Adaptive Cross Approximation II
Fehlerabschätzungen

Roman Jud, 18.6.2004

Vortragsübersicht

- Rückblende
  Erinnerung an die wesentlichen Definitionen, Sätze und den ACA-Algorithmus
- Fehleranalyse beim ACA-Algorithmus
- Wahl der Pivot-Elemente

Rückblende

Worum geht es?

Das ACA-Verfahren
(**A**daptive **C**ross **A**pproximation)
ist ein **Algorithmus zur Erzeugung von Niedrigrang-Approximationen** von Matrizen, basierend auf Matrixeinträgen **ohne explizite Verwendung des Kerns** $\kappa(x, y)$.

unvollständige Niedrigrang-Approximation

Ziel: Matrix-Partition in Blöcke $(t_1, t_2) \subset \mathcal{I} \times \mathcal{I}$ so, dass für die zugehörigen Definitionsbereiche $X_t := \bigcup_{i \in t} X_i = \bigcup_{i \in t} \operatorname{supp} \varphi_i$ gilt:

- $\operatorname{diam}(X_{t_2}) \leq \eta \cdot \operatorname{dist}(X_{t_1}, X_{t_2})$ (Zulässigkeitsbedingung)

oder

- $\min\{\#t_1, \#t_2\} = 1$ (Block degeneriert zu einem Vektor)

Vorgehen:

Wir betrachten einen einzelnen Block $B \in \mathbb{R}^{m \times n}$ der zu partitionierenden Matrix in $\mathbb{R}^{N \times N}$ mit den Einträgen:

$$b_{ij} := \int_\Gamma \kappa(x, y_i) \varphi_j(x) \, ds_x \qquad \begin{matrix} i = 1, \ldots, m \\ j = 1, \ldots, n \end{matrix}$$

Γ Hyperfläche des $\mathbb{R}^d$

κ Kern der Fredhomschen IG

$y_i \in X_i := \operatorname{supp} \varphi_i$ Kollokationspunkte (vgl. Kollokationsverfahren)

$\{\varphi_1, \ldots, \varphi_N\}$ Funktionenbasis für die gesuchte Lösung

s_x Flächenelement des x-Bereichs

Definition: $\mathscr{L}_j\kappa -$ Funktionen

Vor: $\kappa : \Gamma \times \mathbb{R}^d \to \mathbb{R}$ asymptotisch glatt bzgl. y

Def: $\mathscr{L}_j\kappa : \mathbb{R}^d \backslash \overline{X_{t_1}} \to \mathbb{R}$

$$(\mathscr{L}_j\kappa)(y) := \int_\Gamma \kappa(x,y)\varphi(x)\, ds_x \qquad j = 1, \ldots, n$$

Bemerkungen:

- κ asymptotisch glatt bzgl. $y :\Leftrightarrow \kappa(x,\cdot) \in C^\infty(\mathbb{R}^d \backslash \{x\})$ für fast alle $x \in \Gamma$

$$|D_y^\alpha \kappa(x,y)| \leq c_p |x-y|^{g-p} \quad \forall\, g < 0, \alpha \in \mathbb{N}_0^d$$

- κ asymptotisch glatt bzgl. $y \implies \mathscr{L}_j\kappa$ wohldefiniert

- Einträge des betrachteten Matrixblocks $B \in \mathbb{R}^{m \times n}$

$$b_{ij} := \int_\Gamma \kappa(x, y_i)\varphi_j(x)\, ds_x = \mathscr{L}_j(y_i)$$

Übersicht zum Algorithmus

geg:
- $y_1, \ldots, y_m$ mit $y_i \in X_i = \operatorname{supp} \varphi_i \subset \Gamma$ Kollokationspunkte
- $B \in \mathbb{R}^{m \times n}$ einzelner Block der zu approx. Matrix
- $m' \in \mathbb{N}$ "Matrix-Erweiterungsgrad"

Vorbereitungsschritt: Matrix-Erweiterung

Mit geeigneten Punkten $\zeta_1, \ldots, \zeta_{m'} \in \mathbb{R}^d$ (nicht genauer beschrieben) bilde den Vektor

$$z := (\zeta_1, \ldots, \zeta_{m'}, y_1, \ldots, y_m) \in \mathbb{R}^{(m'+m)}$$

Damit erweitern wir die Matrix $B \in \mathbb{R}^{m \times n}$, $b_{ij} = (\mathscr{L}_j \kappa)(y_i) = \int_\Gamma \kappa(x, y_i) \varphi_j(x)\, ds_x$

um m' Zeilen zu einer Matrix $\widehat{B} \in \mathbb{R}^{(m'+m) \times n}$ mit $\widehat{b_{ij}} := (\mathscr{L}_j \kappa)(z_i)$

$b_{ij} = (\mathscr{L}_j \kappa)(y_i)$ $B \in \mathbb{R}^{m \times n}$

erweitern

um m' Zeilen

$\widehat{b_{ij}} = (\mathscr{L}_j \kappa)(\zeta_i)$

$\widehat{b_{ij}} = (\mathscr{L}_j \kappa)(y_i)$

$\widehat{B} \in \mathbb{R}^{(m'+m) \times n}$

Übersicht zum Algorithmus

- Der Algorithmus liefert für den steigenden "Approximationsgrad" $k \in \mathbb{N}$ (Laufvariable) die Spaltenvektoren

$$\hat{u}_k \in \mathbb{R}^{m'+m} \quad \text{und} \quad v_k \in \mathbb{R}^n$$

- Die approximierende Matrix für die um m' erweiterte Matrix $\widehat{B}$ ist dann gegeben durch

$$\widehat{B} \approx \widehat{S}_k := \sum_{\ell=1}^{k} \underbrace{\hat{u}_l v_l^T}_{\in \mathbb{R}^{(m'+m)\times n}}$$

$k \in \mathbb{R}$ ist der Wert der Laufvariable k am Ende des Algorithmus

respektive in Restterm-Schreibweise

$$\widehat{B} = \widehat{S}_k + \widehat{R}_k$$

Der Algorithmus “en détail”

$k := 1;$

$i_1 := 1;$

wiederhole für $j = 1, \ldots, n$

$$(\tilde{v}_k)_j := (\mathscr{L}_j \kappa)(\zeta_{i_k}) - \sum_{\ell=1}^{k-1} (\hat{u}_\ell)_{i_k} (v_\ell)_j$$

falls $\tilde{v}_k$ verschwindet, dann $i_k := i_k + 1;$

sonst beginne

$j_k := \operatorname{argmax}_{j=1,\ldots,n} |(\tilde{v}_k)_j|;$

$\mathbb{R} \ni \gamma_k := (\tilde{v}_k)_{j_k}{}^{-1};$

$v_k := \gamma_k \hat{v}_k;$

für $i = 1, \ldots, m' + m$

$$(\hat{u}_k)_i := (\mathscr{L}_{j_k} \kappa)(z_i) - \sum_{\ell=1}^{k-1} (\hat{u}_\ell)_i (v_\ell)_{j_k}$$

