P. Grohs
Ž. Kereta
C. Schillings

Numerical Analysis II

ETH Zürich
D-MATH

Exam Summer 2014

Problem 1 A Parameter-Dependent Method

Consider the following family of single-step methods, depending on the real parameter α,

$$\mathbf{\Psi}^h \mathbf{y}_0 := \mathbf{y}_0 + h\Big(1 - \frac{\alpha}{2}\Big)\mathbf{f}(0, \mathbf{y}_0) + h\frac{\alpha}{2}\mathbf{f}(h, \mathbf{\Psi}^h \mathbf{y}_0). \tag{1.1}$$

(1a) Write down the Butcher scheme of (**??**). For which value of α is (**??**) consistent?

Solution:

$$\begin{array}{c|cc} 0 & 0 & 0 \\ 1 & 1\text{-}\alpha/2 & \alpha/2 \\ \hline & 1\text{-}\alpha/2 & \alpha/2 \end{array}$$

The method is always consistent, since $\sum b_i = 1$.

(1b) Investigate the maximal order of convergence of (**??**) with respect to α.

Solution: The method has order of convergence of at least one due to **??**. It has order two if $\sum b_i c_i = 0.5$, i.e. if $\alpha = 1$. It does not attain order three, since for $\alpha = 1$ we see that $\sum b_i c_i^2 \neq \frac{1}{3}$.

(1c) Deduce the stability function of (**??**). For which value of α is the method A-stable? What about L-stability?

Solution:

$$S(z) = \frac{1 + z(1 - \alpha/2)}{1 - z\alpha/2}.$$

Let $z = x + \mathrm{i}y$, then

$$\begin{aligned} |S(z)| < 1 &\iff |S(z)|^2 < 1 \\ &\iff \Big|1 + z\Big(1 - \frac{\alpha}{2}\Big)\Big|^2 < \Big|1 - \frac{z\alpha}{2}\Big|^2 \\ &\iff \Big(1 + x\Big(1 - \frac{\alpha}{2}\Big)\Big)^2 + \Big(1 - \frac{\alpha}{2}\Big)^2 y^2 < \Big(1 - \frac{x\alpha}{2}\Big)^2 + \frac{\alpha^2}{4}y^2 \\ &\iff (1 - \alpha)x^2 + 2x + (1 - \alpha)y^2 < 0. \end{aligned}$$

A-stability is only satisfied if $1 - \alpha < 0$, thus if $\alpha > 1$. For L-stability, we need that also $1 - \alpha/2 = 0$, thus $\alpha = 2$ (which implies that the numerator of $S(z)$ is a polynomial of order zero).

(1d) Write the function

```
y1 = parameterstep(t0,y0,h,alpha,f,Df),
```

which executes one step of the method (**??**). The inputs `f` and `Df` are function-handles and correspond to the right-hand side $\mathbf{f}(t,\mathbf{y})$ and its derivative $\mathbf{D_y f}(t,\mathbf{y})$. Since the method can be implicit, equation (**??**) should be solved with one iteration of Newton's method. Use `y0` as the initial value for the Newton iteration.

Solution:

Listing 1.1: Solution of **??**

```
function y1 = parameterstep(t0,y0,h,alpha,f,Df)

% Function for root finding
F=@(y) y - y0 - h*(1-alpha/2)*f(t0,y0)-h*alpha/2*f(t0+h,y);

% Its derivative
DF = @(y) eye(length(y))-h*alpha/2*Df(t0+h,y);

% Newton step
y1 = y0 - DF(y0)\F(y0);

end
```

Listing 1.2: Testcalls for **??**

```
MISSING!
```

Listing 1.3: Output for Testcalls for **??**

```
MISSING!
```

Problem 2 Composition Methods

Let $\mathbf{\Psi}^h$ be the discrete evolution operator of a consistent single-step method for an autonomous IVP

$$\dot{\mathbf{y}} = \mathbf{f}(\mathbf{y}), \qquad \mathbf{y}_0 = \mathbf{y}(0).$$

