = 3.001176$
- $h = 0.066667, t = 4.000000$

Finite volume solution
Exact solution
Observation: Transonic rarefaction captured by discretization, but small remnants of an expansion shock still observed.

8.3.4 Montone schemes

Observations made for some piecewise constant solutions $u_N(t)$ of semi-discrete evolutions arising from spatial finite volume discretization in conservation form (8.3.12):

- Ex. 8.3.27 (Lax-Friedrichs numerical flux (8.3.26))
- Ex. 8.3.39 (Godunov numerical flux (8.3.37))

\[
\min_{x \in \mathbb{R}} u_0(x) \leq u_N(x, t) \leq \max_{x \in \mathbb{R}} u_0(x)
\]

- no new local extrema in numerical solution

In these respects the conservative finite volume discretizations based on either the Lax-Friedrichs numerical flux or the Godunov numerical flux inherit crucial structural properties of the exact solution,
see Sect. 8.2.7, in particular, Thm. 8.2.39 and the final remark: they display structure preservation, cf. (5.7).

Is this coincidence for the special settings examined in Ex. 8.3.27 and Ex. 8.3.39?

Focus: semi-discrete evolution (8.3.12) resulting from finite volume discretization in conservation form on an equidistant infinite mesh

\[
\frac{d\mu_j}{dt}(t) = -\frac{1}{h} \left( F(\mu_j(t), \mu_{j+1}(t)) - F(\mu_{j-1}(t), \mu_j(t)) \right), \quad j \in \mathbb{Z}, \tag{8.3.12}
\]

for Cauchy problem

\[
\frac{\partial u}{\partial t} + \frac{\partial}{\partial x} f(u) = 0 \quad \text{in} \quad \mathbb{R} \times ]0, T[ \; , \; u(x, 0) = u_0(x) \; , \; x \in \mathbb{R}, \tag{8.2.9}
\]

induced by Lax-Friedrichs numerical flux (8.3.26)

\[
F_{LF}(v, w) = \frac{1}{2}(f(v) + f(w)) - \frac{1}{2} \max_{\min\{v,w\} \leq u \leq \max\{v,w\}} |f'(u)|(w - v). \tag{8.3.26}
\]

\[
\frac{d\mu_j}{dt} = -\frac{1}{2h} \left( \max_{u \in [\mu_{j+1}, \mu_j]} |f'(u)|(\mu_{j+1} - \mu_j) + \max_{u \in [\mu_j, \mu_{j+1}]} |f'(u)|(\mu_j - \mu_{j-1}) \right). \tag{8.3.40}
\]
Goal: show that $u_N(t)$ linked to $\mathbf{\bar{\mu}}(t)$ from (8.3.40) through piecewise constant reconstruction (8.3.6) satisfies

$$\min_{x \in \mathbb{R}} u_N(x, 0) \leq u_N(x, t) \leq \max_{x \in \mathbb{R}} u_N(x, 0) \quad \forall x \in \mathbb{R}, \quad \forall t \in [0, T]. \quad (8.3.41)$$

Recall from Sect. 8.2.7: estimate (8.3.41) for the exact solution $u(x, t)$ of (8.2.9) is a consequence of the comparison principle of Thm. 8.2.39 and the fact that constant initial data are preserved during the evolution. The latter property is straightforward for conservative finite volume spatial semi-discretization, see (8.3.15).

Goal: Establish comparison principle for finite volume semi-discrete solutions based on Lax-Friedrichs numerical flux:

$$\left\{ \begin{array}{l} \mathbf{\bar{\mu}}(t), \mathbf{\bar{\eta}}(t) \text{ solve (8.3.40)}, \\ \eta_j(0) \leq \mu_j(0) \quad \forall j \in \mathbb{Z} \end{array} \right\} \implies \eta_j(t) \leq \mu_j(t) \quad \forall j \in \mathbb{Z}, \quad \forall 0 \leq t \leq T.$$
Assumption: \( \vec{\mu} = \vec{\mu}(t) \) and \( \vec{\eta} = \vec{\eta}(t) \) solve (8.3.40) and satisfy for some \( t \in [0, T] \)

\[
\eta_k(t) \leq \mu_k(t) \quad \forall k \in \mathbb{Z}, \quad \xi := \eta_j(t) = \mu_j(t) \quad \text{for some } j \in \mathbb{Z}.
\]

Can \( \eta_j \) raise above \( \mu_j \)?

\[
\frac{d}{dt}(\mu_j - \eta_j) = -\frac{1}{h} \left( F_{LF}(\xi, \mu_{j+1}) - F_{LF}(\xi, \eta_{j+1}) + F_{LF}(\eta_{j-1}, \xi) - F_{LF}(\mu_{j-1}, \xi) \right).
\]

To show: \( \frac{d}{dt}(\mu_j - \eta_j) \geq 0 \quad \Rightarrow \quad \mu_j(t) \) will stay above \( \eta_j(t) \).

This can be concluded, if

\[
F_{LF}(\xi, \mu_{j+1}) - F_{LF}(\xi, \eta_{j+1}) \leq 0 \quad \text{and} \quad F_{LF}(\eta_{j-1}, \xi) - F_{LF}(\mu_{j-1}, \xi) \leq 0.
\]  \( \text{(8.3.42)} \)

The only piece of information we are allowed to use is

\[
\mu_{j+1} \geq \eta_{j+1} \quad \text{and} \quad \mu_{j-1} \geq \eta_{j-1}.
\]

This would imply (8.3.42), if \( F_{LF} \) was increasing in the first argument and decreasing in the second argument.
Definition 8.3.43 (Monotone numerical flux function).

A 2-point numerical flux function $F = F(v, w)$ is called monotone, if

\[ F \text{ is an increasing function of its first argument} \]

and

\[ F \text{ is a decreasing function of its second argument}. \]

Simple criterion: A continuously differentiable 2-point numerical flux function $F = F(v, w)$ is monotone, if and only if

\[ \frac{\partial F}{\partial v}(v, w) \geq 0 \quad \text{and} \quad \frac{\partial F}{\partial w}(v, w) \leq 0 \quad \forall (v, w). \quad (8.3.44) \]

Lemma 8.3.45 (Monotonicity of Lax-Friedrichs numerical flux and Godunov flux).

For any continuously differentiable flux function $f$ the associated Lax-Friedrichs flux (8.3.26) and Godunov flux (8.3.37) are monotone.
Proof.

1. Lax-Friedrichs numerical flux:

\[
F_{\text{LF}}(v, w) = \frac{1}{2}(f(v) + f(w)) - \frac{1}{2} \max_{\min\{v, w\} \leq u \leq \max\{v, w\}} |f'(u)|(w - v). \tag{8.3.26}
\]

Application of the criterion (8.3.44) is straightforward:

\[
\frac{\partial F_{\text{LF}}}{\partial v}(v, w) = f'(v) + \max_{\min\{v, w\} \leq u \leq \max\{v, w\}} |f'(u)| \geq 0, \tag{8.3.27}
\]

\[
\frac{\partial F_{\text{LF}}}{\partial w}(v, w) = f'(w) - \max_{\min\{v, w\} \leq u \leq \max\{v, w\}} |f'(u)| \leq 0. \tag{8.3.28}
\]

2. Godunov numerical flux

\[
F_{\text{GD}}(v, w) = \begin{cases} 
\min_{v \leq u \leq w} f(u), & \text{if } v < w, \\
\max_{w \leq u \leq v} f(u), & \text{if } w \leq v. 
\end{cases} \tag{8.3.37}
\]

\(v < w\): If \(v\) increases, then the range of values over which the minimum is taken will shrink, which makes \(F_{\text{GD}}(v, w)\) increase.

If \(w\) is raised, then the minimum is taken over a larger interval, which causes \(F_{\text{GD}}(v, w)\) to become smaller.
If \( v \) increases, then the range of values over which the maximum is taken will grow, which makes \( F_{GD}(v, w) \) increase.

If \( w \) is raised, then the maximum is taken over a smaller interval, which causes \( F_{GD}(v, w) \) to decrease.

\[ v \geq w: \]

\[ \text{If } v \text{ increases, then the range of values over which the maximum is taken will grow, which makes } F_{GD}(v, w) \text{ increase.} \]

\[ \text{If } w \text{ is raised, then the maximum is taken over a smaller interval, which causes } F_{GD}(v, w) \text{ to decrease.} \]

**Lemma 8.3.46** (Comparison principle for monotone semi-discrete conservative evolutions).

Let the 2-point numerical flux function \( F = F(v, w) \) be monotone (\( \rightarrow \) Def. 8.3.43) and \( \bar{\mu} = \bar{\mu}(t), \bar{\eta} = \bar{\eta}(t) \) solve (8.3.12). Then

\[ \eta_k(0) \leq \mu_k(0) \quad \forall k \in \mathbb{Z} \quad \Rightarrow \quad \eta_k(t) \leq \mu_k(t) \quad \forall k \in \mathbb{Z} \text{, } \forall 0 \leq t \leq T. \]

The assertion of Lemma 8.3.46 means that for monotone numerical flux, the semi-discrete evolution satisfies the comparison principle of Thm. 8.2.39.
Proof (of Lemma 8.3.46, following the above considerations for the Lax-Friedrichs flux).

The two sequences of nodal values satisfy (8.3.12)

$$\frac{d\mu_j}{dt}(t) = -\frac{1}{h} \left( F(\mu_j(t), \mu_{j+1}(t)) - F(\mu_{j-1}(t), \mu_j(t)) \right), \quad j \in \mathbb{Z},$$  \hspace{1cm} (8.3.47)

$$\frac{d\eta_j}{dt}(t) = -\frac{1}{h} \left( F(\eta_j(t), \eta_{j+1}(t)) - F(\eta_{j-1}(t), \eta_j(t)) \right), \quad j \in \mathbb{Z}. \hspace{1cm} (8.3.48)$$

Let \( t_0 \) be the earliest time, at which \( \eta \) “catches up” with \( \mu \) in at least one node \( x_j, j \in \mathbb{Z}, \) of the mesh, that is

$$\eta_k(t_0) \leq \mu_k(t_0) \quad \forall k \in \mathbb{Z}, \quad \xi := \eta_j(t_0) = \mu_j(t_0).$$

By subtracting (8.3.47) and (8.3.48) we get

$$\frac{d}{dt}(\mu_j - \eta_j)(t_0) = -\frac{1}{h} \left( F(\xi, \mu_{j+1}(t_0)) - F(\xi, \eta_{j+1}(t_0)) + F(\eta_{j-1}(t_0), \xi) - F(\mu_{j-1}(t_0), \xi) \right) \geq 0,$$

because for a monotone numerical flux function (\( \rightarrow \) Def. 8.3.43)

$$\eta_{j-1}(t_0) \leq \mu_{j-1}(t_0) \quad \Rightarrow \quad F(\eta_{j-1}(t_0), \xi) - F(\mu_{j-1}(t_0), \xi) \leq 0,$$

$$\eta_{j+1}(t_0) \leq \mu_{j+1}(t_0) \quad \Rightarrow \quad F(\xi, \mu_{j+1}(t_0)) - F(\xi, \eta_{j+1}(t_0)) \leq 0.$$

This means that “\( \eta \) cannot overtake \( \mu \)”: no value \( \eta \) can ever raise above \( \mu \). \( \square \)
Now we want to study the “preservation of the number of local extrema” during a semi-discrete evolution, another structural property of exact solutions of conservation laws, see Sect. 8.2.7.

Intuitive terminology: \( \bar{\mu} \) has a local maximum \( u_m \in \mathbb{R} \), if

\[
\exists j \in \mathbb{Z}: \quad \mu_j = u_m \quad \text{and} \quad \exists k_l < j < k_r \in \mathbb{N}: \quad \max_{k_l < l < k_r} \mu_l = u_m \quad \text{and} \quad \mu_{k_l} < u_m, \ \mu_{k_r} < u_m.
\]

In analogous fashion, we define a local minimum. If \( \bar{\mu} \) is constant for large indices, these values are also regarded as local extrema.
Counting local extrema of $\vec{\mu}$ and the associated piecewise constant reconstruction.

Lemma 8.3.49 (Non-oscillatory monotone semi-discrete evolutions).

If $\vec{\mu} = \vec{\mu}(t)$ solves (8.3.12) with a monotone numerical flux function $F = F(v, w)$ and $\vec{\mu}(0)$ has finitely many local extrema, then the number of local extrema of $\vec{\mu}(t)$ cannot be larger than that of $\vec{\mu}(0)$. 

\[ \mu_j / u_N \]

Local maximum

Local minimum

Local minimum

Local minimum

Fig. 281
Proof. \( i \) is index of local maximum of \( \vec{\mu}(t), t \) fixed

\[
\mu_{i-1}(t) \leq \mu_i(t), \quad \mu_{i+1}(t) \leq \mu_i(t) \quad \Rightarrow \quad F(\mu_i, \mu_{i+1}) \geq F(\mu_i, \mu_i) \geq F(\mu_{i-1}, \mu_i),
\]

\[
\Rightarrow \quad \frac{d}{dt}\mu_i(t) = -\frac{1}{h}(F(\mu_i, \mu_{i+1}) - F(\mu_{i-1}, \mu_i)) \leq 0.
\]

\[\Rightarrow\] maxima of \( \vec{\mu} \) subside, (minima of \( \vec{\mu} \) rise !)

Idea of proof:

No new (local) extrema can arise !

Adjacent values cannot "overtake":
local maximum: cannot move up
local minimum: cannot move down

---

Fig. 282
8.4 Timestepping

Focus: *Explicit* Runge-Kutta timestepping methods (→ Def. 6.1.26)

Recall [18, Def. 12.4.9]: for explicit \( s \)-stage Runge-Kutta single step methods the coefficients \( a_{ij} \) vanish for \( j \geq i, 1 \leq i, j \leq s \) ➤ the increments \( K_i \) can be computed in turns (without solving a non-linear system of equations).

Initial value problem for abstract semi-discrete evolution in \( \mathbb{R}^\mathbb{Z} \):

\[
\frac{d\vec{\mu}}{dt}(t) = \mathcal{L}_h(\vec{\mu}(t)) , \quad 0 \leq t \leq T , \quad \vec{\mu}(0) = \vec{\mu}_0 \in \mathbb{R}^\mathbb{Z} .
\]

(8.4.1)

Here: \( \mathcal{L}_h : \mathbb{R}^\mathbb{Z} \rightarrow \mathbb{R}^\mathbb{Z} \) \( \hat{=} \) (non-linear) finite difference operator, e.g. for finite volume semi-discretization in conservation form with 2-point numerical flux:

(8.3.12) ➤ \( (\mathcal{L}_h\vec{\mu})_j := -\frac{1}{h}(F(\mu_j, \mu_{j+1}) - F(\mu_{j-1}, \mu_j)) \).

(8.4.2)
\( L_h \) is local: \((L_h(\mu))_j\) depends only on “neighboring values” \(\mu_{j-n_l}, \ldots, \mu_{j+n_r}\).

Explicit \(s\)-stage Runge-Kutta single step method for (8.4.1), timestep \(\tau > 0\):

\begin{align*}
\vec{\kappa}_1 &= L_h(\vec{\mu}^{(k)}) , \\
\vec{\kappa}_2 &= L_h(\vec{\mu}^{(k)} + \tau a_{21} \vec{\kappa}_1) , \\
\vec{\kappa}_3 &= L_h(\vec{\mu}^{(k)} + \tau a_{31} \vec{\kappa}_1 + \tau a_{32} \vec{\kappa}_2) , \\
&\vdots \\
\vec{\kappa}_s &= L_h(\vec{\mu}^{(k)} + \tau \sum_{j=1}^{s-1} a_{sj} \vec{\kappa}_j) , \\
\end{align*}

\[ \vec{\mu}^{(k+1)} = \vec{\mu}^{(k)} + \tau \sum_{l=1}^{s} b_l \vec{\kappa}_l . \quad (8.4.3) \]

Here, \(a_{ij} \in \mathbb{R}\) and \(b_l \in \mathbb{R}\) are the coefficients from the Butcher scheme (6.1.27). For explicit RK-methods the coefficient matrix \(A\) is strictly lower triangular.
Setting: equidistant spatial mesh $\mathcal{M}$, meshwidth $h > 0$, nodes $x_j := hj$, $j \in \mathbb{Z}$, uniform timestep $\tau > 0$, $t_k := \tau k$, $k \in \mathbb{N}_0$.

Single step timestepping for (8.4.1) produces a sequence $(\vec{\mu}^{(k)})_{k \in \mathbb{N}_0}$

$$\mu_j^{(k)} \approx u(x_j, t_k), \quad j \in \mathbb{Z}, \quad k \in \mathbb{N}_0.$$  

**Fully discrete evolution**

$$\vec{\mu}^{(k+1)} = \mathcal{H}_h(\vec{\mu}^{(k-1)}), \quad k \in \mathbb{N}_0.$$

$\mathcal{H}_h : \mathbb{R}^\mathbb{Z} \mapsto \mathbb{R}^\mathbb{Z}$: fully discrete evolution operator, arising from applying single step timestepping (8.4.3) to (8.4.1).

**Example 8.4.4 (Fully discrete evolutions).**
Fully discrete evolution arising from finite volume semi-discretization in conservation form with 2-point numerical flux $F = F(v, w)$

$$\mathcal{L}_h \vec{\mu}(k+1) = \vec{\mu}(k) + \tau \mathcal{L}_h(\vec{\mu}(k)).$$

(8.4.2)

in combination with explicit Euler timestepping ($\equiv$ 1-stage explicit RK-method)

$$\vec{\mu}(k+1) = \vec{\mu}(k) + \tau L_h(\vec{\mu}(k)).$$

(8.4.5)

In the case of explicit trapezoidal rule timestepping [18, Eq. 12.4.6] (method of Heun)

$$\vec{\kappa} = \vec{\mu}(k) + \frac{\tau}{2} \mathcal{L}_h(\vec{\mu}(k)), \quad \vec{\mu}(k+1) = \vec{\mu}(k) + \tau \mathcal{L}_h(\vec{\kappa}).$$

$$\kappa_j := (\vec{\kappa})_j = \mu_j^{(k)} - \frac{\tau}{h}(F(\mu_j^{(k)}, \mu_{j+1}^{(k)}) - F(\mu_{j-1}^{(k)}, \mu_j^{(k)})),$$

(8.4.6)
8.4.1 CFL-condition

Remark 8.4.7 (Difference stencils).

Stencil notation: Visualization of flow of information in fully discrete explicit evolution (action of $\mathcal{H}_h$), cf. Fig. 201.

2-point numerical flux & explicit Euler timestepping

upwinding & explicit trapezoidal rule

2-point numerical flux & explicit trapezoidal rule

Fig. 283 2-point numerical flux & explicit Euler timestepping

Fig. 284 upwinding & explicit trapezoidal rule

Fig. 285 2-point numerical flux & explicit trapezoidal rule
A consequence of *explicit* timestepping: *locality* of fully discrete evolution operator:

\[ \exists m_l, m_r \in \mathbb{N}_0: \quad (\mathcal{H}(\bar{\mu}))_j = \mathcal{H}_j(\mu_{j-m_l}, \ldots, \mu_{j+m_r}) \cdot \tag{8.4.8} \]

If flux function \( f \) does not depend on \( x \), \( f = f(u) \) as in (8.2.9), we can expect \( \mathcal{H}_h \) is translation-invariant:

\[ \mathcal{H}_j = \mathcal{H} \quad \forall j \in \mathbb{Z} . \]

This is the case for (8.4.5) and (8.4.6).

By inspection of (8.4.3): if \( \mathcal{L}_h \) is translation-invariant

\[ (\mathcal{L}_h(\bar{\mu}))_j = \mathcal{L}(\mu_{j-n_l}, \ldots, \mu_{j+n_r}) , \quad j \in \mathbb{Z} , \]

and timestepping relies on an \( s \)-stage explicit Runge-Kutta method, then we conclude for \( m_l, m_r \) in (8.4.8)

\[ m_l \leq s \cdot n_l \quad , \quad m_r \leq s \cdot n_r . \]
Now we revisit a concept from Sect. 6.2.5, see, in particular, Rem. 6.2.41:

**Definition 8.4.9 (Numerical domain of dependence).**

Consider explicit translation-invariant fully discrete evolution $\vec{\mu}^{(k+1)} := \mathcal{H}(\vec{\mu}^{(k)})$ on uniform spatio-temporal mesh $(x_j = h_j, j \in \mathbb{Z}, t_k = k\tau, k \in \mathbb{N}_0)$ with

$$\exists m \in \mathbb{N}_0: \quad (\mathcal{H}(\vec{\mu}^{(k)}))_j = \mathcal{H}(\mu_{j-m}, \ldots, \mu_{j+m}), \quad j \in \mathbb{Z}. \quad (8.4.10)$$

Then the **numerical domain of dependence** is given by

$$D_h^-(x_j, t_k) := \{(x_m, t_l) \in \mathbb{R} \times [0, t_k]: j - m(k - l) \leq m \leq j + m(k - l)\}.$$ 

From Thm. 8.2.41 recall the **maximal analytical domain of dependence** for a solution of $\text{(8.2.9)}$

$$D^-(\bar{x}, \bar{t}) := \{(x, t) \in \mathbb{R} \times [0, \bar{t}]: \dot{s}_{\text{min}}(\bar{t} - t) \leq x - \bar{x} \leq \dot{s}_{\text{max}}(\bar{t} - t)\}. \quad (8.4.11)$$

with **maximals speeds of propagation**

$$\dot{s}_{\text{min}} := \min\{f'(\xi): \inf_{x \in \mathbb{R}} u_0(x) \leq \xi \leq \sup_{x \in \mathbb{R}} u_0(x)\}, \quad (8.4.11)$$

$$\dot{s}_{\text{max}} := \max\{f'(\xi): \inf_{x \in \mathbb{R}} u_0(x) \leq \xi \leq \sup_{x \in \mathbb{R}} u_0(x)\}. \quad (8.4.12)$$
$D^{-}(\bar{x}, \bar{t}) \subset \mathbb{R} \times [0, T]$

$D_{h}^{-}(\bar{x}, \bar{t})$ for 3-point stencil
Definition 8.4.13 (Courant-Friedrichs-Lewy (CFL)-condition). → Rem. 6.2.41
An explicit translation-invariant local fully discrete evolution $\vec{\mu}^{(k+1)} := \mathcal{H}(\vec{\mu}^{(k)})$ on uniform spatio-temporal mesh $(x_j = hj, j \in \mathbb{Z}, t_k = k\tau, k \in \mathbb{N}_0)$ as in Def. 8.4.9 satisfies the Courant-Friedrichs-Lewy (CFL-)condition, if the convex hull of its numerical domain of dependence contains the maximal analytical domain of dependence:

$$D^-(x_j, t_k) \subset \text{convex}(D^-_h(x_j, t_k))$$

By definition of $D^-(\vec{x}, \vec{t})$ and $D^-_h(x_j, t_k)$ sufficient for the CFL-condition is

$$\frac{\tau}{h} \leq \frac{m}{\dot{s}} \quad \iff \quad \text{timestep constraint}! \, . \quad (8.4.14)$$

This is a timestep constraint similar to the one encountered in Sect. 6.2.5 in the context of leapfrog timestepping for the semi-discrete wave equation.
As discussed in Rem. 6.2.41, we cannot expect convergence for fixed ratio $\tau : h$, for $h \to 0$ in case the CFL-condition is violated.