$i_{k+1} := i_k + 1;$

$k := k + 1;$

beende

solange bis $i_k > m'$

Der Algorithmus “en détail”

$k := 1;$

$i_1 := 1;$

wiederhole für $j = 1, \ldots, n$

$$(\tilde{v}_k)_j := (\mathscr{L}_j \kappa)(\zeta_{i_k}) - \sum_{\ell=1}^{k-1} (\hat{u}_\ell)_{i_k} (v_\ell)_j$$

`falls` $\tilde{v}_k$ `verschwindet, dann` $i_k := i_k + 1;$

`sonst beginne`

$j_k := \operatorname{argmax}_{j=1,\ldots,n} |(\tilde{v}_k)_j|;$

$\mathbb{R} \ni \gamma_k := (\tilde{v}_k)_{j_k}{}^{-1};$

$v_k := \gamma_k \hat{v}_k;$

`für` $i = 1, \ldots, m' + m$

$$(\hat{u}_k)_i := (\mathscr{L}_{j_k} \kappa)(z_i) - \sum_{\ell=1}^{k-1} (\hat{u}_\ell)_i (v_\ell)_{j_k}$$

$i_{k+1} := i_k + 1;$

$k := k + 1;$

`beende`

solange bis $i_k > m'$

- Der Approximationsgrad k und der Index i_1 werden mit 1 initialisiert.
- Wie gross i_1 wird, entscheidet sich in der kleinen Programmschleife.

Der Algorithmus “en détail”

$k := 1;$

$i_1 := 1;$

wiederhole für $j = 1, \ldots, n$

$$(\tilde{v}_k)_j := (\mathscr{L}_j \kappa)(\zeta_{i_k}) - \sum_{\ell=1}^{k-1} (\hat{u}_\ell)_{i_k} (v_\ell)_j$$

`falls` $\tilde{v}_k$ `verschwindet, dann` $i_k := i_k + 1;$

`sonst beginne`

$j_k := \operatorname{argmax}_{j=1,\ldots,n} |(\tilde{v}_k)_j|;$

$\mathbb{R} \ni \gamma_k := (\tilde{v}_k)_{j_k}{}^{-1};$

$v_k := \gamma_k \hat{v}_k;$

`für` $i = 1, \ldots, m' + m$

$$(\hat{u}_k)_i := (\mathscr{L}_{j_k} \kappa)(z_i) - \sum_{\ell=1}^{k-1} (\hat{u}_\ell)_i (v_\ell)_{j_k}$$

$i_{k+1} := i_k + 1;$

$k := k + 1;$

`beende`

solange bis $i_k > m'$

- In der kleinen Programmschleife wird der Hilfsvektor $\tilde{v}_k \in \mathbb{R}^n$ konstruiert. Er dient zur Berechnung des Vektors $v_k \in \mathbb{R}^n$, welcher in

$$\widehat{B} \approx \widehat{S}_k := \sum_{\ell=1}^{k} \hat{u}_\ell v_\ell^T$$

als v_ℓ in Erscheinung tritt.

- v_k darf nicht der Nullvektor sein.

Der Algorithmus “en détail”

$k := 1;$

$i_1 := 1;$

wiederhole für $j = 1, \ldots, n$

$$(\tilde{v}_k)_j := (\mathscr{L}_j \kappa)(\zeta_{i_k}) - \sum_{\ell=1}^{k-1} (\hat{u}_\ell)_{i_k} (v_\ell)_j$$

`falls` $\tilde{v}_k$ `verschwindet, dann` $i_k := i_k + 1;$

`sonst beginne`

$j_k := \operatorname{argmax}_{j=1,\ldots,n} |(\tilde{v}_k)_j|;$

$\mathbb{R} \ni \gamma_k := (\tilde{v}_k)_{j_k}{}^{-1};$

$v_k := \gamma_k \hat{v}_k;$

`für` $i = 1, \ldots, m' + m$

$$(\hat{u}_k)_i := (\mathscr{L}_{j_k} \kappa)(z_i) - \sum_{\ell=1}^{k-1} (\hat{u}_\ell)_i (v_\ell)_{j_k}$$

$i_{k+1} := i_k + 1;$

$k := k + 1;$

`beende`

solange bis $i_k > m'$

- “argmax” liefert den Index des maximalen Elementes einer Liste (hier: Vektor $\tilde{v}_k$)
- Für das Pivotelement $(\tilde{v}_k)_{j_k}$ (vgl. später) gilt stets
$$(\tilde{v}_k)_{j_k} \neq 0\,,$$
da der Vektor $\tilde{v}_k$ gemäss der kleinen Schlaufe des Algorithmus nicht verschwindet. γ_k ist also wohldefiniert.

Der Algorithmus “en détail”

$k := 1;$

$i_1 := 1;$

wiederhole für $j = 1, \ldots, n$

$$(\tilde{v}_k)_j := (\mathscr{L}_j \kappa)(\zeta_{i_k}) - \sum_{\ell=1}^{k-1} (\hat{u}_\ell)_{i_k} (v_\ell)_j$$

falls $\tilde{v}_k$ verschwindet, dann $i_k := i_k + 1;$

sonst beginne

$j_k := \operatorname{argmax}_{j=1,\ldots,n} |(\tilde{v}_k)_j|;$

$\mathbb{R} \ni \gamma_k := (\tilde{v}_k)_{j_k}{}^{-1};$

$v_k := \gamma_k \hat{v}_k;$

für $i = 1, \ldots, m' + m$

$$(\hat{u}_k)_i := (\mathscr{L}_{j_k} \kappa)(z_i) - \sum_{\ell=1}^{k-1} (\hat{u}_\ell)_i (v_\ell)_{j_k}$$

$i_{k+1} := i_k + 1;$

$k := k + 1;$

beende

solange bis $i_k > m'$

Hauptteil
Berechnung von
$\hat{u}_k \in \mathbb{R}^{m'+m}$ und $v_k \in \mathbb{R}^n$
welche zur Berechnung von

$$\widehat{B} \approx \widehat{S}_k := \sum_{\ell=1}^{k} \hat{u}_\ell v_\ell^T$$

verwendet werden.
(Wiederum treten die Vektoren in der Summe als $\hat{u}_\ell, v_\ell$ in Erscheinung.)