The discrete evolution $\tilde{\mathbf{\Psi}}^h$ of the composition method $\mathbf{\Psi}^h$ with real step-sizes $\gamma_1 h$, $\gamma_2 h$ and $\gamma_3 h$ is defined by

$$\tilde{\mathbf{\Psi}}^h := \mathbf{\Psi}^{\gamma_3 h} \circ \mathbf{\Psi}^{\gamma_2 h} \circ \mathbf{\Psi}^{\gamma_1 h}. \tag{2.1}$$

(2a) Prove that $\tilde{\mathbf{\Psi}}^h$ is consistent of order $p+1$, provided $\mathbf{\Psi}^h$ is consistent of order p and p is even,

$$\gamma_1 + \gamma_2 + \gamma_3 = 1, \quad \text{und} \quad \gamma_1^{p+1} + \gamma_2^{p+1} + \gamma_3^{p+1} = 0. \tag{2.2}$$

HINT: The order p must be even, because otherwise (**??**) would have no real roots.

Solution: It holds that (for $\gamma_1 + \gamma_2 + \gamma_3 = 1$)

$$\begin{aligned}
\mathbf{\Phi}^h \mathbf{y}_0 - \tilde{\mathbf{\Psi}}^h \mathbf{y}_0 &= \mathbf{\Phi}^{\gamma_3 h} \mathbf{\Phi}^{\gamma_2 h + \gamma_1 h} \mathbf{y}_0 - \mathbf{\Psi}^{\gamma_3 h} \mathbf{\Phi}^{\gamma_2 h + \gamma_1 h} \mathbf{y}_0 \\
&\quad + \mathbf{\Psi}^{\gamma_3 h} \mathbf{\Phi}^{\gamma_2 h + \gamma_1 h} \mathbf{y}_0 - \mathbf{\Psi}^{\gamma_3 h} \mathbf{\Psi}^{\gamma_2 h} \mathbf{\Psi}^{\gamma_1 h} \mathbf{y}_0 \\
&\le C\big(\mathbf{\Phi}^{\gamma_2 h + \gamma_1 h} \mathbf{y}_0\big) \gamma_3^{p+1} h^{p+1} \\
&\quad + (\mathbf{I} + \mathcal{O}(h))(\mathbf{\Phi}^{\gamma_2 h + \gamma_1 h} \mathbf{y}_0 - \mathbf{\Psi}^{\gamma_2 h} \mathbf{\Psi}^{\gamma_1 h} \mathbf{y}_0) + \mathcal{O}(h^{p+2}) \\
&\le C\big(\mathbf{\Phi}^{\gamma_2 h + \gamma_1 h} \mathbf{y}_0\big) \gamma_3^{p+1} h^{p+1} + C\big(\mathbf{\Phi}^{\gamma_1 h} \mathbf{y}_0\big) \gamma_2^{p+1} h^{p+1} \\
&\quad + (\mathbf{I} + \mathcal{O}(h))(\mathbf{\Phi}^{\gamma_1 h} \mathbf{y}_0 - \mathbf{\Psi}^{\gamma_1 h} \mathbf{y}_0) + \mathcal{O}(h^{p+2}) \\
&\le C\big(\mathbf{\Phi}^{\gamma_2 h + \gamma_1 h} \mathbf{y}_0\big) \gamma_3^{p+1} h^{p+1} + C\big(\mathbf{\Phi}^{\gamma_1 h} \mathbf{y}_0\big) \gamma_2^{p+1} h^{p+1} \\
&\quad + C(\mathbf{y}_0) \gamma_1^{p+1} h^{p+1} + \mathcal{O}(h^{p+2}).
\end{aligned}$$

From $\mathbf{\Phi}^{\gamma_i h} \mathbf{y}_0 = \mathbf{y}_0 + \mathcal{O}(h)$ we can deduce $C\big(\mathbf{\Phi}^{\gamma_i h} \mathbf{y}_0\big) = C(\mathbf{y}_0) + \mathcal{O}(h)$. The condition $\gamma_1^{p+1} + \gamma_2^{p+1} + \gamma_3^{p+1} = 0$ then causes the term h^{p+1} to vanish.