Refer to Fig. 204 for a “graphical argument”:

- $\bullet$ coarse grid,
- ■ fine grid,
- □ d.o.d.

Sequence of equidistant space-time grids of $\mathbb{R} \times [0, T]$ with $\tau = \gamma h$ ($\tau/h =$ meshwidth in time/space)

If $\gamma >$ CFL-constraint (8.4.14) then analytical domain of dependence $\not\subset$ numerical domain of dependence
8.4.2 Linear stability

In Sect. 6.1.4.2 and Sect. 6.2.5 we found that for explicit time-stepping

\[ \tau \leq O(h^r) \quad , \quad r \in \{1, 2\}, \quad \text{necessary to avoid exponential blow-up (instability)} \]

Is the timestep constraint (8.4.14) suggested by the CFL-condition also stipulated by stability requirements?

We are going to investigate the question only for the Cauchy problem for scalar linear advection in 1D with constant velocity \( v > 0 \):

\[ \frac{\partial u}{\partial t} + v \frac{\partial u}{\partial x} = 0 \quad \text{in} \quad \mathbb{R} \times]0, T[, \quad u(x, 0) = u_0(x) \quad \forall x \in \mathbb{R} . \quad (8.1.5) \]

Semi-discretization in space on equidistant mesh with meshwidth \( h > 0 \)

\( \text{linear, local, and translation-invariant semi-discrete evolution} \)

\[ \frac{d\bar{\mu}}{dt}(t) = \mathcal{L}_h(\bar{\mu}(t)) , \quad \text{with} \quad (\mathcal{L}_h(\bar{\mu}))(j) = \sum_{l=-m}^{m} c_l \mu_{j+l} , \quad j \in \mathbb{Z} , \quad (8.4.15) \]
for suitable weights $c_l \in \mathbb{R}$.

**Example 8.4.16** (Upwind difference operator for linear advection).

Finite volume semi-discretization of (8.1.5) in conservation form with Godunov numerical flux (8.3.37) (= upwind flux (8.3.28))

\[
(L_h(\vec{\mu}))_j = -\frac{v}{h}(\mu_j - \mu_{j-1}) \quad (8.4.17)
\]

In (8.4.15): $c_0 = -\frac{v}{h}$, $c_{-1} = \frac{v}{h}$.

Note: Lax-Friedrichs numerical flux (8.3.26) yields the same $L_h$. 
As in Sect. 6.1.4.2 and Sect. 6.2.5: **diagonalization technique** (with a new twist)

The new twist is that $\mathcal{L}_h$ acts on the sequence space $\mathbb{R}^\mathbb{Z}$!

**Idea:** trial expression for “eigenvectors”

$$\begin{pmatrix} \zeta_j \end{pmatrix} := \exp(i\xi j), \quad j \in \mathbb{Z}, \quad -\pi < \xi \leq \pi. \quad (8.4.18)$$

By straightforward computations:

$$\left( \mathcal{L}_h(\tilde{\mu}) \right)_j = \sum_{l=-m}^{m} c_l \mu_{j+l} \Rightarrow \mathcal{L}_h \zeta^\xi = \left( \sum_{l=-m}^{m} c_l \exp(i\xi l) \right) \zeta^\xi.$$

**Terminology:** The function $\hat{c}_h(\xi)$ is known as the **symbol** of the difference operator $\mathcal{L}_h$, *cf.* the concept of symbol of a differential operator.

**Spectrum of $\mathcal{L}_h$:**

$$\sigma(\mathcal{L}_h) = \{ \hat{c}_h(\xi) := \sum_{l=-m}^{m} c_l \exp(i\xi l): -\pi < \xi \leq \pi \}. \quad (8.4.19)$$
Example 8.4.20 (Spectrum of upwind difference operator).

Apply formula (8.4.19) with $c_0 = -\frac{v}{h}$, $c_{-1} = \frac{v}{h}$ (from (8.4.17)):

For $\mathcal{L}_h$ from (8.4.17): $\sigma(\mathcal{L}_h) = \left\{ \frac{v}{h}(\exp(-iv\xi) - 1): -\pi < \xi \leq \pi \right\}$

Spectrum of upwind finite difference operator for linear advection with velocity $v > 0$ (meshwidth $h > 0$)
Also here: diagonalization of semi-discrete evolution leads to decoupled scalar linear ODEs. However, now we have uncountably many “eigenvectors” $\vec{\zeta}$, so that linear combination becomes integration:

$$\vec{\mu}(t) = \int_{-\pi}^{\pi} \hat{\mu}(t, \xi) \vec{\zeta} \, d\xi \quad \Leftrightarrow \quad \mu_j(t) = \int_{-\pi}^{\pi} \hat{\mu}(t, \xi) \exp(i\xi j) \, d\xi.$$  \hspace{1cm} (8.4.21)

$$\frac{d\vec{\mu}}{dt}(t) = \mathcal{L}_h(\vec{\mu}(t)) \quad \Rightarrow \quad \frac{\partial \hat{\mu}}{\partial t}(t, \xi) = \hat{c}_h(\xi) \hat{\mu}(t, \xi).$$ \hspace{1cm} (8.4.22)

This is a family of scalar, linear ODEs parameterized by $\xi \in [-\pi, \pi]$.

\begin{remark}
Up to normalization the relationship

$$\vec{\mu}^{(0)} \in \mathbb{R}^\mathbb{Z} \quad \Leftrightarrow \quad \hat{\mu}^{(0)} : [-\pi, \pi] \mapsto \mathbb{C}$$

\end{remark}
from (8.4.21) is the **Fourier series transform**, which maps a sequence to a $2\pi$-periodic function. It has the important isometry property

$$\sum_{j=-\infty}^{\infty} |\mu_j|^2 = 2\pi \int_{-\pi}^{\pi} |\hat{\mu}(\xi)|^2 \, d\xi.$$  

The symbol $\hat{c}_h$ can be viewed as the *representation of a difference operator in Fourier domain.*

The decoupling manifest in (8.4.22) carries over to Runge-Kutta timestepping in the sense of the commuting diagram (6.1.54).

We introduce the Fourier transforms of the members of the sequence $\left(\vec{\mu}^{(k)}\right)_k$ created by timestepping

$$\vec{\mu}^{(k)} = \int_{-\pi}^{\pi} \mu^{(k)}(\xi) \hat{\zeta}_\xi \, d\xi \quad \iff \quad \mu_j^{(k)} = \int_{-\pi}^{\pi} \hat{\mu}^{(k)}(\xi) \exp(i\xi j) \, d\xi.$$  

(8.4.24)
Example 8.4.25 (Explicit Euler in Fourier domain).

Explicit Euler timestepping \[\text{[18, Eq. 12.2.4]}\] for semi-discrete evolution (8.4.15), see also (8.4.5),

\[
\hat{\mu}^{(k+1)}(\xi) = \hat{\mu}^{(k)}(\xi) + \tau L_h \hat{\mu}^{(k)}(\xi). 
\]

\[
\int_{-\pi}^{\pi} \hat{\mu}^{(k+1)}(\xi) \xi^\xi d\xi = (\text{Id} + \tau L_h) \int_{-\pi}^{\pi} \hat{\mu}^{(k)}(\xi) \xi^\xi d\xi = \int_{-\pi}^{\pi} \hat{\mu}^{(k)}(\xi)(1 + \tau \hat{c}_h(\xi)) d\xi. 
\]

\[
\hat{\mu}^{(k+1)}(\xi) = \hat{\mu}^{(k)}(\xi)(1 + \tau \hat{c}_h(\xi)). 
\]

In Fourier domain a single explicit Euler timestep corresponds to a multiplication of \(\hat{\mu} : [-\pi, \pi] \mapsto \mathbb{C}\) with the function \((1 + \tau \hat{c}_h) : [-\pi, \pi] \mapsto \mathbb{C}\).

Relate this to an explicit Euler step for the ODE \(\frac{\partial \hat{\mu}}{\partial t}(t, \xi) = \hat{c}_h(\xi)\hat{\mu}(t, \xi)\) from (8.4.22) with paramter \(\xi\):

\[
\hat{\mu}^{(k+1)}(\xi) = (1 + \tau \hat{c}_h(\xi))\hat{\mu}^{(k)}(\xi). 
\]
Generalize the observation made in the previous example:

\[ \tilde{\mu}^{(k)} = \int_{-\pi}^{\pi} \hat{\mu}^{(k)}(\xi) \tilde{\zeta}^\xi \, d\xi, \]

where \( \left( \eta^{(k)}(\xi) \right)_{k \in \mathbb{N}_0} \) is the sequence of approximations created by the Runge-Kutta method when applied to the scalar linear initial value problem

\[ \dot{y} = \hat{c}(\xi) y, \quad y(0) = \hat{\mu}^{(0)}(\xi). \]

Clearly, timestepping can only be stable, if blowup \( |\hat{\mu}^{(k)}(\xi)| \to \infty \) for \( k \to \infty \) can be avoided for all \( -\pi < \xi \leq \pi \).

From [18, Thm. 13.1.18] we know:
Theorem 8.4.26 (Stability function of explicit Runge-Kutta methods).

The execution of one step of size $\tau > 0$ of an explicit $s$-stage Runge-Kutta single step method (→ Def. 6.1.26) with Butcher scheme $\begin{bmatrix} c \\ \mathbf{A} \\ \mathbf{b}^T \end{bmatrix}$ (see (6.1.27)) for the scalar linear ODE $\dot{y} = \lambda y$, $\lambda \in \mathbb{C}$, amounts to a multiplication with the number

$$
\Psi_\lambda^\tau = 1 + z \mathbf{b}^T (\mathbf{I} - z \mathbf{A})^{-1} \mathbf{1} = \det(\mathbf{I} - z \mathbf{A} + z \mathbf{1} \mathbf{b}^T), \quad z := \lambda \tau, \quad \mathbf{1} = (1, \ldots, 1)^T \in \mathbb{R}^s.
$$

Example 8.4.27 (Stability functions of explicit RK-methods).

- Explicit Euler method (8.4.5) : $\begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$ $\Rightarrow$ $S(z) = 1 + z$.

- Explicit trapezoidal rule (8.4.6) : $\begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 0 \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{bmatrix}$ $\Rightarrow$ $S(z) = 1 + z + \frac{1}{2}z^2$.
Thm 8.4.26 together with the combinatorial formula for the determinant means that $\Psi^T_\lambda(z)$ is a polynomial of degree $\leq s$ in $z \in \mathbb{C}$.

So we conclude for the evolution of “Fourier transforms” $\hat{\mu}^{(k)}(\xi)$:

$$\hat{\mu}^{(k+1)}(\xi) = S(\tau \hat{c}(\xi)) \cdot \hat{\mu}^{(k)}(\xi), \quad k \in \mathbb{N}_0, \quad -\pi < \xi \leq \pi,$$

where $z \mapsto S(z)$ is the stability function of the Runge-Kutta timestepping method, see Thm. 8.4.26. For the explicit Euler method we recover the formula of Ex. 8.4.25.
The linear stability analysis based on Fourier symbols of difference operators for Cauchy problems is often referred to as \textit{von Neumann stability analysis}.

\textit{Remark 8.4.28 (Stability domains).}

Terminology in the theory of Runge-Kutta single step methods

\textbf{Stability domain}: \( \{ z \in \mathbb{C} : |S(z)| \leq 1 \} \).

Stability domains:
Necessary stability condition:

\[ \{ \tau \hat{c}(\xi), \ -\pi < \xi \leq \pi \} \subset \text{stability domain of RK-method} \]

Example 8.4.29 (Stability and CFL condition).

Consider: upwind spatial discretization (8.4.17) & explicit Euler timestepping
symbol of difference operator (→ Ex. 8.4.20): \( \hat{c}_h(\xi) = \frac{v}{h}(\exp(-i\xi) - 1) \),

stability function:

\[
S(z) = 1 + z.
\]

Locus of

\[
\Sigma := S(\tau \hat{c}(\xi)) , \quad -\pi < \xi \leq \pi ,
\]

in the complex plane

\[
|S(\tau \hat{c}(\xi))| \leq 1 \quad \forall \quad -\pi < \xi \leq \pi \quad \Longleftrightarrow \quad v \frac{\tau}{h} \leq 1 .
\]

= CFL-condition of Def. 8.4.13!

Note that the maximal analytic region of dependence for constant velocity \( v \) linear advection is merely a line with slope \( v \) in the \( x - t \)-plane, see Ex. 8.2.12.
Consider: upwind spatial discretization (8.4.17) & explicit trapezoidal rule: stability function $S(z) = 1 + z + \frac{1}{2}z^2$

Plots for $v = 1, \tau = 1$

$|S(\tau \hat{c}(\xi))| \leq 1 \quad \forall -\pi < \xi \leq \pi \iff \frac{\tau}{h} v \leq 1$. 

= tighter timestep constraint than stipulated by mere CFL-condition (8.4.14).
To see this note that the explicit trapezoidal rule is a 2-stage Runge-Kutta method. Hence, the spatial stencil has width 2 in upwind direction, see Fig. 284.

8.4.3 Convergence

Example 8.4.30 (Convergence of fully discrete finite volume methods for Burgers equation).

Cauchy problem for Burgers equation (8.1.16)

\[ \frac{\partial u}{\partial t} + \frac{\partial}{\partial x} \left( \frac{1}{2} u^2 \right) = 0 \quad \text{in} \; \mathbb{R} \times ]0, T[, \quad u(x, 0) = u_0(x), \quad x \in \mathbb{R}. \]
Smooth, non-smooth and discontinuous initial data, supported in $[0, 1]$:

- $u_0(x) = 1 - \cos^2(\pi x)$, $0 \leq x \leq 1$, 0 elsewhere, \quad \text{(BUMP)}
- $u_0(x) = 1 - 2*|x - \frac{1}{2}|$, $0 \leq x \leq 1$, 0 elsewhere, \quad \text{(WEDGE)}
- $u_0(x) = 1$, $0 \leq x \leq 1$, 0 elsewhere. \quad \text{(BOX)}

- Maximum speed of propagation $\dot{s} = 1$.

Spatial discretization on equidistant mesh with meshwidth $h > 0$ based on finite volume method in conservation form with

1. (local) Lax-Friedrichs numerical flux (8.3.26),
2. Godunov numerical flux (8.3.37).

Initial values $\bar{\mu}^{(0)}$ obtained from dual cell averages.
Explicit Runge-Kutta (order 4) timestepping with uniform timestep $\tau > 0$.

Fixed ratio: $\tau : h = 1 \ (\text{CFL-condition satisfied})$

Monitored: error norms (log-log plots)

\[
\text{err}_1(h) := \max_{k>0} h \sum_j |\mu_j^{(k)} - u(x_j, t_k)| \approx \max_{k>0} \left\| u_N^{(k)} - u(\cdot, t_k) \right\|_{L^1(\mathbb{R})}, \tag{8.4.31}
\]

\[
\text{err}_\infty(h) := \max_{k>0} \max_{j \in \mathbb{Z}} |\mu_j^{(k)} - u(x_j, t_k)| \approx \max_{k>0} \left\| u_N^{(k)} - u(\cdot, t_k) \right\|_{L^\infty(\mathbb{R})}. \tag{8.4.32}
\]

for different final times $T = 0.3, 4, h \in \{\frac{1}{20}, \frac{1}{40}, \frac{1}{80}, \frac{1}{160}, \frac{1}{320}, \frac{1}{640}, \frac{1}{1280}\}$.
These “exact solutions’ were computed with a MUSCL scheme (→ Sect. 8.5.3) on an equidistant mesh with $h = 10^{-4}$.

Note: for bump initial data (BUMP) we can still expect $u(\cdot, 0.3)$ to be smooth, because characteristics will not intersect before that time, cf. (8.2.14) and Ex. 8.2.15.
Why do we study the particular error norms (8.4.31) and (8.4.32)?

From Thm. 8.2.39 and Thm. 8.2.41 we know that the evolution for a scalar conservation law in 1D enjoys stability on the norms $\| \cdot \|_{L^1(\mathbb{R})}$ and $\| \cdot \|_{L^\infty(\mathbb{R})}$. Hence, these norms are the natural norms for measuring discretization errors, cf. the use of the energy norm for measuring the finite element discretization error for 2nd order elliptic BVP.

$T = 4$, error $err_1$

$T = 0.3$: error $err_\infty$
$T = 4$: error $\text{err}_{\infty}$
Error obtained by comparison with numerical “reference solution” obtained on a very fine spatio-temporal grid.

Observations: for either numerical flux function

- (near) first order algebraic convergence (→ Def. 1.6.20) w.r.t. mesh width $h$ in $\| \text{err} \|_1$,
- algebraic convergence w.r.t. mesh width $h$ in $\| \text{err} \|_\infty$ before the solution develops discontinuities (shocks),
- no convergence in norm $\| \text{err} \|_\infty$ after shock formation.

Best we get: merely first order algebraic convergence $O(h)$
Heuristic explanation for limited order:

\[ u = u(x, t) \doteq \text{smooth} \text{entropy solution of Cauchy problem} \]

\[
\frac{\partial u}{\partial t} + \frac{\partial}{\partial x} f(u) = 0 \quad \text{in } \mathbb{R} \times ]0, T[, \quad u(x, 0) = u_0(x), \quad x \in \mathbb{R}.
\tag{8.2.9}
\]

We study the so-called consistency error of the numerical flux \( F = F(v, w) \)

\[
(\vec{\tau}_F(t))_j = F(u(x_j), u(x_{j+1}, t)) - f(u(x_{j+1/2}, t)), \quad j \in \mathbb{Z},
\]

which measures the deviation of the approximate flux and the true flux, when the approximate solution agreed with the exact solution at the nodes of the mesh.

What we are interested in

behavior of \((\vec{\tau}_F(t))_j\) as mesh width \( h \to 0 \),

where an equidistant spatial mesh is assumed.

Terminology:

\[
\max_{j \in \mathbb{Z}} (\vec{\tau}_F(t))_j = O(h^q) \quad \text{for } h \to 0 \quad \leftrightarrow \quad \text{numerical flux consistent of order } q \in \mathbb{N}. \tag{8.4.34}
\]

Rule of thumb: Order of consistency of numerical flux function limits (algebraic) order of convergence of (semi-discrete and fully discrete) finite volume schemes.
Example 8.4.35 (Consistency error of upwind numerical flux).

Assumption: $f$ continuously differentiable $u_0 \geq 0$ and $f'(u) \geq 0$ for $u \geq 0$ ➞ no transsonic rarefactions!

In this case the upwind numerical flux (8.3.28) agrees with the Godunov flux (8.3.37), see Rem. 8.3.38 and

\[
F_{uw}(u(x_j, t), u(x_{j+1}, t)) = f(u(x_j), t) , \quad j \in \mathbb{Z} .
\]

\[
(\vec{r}_{F_{uw}}(t))_j = f(u(x_j, t)) - f(u(x_{j+1/2}, t))
\]
\[
= f'(u(x_{j+1/2}, t))(u(x_j, t) - u(x_{j+1/2}, t)) + O(|u(x_j, t) - u(x_{j+1/2}, t)|^2)
\]
\[
= -f'(u(x_{j+1/2}, t))\frac{\partial u}{\partial x}(x_{j+1/2}, t)\frac{1}{2}h + O(h^2) \quad \text{for } h \to 0 ,
\]

by Taylor expansion of $f$ and $u$.

This means that the upwind/Godunov numerical flux is (only) first order consistent.
Example 8.4.36 (Consistency error of Lax-Friedrichs numerical flux).

Assumption: smooth flux function

Recall: The (local) Lax-Friedrichs numerical flux

\[ F_{LF}(v, w) = \frac{1}{2}(f(v) + f(w)) - \frac{1}{2} \max_{\min\{v, w\} \leq u \leq \max\{v, w\}} |f'(u)|(w - v), \tag{8.3.26} \]

is composed of the central flux and a diffusive flux.