Der Algorithmus “en détail”

$k := 1;$

$i_1 := 1;$

wiederhole für $j = 1, \ldots, n$

$$(\tilde{v}_k)_j := (\mathscr{L}_j \kappa)(\zeta_{i_k}) - \sum_{\ell=1}^{k-1} (\hat{u}_\ell)_{i_k} (v_\ell)_j$$

falls $\tilde{v}_k$ verschwindet, dann $i_k := i_k + 1;$

sonst beginne

$j_k := \operatorname{argmax}_{j=1,\ldots,n} |(\tilde{v}_k)_j|;$

$\mathbb{R} \ni \gamma_k := (\tilde{v}_k)_{j_k}{}^{-1};$

$v_k := \gamma_k \hat{v}_k;$

für $i = 1, \ldots, m' + m$

$$(\hat{u}_k)_i := (\mathscr{L}_{j_k} \kappa)(z_i) - \sum_{\ell=1}^{k-1} (\hat{u}_\ell)_i (v_\ell)_{j_k}$$

$i_{k+1} := i_k + 1;$

$k := k + 1;$

beende

solange bis $i_k > m'$

Beachte:
Mit i_{k+1} wird eine neue Variable initialisiert. k wird hingegen nur inkrementiert.

Der Algorithmus “en détail”

$k := 1;$

$i_1 := 1;$

wiederhole für $j = 1, \ldots, n$

$$(\tilde{v}_k)_j := (\mathscr{L}_j \kappa)(\zeta_{i_k}) - \sum_{\ell=1}^{k-1} (\hat{u}_\ell)_{i_k} (v_\ell)_j$$

falls $\tilde{v}_k$ verschwindet, dann $i_k := i_k + 1;$

sonst beginne

$j_k := \mathrm{argmax}_{j=1,\ldots,n} |(\tilde{v}_k)_j|;$

$\mathbb{R} \ni \gamma_k := (\tilde{v}_k)_{j_k}{}^{-1};$

$v_k := \gamma_k \hat{v}_k;$

für $i = 1, \ldots, m' + m$

$$(\hat{u}_k)_i := (\mathscr{L}_{j_k} \kappa)(z_i) - \sum_{\ell=1}^{k-1} (\hat{u}_\ell)_i (v_\ell)_{j_k}$$

$i_{k+1} := i_k + 1;$

$k := k + 1;$

beende

solange bis $i_k > m'$

Abruchbedingung der grossen Programmschlaufe.

Der Algorithmus “en détail”

$k := 1;$

$i_1 := 1;$

wiederhole für $j = 1, \ldots, n$

$$(\tilde{v}_k)_j := (\mathscr{L}_j \kappa)(\zeta_{i_k}) - \sum_{\ell=1}^{k-1} (\hat{u}_\ell)_{i_k} (v_\ell)_j$$

`falls` $\tilde{v}_k$ `verschwindet, dann` $i_k := i_k + 1;$

`sonst beginne`

$j_k := \operatorname{argmax}_{j=1,\ldots,n} |(\tilde{v}_k)_j|;$

$\mathbb{R} \ni \gamma_k := (\tilde{v}_k)_{j_k}{}^{-1};$

$v_k := \gamma_k \hat{v}_k;$

`für` $i = 1, \ldots, m' + m$

$$(\hat{u}_k)_i := (\mathscr{L}_{j_k} \kappa)(z_i) - \sum_{\ell=1}^{k-1} (\hat{u}_\ell)_i (v_\ell)_{j_k}$$

$i_{k+1} := i_k + 1;$

$k := k + 1;$

`beende`

solange bis $i_k > m'$

Schlussbemerkung

$\widehat{B} = (b_{ij})_{i,j} = ((\mathscr{L}_j \kappa)(z_i))_{i,j}$

wird im Algorithmus nur in den ersten m' Zeilen und in den Spalten $j_1, \ldots, j_k$ berechnet; daher die Bezeichnung ***unvollständige*** Approximation für diesen Algorithmus.

Definition: k-tes Restglied $\widehat{R}_k$

Vor: • k “Approx’grad” bei Abbruch des Algorithmus

• $\hat{S}_k := \sum_{\ell=1}^{k} \hat{u}_\ell v_\ell^T$ Approximation von $\widehat{B} \in \mathbb{R}^{(m'+m)\times n}$

Def: $\boxed{\widehat{R}_k := \widehat{B} - \widehat{S}_k}$ k-tes Restglied

weiteres Vorgehen:

Wir konstruieren rekursiv Funktionen $r_k(x,y)$, $s_k(x,y)$ und stellen eine Beziehung zum Restglied $\widehat{R}_k$ her.

Definition: r_k, s_k

Verankerung:

$$r_0(x,y) := \kappa(x,y)$$
$$s_0(x,y) := 0$$

rekursiv:

$$r_k(x,y) := r_{k-1}(x,y) - \gamma_k(\mathscr{L}_{j_k} r_{k-1})(y) r_{k-1}(x,\zeta_{i_k})$$
$$s_k(x,y) := s_{k-1}(x,y) + \gamma_k(\mathscr{L}_{j_k} r_{k-1})(y) r_{k-1}(x,\zeta_{i_k})$$

Erinnerung/Bemerkungen:

- $(\mathscr{L}_{j_k} r_{k-1}) := \int_\Gamma r_{k-1}(x,y)\varphi_{j_k}(x)\, ds_x$
- $j_k = \operatorname{argmax}_{j=1,\ldots,n} |(\tilde{v}_k)_j|$
- i_k wird in der kleinen Schlaufe des Algorithmus bestimmt

Lemma 3.2: **Beziehung** $r_k \leftrightarrow \widehat{R}_k$

Vor: $k \geq 1$

Beh: Für alle $1 \leq i \leq m' + m;\ 1 \leq j \leq n$ gilt

$$(\widehat{R}_k)_{ij} = (\mathscr{L}_j r_k)(z_i)$$

Erinnerung: $z = (\zeta_1, \ldots, \zeta_{m'}, y_1, \ldots, y_m)$

Lemma 3.3: **Trägerbereich von** $\mathscr{L}_{j\ell} r_k$

Vor:
- $k \geq 0$
- $1 \leq \ell \leq k$

Beh:

$$(\mathscr{L}_{j\ell} r_k)(y) = 0 \qquad \forall y \in \mathbb{R}^d \backslash X_{t_1}$$

$\Longrightarrow$ Der Träger von $\mathscr{L}_{j\ell} r_k$ liegt also in $X_{t_1} = \cup_{i \in t_1} \operatorname{supp} \varphi_i$.