(2b) Let $\mathbf{\Psi}^h$ be reversible (i.e., $\mathbf{\Psi}^h \circ \mathbf{\Psi}^{-h} \mathbf{y} = \mathbf{\Psi}^{-h} \circ \mathbf{\Psi}^h \mathbf{y} = \mathbf{y}$, see [**?**, Def. 2.1.27]). Find γ_1, γ_2 and γ_3, such that $\tilde{\mathbf{\Psi}}^h$ is reversible as well. What is the maximal order of convergence for $\tilde{\mathbf{\Psi}}^h$?

Solution: $\tilde{\mathbf{\Psi}}^h$ can only be reversible if $\gamma_1 = \gamma_3$. This implies (recall that p has to be even) that

$$\gamma_1 = \gamma_3 = \frac{1}{2 - 2^{1/(p+1)}}, \quad \gamma_2 = \frac{-2^{1/(p+1)}}{2 - 2^{1/(p+1)}}.$$

For the second question, **??** implies $p+1$. However, since reversible methods have an even order of convergence, this automatically improves to $p+2$.

(2c) Determine empirically the order of convergence for the composition method based on the Störmer–Verlet method with parameters

$$\gamma_1 = \gamma_3 = \frac{1}{2 - 2^{1/3}}, \quad \gamma_2 = \frac{-2^{1/3}}{2 - 2^{1/3}}.$$

Conduct an experiment with the Kepler problem, which is described by the Hamiltonian

$$H(p_1, p_2, q_1, q_2) = \frac{1}{2}(p_1^2 + p_2^2) - \frac{1}{\sqrt{q_1^2 + q_2^2}}.$$

The initial values are

$$q_1(0) = 1 - e, \quad q_2(0) = p_1(0) = 0, \quad p_2(0) = \sqrt{\frac{1+e}{1-e}},$$

where $e = 0.6$ and the integration interval is $[0, 7.5]$. Calculate the error at the end time for step sizes $h = 2^{-4}, \dots, 2^{-8}$ and determine the algebraic convergence rate. The reference solution is given in the template `composition.m`.

Solution: The Hamiltonian differential equation is

$$\begin{cases} \dot{\mathbf{q}} = \mathbf{p}, \\ \dot{\mathbf{p}} = -(q_1^2 + q_2^2)^{-3/2}\mathbf{q}. \end{cases}$$

The approximate algebraic convergence rate is 3.9678.

Listing 2.1: Solution to **??**

```
function [p,q] = stoermerverlet(h,p,q,g)

% One step of the Stoermer-Verlet method

p05 = p + h/2*g(q);

q = q + h*p05;

p = p05 + h/2*g(q);

end
```

Listing 2.2: Solution to **??**

```
function composition
close all

% Parameters
gamma1 = 1/(2-2^(1/3));
gamma3 = gamma1;
gamma2 = -2^(1/3)/(2-2^(1/3));

e = 0.6;

% Initial values
q0 = [1-e;0];
p0 = [0; sqrt((1+e)/(1-e))];

% Reference solution
qref = [-0.828164402690770818204757585370;
         0.778898095658635447081654480796];
pref = [-0.856384715343395351524486215030;
        -0.160552150799838435254419104102];
```

```
% Step-sizes
h = 2.^-(4:8);

% Right-hand side
g = @(q) -(q(1)^2+q(2)^2)^(-3/2)*q;

err=zeros(size(h));
% For all step-sizes...
for jj = 1:length(h)

    % ...apply the Stoermer-Verlet method...
    p = p0;
    q = q0;

    for ii=1:7.5/h(jj)

        [p,q] = stoermerverlet(gamma1*h(jj),p,q,g);
        [p,q] = stoermerverlet(gamma2*h(jj),p,q,g);
        [p,q] = stoermerverlet(gamma3*h(jj),p,q,g);

    end

    % ...and determine the error
    err(jj) = norm([qref;pref]-[q;p]);
end

figure;
loglog(h,err,'*-');

% Determine the convergence rate
fit = polyfit(log(h),log(err),1);
fprintf('Algebraic convergence rate = %2.4f\n',fit(1));

end
```

