Numerik nicht-lokaler Operatoren

Steffen Börm

Stand 2. Februar 2018

Inhaltsverzeichnis

1 Modellproblem

Zur Einführung in die Thematik befassen wir uns mit einem typischen Modellproblem, an dem sich die Grundideen, die bei der Behandlung nicht-lokaler Phänomene zum Einsatz kommen, gut erklären lassen.

1.1 Gravitation in Vielkörpersystemen

Ein relativ bekanntes Beispiel für eine nicht-lokale Wechselwirkung ist die Gravitation: Die Sonne hält die Erde auf ihrer Bahn, obwohl beide ungefähr 150 Millionen Kilometer voneinander entfernt sind. Wenn wir zur Vereinfachung auf relativistische Effekte verzichten und uns auf die auf Newton zurückgehende Gravitationstheorie beschränken, übt eine Masse m_1 an einem Punkt x_1 auf eine zweite Masse m_2 an einem Punkt x_2 eine Kraft von

$$F_{21} = \gamma \frac{m_1 m_2}{\|x_1 - x_2\|_2^3}(x_1 - x_2)$$

aus, die Kraft ist also umgekehrt proportional zum Quadrat des Abstands und zieht die zweite Masse in Richtung der ersten.

Wenn wir nun zu dem Fall vieler Massen $m_1, \ldots, m_n$ an Punkten $x_1, \ldots, x_n$ übergehen, summieren sich die auf eine Masse m_i wirkenden Kräfte zu

$$F_i = \sum_{\substack{j=1 \\ j \neq i}}^{n} \gamma \frac{m_i m_j}{\|x_j - x_i\|_2^3}(x_j - x_i)$$

und wir stellen fest, dass wir für große Werte von n, also bei Systemen mit vielen Massen, ein Effizienzproblem haben: Für jede Masse müssen $n-1$ Einzelkräfte berechnet werden, so dass insgesamt ein zu $n(n-1)$ proportionaler Gesamtaufwand entsteht.

Wenn man bedenkt, dass unsere Galaxie einige hundert Millionen Sonnen enthält, ist klar, dass ein Algorithmus, dessen Aufwand letztendlich quadratisch in der Anzahl der Sonnen wächst, für größere Simulationen ungeeignet ist. Wir sind also daran interessiert, den Rechenaufwand deutlich zu reduzieren.

Dazu machen wir uns zunutze, dass die Interaktion der Massen m_i und m_j im Wesentlichen durch ihren Abstand $x_j - x_i$ beschrieben ist, und dass dieser Abstand „glatt“ in die Berechnung der Kraft eingeht. Genauer gesagt führen wir die Hilfsfunktion

$$g : \mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}^3, \qquad (x, y) \mapsto \gamma \frac{y - x}{\|y - x\|_2^3},$$

ein und schreiben die Kraft in der Form

$$F_i = \sum_{\substack{j=1 \\ j \neq i}}^{n} m_i m_j g(x_i, x_j). \tag{1.1}$$

Falls x und y hinreichend weit voneinander entfernt sind, ist g auf einer Umgebung des Punktpaars (x, y) unendlich oft differenzierbar und sogar analytisch, so dass wir darauf hoffen dürfen, dass sich die Funktion durch etwas Handlicheres approximieren lässt.

1.2 Approximation per Interpolation

Eine naheliegende Wahl für eine Approximation der Funktion g ist dabei die Interpolation: Wir wählen Interpolationspunkte $\xi_1, \ldots, \xi_k$ innerhalb eines Gebiets σ sowie passende Lagrange-Polynome $\mathcal{L}_1, \ldots, \mathcal{L}_k$ (Details der mehrdimensionalen Interpolation finden sich in Kapitel 3) und interpolieren g in der y-Komponente:

$$\tilde{g}_\sigma(x, y) = \sum_{\nu=1}^{k} g(x, \xi_\nu) \mathcal{L}_\nu(y).$$

Dieses Polynom (zumindest in y ist es eines) approximiert die Funktion g allerdings nur, wenn der Abstand zwischen σ und x groß im Vergleich zu dem Durchmesser von σ ist, denn ansonsten verhindert die Nähe zu der Singularität der Funktion g die schnelle Konvergenz.

Deshalb wählen wir ein weiteres Teilgebiet $\tau \subseteq \mathbb{R}^3$ derart, dass alle Punkte $x \in \tau$ hinreichend weit von allen Punkten $y \in \sigma$ entfernt sind. Nur für derartige Paare von Punkten dürfen wir die Approximation $\tilde{g}$ guten Gewissens verwenden.

Für die Approximation der Gravitationskraft bietet es sich an, diejenigen Massen zu untersuchen, die in τ beziehungsweise σ liegen, deren Indizes also in den Mengen

$$\hat{\tau} = \{i \in \{1, \ldots, n\} \ : \ x_i \in \tau\}, \qquad \hat{\sigma} = \{j \in \{1, \ldots, n\} \ : \ x_j \in \sigma\} \tag{1.2}$$

liegen. So erhalten wir

$$g(x_i, x_j) \approx \tilde{g}_\sigma(x_i, x_j) \qquad \text{für alle } i \in \hat{\tau},\ j \in \hat{\sigma}.$$

Damit können wir nun wenigstens einen Teil der Summe (1.1) umformulieren:

$$\begin{aligned} \sum_{j \in \hat{\sigma}} m_i m_j g(x_i, x_j) &\approx \sum_{j \in \hat{\sigma}} m_i m_j \tilde{g}_\sigma(x_i, x_j) = \sum_{j \in \hat{\sigma}} m_i m_j \sum_{\nu=1}^{k} g(x_i, \xi_\nu) \mathcal{L}_\nu(x_j) \\ &= \sum_{\nu=1}^{k} m_i g(x_i, \xi_\nu) \sum_{j \in \hat{\sigma}} m_j \mathcal{L}_\nu(x_j) \qquad \text{für alle } i \in \hat{\tau}. \end{aligned}$$

Diese Darstellung hat den Vorteil, dass sie sich besonders effizient auswerten lässt: Mit den Hilfsgrößen

$$h_\nu := \sum_{j \in \hat{\sigma}} m_j \mathcal{L}_\nu(x_j) \qquad \text{für alle } \nu \in \{1, \ldots, k\} \tag{1.3}$$

erhalten wir

$$\sum_{j \in \hat{\sigma}} m_i m_j g(x_i, x_j) \approx \sum_{\nu=1}^{k} m_i g(x_i, \xi_\nu) h_\nu \qquad \text{für alle } i \in \hat{\tau}, \tag{1.4}$$

und diese Näherung lässt sich sehr effizient auswerten: Wenn $h_1, \ldots, h_k$ vorliegen, erfordert die Auswertung der Gleichung (1.4) für *alle* $i \in \hat{\tau}$ lediglich die Berechnung von $j\#\hat{\tau}$ Summanden.

Die Berechnung der Hilfsgrößen $h_1, \ldots, h_\nu$ aus (1.3) erfordert die Bestimmung von lediglich $k\#\hat{\sigma}$ Summanden, wobei das Auswerten des Lagrange-Polynoms höchstens eine zu k proportionale Anzahl von Operationen erfordert, so dass der Aufwand für die Berechnung der Hilfsgrößen im schlimmsten Fall proportional zu $k^2\#\hat{\sigma}$ ist.

Im Vergleich zu der direkten Berechnung der Summe, bei der $(\#\hat{\tau})(\#\hat{\sigma})$ Summanden auszuwerten wären, kann die Näherung also sehr viel effizienter sein, falls k nicht allzu groß gewählt werden muss, falls also bereits wenige Interpolationspunkte ausreichen, um g zu approximieren.

Bemerkung 1.1 (Ersatzmassen) *Der Vergleich zwischen (1.4) und (1.1) legt es nahe, die Werte $h_1, \ldots, h_k$ als „Ersatzmassen“ an den Interpolationspunkten $\xi_1, \ldots, \xi_k$ zu interpretieren, die näherungsweise dasselbe Gravitationsfeld wie die Gesamtheit der zu der Indexmenge $\hat{\sigma}$ gehörenden Massen erzeugen.*

1.3 Partitionierung

Wir haben gesehen, dass sich auf hinreichend weit voneinander getrennten Teilmengen τ und σ die Funktion g durch eine Approximation $\tilde{g}_\sigma$ ersetzen lässt, mit deren Hilfe sich Teilsummen

$$\sum_{j \in \hat{\sigma}} m_i m_j \tilde{g}_\sigma(x_i, x_j) \qquad \text{für alle } i \in \hat{\tau}$$

besonders effizient auswerten lassen. Falls wir nun sämtliche wirkenden Kräfte in dieser Weise approximieren wollen, müssen wir versuchen, alle auftretenden Kombinationen von Punkten x_i und x_j abzudecken.

Wir beschränken uns hier auf eine besonders einfache rekursive Konstruktion: Für die Gebiete τ und σ lassen wir lediglich achsenparallele Quader zu, verlangen also

$$\tau = [a_1, b_1) \times [a_2, b_2) \times [a_3, b_3), \qquad \sigma = [c_1, d_1) \times [c_2, d_2) \times [c_3, d_3)$$

für geeignete Koeffizienten. Damit die Interpolationspolynome schnell konvergieren, muss der Durchmesser

$$\operatorname{diam}(\sigma) := \sqrt{(d_1 - c_1)^2 + (d_2 - c_2)^2 + (d_3 - c_3)^2}$$

des Gebiets σ kleiner als der Abstand

$$\operatorname{dist}(\tau,\sigma) := \sqrt{\delta_1^2+\delta_2^2+\delta_3^2}, \qquad \delta_\iota := \max\{c_\iota - b_\iota, a_\iota - d_\iota, 0\}$$

der beiden Gebiete sein, wir dürfen also unsere Approximation nur dann verwenden, wenn die *Zulässigkeitsbedingung*

$$\operatorname{diam}(\sigma) \leq \operatorname{dist}(\tau,\sigma) \tag{1.5}$$

erfüllt ist.

Falls die Bedingung nicht erfüllt ist, müssen wir versuchen, $\tau \times \sigma$ in Teilgebiete zu zerlegen, auf denen sie erfüllt sein könnte. Eine einfache Möglichkeit besteht darin, eine Koordinatenrichtung $\iota \in \{1,2,3\}$ zu wählen, den Mittelpunkt

$$m_\iota := (b_\iota + a_\iota)/2$$

zu berechnen und τ in

$$\tau_1 := \begin{cases} [a_1,m_1)\times[a_2,b_2)\times[a_3,b_3) & \text{falls } \iota = 1,\\ [a_1,b_1)\times[a_2,m_2)\times[a_3,b_3) & \text{falls } \iota = 2,\\ [a_1,b_1)\times[a_2,b_2)\times[a_3,m_3) & \text{falls } \iota = 3 \end{cases} \qquad \text{und}$$

$$\tau_2 := \begin{cases} [m_1,b_1)\times[a_2,b_2)\times[a_3,b_3) & \text{falls } \iota = 1,\\ [a_1,b_1)\times[m_2,b_2)\times[a_3,b_3) & \text{falls } \iota = 2,\\ [a_1,b_1)\times[a_2,b_2)\times[m_3,b_3) & \text{falls } \iota = 3 \end{cases}$$

mit $\tau = \tau_1 \cup \tau_2$ zu zerlegen. Entsprechend können wir aus σ die Teilgebiete σ_1 und σ_2 mit $\sigma = \sigma_1 \cup \sigma_2$ konstruieren.

Insbesondere erhalten wir so

$$\tau\times\sigma = \tau_1\times\sigma_1 \cup \tau_1\times\sigma_2 \cup \tau_2\times\sigma_1 \cup \tau_2\times\sigma_2,$$

können also anstelle des Gesamtgebiets $\tau\times\sigma$ auch die vier Teilgebiete rekursiv behandeln, ohne dadurch etwas zu verlieren.

Theoretisch können wir so fortfahren, bis die Teilgebiete nur noch ein Punktpaar enthalten, und in diesem Fall lässt sich die zwischen diesen beiden Punkten wirkende Kraft direkt berechnen. Effizienter ist es allerdings, das Unterteilen der Teilgebiete abzubrechen, sobald sie so klein geworden sind, dass der Einsatz der Näherung aufwendiger ist als die direkte Berechnung. Ein sinnvoller Ansatz wäre beispielsweise, nicht weiter zu unterteilen, wenn

$$\#\hat{\tau} \leq k \qquad \text{oder} \qquad \#\hat{\sigma} \leq k$$

gelten, denn dann ist eine Summe über k Terme potentiell aufwendiger zu berechnen als eine über $\hat{\tau}$ oder $\hat{\sigma}$.

Der vollständige Algorithmus ist in Abbildung 1.1 zusammengefasst. Er berechnet jeweils die Kräfte, die sich aus den Interaktionen der Massen zu den Indexmengen $\hat{\tau}$ und

procedure approx_gravitation(τ, $\hat{\tau}$, σ, $\hat{\sigma}$, **m**, **x**, **var F**);
if $\#\hat{\tau} \leq k$ oder $\#\hat{\sigma} \leq k$ **then** {Direkte Berechnung}
  for $i \in \hat{\tau},\ j \in \hat{\sigma} \setminus \{i\}$ **do**
    $F_i \leftarrow F_i + \gamma m_i m_j \frac{x_j - x_i}{\|x_j - x_i\|_2^3}$
  end for
else if $\operatorname{diam}(\sigma) \leq \operatorname{dist}(\tau, \sigma)$ **then** {Approximation}
  Wähle Interpolationspunkte $\xi_1, \ldots, \xi_k$ und Lagrange-Polynome $\mathcal{L}_1, \ldots, \mathcal{L}_k$ für σ
  for $\nu \in \{1, \ldots, k\}$ **do**
    $h_\nu \leftarrow 0$;
    for $j \in \hat{\sigma}$ **do**
      $h_\nu \leftarrow h_\nu + m_j \mathcal{L}_\nu(x_j)$
    end for;
    for $i \in \hat{\tau}$ **do**
      $F_i \leftarrow F_i + m_i g(x_i, \xi_\nu) h_\nu$
    end for
  end for
else {Unterteilung und Rekursion}
  Wähle ein $\iota \in \{1, 2, 3\}$;
  $[a_1, b_1) \times [a_2, b_2) \times [a_3, b_3) = \tau$; $\quad m_\iota \leftarrow (b_\iota + a_\iota)/2$;
  $\tau_1 \leftarrow \{x \in \tau \ :\ x_\iota < m_\iota\}$; $\quad \hat{\tau}_1 \leftarrow \{i \in \hat{\tau} \ :\ x_{i,\iota} < m_\iota\}$;
  $\tau_2 \leftarrow \{x \in \tau \ :\ x_\iota \geq m_\iota\}$; $\quad \hat{\tau}_2 \leftarrow \{i \in \hat{\tau} \ :\ x_{i,\iota} \geq m_\iota\}$;
  $[c_1, d_1) \times [c_2, d_2) \times [c_3, d_3) = \sigma$; $\quad m_\iota \leftarrow (d_\iota + c_\iota)/2$;
  $\sigma_1 \leftarrow \{x \in \sigma \ :\ x_\iota < m_\iota\}$; $\quad \hat{\sigma}_1 \leftarrow \{i \in \hat{\sigma} \ :\ x_{i,\iota} < m_\iota\}$;
  $\sigma_2 \leftarrow \{x \in \sigma \ :\ x_\iota \geq m_\iota\}$; $\quad \hat{\sigma}_2 \leftarrow \{i \in \hat{\sigma} \ :\ x_{i,\iota} \geq m_\iota\}$;
  approx_gravitation(τ_1, $\hat{\tau}_1$, σ_1, $\hat{\sigma}_1$, **m**, **x**, **F**);
  approx_gravitation(τ_1, $\hat{\tau}_1$, σ_2, $\hat{\sigma}_2$, **m**, **x**, **F**);
  approx_gravitation(τ_2, $\hat{\tau}_2$, σ_1, $\hat{\sigma}_1$, **m**, **x**, **F**);
  approx_gravitation(τ_2, $\hat{\tau}_2$, σ_2, $\hat{\sigma}_2$, **m**, **x**, **F**)
end if

Abbildung 1.1: Schnelle approximative Auswertung aller wirkenden Gravitationskräfte.

$\hat{\sigma}$ ergeben, und addiert sie zu **F** hinzu. Wir müssen den Algorithmus also mit $\mathbf{F} = \mathbf{0}$ und $\hat{\tau} = \hat{\sigma} = \{1, \ldots, n\}$ aufrufen, um alle wirkenden Kräfte zu approximieren. Einen zu diesen Indexmengen passenden Quader können wir durch

$$a_\iota := \min\{x_{i,\iota} \ :\ i \in \{1, \ldots, n\}\}, \qquad b_\iota := \max\{x_{i,\iota} \ :\ i \in \{1, \ldots, n\}\} + \epsilon,$$
$$\tau = \sigma = [a_1, b_1) \times [a_2, b_2) \times [a_3, b_3)$$

mit einem kleinen $\epsilon \in \mathbb{R}_{>0}$ einfach bestimmen, denn diese Wahl stellt sicher, dass die Bedingungen (1.2) erfüllt sind. Da der Algorithmus so geschrieben ist, dass es keine Rolle spielt, ob die Quader abgeschlossen oder halb-offen sind, können wir in einer praktischen Implementierung auch einfach $\epsilon = 0$ setzen.

1.4 Eigenschaften und Ausblick

Wir haben bereits gesehen, dass für ein zulässiges Teilgebiet $\tau \times \sigma$ die wirkenden Kräfte mit einem Aufwand von $k(\#\hat{\tau} + \#\hat{\sigma})$ berechnet werden können. Diese Beobachtung alleine hilft uns allerdings nicht viel weiter, falls sehr viele solcher Teilgebiete auftreten.