Abkürzende Schreibweise

$$(\mathscr{L}_j\kappa)([\zeta]_k) := \begin{pmatrix} (\mathscr{L}_j\kappa)(\zeta_{i_1}) \\ \vdots \\ (\mathscr{L}_j\kappa)(\zeta_{i_k}) \end{pmatrix}$$

Der **Zeilen**auswahlindex i_ℓ läuft von $\ell = 1$ bis $\ell = k$.

$$([\mathscr{L}]_k\kappa)(y) := \begin{pmatrix} (\mathscr{L}_{j_1}\kappa)(y) \\ \vdots \\ (\mathscr{L}_{j_k}\kappa)(y) \end{pmatrix}$$

Der **Spalten**auswahlindex j_ℓ läuft von $\ell = 1$ bis $\ell = k$.

Definition: Matrix $\widehat{A}_k \in \mathbb{R}^{k \times k}$

wichtig!

$$\widehat{A}_k := \begin{pmatrix} (\mathscr{L}_{j_1}\kappa)(\zeta_{i_1}) & \cdots & (\mathscr{L}_{j_k}\kappa)(\zeta_{i_1}) \\ \vdots & & \vdots \\ (\mathscr{L}_{j_1}\kappa)(\zeta_{i_k}) & \cdots & (\mathscr{L}_{j_k}\kappa)(\zeta_{i_k}) \end{pmatrix}$$ **Volldarstellung**

$$= \begin{pmatrix} (\mathscr{L}_{j_1}\kappa)([\zeta]_k) & \cdots & (\mathscr{L}_{j_k}\kappa)([\zeta]_k) \end{pmatrix}$$ **Spaltendarstellung**

$$= \begin{pmatrix} ([\mathscr{L}]_k\,\kappa)(\zeta_{i_1})^T \\ \vdots \\ ([\mathscr{L}]_k\,\kappa)(\zeta_{i_k})^T \end{pmatrix}$$ **Zeilendarstellung**

Erinnerung (Lineare Algebra): **Cramersche Regel**

Vor:
- $A \in GL(n, \mathbb{K})$
- $a^1, \ldots, a^n \in \mathbb{R}^n$ Spaltenvektoren von A

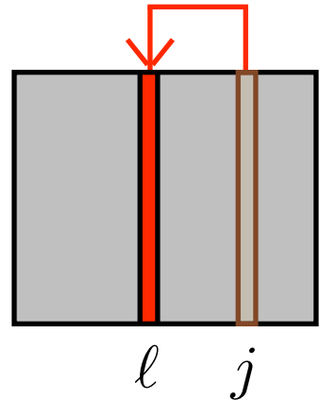
Ann: $Ax = b$ mit $b \in \mathbb{R}^n$

Beh: $$x_i = \frac{\det(a^1, \ldots, a^{i-1}, b, a^{i+1}, \ldots, a^n)}{\det A}$$

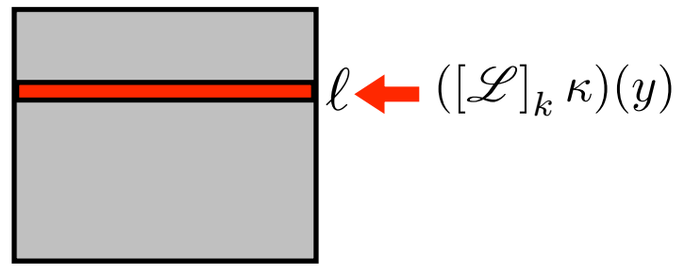
Bemerkung:
Wegen $\det(A^T) = \det(A)$ gilt eine analoge Aussage auch für das Ersetzen der i-ten Zeile durch den Vektor b.

Definition: "Cramersche" Matrizen $\widehat{A}_k(\ell, j),\ M_k(\ell, y)$ wichtig!

- $\widehat{A}_k(\ell, j) \in \mathbb{R}^{k\times k}$ entsteht aus $\widehat{A}_k$ durch **Ersetzen der ℓ-ten Spalte durch die j-te Spalte.**



- $M_k(\ell, y) \in \mathbb{R}^{k\times k}$ entsteht aus $\widehat{A}_k$ durch **Ersetzen der ℓ-ten Zeile $([\mathscr{L}]_k\,\kappa)(\zeta_{i_\ell})$ durch den allgemeinen Zeilenvektor $([\mathscr{L}]_k\,\kappa)(y)$.**



Beachte:

$\widehat{A}_k = \widehat{A}_k(\ell, j_\ell)$ $\quad$ ℓ-te Spalte durch ℓ-te Spalte ersetzen

$\widehat{A}_k = M_k(\ell, \zeta_{i_\ell})$ $\quad$ ℓ-te Zeile durch ℓ-te Zeile ersetzen

Lemma 3.4: **Determinante von** $\widehat{A}_k$

Beh: Es gelten folgende rekursive Formeln:

- Fall: $1 \leq \ell < k$

$$\det \widehat{A}_k(\ell, j) = \gamma_k^{-1} \det \widehat{A}_{k-1}(\ell, j) - (\mathscr{L}_j r_{k-1})(\zeta_{i_k}) \det \widehat{A}_{k-1}(\ell, j_k)$$

- Fall: $\ell = k$

$$k = 1: \quad \det \widehat{A}_1(1, j) = (\mathscr{L}_j \kappa)(\zeta_{i_1})$$

$$k > 1: \quad \det \widehat{A}_k(k, j) = (\mathscr{L}_j r_{k-1})(\zeta_{i_k}) \det \widehat{A}_{k-1}$$

insbesondere wegen $\widehat{A}_k = \widehat{A}_k(\ell, j_\ell)$

$$\det \widehat{A}_k = \prod_{\ell=1}^{k} \gamma_\ell^{-1}$$

Beachte:

$$\gamma_\ell^{-1} \neq 0 \quad \forall \ell = 1, \ldots, k \implies \widehat{A}_k \in GL(k, \mathbb{R})$$

wichtig!