Composition with the adjoint Method

Composition methods like (**??**) are only defined for an even p. For odd orders of convergence, it is possible to construct compositions with the adjoint method. The adjoint evolution operator

$$\mathbf{y}_1 = \overline{\mathbf{\Psi}}^h \mathbf{y}_0$$

of a single-step method $\mathbf{\Psi}^h$ is implicitly defined by

$$\mathbf{\Psi}^{-h} \mathbf{y}_1 = \mathbf{y}_0$$

(recall that for reversible methods, we have $\overline{\boldsymbol{\Psi}}^h = \boldsymbol{\Psi}^h$). The discrete evolution $\widehat{\boldsymbol{\Psi}}^h$ of the composition method with the adjoint of $\boldsymbol{\Psi}^h$ is defined by

$$\widehat{\boldsymbol{\Psi}}^h := \boldsymbol{\Psi}^{h/2} \circ \overline{\boldsymbol{\Psi}}^{h/2}.$$

(2d) Determine the Butcher scheme of the composition method with the adjoint of the implicit Euler method.

Solution: The implicit Euler method is

$$\boldsymbol{\Psi}^h \mathbf{y}_0 = \mathbf{y}_0 + h\mathbf{f}(\mathbf{y}_1).$$

Its adjoint is

$$\overline{\boldsymbol{\Psi}}^h \mathbf{y}_0 = \mathbf{y}_0 + h\mathbf{f}(\mathbf{y}_0).$$

The composition method is thus

$$\begin{cases} \tilde{\mathbf{y}} = \mathbf{y}_0 + \frac{h}{2}\mathbf{f}(\mathbf{y}_0) \\ \mathbf{y}_1 = \tilde{\mathbf{y}} + \frac{h}{2}\mathbf{f}(\mathbf{y}_1) = \mathbf{y}_0 + \frac{h}{2}\mathbf{f}(\mathbf{y}_0) + \frac{h}{2}\mathbf{f}(\mathbf{y}_1). \end{cases}$$

This is exactly the implicit trapezoidal rule, whose Butcher scheme is

$$\begin{array}{c|cc} 0 & 0 & 0 \\ 1 & 1/2 & 1/2 \\ \hline & 1/2 & 1/2 \end{array}.$$

(2e) Show that the composition of the explicit Euler method with its adjoint results in the implicit mid-point rule.

Solution: The explicit Euler scheme is

$$\boldsymbol{\Psi}^h \mathbf{y}_0 = \mathbf{y}_0 + h\mathbf{f}(\mathbf{y}_0),$$

whereas its adjoint is

$$\overline{\boldsymbol{\Psi}}^h \mathbf{y}_0 = \mathbf{y}_0 + h\mathbf{f}(\mathbf{y}_1).$$

Together, the composition is

$$\begin{cases} \tilde{\mathbf{y}} = \mathbf{y}_0 + \frac{h}{2}\mathbf{f}(\tilde{\mathbf{y}}) \\ \mathbf{y}_1 = \tilde{\mathbf{y}} + \frac{f}{2}\mathbf{f}(\tilde{\mathbf{y}}) = \mathbf{y}_0 + h\mathbf{f}(\tilde{\mathbf{y}}). \end{cases}$$

Define $\mathbf{k} := \mathbf{f}(\tilde{\mathbf{y}})$. We have that

$$\begin{cases} \mathbf{k} = \mathbf{f}(\tilde{\mathbf{y}}) = \mathbf{f}(\mathbf{y}_0 + \frac{h}{2}\mathbf{k}), \\ \mathbf{y}_1 = \mathbf{y}_0 + h\mathbf{k}, \end{cases}$$

which is the implicit mid-point rule, as claimed.