Glücklicherweise lässt sich das einfach ausschließen: Unser Algorithmus untersucht ein Teilgebiet $\tau \times \sigma$ nur dann, wenn es entweder das Gesamtgebiet ist oder aber aus einem größeren Teilgebiet $\tau_0 \times \sigma_0$ durch Unterteilung entstanden ist. Der Algorithmus unterteilt Gebiete aber nur dann, wenn sie nicht zulässig sind, also muss

$$\operatorname{diam}(\sigma_0) > \operatorname{dist}(\tau_0, \sigma_0)$$

gelten, so dass σ_0 relativ nahe an τ_0 liegen muss. Damit muss auch $\sigma \subseteq \sigma_0$ relativ nahe an $\tau \subseteq \tau_0$ liegen, so dass unser Algorithmus immer nur mit Teilgebieten arbeitet, die nahe, aber nicht zu nahe, aneinander liegen. Mit Hilfe dieser Beobachtung lässt sich nachweisen (siehe Kapitel 2, für unseren Fall insbesondere Abschnitt 2.6), dass der Rechenaufwand unseres Verfahrens proportional zu $nk(\log_2 n + 1)$ ist, also ist die Anzahl der Rechenoperationen „fast proportional“ zu n.

In Kapitel 3 wird untersucht, wie der Approximationsfehler von dem Parameter k abhängt. Mit Hilfe einer Fehlerabschätzung für den Interpolationsfehler lässt sich zeigen, dass wir erwarten dürfen, dass sich der Fehler wie $e^{-\alpha k^{1/3}}$ mit einem $\alpha > 0$ verhält, also exponentiell konvergiert.

Unter bestimmten Bedingungen kann der Aufwand für die Interpolation zu hoch sein, etwa falls Funktionen mit besonderen Eigenschaften approximiert werden müssen. In diesem Fall ist es auch möglich, eine Approximation mit Hilfe einer geeignet modifizierten LR-Zerlegung zu gewinnen. Unter bestimmten Umständen kann eine derartige Approximation sehr effizient sein, unter anderen Umständen kann sie auch fehlschlagen (siehe Kapitel 4).

Für einzelne Teilgebiete $\tau \times \sigma$ lässt sich der durch die Approximation eingeführte Fehler in der Regel relativ einfach abschätzen. Mit Hilfe geeigneter algebraischer Argumente (siehe Kapitel 5) ist es möglich, aus diesen Abschätzungen Aussagen über die Größe des Gesamtfehlers zu gewinnen. Umgekehrt lässt sich in dieser Weise präzise festlegen, wie genau die Approximationen auf den Teilgebieten sein müssen, um eine gewisse Genauigkeit des gesamten Algorithmus sicher zu stellen.

Die diskutierten Techniken lassen sich auch verwenden, um vollbesetzte Matrizen zu behandeln, die beispielsweise bei arithmetischen Operationen (siehe Kapitel 6) wie der Inversion (siehe Kapitel 7) entstehen. Durch geschickt konstruierte Algorithmen erhält man Verfahren, die Berechnungen mit vollbesetzten Matrizen mit fast linearem Aufwand in der Anzahl der Unbekannten durchführen können.

Eine weitere Verbesserung der Effizienz erreichen Techniken, bei denen durch eine kleine Modifikation der Entwicklung der Funktion g und die Ausnutzung bestimmter Eigenschaften der Taylor- oder Interpolationspolynome ein zu nk proportionaler Rechenaufwand erreicht wird (siehe Kapitel 8). Ein in n linear wachsender Aufwand ist sehr nahe am theoretisch bestmöglich Machbaren.

2 Struktur hierarchischer Matrizen

Bisher haben wir die Approximation mit Hilfe geometrischer Argumente konstruiert: Verwendet wurden Teilgebiete eines Definitionsgebiets, die geeignet unterteilt wurden und deren Unterteilung Teilmengen der Menge der zu behandelnden Punkte definierten.

Wir sind an einer Verallgemeinerung interessiert: Statt wechselwirkender Punktmassen wollen wir eine vollbesetzte Matrix approximieren. An die Stelle der Näherungen auf Teilgebieten treten dann Teilmatrizen niedrigen Rangs, die Gesamtmatrix wird ersetzt durch eine *hierarchische Matrix* (siehe [25, 26]).

Um allgemeine Matrizen behandeln zu können, müssen wir auch allgemeine Indexmengen statt der bisher untersuchten Punktmengen behandeln können. Einfache Ansätze für regelmäßige Strukturen ermöglichen es zwar, die Komplexität bestimmter Algorithmen präzise zu bestimmen (vgl. die grundlegende Arbeit [25]), doch schon die Verallgemeinerung auf zweidimensionale äquidistante Gitter (vgl. [27]) ist sehr aufwendig und nicht allzu handlich.

Wesentlich flexibler und eleganter ist der in [19, 21] entwickelte Zugang: Die Zeilen- und Spaltenindizes einer Matrix werden zu *Clustern* zusammengefasst, die in einem *Clusterbaum* hierarchisch geordnet sind. Aus diesen Clustern wird dann mit Hilfe einer *Zulässigkeitsbedingung* eine Zerlegung der Matrix in Teilmatrizen konstruiert, die mit niedrigem Rang approximiert werden können.

In diesem Kapitel führen wir die grundlegenden Begriffe ein und demonstrieren, wie sich auf ihrer Grundlage einige zentrale Aussagen über allgemeine hierarchische Matrizen beweisen lassen. Dabei interessieren wir uns für allgemeine Matrizen $\mathbf{X} \in \mathbb{K}^{\mathcal{I}\times\mathcal{J}}$ mit einer endlichen Zeilenindexmenge $\mathcal{I}$ und einer endlichen Spaltenindexmenge $\mathcal{J}$.

2.1 Cluster und Clusterbäume

Wie wir bereits gesehen haben, besteht eine zentrale Idee unserer Approximation darin, geeignete Teilgebiete oder Teilmatrizen in effizienter Form zu approximieren. Diese Teilmatrizen haben die Form $\mathbf{X}|_{\hat{t}\times\hat{s}} \in \mathbb{K}^{\hat{t}\times\hat{s}}$ für Teilmengen von Zeilenindizes $\hat{t} \subseteq \mathcal{I}$ und Spaltenindizes $\hat{s} \subseteq \mathcal{J}$. Im Interesse der Effizienz wählen wir diese Teilmengen nicht beliebig, sondern verwenden eine Baumstruktur, die es uns ermöglicht, wichtige Algorithmen sehr elegant zu formulieren.

Definition 2.1 (Baum) *Sei $\mathcal{T} = (V, S, r)$ ein Tripel aus einer endlichen Menge V, einer Abbildung $S : V \to \mathcal{P}(V)$, die jedem Element aus V eine Teilmenge von V zuordnet, und einem Element $r \in V$.*

Ein Tupel $t_0, t_1, t_2, \ldots, t_\ell \in V$ mit

$$t_i \in S(t_{i-1}) \qquad \text{für alle } i \in \{1, \ldots, \ell\}$$

bezeichnen wir als Pfad *von* t_0 *nach* t_ℓ.

Das Tripel $\mathcal{T}$ *bezeichnen wir als* Baum*, falls für jedes* $t \in V$ *genau ein Pfad von* r *nach* t *existiert. In diesem Fall nennen wir* r *die* Wurzel *von* $\mathcal{T}$*, jedes* $t \in V$ *nennen wir* Knoten *des Baums, und* $S(t)$ *nennen wir* Menge der Söhne *eines* $t \in V$*. Falls* $w \in S(t)$ *gilt, bezeichnen wir* w *entsprechend als* Sohn *von* t.

Die Arbeit mit mehreren Bäumen erleichtern wir uns, indem wir $t \in V$ *durch* $t \in \mathcal{T}$ *abkürzen, die Wurzel mit* $r = \operatorname{root}(\mathcal{T})$ *bezeichnen und die Söhne mit* $\operatorname{sons}(t, \mathcal{T}) = S(t)$ *für alle* $t \in V$*. Falls Verwechslungen ausgeschlossen sind, kürzen wir letzteres auch mit* $\operatorname{sons}(t)$ *ab.*

Eine wichtige Folgerung aus der Definition des Baums ist, dass kein Knoten, der auf dem Pfad von der Wurzel zu einem Knoten $t \in \mathcal{T}$ auftritt, ein Sohn von t sein kann, denn ansonsten ließe sich der Pfad verlängern und so die geforderte Eindeutigkeit verletzen.

Die *Menge der Nachfahren* eines Knotens $t \in \mathcal{T}$ ist durch

$$\operatorname{sons}^*(t) = \begin{cases} \{t\} \cup \bigcup_{t' \in \operatorname{sons}(t)} \operatorname{sons}^*(t') & \text{falls } \operatorname{sons}(t) \neq \emptyset, \\ \{t\} & \text{ansonsten} \end{cases} \tag{2.1}$$

induktiv definiert. Die *Menge der Vorfahren* eines Knotens $t \in \mathcal{T}$ ist korrespondierend durch

$$\operatorname{pred}(t) = \{t^* \in \mathcal{T} \ : \ t \in \operatorname{sons}^*(t^*)\} \tag{2.2}$$

definiert. Es ist zu beachten, dass diese Definitionen $t \in \operatorname{sons}^*(t)$ und $t \in \operatorname{pred}(t)$ beinhalten.

Falls $\operatorname{sons}(t) = \emptyset$ gilt, nennen wir t ein *Blatt* des Baums. Die Menge aller Blätter eines Baums $\mathcal{T}$ bezeichnen wir mit

$$\mathcal{L} = \{t \in \mathcal{T} \ : \ \operatorname{sons}(t) = \emptyset\}.$$

Für jeden Knoten $t \in \mathcal{T}$ kann höchstens ein $t^+ \in \mathcal{T}$ mit $t \in \operatorname{sons}(t^+)$ existieren. Dieses t^+ nennen wir den *Vater* von t. Es gibt genau einen Knoten $t \in \mathcal{T}$, der keinen Vater besitzt, und zwar gerade seine Wurzel $\operatorname{root}(\mathcal{T})$.

Für uns von besonderem Interesse sind *beschriftete Bäume*, bei denen jedem Knoten $t \in \mathcal{T}$ ein Element $\hat{t}$ einer geeigneten Menge zugeordnet ist. $\hat{t}$ nennen wir dann die *Beschriftung* des Knotens t.

Mit Hilfe beschrifteter Bäume lässt sich nun unser Ansatz zur Unterteilung der Indexmengen formal präzise beschreiben:

Definition 2.2 (Clusterbaum) *Ein beschrifteter Baum* $\mathcal{T}$ *ist ein* Clusterbaum *für die Indexmenge* $\mathcal{I}$*, falls die folgenden Bedingungen erfüllt sind:*

- *Falls* $r \in \mathcal{T}$ *die Wurzel des Baums ist, gilt* $\hat{r} = \mathcal{I}$.
- *Falls* $t \in \mathcal{T}$ *kein Blatt ist, gilt*

$$\bigcup_{t' \in \operatorname{sons}(t)} \hat{t}' = \hat{t}. \tag{2.3}$$

- *Falls* $t', s' \in \operatorname{sons}(t)$ *mit* $t' \neq s'$ *existieren, gilt*

$$\hat{t}' \cap \hat{s}' = \emptyset. \tag{2.4}$$

In diesem Fall verwenden wir in der Regel die Notation $\mathcal{T}_\mathcal{I}$ *für* $\mathcal{T}$*, um auf die verwendete Indexmenge hinzuweisen, und bezeichnen die Knoten* $t \in \mathcal{T}_\mathcal{I}$ *als* Cluster*.*

Die Menge aller Blätter eines Clusterbaums $\mathcal{T}_\mathcal{I}$ *bezeichnen wir mit* $\mathcal{L}_\mathcal{I}$*.*

Ein Clusterbaum ist also ein beschrifteter Baum, bei dem die Beschriftungen Teilmengen der Indexmenge $\mathcal{I}$ sind und bei dem die Beschriftungen der Söhne eines Knotens eine disjunkte Partition der Beschriftungen des Vaters beschreiben. In der Praxis vernachlässigt man die Unterscheidung zwischen einem Cluster und seiner Beschriftung, solange dabei Missverständnisse ausgeschlossen sind, man sagt also „i ist in t enthalten“, wenn $i \in \hat{t}$ gemeint ist, oder man sagt, dass die Söhne eines Clusters eine disjunkte Zerlegung ihres Vaters sind, wenn gemeint ist, dass ihre Beschriftungen eine disjunkte Zerlegung der Beschriftung des Vaters sind.

Für die praktische Konstruktion eines Clusterbaums bietet es sich an, von der Definition 2.2 auszugehen: Die erste Bedingung an einen Clusterbaum legt fest, wie seine Wurzel auszusehen hat, die zweite und dritte Bedingung bestimmen die Beziehung zwischen einem Vater und seinen Söhnen.

Ein erfolgversprechender Ansatz besteht deshalb darin, eine Regel für die Zerlegung eines gegebenen Clusters in mehrere Söhne zu definieren und diese Zerlegung ausgehend von der bekannten Wurzel rekursiv so lange anzuwenden, bis die resultierenden Cluster klein genug sind.

Zur Vereinfachung gehen wir davon aus, dass jedem Index ein Punkt in einem d-dimensionalen Raum zugeordnet ist, denn dann können wir geometrische Konstruktionen aufgrund dieser Punkte verwenden.

Definition 2.3 (Charakteristische Punkte) *Sei* $\mathcal{I}$ *eine Indexmenge und sei* $d \in \mathbb{N}$*. Eine Familie* $(x_i)_{i\in\mathcal{I}}$ *in* $\mathbb{R}^d$ *nennen wir eine* Familie charakteristischer Punkte *für die Indexmenge* $\mathcal{I}$*.*

In unserem Modellproblem bieten sich die Schwerpunkte der Sonnen als charakteristische Punkte an, bei allgemeineren Anwendungen lassen sich oft in ähnlicher Weise „typische“ geometrische Orte zu jedem Index finden.

Wenn eine Familie charakteristischer Punkte für $\mathcal{I}$ gegeben ist, können wir jedem Cluster $t \in \mathcal{T}_\mathcal{I}$ eine Menge

$$\{\mathbf{x}_i \ : \ i \in \hat{t}\} \subseteq \mathbb{R}^d$$

von Punkten zuordnen, und eine Zerlegung dieser Menge in Teilmengen induziert eine Zerlegung des Clusters in Söhne.

Im eindimensionalen Fall bietet es sich an, ein Intervall $[a, b]$ zu finden, das die Punkte $\mathbf{x}_i$ für alle $i \in \hat{t}$ enthält, und dieses Intervall in der Mitte zu unterteilen, um zwei möglichst kleine Teilintervalle zu erhalten. Die Indizes, die zu Punkten aus dem linken Teilintervall

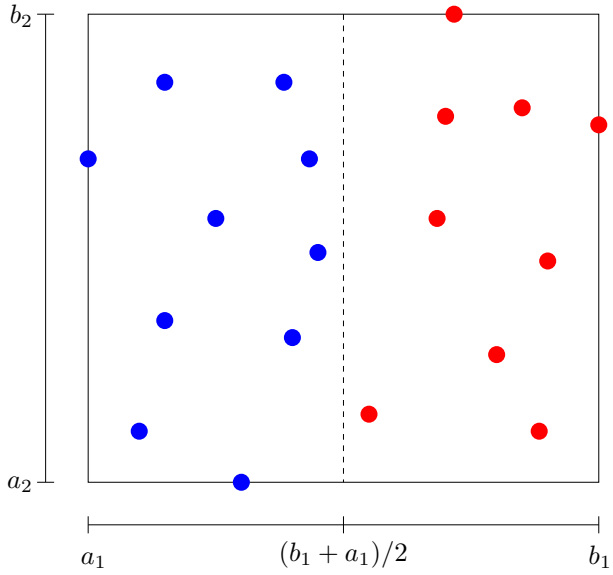

Abbildung 2.1: Geometrische Zerlegung eines Clusters durch Bisektion in der ersten Koordinatenrichtung eines einschließenden achsenparallelen Quaders

gehören, werden dann dem ersten Sohn des Clusters zugeordnet, die restlichen Indizes muss der zweite Sohn aufnehmen, damit Definition 2.2 erfüllt werden kann.

Dieser Ansatz lässt sich auf den mehrdimensionalen Fall übertragen, indem wir einfach eine Koordinatenrichtung $\iota \in \{1, \ldots, d\}$ auswählen, ein Intervall $[a_\iota, b_\iota]$ mit

$$x_{i,\iota} \in [a_\iota, b_\iota] \qquad \text{für alle } i \in \hat{t}$$

konstruieren und die Punkte entsprechend ihrer Zugehörigkeit zur linken oder rechten Hälfte des Intervalls sortieren:

$$\hat{t}_1 := \{i \in \hat{t} \ : \ x_{i,\iota} < (b_\iota + a_\iota)/2\},$$
$$\hat{t}_2 := \{i \in \hat{t} \ : \ x_{i,\iota} \geq (b_\iota + a_\iota)/2\} = \hat{t} \setminus \hat{t}_1$$

(vgl. Abbildung 2.1). Offenbar gilt nun $\hat{t} = \hat{t}_1 \dot{\cup} \hat{t}_2$, also eignen sich die so konstruierten t_1, t_2 als Söhne des Clusters t in einem Clusterbaum.

Bemerkung 2.4 (Wahl der Richtung) *Für die Auswahl einer geeigneten Richtung $\iota \in \{1, \ldots, d\}$, senkrecht zu der der Cluster unterteilt werden soll, sind zwei Ansätze üblich: Wenn wir an regelmäßigen Strukturen interessiert sind, beispielsweise für theoretische Untersuchungen, ist es eine gute Idee, ι zyklisch zu wählen, also in den Sohnclustern die Richtung $(\iota - 1) \mod d + 1$ zu verwenden. Dadurch ist sichergestellt, dass alle Cluster auf einer Stufe des Baums in derselben Richtung unterteilt werden. Diese Strategie bezeichnen wir als* regelmäßige *Wahl der Richtung.*

procedure cluster_adaptive($r_{\max}$, $\ell_{\max}$, **var** t);
if $\#\hat{t} \leq r_{\max}$ oder $\ell_{\max} = 0$ **then**
  $\text{sons}(t) = \emptyset$
else
  for $\iota \in \{1, \ldots, d\}$ **do**
    $a_\iota \leftarrow \min\{x_{i,\iota} \ : \ i \in \hat{t}\}$; $b_\iota \leftarrow \max\{x_{i,\iota} \ : \ i \in \hat{t}\}$
  end for;
  $\iota \leftarrow \text{argmax}\{b_k - a_k \ : \ k \in \{1, \ldots, d\}\}$;
  $c \leftarrow (b_\iota + a_\iota)/2$;
  $\hat{t}_1 \leftarrow \{i \in \hat{t} \ : \ x_{i,\iota} < c\}$; $\hat{t}_2 \leftarrow \{i \in \hat{t} \ : \ x_{i,\iota} \geq c\}$
  if $\hat{t}_1 \neq \emptyset$ **then**
    $\text{sons}(t) = \{t_1, t_2\}$;
    cluster_adaptive($r_{\max}$, $\ell_{\max} - 1$, t_1);
    cluster_adaptive($r_{\max}$, $\ell_{\max} - 1$, t_2)
  else
    $\text{sons}(t) = \emptyset$, Cluster enthält nur einen Punkt
  end if
end if

Abbildung 2.2: Konstruktion eines adaptiven Clusterbaums.