We examine the consistency error for both parts separately, using Taylor expansion

1. central flux:

\[
\frac{1}{2}(f(u(x_j, t)) + f(u(x_{j+1}, t))) - f(u(x_{j+1/2}, t)) \\
= \frac{1}{2}f'(u(x_{j+1/2}, t))(u(x_{j}, t) - u(x_{j+1/2}, t) + u(x_{j+1}, t) - u(x_{j+1/2}, t)) + O(h^2) \tag{8.4.37} \\
= \frac{1}{2}f'(u(x_{j+1/2}, t))\left(\frac{\partial u}{\partial x}(x_{j+1/2}, t)(-\frac{1}{2}h + \frac{1}{2}h) + O(h^2)\right) + O(h^2) \\
= O(h^2) \text{ for } h \to 0.
\]

The central flux is \textit{second order consistent}.
However, due to instability the central flux is useless, see Sect. 8.3.3.1.

Diffusive flux part:

\[ u(x_{j+1}, t) - u(x_j, t) = \frac{\partial u}{\partial x}(x_{j+1/2}, t)h + O(h^2) \quad \text{for} \quad h \to 0. \]

\[ F_{\text{LF}}(u(x_j, t), u(x_{j+1}, t)) - f(u(x_{j+1/2}, t)) = O(h) \quad \text{for} \quad h \to 0, \]

because the consistency error is dominated by the diffusive flux.

The observations made in the above examples are linked to a general fact:

Monotone numerical fluxes (→ Def. 8.3.43) are at most first order consistent.
Formally, high-order conservative finite volume methods are distinguished by numerical flux functions that are consistent of order $\geq 2$, see (8.4.34).

However, solutions of (systems of) conservation laws will usually not even be continuous (because of shocks emerging even in the case of smooth $u_0$, see (8.2.15)), let alone smooth, so that the formal order of consistency may not have any bearing for the (rate of) convergence observed for the method for a concrete Cauchy problem.

Therefore in the field of numerics of conservation laws “high-order” is desired not so much for the promise of higher rates of convergence, but for the following advantages:

- for the same spatial resolution. high-order methods frequently provide more accurate solutions in the sense of global error norms as first-order methods,
- high-order methods often provide better resolution of local features of the solution (shocks, etc.).
In standard semi-discrete finite volume schemes in conservation form for 2-point numerical flux function,

\[ \frac{d\mu_j(t)}{dt} = -\frac{1}{h}(F(\mu(t), \mu_{j+1}(t)) - F(\mu_{j-1}(t), \mu_j(t))) \quad j \in \mathbb{Z}, \quad (8.3.12) \]

the numerical flux function is evaluated for the cell averages \( \mu_j \), which can be read as approximate values of a projection of the exact solution onto piecewise constant functions (on dual cells)

\[ \mu_j(t) \approx \frac{1}{h} \int_{x_{j-1/2}}^{x_{j+1/2}} u(x, t) \, dx. \quad (8.3.5) \]

By Taylor expansion we find for \( u \in C^1 \)

\[ u(x_{j+1/2}, t) - \frac{1}{h} \int_{x_{j-1/2}}^{x_{j+1/2}} u(x, t) \, dx = O(h) \quad \text{for} \ h \to 0 , \]
and, unless some lucky cancellation occurs as in the case of the central flux, see Ex. 8.4.36, this does not allow more than first order consistency.

8.5.1 Piecewise linear reconstruction

Idea: Plug “better” approximations of $u(x_j \pm 1/2, t)$ into numerical flux function in (8.3.12)

$$\frac{d\mu_j}{dt}(t) = -\frac{1}{h} (F(\nu_j^+(t), \nu_{j+1}^-(t)) - F(\nu_{j-1}^+(t), \nu_j^-(t))) , \quad j \in \mathbb{Z} , \quad (8.5.1)$$

where $\nu_j^\pm$ are obtained by piecewise linear reconstruction from the (dual) cell values $\mu_j$. 

Fig. 308
\[\nu_j^-(t) := \mu_j(t) - \frac{1}{2} h \sigma_j(t), \quad j \in \mathbb{Z}, \quad (8.5.2)\]

with suitable slopes \(\sigma_j(t) = \sigma(\bar{\mu}(t))\).

Analogy: piecewise cubic Hermite interpolation with reconstructed slopes, discussed in the context of shape preserving interpolation in [18, Sect. 3.6.2]. However, we do not aim for smooth functions now.

**Definition 8.5.5 (Linear reconstruction).**

Given an (infinite) mesh \(\mathcal{M} := \{[x_{j-1}, x_j] \mid j \in \mathbb{Z}, (x_{j-1} < x_j)\)\), a linear reconstruction operator \(R_{\mathcal{M}}\) is a mapping

\[R_{\mathcal{M}} : \mathbb{R}^\mathbb{Z} \rightarrow \{v \in L^\infty(\mathbb{R}) : v \text{ linear on } [x_{j-1/2}, x_{j+1/2}] \forall j \in \mathbb{Z}\},\]

taking a sequence \(\bar{\mu} \in \mathbb{R}^\mathbb{Z}\) of cell averages to a possibly discontinuous function \(R_{\mathcal{M}} \bar{\mu}\) that is piecewise linear on dual cells.
Linear reconstruction & $\text{(8.5.1)}$\ IMP\ semi-discrete evolution in conservation form, cf. $\text{(8.3.11)}$

For 2-point numerical flux $F = F(u, w)$

\[
\frac{d\mu_j(t)}{dt} = -\frac{1}{h} \left( F(\nu_j^+(t), \nu_{j+1}^-(t)) - F(\nu_{j-1}^+(t), \nu_j^-(t)) \right), \quad j \in \mathbb{Z}. \tag{8.5.6}
\]

**Code 8.5.8**: Conservative FV with linear reconstruction: `ode45` timestepping

```matlab
function ufinal = highresevl(a,b,N,u0,T,F,slopes)
% finite volume discrete evolution in conservation form with linear
% reconstruction, see (8.5.6)
% Cauchy problem over time [0,T] restricted to finite interval [a,b],
% equidistant mesh with meshwidth N cells, meshwidth $h := \frac{b-a}{N}$.
% 2-point numerical flux function $F = F(v, w)$ passed in handle F
% 3-point slope reconstruction rule passed as handle slopes = @(v,u,w) ...
% (Note: no division by $h$ must be done in slope computation)
% returns cell averages for approximate solution at final time in a row vector
h = (b-a)/N; x = a+0.5*h:h:b-0.5*h; % cell centers
mu0 = h*u0(x)'; % vector of initial cell averages (column vector)
% right hand side function for MATLAB ode solvers
odefun = @(t,mu) (-1/h*fluxdiff(h,mu,F,slopes));
% timestepping by explicitity Runge-Kutta method of order 5
options = odeset('abstol',1E-8,'reltol',1E-6,'stats','on');
[t,MU] = ode45(odefun,[0 T],mu0,options);
% Graphical output
[X,T] = meshgrid(x,t);
figure; surf(X,T,MU/h); colormap(copper);
xlabel('
{\bf x}','fontsize',14);
ylabel('
{\bf t}','fontsize',14);
zlabel('
{\bf u}','fontsize',14);
ufinal = MU(end,:);'';
end

function fd = fluxdiff(h,mu,F,slopes)
% MATLAB function that realizes the right hand side operator \( L_h \) for the ODE
% (8.4.1) arising from conservative finite volume semidiscretization of the
% Cauchy problem for a 1D scalar conservation law (8.2.9).
% h: meshwidth of equidistant spatial grid
% mu: (finite) vector \( \tilde{\mu} \) of cell averages
% F: handle to 2-point numerical flux function \( F = F(v,w) \)
% slope: handle to slope function $\sigma_j = \text{slopes}(\mu_{j-1}, \mu_j, \mu_{j+1})$

n = length(mu); sigma = zeros(n,1); fd = zeros(n,1);

% Computation of slopes $\sigma_j$, uses $\mu_0 = \mu_1$,

% $m_{N+1} = \mu_N$, which amounts to constant extension of state beyond domain of
% influence $[a,b]$ of non-constant initial data.

sigma(1) = slopes(mu(1),mu(1),mu(2));

for j=2:n-1, sigma(j) = slopes(mu(j-1),mu(j),mu(j+1)); end

sigma(n) = slopes(mu(n-1),mu(n),mu(n));

% Compute linear reconstruction at endpoints of dual cells

nup = mu+0.5*sigma;  % $\nu_j^+$ at right endpoint
num = mu-0.5*sigma;  % $\nu_j^-$ at left endpoint

% Also here: constant continuation of data outside $[a,b]$!

fd(1) = F(nup(1),num(2)) - F(mu(1),num(1));

for j=2:n-1
    fd(j) = F(nup(j),num(j+1)) - F(nup(j-1),num(j)); % see (8.5.6)
end

fd(n) = F(nup(n),mu(n)) - F(nup(n-1),num(n));

end

“Natural” choice: central slope (averaged slope)

$$\sigma_j(t) = \frac{1}{2} \left( \frac{\mu_{j+1}(t) - \mu_j(t)}{h} + \frac{\mu_j(t) - \mu_{j-1}(t)}{h} \right) = \frac{1}{2} \frac{\mu_{j+1}(t) - \mu_{j-1}(t)}{h}. \quad (8.5.11)$$
By Taylor expansion: for \( u \in C^2 \) (that is, \( u \) sufficiently smooth), central slope (8.5.11), \( \nu_j^\pm \) according to (8.5.2)

\[
|\nu_j^- (t) - u(x_{j-1/2}, t)|, |\nu_j^+ (t) - u(x_{j+1/2}, t)| = O(h^2) .
\]

**Example 8.5.12** (Convergence of FV with linear reconstruction).

- Cauchy problem for Burgers equation (8.1.16) (flux function \( f(u) = \frac{1}{2}u^2 \)) from Ex. 8.2.38 with \( C^1 \) bump initial data (BUMP)
- Equidistant spatial mesh with meshwidth \( h = \)
- Linear reconstruction with central slope (8.5.11)
- Godunov numerical flux (8.3.37): \( F = F_{GD} \)
- 2n-order Runge-Kutta timestepping (method of Heun), timestep \( \tau = 0.5h \) (“CFL = 0.5”)

Monitored: Approximate \( L^1 \)- and \( L^\infty \)-norms of error at final time \( T = 0.3 \) (exact solution still smooth at this time, see Ex. 8.4.30)
Burgers equation, BUMP initial data

Fig. 310

“exact solution” computed by means of a high-order finite volume method (WENO) on an equidistant mesh with $2^{14}$ points., U. Fjordholm (SAM)

Observation: 2nd-order convergence in both norms

Example 8.5.13 (Linear reconstruction with central slope).
Cauchy problem of Ex. 8.3.19:

- Cauchy problem for Burgers equation (8.1.16) (flux function $f(u) = \frac{1}{2}u^2$) from Ex. 8.2.38 (“box” initial data)
- Equidistant spatial mesh with meshwidth $h =$
- Linear reconstruction with central slope (8.5.11)
- Godunov numerical flux (8.3.37): $F = F_{GD}$
- Timestepping based on adaptive Runge-Kutta method ode45 of MATLAB
  \[ \text{opts} = \text{odeset}('\text{abstol}',1E-7,'\text{reltol}',1E-6); \]

![Fig. 311](image.png)
Emergence of spurious oscillations in the vicinity of shock (in violation of structural properties of the exact solution, see (8.2.40).)

Compare: Oscillations occurring in FV schemes relying on central flux, see Ex. 8.3.19.
In Ex. 8.3.19, 7.2.17, the spurious oscillations can be blamed on the unstable central flux/central finite differences. Maybe, this time the central slope formula is the culprit. Thus, we investigate slope reconstruction connected with backward and forward difference quotients.

\textit{Example} 8.5.14 (Linear reconstruction with one-sided slopes).

One-sided slopes for use in (8.5.2)

Right slope: \[ \sigma_j(t) = \frac{\mu_{j+1}(t) - \mu_j(t)}{h}, \quad (8.5.15) \]

Left slope: \[ \sigma_j(t) = \frac{\mu_j(t) - \mu_{j-1}(t)}{h}. \quad (8.5.16) \]

Same setting as in Ex. 8.5.13, with central slope replaced with one-sided slopes.

Left slope:
Right slope:

Observation: spurious oscillations/overshoots, massive and global for (8.5.15), moderate close to shock for (8.5.16).
It seems to be the very process of linear reconstruction that triggers oscillations near shocks. Theses oscillations can be traced back to “overshooting” of linear reconstruction at jumps.

Slope from central differencing:

$$\sigma_j = \frac{1}{2h}(\mu_{j+1} - \mu_{j-1})$$  \hspace{1cm} (8.5.11)
Slope from forward differencing:

\[ \sigma_j = \frac{1}{h} (\mu_{j+1} - \mu_j) . \] (8.5.15)
Slope from backward differencing:

\[
\sigma_j = \frac{1}{h}(\mu_j - \mu_{j-1}). \tag{8.5.16}
\]

8.5.2 Slope limiting
Guarantee for suppression of “overshoots” (→ Figs. 314, 315, 316)

local monotonicity preservation of linear reconstruction

**Definition 8.5.18** (Monotonicity preserving linear interpolation).

An linear reconstruction operator $R_{\mathcal{M}}$ (→ Def. 8.5.5) is **monotonicity preserving**, if

\[
(R_{\mathcal{M}} \vec{\mu})(x_j) = \mu_j \quad \land \quad \begin{align*}
\mu_j &\leq \mu_{j+1} \implies R_{\mathcal{M}} \vec{\mu} \text{ non-decreasing in } [x_j, x_{j+1}], \\
\mu_j &\geq \mu_{j+1} \implies R_{\mathcal{M}} \vec{\mu} \text{ non-increasing in } [x_j, x_{j+1}].
\end{align*}
\]

Monotonicity preserving linear reconstruction:

- constant at plateaus
- constant at (local) extrema
Related: shape preserving Hermite interpolation, see [18, Sect. 3.6.2], achieved by using

- zero slope, in case of local slopes with opposite sign, see [18, (3.6.7)],
- harmonic averaging of local slopes, see [18, (3.6.9)].

**Remark 8.5.22 (Consequence of monotonicity preservation).**

A monotonicity preserving linear reconstruction operator $R_{\mathcal{M}}$ (→ Def. 8.5.18)

- respects the range of cell averages
  \[ \min\{\mu_k, \mu_{k+1}, \ldots, \mu_m\} \leq (R_{\tilde{\mu}})(x) \leq \max\{\mu_k, \mu_{k+1}, \ldots, \mu_m\}, \quad x_k < x < x_m. \quad (8.5.23) \]
  \[ \iff \text{"range preservation" by entropy solutions, see Thm. 8.2.39.} \]
- does not allow the creation of new extrema
  \[ \#\{\text{extrema of } R_{\mathcal{M}\tilde{\mu}}\} \leq \#\{\text{extrema of } \tilde{\mu}\}. \quad (8.5.24) \]
  \[ \iff \text{preservation of number of extrema in entropy solution, Sect. 8.2.7.} \]
Remark 8.5.28 (Linearity and monotonicity preservation).

The linear reconstruction operators (→ Def. 8.5.5) based on the slope formulas (8.5.11) (central slope), (8.5.15) (forward slope), (8.5.16) (backward slope) are linear in the sense that

\[ R_M(\alpha \vec{\mu} + \beta \vec{\nu}) = \alpha R_M(\vec{\mu}) + \beta R_M(\vec{\nu}) \quad \forall \vec{\mu}, \vec{\nu} \in \mathbb{R}^Z, \alpha, \beta \in \mathbb{R}. \tag{8.5.29} \]

Lemma 8.5.30 (Linear monotonicity preserving reconstruction trivial).

Every linear, monotonicity preserving (→ Def. 8.5.18) linear reconstruction yields piecewise constant functions.

Proof. Define \( \epsilon^k \in \mathbb{R}^Z, k \in \mathbb{Z} \), by

\[ \epsilon^k_j = \begin{cases} 1 & \text{for } k = j, \\ 0 & \text{else.} \end{cases} \]
The $\vec{e}_k$ form a basis of $\mathbb{R}^Z$. Thus, due to linearity, $R_M$ is fixed by its action on the basis vectors $\vec{e}_k$ and its image is spanned by $\{R_M\vec{e}_k\}_{k \in \mathbb{Z}}$.

However, monotonicity preservation entails that $R_M\vec{e}_k$ is piecewise constant, see Fig. 317.

Necessary (for monotonicity preservation): Non-linear linear reconstruction

A simple consideration, see Fig. 317

\[ \mu_{j-1} \leq \mu_j \quad \text{and} \quad \mu_j \geq \mu_{j+1} \Rightarrow R_M\vec{\mu} \equiv \text{const} \quad \text{on} \quad [x_{j-1/2}, x_{j+1/2}], \quad (8.5.32) \]

for any monotonicity preserving (→ Def. 8.5.18) linear reconstruction operator $R_M$ (→ Def. 8.5.5).

monotonicity preserving linear reconstruction $R_M\vec{\mu}$ must be constant at local extrema of $\vec{\mu}$!
Definition 8.5.33 (Minmod reconstruction). The minmod reconstruction $R_{\text{mm}}$ is a piecewise linear reconstruction (→ Def. 8.5.5) defined by

$$(R_{\text{mm}}\bar{\mu})(x) = \mu_j + \sigma_j (x - x_j)$$

for $x_{j-1/2} < x < x_{j+1/2}$, $j \in \mathbb{Z}$, $j \in \mathbb{Z}$,

$$\sigma_j := \minmod\left(\frac{\mu_{j+1} - \mu_j}{x_{j+1} - x_j}, \frac{\mu_j - \mu_{j-1}}{x_j - x_{j-1}}\right),$$

minmod$(v, w) := \begin{cases} 
  v, & vw > 0, |v| < |w|, \\
  w, & vw > 0, |w| < |v|, \\
  0, & vw \leq 0.
\end{cases}$

Lemma 8.5.34 (Monotonicity preservation of minmod reconstruction).

Minmod reconstruction (→ Def. 8.5.33) is monotonicity preserving (→ Def. 8.5.18)
**Proof.** w.l.o.g. assume $\mu_{j+1} \geq \mu_j \Rightarrow \sigma_j \geq 0 \land \sigma_{j+1} \geq 0$

$\Rightarrow \mu_j + \frac{1}{2} h \sigma_j \leq \frac{1}{2} (\mu_j + \mu_{j+1}) \leq \mu_{j+1} - \frac{1}{2} h \sigma_{j+1}$

Terminology: effect of minmod-function in $R_{mm}$: **slope limiting**: minmod = slope limiter

*Example 8.5.35 (Linear reconstruction with minmod limiter).*

Same setting as in Ex. 8.5.13. Cauchy problem as in Ex. 8.3.19:

- Cauchy problem for Burgers equation (8.1.16) (flux function $f(u) = \frac{1}{2}u^2$) from Ex. 8.2.38 (“box” intial data)
- Equidistant spatial mesh with meshwidth $h = \frac{1}{15}$
- Linear reconstruction with minmod limited slope ($\rightarrow$ Def. 8.5.33)

$$\sigma_j := \text{minmod} \left( \frac{\mu_j - \mu_{j-1}}{h}, \frac{\mu_{j+1} - \mu_j}{h} \right).$$

- Godunov numerical flux (8.3.37): $F = F_{GD}$
- timestepping based on adaptive Runge-Kutta method `ode45` of MATLAB