Folgende Darstellung des Kerns κ ist von fundamentaler Bedeutung für die Analyse des Approximationsfehlers

Lemma 3.2: **Kern-Zerlegungssatz**

Vor: Die Folgen $\{s_k\}_k$ und $\{r_k\}_k$ seien gegeben durch

$$r_0(x,y) := \kappa(x,y); \quad s_0(x,y) = 0$$
$$r_k(x,y) := r_{k-1}(x,y) - \gamma_k(\mathscr{L}_{j_k} r_{k-1})(y) r_{k-1}(x,\zeta_{i_k})$$
$$s_k(x,y) := s_{k-1}(x,y) + \gamma_k(\mathscr{L}_{j_k} r_{k-1})(y) r_{k-1}(x,\zeta_{i_k})$$

Beh: Es gilt folgende Zerlegung des Kerns

$$\boxed{s_k(x,y) + r_k(x,y) = \kappa(x,y)}$$

wobei für $k \geq 1$

$$\boxed{s_k(x,y) = \underbrace{([\mathscr{L}]_k\,\kappa)(y)^T}_{\in\mathbb{R}^{1\times k}} \quad \underbrace{\widehat{A}_k^{-1}}_{\in\mathbb{R}^{k\times k}} \quad \underbrace{\kappa(x,[\zeta]_k)}_{\in\mathbb{R}^{k\times 1}}} \quad \in \mathbb{R}$$

Hauptteil

Motivation

Satz: $\mathscr{L}_j s_k$ **als Interpolant von** $\mathscr{L}_j \kappa$

Beh: i) $$(\mathscr{L}_j s_k)(y) = \sum_{k=1}^{k} \frac{\det M_k(\ell, y)}{\det \widehat{A}_k} (\mathscr{L}_j \kappa)(\zeta_{i_k})$$

ii) $\mathscr{L}_j s_k$ ist der eindeutig definierte Interpolant von $\mathscr{L}_j \kappa$ zu den Stützstellen $\zeta_{i_1}, \ldots, \zeta_{i_k}$ im Raum $\operatorname{span}\{\mathscr{L}_{j_1}\kappa, \ldots, \mathscr{L}_{j_k}\kappa\}$.

Bemerkungen:

- i) folgt aus der Cramerschen Regel und dem Kern-Zerlegungssatz.
  Wir beweisen ihn hier nicht.
- i) $\Rightarrow$ ii) zeigen wir nun anhand eines allgemeinen Interpolationsprinzips für $\mathbb{R}^d \to \mathbb{R}$-Funktionen.

Beweis $i) \Rightarrow ii)$

- Sei $\Phi := \{\varphi_1, \ldots, \varphi_k\}$ eine Basis von $\mathbb{R}^d \to \mathbb{R}$-Funktonen,
  $x_1, \ldots, x_k \in \mathbb{R}^d$, $f_1, \ldots, f_k \in \mathbb{R}$ Stützpunkte, Stützwerte

- **Def:**

$$M_\ell(x) := \begin{pmatrix} \varphi_1(x_1) & \cdots & \varphi_k(x_1) \\ \vdots & & \vdots \\ \varphi_1(x) & \cdots & \varphi_k(x) \\ \vdots & & \vdots \\ \varphi_1(x_k) & \cdots & \varphi_k(x_k) \end{pmatrix} \leftarrow \ell$$

$$\boxed{M := M_\ell(x_\ell)} \in \mathbb{R}^{k \times k}$$

Annahme: $\det M \neq 0$

- **Def:** $\chi_\ell(x) := \dfrac{\det M_\ell(x)}{\det M} \in \operatorname{span} \Phi \Rightarrow \boxed{\chi_\ell(x_i) = \delta_{i\ell}}$

χ ist also eine Lagrange-Funktion für die Interpolation

- **Def:** $\boxed{L_k^\Phi(x) := f_1\chi_1(x) + \ldots + f_k\chi_k(x)}$

  Dieser Interplant löst das Interpolationsproblem $L_k^\Phi(x_i) = f_i \quad i = 1, \ldots, k$ eindeutig.

Bemerkung: $(\mathscr{L}_j s_k)(y) = \sum_{k=1}^{k} \dfrac{\det M_k(\ell, y)}{\det \widehat{A}_k} (\mathscr{L}_j \kappa)(\zeta_{i_k})$

also entspricht in unserem Satz $\widehat{A} \leftrightarrow M$ und $M(\ell, y) \leftrightarrow M_\ell(x)$. □

Fehleranalyse

Vorgehensweise:

Wir wollen den Interpolationsfehler des Interpolanten

$$\mathscr{L}_j s_k \text{ von } \mathscr{L}_j \kappa \text{ an den Stellen } \zeta_{i_1}, \ldots, \zeta_{i_k}$$

untersuchen.
Um dies zu tun, setzen wir den Interpolationsfehler

$$\mathscr{L}_j r_k \text{ auf dem Raum } \operatorname{span}\{\mathscr{L}_{j_1}\kappa, \ldots, \mathscr{L}_{j_k}\kappa\}$$

zum Fehler des Interpolanten in einem anderen Funktionenraum

$$\operatorname{span}\Psi \quad \text{mit} \quad \Psi := \{\psi_1, \ldots, \psi_{m'}\}$$

in Beziehung.

Lemma 3.6: **Fehlerabschätzung der Koll'matrix-Komponenten**

Vor: $i_1, \ldots, i_k$ durch den Algorithmus erzeugt

Beh: Die Funktionen $\mathscr{L}_j r_k$ erfüllen

$$(\mathscr{L}_j r_k)(y) = E^{\Psi}_{m'}\left[\mathscr{L}_j \kappa\right] - \sum_{\ell=1}^{k} \frac{\det \widehat{A}_k(\ell, j)}{\det \widehat{A}_k} E^{\Psi}_{m'}\left[\mathscr{L}_{j_\ell} \kappa\right](y) \quad \forall y \in \mathbb{R}^d \backslash \overline{X}_{t_1}$$

Bemerkung:

Wenn wir für y die Kollokationspunkte $y_i \in X_i = \operatorname{supp} \psi_i$ einsetzen, erhalten wir eine Fehlerabschätzung für die Matrixeinträge der Kollokationsmatrix

$$a_{ij} = (\mathscr{L}_j \kappa)(y_i) = (\mathscr{L}_j (s_k + r_k))(y_i) = (\mathscr{L}_j s_k)(y_i) + (\mathscr{L}_j r_k)(y_i)$$

$$(\mathscr{L}_j r_k)(y) = E^{\Psi}_{m'}\left[\mathscr{L}_j \kappa\right] - \sum_{\ell=1}^{k} \frac{\det \widehat{A}_k(\ell, j)}{\det \widehat{A}_k} E^{\Psi}_{m'}\left[\mathscr{L}_{j_\ell} \kappa\right](y) \quad \forall y \in \mathbb{R}^d \backslash \overline{X}_{t_1}$$

Beweis-Übersicht

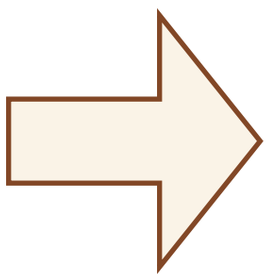
- Wir zeigen im Wesentlichen die Beziehung

$$(\mathscr{L}_j r_k) = E^{\Psi}_{m'}\left[\mathscr{L}_j \kappa\right](y) - \sum_{\ell=1}^{k} E^{\Psi}_{m'}\left[\mathscr{L}_{j_\ell} \kappa\right](y) \left(\widehat{A}_k^{-1}(\mathscr{L}_j \kappa)([\zeta]_k)\right)_\ell$$