In der Praxis ist es häufig besser, die Richtung ι so zu wählen, dass der Durchmesser der Cluster möglichst schnell schrumpft, also so, dass der Durchmesser

$$\max\{x_{i,\iota} - x_{k,\iota} \ : \ i, k \in \hat{t}\}$$

maximal ist und deshalb seine Halbierung den größten Gewinn bringt. In diesem Fall sprechen wir von einer adaptiven *Wahl der Richtung.*

Bemerkung 2.5 (Wahl der Intervalle) *Die Intervalle $[a_\iota, b_\iota]$ können ebenfalls regelmäßig oder adaptiv konstruiert werden. Bei der* regelmäßigen *Strategie gehen wir so vor, dass wir davon ausgehen, dass zu jedem Cluster t ein achsenparalleler Quader*

$$Q_t = [a_1, b_1] \times \ldots \times [a_d, b_d]$$

gegeben ist, der alle zu t gehörenden Punkte enthält, also

$$\mathbf{x}_i \in Q_t \qquad \text{für alle } i \in \hat{t}$$

erfüllt. Für die Söhne t_1 und t_2 des Clusters t können wir dann die Quader

$$Q_{t_1} = [a_1, b_1] \times \ldots \times [a_\iota, (b_\iota + a_\iota)/2] \times \ldots [a_d, b_d],$$
$$Q_{t_2} = [a_1, b_1] \times \ldots \times [(b_\iota + a_\iota)/2, b_\iota] \times \ldots [a_d, b_d]$$

für die gewählte Richtung ι verwenden. Bei diesem Zugang ist sichergestellt, dass alle Quader auf einer Stufe des Clusterbaums bis auf Verschiebungen identisch sind, und

```
procedure cluster_regular(r_max, ℓ_max, ι, var a, b, t);
if #t̂ ≤ r_max  oder  ℓ_max = 0 then
  sons(t) = ∅
else
  c ← (b_ι + a_ι)/2;
  t̂_1 ← {i ∈ t̂ : x_{i,ι} < c};    t̂_2 ← {i ∈ t̂ : x_{i,ι} ≥ c};
  a^(1) ← a;   b^(1) ← b;   b_ι^(1) ← c;
  a^(2) ← a;   b^(2) ← b;   a_ι^(2) ← c;
  if t̂_1 ≠ ∅ then
    if t̂_2 ≠ ∅ then
      sons(t) = {t_1, t_2};
      cluster_regular(r_max, ℓ_max − 1, (ι mod d) + 1, a^(1), b^(1), t_1);
      cluster_regular(r_max, ℓ_max − 1, (ι mod d) + 1, a^(2), b^(2), t_2)
    else
      sons(t) = {t_1};
      cluster_regular(r_max, ℓ_max − 1, (ι mod d) + 1, a^(1), b^(1), t_1)
    end if
  else
    if t̂_2 ≠ ∅ then
      sons(t) = {t_2};
      cluster_regular(r_max, ℓ_max − 1, (ι mod d) + 1, a^(2), b^(2), t_2)
    else
      Problem, leerer Cluster t
    end if
  end if
end if
```

Abbildung 2.3: Konstruktion eines regelmäßigen Clusterbaums.

diese Eigenschaft lässt sich sowohl für theoretische Betrachtungen als auch in der Praxis ausnutzen.

Bei einer adaptiven *Strategie dagegen besteht das Ziel wieder darin, möglichst schnell den Durchmesser zu reduzieren, deshalb bestimmt man in diesem Fall das Intervall* $[a_\iota, b_\iota]$ *so, dass es die minimal mögliche Größe besitzt:*

$$a_\iota := \min\{x_{i,\iota} \ : \ i \in \hat{t}\}, \qquad b_\iota := \max\{x_{i,\iota} \ : \ i \in \hat{t}\}.$$

Für kompliziertere Geometrien führt dieser Ansatz in Kombination mit der adaptiven Wahl der Richtung ι *dazu, dass die Durchmesser der Cluster so schnell wie möglich schrumpfen.*

Bemerkung 2.6 (Abbruchkriterium) *Die Unterteilung der Cluster in Teilcluster verwenden wir nur so lange, wie die Teilcluster „zu groß“ sind. Vom algebraischen Standpunkt aus gesehen sind sie zu groß, solange sie zuviele Indizes enthalten, wobei*

„zuviele“ unter anderem vom zu erwartenden Rang der Matrixapproximationen abhängt: Es lohnt sich nicht, einer 2×3*-Matrix eine Approximation mit* 17 *Termen zuzuordnen. Also sollte man das Unterteilen beenden, sobald die Mächtigkeiten der Söhne dabei den erwarteten Rang unterschreiten würden.*

Da praktisch alle modernen Prozessoren mit Cache-Speichern ausgestattet sind, mit deren Inhalten sie besonders schnell rechnen können, kann es sich lohnen, noch etwas größere Cluster zuzulassen, um die Rechenzeit zu reduzieren.

Im allgemeinen Fall werden nicht einzelne Punkte $\mathbf{x}_i$ zu den Indizes $i \in \mathcal{I}$ gehören, sondern Mengen $\Omega_i \subseteq \mathbb{R}^d$. Beispielsweise sind die bei der Simulation des Gravitationsfeldes einer Galaxis auftretenden Sonnen keine Punkte, sondern eher Kugeln, und bei der Behandlung von Differentialgleichungen mit dem Galerkin-Verfahren müssen wir mit Polyedern rechnen. Unter diesen Bedingungen verwenden wir einen einfachen Trick: Wir wählen für jedes Ω_i ein $\mathbf{x}_i \in \Omega_i$ und konstruieren den Clusterbaum wie zuvor. Da in der Regel die Durchmesser der Mengen Ω_i sehr klein im Vergleich zum Durchmesser des Gesamtgebiets sind, entstehen durch diese Vereinfachung keine allzu großen Probleme.

2.2 Blöcke und Blockbäume

In Gestalt der Clusterbäume steht uns eine Auswahl von Teilmengen der Indexmenge zur Verfügung, mit deren Hilfe wir nach Teilmatrizen einer gegebenen Matrix suchen können, die sich durch Matrizen niedrigen Rangs approximieren lassen: Falls wir eine Matrix $\mathbf{X} \in \mathbb{K}^{\mathcal{I} \times \mathcal{J}}$ behandeln wollen, konstruieren wir Clusterbäume $\mathcal{T}_\mathcal{I}$ und $\mathcal{T}_\mathcal{J}$ für ihre Zeilen- und Spaltenindizes und suchen dann Teilmatrizen der Form $\mathbf{X}|_{\hat{t} \times \hat{s}}$ mit $t \in \mathcal{T}_\mathcal{I}$ und $s \in \mathcal{T}_\mathcal{J}$.

Alle möglichen Zerlegungen der Matrix $\mathbf{X}$ in derartige Teilmatrizen zu untersuchen hätte einen zu hohen Aufwand: Bei $n = \#\mathcal{I}$ Zeilen und $m = \#\mathcal{J}$ Spalten müssen wir mit $\sim n$ Zeilenclustern und $\sim m$ Spaltenclustern rechnen, so dass $\sim nm$ Kombinationen zu prüfen wären.

Glücklicherweise lässt sich dieser Aufwand vermeiden, indem wir wieder rekursiv vorgehen: Für ein gegebenes Paar $(t, s) \in \mathcal{T}_\mathcal{I} \times \mathcal{T}_\mathcal{J}$ prüfen wir, ob wir auf eine Approximation niedrigen Rangs hoffen dürfen. Falls ja, approximieren wir $\mathbf{X}_{\hat{t} \times \hat{s}}$. Anderenfalls wiederholen wir den Test für die Paare $(t', s') \in \operatorname{sons}(t) \times \operatorname{sons}(s)$ der Söhne von t und s.

Dieser Vorgehensweise entspricht ein Baum, dessen Knoten aus Paaren von Clustern besteht:

Definition 2.7 (Blockbaum) *Ein beschrifteter Baum* $\mathcal{T}$ *ist ein* Blockbaum *für die Clusterbäume* $\mathcal{T}_\mathcal{I}$ *und* $\mathcal{T}_\mathcal{J}$*, falls die folgenden Bedingungen erfüllt sind:*

- *Für jeden Knoten* $b \in \mathcal{T}$ *existieren Cluster* $t \in \mathcal{T}_\mathcal{I}$ *und* $s \in \mathcal{T}_\mathcal{J}$ *mit* $b = (t, s)$.
- *Für die Wurzel* r *des Baums* $\mathcal{T}$ *gilt* $r = (\operatorname{root}(\mathcal{T}_\mathcal{I}), \operatorname{root}(\mathcal{T}_\mathcal{J}))$.
- *Für jedes* $b = (t, s) \in \mathcal{T}$ *gilt* $\hat{b} = \hat{t} \times \hat{s}$.

- *Falls $b = (t, s) \in \mathcal{T}$ kein Blatt ist, gilt*

$$\operatorname{sons}(b) = \begin{cases} \{t\} \times \operatorname{sons}(s) & \text{falls } \operatorname{sons}(t) = \emptyset, \operatorname{sons}(s) \neq \emptyset, \\ \operatorname{sons}(t) \times \{s\} & \text{falls } \operatorname{sons}(t) \neq \emptyset, \operatorname{sons}(s) = \emptyset, \\ \operatorname{sons}(t) \times \operatorname{sons}(s) & \text{ansonsten.} \end{cases} \tag{2.5}$$

In diesem Fall verwenden wir in der Regel die Notation $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ für $\mathcal{T}$, die durch Lemma 2.16 gerechtfertigt ist. Die Knoten $b = (t, s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ bezeichnen wir als Blöcke *und die Komponenten $t \in \mathcal{T}_{\mathcal{I}}$ und $s \in \mathcal{T}_{\mathcal{J}}$ als* Zeilen- *beziehungsweise* Spaltencluster *von b.*

Die Menge aller Blätter eines Blockbaums $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ bezeichnen wir mit $\mathcal{L}_{\mathcal{I}\times\mathcal{J}}$.

Ein Blick auf Definition 2.7 führt zu der Erkenntnis, dass sich zwei Blockbäume zu denselben Clusterbäumen nur durch die Wahl ihrer Blätter unterscheiden können.

Wir benötigen also ein Kriterium, das es uns ermöglicht, zu entscheiden, ob ein Block $b = (t, s)$ effizient approximiert werden kann oder ob er weiter unterteilt werden sollte. Das Kriterium sollte so beschaffen sein, dass

- nur solche Blöcke zu Blättern werden, die zuverlässig approximiert werden können,
- approximierbare Blöcke nicht weiter unterteilt werden und
- das Kriterium effizient angewendet werden kann.

Da derartige Kriterien sehr häufig vorkommen werden, geben wir ihnen einen Namen:

Definition 2.8 (Zulässigkeitsbedingung) *Eine Abbildung, die jedem Paar (t, s) mit $t \in \mathcal{T}_{\mathcal{I}}$, $s \in \mathcal{T}_{\mathcal{J}}$ entweder „zulässig“ oder „unzulässig“ zuordnet, bezeichnen wir als* Zulässigkeitsbedingung.

Die Blätter eines Clusterbaums beschreiben eine brauchbare Zerlegung einer Matrix, wenn sie entweder zulässig sind, denn dann lassen sich die zugehörigen Teilmatrizen approximieren, oder wenn sie klein sind, denn dann können wir sie direkt abspeichern, ohne Effizienz einzubüßen.

Definition 2.9 (Zulässiger Blockbaum) *Ein Blockbaum $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ heißt* zulässig *bezüglich einer Zulässigkeitsbedingung, wenn*

$$(t, s) \text{ zulässig oder } \operatorname{sons}(t) = \emptyset \text{ oder } \operatorname{sons}(s) = \emptyset \qquad \text{für alle } (t, s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$$

gilt, wenn also bei nicht zulässigen Blättern entweder der Zeilen- oder der Spaltencluster ein Blatt ist. Der Baum heißt streng zulässig*, wenn*

$$(t, s) \text{ zulässig oder } \operatorname{sons}(t) = \emptyset = \operatorname{sons}(s) \qquad \text{für alle } (t, s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$$

gilt, wenn nicht zulässige Blätter also aus Blattclustern zusammengesetzt sein müssen.

```
procedure blockcluster(var b);
(t, s) ← b;
if (t, s) zulässig then
  sons(b) = ∅
else
  if strenge Konstruktion then
    if sons(t) = ∅ then
      sons(b) = {t} × sons(s)
    else if sons(s) = ∅ then
      sons(b) = sons(t) × {s}
    else
      sons(b) = sons(t) × sons(s)
    end if
  else
    sons(b) = sons(t) × sons(s)
  end if
  for b' ∈ sons(b) do
    blockcluster(b')
  end for
end if
```

Abbildung 2.4: Konstruktion eines zulässigen Blockbaums.

Basierend auf einer Zulässigkeitsbedingung und zwei Clusterbäumen können wir relativ einfach einen Blockbaum konstruieren, der sogar die kleinste mögliche Anzahl von Blättern besitzt: Wir prüfen die Zulässigkeit eines Blocks. Falls er nicht zulässig ist, zerlegen wir ihn entsprechend Definition 2.7 und fahren rekursiv fort. Der entsprechende Algorithmus ist in Abbildung 2.4 angegeben, bei seiner Anwendung auf ein eindimensionales Modellproblem erhalten wir den in Abbildung 2.5 stufenweise dargestellten Baum.

Falls den Indizes $i \in \mathcal{I}$ und $j \in \mathcal{J}$ geometrische Objekte $\Omega_i, \Omega_j \subseteq \mathbb{R}^d$ zugeordnet sind (beispielsweise Träger von Basisfunktionen, vgl. Abschnitt 5.3), sind Zulässigkeitsbedingungen der Form

$$(t,s) \text{ zulässig} \qquad \Longleftrightarrow \qquad \operatorname{diam}_2(\Omega_t) \leq \operatorname{dist}_2(\Omega_t, \Omega_s)$$

üblich, bei denen die Mengen

$$\Omega_t := \bigcup_{i \in \hat{t}} \Omega_i, \qquad \Omega_s := \bigcup_{j \in \hat{s}} \Omega_j \tag{2.6}$$

verwendet werden. In der Praxis kann es sehr schwierig werden, mit derartigen Mengen zu arbeiten: Den geometrischen Abstand zweier Punktmengen zu berechnen ist eine relativ aufwendige Operation, und für kompliziertere Mengen wird es in der Regel nicht einfacher.

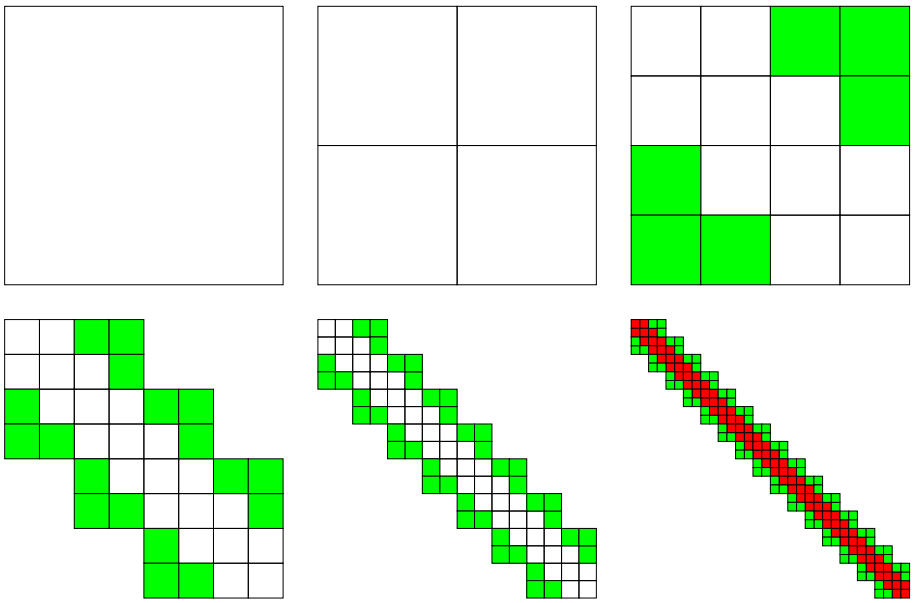

Abbildung 2.5: Blockbaum nach fünf Zerlegungsschritten für ein eindimensionales Modellproblem. Die einzelnen Stufen sind von links nach rechts und von oben nach unten angegeben. Grüne Quadrate stehen für zulässige Blöcke. Unzulässige Blöcke werden als leere Quadrate dargestellt, falls sie noch weiter zerlegt werden, und anderenfalls rot.