```matlab
opts = odeset('abstol',1E-7,'reltol',1E-6);
```
Fig. 319

R. Hiptmair
C. Schwab,
H. Harbrecht
V. Gradinaru
A. Chernov
SAM, ETHZ

Numerical Methods for PDEs

p. 968
Observation: spurious oscillations successfully suppressed!

**Example 8.5.36** (Improved resolution by limited linear reconstruction).

- Same setting as in Ex. 8.3.30: Cauchy problem for Burgers equation (8.1.16) (flux function $f(u) = \frac{1}{2}u^2$) from Ex. 8.2.38 (shifted “box” initial data, $u_0(x) = -1$ for $x \notin [0, 1]$, $u_0(x) = 1$ for $x \in [0, 1]$).
- Equidistant spatial mesh with meshwidth $h = \frac{1}{15}$.
- “High-order” method based on linear reconstruction with minmod limited slope ($\rightarrow$ Def. 8.5.33)

\[
\sigma_j := \text{minmod} \left( \frac{\mu_j - \mu_{j-1}}{h}, \frac{\mu_{j+1} - \mu_j}{h} \right).
\]

- Godunov numerical flux (8.3.37): $F = F_{GD}$
- Timestepping based on adaptive Runge-Kutta method `ode45` of MATLAB ($\text{opts} = \text{odeset}('\text{abstol}',1\times10^{-10},'\text{reltol}',1\times10^{-8});$).
Observation: *Better resolution* of rarefaction fan compared with the conservative finite volume method based on of Godunov numerical flux without linear reconstruction. Good resolution of shock.

This improved resolution is the main rationale for the use of piecewise linear reconstruction.
8.5.3 MUSCL scheme

= Monotone Upwind Scheme for Conservation Laws

Case of equidistant spatial mesh with meshwidth $h > 0$:

- Conservative finite volume spatial discretization (8.5.1) with monotone consistent 2-point flux, e.g., Godunov numerical flux (8.3.37)

- Piecewise linear reconstruction ($\rightarrow$ Def. 8.5.5) with minmod slope limiting ($\rightarrow$ Def. 8.5.33):

$$\nu_{j}^{\pm} := \mu_{j} \pm \frac{1}{2} \text{minmod} (\mu_{j+1} - \mu_{j}, \mu_{j} - \mu_{j-1}) .$$  \hspace{1cm} (8.5.39)

- 2nd-order Runge-Kutta timestepping for (8.5.1): method of Heun, cf. (8.4.6):

If the right hand side of (8.5.1) is abbreviated by

$$\mathcal{L}_{h}(\bar{\mu}) := -\frac{1}{h} (F(\nu_{j}^{+}(t), \nu_{j+1}^{-}(t)) - F(\nu_{j-1}^{+}(t), \nu_{j}^{-}(t))) ,$$
then the fully discrete scheme (uniform timestep $\tau > 0$) reads

\[
\vec{\kappa} := \vec{\mu}^{(k)} + \frac{1}{2} \tau \mathcal{L}_h(\vec{\mu}^{(k)}), \\
\vec{\mu}^{(k+1)} := \vec{\mu}^{(k)} + \tau h \mathcal{L}_h(\vec{\kappa}).
\]  

\[\text{(8.5.40)}\]

**Example 8.5.41** (Adequacy of 2nd-order timestepping).

- Same setting as in Ex. 8.3.30: Cauchy problem for Burgers equation (8.1.16) (flux function $f(u) = \frac{1}{2} u^2$) from Ex. 8.2.38 (shifted “box” initial data, $u_0(x) = -1$ for $x \notin [0, 1]$, $u_0(x) = 1$ for $x \in [0, 1]$)

- Equidistant spatial mesh with meshwidth $h = \frac{1}{15}$

- Linear reconstruction with minmod limited slope ($\rightarrow$ Def. 8.5.33)

\[
\sigma_j := \minmod\left(\frac{\mu_j - \mu_{j-1}}{h}, \frac{\mu_{j+1} - \mu_j}{h}\right).
\]

- Godunov numerical flux (8.3.37): $F = F_{GD}$
Two options for timestepping

1. Timestepping based on adaptive Runge-Kutta method `ode45` of MATLAB
   
   ```
   opts = odeset('abstol',1E-10,'reltol',1E-8);
   ```

2. Heun timestepping (8.5.40) with uniform timestep $\tau = h$

---

Burgers equation (transsonic rarefaction), $N = 60$

<table>
<thead>
<tr>
<th>Figure</th>
<th>Method</th>
<th>Timesteps</th>
<th>CFL</th>
</tr>
</thead>
<tbody>
<tr>
<td>322</td>
<td><code>ode45</code></td>
<td>157</td>
<td>1.0</td>
</tr>
<tr>
<td>323</td>
<td>Heun</td>
<td>725</td>
<td>1.0</td>
</tr>
</tbody>
</table>
Observation: 2nd-order Runge-Kutta method (8.5.40) provides same accuracy as “overkill integration” by means of `ode45` with tight tolerances.

➢ For the sake of efficiency balance order of spatial and temporal discretizations and use Heun timestepping.


Example 8.5.42 (Convergence of MUSCL scheme).

Numerical experiments of Ex. 8.4.30 repeated for

- conservative finite volume discretization with Godunov numerical flux and `minmod`-limited linear reconstruction, see Ex. 8.5.35 (`ode45` timestepping),
- MUSCL scheme as introduced above with fixed timestep $\tau = 0.5h$.

Monitored: “discrete” error norms (8.4.31), (8.4.32)
Fig. 324

Fig. 325
Observation: 2nd-order Heun method produces solutions whose convergence and accuracy matches those of solutions obtained by highly accurate high-order Runge-Kutta timestepping.
8.6 Outlook: systems of conservation laws
Finite Elements for the Stokes Equations

Supplementary and further reading:

Books (chapters) dealing with the (mathematical foundations of) discretization of the Stokes boundary value problem:

[15], Ch. 12: Concise introduction to the Stokes PDEs and Galerkin discretization. Modelling is not discussed.

[5], Ch. 12: Numerical analysis of variational saddle point problems

[3], III.§5: Concise presentation of principles of finite element discretization of the Stokes problem
9.1 Viscous fluid flow

Task: simulation of stationary fluid flow

computation of the velocity \( \mathbf{v} = \mathbf{v}(\mathbf{x}) \) of a fluid moving in a container \( \Omega \subset \mathbb{R}^d \), \( d = 2, 3 \), under the influence of an external force field \( \mathbf{f} : \Omega \mapsto \mathbb{R}^d \).

\( d = 2 ? \leftrightarrow \) translational symmetry \( \Rightarrow \) dimensionally reduced model

\( \bowtie \) notation: as before, bold typeface for vector valued functions

Recall: description of fluid motion through a velocity field \( \rightarrow \) Sect. 7.1.1

We restrict ourselves to incompressible fluids \( \rightarrow \) Def. 7.1.7

Thm. 7.1.12 \( \Rightarrow \) Constraint \( \text{div} \, \mathbf{v} = 0 \). (9.1.2)
configuration space for incompressible fluid

\[ V := \left\{ v : \overline{\Omega} \mapsto \mathbb{R}^d \text{ continuous} , \, \text{div} \, v = 0 \right\} . \tag{9.1.3} \]

Flow regimes of an incompressible **Newtonian fluid** (a fluid, for which stress is linearly proportional to strain) are distinguished by the size of a fundamental *non-dimensional* quantity, the Reynolds number

\[ \text{Re := } \frac{\rho V L}{\mu}, \]

where (for \( d = 3 \))
- \( \rho \hat{=} \text{density} \, ([\rho] = \text{kg m}^{-3}) \)
- \( V \hat{=} \text{mean velocity} \, ([V] = \text{m s}^{-1}) \)
- \( L \hat{=} \text{characteristic length of region of interest} \, ([L] = \text{m}) \)
- \( \mu \hat{=} \text{dynamic viscosity} \, ([\mu] = \text{kg m}^{-1} \text{s}^{-1}) \)

Reynolds number = ratio of *inertia forces : viscous (friction) forces*

The Reynolds number becomes small, if
• the speed of the flow is very small (slowly flowing fluids), or
• the flow is studied at tiny length scales (micro flows), or
• the fluid is highly viscous (“sticky”).

In this case acceptably accurate modelling can neglect inertia forces ➢ creeping flow

Viscous fluids “stick to the walls of the container”

\[
\text{no-slip boundary conditions: } \mathbf{v} = 0 \text{ on } \partial \Omega. \tag{9.1.5}
\]

**configuration space** for viscous incompressible fluid

\[
V := \left\{ \mathbf{v} : \overline{\Omega} \mapsto \mathbb{R}^d \text{ continuous}, \quad \text{div } \mathbf{v} = 0, \quad \mathbf{v}|_{\partial \Omega} = 0 \right\}. \tag{9.1.6}
\]
We appeal to an extremal principle to derive governing equations for incompressible creeping flow: the state of the system renders a physical quantity minimal.

For the elastic string (→ Sect. 1.2), taut membrane (→ Sect. 2.1.1), electrostatic field (→ Sect. 2.1.2) this quantity was the total potential energy. For stationary viscous fluid flow, this role is played by the energy dissipation:

\[
\text{energy dissipation} = \text{conversion of kinetic energy into internal energy (heat)} \quad (\leftrightarrow \text{entropy production})
\]

**AXIOM:** energy dissipation functional for viscous fluid \( (P_{\text{diss}}) = W \)

\[
P_{\text{diss}}(\mathbf{v}) = \int_{\Omega} \mu \| \mathbf{\text{curl}} \, \mathbf{v}(\mathbf{x}) \|^2 \, d\mathbf{x}
\]  
(9.1.7)

\[
\text{rotation/curl} \overset{\triangleq}{=} \text{first-order differential operator}
\]

\[
\mathbf{\text{curl}} \, \mathbf{v} := \begin{pmatrix}
\frac{\partial v_2}{\partial x_3} - \frac{\partial v_3}{\partial x_2} \\
\frac{\partial v_3}{\partial x_2} - \frac{\partial v_1}{\partial x_3} \\
\frac{\partial v_1}{\partial x_2} - \frac{\partial v_2}{\partial x_1}
\end{pmatrix}
\]

for \( d = 3 \), \( \mathbf{\text{curl}} \, \mathbf{v} := \frac{\partial v_1}{\partial x_2} - \frac{\partial v_2}{\partial x_1} \)

for \( d = 2 \).  
(9.1.8)
Thus, in viscous fluid flow the conversion of kinetic energy into heat due to friction presumably happens in vortical flow patterns (eddies).
Second law of thermodynamics for creeping flow:

Maximization of energy dissipation in flow

(9.1.9)

entropy production

First law of thermodynamics: conservation of energy/power balance

\[ \int_{\Omega} \mu \| \text{curl} \, \mathbf{v}(x) \|^2 \, dx = \int_{\Omega} \mathbf{f} \cdot \mathbf{v} \, dx . \]  

(9.1.10)

dissipated energy energy injected through forces

First equilibrium condition for viscous stationary flow:

\[ \mathbf{v}^* = \arg\max \ \left\{ \int_{\Omega} \mu \| \text{curl} \, \mathbf{v}(x) \|^2 \, dx : \mathbf{v} \in V , \ \mathbf{v} \text{ satisfies } (9.1.10) \right\} \]  

(9.1.14)

\[ \overset{\hat{=}}{=} \text{constrained optimization problem with constraint } (9.1.10). \]

Goal: Convert (9.1.14) into a “more standard” optimization problem.
To that end we study a related problem in finite dimensional context $\mathbb{R}^n$:

$$\mathbf{x}^* = \arg\max_{\mathbf{x}^T A \mathbf{x} = b^T \mathbf{x}} \mathbf{x}^T A \mathbf{x}, \quad (9.1.15)$$

with s.p.d. $A \in \mathbb{R}^{n,n}$, $b \in \mathbb{R}^n$. With the transformation $y = A^{-1/2}x \ (\rightarrow [18, \text{Rem. 5.3.2}])$ we arrive at the equivalent maximization problem

$$y^* = \arg\max_{\|y\|^2 = (A^{-1/2}b)^T y} \|y\|^2. \quad (9.1.15)$$

The set $\{y : \|y\|^2 = (A^{-1/2}b)^T y\}$ is a sphere through 0 around $\frac{1}{2}A^{-1/2}b$ and we are looking for its point farthest away from 0. By “geometric considerations” this will be the point $y^* = A^{-1/2}b \geq x^* = A^{-1}b$.

Recall: relationship between linear systems of equations and quadratic minimization problems, see [18, Sect. 5.1.1] and Sect. 2.1.3.
\[ \mathbf{x}^* = \mathbf{A}^{-1} \mathbf{b} \] can be obtained as solution of

\[ \mathbf{x}^* = \arg\min_{\mathbf{x} \in \mathbb{R}^n} \frac{1}{2} \mathbf{x}^T \mathbf{A} \mathbf{x} - \mathbf{b}^T \mathbf{x}. \quad (9.1.16) \]

To have faith that this reasoning applies to (9.1.14) as well, the bilinear form \((\mathbf{u}, \mathbf{v}) \mapsto \int_{\Omega} \text{curl} \, \mathbf{u} \cdot \text{curl} \, \mathbf{v} \, d\mathbf{x}\) should be positive definite (\(\mapsto\) Def. 2.1.25) \(\Rightarrow\) see Lemma 9.2.1 below.

Another issue, of course, is, whether the above arguments remain true for (infinite dimensional) function spaces \(\Rightarrow\) theory of variational calculus [30, Ch. 49], not elaborated here.

Second equilibrium condition for viscous stationary flow, cf. (2.1.4), (2.1.14):

\[ \mathbf{v}^* = \arg\min_{\mathbf{v} \in \mathbf{V}} \frac{1}{2} \int_{\Omega} \mu \|\text{curl} \, \mathbf{v}(\mathbf{x})\|^2 \, d\mathbf{x} - \int_{\Omega} \mathbf{f} \cdot \mathbf{v} \, d\mathbf{x}. \quad (9.1.17) \]
9.2 The Stokes equations

9.2.1 Constrained variational formulation

Lemma 9.2.1 \((-\Delta = \text{curl \ curl} - \text{grad \ div})\).

For \(\mathbf{v} \in C^2(\Omega), \mathbf{v}|_{\partial\Omega} = 0\), holds

\[
\int_{\Omega} \|\text{curl} \ \mathbf{v}\|^2 \, d\mathbf{x} + \int_{\Omega} |\text{div} \ \mathbf{v}|^2 \, d\mathbf{x} = \int_{\Omega} \|D\mathbf{v}\|^2_F \, d\mathbf{x}.
\]

notations:

\(D\mathbf{v} := \left( \frac{\partial v_i}{\partial x_j} \right)^d_{i,j=1} : \Omega \mapsto \mathbb{R}^{d,d}\) Jacobian,

\(\|\mathbf{M}\|_F \overset{\text{Frobenius matrix norm}}{=} \rightarrow [18, \text{Def. 6.5.24}]\)
Proof (of Lemma 9.2.1)

Use the variant of Green's first formula Thm. 2.4.7

\[ \int_\Omega \frac{\partial u}{\partial x_j} v \, dx = - \int_\Omega \frac{\partial v}{\partial x_j} u \, dx \quad \forall u, v \in C^1(\overline{\Omega}) \setminus \{0\} \quad u, v = 0 \text{ on } \partial \Omega, \tag{9.2.3} \]

and the fact that different partial derivatives can be interchanged, which implies

\[ \int_\Omega \frac{\partial u}{\partial x_j} \frac{\partial v}{\partial x_k} \, dx = \int_\Omega \frac{\partial u}{\partial x_k} \frac{\partial v}{\partial x_j} \, dx \quad k, j = 1, \ldots, d. \]

Then use the definitions of curl and div.

In light of the properties \( \text{div} \, \mathbf{v} = 0, \mathbf{v} = 0 \) on \( \partial \Omega \), for eligible fluid velocity fields, see (9.1.6), we have the equivalence:

\[ \text{Lemma 9.2.1} \quad \mathbf{v}^* = \arg \min_{\mathbf{v} \in V} \frac{1}{2} \int_\Omega \mu \|D\mathbf{v}\|_F^2 \, dx - \int_\Omega \mathbf{f} \cdot \mathbf{v} \, dx. \tag{9.2.4} \]
Referring to a quadratic minimization problem (Def. 2.1.21) on function space $V$.

Rewrite the quadratic form ($\mu \equiv \text{const}$)

$$
\mathbf{v} = (v_1, \ldots, v_d)^T: \int_{\Omega} \mu \|D\mathbf{v}\|^2_F \, d\mathbf{x} = \mu \sum_{i=1}^{d} \|\text{grad} \, v_i\|^2 \, d\mathbf{x}.
$$

By the first Poincaré-Friedrichs inequality of Thm. 2.2.16

$$
\|\mathbf{v}\|^2_{L^2(\Omega)} \leq \text{diam}(\Omega)^2 \int_{\Omega} \|D\mathbf{v}\|^2_F \, d\mathbf{x} \quad \forall \mathbf{v} \in V \subset (H^1_0(\Omega))^3.
$$

Bilinear form $a$ from (9.2.4) is positive definite (Def. 2.1.25).

**Remark 9.2.7 (Decoupling of velocity components?).**

Rewrite (9.2.4) in terms of components $v_i$ of velocity (with force field $\mathbf{f} = (f_1, f_2, f_3)^T$):

$$
(9.2.4) \Leftrightarrow \arg\min_{\mathbf{v} \in V} \sum_{i=1}^{3} \left( \frac{1}{2} \int_{\Omega} \mu \|\text{grad} \, v_i\|^2 \, d\mathbf{x} - \int_{\Omega} f_i v_i \, d\mathbf{x} \right). \quad (9.2.8)
$$
Well, three copies of (2.1.14) ?!

NO! \( \text{div} \, \mathbf{v} = 0 \) constraint (9.1.2) links components of velocity field \( \mathbf{v} \).

This constraint in the space \( V \) represents the crucial difference compared to minimization problems (2.1.4), (2.1.14) underlying scalar 2nd-order elliptic variational equations.

As in Sect. 2.2: put (9.2.4) into Hilbert space (more precisely, Sobolev space) framework, where we have existence and uniqueness of solutions.

(9.2.8) offers hint on how to choose suitable Sobolev spaces.

Remember: function spaces for a (linear) variational problem are chosen as the largest (Hilbert) spaces on which the involved bilinear forms and linear forms are still continuous, cf. (2.2.1), (3.1.2).

appropriate Sobolev space for (9.2.4):

\[
H_0^1(\text{div} \, 0, \Omega) := \left\{ \mathbf{v} \in (H_0^1(\Omega))^3 : \text{div} \, \mathbf{v} = 0 \right\}
\]
As in Sect. 2.3.1 derive the linear variational problem

$$v \in H^1_0(\text{div } 0, \Omega): \quad a(v, w) = \ell(w) \quad \forall w \in H^1_0(\text{div } 0, \Omega),$$

from (9.2.8), which reads in concrete terms:

Seek $$v \in H^1_0(\text{div } 0, \Omega) := \left\{ v \in (H^1_0(\Omega))^3: \text{div } v = 0 \right\}$$ such that

$$\int_{\Omega} \nabla v_i \cdot \nabla w_i \, dx = \int_{\Omega} f_i w_i \, dx \quad \forall w \in H^1_0(\text{div } 0, \Omega), \quad i = 1, 2, 3,$$

$$\Downarrow$$

$$\int_{\Omega} Dv : Dw \, dx = \int_{\Omega} f \cdot w \, dx \quad \forall w \in H^1_0(\text{div } 0, \Omega).$$

\(\sum\) notation: $$A : B := \sum_{i,j} a_{ij} b_{ij}$$ for matrices $$A, B \in \mathbb{R}^{m,n}$$ (“componentwise dot product”).
For this linear variational problem we verify

- Assumption 5.1.1 from Poincaré-Friedrichs inequality, see above,
- Assumption 5.1.2 for \( f \in (L^2(\Omega))^d \) by Cauchy-Schwarz inequality, see (2.2.15), (2.3.13),
- Assumption 5.1.3, since \( H^1_0(\text{div} \; 0, \Omega) \) is a closed subspace of \( H^1(\Omega) \).

Thm. 5.1.4 \( \Rightarrow \) existence & uniqueness of solutions of (9.2.9)

Remark 9.2.11 (\( H^1_0(\text{div} \; 0, \Omega) \)-conforming finite elements).

In principle, the linear variational problem could be tackled by means of a finite element Galerkin discretization.

However, finding finite element spaces \( \subset H^1_0(\text{div} \; 0, \Omega) \) is complicated [27]: Continuous, piecewise polynomial, locally supported, and divergence free basis fields exist only for polynomial degree \( \geq 4 \).
This remark motivates an approach that removes the constraint from trial and test space (and incorporates it into the variational formulation).

### 9.2.2 Saddle point problem

Idea: weak enforcement of divergence constraint (9.1.2) through Lagrange multiplier

Remark 9.2.20 (Heuristics behind Lagrangian multipliers).

Setting:

- \( U, Q \) = real Hilbert spaces with inner products \((\cdot, \cdot)_U, (\cdot, \cdot)_Q\),
- \( J : U \to \mathbb{R} \) convex and differentiable functional,
- \( B : U \to Q \) linear operator (defining constraint)
Linearly constrained minimization problem

\[ v^* = \arg\min_{v \in U, Bv=0} J(v) . \]  \hspace{1cm} (9.2.21)

Introduce Lagrangian functional:

\[ L(v, p) := J(v) + (p, Bv)_Q \quad \Rightarrow \quad v^* = \arg\min_{v \in U} \sup_{p \in Q} L(v, p) , \]  \hspace{1cm} (9.2.22)

because, if \( Bv \neq 0 \), the value of the inner supremum will be \( +\infty \), and, thus, such a \( v \) can never be a candidate for a minimizer.

Terminology: \( p \) is called a Lagrange multiplier, \( Q \) the multiplier space.

Terminology: a min-max problem like (9.2.22) = saddle point problem

**Lemma 9.2.23** (Necessary conditions for solution of saddle point problem). \( \rightarrow [30, \text{Ch. 50}] \)

Any solution \( v^* \) of (9.2.22) will be the first component of a zero \( (v^*, p^*) \) of the derivative ("gradient") of the Lagrangian functional \( L \).
\((v^*, p^*)\) will satisfy
\[
\lim_{t \to 0} \frac{L(v^* + tw, p^*) - L(v^*, p^*)}{t} = 0 \quad \forall w \in U,
\]
\[
\lim_{t \to 0} \frac{L(v^*, p^* + tq) - L(v^*, p^*)}{t} = 0 \quad \forall q \in Q.
\]

because by the very structure of the saddle point problem, see Fig. 330 for illustration,
\[
L(v^*, p) \leq L(v^*, p^*) \leq L(v, p^*) \quad \forall v \in U, p \in Q.
\]

Computing these “directional derivatives” as in Sect. 1.3.1 (for the elastic string energy functional there), we obtain
\[
\langle DJ(v^*), w \rangle + (p^*, Bw)_Q = 0 \quad \forall w \in U,
\]
\[
(q, Bv^*)_Q = 0 \quad \forall q \in Q.
\]

This is a variational saddle point problem.

Special case: \textbf{quadratic functional} \(J : U \mapsto \mathbb{R} \rightarrow \text{Def. 2.1.17}\)
\[
J(v) := \frac{1}{2}a(v, v) - \ell(v),
\]
with a\textbf{positive definite, symmetric bilinear form} \(a : U \times U \mapsto \mathbb{R}\) (\(\rightarrow\) Defs. 1.3.13, 2.1.25), continuous \textbf{linear form} \(\ell : U \mapsto \mathbb{R}\).
\[
\langle DJ(v^*), w \rangle = a(v^*, w) - \ell(w), \quad w \in U.
\]
In this special case (9.2.26) becomes a linear variational saddle point problem:

Seek \( v^* \in U, p^* \in Q \)

\[
\begin{align*}
\mathbf{a}(v^*, w) + (p^*, \mathbf{B}w)_Q &= \ell(w) \quad \forall w \in U, \\
(q, \mathbf{B}v^*)_Q &= 0 \quad \forall q \in Q.
\end{align*}
\] (9.2.27)

For rigorous mathematical treatment of constrained optimization in Banach spaces refer to [30, Ch. 49 & Ch. 50]. A discussion in finite-dimensional setting is given in [18, Sect. 7.4.1].
Solution of min-max problem:

saddle point
(non-extremal critical point)

The saddle point is a minimum when approached from the “$U$-direction”, and a maximum, when approached from the “$Q$-direction”.

Adapt abstract approach outline in Rem. 9.2.20 to (9.2.9):

- Hilbert spaces: $U = H^1_0(\Omega)$, $Q = L^2(\Omega)$,
- Constraint $\text{div } \mathbf{v} = 0 \Rightarrow B := \text{div} : U \mapsto Q$ continuous,
- $J \leftrightarrow \mathbf{v} \mapsto \frac{1}{2} \int_{\Omega} \mu \|D\mathbf{v}\|^2 \, d\mathbf{x} - \int_{\Omega} \mathbf{f} \cdot \mathbf{v} \, d\mathbf{x}$, a strictly convex quadratic functional ($\rightarrow$ Def. 2.1.17)
Lagrangian functional for (9.2.9)

\[ L(v, p) = \frac{1}{2} \int_{\Omega} \mu \|Dv\|_F^2 \, dx - \int_{\Omega} f \cdot v \, dx + \int_{\Omega} \text{div} v \, p \, dx , \quad v \in H^1_0(\Omega), \ p \in L^2(\Omega). \] (9.2.28)

Next use formula for derivative of quadratic functionals, see Sect. 2.3.1 (2.3.6), which yields a concrete specimen of (9.2.27).

Stokes problem: Linear variational saddle point problem for viscous flow (preliminary version)

seek velocity \( v \in H^1_0(\Omega) \), Lagrange multiplier \( p \in L^2(\Omega) \)

\[ \int_{\Omega} \mu Dv : Dw \, dx + \int_{\Omega} \text{div} w \, p \, dx = \int_{\Omega} f \cdot w \, dx \quad \forall w \in H^1_0(\Omega) , \]
\[ \int_{\Omega} \text{div} v \, q \, dx = 0 \quad \forall q \in L^2(\Omega) . \]

Lagrange multiplier \( p = \) pressure \( ([p] = N \text{ m}^{-2}) \)

No differential constraints in test/trial spaces for (9.2.33)!
Remark 9.2.31 (Ensuring uniqueness of pressure).

Notice:

\[ \int_{\Omega} \text{div} \mathbf{v} \, d\mathbf{x} = \int_{\partial\Omega} \mathbf{v} \cdot \mathbf{n} \, dS = 0, \text{ since } \mathbf{v}|_{\partial\Omega} = 0. \]

Pressure solution \( p \) in (9.2.33) can be unique only up to an constant!


Remedy, cf. (2.8.14)

Choose

\[ p \in L^2_*(\Omega) := \{ q \in L^2(\Omega): \int_{\Omega} q \, d\mathbf{x} = 0 \} . \tag{9.2.32} \]

\[ \text{constraint on trial/test space } L^2(\Omega) \]
Stokes problem: Variational saddle point problem for viscous flow

seek velocity \( v \in H_0^1(\Omega) \), Lagrange multiplier \( p \in L_2^*(\Omega) \)

\[
\int_\Omega \mu Dv : Dw \, dx + \int_\Omega \text{div} \, w \, p \, dx = \int_\Omega f \cdot w \, dx \quad \forall w \in H_0^1(\Omega),
\]

\[
\int_\Omega \text{div} \, v \, q \, dx = 0 \quad \forall q \in L_2^*(\Omega). \tag{9.2.33}
\]

\textbf{Theorem 9.2.34} (Existence and uniqueness of weak solutions of Stokes problem).

The linear variational saddle point problem (9.2.33) (“Stokes problem”) has a unique solution.

\textbf{Proof.} (crude outline; this sketch of the proof is included, because its ideas carry over to the discrete setting.)

Preparatory considerations: \( a(v, w) := \int_\Omega \mu Dv : Dw \, dx \) is an inner product on \( H_0^1(\Omega) \).

\( a \)-orthogonal decomposition \( H_0^1(\Omega) = H_0^1(\text{div 0, } \Omega) \oplus V^\perp \)
Unique solution $v \in H^1_0(\text{div} \ 0, \Omega)$ of (9.2.9) ➤ unique $v$-solution for (9.2.33)
(first test with $w \in H^1_0(\text{div} \ 0, \Omega)$, then with $w \in V^\perp$.)

Use the following profound result from functional analysis [3, Thm. 5.3]:

**Theorem 9.2.36** (Existence of stable velocity potentials).

$$\exists C = C(\Omega) > 0: \ \forall q \in L^2_*(\Omega): \ \exists v \in H^1_0(\Omega): \ q = \text{div} \ v \ \land \ \|v\|_{H^1(\Omega)} \leq C \|q\|_{L^2(\Omega)}.$$ 

Idea: Assume $f = 0$, test first equation with $w \in V^\perp$ satisfying $\text{div} \ w = p$ ➤ $\|p\|_{L^2(\Omega)} = 0 \iff p = 0$, for any pressure solution $p \in L^2_*(\Omega)$.

uniqueness of pressure solution

Existence of pressure solution from Riesz representation theorem (→ functional analysis) and Thm. 9.2.36, not elaborated here.
Remaining issue: (9.2.32) introduces another constraint into (9.2.33)!

Relax, Lagrangian multipliers can deal with this, too. Now we study their use to enforce a zero mean constraint in the simpler setting of 2nd-order elliptic Neumann BVPs.

*Remark 9.2.39 (Enforcing zero mean).*  \[ \rightarrow [2] \]

As in Sect. 2.4, Rem. 2.8.13, we consider a 2nd-order linear Neumann BVP (with zero Neumann boundary conditions, \( h = 0 \), cf. (2.8.15),

\[
\begin{align*}
    u \in H_1^*(\Omega) : \quad & \int_{\Omega} \kappa(x) \nabla u \cdot \nabla v \, dx = \int_{\Omega} f v \, dx \quad \forall v \in H_1^*(\Omega) .
\end{align*}
\]

with the *constrained* trial/test space

\[
H_1^*(\Omega) := \{ v \in H^1(\Omega) : \int_{\Omega} v(x) \, dx = 0 \} . \tag{2.8.14}
\]

The related quadratic minimization problem reads (\( \rightarrow \) Sect. 2.1.3)

\[
u = \arg\min_{v \in H_1^*(\Omega)} J(v) , \quad J(v) := \frac{1}{2} \int_{\Omega} \kappa(x) \| \nabla v \|^2 \, dx - \int_{\Omega} f v \, dx .
\]
Idea: enforce linear constraint $\int_{\Omega} v(\mathbf{x}) \, d\mathbf{x} = 0$ by means of Lagrangian multiplier, see Rem. 9.2.20

Here: scalar constraint ($Q = \mathbb{R}$) $\Rightarrow$ scalar multiplier $p \in \mathbb{R}$

Lagrangian functional:

$$L(v, p) = J(v) + p \int_{\Omega} v(\mathbf{x}) \, d\mathbf{x}, \quad v \in H^1(\Omega), \quad p \in \mathbb{R}.$$  

related (augmented) linear variational saddle point problem, specialization of (9.2.27):

seek $u \in H^1(\Omega), \ p \in \mathbb{R}$

$$\int_{\Omega} \kappa(\mathbf{x}) \text{grad} \, u \cdot \text{grad} \, v \, d\mathbf{x} + p \int_{\Omega} v \, d\mathbf{x} = \int_{\Omega} f v \, d\mathbf{x} \quad \forall v \in H^1(\Omega),$$

$$\int_{\Omega} v \, d\mathbf{x} = 0.$$  

(9.2.40)

The same technique can be applied to (9.2.33).

Stokes variational saddle point problem with pressure normalization:
seek velocity $v \in H^1_0(\Omega)$, pressure $p \in L^2(\Omega)$, multiplier $\lambda \in \mathbb{R}$

$$
\int_\Omega \mu \nabla v : \nabla w \, d\mathbf{x} + \int_\Omega \text{div } w \, p \, d\mathbf{x} = \int_\Omega f \cdot w \, d\mathbf{x} \quad \forall w \in H^1_0(\Omega),
$$

$$
\int_\Omega \text{div } v \, q \, d\mathbf{x} + \lambda \int_\Omega q \, d\mathbf{x} = 0 \quad \forall q \in L^2(\Omega),
$$

$$
\int_\Omega p \, d\mathbf{x} = 0.
$$

\(9.2.41\)

### 9.2.3 Stokes system

As in Sect. 2.4: derivation of the BVP in PDE form corresponding to \(9.2.41\).

**Approach:** Remove spatial derivatives from test functions by **integration by parts** \((1.3.22)\).
Assuming sufficient smoothness of solution \((v, p)\), constant \(\mu\) and (9.2.41) and taking into account boundary conditions, apply Green’s formula of Thm 2.4.7:

\[
\int_{\Omega} \mu \nabla v \cdot \nabla w \, dx = \mu \sum_{i=1}^{d} \int_{\Omega} \nabla v_i \cdot \nabla w_i \, dx = -\mu \sum_{i=1}^{d} \int_{\Omega} \Delta v_i w_i \, dx ,
\]

\[
\int_{\Omega} \text{div} \, w \, p \, dx = -\int_{\Omega} \nabla p \cdot w \, dx .
\]

(9.2.41) \Rightarrow

\[
-\mu \Delta v - \nabla p = f \quad \text{in} \quad \Omega ,
\]

\[
\text{div} \, v = 0 \quad \text{in} \quad \Omega ,
\]

\[
\int_{\Omega} p \, dx = 0
\]

\[
v = 0 \quad \text{on} \quad \partial\Omega .
\]

notation: \(\Delta \triangleq \text{componentwise Laplacian}\), see (2.4.13) ("vector Laplacian")
Remark 9.2.44 (Pressure Poisson equation).

Manipulating the PDEs in (9.2.42):

\[
\begin{align*}
\text{div} \cdot (9.2.42) & \quad \Rightarrow \quad -\mu \text{div} \Delta \mathbf{v} + \text{div} \text{grad} p = \text{div} \mathbf{f} \quad \text{in} \ \Omega, \\
\text{div} \mathbf{v} = 0 & \quad \Rightarrow \quad -\mu \Delta(\text{div} \mathbf{v}) + \Delta p = \text{div} \mathbf{f} \quad \text{in} \ \Omega, \\
\Delta p = \text{div} \mathbf{f} & \quad .
\end{align*}
\]

Appearance: (9.2.42) can be solved by solving \(d + 1\) Poisson equations,

- first solve pressure Poisson equation \(\Delta p = \text{div} \mathbf{f}\)
- then solve Dirichlet boundary value problems for velocity components

\[
-\Delta v_i = f_i + \frac{\partial p}{\partial x_i} \quad \text{in} \ \Omega, \quad v_i = 0 \quad \text{on} \ \partial \Omega.
\]

Problems

- above manipulations only valid for sufficiently smooth \(u\) (not guaranteed).
- we cannot solve a “Poisson equation”, we also need boundary conditions for \(p\): not available!
9.3 Saddle point problems: Galerkin discretization

**Example 9.3.1** (Naive finite difference discretization of Stokes system).

- **BVP** (9.2.42) on $\Omega = [0, 1]^2$, $\mu \equiv 1$, $f = \cos(\pi x_1)(0)$, $f(x) = \cos(\pi x_1)(0)$, $0 < i, j < N$.

- **Finite difference** discretization on $\to$ Sect. 4.1 equidistant tensor product grid. Unknowns: $v_{1,ij}, v_{2,ij}, p_{ij}$ = approximations of $v_1(ih, jh)$, $v_2(ih, jh)$, $p(ih, jh)$, $0 < i, j < N$.

- Zero boundary values for $v_1$, $v_2$, and $p$

- 5-point stencil discretization of $-\Delta$, see (4.1.1)

- Central finite difference approximation of $\text{grad} p$, e.g.,

$$\frac{\partial p}{\partial x_1} |_{(ih,jh)} \approx \frac{1}{2h} \left( p_{i+1,j} - p_{i-1,j} \right), \quad 1 \leq i, j < N.$$
Code 9.3.3: Central finite difference discretization of Stokes system