- Mit der Cramerschen Regel können wir $\left(\widehat{A}_k^{-1}(\mathscr{L}_j \kappa)([\zeta]_k)\right)_\ell$ bestimmen

$$\left.\begin{aligned} \widehat{A}_k x = (\mathscr{L}_j \kappa)([\zeta]_k) \in \mathbb{R}^k \Rightarrow x_\ell = \frac{\det \widehat{A}_k(\ell, j)}{\det \widehat{A}_k} \\ \widehat{A}_k x = (\mathscr{L}_j \kappa)([\zeta]_k) \in \mathbb{R}^k \Rightarrow x = \widehat{A}_k^{-1}(\mathscr{L}_j \kappa)([\zeta]_k) \end{aligned}\right\} \left(\widehat{A}_k^{-1}(\mathscr{L}_j \kappa)([\zeta]_k)\right)_\ell = \frac{\det \widehat{A}_k(\ell, j)}{\det \widehat{A}_k}$$

Die Behauptung ist dann offensichtlich bewiesen.



Wir zeigen im Folgenden also den ersten Punkt.

Schritt 1: Notation

Beweis

Sei χ_i die i-te *Lagrange-Funktion* bzgl. System $\Psi := \{\psi_1, \ldots, \psi_{m'}\}$ und Stützstellen $\{\zeta_1, \ldots, \zeta'_m\}$, d.h.

$$\chi_i(\zeta_\ell) = \delta_{i\ell}$$

- **Def:**

$$\chi(y) := \begin{pmatrix} \chi_1(y) \\ \vdots \\ \chi_{m'}(y) \end{pmatrix}$$

- Der Interpolant

$$L^{\Psi}_{m'}\left[\mathscr{L}_j\kappa\right](y) := \sum_{\ell=1}^{m'} (\mathscr{L}_j\kappa)(\zeta_{i_\ell})\,\chi_\ell(y)$$

hat in Vektorschreibsweise also die Form

$$L^{\Psi}_{m'}\left[\mathscr{L}_j\kappa\right](y) = \underbrace{(\mathscr{L}_j\kappa)([\zeta]_{m'})^T}_{\in\mathbb{R}^{1\times m'}}\underbrace{\chi(y)}_{\in\mathbb{R}^{m'}}$$

Schritt 2: Summenaufspaltung bzgl. Indizes $\{i_1, \ldots, i_k\}$

Beweis

Die im Algorithmus erzeugten Auswahlindizes $\{i_1, \ldots, i_k\}$ können auf $\mathbb{N} \cap [1, m']$ in unregelmässigen Abständen verteilt sein:

wiederhole für $j = 1, \ldots, n$

$$(\tilde{v}_k)_j := (\mathscr{L}_j \kappa)(\zeta_{i_k}) - \sum_{\ell=1}^{k-1} (\hat{u}_\ell)_{i_k} (v_\ell)_j$$

falls $\tilde{v}_k$ verschwindet, dann $i_k := i_k + 1$;

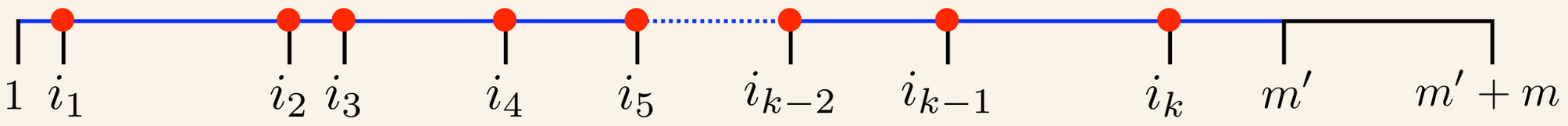
sonst beginne

$\vdots$

$i_{k+1} := i_k + 1$;

beende

solange bis $i_k > m'$



rot: Auswahlindizes

blau: Zwischenbereiche

Schritt 2: Summenaufspaltung bzgl. Indizes $\{i_1, \ldots, i_k\}$

Beweis

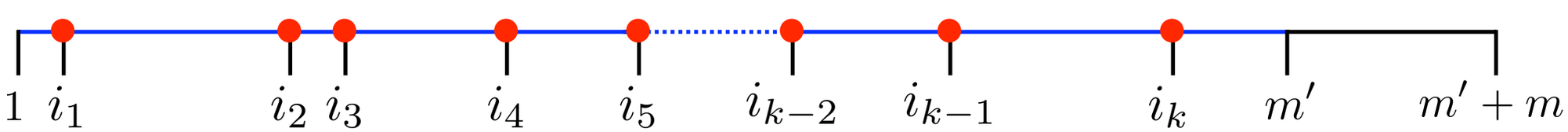
Die im Algorithmus erzeugten Auswahlindizes $\{i_1, \ldots, i_k\}$ können auf $\mathbb{N} \cap [1, m']$ in unregelmässigen Abständen verteilt sein:

Wir spalten die Summe des Interpolanten

$$L_{m'}^{\Psi}\left[\mathscr{L}_j \kappa\right](y) = \sum_{\ell=1}^{m'} (\mathscr{L}_j \kappa)(\zeta_{i_\ell})\, \chi_\ell(y)$$

nun bzgl. dieser Auswahlindizes auf und erhalten **(1)**

$$L_{m'}^{\Psi}\left[\mathscr{L}_j \kappa\right](y) = \underbrace{\sum_{i=1}^{i_1-1} (\mathscr{L}_j \kappa)(\zeta_i)\chi_i(y)}_{\text{blau}} + \sum_{\ell=1}^{k} \Bigg[\underbrace{(\mathscr{L}_j \kappa)(\zeta_{i_\ell})\chi_{i_\ell}(y)}_{\text{rot}} + \underbrace{\sum_{i=i_\ell+1}^{i_{\ell+1}-1} (\mathscr{L}_j \kappa)(\zeta_i)\chi_i(y)}_{\text{blau}} \Bigg]$$



rot: Auswahlindizes

blau: Zwischenbereiche

Bemerkung: Der Beweis orientiert sich im Wesentlichen an dieser Aufspaltung!