(2f) Show that, for the ODE

$$\dot{q} = p, \quad \dot{p} = g(q),$$

with locally Lipschitz function g, the composition method of the symplectic Euler method

$$\begin{cases} q_1 = q_0 + hp_0 \,, \\ p_1 = p_0 + hg(q_1) \,, \end{cases}$$

with its adjoint is the Störmer–Verlet method.

Solution: The adjoint method is

$$\begin{cases} q_1 = q_0 + hp_1, \\ p_1 = p_0 + hg(q_0). \end{cases}$$

The composition method is

$$\begin{cases} \tilde{q} = q_0 + \frac{h}{2}\tilde{p}, & \\ \tilde{p} = p_0 + \frac{h}{2}g(q_0) & =: p_{1/2}, \\ q_1 = \tilde{q} + \frac{h}{2}\tilde{p} & = q_0 + hp_{1/2}, \\ p_1 = \tilde{p} + \frac{h}{2}g(q_1) & = p_{1/2} + \frac{h}{2}g(q_1), \end{cases}$$

which is the Störmer-Verlet method.

Problem 3 Perturbation Theory for DAEs

A DAE of the form

$$\begin{cases} y' &= z \\ 0 &= (1-y^2)z - y \end{cases} \tag{3.1}$$

can be approximated by

$$\begin{cases} y' &= z \\ \varepsilon z' &= (1-y^2)z - y \end{cases} \tag{3.2}$$

with $0 < \varepsilon \ll 1$. We speak of smooth solutions to (**??**), if the functions y and z have a Taylor expansion ε; or in other words, that there exist functions y_0, y_1, z_0, z_1, such that

$$y(t) = y_0(t) + \varepsilon y_1(t) + \mathcal{O}(\varepsilon^2)$$
$$z(t) = z_0(t) + \varepsilon z_1(t) + \mathcal{O}(\varepsilon^2)$$

holds for all times t.

(3a) What is the index of the DAE (**??**)?

Solution: It has index one for $y \neq 1$, since

$$\mathbf{D}_z\mathbf{c}(y,z) = 1 - y^2.$$

(3b) Show that the functions y_0, y_1, z_0 and z_1 satisfy the following differential equations

$$\begin{cases} y_0' &= z_0, \\ 0 &= (1-y_0^2)z_0 - y_0, \end{cases} \qquad \begin{cases} y_1' &= z_1, \\ z_0' &= (1-y_0^2)z_1 - y_1 - 2y_0y_1z_0. \end{cases}$$

Solution: By inserting the Taylor expansions in (**??**), we see that

$$\begin{cases} y_0' + \varepsilon y_1' = z_0 + \varepsilon z_1 + \mathcal{O}(\varepsilon^2), \\ \varepsilon z_0' = (1 - y_0^2 - 2\varepsilon y_0 y_1)(z_0 + \varepsilon z_1) - y_0 - y_1\varepsilon + \mathcal{O}(\varepsilon^2) \\ \qquad = (1-y_0^2)z_0 - y_0 + \varepsilon(-2y_0y_1z_0 + z_1(1-y_0^2) - y_1) + \mathcal{O}(\varepsilon^2). \end{cases}$$

Since this holds for all ε in some interval $[0, \varepsilon_0]$, we arrive at the result by comparing coefficients.

(3c) The perturbed problem (**??**) has a smooth solution if the initial value of z is adapted to the initial value of y. Determine the initial values of z_0 and z_1 in with respect to y_0 for the choice of

$$y(0) = y_0(0).$$

HINT: The differential equations from **??** hold for all times $t \geq 0$.