```matlab
function [u1,u2,p] = StokesFD(N,f)

% Naive finite difference discretization of the Stokes system (9.2.42)
% N: number of grid cells in each direction.
% f: handle to a (vector valued!) function implementing the force field f
% Return values u1, u2 give the velocity components \( v = (v_1, v_2)^T \)
% in a matrix whose entries correspond to the vertices of the mesh,
% p returns the pressure.
```

finite difference grid

force field \( \mathbf{f} \)
% mesh width
h = 1/N;

% coordinates of interior grid points
x = h:h:1-h;

% number of interior points in each direction
unk = N-1;

% total number of unknowns for \( \mathbf{v} \) and \( p \)
n = 3 * unk^2;

% A line-by-line numbering (lexikographic numbering) of the grid points is assumed,
% see Sect. 4.1, Fig. 335.

% Matrix for 5-point stencil discretization of \(-\Delta\)
A = gallery('poisson', unk);

% Build matrix representation of \( \nabla p \). Note the efficient assembly based on the special structure of the matrices.
% Auxiliary 1D central finite difference matrix
e = ones(unk, 1);
CD = spdiags([-h/2*e h/2*e], [-1 1], unk, unk);

% Central difference matrix for \( \frac{\partial}{\partial x} \): This matrix is a block diagonal matrix with \( N-1 \) diagonal blocks corresponding to the grid rows. Its diagonal blocks are skew-symmetric and bidiagonal with non-zero first off-diagonals only.
P1 = kron(speye(unk), CD);

% Central difference matrix for \( \frac{\partial}{\partial x} \): This matrix is a block matrix with non-zero first off-diagonal blocks only. Each non-zero block is a multiple of the identity.
P2 = kron(CD, speye(unk));

% Build the complete \( n \times n \) system matrix and make sure that it is a sparse matrix.
Z = sparse(unk^2, unk^2);
H = [A Z P1; Z A P2; P1' P2' Z];

% Assemble the right hand side (sampling of \( f \) at interior grid points)

R. Hiptmair
C. Schwab
H. Harbrecht
V. Gradinaru
A. Chernov
SAM, ETHZ
F = zeros(n,1);
pidx1 = 1; pidx2 = n/3+1;
for j = 1:size(x), for i = 1:size(x)
  frc = h^2*f(x(i),x(j));
  F(pidx1) = frc(1); F(pidx2) = frc(2);
  pidx1 = pidx1+1; pidx2 = pidx2 + 1;
end, end

% Direct solution of sparse indefinite symmetric system
X = H\F;

% Convert vectors of nodal values into matrix representations of grid functions
u1 = rot90(reshape(X(1:unk^2),unk,unk));
u2 = rot90(reshape(X(unk^2+1:2*unk^2),unk,unk));
p = rot90(reshape(X(2*unk^2+1:end),unk,unk));
end
Physically meaningless solution marred by massive spurious oscillations of the pressure.
9.3.1 Pressure instability

Lesson learned: discretizing saddle point problems can be tricky!

Now, we examine the *Galerkin discretization* (→ Sect. 3.1) of the linear variational problem (9.2.33) (Practical schemes will rely on (9.2.41), but here, for the sake of simplicity, we skirt the treatment of zero mean constraint.)

Shorthand notation for (9.2.33) (↔ abstract linear variational saddle point problem, see (9.2.27))

\[
\begin{align*}
v & \in U := H^1_0(\Omega), \\
p & \in Q := L^2_*(\Omega), \\
\end{align*}
\]

\[
\begin{align*}
a(v, w) + b(w, p) &= \ell(w) \quad \forall w \in U, \\
b(v, q) &= 0 \quad \forall q \in Q. \\
\end{align*}
\tag{9.3.13}
\]

with concrete *bilinear forms*

\[
\begin{align*}
a(v, w) := \int_{\Omega} \mu Dv : Dw \, dx, \\
b(v, q) := \int_{\Omega} \text{div} \ v \ q \, dx. \\
\end{align*}
\tag{9.3.14}
\]

First step of Galerkin discretization:

Replace \(H^1_0(\Omega)\) with finite dimensional *subspaces* \(U_N \subset H^1_0(\Omega)\) \(L^2_*(\Omega)\) with \(Q_N \subset L^2_*(\Omega)\)
Discrete linear variational saddle point problem:

\[ \mathbf{v}_N \in U_N : \quad a(\mathbf{v}_N, \mathbf{w}_N) + b(\mathbf{w}_N, p_N) = \ell(\mathbf{w}_N) \quad \forall \mathbf{w}_N \in U_N, \]

\[ p_N \in Q_N : \quad b(\mathbf{v}_N, q_N) = 0 \quad \forall q_N \in Q_N. \]  

(9.3.15)

Second step of Galerkin discretization:

Introduce ordered bases

\[ \mathcal{B}_U := \{b_1^N, \ldots, b_N^N\}, \quad \mathcal{B}_Q := \{\beta_1^N, \ldots, \beta_M^N\} \]

of \( U_N, \quad N := \dim U_N, \)

\( Q_N, \quad M := \dim Q_N. \)

\((N + M) \times (N + M)\) linear system of equations

\[ \begin{pmatrix} A & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} \mathbf{v} \\ \mathbf{\pi} \end{pmatrix} = \begin{pmatrix} \mathbf{\varphi} \\ 0 \end{pmatrix}, \]

(9.3.16)

(9.3.17)

with Galerkin matrices, right hand side vectors

\[ A := \left( a(b_i^j, b_i^j) \right)_{i,j=1}^N = \left( \int_{\Omega} \mu D\mathbf{b}_i^N(x) : D\mathbf{b}_j^N(x) \, dx \right)_{i,j=1}^N \in \mathbb{R}^{N,N}, \]

(9.3.18)
\[ B := \left( b^{jN}(\beta^i_N) \right)_{1 \leq i \leq M, 1 \leq j \leq N} = \left( \int_{\Omega} \text{div} \, b^{jN}(x) \beta^i_N(x) \, dx \right)_{1 \leq i \leq M, 1 \leq j \leq N} \in \mathbb{R}^{M,N}, \quad (9.3.19) \]

\[ \vec{\varphi} := \left( \ell(b^{jN}) \right)^N_{j=1} = \left( \int_{\Omega} f(x) \cdot b^{jN}(x) \, dx \right)^N_{j=1} \in \mathbb{R}^N, \quad (9.3.20) \]

and basis expansions

\[ v_N = \sum_{j=1}^{N} \nu_j b^{jN}, \quad p_N = \sum_{j=1}^{M} \pi_j \beta^j_N. \quad (9.3.21) \]

**Issue:**

existence, uniqueness and stability of solutions of \((9.3.15)\)

Existence, uniqueness and stability of solutions of discrete variational saddle point problems cannot be inferred from these properties for the continuous saddle point problem (\(\rightarrow\) Thm. 9.2.34).

(Unlike in the case of linear variational problems with s.p.d. bilinear forms, \(\text{cf.}\) Thm. 3.1.5)
A simple consideration:

\[ M > N \implies \text{Ker}(B) \neq \{0\} \implies \text{non-uniqueness of } p_N. \]

\[ \implies \dim U_N \geq \dim Q_N \text{ is a necessary condition for uniqueness of solution } p_N \text{ of (9.3.15)} \]

Some “natural” finite element Galerkin schemes for (9.2.33) ↔ (9.3.13) fail to meet this condition:

**Example 9.3.22** (Unstable P1-P0 finite element pair on triangular mesh).

Notation: \((\text{cf. } S_p^0(M))\): \(S_p^{-1}(M)\) \text{ discontinuous functions} \(\text{ locally polynomials of degree } p\), \(\text{cf. } P_p(\mathbb{R}^d)\)

The spaces \(S_p^{-1}(M)\) are the natural finite element spaces for test/trial functions \(\in L^2(\Omega)\), because this function space does not enforce any continuity conditions on piecewise smooth functions. Conversely, \(H^1(\Omega)\) does, see Thm. 2.2.17.
Regular triangular mesh of \([0, 1]^2\)

Finite element spaces for (9.2.33)

\[
U_N := (S^0_{1,0}(\mathcal{M}))^2, \\
Q_N := S^{-1}_0(\mathcal{M}) \cap L^2_*(\Omega) \quad (\mathcal{M}\text{-piecewise constants}) .
\]

\(K \in \mathbb{N} \triangleq \text{no. of mesh cells in one coordinate direction},\)

\[
\dim U_N = 2(K - 1)^2, \quad \dim Q_N = 2K^2 - 1 .
\]

\[
\dim Q_N > \dim U_N
\]

In this case we end up with a singular linear system (9.3.16), which will make the linear solver bail out or produce a pressure solution, which is polluted by “noise” from \(\text{Ker}(B)\).
But $\dim U_N \geq \dim Q_N$ is not enough: even if this condition is satisfied, the pressure may not be unique:

**Example 9.3.23** (Checkerboard instability for quadrilateral P1-P0 pair).  \(\rightarrow [3, \S 6]\)

- $\mathcal{M} =$ uniform tensor product mesh of $[0, 1]^2$
- velocity space $U_N = (\mathcal{S}_{1,0}^0(\mathcal{M}))^2$
- pressure space $Q_N = \mathcal{S}_{0}^{-1}(\mathcal{M}) \cap L^2_*(\Omega)$

If $K \in \mathbb{N}$ mesh cells in one coordinate direction,

$$\dim U_N = 2(K - 1)^2, \quad \dim Q_N = K^2 - 1.$$

$$\dim Q_N < \dim U_N \quad \text{for } K \geq 4.$$
Consider interior grid point $p = (ih, jh)$, $1 \leq i, j \leq K$, with adjacent quadratic cells $C_1, C_2, C_3, C_4$, see figure. Denote by $p_i$ the piecewise constant values of $p_N$ on $C_i$, $i = 1, 2, 3, 4$.

$b_{N,1}^p \doteq$ nodal basis function for $x_1$ velocity component at vertex $p$: $b_{N,1}^p = b_N^p \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix}$, where $b_N^p$ is the 2D “tent function” ($\rightarrow$ Fig. 87) associated with $p$.

$$\text{supp}(b_{N,1}^p) = C_1 \cup C_2 \cup C_3 \cup C_4$$

Apply Gauss’ theorem Thm. 2.4.5 on $C_i$ taking into account that $b_N^p \perp$ normals at $e_2, e_4$, and $b_N^p \parallel$
normals at $e_1, e_3$,
\[
\int_\Omega \text{div} \mathbf{b}_N^p \mathbf{p}_N \, d\mathbf{x} = p_1 \int_{e_1} \mathbf{b}_N^p \, d\mathbf{x} - p_2 \int_{e_1} \mathbf{b}_N^p \, d\mathbf{x} + p_3 \int_{e_3} \mathbf{b}_N^p \, d\mathbf{x} - p_4 \int_{e_3} \mathbf{b}_N^p \, d\mathbf{x} = \frac{1}{2}(p_1 - p_2 + p_3 - p_4) .
\]

Similarly, if $\mathbf{b}_N^p$ is the nodal basis function at $\mathbf{p}$ for the $x_2$-component of the velocity $\mathbf{v}_N$, then
\[
\int_\Omega \text{div} \mathbf{b}_N^{p,2} \mathbf{p}_N \, d\mathbf{x} = \frac{1}{2}(p_1 + p_2 - p_3 - p_4) .
\]

\[
p_1 = 1, p_2 = -1, p_3 = 1, p_4 = -1 \quad \Rightarrow \quad \int_\Omega \text{div} \mathbf{b}_N^{p,1} \mathbf{p}_N \, d\mathbf{x} = \int_\Omega \text{div} \mathbf{b}_N^{p,2} \mathbf{p}_N \, d\mathbf{x} = 0 . \quad (9.3.25)
\]

Now, realize that the setting is translation invariant!
By (9.3.25) the discrete pressure with alternating values \( \pm 1 \) in checkerboard fashion will belong to \( \text{Ker}(B) \) for this finite element Galerkin method (for odd \( K \)).

Observation:

\[ \{ p_N \in Q_N : b(v_N, p_N) = 0 \quad \forall v_N \in U_N \} \neq \emptyset. \]

= 1-dimensional space of checkerboard modes

\( \triangleright \) p.w. constant checkerboard mode

\textit{Example} 9.3.26 (P1-P0 quadrilateral finite elements for Stokes problem).
BVP (9.2.42) on $\Omega = [0, 1]^2$, $\mu \equiv 1$, $f = \cos(\pi x_1) (0)$, see Ex. 9.3.1

P1-P0 finite element Galerkin discretization on equidistant tensor product quadrilateral mesh, as in Ex. 9.3.23

---

Code 9.3.28: P1-P0 finite difference discretization of augmented Stokes problem