Schritt 3:

Beweis

Für alle $i \in \mathbb{N}$ der Zwischenbereiche $i_\ell < i < i_{\ell+1}$ gilt:

a) $(\mathscr{L}_j r_\ell)(\zeta_i) = 0$

daher folgt mit

Lemma 3.5: **Kern-Zerlegungssatz**

i) $s_k(x,y) + r_k(x,y) = \kappa(x,y)$

ii) $s_k(x,y) = ([\mathscr{L}]_k\,\kappa)(y)^T\,\widehat{A}_k^{-1}\,\kappa(x,[\zeta]_k)$

$$0 \overset{\text{a)}}{=} (\mathscr{L}_j r_\ell)(\zeta_i) \overset{\text{i)}}{=} \left(\mathscr{L}_j(\kappa - s_\ell)\right)(\zeta_i) = (\mathscr{L}_j\kappa)(\zeta_i) - (\mathscr{L}_j s_\ell)(\zeta_i)$$

$$\overset{\text{ii)}}{=} (\mathscr{L}_j\kappa)(\zeta_i) - ([\mathscr{L}]_\ell\,\kappa)(\zeta_i)^T\,\widehat{A}_\ell^{-1}(\mathscr{L}_j\kappa)([\zeta])_\ell)$$

$$\Longleftrightarrow$$

$$(\mathscr{L}_j\kappa)(\zeta_i) = ([\mathscr{L}]_\ell\,\kappa)(\zeta_i)^T\,\widehat{A}_\ell^{-1}(\mathscr{L}_j\kappa)([\zeta]_\ell)$$

Schritt 4:

- Wir schreiben diesen Ausdruck $(\mathscr{L}_j\kappa)(\zeta_i) = ([\mathscr{L}]_\ell\,\kappa)(\zeta_i)^T\,\widehat{A}_\ell^{-1}(\mathscr{L}_j\kappa)([\zeta]_\ell)$ nun in Summenschreibweise aus und erhalten

$$(\mathscr{L}_j\kappa)(\zeta_i) = \sum_{\nu=1}^{\ell}\left(([\mathscr{L}]_\ell\,\kappa)(\zeta_i)^T\,\widehat{A}_\ell^{-1}\right)_\nu (\mathscr{L}_j\kappa)(\zeta_{i_\nu}) \quad \textbf{(2)}$$

- Die Zwischenbereichs-Terme nach dem Index i_1

$$\textbf{(1)}\quad L^{\Psi}_{m'}\left[\mathscr{L}_j\kappa\right](y) = \sum_{i=1}^{i_1-1}(\mathscr{L}_j\kappa)(\zeta_i)\chi_i(y) + \sum_{\ell=1}^{k}\left[(\mathscr{L}_j\kappa)(\zeta_{i_\ell})\chi_{i_\ell}(y) + \sum_{i=i_\ell+1}^{i_{\ell+1}-1}(\mathscr{L}_j\kappa)(\zeta_i)\chi_i(y)\right]$$

können wir nun schreiben als

$$\sum_{i=i_\ell+1}^{i_{\ell+1}-1}(\mathscr{L}_j\kappa)(\zeta_i)\chi_i(y) \overset{\textbf{(2)}}{=} \sum_{i=i_\ell+1}^{i_{\ell+1}-1}\sum_{\nu=1}^{\ell}\left(([\mathscr{L}]_\ell\,\kappa)(\zeta_i)^T\,\widehat{A}_\ell^{-1}\right)_\nu (\mathscr{L}_j\kappa)(\zeta_{i_\nu})\chi_i(y)$$

$$= \sum_{\nu=1}^{\ell}(\mathscr{L}_j\kappa)(\zeta_{i_\nu}) \underbrace{\sum_{i=i_\ell+1}^{i_{\ell+1}-1}\left([\zeta]_\ell\,\kappa)(\zeta_i)^T\,\widehat{A}_\ell^{-1}\right)_\nu \chi_i(y)}_{=:\alpha_\nu^{(\ell)}(y)} = \sum_{\nu=1}^{\ell}(\mathscr{L}_j\kappa)(\zeta_{i_\nu})\alpha_\nu^{(\ell)}(y)$$

- **Fazit:**

$$\sum_{i=i_\ell+1}^{i_{\ell+1}-1}(\mathscr{L}_j\kappa)(\zeta_i)\chi_i(y) = \sum_{\nu=1}^{\ell}(\mathscr{L}_j\kappa)(\zeta_{i_\nu})\alpha_\nu^{(\ell)}(y)$$

$$\text{wobei}\quad \alpha_\nu^{(\ell)}(y) := \sum_{i=i_\ell+1}^{i_{\ell+1}-1}\left([\zeta]_\ell\,\kappa)(\zeta_i)^T\,\widehat{A}_\ell^{-1}\right)_\nu \chi_i(y) \qquad \textbf{(3)}$$

Schritt 5:

Beweis

- Im ersten Zwischenbereich $1 \leq i < i_1$ gilt: $\boxed{(\mathscr{L}_j\kappa)(\zeta_i) = 0}$ **a)**

  Daher verschwindet der erste Summenterm wie folgt:

$$L_{m'}^{\Psi}\left[\mathscr{L}_j\kappa\right](y) \overset{(1)}{=} \underbrace{\sum_{i=1}^{i_1-1}(\mathscr{L}_j\kappa)(\zeta_i)\chi_i(y)}_{\text{a)}\,=0} + \sum_{\ell=1}^{k}\left[(\mathscr{L}_j\kappa)(\zeta_{i_\ell})\chi_{i_\ell}(y) + \sum_{i=i_\ell+1}^{i_{\ell+1}-1}(\mathscr{L}_j\kappa)(\zeta_i)\chi_i(y)\right]$$

- Wegen Formel von Schritt 4 $\boxed{\sum_{i=i_\ell+1}^{i_{\ell+1}-1}(\mathscr{L}_j\kappa)(\zeta_i)\chi_i(y) = \sum_{\nu=1}^{\ell}(\mathscr{L}_j\kappa)(\zeta_{i_\nu})\alpha_\nu^{(\ell)}(y)}$ **(3)**

$$L_{m'}^{\Psi}\left[\mathscr{L}_j\kappa\right](y) = \sum_{\ell=1}^{k}\left[(\mathscr{L}_j\kappa)(\zeta_{i_\ell})\chi_{i_\ell}(y) + \sum_{\nu=1}^{\ell}(\mathscr{L}_j\kappa)(\zeta_{i_\nu})\alpha_\nu^{(\ell)}(y)\right]$$