Deshalb verwendet man eine vereinfachte Zulässigkeitsbedingung: Zwar ist es für allgemeine Mengen Ω_t, Ω_s schwierig, Abstände und Durchmesser zu berechnen, für Kugeln oder achsenparallele Quader dagegen ist es sehr einfach:

Lemma 2.10 *Seien* $Q_t = [a_{t,1}, b_{t,1}] \times \ldots \times [a_{t,d}, b_{t,d}]$ *und* $Q_s = [a_{s,1}, b_{s,1}] \times \ldots \times [a_{s,d}, b_{s,d}]$ *achsenparallele Quader in* $\mathbb{R}^d$*. Dann gelten*

$$\operatorname{diam}_2(Q_t) = \left(\sum_{\nu=1}^{d} \operatorname{diam}_2([a_{t,\nu}, b_{t,\nu}])^2 \right)^{1/2},$$

$$\operatorname{dist}_2(Q_t, Q_s) = \left(\sum_{\nu=1}^{d} \operatorname{dist}_2([a_{t,\nu}, b_{t,\nu}], [a_{s,\nu}, b_{s,\nu}])^2 \right)^{1/2}.$$

Beweis. Der Abstand ist durch

$$\operatorname{dist}_2(Q_t, Q_s) = \min\{\|\mathbf{x} - \mathbf{y}\|_2 \ : \ \mathbf{x} \in Q_t, \mathbf{y} \in Q_s\}$$

definiert. Wir wählen zunächst $\mathbf{x} \in Q_t$ und $\mathbf{y} \in Q_s$ und folgern aus der Definition

$|x_\nu - y_\nu| \geq \operatorname{dist}_2([a_{t,\nu}, b_{t,\nu}], [a_{s,\nu}, b_{s,\nu}])$ für alle $\nu \in \{1, \ldots, d\}$ und damit

$$\|\mathbf{x} - \mathbf{y}\|_2 = \left(\sum_{\nu=1}^{d} |x_\nu - y_\nu|^2 \right)^{1/2} \geq \left(\sum_{\nu=1}^{d} \operatorname{dist}_2([a_{t,\nu}, b_{t,\nu}], [a_{s,\nu}, b_{s,\nu}])^2 \right)^{1/2}.$$

Da $\mathbf{x}$ und $\mathbf{y}$ beliebig gewählt sind, muss der rechte Term eine untere Schranke für den Abstand sein. Nun wählen wir die Komponenten $x_\nu \in [a_{t,\nu}, b_{t,\nu}]$ und $y_\nu \in [a_{s,\nu}, b_{s,\nu}]$ der Vektoren $\mathbf{x}, \mathbf{y}$ so, dass $|x_\nu - y_\nu| = \operatorname{dist}_2([a_{t,\nu}, b_{t,\nu}], [a_{s,\nu}, b_{s,\nu}])$ gilt. Dann gelten $\mathbf{x} \in Q_t$ und $\mathbf{y} \in Q_s$ und wir erhalten

$$\left(\sum_{\nu=1}^{d} \operatorname{dist}_2([a_{t,\nu}, b_{t,\nu}], [a_{s,\nu}, b_{s,\nu}])^2 \right)^{1/2} = \left(\sum_{\nu=1}^{d} |x_\nu - y_\nu|^2 \right)^{1/2} = \|\mathbf{x} - \mathbf{y}\|_2 \geq \operatorname{dist}_2(Q_t, Q_s).$$

Aus der Kombination beider Ungleichungen folgt die gesuchte Identität für den Abstand.

Für den durch

$$\operatorname{diam}_2(Q_t) = \max\{\|\mathbf{x} - \mathbf{y}\|_2 \ : \ \mathbf{x}, \mathbf{y} \in Q_t\},$$

gegebenen Durchmesser können wir entsprechend verfahren. ∎

Wenn $\Omega_t \subseteq Q_t$ und $\Omega_s \subseteq Q_s$ gelten, können wir die verschärfte Zulässigkeitsbedingung

$$(t, s) \text{ zulässig} \qquad \Longleftrightarrow \qquad \operatorname{diam}_2(Q_t) \leq \operatorname{dist}_2(Q_t, Q_s)$$

definieren, die wegen

$$\operatorname{diam}_2(\Omega_t) \leq \operatorname{diam}_2(Q_t) \leq \operatorname{dist}_2(Q_t, Q_s) \leq \operatorname{dist}_2(\Omega_t, \Omega_s)$$

die ursprüngliche Bedingung impliziert und in der Regel wesentlich effizienter berechnet werden kann.

Definition 2.11 (Überdeckende Quader) *Eine Familie $(Q_t)_{t \in \mathcal{T}_\mathcal{I}}$ achsenparalleler Quader*

$$Q_t = [a_{t,1}, b_{t,1}] \times \ldots \times [a_{t,d}, b_{t,d}]$$

bezeichnen wir als überdeckende Quader*, falls*

$$\Omega_t \subseteq Q_t \qquad \text{für alle } t \in \mathcal{T}_\mathcal{I}$$

gilt. Ebenfalls gebräuchlich ist die Bezeichnung bounding box*.*

Der optimale, also kleinste, überdeckende Quader ist durch die Gleichungen

$$a_{t,\nu} := \min\{x_\nu \ : \ \mathbf{x} \in \Omega_t\}, \tag{2.7a}$$

$$b_{t,\nu} := \max\{x_\nu \ : \ \mathbf{x} \in \Omega_t\} \qquad \text{für alle } t \in \mathcal{T}_\mathcal{I}, \nu \in \{1, \ldots, d\} \tag{2.7b}$$

gegeben. Für ein $t \in \mathcal{T}_\mathcal{I}$ mit $\operatorname{sons}(t) \neq \emptyset$ implizieren Definition 2.2 und (2.6) die Gleichung

$$\Omega_t = \bigcup_{i \in \hat{t}} \Omega_i = \bigcup_{t' \in \operatorname{sons}(t)} \bigcup_{i \in \hat{t}'} \Omega_i = \bigcup_{t' \in \operatorname{sons}(t)} \Omega_{t'},$$

und aus (2.7) folgt dann

$$a_{t,\nu} = \min\{\min\{x_\nu \ : \ \mathbf{x} \in \Omega_{t'}\} \ : \ t' \in \operatorname{sons}(t)\} = \min\{a_{t',\nu} \ : \ t' \in \operatorname{sons}(t)\},$$
$$b_{t,\nu} = \max\{\max\{x_\nu \ : \ \mathbf{x} \in \Omega_{t'}\} \ : \ t' \in \operatorname{sons}(t)\} = \max\{b_{t',\nu} \ : \ t' \in \operatorname{sons}(t)\},$$

so dass wir den optimalen überdeckenden Quader für t aus den optimalen Quadern für dessen Söhne konstruieren können.

Die potentiell immer noch komplizierten Mengen Ω_i benötigen wir also nur, um die Quader für Blätter des Clusterbaums zu konstruieren. In der Praxis bietet es sich an, zunächst überdeckende Quader für Ω_i zu bestimmen und diese dann in der bereits beschriebenen Weise zu Quadern für Ω_t zusammenzufassen.

Mit Hilfe der überdeckenden Quader können wir nun die Zulässigkeitsbedingung

$$\operatorname{diam}_2(Q_t) \leq \operatorname{dist}_2(Q_t, Q_s)$$

einfach nachprüfen und mit Hilfe des Algorithmus aus Abbildung 2.4 einen zulässigen Blockbaum konstruieren, der sich zur Approximation einer Matrix eignet.

2.3 Hierarchische Matrizen

Auf Grundlage eines zulässigen Blockbaums $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ können wir die Struktur einer hierarchischen Matrix definieren: Jedes zulässige Blatt des Baums entspricht einer Teilmatrix, die mit niedrigem Rang approximiert werden soll, während jedes unzulässige Blatt als konventionelle vollbesetzte Matrix gespeichert werden kann.

Definition 2.12 (Rang-k-Matrix) *Seien $t \in \mathcal{T}_\mathcal{I}$ und $s \in \mathcal{T}_\mathcal{J}$ gegeben, und sei $k \in \mathbb{N}_0$. Eine Matrix $\mathbf{X} \in \mathbb{K}^{\hat{t}\times\hat{s}}$ bezeichnen wir als* Rang-k-Matrix*, falls Matrizen*

$$\mathbf{A} \in \mathbb{K}^{\hat{t}\times k}, \qquad \mathbf{B} \in \mathbb{K}^{\hat{s}\times k}$$

mit der Eigenschaft

$$\mathbf{X} = \mathbf{A}\mathbf{B}^*$$

existieren. Die Darstellung von $\mathbf{X}$ durch dieses Matrixpaar $(\mathbf{A}, \mathbf{B})$ bezeichnen wir als faktorisierte Rang-k-Darstellung*. Die Menge aller derartigen Rang-k-Matrizen bezeichnen wir mit*

$$\mathcal{R}(t, s, k) := \{\mathbf{X} \in \mathbb{K}^{\hat{t}\times\hat{s}} \ : \ \mathbf{X} = \mathbf{A}\mathbf{B}^* \text{ für } \mathbf{A} \in \mathbb{K}^{\hat{t}\times k}, \mathbf{B} \in \mathbb{K}^{\hat{s}\times k}\}.$$

Für unsere Zwecke ist vor allem wichtig, dass sich eine Rang-k-Matrix in faktorisierter Darstellung durch nur $k(\#\hat{t} + \#\hat{s})$ Koeffizienten beschreiben lässt. Falls $k \ll \#\hat{t}, \#\hat{s}$ gilt, ist deshalb die faktorisierte Darstellung wesentlich effizienter als die konventionelle Darstellung durch ein zweidimensionales Array, die $(\#\hat{t})(\#\hat{s})$ Koeffizienten benötigt.

Bemerkung 2.13 (Rang) *In der linearen Algebra definiert man den Rang einer Matrix beispielsweise als die Dimension ihres Bildes. Eine Rang-k-Matrix gemäß Definition 2.12 ist in diesem Sinne eine Matrix, die* höchstens *Rang k besitzt.*

Beispielsweise könnte $\mathbf{A} = \mathbf{B} = \mathbf{0}$ *gelten, dann wäre der Rang zwar null, aber die Matrix* $\mathbf{AB}^*$ *würde trotzdem Definition 2.12 entsprechend eine Rang-k-Matrix sein.*

Mit Hilfe von Rang-k-Matrizen lässt sich nun definieren, was eine allgemeine hierarchische Matrix ist: Wir fassen zulässige und unzulässige Blätter in Mengen

$$\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}} := \{b \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}} \ : \ b \text{ ist zulässig}\}, \qquad \mathcal{L}^-_{\mathcal{I}\times\mathcal{J}} := \{b \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}} \ : \ b \text{ ist unzulässig}\}$$

zusammen und verlangen, dass jedes zulässige Blatt durch eine Rang-k-Matrix dargestellt werden kann.

Definition 2.14 (Hierarchische Matrix) *Sei* $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ *ein zulässiger Blockbaum, und sei* $k \in \mathbb{N}_0$*. Eine Matrix* $\mathbf{X} \in \mathbb{K}^{\mathcal{I}\times\mathcal{J}}$ *bezeichnen wir als* hierarchische Matrix *(oder kurz* $\mathcal{H}$-Matrix*), falls*

$$\mathbf{X}|_{\hat{t}\times\hat{s}} \in \mathcal{R}(t,s,k) \qquad \text{für alle } b = (t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$$

gilt. Die Zahl k nennen wir dann den lokalen Rang *von* $\mathbf{X}$*. Die Menge aller hierarchischen Matrizen mit lokalem Rang k bezeichnen wir mit*

$$\mathcal{H}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, k) := \{\mathbf{X} \in \mathbb{K}^{\mathcal{I}\times\mathcal{J}} \ : \ \mathbf{X} \text{ ist } \mathcal{H}\text{-Matrix mit lokalem Rang } k\}.$$

Für uns von Interesse ist die Tatsache, dass sich eine hierarchische Matrix in besonders kompakter Form in einem Computer darstellen lässt:

Definition 2.15 ($\mathcal{H}$-Matrix-Darstellung) *Sei* $\mathbf{X} \in \mathbb{K}^{\mathcal{I}\times\mathcal{J}}$ *eine hierarchische Matrix mit lokalem Rang* $k \in \mathbb{N}_0$*. Nach Definition können wir die zulässigen Blätter effizient darstellen, denn es existieren Familien* $(\mathbf{A}_b)_{b\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}$ *und* $(\mathbf{B}_b)_{b\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}$ *mit*

$$\mathbf{A}_b \in \mathbb{K}^{\hat{t}\times k}, \qquad \mathbf{B}_b \in \mathbb{K}^{\hat{s}\times k}, \qquad \mathbf{X}|_{\hat{t}\times\hat{s}} = \mathbf{A}_b\mathbf{B}_b^* \qquad \text{für alle } b = (t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}.$$

Für die Behandlung der unzulässigen Blätter führen wir die Familie $(\mathbf{N}_b)_{b\in\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}}$ *mit*

$$\mathbf{N}_b := \mathbf{X}|_{\hat{t}\times\hat{s}} \qquad \text{für alle } b = (t,s) \in \mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}$$

ein. Das Tripel $((\mathbf{A}_b)_{b\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}, (\mathbf{B}_b)_{b\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}, (\mathbf{N}_b)_{b\in\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}})$ *bezeichnen wir dann als* $\mathcal{H}$-Matrix-Darstellung *der Matrix* $\mathbf{X}$*.*

2.4 Blockpartitionen

Definition 2.15 beschreibt eine Matrix $\mathbf{X} \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ durch ihre Einschränkungen $\mathbf{X}|_{\hat{t}\times\hat{s}}$ auf Blätter $b = (t, s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ des Blockbaums. Offenbar ist dieser Ansatz nur sinnvoll, falls er die gesamte Matrix beschreibt, wenn also jedes Indexpaar $(i, j) \in \mathcal{I} \times \mathcal{J}$ in genau einem Block vertreten ist. Diese Eigenschaft werden wir nun nachweisen und dabei auch einige wichtige weitere Eigenschaften von Cluster- und Blockbäumen kennenlernen.

Zunächst stellen wir fest, dass ein Blockbaum sich auch als Clusterbaum interpretieren lässt. Diese Eigenschaft ermöglicht es uns, eine Reihe wichtiger Beweise für allgemeine Clusterbäume zu führen.

Lemma 2.16 (Blockbaum) *Sei $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Blockbaum für $\mathcal{T}_{\mathcal{I}}$ und $\mathcal{T}_{\mathcal{J}}$. Dann ist $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ auch ein Clusterbaum für die Indexmenge $\mathcal{I} \times \mathcal{J}$.*

Beweis. Sei $r \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ die Wurzel des Blockbaums, und seien $r_{\mathcal{I}} \in \mathcal{T}_{\mathcal{I}}$ und $r_{\mathcal{J}} \in \mathcal{T}_{\mathcal{J}}$ die Wurzeln der Clusterbäume $\mathcal{T}_{\mathcal{I}}$ und $\mathcal{T}_{\mathcal{J}}$. Nach Definition gilt $r = (r_{\mathcal{I}}, r_{\mathcal{J}})$, und damit auch $\hat{r} = \hat{r}_{\mathcal{I}} \times \hat{r}_{\mathcal{J}} = \mathcal{I} \times \mathcal{J}$.

Sei nun $b = (t, s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ mit $\operatorname{sons}(b) \neq \emptyset$ fixiert.

Sei $(i, j) \in \hat{b} = \hat{t} \times \hat{s}$ gewählt. Falls t kein Blatt ist, existiert nach Definition 2.2 ein $t' \in \operatorname{sons}(t)$ mit $i \in \hat{t}'$, anderenfalls setzen wir $t' = t$. Falls s kein Blatt ist, existiert entsprechend ein $s' \in \operatorname{sons}(s)$ mit $j \in \hat{s}'$, anderenfalls setzen wir $s' = s$. Es folgen $b' := (t', s') \in \operatorname{sons}(b)$ und $(i, j) \in \hat{t}' \times \hat{s}' = \hat{b}'$. Damit ist

$$\bigcup_{b'\in\operatorname{sons}(b)} \hat{b}' = \hat{b}$$

bewiesen.

Seien nun $b_1 = (t_1, s_1), b_2 = (t_2, s_2) \in \operatorname{sons}(b)$ mit $b_1 \neq b_2$ gewählt. Dann folgt $t_1 \neq t_2$ oder $s_1 \neq s_2$, also nach Definition 2.2 auch $\hat{t}_1 \cap \hat{t}_2 = \emptyset$ oder $\hat{s}_1 \cap \hat{s}_2 = \emptyset$ und damit insbesondere

$$\hat{b}_1 \cap \hat{b}_2 = (\hat{t}_1 \times \hat{s}_1) \cap (\hat{t}_2 \times \hat{s}_2) = (\hat{t}_1 \cap \hat{t}_2) \times (\hat{s}_1 \cap \hat{s}_2) = \emptyset,$$

und wir haben alle Bedingungen aus Definition 2.2 nachgewiesen. ∎

Für die theoretische Untersuchung von Bäumen ist es sehr oft nützlich, den Baum in *Stufen* (engl. *levels*) einzuteilen, die jedem Knoten des Baums seine „Entfernung“ zu der Wurzel zuordnen:

Definition 2.17 (Stufen) *Sei $\mathcal{T}_{\mathcal{I}}$ ein Clusterbaum. Die Abbildung*

$$\operatorname{level} : \mathcal{T}_{\mathcal{I}} \to \mathbb{N}_0, \quad t \mapsto \begin{cases} \operatorname{level}(t^+) + 1 & \text{falls ein } t^+ \in \mathcal{T}_{\mathcal{I}} \text{ mit } t \in \operatorname{sons}(t^+) \text{ existiert,} \\ 0 & \text{ansonsten, also falls } t = \operatorname{root}(\mathcal{T}_{\mathcal{I}}) \text{ gilt,} \end{cases}$$

ordnet jedem Cluster $t \in \mathcal{T}_{\mathcal{I}}$ eine Stufe $\operatorname{level}(t)$ *im Baum $\mathcal{T}_{\mathcal{I}}$ zu.*

Die Möglichkeit, sich bei der Untersuchung von Clusterbäumen auf Stufen zurückzuziehen, ist bei Beweisen sehr nützlich. Als erstes Beispiel zeigen wir, dass die Vereinigung der Beschriftungen der Blätter eines Clusterbaums die Indexmenge $\mathcal{I}$ enthält:

Lemma 2.18 (Überdeckung) *Sei $\mathcal{T}_\mathcal{I}$ ein Clusterbaum, und seien $\mathcal{L}_\mathcal{I}$ seine Blätter. Für jedes $i \in \mathcal{I}$ existiert ein Blatt $t \in \mathcal{L}_\mathcal{I}$ mit $i \in \hat{t}$.*

Beweis. Sei $i \in \mathcal{I}$, und sei

$$C := \{t \in \mathcal{T}_\mathcal{I} \ : \ i \in \hat{t}\}$$

die Menge aller Cluster, die i enthalten. Nach Definition 2.2 gilt $\mathrm{root}(\mathcal{T}_\mathcal{I}) \in C$, also kann die Menge C nicht leer sein. Da sie eine Teilmenge von $\mathcal{T}_\mathcal{I}$ ist, muss sie endlich sein, also existiert ein Maximum

$$m := \max\{\mathrm{level}(t) \ : \ t \in C\}.$$

Per Kontraposition werden wir nun beweisen, dass jedes $t \in C$ mit $\mathrm{level}(t) = m$ ein Blatt sein muss. Sei dazu ein $t \in C$ fixiert, das *kein* Blatt ist, für das also $\mathrm{sons}(t) \neq \emptyset$ gilt. Nach (2.3) muss dann ein $t' \in \mathrm{sons}(t)$ mit $i \in \hat{t}'$ existieren, und es folgt $t' \in C$. Somit haben wir $\mathrm{level}(t) = \mathrm{level}(t') - 1 \leq m - 1 < m$ bewiesen.