```matlab
function [v1,v2,p] = stokesP1P0FD(N,frc)
% P1-P0 finite element discretization of Stokes problem (9.2.41) on a
% quadrilateral tensor product mesh, see Ex. [9.3.23].
% N: number of mesh cells in one coordinate direction.
% f: function handle of type symbol64(x1,x2) to right hand side
% h = 1/N; nv = (N-1)^2; nc = N^2; % meshwidth, #V(M), #M
% Assemble system matrix from Kronecker products of 1D Galerkin matrices
% Tridiagonal 1D mass matrix for linear finite elements
M = h*spdiags(ones(N-1,3)*diag([1/6 2/3 1/6]),[-1 0 1],N-1,N-1);
% Tridiagonal 1D Galerkin matrix for $\frac{d^2}{dx^2}$, see (1.5.60)
D = spdiags(ones(N-1,3)*diag([-1 2 -1]),[-1 0 1],N-1,N-1)/h;
% 1D Galerkin matrix for p.w. linear/p.w. constant finite elements and the bilinear
% form $\int_0^1 \frac{du}{dx} q \, dx$
```

---

R. Hiptmair
C. Schwab,
H. Harbrecht
V. Gradinaru
A. Chernov
SAM, ETHZ
\[
G = \text{spdiags}(\text{ones}(N,2) \ast \text{diag}([-1 1],[1 0],N,N-1));
\]

% 1D mass matrix for p.w. linear and p.w. constant finite elements
\[
C = 0.5 \ast h \ast \text{spdiags}(\text{ones}(N,2),[-1 0],N,N-1);
\]

% constraint on pressure, see Rem. 9.2.39
\[
\Delta = \text{kron}(M,D)+\text{kron}(D,M); \quad \% 9-point stencil matrix for discrete Laplacian
\]
\[
divx = \text{kron}(C,G); \quad \text{divy} = \text{kron}(G,C); \quad \% discrete divergence
\]

% Complete saddle point system matrix including Lagrangian multiplier for enforcing mean zero
\[
A = [\Delta, \text{sparse}(nv,nv), \text{divx}', \text{sparse}(nv,1); \ldots \text{sparse}(nv,nv), \Delta, \text{divy}', \text{sparse}(nv,1); \ldots \text{divx}, \text{divy}, \text{sparse}(nc,nc), \text{ones}(nc,1); \ldots \text{sparse}(1,nv), \text{sparse}(1,nv), \text{ones}(1,nc), 0];
\]

% Assembly of right hand side
\[
\phi = \text{zeros}(2 \ast nv + nc+1,1); \quad x = h:h:1-h; \quad \text{idx} = 1;
\]
\[
\text{for } j=1:N-1, \text{ for } i=1:N-1,
\quad \phi([\text{idx idx+nv}]) = h \ast h \ast \text{frc}(x(i),x(j)); \quad \text{idx} = \text{idx}+1;
\quad \text{end, end;}
\]

% Direct solve of (singular for even N) linear saddle point system
\[
u = A \backslash \phi;
\]

% Recover velocity and pressure values on the grid
\[
v1 = \text{rot90(reshape(u(1:nv),N-1,N-1))};
v2 = \text{rot90(reshape(u(nv+1:2 \ast nv),N-1,N-1))};
p = \text{rot90(reshape(u(2 \ast nv+1:end-1),N,N))};
Observation: pressure solution marred by checkerboard mode
computed velocity field ok!
In the previous examples we found a subspace of $Q_N$, which dodges $\text{div} \ v_N$ in the bilinear form $b$.

We arrive at the important heuristic insight:

$\text{div} \ v_N$ must be “large enough to fix the pressure” $p_N \in Q_N$

Idea: Use larger velocity trial/test spaces $v_N$

- larger space $\text{div} \ v_N$
- more control of $p_N$

How to get a larger trial space for the velocity? Raise polynomial degree!
Example 9.3.29 (P2-P0 finite element scheme for the Stokes problem).

\( \Omega = [0, 1]^2 \), \( \mathbf{u}(\mathbf{x}) = (\cos(\pi/2(x_1 + x_2)), -\cos(\pi/2(x_1 + x_2)))^T \), \( p(\mathbf{x}) = \sin(\pi/2(x_1 - x_2)) \), \( \mathbf{f} \) and inhomogeneous Dirichlet boundary values for \( \mathbf{u} \) accordingly.

Sequence of (a) uniform triangular meshes, created by regular refinement, (b) randomly perturbed meshes from (a) (still uniformly shape-regular & quasi-uniform).

“P2-P0-scheme” velocity finite element space \( U_N = (S_{2,0}(\mathcal{M}))^2 \) (continuous, piecewise quadratic \( \rightarrow \) Sect. 3.4.1, Ex. 3.4.2), pressure finite element space \( Q_N = S_0^{-1}(\mathcal{M}) \cap L^2_\ast(\Omega) \) (piecewise constant).

Monitored: Error norms \( \|\mathbf{u} - \mathbf{u}_N\|_1, \|\mathbf{u} - \mathbf{u}_N\|_0, \|p - p_N\|_0 \)
% LehrFEm driver script for computing solutions of the steady Stokes problem on
% the unit square
% using piecewise quadratic finite elements for the velocity and piecewise
% constants for the
% pressure.

NREFS = 4; % Number of red refinements
NU = 1;    % Viscosity

% Dirichlet boundary data
GD_HANDLE = @(x,varargin)[cos(pi/2*(x(:,1)+x(:,2)))
                          -cos(pi/2*(x(:,1)+x(:,2)))];

% Right hand side source (force field)
F_HANDLE = @(x,varargin)[sin(pi*x(:,1)) zeros(size(x(:,1))) ];

% Initialize mesh
Mesh = load_Mesh('Coord_Sqr.dat','Elem_Sqr.dat');
Mesh.ElemFlag = ones(size(Mesh.Elements,1),1);
Mesh = add_Edges(Mesh);
Loc = get_BdEdges(Mesh);
Mesh.BdFlags = zeros(size(Mesh.Edges,1),1);
Mesh.BdFlags(Loc) = -1;
for i = 1:NREFS, Mesh = refine_REG(Mesh); end

% Assemble Galerkin matrix and load vector
A = assemMat_Stokes_P2P0(Mesh,@STIMA_Stokes_P2P0,NU,P7O6());
L = assemLoad_Stokes_P2P0(Mesh,P7O6(),F_HANDLE);

% Incorporate Dirichlet boundary data
[U,FreeDofs] = assemDir_Stokes_P2P0(Mesh,-1,GD_HANDLE); L = L - A*U;
% Solve the linear system (direct solver)
U(FreeDofs) = A(FreeDofs,FreeDofs) \backslash L(FreeDofs);

% Plot and print solution
plot_Stokes(U,Mesh,'P2P0');
title('\textbf{Steady Stokes equation (P2 elements)}');
xlabel(['\textbf{\# Dofs : ' int2str(size(U,1)) '}']);
colorbar;
print('-depsc','func_P2P0.eps')

\begin{verbatim}
function varargout = assemMat_Stokes_P2P0(Mesh,EHandle,varargin)
% Assemble Galerking matrix for P2-P0 finite element discretization of Stokes problem
% (9.2.41): piecewise quadratic continuous velocity components and piecewise constant pressure approximation.
% mesh LehrFEM mesh data structure, complete with edge information,
% Sect. 3.5.2 The struct MESH must at least contain the following fields:
% COORDINATES M-by-2 matrix specifying the vertices of the mesh.
% ELEMENTS N-by-3 or matrix specifying the elements of the mesh.
% EDGES P-by-2 matrix specifying the edges of the mesh.
% ELEMFLAG N-by-1 matrix specifying additional element information.
\end{verbatim}
% EHandle passes function for computation of element matrix.
% See Sect. 3.5.3 for a discussion of the generic assembly algorithm

nCoordinates = size(Mesh.Coordinates,1);
nElements = size(Mesh.Elements,1);
nEdges = size(Mesh.Edges,1);

% Preallocate memory for the efficient initialization of sparse matrix,
% Ex. 3.5.18
I = zeros(196*nElements,1); J = zeros(196*nElements,1); A = zeros(196*nElements,1);

% Local assembly: loop over all cells of the mesh
loc = 1:196;
for i = 1:nElements
    % Extract vertex coordinates
    vidx = Mesh.Elements(i, :);
    Vertices = Mesh.Coordinates(vidx, :);
    % Compute $14 \times 14$ element matrix: there are 6 local shape functions for the finite
    % element space $S_0(M)$, and 1 (constant) local shape function for
    % $S^{-1}(M)$: $6 + 6 + 1 = 13$ local shape functions for the P2-P0 scheme
    Aloc = EHandle(Vertices, Mesh.ElemFlag(i), varargin{:});
    % Add contributions to global Galerkin matrix: the numbering convention is as follows:
    % d.o.f. for $x_1$-components of the velocity are numbered first, then
    % $x_2$-components of the velocity, then the pressure d.o.f.
    eidx = [Mesh.Vert2Edge(vidx(1), vidx(2)) ... 
            Mesh.Vert2Edge(vidx(2), vidx(3)) ... 
            Mesh.Vert2Edge(vidx(3), vidx(1))];
    % Note: entries of an extra last row/column of the Galerkin matrix corresponding to
    % pressure d.o.f. are filled with one to enforce zero mean pressure, see
    % Ex. 9.2.39

R. Hiptmair
C. Schwab,
H. Harbrecht
V. Gradinaru
A. Chernov
SAM, ETHZ
Numerical Methods for PDEs
p. 1028
function Aloc = STIMA_Stokes_P2P0(Vertices, ElemInfo, nu, QuadRule, varargin)

% Computation of element matrix for P2-P0 finite element discretization of 2D Stokes problem
% Vertices passes the location of the vertices of the triangle
% nu is the viscosity parameter
% QuadRule specifies local quadrature rule, see Rem. 3.5.39
% The function returns a 14 × 14 dense matrix
Aloc = zeros(14,14); % Preallocate memory
% Compute element mapping
bK = Vertices(1,:); BK = [Vertices(2,:)-bK; Vertices(3,:)-bK];
inv_BK_t = transpose(inv(BK)); det_BK = abs(det(BK));

% Compute gradients element shape functions and their values at quadrature points
grad_N = grad_shap_QFE(QuadRule.x);
grad_N(:,1:2) = grad_N(:,1:2) * inv_BK_t;
grad_N(:,3:4) = grad_N(:,3:4) * inv_BK_t;
grad_N(:,5:6) = grad_N(:,5:6) * inv_BK_t;
grad_N(:,7:8) = grad_N(:,7:8) * inv_BK_t;
grad_N(:,9:10) = grad_N(:,9:10) * inv_BK_t;
grad_N(:,11:12) = grad_N(:,11:12) * inv_BK_t;

% The first 6 rows/columns of the element matrix correspond to the x\textsubscript{1}-component of the velocity. The corresponding block of the element matrix agrees with that for −Δ discretized by means of quadratic Lagrangian finite elements. The local shape functions are described in Ex. 3.4.2.

Aloc(1,1) = nu * sum(QuadRule.w. * sum(grad_N(:,1:2) .* grad_N(:,1:2), 2)) * det_BK;
Aloc(1,2) = nu * sum(QuadRule.w. * sum(grad_N(:,1:2) .* grad_N(:,3:4), 2)) * det_BK;
Aloc(1,3) = nu * sum(QuadRule.w. * sum(grad_N(:,1:2) .* grad_N(:,5:6), 2)) * det_BK;
Aloc(1,4) = nu * sum(QuadRule.w. * sum(grad_N(:,1:2) .* grad_N(:,7:8), 2)) * det_BK;
Aloc(1,5) = nu * sum(QuadRule.w. * sum(grad_N(:,1:2) .* grad_N(:,9:10), 2)) * det_BK;
Aloc(1,6) = nu * sum(QuadRule.w. * sum(grad_N(:,1:2) .* grad_N(:,11:12), 2)) * det_BK;
Aloc(2,2) = nu * sum(QuadRule.w. * sum(grad_N(:,3:4) .* grad_N(:,3:4), 2)) * det_BK;
Aloc(2,3) = nu * sum(QuadRule.w. * sum(grad_N(:,3:4) .* grad_N(:,5:6), 2)) * det_BK;
Aloc(2,4) = nu * sum(QuadRule.w. * sum(grad_N(:,3:4) .* grad_N(:,7:8), 2)) * det_BK;
Aloc(2,5) = nu * sum(QuadRule.w. * sum(grad_N(:,3:4) .* grad_N(:,9:10), 2)) * det_BK;
Aloc(2,6) = nu * sum(QuadRule.w. * sum(grad_N(:,3:4) .* grad_N(:,11:12), 2)) * det_BK;
Aloc(3,3) = nu * sum(QuadRule.w. * sum(grad_N(:,5:6) .* grad_N(:,5:6), 2)) * det_BK;
Aloc(3,4) = nu * sum(QuadRule.w. * sum(grad_N(:,5:6) .* grad_N(:,7:8), 2)) * det_BK;
Aloc(3,5) = nu * sum(QuadRule.w. * sum(grad_N(:,5:6) .* grad_N(:,9:10), 2)) * det_BK;
Aloc(3, 6) = nu * sum(QuadRule.w. * sum(\(\nabla_N(:, 5:12) * \nabla_N(:, 11:12), 2)\)) * det_BK;
Aloc(4, 4) = nu * sum(QuadRule.w. * sum(\(\nabla_N(:, 7:8) * \nabla_N(:, 7:8), 2)\)) * det_BK;
Aloc(4, 5) = nu * sum(QuadRule.w. * sum(\(\nabla_N(:, 7:8) * \nabla_N(:, 9:10), 2)\)) * det_BK;
Aloc(4, 6) = nu * sum(QuadRule.w. * sum(\(\nabla_N(:, 7:8) * \nabla_N(:, 11:12), 2)\)) * det_BK;
Aloc(5, 5) = nu * sum(QuadRule.w. * sum(\(\nabla_N(:, 9:10) * \nabla_N(:, 9:10), 2)\)) * det_BK;
Aloc(5, 6) = nu * sum(QuadRule.w. * sum(\(\nabla_N(:, 9:10) * \nabla_N(:, 11:12), 2)\)) * det_BK;
Aloc(6, 6) = nu * sum(QuadRule.w. * sum(\(\nabla_N(:, 11:12) * \nabla_N(:, 11:12), 2)\)) * det_BK;

% the same for the \(x_2\)-component of the velocity
Aloc(7, 7) = Aloc(1, 1); Aloc(7, 8) = Aloc(1, 2); Aloc(7, 9) = Aloc(1, 3);
Aloc(7, 10) = Aloc(1, 4); Aloc(7, 11) = Aloc(1, 5); Aloc(7, 12) = Aloc(1, 6);
Aloc(8, 8) = Aloc(2, 2); Aloc(8, 9) = Aloc(2, 3); Aloc(8, 10) = Aloc(2, 4);
Aloc(8, 11) = Aloc(2, 5); Aloc(8, 12) = Aloc(2, 6); Aloc(9, 9) = Aloc(3, 3);
Aloc(9, 10) = Aloc(3, 4); Aloc(9, 11) = Aloc(3, 5); Aloc(9, 12) = Aloc(3, 6);
Aloc(10, 10) = Aloc(4, 4); Aloc(10, 11) = Aloc(4, 5); Aloc(10, 12) = Aloc(4, 6);
Aloc(11, 11) = Aloc(5, 5); Aloc(11, 12) = Aloc(5, 6); Aloc(12, 12) = Aloc(6, 6);

% Interaction of pressure shape function (constant \(= 1\)) with velocity:
% evaluation of local bilinear form \(b_K\).
% First for \(x_1\)-components, then for
Aloc(1, 13) = sum(QuadRule.w. * \(\nabla_N(:, 1)\)) * det_BK;
Aloc(2, 13) = sum(QuadRule.w. * \(\nabla_N(:, 3)\)) * det_BK;
Aloc(3, 13) = sum(QuadRule.w. * \(\nabla_N(:, 5)\)) * det_BK;
Aloc(4, 13) = sum(QuadRule.w. * \(\nabla_N(:, 7)\)) * det_BK;
Aloc(5, 13) = sum(QuadRule.w. * \(\nabla_N(:, 9)\)) * det_BK;
Aloc(6, 13) = sum(QuadRule.w. * \(\nabla_N(:, 11)\)) * det_BK;
% Next for \(x_2\)-components of velocity
Aloc(7, 13) = sum(QuadRule.w. * \(\nabla_N(:, 2)\)) * det_BK;
Aloc(8, 13) = sum(QuadRule.w. * \(\nabla_N(:, 4)\)) * det_BK;
Aloc(9, 13) = sum(QuadRule.w. * \(\nabla_N(:, 6)\)) * det_BK;
Aloc(10, 13) = sum(QuadRule.w. * \(\nabla_N(:, 8)\)) * det_BK;
Aloc(11,13) = sum(QuadRule.w. * grad_N(:,10)) * det_BK;
Aloc(12,13) = sum(QuadRule.w. * grad_N(:,12)) * det_BK;
% Entry corresponding to zero mean multiplier
Aloc(13,14) = det_BK;
% Fill in lower triangular part
tri = triu(Aloc); Aloc = tri+tril(tri',-1);
return
Raising the polynomial degree has cured the instability!

Observation: algebraic convergence

\[ \| u - u_N \|_1 = O(h_M), \]
\[ \| u - u_N \|_0 = O(h_M^2), \]
\[ \| p - p_N \|_0 = O(h_M). \]

The pair \( U_N = S_{2,0}^0(M) \), \( Q_N = S_{0}^{-1}(M) \) is the first combination of finite element spaces that we find to provide a **stable** Galerkin discretization of the variational Stokes problem (9.2.33) ↔ (9.2.26).

Recall the concept of **stability/well-posedness** for linear problems, see Sect. 2.3.2, “stability estimate” of Thm. 3.1.5.

\[ \| \text{solution} \| \leq C \| \text{right hand side} \| \quad \text{for all data}, \]
where relevant norms have to be considered.

For the Stokes problem: relevant norms = norms of Sobolev spaces fitting (9.2.33)

For velocity $\mathbf{v}$: use “energy norm” $\| \cdot \|_a := a(\cdot, \cdot)^{1/2} \sim \| \cdot \|_{H^1(\Omega)}$, cf. Def. 2.1.27

For pressure $p$: use $\| \cdot \|_{L^2(\Omega)}$.

**Definition 9.3.38** (Stable finite element pair).

A pair of finite element spaces $U_N \subset H^1_0(\Omega)$, $Q_N \subset L^2_*(\Omega)$ is a **stable finite element pair**, if the solution $(\mathbf{v}_N, p_N)$ of the discrete saddle point problem (9.3.15) satisfies

$$|\ell(w_N)| \leq C_\ell \| w_N \|_a \quad \forall w_N \quad \Rightarrow \quad \exists C > 0: \quad \| \mathbf{v}_N \|_a + \| p_N \|_{L^2(\Omega)} \leq CC_\ell,$$

where $C > 0$ may depend only on $\Omega$, the coefficient $\mu$, and the **shape regularity measure** (→ Def. 5.3.26) of $\mathcal{M}$. 
We have already encountered an estimate like

$$\| \ell(w_N) \| \leq C_{\ell} \| w_N \|_a \quad \forall w_N \in U_N ,$$

(9.3.40)

when finding that the existence of solutions of quadratic minimization problems (→ Def. 2.1.21) hinges on the continuity of the involved linear form, see (2.2.1).

Let us embark on a mathematical analysis of the stability issue, which turns out to be much simpler than expected.

*Remark* 9.3.43 (Stable velocity solution).

Consider (9.2.33) ↔ (9.2.26), and Galerkin discretization (9.3.15), define the subspace

$$\mathcal{N}(b_N) := \{ w_N \in U_N : b(w_N, q_N) = 0 \quad \forall q_N \in Q_N \} \subset U_N .$$

(9.3.44)

From 2nd equation for any solution $(v_N, p_N)$ of (9.3.15): $v_N \in \mathcal{N}(b_N)$

Test the first equation of (9.3.15) with $w_N \in \mathcal{N}(b_N)$

$$a(v_N, w_N) = \ell(w_N) \quad \overset{w_N:=v_N}{\Rightarrow} \quad \| v_N \|_a^2 \leq \ell(v_N) \leq C_{\ell} \| v_N \|_a .$$

(9.3.40)
This explains the observation made in Ex. 9.3.26: reasonable approximation for velocity $v$ despite pressure instability.

**Remark 9.3.50 (Stability of pressure solution: inf-sup condition).**

Goal: stability of pressure solution $p_N \in Q_N$ of (9.3.15)

$$\|p_N\|_{L^2(\Omega)} \leq C \sup_{w_N \in U_N} \frac{\ell(w_N)}{\|w_N\|_a}$$

(9.3.51)

From the first equation of (9.3.15)

$$a(v_N, w_N) + b(w_N, p_N) = \ell(w_N) \quad \forall w_N \in U_N,$$

and the stability of the velocity solution (→ Rem. 9.3.43) we conclude (9.3.51), once we know

$$b(w_N, p_N) = g(w_N) \quad \forall w_N \in U_N \quad \Rightarrow \quad \|p_N\|_{L^2(\Omega)} \leq C \sup_{w_N \in U_N} \frac{|g(w_N)|}{\|w_N\|_a}.$$
**Theorem 9.3.53 (inf-sup condition).**

The finite element spaces $U_N \subset H^1_0(\Omega)$, $Q_N \subset L^2_*(\Omega)$ provide a stable finite element pair (\(\rightarrow\) Def. 9.3.38) for the Stokes problem (9.2.33)/(9.2.26) if there is a constant $\beta > 0$ depending only on $\Omega$ and the shape regularity measure (\(\rightarrow\) Def. 5.3.26) of $M$ such that

$$\sup_{w_N \in U_N} \frac{|b(w_N, q_N)|}{\|w_N\|_a} \geq \beta \|q_N\|_{L^2(\Omega)} \quad \forall q_N \in Q_N.$$  \hfill (9.3.54)

The estimate (9.3.54) is known as the *inf-sup condition* or the LBB (Ladyzhenskaya-Babuska-Brezzi) condition.

It is the linchpin of the numerical analysis of finite element methods for the Stokes problem, see [13].
9.3.3 Convergence

Abstract considerations (easier this way!):

- $H \doteq \text{normed vector space, norm } ||\cdot||$ (think of a function space),
- $c: H \times H \mapsto \mathbb{R}$ bilinear form on $H$, not necessarily s.p.d. ($\rightarrow \text{Def. 2.1.25}$),
- $\ell: H \mapsto \mathbb{R}$ linear form on $H$,
- Assumption: $c$ is continuous, cf. Rem. 7.2.2, (3.1.2)

\[ \exists C_c > 0: \quad |c(u, v)| \leq C_c \|u\| \|v\| \quad \forall u, v \in H. \quad (9.3.61) \]

We consider the linear variational problem ($\rightarrow \text{Rem. 1.4.5}$)

\[ u \in H: \quad c(u, v) = \ell(v) \quad \forall v \in H, \quad (9.3.62) \]
and its Galerkin discretization, based on finite-dimensional subspace $H_N \subset H$, cf. (3.1.4),