Rechnen mit Summenzeichen

$$= \sum_{\ell=1}^{k}(\mathscr{L}_j\kappa)(\zeta_{i_\ell})\chi_{i_\ell}(y) + \sum_{\nu=1}^{k}\sum_{\ell=\nu}^{k}(\mathscr{L}_j\kappa)(\zeta_{i_\nu})\alpha_\nu^{(\ell)}(y)$$

zusammenfassen der Summe bzgl. $(\mathscr{L}_j\kappa)(\zeta_{i_\ell})$

$$= \sum_{\ell=1}^{k}(\mathscr{L}_j\kappa)(\zeta_{i_\ell})\underbrace{\left(\chi_{i_\ell}(y) + \sum_{\nu=\ell}^{k}\alpha_\ell^{(\nu)}(y)\right)}_{=:q_\ell(y)} = \sum_{\ell=1}^{k}(\mathscr{L}_j\kappa)(\zeta_{i_\ell})q_\ell(y)$$

$$\Rightarrow \boxed{L_{m'}^{\Psi}\left[\mathscr{L}_j\kappa\right](y) = (\mathscr{L}_j\kappa)([\zeta]_k)^T q(y)} \quad \textbf{(4)} \text{ wobei } \quad q(y) := \begin{pmatrix} q_1(y) \\ \vdots \\ q_k(y) \end{pmatrix} \in \mathbb{R}^k$$

Schritt 6: Finale

Beweis

Für den Schluss des Beweises benötigen wir nun im Wesentlichen

Lemma 3.5: Kern-Zerlegungssatz

i) $s_k(x,y) + r_k(x,y) = \kappa(x,y)$

ii) $s_k(x,y) = ([\mathscr{L}]_k\,\kappa)(y)^T\,\widehat{A}_k^{-1}\,\kappa(x,[\zeta]_k)$

$$L_{m'}^{\Psi}\,[\mathscr{L}_j\kappa]\,(y) = (\mathscr{L}_j\kappa)([\zeta]_k)^T q(y) \qquad (4)$$

$$(\mathscr{L}_j r_k)(y) \overset{\text{i)}}{=} (\mathscr{L}_j\kappa)(y) - (\mathscr{L}_j s_k)(y) \overset{\text{ii)}}{=} (\mathscr{L}_j\kappa)(y) - ([\mathscr{L}]_k\,\kappa)(y)^T\,\widehat{A}_k^{-1}(\mathscr{L}_j\kappa)([\zeta]_k)$$

$$= \underbrace{\left((\mathscr{L}_j\kappa)(y) - L_{m'}^{\Psi}\,[\mathscr{L}_j\kappa]\,(y)\right)}_{=E_{m'}^{\Psi}\,[\mathscr{L}_j\kappa]} + L_{m'}^{\Psi}\,[\mathscr{L}_j\kappa]\,(y) - ([\mathscr{L}]_k\,\kappa)(y)^T\,\widehat{A}_k^{-1}(\mathscr{L}_j\kappa)([\zeta]_k)$$

$$\overset{(4)}{=} E_{m'}^{\Psi}\,[\mathscr{L}_j\kappa]\,(y) + (\mathscr{L}_j\kappa)([\zeta]_k)^T q(y) - ([\mathscr{L}]_k\,\kappa)(y)^T\,\widehat{A}_k^{-1}(\mathscr{L}_j\kappa)([\zeta]_k)$$

$$= E_{m'}^{\Psi}\,[\mathscr{L}_j\kappa]\,(y) - ([\mathscr{L}]_k\,\kappa)(y)^T\,\widehat{A}_k^{-1}(\mathscr{L}_j\kappa)([\zeta]_k) + q(y)^T\,\underbrace{\widehat{A}_k\,\widehat{A}_k^{-1}}_{=\mathbb{1}}(\mathscr{L}_j\kappa)([\zeta]_k)$$

Transpo-nieren von Matrizen

$$= E_{m'}^{\Psi}\,[\mathscr{L}_j\kappa]\,(y) - \left(([\mathscr{L}]_k\,\kappa)(y) - \widehat{A}_k^T q(y)\right)^T\,\widehat{A}_k^{-1}(\mathscr{L}_j\kappa)([\zeta]_k)$$

$$= E_{m'}^{\Psi}\,[\mathscr{L}_j\kappa]\,(y) - \sum_{\ell=1}^{k} E_{m'}^{\Psi}\,[\mathscr{L}_{j_\ell\kappa}]\,(y)\left(\widehat{A}_k^{-1}(\mathscr{L}_j\kappa)([\zeta]_k)\right)_\ell$$

Dies war gemäss Beweisübersicht zu zeigen. □

Wahl der Pivotelemente

Mit Hilfe von

Lemma 3.6: **Fehlerabschätzung der Koll'matrix-Komponenten**

Beh: $$(\mathscr{L}_j r_k)(y) = E^{\Psi}_{m'}\left[\mathscr{L}_j \kappa\right] - \sum_{\ell=1}^{k} \frac{\det \widehat{A}_k(\ell, j)}{\det \widehat{A}_k} E^{\Psi}_{m'}\left[\mathscr{L}_{j_\ell} \kappa\right](y) \quad \forall y \in \mathbb{R}^d \backslash \overline{X}_{t_1}$$

lässt sich nun der **Approximationsfehler kontrollieren**, indem wir die **Koeffizienten abschätzen**.

Im Allgemeinen erzeugt die Wahl von j_k im Algorithmus keine Untermatrix mit maximaler Determinante im Betrag. Dennoch können wir sehen, dass die Strategie, immer das maximale Element zu wählen, die beste Wahl ist bezüglich maximaler Determinanten, wenn wir alle zuvor gewählten Indizes $j_1, \dots j_{k-1}$ festhalten.

Dies zu zeigen, ist Gegenstand des folgenden Lemmas.

Lemma 3.7: **Abschätzung der Koeffizienten**

Ann: In jedem Schritt des Algorithmus wählen wir j_k so, dass

$$|(\mathscr{L}_{j_k} r_{k-1})| \geq |(\mathscr{L}_j r_{k-1})(\zeta_{i_k})| \qquad \forall 1 \leq j \leq n$$

Beh: Dann gilt für alle $1 \leq \ell \leq k$ und $j = 1, \ldots, n$

$$|\det \widehat{A}_k(\ell, j)| \leq 2^{k-\ell} |\det \widehat{A}_k|$$

Mit diesem Lemma und

Lemma 3.6: **Fehlerabschätzung der Koll'matrix-Komponenten**

Beh:
$$(\mathscr{L}_j r_k)(y) = E^{\Psi}_{m'}[\mathscr{L}_j \kappa] - \sum_{\ell=1}^{k} \frac{\det \widehat{A}_k(\ell, j)}{\det \widehat{A}_k} E^{\Psi}_{m'}[\mathscr{L}_{j_\ell} \kappa](y) \quad \forall y \in \mathbb{R}^d \backslash \overline{X}_{t_1}$$

erhalten wir die gesuchte Abschätzung des Approximationsfehlers:

$$|(\mathscr{L}_j r_k)(y)| \leq (1 + 2^k) \sup_{y \in X_{t_2}} |E^{\Psi}_{m'}(\mathscr{L}_j \kappa)(y)|$$

Maus