Also wählen wir ein $t \in C$ mit $\mathrm{level}(t) = m$ und folgern, dass $t \in \mathcal{L}_\mathcal{I}$ gelten muss. ∎

Indem wir Lemma 2.16 und Lemma 2.18 kombinieren, folgt bereits, dass jedes Paar $(i, j) \in \mathcal{I} \times \mathcal{J}$ in mindestens einem Blatt $b = (t, s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ auftreten muss. Als nächstes ist zu zeigen, dass es genau ein Blatt mit dieser Eigenschaft gibt.

Da die Stufenzahl in $\mathbb{N}_0$ liegt, können wir mit ihrer Hilfe Induktionsbeweise führen, beispielsweise um zu zeigen, dass auf jeder Stufe des Clusterbaums ein Index höchstens einmal auftreten kann:

Lemma 2.19 (Disjunkt) *Sei $\mathcal{T}_\mathcal{I}$ ein Clusterbaum. Für alle $t, s \in \mathcal{T}_\mathcal{I}$ gilt*

$$t \neq s \text{ und } \mathrm{level}(t) = \mathrm{level}(s) \qquad \Longrightarrow \qquad \hat{t} \cap \hat{s} = \emptyset, \tag{2.8}$$

die Cluster auf einer Ebene eines Clusterbaums sind also disjunkt.

Beweis. Wir führen den Beweis per Induktion über die Stufe. Für $t, s \in \mathcal{T}_\mathcal{I}$ mit $\mathrm{level}(t) = \mathrm{level}(s) = 0$ muss sowohl t als auch s die Wurzel sein, es gilt also $t = \mathrm{root}(\mathcal{T}_\mathcal{I}) = s$ und (2.8) ist trivial erfüllt.

Sei nun $\ell \in \mathbb{N}_0$ so gewählt, dass (2.8) für alle $t, s \in \mathcal{T}_\mathcal{I}$ mit $\mathrm{level}(t) = \mathrm{level}(s) = \ell$ gilt. Seien $t, s \in \mathcal{T}_\mathcal{I}$ mit $\mathrm{level}(t) = \mathrm{level}(s) = \ell + 1$ fixiert. Damit gelten insbesondere $\mathrm{level}(t) > 0$ und $\mathrm{level}(s) > 0$, also $t \neq \mathrm{root}(\mathcal{T}_\mathcal{I})$ und $s \neq \mathrm{root}(\mathcal{T}_\mathcal{I})$, und es müssen Väter $t^+ \in \mathcal{T}_\mathcal{I}$ und $s^+ \in \mathcal{T}_\mathcal{I}$ mit $t \in \mathrm{sons}(t^+)$ sowie $s \in \mathrm{sons}(s^+)$ existieren.

Falls $t^+ = s^+$ gilt, sind t und s Söhne desselben Vaters, und aus (2.4) folgt $\hat{t} \cap \hat{s} = \emptyset$.

Falls $t^+ \neq s^+$ gilt, folgt aus Definition 2.17 die Gleichung

$$\mathrm{level}(t^+) = \mathrm{level}(t) - 1 = \ell = \mathrm{level}(s) - 1 = \mathrm{level}(s^+),$$

also können wir die Induktionsvoraussetzung auf t^+ und s^+ anwenden und folgern, dass $\hat{t}^+ \cap \hat{s}^+ = \emptyset$ gelten muss. Die Bedingung (2.3) impliziert $\hat{t} \subseteq \hat{t}^+$ und $\hat{s} \subseteq \hat{s}^+$, also auch $\hat{t} \cap \hat{s} \subseteq \hat{t}^+ \cap \hat{s}^+ = \emptyset$. ∎

Infolge dieser Eigenschaft kann ein Index nur in höchstens einem Cluster pro Stufe des Clusterbaums auftreten, und (2.3) impliziert, dass diese Cluster miteinander verwandt sein müssen:

Lemma 2.20 (Nachfahren) *Sei $\mathcal{T}_\mathcal{I}$ ein Clusterbaum. Für alle $t, s \in \mathcal{T}_\mathcal{I}$ gilt*

$$\hat{t} \cap \hat{s} \neq \emptyset \text{ und } \operatorname{level}(t) \leq \operatorname{level}(s) \qquad \Longrightarrow \qquad s \in \operatorname{sons}^*(t), \tag{2.9}$$

falls also zwei Cluster nicht disjunkt sind, muss einer der beiden ein Nachfahre des anderen sein.

Beweis. Wir führen den Beweis per Induktion über die Differenz der Stufen.

Für $t, s \in \mathcal{T}_\mathcal{I}$ mit $\operatorname{level}(s) - \operatorname{level}(t) = 0$ folgt die Aussage (2.9) direkt aus Lemma 2.19.

Sei nun $n \in \mathbb{N}_0$ so gewählt, dass (2.9) für alle $t, s \in \mathcal{T}_\mathcal{I}$ mit $\operatorname{level}(s) - \operatorname{level}(t) = n$ gilt. Seien $t, s \in \mathcal{T}_\mathcal{I}$ mit $\hat{t} \cap \hat{s} \neq \emptyset$ und $\operatorname{level}(s) - \operatorname{level}(t) = n + 1$ gegeben. Aus $\operatorname{level}(s) = n+1+\operatorname{level}(t) > 0$ folgt $s \neq \operatorname{root}(\mathcal{T}_\mathcal{I})$, also existiert ein $s^+ \in \mathcal{T}_\mathcal{I}$ mit $s \in \operatorname{sons}(s^+)$. Wegen $\hat{s}^+ \cap \hat{t} \supseteq \hat{s} \cap \hat{t} \neq \emptyset$ können wir die Induktionsvoraussetzung anwenden und $s^+ \in \operatorname{sons}^*(t)$ folgern. Nach (2.1) gilt dann auch $s \in \operatorname{sons}^*(t)$. ∎

Dieses Lemma impliziert bereits, dass ein Index nur in höchstens einem Blatt eines Clusterbaums auftreten kann, und zusammen mit Lemma 2.18 erhalten wir die folgende Aussage:

Folgerung 2.21 (Blattpartition) *Sei $\mathcal{T}_\mathcal{I}$ ein Clusterbaum. Die Beschriftungen seiner Blätter bilden eine disjunkte Partition der Indexmenge $\mathcal{I}$:*

$$\dot{\bigcup}_{t \in \mathcal{L}_\mathcal{I}} \hat{t} = \mathcal{I}.$$

Beweis. Dank Lemma 2.18 wissen wir, dass jedes $i \in \mathcal{I}$ in mindestens einem Blattcluster vorkommt. Falls ein Index $i \in \mathcal{I}$ in mehreren Clustern vorkommt, kann nach Lemma 2.20 nur einer davon ein Blatt sein, da Blätter keine Nachfahren außer sich selbst besitzen. ∎

Da nach Lemma 2.16 jeder Blockbaum auch ein Clusterbaum ist, können wir Folgerung 2.21 anwenden, um das Hauptresultat dieses Abschnitts zu erhalten:

Folgerung 2.22 (Blattpartition) *Die Beschriftungen der Blätter eines Blockbaums $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ beschreiben eine disjunkte Partition der Indexmenge $\mathcal{I} \times \mathcal{J}$:*

$$\dot{\bigcup}_{b \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}} \hat{b} = \mathcal{I} \times \mathcal{J}.$$

Beweis. Dank Lemma 2.16 können wir Folgerung 2.21 auch auf $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ anwenden. ∎

Für jedes Indexpaar $(i, j) \in \mathcal{I} \times \mathcal{J}$ existiert also *genau ein Blatt* $b = (t, s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ des Blockbaums mit $(i, j) \in \hat{t} \times \hat{s}$. Da die $\mathcal{H}$-Matrix-Darstellung für jedes solche Blatt die Einschränkung $\mathbf{X}|_{\hat{t}\times\hat{s}}$ definiert, beschreibt sie die $\mathcal{H}$-Matrix $\mathbf{X}$ vollständig und eindeutig.

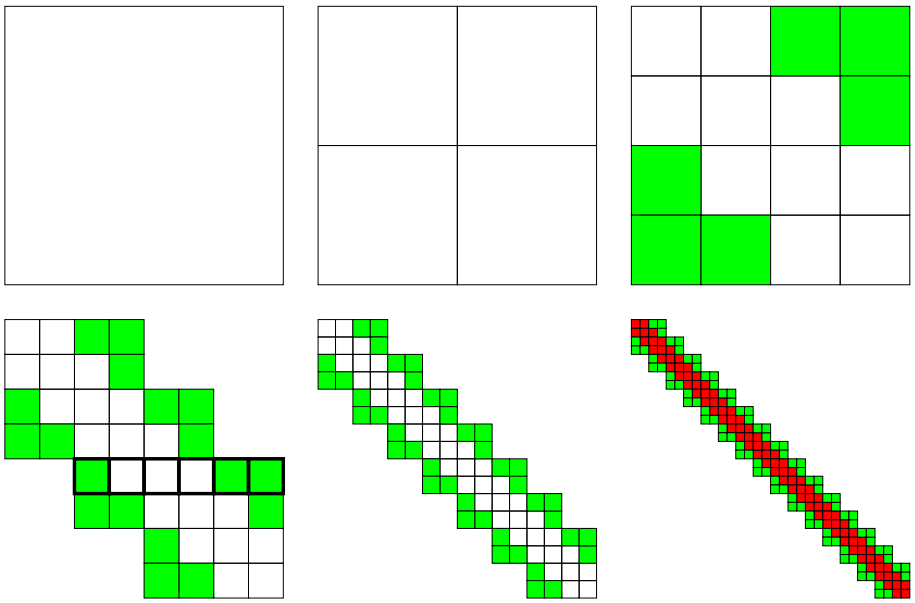

Abbildung 2.6: Zu einem Cluster gehörende Blockzeile.

2.5 Komplexität

Es stellt sich natürlich die Frage, ob die $\mathcal{H}$-Matrix-Darstellung einen Vorteil gegenüber der konventionellen Darstellung einer Matrix durch ein zweidimensionales Feld bietet. Zur Klärung dieser Frage bietet es sich an, das Konzept der *schwachbesetzten Blockbäume* [19, 21] zu verwenden, das sich durch besondere Eleganz auszeichnet.

Die entscheidende Idee dieses Ansatzes besteht darin, die Untersuchung des Blockbaums $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ auf die Untersuchung der Clusterbäume $\mathcal{T}_{\mathcal{I}}$ und $\mathcal{T}_{\mathcal{J}}$ zurückzuführen, über die wir dank der Lemmas 2.19 und 2.20 bereits viel wissen. Dazu untersuchen wir, bei wievielen Blöcken bestimmte Cluster als Zeilen- und Spaltencluster auftreten:

Definition 2.23 (Blockzeilen und -spalten) *Sei $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Blockbaum. Für jedes $t \in \mathcal{T}_{\mathcal{I}}$ definieren wir die zugehörige* Blockzeile *durch*

$$\operatorname{row}(t, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}) = \{s \in \mathcal{T}_{\mathcal{J}} \ : \ (t,s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}\},$$

und für jedes $s \in \mathcal{T}_{\mathcal{J}}$ definieren wir entsprechend die zugehörige Blockspalte *durch*

$$\operatorname{col}(s, \mathcal{T}_{\mathcal{I}\times\mathcal{J}}) = \{t \in \mathcal{T}_{\mathcal{I}} \ : \ (t,s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}\}.$$

Wir verwenden die Abkürzungen $\operatorname{row}(t)$ *und* $\operatorname{col}(s)$*, sofern der Blockbaum durch den Kontext gegeben ist.*

Definition 2.24 (Schwachbesetzter Blockbaum) *Sei $C_{sp} \in \mathbb{N}$, und sei $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Blockbaum für die Clusterbäume $\mathcal{T}_{\mathcal{I}}$ und $\mathcal{T}_{\mathcal{J}}$. Der Baum $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ heißt* C_{sp}-schwachbesetzt*, falls*

$$\#\operatorname{row}(t) \leq C_{sp}, \qquad \#\operatorname{col}(s) \leq C_{sp} \qquad \text{für alle } t \in \mathcal{T}_{\mathcal{I}}, s \in \mathcal{T}_{\mathcal{J}}$$

gelten, wenn also alle Blockzeilen und -spalten höchstens C_{sp} Cluster enthalten.

In dem in Abbildung 2.6 dargestellten Beispiel erkennen wir relativ leicht, dass der Blockbaum C_{sp}-schwachbesetzt mit $C_{\mathrm{sp}} = 6$ ist, denn jede Blockzeile und -spalte enthält höchstens sechs Blöcke.

Dank Lemma 2.19 wissen wir, dass die Beschriftungen der Cluster auf einer Stufe eines Clusterbaums $\mathcal{T}_{\mathcal{I}}$ disjunkte Teilmengen der korrespondierenden Indexmenge $\mathcal{I}$ sind. Damit kann insbesondere jeder Index $i \in \mathcal{I}$ auf jeder Stufe des Baums nur einmal auftreten, und mit einer Schranke für die Anzahl der Stufen können wir zu einer Abschätzung gelangen:

Definition 2.25 (Tiefe eines Baums) *Sei $\mathcal{T}_{\mathcal{I}}$ ein Clusterbaum. Die Größe*

$$\operatorname{depth}(\mathcal{T}_{\mathcal{I}}) := \max\{\operatorname{level}(t) \ : \ t \in \mathcal{T}_{\mathcal{I}}\}$$

nennen wir die Tiefe *des Baums $\mathcal{T}_{\mathcal{I}}$.*

Aus der Definition 2.7 erhalten wir die folgende Beziehung zwischen der Tiefe eines Blockbaums und den Tiefen der Clusterbäume, aus denen er konstruiert wurde:

Lemma 2.26 (Stufenzahl eines Blocks) *Sei $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Blockbaum zu den Clusterbäumen $\mathcal{T}_{\mathcal{I}}$ und $\mathcal{T}_{\mathcal{J}}$. Für jeden Block $b = (t, s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ gelten*

$$\operatorname{level}(b) = \max\{\operatorname{level}(t), \operatorname{level}(s)\}, \tag{2.10a}$$

$$\operatorname{level}(b) > \operatorname{level}(t) \Longrightarrow \operatorname{sons}(t) = \emptyset, \tag{2.10b}$$

$$\operatorname{level}(b) > \operatorname{level}(s) \Longrightarrow \operatorname{sons}(s) = \emptyset. \tag{2.10c}$$

Wenn wir mit $p_{\mathcal{I}\times\mathcal{J}}, p_{\mathcal{I}}, p_{\mathcal{J}} \in \mathbb{N}_0$ die Tiefen der drei Bäume bezeichnen, folgt daraus die Abschätzung $p_{\mathcal{I}\times\mathcal{J}} \leq \max\{p_{\mathcal{I}}, p_{\mathcal{J}}\}$.

Beweis. Die erste Behauptung beweisen wir per Induktion über $\operatorname{level}(b) \in \mathbb{N}_0$. Falls $\operatorname{level}(b) = 0$ gilt, ist b die Wurzel des Blockbaums und deshalb müssen nach Definition 2.7 t und s die Wurzeln der Clusterbäume $\mathcal{T}_{\mathcal{I}}$ und $\mathcal{T}_{\mathcal{J}}$ sein, so dass wir $\operatorname{level}(t) = 0 = \operatorname{level}(s)$ erhalten.

Sei nun $m \in \mathbb{N}_0$ so gewählt, dass die Behauptung für alle $b \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ mit $\operatorname{level}(b) = m$ gilt. Sei $b \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Block mit $\operatorname{level}(b) = m + 1$. Wegen $\operatorname{level}(b) > 0$ muss ein Vater $b^+ = (t^+, s^+) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ mit $b \in \operatorname{sons}(b^+)$ und $\operatorname{level}(b^+) = m$ existieren.

1. Fall: $\operatorname{sons}(t^+) = \emptyset$. Nach (2.5) gilt $b = (t, s) \in \operatorname{sons}(b^+) = \{t^+\} \times \operatorname{sons}(s^+)$, also folgen $t = t^+$ und $s \in \operatorname{sons}(s^+)$. Indem wir die Induktionsvoraussetzung auf b^+ anwenden, folgern wir zunächst per Kontraposition aus (2.10c), dass $\operatorname{level}(b^+) \leq \operatorname{level}(s^+)$ gelten

muss. Mit (2.10a) erhalten wir damit $\operatorname{level}(b^+) = \operatorname{level}(s^+)$ sowie $\operatorname{level}(t) = \operatorname{level}(t^+) \leq \operatorname{level}(s^+) < \operatorname{level}(s)$, also

$$\operatorname{level}(b) = \operatorname{level}(b^+) + 1 = \operatorname{level}(s^+) + 1 = \operatorname{level}(s) = \max\{\operatorname{level}(t), \operatorname{level}(s)\},$$

so dass (2.10a) bewiesen ist. Wegen $\operatorname{sons}(t) = \operatorname{sons}(t^+) = \emptyset$ ist (2.10b) trivial erfüllt, und wegen $\operatorname{level}(b) = \operatorname{level}(s)$ ist die linke Seite von (2.10c) nie erfüllt, so dass auch diese Implikation gilt.