$$u_N \in H_N: \quad c(u_N, v_N) = \ell(v_N) \quad \forall v_N \in H_N.$$  \hfill (9.3.63)

Assumption: stability

$$u_N \text{ solves (9.3.63)} \implies \exists C_s > 0: \quad \|u_N\| \leq \sup_{w_N \in H_N} \frac{|\ell(w_N)|}{\|w_N\|}. \hfill (9.3.64)$$

Trick! For any $v_N \in H_N$ the difference $u_N - v_N$ ($u_N$ solution of (9.3.63)) solves

$$c(u_N - v_N, w_N) = \ell(w_N) - c(v_N, w_N) \quad \forall w_N \in H_N.$$  \hfill (9.3.62)

$$\implies \quad \|u_N - v_N\| \leq C_s \sup_{w_N \in H_N} \frac{|\ell(w_N) - c(v_N, w_N)|}{\|w_N\|} \hfill (9.3.65)$$

"Trick" Triangle inequality

$$\|u - u_N\| \leq \|u - v_N\| + \|u_N - v_N\| \leq (1 + C_c C_s) \|u - v_N\| \quad \forall v_N \in H_N.$$  \hfill (9.3.66)
(9.3.66) is a fundamental insight into the properties of Galerkin discretizations, cf. Thm. 5.1.10 that was confined to s.p.d. bilinear forms:

For the Galerkin discretization of linear variational problems:

\[ \text{Stability} \implies \text{Quasi-optimality (\ast)} \]

Terminology: Quasi-optimality of Galerkin solutions: with \( C > 0 \) independent of data and discretization parameters

\[
\| u - u_N \| \leq C \inf_{v_N \in H_N} \| u - v_N \|, \quad (9.3.67)
\]

(norm of) discretization error \quad best approximation error

Application of abstract theory to finite element discretization of Stokes problem (9.2.33):
\( H := H_0^1(\Omega) \times L^2(\Omega) \) (combination of two function spaces!)

Role of \( c \) played by

\[
c \left( \begin{pmatrix} v \\ p \end{pmatrix}, \begin{pmatrix} w \\ q \end{pmatrix} \right) := a(v, w) + b(w, p) + b(v, q) .
\] (9.3.69)

Right hand side functional "\( \ell(\begin{pmatrix} w \\ q \end{pmatrix}) = \ell(w) \)"

Galerkin trial/test space \( H_N := U_N \times Q_N \).

Then, along the lines of the above abstract considerations, one can show the following a priori error estimate:
Theorem 9.3.70 (Convergence of stable FE for Stokes problem).

If $U_N, Q_N$ is a stable finite element pair (→ Def. 9.3.38) for the Stokes problem (9.2.33), then the corresponding finite element Galerkin solution $(v_N, p_N)$ satisfies

$$
\|v - v_N\|_{H^1(\Omega)} + \|p - p_N\|_{L^2(\Omega)} \leq C \left( \inf_{w_N \in U_N} \|v - w_N\|_{H^1(\Omega)} + \inf_{q_N \in Q_N} \|p - q_N\|_{L^2(\Omega)} \right),
$$

with a constant $C > 0$ that depends only on $\Omega$, $\mu$, and the shape regularity of the finite element mesh.

Note: the a priori error bound of Thm. 9.3.70 involves the sum of the best approximation errors for both the velocity and pressure trial/test spaces.

Example 9.3.72 (Convergence of P2-P0 scheme for Stokes equation).

Interpretation of error curves observed in Ex. 9.3.29:

Smooth solutions for both $v$ and $p$: 
\begin{align*}
\inf_{w_N \in S_{2,0}^0(M)} \| v - w_N \|_{H^1(\Omega)} & \leq C h_M^2 \| v \|_{H^3(\Omega)} \quad \text{(Thm. 5.3.42)}, \\
\inf_{q_N \in S_{0}^{-1}(M)} \| p - q_N \|_{L^2(\Omega)} & \leq C h_M \| p \|_{H^1(\Omega)},
\end{align*}

with constants depending only on the shape regularity measure (\rightarrow Def. 5.3.26) of triangulation $M$.

The observed $O(h)$ algebraic convergence in the $H^1(\Omega)$-norm (for $v_N$) and $L^2(\Omega)$-norm (for $p_N$) results, because

the larger best approximation error of $S_{0}^{-1}(M)$ dominates.
The Taylor-Hood element

A: The ultimate cure for instability

chose trial/test space for velocity large enough \(\rightarrow\) very large (to play safe).

B: Well, but a large finite element space leads to a large system of linear equations, that is, high computational cost.

A: Never mind, a large space buys good accuracy, which is what we also want!

Remark 9.4.2 (Efficient finite element discretization of Stokes problem).

Thm. [9.3.70], cf. discussion in Ex. [9.3.72]: the finite element discretization error for a stable finite element pair \((U_N, Q_N)\) (\(\rightarrow\) Def. [9.3.38]) for the Stokes problem (9.2.33) is the sum of approximation errors for the velocity \(v\) in \(U_N\) and the pressure \(p\) in \(Q_N\).
Excellent approximation of either $v$ or $p$ alone may not lead to an accurate solution.

Recall similar situation for method of lines, where errors of spatial discretization and timestepping add up, see “Meta-Thms.” [6.1.63, 6.2.42].

For the sake of efficiency

$$\inf_{w_N \in U_N} \| v - w_N \|_{H^1(\Omega)} \quad \text{and} \quad \inf_{q_N \in Q_N} \| p - q_N \|_{L^2(\Omega)}$$

Too ambitious: we have no chance of guessing the best approximation errors a priori.

Thus we settle for a more modest asymptotic balance condition, cf. the considerations in Sect. [6.1.5].

Guideline for viable and efficient choice of Galerkin finite element spaces for Stokes problem:
The pair \((U_N, Q_N)\) of finite element spaces must be stable \(\rightarrow \text{Def. 9.3.38}\).

The velocity finite element space \(U_N\) should provide the same rate of algebraic convergence of the \(H^1(\Omega)\)-best approximation error w.r.t. \(h_M \rightarrow 0\) as the pressure space in \(L^2(\Omega)\).

The velocity finite element space \(U_N\) should guarantee \(\mathbf{1}\) and \(\mathbf{2}\) with as few degrees of freedom as possible.

Note that the stable finite element pair \((S_{2;0}^0(\mathcal{M}), S_{0}^{-1}(\mathcal{M}))\) does not meet the efficiency criterion, because the velocity space offers a better asymptotic rate of convergence than the pressure space, see Ex. 9.3.72.

There is a stable, perfectly balanced pair of spaces:
Taylor-Hood finite element method for Stokes problem:

- \( \mathcal{M} \): triangular/tetrahedral or rectangular/hexahedral mesh of \( \Omega \), may be hybrid, see Sect. 3.3.1

- Velocity space: \( U_N := S_0^2(\mathcal{M}) \subset \mathbf{H}_0^1(\Omega) \)

- Pressure space: \( Q_N := S_1^0(\mathcal{M}) \) (continuous pressure)
Balanced approximation properties of finite element spaces (for sufficiently smooth velocity and pressure solution):

velocity: \[
\inf_{w_N \in U_N} \| v - w_N \|_{H^1(\Omega)} \leq C h_{\mathcal{M}}^2 \| v \|_{H^3(\Omega)}
\]
by Thm. 5.3.42.

pressure: \[
\inf_{q_N \in S_0^{-1}(\mathcal{M})} \| p - q_N \|_{L^2(\Omega)} \leq C h_{\mathcal{M}}^2 \| p \|_{H^2(\Omega)}
\]
by Thm. 5.3.27.

**Theorem 9.4.3** (Stability and convergence of Taylor-Hood finite element). \[ \rightarrow [28] \]
The Taylor-Hood element provides a stable finite element pair for the Stokes problem (\[ \rightarrow \text{Def. 9.3.38} \]) and for sufficiently smooth velocity and pressure solution

\[
\| v - v_N \|_{H^1(\Omega)} + \| p - p_N \|_{L^2(\Omega)} \leq C h_{\mathcal{M}}^2 \left( \| v \|_{H^3(\Omega)} + \| p \|_{H^2(\Omega)} \right),
\]
with a constant \( C > 0 \) that depends only on \( \Omega, \mu, \) and the shape regularity of the finite element mesh.

**Example 9.4.4** (Convergence of Taylor-Hood method for Stokes problem).

- Stokes problem (9.2.41) as in Ex. 9.3.29
- perturbed triangular meshes as in Ex. 9.3.29
- Taylor-Hood finite element Galerkin discretization
Monitored: Error norms \( \| u - u_N \|_{H^1(\Omega)} \),
\( \| u - u_N \|_{L^2(\Omega)} \), \( \| p - p_N \|_{L^2(\Omega)} \)
Observation: algebraic convergence
\[
\begin{align*}
\| u - u_N \|_{H^1(\Omega)} &= O(h_M^2), \\
\| u - u_N \|_{L^2(\Omega)} &= O(h_M^3), \\
\| p - p_N \|_{L^2(\Omega)} &= O(h_M^2).
\end{align*}
\]
Adaptive Finite Element Discretization

10.1 Concept of adaptivity

10.2 A priori hp-adaptivity

10.2.1 Graded meshes in 1D

10.2.2 Triangular graded meshes

10.2.3 hp-approximation in 1D
11 Multilevel iterative solvers

11.1 Solving finite element linear systems

11.2 Subspace correction

11.2.1 Successive subspace correction algorithm (SSC)

11.2.2 Gauss-Seidel iteration

11.2.3 Hierarchical basis multigrid
12 Sparse Grids Galerkin Methods

12.1 The curse of dimension

12.2 Hierarchical basis

12.3 Sparse grids

12.4 Approximation on sparse grids

12.5 Sparse grids algorithms
Index

$H^1$-semi-norm, 147
$L^2$-norm, 144
2-regularity
  of Dirichlet problem, 580
convergence
  exponential, 156
a priori estimates, 483
a-orthogonal, 462
affine linear function
  in 2D, 287
affine transformation, 378
algebraic convergence, 156
algorithm
  numerical, 73
analytic solution, 72
angle condition
  for Delaunay mesh, 448
anisotropic diffusion, 750
artificial diffusion, 745
artificial viscosity, 745
assembly, 299
  cell oriented, 365
  in FEM, 358
  linear finite elements, 306
barycentric coordinate representation
  of local shape functions, 371
barycentric coordinates, 300
basis
  change of, 280
  best approximation error, 464
  beta function, 375
  bilinear form, 51, 66
  continuous, 717
  positive definite, 186
  bilinear transformation, 406
boundary conditions, 26, 69, 234
  Dirichlet, 234, 245
  homogeneous, 247
  Neumann, 237, 246
  no slip, 981
  radiation, 246
convective terms, 708
convergence, 140, 468
   algebraic, 156
   asymptotic, 150
convex function, 189
corner singular function, 539
Courant-Friedrichs-Levy condition (CFL), 691
Crank-Nicolson method, 621
creeping flow, 981
curve, 26, 27
   length, 28
   parametrization, 27
cut-off function, 571
d'Alembert solution, 666
Delaunay mesh
   angle condition, 448
delta distribution, 222
dielectric tensor, 178
difference quotient, 137
differential operator, 129, 167
diffusion tensor, 750
diffusive flux, 829
diffusive terms, 708
Dirac delta function, 222
Dirichlet boundary conditions, 234, 245
   for linear FE, 390
Dirichlet data, 215, 262
Dirichlet problem, 234
   variational formulation, 214
discrete maximum principle, 590
discrete model, 33
discrete variational problem, 274, 278
discretization, 73, 272
discretization error, 140, 464
discretization parameter, 468
displacement, 69
displacement function, 69
dissipation, 614
   in fluid, 984
DistMesh, 351
divergence
   of a vectorfield, 228
domain, 167
   computational, 239
   spatial, 71, 172
domain of dependence, 668
domain of influence, 668
dual mesh, 446
dual problem, 561
duality estimate, 561
dynamic estimate, 980
eddy
   in a fluid, 983
edge, 315
elastic energy, 32
   mass-spring model, 33
elastic string, 24
electric field, 177
electric scalar potential, 178
electromagnetic field energy, 178
electrostatic field energy, 178
electrostatics, 177
element, 315
element load vector, 361
element stiffness matrix, 361
equidistant mesh, 107, 137, 323
elliptic
  linear scalar second order PDE, 244
eikhoes boundary value problem, 248
energy
  conservation, 240
  of electrostatic field, 178
energy conservation
  for wave equation, 671
energy norm, 187
entropy, 982
forcing density, 175
forcing function, 827, 832
forcing term, 832
finite difference methods, 430
finite differences
  1D, 136
  in 2D, 430
finite elements
  parametric, 410
finite volume methods, 442
flow field, 701
flow map, 704
flux function, 827, 832
force density, 175
Fourier's law, 241
  if fluid, 707
Frobenius norm, 499
functional, 555
  linear, 557
fundamental lemma of calculus of variations, 232
Galerkin discretization, 76, 273
Galerkin matrix, 359
Galerkin orthogonality, 462
Galerkin solution, 78
  quasi-optimality, 464
Galerkin test space, 78
Galerkin trial space, 78
Gamma function, 375
Gauss' theorem, 229, 243, 706, 838
Gauss-Legendre quadrature, 385
Gauss-Lobatto quadrature, 385
explicit Euler method, 619
General entropy solution for 1D scalar Riemann problem, 858

generic constants, 525
global shape functions, 322
Godunov numerical flux, 896
gradient, 174
  of a function, 175
gravitational force, 30
Green's first formula, 229
grid
  1D, 107, 137
grid function, 141
h-refinement, 533
hanging node, 318
hat function, 211, 290
heat capacity, 603
heat conductivity, 242
heat equation, 603
heat flux, 239, 242
  computation of, 564
convective, 707
diffusive, 707
heat source, 241
Hessian, 495
Heun method, 914
Hilbert space, 459
homogeneous boundary conditions, 247
Hooke's law, 31
hyperbolic evolution problem, 662
  discrete case, 672
implicit Euler method, 619
implicit midpoint rule, 621
increments
  Runge-Kutta, 622
index mapping matrix, 363
inf-sup condition, 1037
inflow, 827
inflow boundary, 722
initial conditions, 600
initial value problem
  stiff, 631
initial-boundary value problems (IBVP), 601
  parabolic, 605
integrated Legendre polynomials, 88
integration by parts
  in 1D, 58
  multidimensional, 229
intermediate state, 892
interpolant
  piecewise linear, 451
interpolation error, 494
interpolation error estimates
  anisotropic, 510
  in 1D, 485
interpolation nodes, 330
inviscid, 821
kinetic energy, 671
$L(\pi)$-stability, 645
L-shaped domain, 475
L-stability, 645
simplicial, 318
triangular, 316
mesh data structure, 352
mesh file format, 346
triangular mesh, 347
mesh generation, 351
mesh generator, 346
mesh width, 471
method of characteristics, 721
method of lines, 617
midpoint rule
  composite, 114
minmod, 966
mixed boundary conditions, 247
Mixed Neumann–Dirichlet problem, 238
model
  continuous, 73
  discrete, 33; 73
monomial basis, 87
monotonicity preserving linear interpolation, 962
multi-index notation, 319
multiplicative trace inequality, 266
MUSCL scheme, 971
NETGEN, 351
Neumann boundary conditions, 237, 246
Neumann data
  admissibility conditions, 264
Neumann problem, 257
  compatibility condition, 258
  variational form, 257
Newton update, 426
Newton’s method, 422
  in function space, 422
  termination, 427
Newton’s second law of motion, 659
Newton-Cotes formula, 385
Newton-Galerkin iteration, 426
nodal basis, 290
nodal interpolation operators, 521
nodal value, 290
node, 284
  1D, 107, 137
  quadrature, 94
norm, 142
  on function space, 143
numerical domain of dependence, 917
numerical flux, 444, 869
numerical flux function, 444
numerical quadrature, 93
  nodex, 376
  weights, 376
offset function, 52, 387
  for linear FE, 388
order of quadrature rule, 377
outflow, 827
outflow boundary, 722
output functional, 555
p-refinement, 533
parametric finite elements, 410, 411
parametric quadrature rule, 376
parametrization
of curve, 27
particle method, 779
PDE
  linear scalar second order elliptic, 244
  perpendicular bisector, 447
  phase space, 826
Phythagoras’ theorem, 463
piecewise linear interpolant, 451
piecewise linear reconstruction, 948
piecewise quadratic interpolation, 522
Poincaré-Friedrichs inequality, 203, 264
point force, 57
Poisson equation, 233, 469
Poisson matrix, 434
polar coordinates, 223
polynomials
  degree, 319
  multivariate, 319
  univariate, 85
positive definite
  bilinear form, 186
  uniformly, 180
postprocessing, 141
potential energy, 671
  of taut membrane, 174
pressure, 998
pressure Poisson equation, 1006
problem parameters
  for elastic string, 30
problem size, 527
procedural form
  of functions, 271
product rule, 612
  in higher dimensions, 228
production term, 827
pullback, 396
quadratic functional, 183
quadratic local shape functions, 331
quadratic minimization problem, 64
quadratic minimization problems, 182
quadrature formula, 94
quadrature node, 94
quadrature nodes, 376
quadrature rule, 376
  on triangle, 381
  order, 377
  parametric, 376
quadrature rules
  Gauss-Legendre, 385
  Gauss-Lobatto, 385
quadrature weight, 94
quadrature weights, 376
quadrilateral mesh, 316
quasi-optimality, 464, 1040
Radau timestepping, 645
radiation boundary conditions, 246, 255
rarefaction
  subsonic, 895
  supersonic, 895
  transonic, 895
rarefaction wave/fan, 855
reaction term
  in 2nd-order BVP, 282
reference elements, 411
regular refinement, 466
reversibility, 675
Reynolds number, 980
Riemann problem, 846
Riesz representation theorem, 459
right hand side vector, 359
Robin boundary conditions, 246
rubber band, 24
Runge-Kutta
  increments, 622
Runge-Kutta method, 622
Runge-Kutta methods
  stability function, 929
saddle point problem, 994
  linear, 996, 1012
  variational, 995
SDIRK timestepping, 645
semi-discrete evolution problem, 617
semi-norm, 147
sensitivity
  of a problem, 218
shape functions
  global, 322
shape regularity measure, 502
shock, 847
  physical, 857
  subsonic, 895
  supersonic, 895
similarity solution, 853
simplicial mesh, 318
singular perturbation, 724
slope limiter, 967
slope limiting, 961
Sobolev norms, 506
Sobolev semi-norms, 508
Sobolev space $H^1(\Omega)$, 201
Sobolev space $H^1_0(\Omega)$, 199
Sobolev spaces, 193, 506
solution
  analytic, 72
  approximate, 73
source term, 167
space-time-cylinder, 600
sparsity pattern, 297
spatial domain, 172
spectrum, 923
spline
  cubic, 134
  spline collocation, 133
spring constant, 32
Störmer scheme, 680
stability, 1033
  of linear variational problem, 218
stability domain, 931
stability function
  of explicit Runge-Kutta methods, 929
  of RK-SSM, 645
Stable finite element pair, 1034
vertex, 315
virtual work principle, 50
von Neumann stability analysis, 931
Voronoi cell, 446
Voronoi dual mesh, 446
vortex, 983
wave equation, 662
weak form, 61
weak solution, 840
weight
  quadrature, 94
well-posedness, 1033
width
  of a mesh, 471
Examples and Remarks