Solution: The equation

$$0 = (1 - y_0(t)^2)z_0(t) - y_0(t),$$

implies

$$z_0(t) = \frac{y_0(t)}{1 - y_0(t)^2} \quad \forall t,$$

and thus

$$z_0(0) = \frac{y_0(0)}{1 - y_0(0)^2}.$$

The derivative of $z_0(t)$ is

$$z_0'(t) = ... = z_0(t)\frac{1+y_0^2(t)}{(1-y_0^2(t))^2}.$$

From $y(0) = y_0(0)$, we see that $y_1(0) = 0$, and therefore

$$z_1(0) = ... = y_0(0)\frac{1+y_0(0)^2}{(1-y_0(0)^2)^5},$$

since

$$z_0'(t) = (1-y_0(t)^2)z_1(t) - y_1(t) - 2y_0(t)y_1(t)z_0(t) \quad \forall t.$$

(3d) Write a MATLAB-function `referencesol`, which calculates the solution to the DAE (**??**) with the MATLAB-integrator `ode23t` at time $t = 0.5$. Set the absolute and relative tolerance to 10^{-8}.

Solution:

Listing 3.1: Solution to **??**

```
function Y = referencesol

% End time
T=0.5;

% Initial value
y0 = 2;
z0 = -2/3;

% Right-hand side
f = @(t,y) [y(2); (1-y(1)^2)*y(2)-y(1)];

% Mass Matrix
M = @(t,y) [1 0 ;0 0];

% Set options
opts = odeset('Mass',M,'AbsTol',1e-8,'RelTol',1e-8);

% Integrate
[~,Y] = ode23t(f,[0 T],[y0;z0],opts);

% Read solution for end time
Y=Y(end,:).';

end
```

(3e) Write a MATLAB-function `referencesol2`, which calculates the solution of the DAE (**??**) at end time $t = 0.5$ with 1000 steps of the semi-implicit Euler method.

Solution:

Listing 3.2: Solution to **??**

```
function [y,z]=referencesol2

% End time
T=0.5;

% Step-size
h=T/1000;

% Function for finding roots
F=@(y1,z1,y0) [y1-y0-h*z1;(1-y1^2)*z1-y1];

% Its derivative
DF=@(y1,z1) [1, -h; -2*y1*z1-1, (1-y1^2)];

% Initial value
y = 2;
z = -2/3;

% The semi-implicit Euler method
for ii=1:1000

    temp = [y;z]-DF(y,z)\F(y,z,y);
    y=temp(1);
    z=temp(2);

end

end
```

(3f) Complete the template `epsilonconv`, in which the convergence of the solution of(**??**) at time $t = 0.5$ with respect to ε is investigated. The DAE (**??**) should be solved with 1000 steps of the semi-implicit Euler method.

Solution:

Listing 3.3: Solution to **??**

```
function epsilonconv

% End time
T=0.5;

% Initial value (z0 depends on epsilon!)
y0 = 2;
z0 = @(t) -2/3+10/81*t-292/2187*t^2-1814/19683*t^3;

% Different values of epsilon
```

```
epsilon = 2.^-(5:12);

% Reference solution
YREF = [1.596664988001934; -1.030545866002973];

% Number of steps
N = 1000;

% Step size
h=T/N;

% Function for finding roots
F = @(y1,z1,y0,z0,epsilon) [y1-y0-h*z1;
   z1-z0-h/epsilon*((1-y1^2)*z1-y1)];

% Its derivative
DF = @(y1,z1,epsilon) [1, -h; -h/epsilon*(-2*y1*z1-1),
   1-h/epsilon*(1-y1^2)];

% Allocate memory
err=zeros(1,length(epsilon));

% For all values of epsilon...
for ii = 1:length(epsilon)

    % ...integrate with semi-implicit Euler method...
    y=y0;
    z=z0(epsilon(ii));

    for jj = 1:N
        temp=[y;z]-DF(y,z,epsilon(ii))\F(y,z,y,z,epsilon(ii));
        y=temp(1);
        z=temp(2);
    end

    % ...and determine the error
    err(ii) = norm(YREF-temp);

end

% Suitable plot of the errors
figure;
loglog(epsilon,err,'*-');

% Determine the convergence rate
p=polyfit(log(epsilon),log(err),1);
fprintf('rate = %2.4f\n',p(1));
```

Listing 3.4: Testcalls for **??**

```
MISSING!
```

Listing 3.5: Output for Testcalls for **??**

```
MISSING!
```

Problem 4 Stability Regions of ODE-Integrators

The goal of this problem is to plot the stability regions of the MATLAB-integrators `ode45` and `ode23s`.