2. Fall: $\operatorname{sons}(s^+) = \emptyset$. Dieser Fall kann entsprechend dem ersten behandelt werden.

3. Fall: $\operatorname{sons}(t^+) \neq \emptyset$ und $\operatorname{sons}(s^+) \neq \emptyset$. Nach (2.5) gilt $b = (t, s) \in \operatorname{sons}(b^+) = \operatorname{sons}(t^+) \times \operatorname{sons}(s^+)$, also $t \in \operatorname{sons}(t^+)$ und $s \in \operatorname{sons}(s^+)$. Indem wir die Induktionsvoraussetzung auf b^+ anwenden, folgern wir per Kontraposition aus (2.10b) und (2.10c), dass $\operatorname{level}(b^+) \leq \operatorname{level}(t^+)$ und $\operatorname{level}(b^+) \leq \operatorname{level}(s^+)$ gelten müssen. Mit (2.10a) erhalten wir $\operatorname{level}(t^+) = \operatorname{level}(b^+) = \operatorname{level}(s^+)$ und

$$\begin{aligned}\operatorname{level}(b) &= \operatorname{level}(b^+) + 1 = \max\{\operatorname{level}(t^+), \operatorname{level}(s^+)\} + 1 \\ &= \max\{\operatorname{level}(t^+) + 1, \operatorname{level}(s^+) + 1\} = \max\{\operatorname{level}(t), \operatorname{level}(s)\}.\end{aligned}$$

Die linken Seiten von (2.10b) und (2.10c) sind nicht erfüllt, so dass beide Implikationen gelten. Damit ist der Induktionsbeweis vollständig.

Für die zweite Aussage sei ein $b = (t, s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ mit $\operatorname{level}(b) = p_{\mathcal{I}\times\mathcal{J}}$ gewählt. Aus (2.10a) folgt direkt die gewünschte Abschätzung. ∎

Allein durch eine Kombination des Lemmas 2.19 mit den Definitionen 2.24 und 2.25 ließe sich bereits eine befriedigende Schranke für den Speicherbedarf aller zulässigen Blätter angeben. Für die Abschätzung der unzulässigen Blätter benötigen wir eine zusätzliche Voraussetzung, die sicher stellt, dass solche Blätter klein genug sind. Da in einem zulässigen Blockbaum bei einem unzulässigen Blatt $b = (t, s) \in \mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}$ entweder t oder s ein Blatt des jeweiligen Clusterbaums sein muss, genügt es, die Größe dieser Blätter zu beschränken:

Definition 2.27 (Auflösung) *Sei $\mathcal{T}_\mathcal{I}$ ein Clusterbaum. Die Größe*

$$\operatorname{res}(\mathcal{T}_\mathcal{I}) := \max\{\#\hat{t} \ : \ t \in \mathcal{L}_\mathcal{I}\}$$

nennen wir die Auflösung *des Baums $\mathcal{T}_\mathcal{I}$.*

Wichtig bei dieser Definition ist, dass das Maximum sich nur auf die Menge der *Blätter* des Clusterbaums erstreckt, denn ansonsten wäre es nach Definition 2.2 immer gerade $\#\mathcal{I}$ und damit in der Regel zu groß für unsere Zwecke.

Jetzt können wir uns um die Abschätzung des Speicherbedarfs einer $\mathcal{H}$-Matrix-Darstellung kümmern. Als Vorbereitung beweisen wir die folgende zentrale Abschätzung für Summen über die Knoten des Blockbaums:

Lemma 2.28 (Blocksumme) *Sei der Blockbaum $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ C_{sp}-schwachbesetzt, und bezeichne $p_{\mathcal{I}\times\mathcal{J}} := \operatorname{depth}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}})$ seine Tiefe. Dann gelten die Abschätzungen*

$$\sum_{b=(t,s)\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}}} \#\hat{t} \leq C_{sp}(p_{\mathcal{I}\times\mathcal{J}}+1)\#\mathcal{I},$$
$$\sum_{b=(t,s)\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}}} \#\hat{s} \leq C_{sp}(p_{\mathcal{I}\times\mathcal{J}}+1)\#\mathcal{J}.$$

Beweis. Sei $b=(t,s)\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$. Nach Lemma 2.26 gilt dann $\operatorname{level}(t)\leq\operatorname{level}(b)$, also folgt mit Definition 2.24

$$\sum_{b=(t,s)\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}}} \#\hat{t} = \sum_{\substack{t\in\mathcal{T}_{\mathcal{I}}\\ \operatorname{level}(t)\leq p_{\mathcal{I}\times\mathcal{J}}}} \sum_{s\in\operatorname{row}(t)} \#\hat{t} \leq C_{\mathrm{sp}} \sum_{\substack{t\in\mathcal{T}_{\mathcal{I}}\\ \operatorname{level}(t)\leq p_{\mathcal{I}\times\mathcal{J}}}} \#\hat{t} = C_{\mathrm{sp}} \sum_{\ell=0}^{p_{\mathcal{I}\times\mathcal{J}}} \sum_{\substack{t\in\mathcal{T}_{\mathcal{I}}\\ \operatorname{level}(t)=\ell}} \#\hat{t}.$$

Aufgrund des Lemmas 2.19 sind alle $t\in\mathcal{T}_{\mathcal{I}}$ auf derselben Stufe des Baums $\mathcal{T}_{\mathcal{I}}$ disjunkt, also folgt

$$\sum_{\substack{t\in\mathcal{T}_{\mathcal{I}}\\ \operatorname{level}(t)=\ell}} \#\hat{t} = \#\bigcup_{\substack{t\in\mathcal{T}_{\mathcal{I}}\\ \operatorname{level}(t)=\ell}} \hat{t} \leq \#\mathcal{I},$$

und wir erhalten

$$\sum_{b=(t,s)\in\mathcal{T}_{\mathcal{I}\times\mathcal{J}}} \#\hat{t} \leq C_{\mathrm{sp}} \sum_{\ell=0}^{p_{\mathcal{I}\times\mathcal{J}}} \#\mathcal{I} = C_{\mathrm{sp}}(p_{\mathcal{I}\times\mathcal{J}}+1)\#\mathcal{I}.$$

Für die zweite Summe können wir entsprechend verfahren. ∎

Die wesentliche Arbeit für die Abschätzung des Speicherbedarfs ist mit diesem Lemma bereits geleistet, so dass sich das gesuchte Ergebnis nun einfach aus Abschätzungen für die Blöcke gewinnen lässt:

Satz 2.29 (Speicherbedarf) *Die $\mathcal{H}$-Matrix-Darstellung einer Matrix $\mathbf{X}\in\mathbb{K}^{\mathcal{I}\times\mathcal{J}}$ für einen C_{sp}-schwachbesetzten zulässigen Blockbaum der Tiefe $p_{\mathcal{I}\times\mathcal{J}}$, Clusterbäume mit den Auflösungen $r_{\mathcal{I}}$ und $r_{\mathcal{J}}$ sowie lokalem Rang k benötigt nicht mehr als*

$$C_{sp}\max\{k,r_{\mathcal{I}},r_{\mathcal{J}}\}(p_{\mathcal{I}\times\mathcal{J}}+1)(\#\mathcal{I}+\#\mathcal{J}) \quad \textit{Speicherplätze.}$$

Beweis. Wir stellen zunächst fest, dass bei der $\mathcal{H}$-Matrix-Darstellung nur Daten für Blätter $b=(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ des Blockbaums $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ anfallen. Falls ein solches Blatt zulässig ist, speichern wir die Matrix $\mathbf{A}_b$, die $(\#\hat{t})k$ Speicherplätze benötigt, und die Matrix $\mathbf{B}_b$, die $(\#\hat{s})k$ Speicherplätze erfordert. Für zulässige Blätter erhalten wir also insgesamt

$$k(\#\hat{t}+\#\hat{s}) \quad \text{Speicherplätze.}$$

Falls das Blatt nicht zulässig ist, muss nach Definition 2.9 t oder s ein Blatt sein, es muss also $\#\hat{t}\leq r_{\mathcal{I}}$ oder $\#\hat{s}\leq r_{\mathcal{J}}$ gelten. Im ersten Fall brauchen wir $(\#\hat{t})(\#\hat{s})\leq r_{\mathcal{I}}(\#\hat{s})$

Speicherplätze, im zweiten sind es $(\#\hat{t})(\#\hat{s}) \leq (\#\hat{t})r_{\mathcal{J}}$ Speicherplätze. In beiden Fällen können wir die Auflösung durch das Maximum abschätzen und erhalten insgesamt

$$\max\{r_{\mathcal{I}}, r_{\mathcal{J}}\}(\#\hat{t} + \#\hat{s}) \quad \text{Speicherplätze.}$$

Wir setzen

$$c := \max\{k, r_{\mathcal{I}}, r_{\mathcal{J}}\}$$

und stellen fest, dass wir den Speicherbedarf *jedes* Blatts $b = (t, s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ durch

$$c(\#\hat{t} + \#\hat{s})$$

abschätzen können. Nun summieren wir über alle Blätter und wenden Lemma 2.28 an, um

$$\begin{aligned}\sum_{b=(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}} c(\#\hat{t} + \#\hat{s}) &= c \sum_{b=(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}} \#\hat{t} + c \sum_{b=(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}} \#\hat{s} \\ &\leq C_{\mathrm{sp}} c(p_{\mathcal{I}\times\mathcal{J}} + 1)\#\mathcal{I} + C_{\mathrm{sp}} c(p_{\mathcal{I}\times\mathcal{J}} + 1)\#\mathcal{J}\end{aligned}$$

zu erhalten. ∎

Nun besteht unsere Aufgabe darin, die in der Abschätzung auftretenden Größen näher zu untersuchen. Die Auflösungen $r_{\mathcal{I}}$ und $r_{\mathcal{J}}$ sind dabei besonders einfach zu kontrollieren: Wir können das Abbruchkriterium für die Konstruktion des Clusterbaums so wählen, dass nur Cluster unterhalb einer gewissen Größe Blätter werden dürfen, und in Hinblick auf Satz 2.29 bietet sich k als Schranke an, denn dann reduziert sich das Maximum über k, $r_{\mathcal{I}}$ und $r_{\mathcal{J}}$ gerade auf k.

Schwieriger ist es, die Konstante C_{sp} und die Baumtiefen $p_{\mathcal{I}}$ und $p_{\mathcal{J}}$ unter Kontrolle zu bringen. Für den allgemeinen Fall sei auf [21] verwiesen, wir werden hier nur einen besonders einfachen Modellfall diskutieren, an dem sich trotzdem das wesentliche Verhalten illustrieren lässt.

2.6 Diskussion im Modellfall

Sei $(\mathbf{x}_i)_{i\in\mathcal{I}}$ eine Menge von Punkten $\mathbf{x}_i \in \mathbb{R}^d$ im d-dimensionalen Raum. Sei $\mathcal{I}$ endlich, und seien alle Punkte paarweise verschieden.

Um die Tiefe des Clusterbaums $\mathcal{T}_{\mathcal{I}}$ diskutieren zu können, müssen wir seine Konstruktion untersuchen. Wir verwenden einen besonders einfachen Ansatz: Wir wählen $a, b \in \mathbb{R}$ mit

$$\mathbf{x}_i \in [a, b)^d \qquad \text{für alle } i \in \mathcal{I},$$

wir fixieren also einen achsenparallelen Würfel, der die gesamte Punktmenge enthält. Definition 2.2 lässt es zu, dass wir diesen Würfel als Wurzel r des Clusterbaums $\mathcal{T}_{\mathcal{I}}$ verwenden. Um dieser Definition gerecht zu werden, muss r die Beschriftung $\hat{r} = \mathcal{I}$ besitzen.

Die Menge der Söhne definieren wir, indem wir einen Cluster regelmäßig unterteilen: Falls $t = [a_1, b_1) \times \ldots [a_d, b_d)$ ein Cluster in $\mathcal{T}_\mathcal{I}$ ist, definieren wir die Mittelpunkte der Intervalle durch

$$m_\iota := \frac{b_\iota + a_\iota}{2} \qquad \text{für alle } \iota \in \{1, \ldots, d\}$$

und teilen die Intervalle in zwei Hälften

$$w_{\iota,0} := [a_\iota, m_\iota), \qquad w_{\iota,1} := [m_\iota, b_\iota) \qquad \text{für alle } \iota \in \{1, \ldots, d\}.$$

Kandidaten für die Söhne von t sind dann

$$t_\nu := w_{1,\nu_1} \times \ldots \times w_{d,\nu_d} \qquad \text{für alle } \nu \in \{0,1\}^d.$$

Wir definieren die Beschriftungen der Söhne durch

$$\hat{t}_\nu := \{i \in \mathcal{I} \ : \ \mathbf{x}_i \in t_\nu\} \qquad \text{für alle } \nu \in \{0,1\}^d$$

und definieren die Menge der Söhne des Clusters t durch

$$\operatorname{sons}(t) = \begin{cases} \{t_\nu \ : \ \hat{t}_\nu \neq \emptyset, \ \nu \in \{0,1\}^d\} & \text{falls } \#\hat{t} > k, \\ \emptyset & \text{ansonsten.} \end{cases}$$

Damit ist der Clusterbaum $\mathcal{T}_\mathcal{I}$ vollständig beschrieben. Da die Mengen t_ν eine disjunkte Partition der Menge t definieren, erfüllt dieser Baum alle Bedingungen der Definition 2.2. Die ersten drei Stufen eines derartigen Baums für eine zweidimensionale Punktmenge sind in Abbildung 2.7 dargestellt.

Bemerkung 2.30 (Eigenschaften) *Wir können leicht überprüfen, dass der soeben definierte Clusterbaum $\mathcal{T}_\mathcal{I}$ die folgenden Eigenschaften besitzt:*

- *Für jeden Cluster $t \in \mathcal{T}_\mathcal{I}$ gilt*

$$\hat{t} = \{i \in \mathcal{I} \ : \ \mathbf{x}_i \in t\}.$$

- *Für jeden Cluster $t \in \mathcal{T}_\mathcal{I}$ ist die Menge der Söhne beschränkt durch*

$$\operatorname{sons}(t) \leq 2^d.$$

- *Die Auflösung des Clusterbaums ist beschränkt durch*

$$\operatorname{res}(\mathcal{T}_\mathcal{I}) \leq k.$$

Für uns ausschlaggebend ist die Tatsache, dass die Durchmesser der Cluster, gemessen in der richtigen Norm, „schnell genug“ sinken. Im Interesse einer besonders einfachen

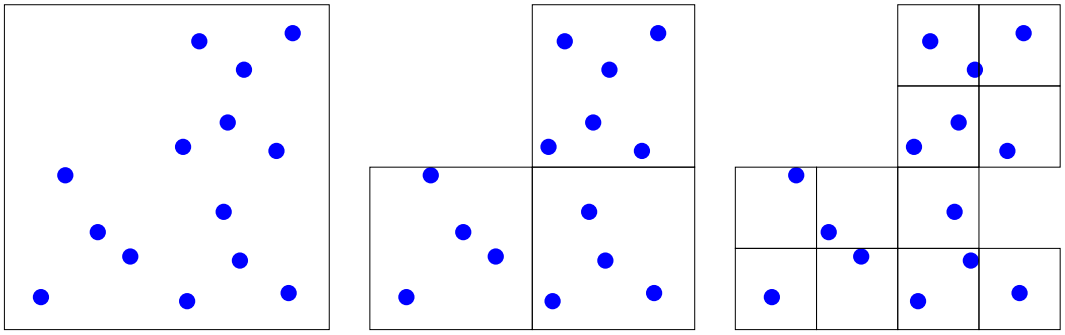

Abbildung 2.7: Erste drei Stufen des regelmäßigen Clusterbaums für eine zweidimensionale Punktmenge

Darstellung beschränken wir uns auf die Messung von Durchmesser und Abstand in der Maximumnorm, also auf

$$\begin{aligned}\operatorname{diam}_\infty(t) &= \max\{\|\mathbf{x}-\mathbf{y}\|_\infty \ : \ \mathbf{x},\mathbf{y}\in t\}\\ &= \max\{b_j - a_j \ : \ j\in\{1,\ldots,d\}\},\\ \operatorname{dist}_\infty(t,s) &= \min\{\|\mathbf{x}-\mathbf{y}\|_\infty \ : \ \mathbf{x}\in t,\mathbf{y}\in s\}\\ &= \max\{0,\hat{a}_j - b_j, a_j - \hat{b}_j \ : \ j\in\{1,\ldots,d\}\}\end{aligned}$$

für Cluster $t,s\in\mathcal{T}_\mathcal{I}$ mit

$$\begin{aligned}t &= [a_1,b_1)\times\ldots\times[a_d,b_d),\\ s &= [\hat{a}_1,\hat{b}_1)\times\ldots\times[\hat{a}_d,\hat{b}_d).\end{aligned}$$

Wir bezeichnen den Durchmesser der Wurzel mit

$$H := \operatorname{diam}_\infty(r) = b - a$$

und stellen fest, dass unsere Konstruktion bewirkt, dass der Sohn t' eines Clusters t exakt den halben Durchmesser seines Vaters besitzt, es gilt also

$$\operatorname{diam}_\infty(t') = \frac{1}{2}\operatorname{diam}_\infty(t) \qquad \text{für alle } t\in\mathcal{T}_\mathcal{I}, t'\in\operatorname{sons}(t).$$

Per Induktion folgt

$$\operatorname{diam}_\infty(t) = \left(\frac{1}{2}\right)^{\operatorname{level}(t)} H \qquad \text{für alle } t\in\mathcal{T}_\mathcal{I}. \tag{2.11}$$

Da wir vorausgesetzt haben, dass die Punkte $\mathbf{x}_i$ paarweise verschieden sind und die Indexmenge $\mathcal{I}$ endlich ist, ist

$$h := \min\{\|\mathbf{x}_i-\mathbf{x}_j\|_\infty \ : \ i,j\in\mathcal{I}, i\neq j\}$$

wohldefiniert und liegt im Intervall $(0, H]$. Die Idee der Abschätzung der Baumtiefe besteht nun darin, eine Stufe p im Baum zu finden, auf der alle Cluster so klein sind, dass sie nur noch einen Index enthalten können. Das ist der Fall, sobald der Durchmesser kleiner als der minimale Abstand h ist. Da die Cluster halboffen sind, genügt es auch schon, wenn der Durchmesser *gleich* h ist. Nach Konstruktion besitzen sie dann keine Söhne mehr, also muss p eine Schranke für die maximale Stufe sein. Die Gleichung (2.11) legt es nahe,

$$\left(\frac{1}{2}\right)^p H \leq h$$

zu wählen, also

$$p = \lceil \log_2 H/h \rceil,$$

wobei $\lceil \cdot \rceil$ durch

$$\lceil x \rceil := \min\{z \in \mathbb{Z} \ : \ z \geq x\} \qquad \text{für alle } x \in \mathbb{R}$$

definiert ist, denn dann gelten

$$2^p \geq 2^{\log_2 H/h} = H/h, \qquad \left(\frac{1}{2}\right)^p = 2^{-p} \leq h/H.$$

Für jedes $t \in \mathcal{T}_\mathcal{I}$ mit $\operatorname{level}(t) \geq p$ erhalten wir

$$\operatorname{diam}_\infty(t) \leq h$$

und damit $\#\hat{t} = 1$, also muss t ein Blatt sein. Demnach kann kein Cluster eine Stufe größer als p besitzen, denn dann müsste ein Vorfahre auf Stufe p existieren, der aber, wie soeben gezeigt, keine Söhne haben kann. Wir fassen dieses Ergebnis in der folgenden Bemerkung zusammen:

Bemerkung 2.31 (Baumtiefe) *Seien*

$$H := \max\{\|\mathbf{x}_i - \mathbf{x}_j\|_\infty \ : \ i, j \in \mathcal{I}\},$$
$$h := \min\{\|\mathbf{x}_i - \mathbf{x}_j\|_\infty \ : \ i, j \in \mathcal{I}, i \neq j\}$$

fixiert. Dann gilt

$$\operatorname{depth}(\mathcal{T}_\mathcal{I}) \leq p := \lceil \log_2 H/h \rceil.$$

In typischen Anwendungen sind die Punkte $(\mathbf{x}_i)_{i \in \mathcal{I}}$ ungefähr gleichmäßig verteilt, so dass $h \approx H n^{-1/d}$ und damit

$$p = \lceil \log_2 H/h \rceil \approx \lceil \log_2 n^{1/d} \rceil \approx \frac{1}{d} \log_2 n$$

gilt, so dass wir in Zukunft von $p \in \mathcal{O}(\log n)$ ausgehen dürfen, ohne uns zu weit von der Praxis zu entfernen.