$L^2$ interpolation error, 578
$L^2$-convergence of FE solutions, 575
$L^2$-estimates on non-convex domain, 582
$L^\infty$ interpolation error estimate in 1D, 520
$H^1_0(\text{div } 0, \Omega)$-conforming finite elements, 992
$|\cdot|_{H^1(\Omega)}$-semi-norm, 202
(Bi)-linear Lagrangian finite elements on hybrid meshes, 343

(Membrane with free boundary values, 235
ode45 for discrete parabolic evolution, 628
“PDEs” for univariate functions, 23
“Physics based” discretization, 74
1D convection-diffusion boundary value problem, 720

Adequacy of 2nd-order timestepping, 972
Admissible Dirichlet data, 262
Admissible Neumann data, 264
Affine transformation of triangles, 377
Approximate computation of norms, 149
Approximate Dirichlet boundary conditions, 389
Approximate sub-steps for Strang splitting time, 776

Approximation of mean temperature, 558, 561
Arrays storing 2D triangular mesh, 354
Assembly algorithm for linear Lagrangian finite elements, 306
Assembly for linear Lagrangian finite elements on triangular mesh, 364
Assembly for quadratic Lagrangian FEM, 367
Assembly of right hand side vector for linear finite elements, 312
Asymptotic nature of a priori estimates, 525

Barycentric representation of local shape functions, 402
Bases for polynomial spectral collocation, 130
Behavior of generalized eigenvalues of $A \tilde{\mu} = \lambda M \tilde{\mu}$, 634
Benefit of variational formulation of BVPs, 109
Bilinear Lagrangian finite elements, 336
Blow-up for leapfrog timestepping, 687
Boundary conditions and $L^2(\Omega)$, 197
Boundary conditions for 2nd-order parabolic IBVPs, 606
Boundary conditions for linear advection, 819
Boundary conditions for wave equation, 663
<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boundary conditions in $H^1_0(\Omega)$</td>
<td>200</td>
</tr>
<tr>
<td>Boundary value problems</td>
<td>167</td>
</tr>
<tr>
<td>Boundary values for conservation laws</td>
<td>831</td>
</tr>
<tr>
<td>Breakdown of characteristic solution formula</td>
<td>835</td>
</tr>
<tr>
<td>Causes for non-smoothness of solutions of elliptic BVPs</td>
<td>544</td>
</tr>
<tr>
<td>Central flux for Burgers equation</td>
<td>876</td>
</tr>
<tr>
<td>Central flux for linear advection</td>
<td>879</td>
</tr>
<tr>
<td>Characteristics for advection</td>
<td>833</td>
</tr>
<tr>
<td>Checkerboard instability for quadrilateral P1-P0 pair</td>
<td>1017</td>
</tr>
<tr>
<td>Choice of basis for polynomial spectral Galerkin methods</td>
<td>87</td>
</tr>
<tr>
<td>Choice of timestepping for m.o.l. for transient convection-diffusion</td>
<td>766</td>
</tr>
<tr>
<td>Coefficients/data in procedural form</td>
<td>75</td>
</tr>
<tr>
<td>Collocation approach on “complicated” domains</td>
<td>429</td>
</tr>
<tr>
<td>Collocation points for polynomial spectral collocation</td>
<td>130</td>
</tr>
<tr>
<td>Collocation: smoothness requirements for coefficients</td>
<td>127</td>
</tr>
<tr>
<td>Compatible boundary and initial data</td>
<td>605</td>
</tr>
<tr>
<td>Computation of heat flux</td>
<td>565, 572</td>
</tr>
<tr>
<td>Conditioning of linear variational problems</td>
<td>218</td>
</tr>
<tr>
<td>Conditioning of spectral Galerkin system matrices</td>
<td>98</td>
</tr>
<tr>
<td>Conrner singular functions</td>
<td>538</td>
</tr>
<tr>
<td>Consequence of monotonicity preservation</td>
<td>963</td>
</tr>
<tr>
<td>Consistency error of Lax-Friedrichs numerical flux</td>
<td>944</td>
</tr>
<tr>
<td>Consistency error of upwind numerical flux</td>
<td>943</td>
</tr>
<tr>
<td>Continuity of interpolation operators</td>
<td>509</td>
</tr>
<tr>
<td>Convective cooling</td>
<td>246</td>
</tr>
<tr>
<td>Convergence of Euler timestepping</td>
<td>623</td>
</tr>
<tr>
<td>Convergence of fully discrete finite volume methods for Burgers equation</td>
<td>935</td>
</tr>
<tr>
<td>Convergence of fully discrete timestepping in one spatial dimension</td>
<td>647</td>
</tr>
<tr>
<td>Convergence of FV with linear reconstruction</td>
<td>953</td>
</tr>
<tr>
<td>Convergence of Lagrangian FEM for $p$-refinement</td>
<td>480</td>
</tr>
<tr>
<td>Convergence of linear and quadratic Lagrangian finite elements in $L^2$-norm</td>
<td>473</td>
</tr>
<tr>
<td>Convergence of linear and quadratic Lagrangian finite elements in energy norm</td>
<td>469</td>
</tr>
<tr>
<td>Convergence of MUSCL scheme</td>
<td>974</td>
</tr>
<tr>
<td>Convergence of P2-P0 scheme for Stokes equation</td>
<td>1042</td>
</tr>
<tr>
<td>Convergence of SUPG and upwind quadrature FEM</td>
<td>758</td>
</tr>
<tr>
<td>Convergence of Taylor-Hood method for Stokes problem</td>
<td>1048</td>
</tr>
<tr>
<td>Crank-Nicolson timestepping</td>
<td>621</td>
</tr>
<tr>
<td>Decoupling of velocity components</td>
<td>989</td>
</tr>
<tr>
<td>Delaunay-remeshing in 2D</td>
<td>789</td>
</tr>
<tr>
<td>Derivative of non-linear $u \mapsto a(u; \cdot)$</td>
<td>424</td>
</tr>
<tr>
<td>Difference stencils</td>
<td>915</td>
</tr>
<tr>
<td>Differentiating a functional on a space of curves</td>
<td>49</td>
</tr>
<tr>
<td>Differentiating bilinear forms with time-dependent arguments</td>
<td>612</td>
</tr>
<tr>
<td>Diffusive flux</td>
<td>829</td>
</tr>
<tr>
<td>Dimensionless equations</td>
<td>29</td>
</tr>
<tr>
<td>Dimensions of Lagrangian finite element spaces on triangular meshes</td>
<td>529</td>
</tr>
<tr>
<td>Discontinuity connecting constant states</td>
<td>844</td>
</tr>
<tr>
<td>Domain of dependence/influence for 1D wave equation, constant coefficient case</td>
<td>667</td>
</tr>
<tr>
<td>Effect of added diffusion</td>
<td>746</td>
</tr>
</tbody>
</table>
Efficient finite element discretization of Stokes problem, 1044
Efficient implementation of assembly, 370
Elastic string shape by finite element discretization, 122
Elliptic lifting result in 1D, 536
Energy conservation for leapfrog, 684
Energy norm, 145
Energy norm and $H^1(\Omega)$-norm, 504
Ensuring uniqueness of pressure, 999
Entropy solution of Burgers equation, 858
Euler equations, 823
Euler timestepping, 619
Euler timestepping for 1st-order form of semi-discrete wave equation, 675
Evaluation of local shape functions at quadrature points, 401
Explicit Euler in Fourier domain, 927
Exploring convergence experimentally, 160
Extended MATLAB mesh data structure, 355
Extra regularity requirements, 62
Extra smoothness requirement for PDE formulation, 234
Finding continuous replacement functionals, 574
Finite differences for convection-diffusion equation in 1D, 726
First-order semidiscrete hyperbolic evolution problem, 674
Fourier series, 925
Fully discrete evolutions, 913
Gap between interpolation error and best approximation error, 518
General asymptotic estimates, 532
Generic constants, 524
Geometric interpretation of CFL condition in 1D, 691
Geometric obstruction to Voronoi dual meshes, 447
Godunov flux for Burgers equation, 898
Good accuracy on “bad” meshes, 515
Graph description of string shape, 69
Grid functions, 141
Heat conduction, 248
Heat conduction with radiation boundary conditions, 420
Heuristics behind Lagrangian multipliers, 993
Higher order timestepping for 1D heat equation, 650
Impact of choice of basis, 279
Impact of linear boundary fitting on FE convergence, 551
Impact of numerical quadrature on finite element discretization error, 548
Implementation of non-homogeneous Dirichlet b.c. for linear FE, 390
Implementation of spectral Galerkin discretization for elastic string problem, 101
Implementation of spectral Galerkin discretization for linear 2nd-order two-point BVP, 95
Implicit Euler method of lines for transient convection-diffusion, 763
Imposing homogeneous Dirichlet boundary conditions, 342
Improved resolution by limited linear reconstruction, 969
Inefficiency of conditionally stable single step methods, 654
Initial time, 601
Internal layers, 748
Interpolation nodes for cubic and quartic Lagrangian FE in 2D, 334
L(π)-stable Runge-Kutta single step methods, 645
Lagrangian finite elements on hybrid meshes, 345
Lagrangian method for convection-diffusion in 1D, 792
Lagrangian method for convection-diffusion in 2D, 796
Laplace operator, 233
Lax-Friedrichs flux for Burgers equation, 882
Leapfrog timestepping, 681
Linear FE discretization of 1D convection-diffusion problem, 727
Linear finite element Galerkin discretization for elastic string model, 117
Linear finite element space for homogeneous Dirichlet problem, 291
Linear reconstruction with central slope, 954
Linear reconstruction with minmod limiter, 967
Linear reconstruction with one-sided slopes, 957
Linear variational problems, 65
Linearity and monotonicity preservation, 964
Local interpolation onto higher degree Lagrangian finite element spaces, 521
Local quadrature rules on quadrilaterals, 384
Local quadrature rules on triangles, 381
Mass lumping, 683
Material coordinate, 29
Mathematical modelling, 22
Maximum principle for finite difference discretization, 590
Maximum principle for higher order Lagrangian FEM, 598
Maximum principle for linear FE for 2nd-order elliptic BVPs, 597
Mesh file format for MATLAB code, 349
Minimal regularity of membrane displacement, 176
Mixed boundary conditions, 247
Naive finite difference discretization of Stokes system, 1007
Naive finite difference scheme, 864
Non-continuity of boundary flux functional, 569
Non-differentiable function in $H^{1/2}_0(0,1)$, 210
Non-existence of solutions of positive definite quadratic minimization problem, 191
Non-homogeneous Dirichlet boundary conditions in LehrFEM, 392
Non-linear variational equation, 50
Non-polynomial “bilinear” local shape functions, 409
Non-smooth external forcing, 56
Norms on grid function spaces, 148
Numerical quadrature in LehrFEM, 383
Offset function for finite element Galerkin discretization, 115
Offset functions and Galerkin discretization, 80
offset functions for linear Lagrangian FE, 388
One-sided difference approximation of convective terms, 731
Ordered basis of test space, 82
Output functionals, 555
P1-P0 quadrilateral finite elements for Stokes problem, 1020
P2-P0 finite element scheme for the Stokes problem, 1025
Parametrization of a curve, 27
Piecewise gradient, 288
Piecewise linear functions (not) in $H^1$, 208
Piecewise quadratic interpolation, 522
Point charge, 222
Point particle method for pure advection, 780
Pressure Poisson equation, 1006
Properties of weak solutions, 840
Pullback of functions, 396
Quadratic functionals with positive definite bilinear form in 2D, 188
Quadratic minimization problem, 64
Quadratic minimization problems on Hilbert spaces, 198
Quadratic tensor product Lagrangian finite elements, 340
Quasi-locality of solution of scalar elliptic boundary value problem, 251
Radiative cooling, 247
Relationship between discrete minimization problem and discrete variational problem, 78
Scalar elliptic boundary value problem in one space dimension, 250
Scaling of entries of element matrix for $-\Delta$, 303
Semi-Lagrangian method for convection-diffusion in 1D, 807
Semi-Lagrangian method for convection-diffusion in 2D, 811
Smoothness of solution of scalar elliptic boundary value problem, 250
Smoothness requirements for collocation trial space, 127
Solution formula for sourceless transport, 768
Space of square integrable functions, 195
Sparse stiffness matrices, 296
Spatial difference operators for linear advection, 922
Spatial discretization options, 617
Spatial domains, 172
Specification of local quadrature rules, 380
Spectral Galerkin discretization of non-linear variational problem, 105
Spectral Galerkin discretization with quadrature, 93
Spectrum of elliptic operators, 639
Spectrum of upwind difference operator, 923
Spurious Galerking solution for 2D convection-diffusion BVP, 734
Stability and CFL condition, 932
Stability domains, 931
Stability functions of explicit RK-methods, 929
Stability of pressure solution: inf-sup condition, 1036
Stable velocity solution, 1035
Streamline-diffusion discretization, 753
Streamlines, 722
Supports of global shape functions in 1D, 323
Supports of global shape functions on triangular mesh, 323
Tense string without external forcing, 42
Timestepping for ODEs, 75
Transformation of basis functions, 92
Transformation techniques for bilinear transformations, 415
Triangular mesh: file format, 347
Triangular quadratic Lagrangian finite elements, 329
Uniqueness of solutions of Neumann problem, 258
Unstable P1-P0 finite element pair on triangular mesh, 1015
Upwind flux and expansion shocks, 897
Upwind flux and transonic rarefaction, 887
Upwind flux for Burgers equation, 885
Upwind quadrature discretization, 743
Vanishing viscosity for Burgers equation, 849
Variational formulation for convection-diffusion BVP, 717
Variational formulation for heat conduction with Dirichlet boundary conditions, 254
Variational formulation for pure Neumann problem, 257
Variational formulation: heat conduction with general radiation boundary conditions, 255
Virtual work principle, 50
Wave equation as first order system in time, 664
Well-posed 2nd-order linear elliptic variational problems, 460
Definitions

\( H^1 \)-semi-norm, 147
Affine transformation, 378
Characteristic curve for one-dimensional scalar conservation law, 832
Congruent matrices, 281
Consistent modifications of variational problems, 751
Consistent numerical flux function, 873
Courant-Friedrichs-Levy (CFL-)condition, 919
Cubic spline, 134
Element load vector, 361
Element stiffness matrix, 361
Energy norm, 187
Higher order Lagrangian finite element spaces, 329
Higher order Sobolev norms, 506
Higher order Sobolev semi-norms, 508
Higher order Sobolev spaces, 506
Imcompressible flow field, 709
\( L(\pi) \)-stability, 645
Lax entropy condition, 857
Legendre polynomials, 89
Linear interpolation in 2D, 494
Linear reconstruction, 949
Local shape functions, 326
Material derivative, 801
Mean square norm/\( L^2 \)-norm, 144
Mesh, 315
Mesh width, 471
Minmod reconstruction, 966
Monotone numerical flux function, 904
Monotonicity preserving linear interpolation, 962
Multivariate polynomials, 319
Norm, 142
Numerical domain of dependence, 917
Parametric finite elements, 411
Positive definite bilinear form, 186
Pullback, 396
Quadratic functional, 183
Quadratic minimization problem, 184
Riemann problem, 846
Runge-Kutta method, 622
Shape regularity measure for simplex, 502
Shock, 847
Singularly perturbed problem, 724
Sobolev space $H^1(\Omega)$, 201
Sobolev space $H^1_0(\Omega)$, 199
Space $L^2(\Omega)$, 196
Sparse matrix, 296
Stable finite element pair, 1034
Support of a function, 111
Supremum norm, 143
Tensor product Lagrangian finite element spaces, 340
tensor product polynomials, 321
Uniformly positive definite tensor field, 180
Weak solution of Cauchy problem for conservation law, 840
MATLAB codes

assemMat_QFE, 368
sparse (MATLAB-function), 368
add_Edge2Elem, 355
add_Edges, 355
init_Mesh, 355
Loading a mesh from file, 350
List of Symbols

$C^2_0([0, 1]) := \{ v \in C^2([0, 1]) : v(0) = v(1) = 0 \}$, 46

$C^\infty_0(\Omega) \triangleq$ smooth functions with support inside $\Omega$, 204

$C^k([a, b]) \triangleq k$-times continuously differentiable functions on $[a, b] \subset \mathbb{R}$, 27

$C^k_{pw}([a, b])$, 56

$D^{-}(\vec{x}, \vec{t}) \triangleq$ maximal analytical domain of dependence of $(\vec{x}, \vec{t})$, 918

$D^\alpha u \triangleq$ multiple partial derivatives, 506

$L^2_*(\Omega) := \{ q \in L^2(\Omega) : \int_{\Omega} q \, dx = 0 \}$, 999

$O(f(N)) \triangleq$ Landau-$O$ for $N \to \infty$, 156

$S(z) \triangleq$ stability function of Runge-Kutta method, 929

$n, 245$

$n \triangleq$ exterior unit normal vectorfield, 229

$\mathcal{H}_h \triangleq$ fully discrete evolution operator, 913

$L_h \triangleq$ semi-discrete evolution operator doe 1D conservation law, 911

$\mathcal{P}_p(\mathbb{R}) \triangleq$ space of univariate polynomials of degree $\leq p$, 85

$\mathcal{P}_p(\mathbb{R}^d)$, 319

$\mathcal{P}_p(\mathbb{R}^d)$ \triangleq space of $d$-variate polynomials, 319

$Q_p(\mathbb{R}^d)$, 321

$\mathcal{V}(\mathcal{M}) \triangleq$ set of vertices of a mesh, 284

$\Delta \triangleq$ Laplace operator, 233

$\Delta \triangleq$ vector Laplacian, 1005

$\text{div} \, j \triangleq$ divergence of a vectorfield, 228

$\mathcal{E}(\mathcal{M})$, 356

$I_1$, 494

$\Gamma_{in} \triangleq$ inflow boundary for advection BVP, 722

$H^m(\Omega) \triangleq m$-th order Sobolev space, 506

$H^1_0(\Omega) \triangleq$ componentwise $H^1(\Omega)$-vectorfields with vanishing divergence., 991

$S^i_0(\mathcal{M})$, 286

$I_1 \triangleq$ piecewise linear interpolation on finite element mesh, 451

$P_n \triangleq n$-th Legendre polynomial, 39

$S^i_p(\mathcal{M}) \triangleq H^1(\Omega)$-conforming Lagrangian FE space, 329

$L^\infty(\Omega) \triangleq$ space of (essentially) bounded functions on $\Omega$, 143

$L^2(\Omega) \triangleq$ space of square-integrable functions on $\Omega$, 196
Bibliography