(4a) Assuming you want to plot the stability function of a single-step method, but you have only an implementation and no explicit information about the discrete evolution $\mathbf{\Psi}$. Develop and explain a strategy to tackle this problem. You may assume that the first line of the implementation is

```
function y1 = discreteEvolution(y0,f,h).
```

The inputs `y0`, `f` and `h` correspond to the initial value, the right-hand side and the step-size. The output `y1` is the discrete solution after one step.

Solution: $y_1 = S(h\lambda)y_0$, thus `S(z) = discreteEvolution(1,@(t,y) z*y,1).`

(4b) Now we want to plot the stability region of `ode23s`. The difficulty here is, that this integrator uses adaptive step-sizes. Modify and explain your strategy from **??** accordingly. Complete the template `ODEstability23s`.

Solution: `ode23s` ist A-stable.

Listing 4.1: Solution to **??**

```
function ODEstability23s

[X,Y] = meshgrid(-30:5:30);

% Create complex mesh
Z=X+1i*Y;

% Integration interval for ODE
T = [0 1];

% Evaluate stability function S on grid
L = length(X);
Z = reshape(Z,L^2,1);

opts = odeset('AbsTol',100,'RelTol',100,'stats','on');
[t,Sz] = ode23s(@(t,y) Z.*y,[0 1],ones(L^2,1),opts);

% Read the discrete solution after one step
Sz = Sz(2,:).';
Sz = reshape(Sz,L,L);

% Take into account the influence of time on X and Y
% and update these accordingly
X = t(2)*X;
Y = t(2)*Y;
```

```

% Call contourf() with level lines abs(S) = 0:0.1:1
figure;
contourf(X,Y, abs(Sz), [0:0.1:1]);

colormap(hot);
colorbar;

xlabel('Re');
ylabel('Im');
axis square;
grid on;
```

HINT: Choose high absolute and relative tolerances (100 is a good choice). Check how many `successful steps` are listed in the command window and at which times the discrete evolution was calculated.

(4c) Use your strategy from **??** to plot the stability region of `ode45`. Set the absolute and relative tolerance to ∞ (`inf` in MATLAB) and observe how many `successful steps` are shown in the command window and at which times the discrete evolution was calculated.

Solution: `ode45` has a bounded stability region.

Listing 4.2: Solution to **??**

```
function ODEstability45

[X,Y] = meshgrid(-50:1:50);

% Create complex mesh
Z=X+1i*Y;

% Integration interval for ODE
T = [0 1];

% Evaluate stability function S on grid Z
L = length(X);
Z = reshape(Z,L^2,1);

opts = odeset('AbsTol',inf,'RelTol',inf,'stats','on');
[t,Sz] = ode45(@(t,y) Z.*y,T,ones(L^2,1),opts);

% Read the discrete solution after one step
Sz = Sz(5,:).';
Sz = reshape(Sz,L,L);

% Take into account the influence of time on X and Y
% and update these accordingly
X = t(5)*X;
```

```
Y = t(5)*Y;

% Call contourf() with level lines abs(S) = 0:0.1:1
figure;
contourf(X,Y, abs(Sz), [0:0.1:1]);

colormap(hot);
colorbar;

xlabel('Re');
ylabel('Im');
axis square;
grid on;
```

Listing 4.3: Testcalls for **??**

```
MISSING!
```

Listing 4.4: Output for Testcalls for **??**

```
MISSING!
```

References

[NUMODE] Lecture Slides for the course "Numerical Methods for Ordinary Differential Equations", SVN revision # 63606.