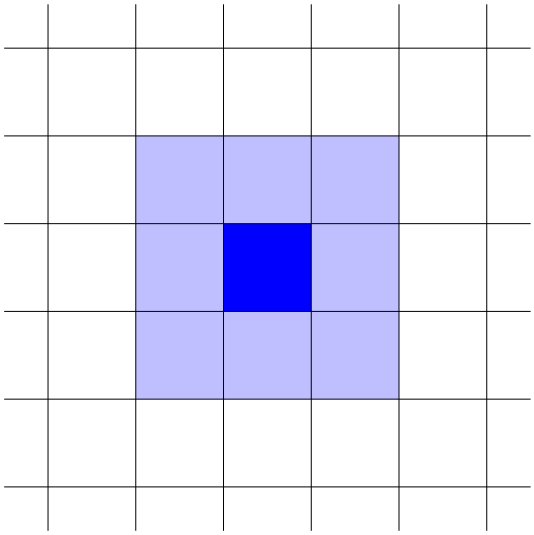

Abbildung 2.8: Bezüglich eines Clusters unzulässige 9 Cluster im zweidimensionalen Fall

Nun wenden wir uns der Konstante C_{sp} zu, die beschreibt, wie schwachbesetzt der Blockbaum $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ist. Diesen Baum konstruieren wir mit dem Algorithmus aus Abbildung 2.4 mit der Zulässigkeitsbedingung

$$\operatorname{diam}_\infty(t) \leq \operatorname{dist}_\infty(t,s)$$

und *ohne* strenge Konstruktion. Der Verzicht auf die strenge Konstruktion bietet den Vorteil, dass alle im Algorithmus auftretenden Clusterpaare (t,s) immer auf derselben Stufe des Clusterbaums liegen und sich die Beweisführung so wesentlich vereinfacht.

Die Beweisführung basiert auf der folgenden Idee: Falls $(t,s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ gilt, muss nach der Konstruktion des Algorithmus aus Abbildung 2.4 entweder $t = s = \operatorname{root}(\mathcal{T}_\mathcal{I})$ gelten, oder die Väter t^+ und s^+ von t und s können nicht zulässig gewesen sein, denn bei einem zulässigen Paar (t^+, s^+) hätte der Algorithmus dessen Söhne gar nicht weiter untersucht.

Wenn wir also wissen, wieviele *nicht* zulässige Cluster s^+ es zu einem Cluster t^+ gibt, können wir abschätzen, wieviele Cluster die Blockzeile des Clusters t enthält. Infolge unserer höchst regelmäßigen Konstruktion besitzen alle Cluster auf einer Stufe $\ell \in \mathbb{N}_0$ die Form

$$t = [a + (i_1 - 1)H2^{-\ell}, a + i_1 H 2^{-\ell}) \times \ldots \times [a + (i_d - 1)H2^{-\ell}, a + i_d H 2^{-\ell})$$

für $i_1, \ldots, i_d \in \{1, \ldots, 2^\ell\}$. Daraus folgt, dass zwei nicht unmittelbar benachbarte Cluster einen Abstand von mindestens $H2^{-\ell}$ besitzen müssen. Das entspricht gerade ihrem Durchmesser, also ist die Zulässigkeitsbedingung erfüllt. Somit können nur Cluster unzulässig sein, die sich berühren.

Davon existieren höchstens $\#\{-1,0,1\}^d = 3^d$ (siehe Abbildung 2.8), also folgt

$$\#\{s^+ \in \mathcal{T}_\mathcal{I} \ : \ (t^+, s^+) \text{ unzulässig}\} \leq 3^d.$$

Nun können wir die Mächtigkeit einer Blockzeile abschätzen: Für $t = \operatorname{root}(\mathcal{T}_\mathcal{I})$ gilt $\#\operatorname{row}(t) = 1$, für $t \in \mathcal{T}_\mathcal{I} \setminus \{\operatorname{root}(\mathcal{T}_\mathcal{I})\}$ dagegen erhalten wir

$$\begin{aligned}
\#\operatorname{row}(t) &= \#\{s \in \mathcal{T}_\mathcal{I} \ : \ (t,s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}\} \\
&= \#\{s \in \mathcal{T}_\mathcal{I} \ : \ (t^+, s^+) \text{ unzulässig}, t \in \operatorname{sons}(t^+), s \in \operatorname{sons}(s^+)\} \\
&= \# \bigcup_{\substack{s^+ \in \mathcal{T}_\mathcal{I} \\ (t^+,s^+) \text{ unzulässig}}} \operatorname{sons}(s^+) = \sum_{\substack{s^+ \in \mathcal{T}_\mathcal{I} \\ (t^+,s^+) \text{ unzulässig}}} \#\operatorname{sons}(s^+) \\
&\leq \sum_{\substack{s^+ \in \mathcal{T}_\mathcal{I} \\ (t^+,s^+) \text{ unzulässig}}} 2^d \leq 6^d.
\end{aligned}$$

Entsprechend können wir mit den Blockspalten verfahren und erhalten die folgende Feststellung:

Bemerkung 2.32 (Schwachbesetztheit) *In diesem Modellfall gilt*

$$\#\operatorname{row}(t) \leq 6^d, \qquad \#\operatorname{col}(s) \leq 6^d \qquad \text{für alle } t, s \in \mathcal{T}_\mathcal{I},$$

also erfüllt $C_{sp} := 6^d$ die Bedingungen von Definition 2.24.

Die schwachbesetzte Struktur des Blockbaums hängt also in diesem Fall ausschließlich von der Dimension des zugrundeliegenden geometrischen Raums ab, nicht von der Verteilung der Punkte oder auch nur deren Anzahl.

In unserem Modellfall nimmt die Abschätzung aus Satz 2.29 die Form

$$\approx \frac{2}{d} 6^d k n \lceil \log_2 n + 1 \rceil$$

an, liegt also in $\mathcal{O}(nk \log n)$ und ist damit mit den Abschätzungen für den eindimensionalen Fall (siehe [25]) durchaus vergleichbar, obwohl die Herleitung mit Hilfe eines allgemeinen Clusterbaums und eines allgemeinen schwachbesetzten Blockbaums wesentlich flexibler als die im eindimensionalen Fall benutzte einfache Rekursion ist.

3 Kompression durch Interpolation

In praktischen Anwendungen muss häufig eine Matrix behandelt werden, die mit Hilfe einer *Kernfunktion* definiert ist. Eine Kernfunktion auf Mengen $\Omega \subseteq \mathbb{R}^d$ ist eine Abbildung

$$g : \Omega \times \Omega \to \mathbb{K}^m, \qquad \mathbb{K} \in \{\mathbb{R}, \mathbb{C}\},$$

die einem Paar von Punkten $x, y \in \Omega$ einen Wert $g(x, y)$ zuordnet. Wir beschränken uns in der Regel auf den Fall $m = 1$, auf den sich der allgemeinere Fall zurückführen lässt, indem man die Komponenten von g einzeln behandelt.

Um g eine Matrix $\mathbf{G} \in \mathbb{K}^{\mathcal{I}\times\mathcal{J}}$ zuzuordnen, können beispielsweise Punktemengen $(x_i)_{i\in\mathcal{I}}$ und $(y_j)_{j\in\mathcal{J}}$ verwendet und dann die Koeffizienten durch

$$G_{ij} := g(x_i, y_j) \qquad \text{für alle } i \in \mathcal{I},\ j \in \mathcal{J} \tag{3.1}$$

definiert werden. In der Praxis wird die Auswertung von g in einem Punkt häufig durch die Berechnung geeigneter Integralmittel ersetzt, für die Diskussion der grundlegenden Ansätze genügen uns zunächst Ansätze der Form (3.1).

3.1 Entartete Kernfunktionen

Um die Matrix $\mathbf{G}$ durch eine hierarchische Matrix approximieren zu können, müssen wir für zulässige Blöcke $b = (t, s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ Matrizen $\mathbf{A}_b \in \mathbb{K}^{\hat{t}\times k}$ und $\mathbf{B}_b \in \mathbb{K}^{\hat{s}\times k}$ konstruieren, deren Produkt $\mathbf{G}|_{\hat{t}\times\hat{s}}$ approximiert, es sollte also

$$\mathbf{G}|_{\hat{t}\times\hat{s}} \approx \mathbf{A}_b\mathbf{B}_b^*$$

gelten. Da $\mathbf{G}$ eng mit der Kernfunktion g zusammenhängt, bietet es sich an, nach einem Gegenstück dieser Rang-k-Approximation im Kontext der Funktionen auf $\Omega \times \Omega$ zu suchen.

Definition 3.1 (Entartete Kernfunktion) *Sei $g : \Omega_t \times \Omega_s \to \mathbb{K}$ eine Kernfunktion. Falls $a_\nu : \Omega_t \to \mathbb{K}$ und $b_\nu : \Omega_s \to \mathbb{K}$ für alle $\nu \in \{1, \ldots, k\}$ so existieren, dass*

$$g(x, y) = \sum_{\nu=1}^{k} a_\nu(x)\bar{b}_\nu(y) \qquad \text{für alle } x \in \Omega_t,\ y \in \Omega_s$$

gilt, nennen wir g entartete Kernfunktion *mit Rang k.*

Die entscheidende Eigenschaft der entarteten Kernfunktion besteht darin, dass die Variablen x und y entkoppelt sind. Falls g entartet ist, gilt nämlich

$$G_{ij} = g(x_i, y_j) = \sum_{\nu=1}^{k} a_\nu(x_i)\bar{b}_\nu(y_j) \qquad \text{für alle } i \in \mathcal{I},\ j \in \mathcal{J},$$

und mit den durch

$$A_{i\nu} := a_\nu(x_i), \qquad B_{j\nu} := b_\nu(y_j) \qquad \text{für alle } i \in \mathcal{I},\ j \in \mathcal{J},\ \nu \in \{1, \ldots, k\} \tag{3.2}$$

definierten Matrizen $\mathbf{A} \in \mathbb{K}^{\mathcal{I}\times k}$ und $\mathbf{B} \in \mathbb{K}^{\mathcal{J}\times k}$ erhalten wir

$$G_{ij} = \sum_{\nu=1}^{k} A_{i\nu}\bar{B}_{j\nu} = (\mathbf{A}\mathbf{B}^*)_{ij} \qquad \text{für alle } i \in \mathcal{I},\ j \in \mathcal{J}.$$

In der Regel dürfen wir nicht darauf hoffen, dass g entartet ist, aber in der Praxis lässt sich g häufig zumindest auf geeigneten Teilgebieten durch eine entartete Funktion approximieren.

Diese Funktion kann beispielsweise mit Hilfe der Taylor-Entwicklung konstruiert werden: Sei $b = (t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ ein zulässiger Block, und seien $\Omega_t, \Omega_s \subseteq \Omega$ mit

$$x_i \in \Omega_t, \qquad y_j \in \Omega_s \qquad \text{für alle } i \in \hat{t},\ j \in \hat{s} \tag{3.3}$$

gegeben. Falls g auf $\Omega_t \times \Omega_s$ analytisch ist, können wir einen Entwicklungspunkt ξ_t wählen und die Kernfunktion durch

$$g(x, y) = \sum_{\nu} \frac{(x - \xi_t)^\nu}{\nu!}\partial_\nu g(\xi_t, y) \qquad \text{für alle } x \in \Omega_t,\ y \in \Omega_s$$

darstellen. Indem wir die unendliche Taylor-Reihe durch ein Anfangsstück ersetzen, erhalten wir

$$g(x, y) \approx \sum_{\nu\le m} \frac{(x - \xi_t)^\nu}{\nu!}\partial_\nu g(\xi_t, y) \qquad \text{für alle } x \in \Omega_t,\ y \in \Omega_s.$$

Da ξ_t nicht von x oder y abhängt, können wir

$$a_\nu(x) := \frac{(x - \xi_t)^\nu}{\nu!}, \qquad b_\nu(y) := \overline{\partial_\nu g}(\xi_t, y) \qquad \text{für alle } x \in \Omega_t,\ y \in \Omega_s$$

setzen und haben die gesuchte entartete Approximation gefunden.

3.2 Interpolation

Die Konstruktion einer entarteten Approximation mit Hilfe der Taylor-Entwicklung ersetzt in $g_{y,\nu}$ die erste Variable x durch den festen Entwicklungspunkt ξ_t, der zu einer Entkopplung von x und y führt.

Da wir ohne die für die Taylor-Entwicklung erforderlichen Ableitungen auskommen wollen, modifizieren wir diesen Ansatz: Statt eines einzelnen Punkts ξ_t erlauben wir uns k Punkte $\xi_{t,\nu}$ und suchen nach einer Näherung der Form

$$g(x,y) \approx \sum_{\nu} \mathcal{L}_{t,\nu}(x) g(\xi_{t,\nu}, y) \qquad \text{für alle } x \in \Omega_t,\ y \in \Omega_s.$$

Diese Gestalt sollte uns bekannt vorkommen: In dieser Form werden Lagrange-Interpolanten in Stützstellen $\xi_{t,\nu}$ dargestellt, die Funktionen $\mathcal{L}_{t,\nu}$ sind dann die passenden Lagrange-Basisfunktionen.

Um diese Idee praktisch umsetzen zu können benötigen wir eine Technik, um Stützstellen und Lagrange-Basisfunktionen auf mehrdimensionalen Gebieten zu konstruieren. Wir beschränken uns hier auf den besonders einfachen Fall eines Tensorgebiets: Sei $Q_t = [a_1, b_1] \times \ldots \times [a_d, b_d]$ mit

$$a_\iota < b_\iota \qquad \text{für alle } \iota \in \{1, \ldots, d\}.$$

Wir werden zunächst systematisch Interpolationsverfahren auf den Intervallen $[a_\iota, b_\iota]$ konstruieren, die wir dann zu einem Verfahren auf Q_t kombinieren können.

Den Ausgangspunkt unserer Untersuchung bildet ein Interpolationsverfahren auf dem Referenzintervall $[-1, 1]$: Seien $m \in \mathbb{N}$ und paarweise verschiedene $\xi_0, \ldots, \xi_m \in [-1, 1]$ gegeben, seien

$$\mathcal{L}_\nu(x) := \prod_{\substack{\mu=0 \\ \mu \neq \nu}}^{m} \frac{x - \xi_\mu}{\xi_\nu - \xi_\mu} \qquad \text{für alle } x \in [-1, 1]$$

die passenden Lagrange-Basispolynome mit der Eigenschaft

$$\mathcal{L}_\nu(\xi_\mu) = \delta_{\nu\mu} = \begin{cases} 1 & \text{falls } \nu = \mu, \\ 0 & \text{ansonsten} \end{cases} \qquad \text{für alle } \nu, \mu \in \{0, \ldots, m\}.$$

Dann ist

$$\mathfrak{I} : C[-1,1] \to \Pi_m, \qquad f \mapsto \sum_{\nu=0}^{m} f(\xi_\nu) \mathcal{L}_\nu,$$

ein Lagrange-Interpolationsoperator, der auf $[-1, 1]$ stetige Funktionen auf Polynome aus Π_m, also aus dem Raum der Polynom eines Grades von höchstens m, abbildet.

Um diesen Operator auf Funktionen auf dem Intervall $[a, b]$ mit $a < b$ zu übertragen führen wir die Transformation

$$\Phi_{[a,b]} : [-1,1] \to [a,b], \qquad x \mapsto \frac{b+a}{2} + \frac{b-a}{2} x,$$

ein, die die Eigenschaften

$$\Phi_{[a,b]}(-1) = a, \qquad \Phi_{[a,b]}(1) = b, \qquad \Phi'_{[a,b]}(x) = \frac{b-a}{2} > 0 \qquad \text{für alle } x \in [-1,1]$$

besitzt, also insbesondere affin und bijektiv ist. Wenn nun $f \in C[a,b]$ eine stetige Funktion auf $[a,b]$ ist, muss $\hat{f} := f \circ \Phi_{[a,b]}$ eine stetige Funktion auf $[-1,1]$ sein, auf die wir den dort definierten Interpolationsoperator $\mathfrak{I}$ anwenden können, um einen Interpolanten $\hat{p} \in \Pi_m$ zu gewinnen. Da dieser Interpolant $\hat{f}$ approximiert, muss $p := \hat{p} \circ \Phi_{[a,b]}^{-1}$ auch $f = \hat{f} \circ \Phi_{[a,b]}^{-1}$ approximieren.

Ein Interpolationsoperator auf dem Intervall $[a,b]$ kann also durch

$$\mathfrak{I}_{[a,b]} : C[a,b] \to \Pi_m, \qquad f \mapsto (\mathfrak{I}[f \circ \Phi_{[a,b]}]) \circ \Phi_{[a,b]}^{-1},$$

definiert werden. Diese Darstellung ist zwar für die Theorie nützlich, für die Praxis aber eher ungeeignet, so dass wir an einer Alternative interessiert sind. Dazu setzen wir in die Definition von $\mathfrak{I}_{[a,b]}$ die Darstellung des Operators $\mathfrak{I}$ mit Lagrange-Polynomen ein:

$$\mathfrak{I}_{[a,b]}[f] = \sum_{\nu=0}^{m} f \circ \Phi_{[a,b]}(\xi_\nu) \mathcal{L}_\nu \circ \Phi_{[a,b]}^{-1} = \sum_{\nu=0}^{m} f(\Phi_{[a,b]}(\xi_\nu)) \mathcal{L}_\nu \circ \Phi_{[a,b]}^{-1}.$$

Da f nur in den durch

$$\xi_{[a,b],\nu} := \Phi_{[a,b]}(\xi_\nu) = \frac{b+a}{2} + \frac{b-a}{2}\xi_\nu \qquad \text{für alle } \nu \in \{0,\dots,m\}$$

gegebenen transformierten Stützstellen ausgewertet wird, bietet es sich an, nach passenden Lagrange-Polynomen zu suchen. Naheliegend ist die Wahl

$$\mathcal{L}_{[a,b],\nu}(x) := \prod_{\substack{\mu=0 \\ \mu \neq \nu}}^{m} \frac{x - \xi_{[a,b],\mu}}{\xi_{[a,b],\nu} - \xi_{[a,b],\mu}} \qquad \text{für alle } x \in [a,b],$$

die wir allerdings noch mit den in der Definition von $\mathfrak{I}_{[a,b]}$ verwendeten Polynomen $\mathcal{L}_\nu \circ \Phi_{[a,b]}^{-1}$ in Verbindung bringen müssen. Nach Definition gilt

$$\begin{aligned} \mathcal{L}_\nu \circ \Phi_{[a,b]}^{-1}(\xi_{[a,b],\mu}) &= \mathcal{L}_\nu(\Phi_{[a,b]}^{-1}(\Phi_{[a,b]}(\xi_\mu))) = \mathcal{L}_\nu(\xi_\mu) = \delta_{\nu\mu} \\ &= \mathcal{L}_{[a,b],\nu}(\xi_{[a,b],\mu}) \qquad \text{für alle } \nu,\mu \in \{0,\dots,m\}, \end{aligned}$$

also stimmen $\mathcal{L}_\nu \circ \Phi_{[a,b]}^{-1}$ und $\mathcal{L}_{[a,b],\nu}$ in $m+1$ paarweise verschiedenen Punkten überein. Da beides Polynome in Π_m sind, müssen sie identisch sein, und es folgt

$$\mathfrak{I}_{[a,b]}[f] = \sum_{\nu=0}^{m} f(\xi_{[a,b],\nu}) \mathcal{L}_{[a,b],\nu} \qquad \text{für alle } f \in C[a,b].$$

Diese Darstellung eignet sich besser für den praktischen Einsatz: Wir können in einem Vorbereitungsschritt die Punkte $\xi_{[a,b],\nu}$ berechnen und dann die üblichen Verfahren wie das Neville-Aitken-Schema oder Newtons dividierte Differenzen verwenden.

Unsere nächste Aufgabe besteht darin, Interpolationsoperatoren auf der Menge $Q = [a_1,b_1] \times \ldots \times [a_d,b_d]$ zu konstruieren. Sei $f \in C(Q)$ eine zu interpolierende Funktion. Für gegebene $\iota \in \{1,\dots,d\}$ und $x \in Q$ können wir aus f die Funktion

$$f_{x,\iota} : [a_\iota, b_\iota] \to \mathbb{K}, \qquad y \mapsto f(x_1,\dots,x_{\iota-1},y,x_{\iota+1},\dots,x_d),$$

gewinnen, bei der wir alle Parameter bis auf den ι-ten festhalten und so eine Abbildung auf dem Intervall $[a_\iota, b_\iota]$ erhalten. Auf diese Funktion können wir den Interpolationsoperator $\mathfrak{I}_{[a_\iota,b_\iota]}$ anwenden und erhalten

$$\mathfrak{I}_{Q,\iota} : C(Q) \to C(Q)$$

mit

$$\mathfrak{I}_{Q,\iota}[f](x) = \sum_{\nu=0}^{m} f(x_1, \ldots, x_{\iota-1}, \xi_{[a_\iota,b_\iota],\nu}, x_{\iota+1}, \ldots, x_d)\mathcal{L}_{[a_\iota,b_\iota],\nu}(x_\iota)$$
$$\text{für alle } f \in C(Q), x \in Q.$$

Dieser Operator bildet eine stetige Funktion auf eine neue stetige Funktion ab, die sich in der ι-ten Komponente wie ein Polynom verhält. Indem wir mehrere derartige Operatoren miteinander kombinieren, erhalten wir ein Polynom in allen Komponenten: Der Operator

$$\mathfrak{I}_Q := \mathfrak{I}_{Q,1} \circ \ldots \circ \mathfrak{I}_{Q,d}$$

erfüllt die Gleichung

$$\mathfrak{I}_Q[f](x) = \sum_{\nu_1=0}^{m} \ldots \sum_{\nu_d=0}^{m} f(\xi_{[a_1,b_1],\nu_1}, \ldots \xi_{[a_d,b_d],\nu_d})\mathcal{L}_{[a_1,b_1],\nu_1}(x_1) \ldots \mathcal{L}_{[a_d,b_d],\nu_d}(x_d)$$
$$\text{für alle } f \in C(Q), x \in Q.$$

Wir kürzen diese Formel ab, indem wir die d-dimensionalen Stützstellen

$$\xi_{Q,\nu} := (\xi_{[a_1,b_1],\nu_1}, \ldots \xi_{[a_d,b_d],\nu_d}), \qquad \text{für alle } \nu \in M := \{0, \ldots, m\}^d$$

und die entsprechenden Lagrange-Polynome $\mathcal{L}_{Q,\nu}$ mit

$$\mathcal{L}_{Q,\nu}(x) := \mathcal{L}_{[a_1,b_1],\nu_1}(x_1) \ldots \mathcal{L}_{[a_d,b_d],\nu_d}(x_d) \qquad \text{für alle } \nu \in M, x \in Q$$

einführen und die gewohnte Darstellung

$$\mathfrak{I}_Q[f] = \sum_{\nu \in M} f(\xi_{Q,\nu})\mathcal{L}_{Q,\nu} \qquad \text{für alle } f \in C(Q)$$

erhalten. Derartige Interpolationsoperatoren, die sich aus einer Kombination eindimensionaler Operatoren ergeben, bezeichnet man als *Tensor-Interpolationsoperatoren*.

Wir können Interpolationsoperatoren verwenden, um eine entartete Approximation einer Kernfunktion zu konstruieren: Wir fixieren überdeckende Quader Q_t und Q_s für zwei Cluster $t \in \mathcal{T}_\mathcal{I}$ und $s \in \mathcal{T}_\mathcal{J}$ und approximieren eine Kernfunktion

$$g : Q_t \times Q_s \to \mathbb{K}$$

durch den Interpolanten in der ersten Komponente

$$\tilde{g}(x, y) := \sum_{\nu \in M} \mathcal{L}_{Q_t,\nu}(x) g(\xi_{Q_t,\nu}, y)$$

oder den Interpolanten in der zweiten Komponente

$$\tilde{g}(x,y) := \sum_{\mu \in M} g(x, \xi_{Q_s,\mu}) \mathcal{L}_{Q_s,\mu}(y).$$

In beiden Fällen erhalten wir offenbar eine entartete Approximation mit einem Rang von höchstens $k := \#M = (m+1)^d$.

Eine Approximation eines Blocks $b = (t,s)$ der durch (3.1) definierten Matrix $\mathbf{G}$ erhalten wir, wenn wir im ersten Fall

$$(A_b)_{i\nu} := \mathcal{L}_{Q_t,\nu}(x_i), \qquad (B_b)_{j\nu} := \bar{g}(\xi_{Q_t,\nu}, y_j) \qquad \text{für alle } i \in \hat{t},\ j \in \hat{s}, \nu \in M$$

und im zweiten Fall

$$(A_b)_{i\mu} := g(x_i, \xi_{Q_s,\mu}), \qquad (B_b)_{j\mu} := \bar{\mathcal{L}}_{Q_s,\mu}(y_j) \qquad \text{für alle } i \in \hat{t}, j \in \hat{s}, \mu \in M$$

setzen und $\mathbf{G}|_{\hat{t}\times\hat{s}} \approx \mathbf{A}_b \mathbf{B}_b^*$ verwenden. In beiden Fällen genügen uns Auswertungen der Kernfunktion g und der Lagrange-Polynome, um die Matrizen zu konstruieren. Beide Berechnungen lassen sich einfach und effizient durchführen.

3.3 Fehlerabschätzung

Mit der bloßen Konstruktion eines Interpolanten ist es natürlich nicht getan, wir müssen auch untersuchen, wie gut die durch diesen Interpolanten gegebene Approximation der ursprünglichen Funktion ist.

Die Fehleranalyse bauen wir auf dem folgenden wohlbekannten Resultat auf:

Lemma 3.2 (Interpolationsfehler) *Sei $f \in C^{m+1}[-1,1]$, und sei $x \in [-1,1]$. Dann existiert ein $\eta \in [-1,1]$ mit*

$$f(x) - \mathfrak{I}[f](x) = (x-\xi_0)\dots(x-\xi_m)\frac{f^{(m+1)}(\eta)}{(m+1)!}.$$

Beweis. (siehe [35]) Für $x \in \{\xi_0,\dots,\xi_m\}$ folgt die Aussage aus der Interpolationseigenschaft. Für $x \in [-1,1] \setminus \{\xi_0,\dots,\xi_m\}$ untersuchen wir die Funktion

$$g : [-1,1] \to \mathbb{K}, \qquad y \mapsto f(y) - \mathfrak{I}[f](y) - R(y-\xi_0)\dots(y-\xi_m),$$

bei der wir $R \in \mathbb{K}$ so wählen, dass $g(x) = 0$ gilt. Damit besitzt g Nullstellen in $\xi_0,\dots,\xi_m$ und x, also existieren nach Voraussetzung $m+2$ Nullstellen. Nach dem Satz von Rolle muss dann $g^{(m+1)}$ mindestens eine Nullstelle η besitzen, und für diese Nullstelle gilt

$$0 = g^{(m+1)}(\eta) = f^{(m+1)}(\eta) - R(m+1)! \quad .$$

Wir lösen nach R auf und erhalten die gewünschte Gleichung. ∎

Indem wir das *Stützstellenpolynom* $\omega \in \Pi_{m+1}$ durch

$$\omega(x) = (x - \xi_0) \dots (x - \xi_m) \qquad \text{für alle } x \in [-1, 1]$$

definieren und das Maximum der rechten Seite der Gleichung aus Lemma 3.2 verwenden, erhalten wir die Abschätzung

$$\|f - \mathfrak{I}[f]\|_{\infty,[-1,1]} \leq \|\omega\|_{\infty,[-1,1]} \frac{\|f^{(m+1)}\|_{\infty,[-1,1]}}{(m+1)!} \qquad \text{für alle } f \in C^{m+1}[-1, 1].$$

Aus der Definition des Stützstellenpolynoms ω folgt bereits die einfache Abschätzung $\|\omega\|_{\infty,[-1,1]} \leq 2^{m+1}$, die für jede beliebige Wahl von Interpolationspunkten gilt.

Wesentlich besser ist die Wahl von *Tschebyscheff-Punkten*:

Definition 3.3 (Tschebyscheff-Polynome) *Durch*

$$T_n(x) = \begin{cases} 1 & \text{falls } n = 0, \\ x & \text{falls } n = 1, \\ 2xT_{n-1}(x) - T_{n-2}(x) & \text{ansonsten} \end{cases} \qquad \text{für alle } n \in \mathbb{N}_0, x \in \mathbb{K}$$

wird eine Familie $(T_n)_{n\in\mathbb{N}_0}$ von Polynomen definiert. Für jedes $n \in \mathbb{N}_0$ bezeichnen wir T_n als das n-te Tschebyscheff-Polynom.

Die folgenden Eigenschaften der Tschebyscheff-Polynome lassen sich mit Hilfe der Additionstheoreme für trigonometrische Funktionen leicht nachprüfen:

Lemma 3.4 *Sei $n \in \mathbb{N}_0$. Dann gilt*

$$T_n(x) = \cos(n \arccos(x)) \qquad \text{für alle } x \in [-1, 1].$$

Daraus folgen insbesondere $\|T_n\|_{\infty,[-1,1]} = 1$ und

$$T_n(\hat{\xi}_\nu) = 0, \qquad \hat{\xi}_\nu := \cos\left(\pi \frac{2\nu + 1}{2n}\right) \qquad \text{für alle } \nu \in \{0, \dots, n-1\},$$

also besitzt T_n gerade n reelle einfache Nullstellen, die im Intervall $[-1, 1]$ liegen.

Beweis. Die alternative Darstellung des Tschebyscheff-Polynoms folgt, indem man das Additionstheorem

$$\cos(\alpha + \beta) = \cos\alpha \cos\beta - \sin\alpha \sin\beta$$

auf T_{n+1} und T_{n-1} anwendet und die resultierenden Terme vergleicht.

Die restlichen Aussagen folgen direkt aus den Eigenschaften des Cosinus. ∎

Das $(m+1)$-te Tschebyscheff-Polynom gehört zu Π_{m+1}, und sein $(m+1)$-ter Koeffizient beträgt 2^m. Deshalb können wir $\hat{\omega} \in \Pi_{m+1}$ durch

$$\hat{\omega}(x) := 2^{-m} T_{m+1}(x) \qquad \text{für alle } x \in [-1, 1]$$

definieren und wissen, dass sein $(m+1)$-ter Koeffizient eins betragen muss. Also ist $\hat{\omega}$ ein normiertes Polynom der Ordnung $m+1$, wie auch das Stützstellenpolynom ω.

Da uns die Nullstellen von T_{m+1}, und damit auch die von $\hat{\omega}$, aus Lemma 3.4 bekannt sind, erhalten wir mit

$$\hat{\xi}_\nu := \cos\left(\pi \frac{2\nu+1}{2m+2}\right) \qquad \text{für alle } \nu \in \{0,\ldots,m\}$$

die Darstellung

$$\hat{\omega}(x) = (x-\hat{\xi}_0)\ldots(x-\hat{\xi}_m) \qquad \text{für alle } x \in [-1,1],$$

also ist $\hat{\omega}$ das Stützstellenpolynom zu den Interpolationspunkten $\hat{\xi}_0,\ldots,\hat{\xi}_m$. Aus Lemma 3.4 folgt

$$\|\hat{\omega}\|_{\infty,[-1,1]} = 2^{-m},$$

und man kann beweisen, dass kein normiertes Polynom aus Π_{m+1} ein geringeres Maximum besitzen kann.

In dieser Hinsicht sind die *Tschebyscheff-Interpolationspunkte* $\hat{\xi}_0,\ldots,\hat{\xi}_m$ die bestmögliche Wahl. Wir werden deshalb im Folgenden davon ausgehen, dass der Interpolationsoperator $\mathfrak{I}$ auf dem Intervall $[-1,1]$ diese Punkte verwendet und deshalb die Abschätzung

$$\|f-\mathfrak{I}[f]\|_{\infty,[-1,1]} \leq 2^{-m}\frac{\|f^{(m+1)}\|_{\infty,[-1,1]}}{(m+1)!} \qquad \text{für alle } f \in C^{m+1}[-1,1] \tag{3.4}$$

gilt. In diesem Fall heißt $\mathfrak{I}$ der *Tschebyscheff-Interpolationsoperator*.

Da wir bei der Konstruktion des mehrdimensionalen Interpolationsoperators $\mathfrak{I}_Q$ mehrere eindimensionale Interpolationsoperatoren miteinander verketten, benötigen wir auch eine Aussage darüber, wie sich bei einer solchen Verkettung die Norm verhält.

Definition 3.5 (Stabilitätskonstante) *Sei $\Lambda \in \mathbb{R}$. Falls*

$$\|\mathfrak{I}[f]\|_{\infty,[-1,1]} \leq \Lambda\|f\|_{\infty,[-1,1]} \qquad \textit{für alle } f \in C[-1,1] \tag{3.5}$$

gilt, heißt Λ Stabilitätskonstante *des Interpolationsoperators $\mathfrak{I}$.*

Die bestmögliche Stabilitätskonstante lässt sich explizit angeben:

Lemma 3.6 (Lebesgue-Zahl) *Die Zahl*

$$\Lambda := \max\left\{\sum_{\nu=0}^{m}|\mathcal{L}_\nu(x)| \;:\; x \in [-1,1]\right\}$$

heißt Lebesgue-Zahl *des Interpolationsoperators $\mathfrak{I}$. Sie ist die kleinste mögliche Stabilitätskonstante des Operators.*

Beweis. Sei $f \in C[-1,1]$. Dann gilt für alle $x \in [-1,1]$ die Abschätzung

$$|\mathfrak{I}[f](x)| = \left| \sum_{\nu=0}^{m} f(\xi_\nu)\mathcal{L}_\nu(x) \right| \leq \sum_{\nu=0}^{m} |f(\xi_\nu)|\ |\mathcal{L}_\nu(x)|$$
$$\leq \|f\|_{\infty,[-1,1]} \sum_{\nu=0}^{m} |\mathcal{L}_\nu(x)| \leq \Lambda \|f\|_{\infty,[-1,1]},$$

also muss Λ eine Stabilitätskonstante sein.

Zum Nachweis der Optimalität wählen wir einen Punkt $x \in [-1,1]$, für den

$$\sum_{\nu=0}^{m} |\mathcal{L}_\nu(x)| = \Lambda$$

gilt. Wir können, beispielsweise durch stückweise lineare Interpolation, eine stetige Funktion $f \in C[-1,1]$ konstruieren, die

$$f(\xi_\nu) = \begin{cases} 1 & \text{falls } \mathcal{L}_\nu(x) \geq 0, \\ -1 & \text{ansonsten} \end{cases} \qquad \text{für alle } \nu \in \{0,\dots,m\}$$

und $\|f\|_{\infty,[-1,1]} = 1$ erfüllt, und für diese Funktion gilt $\|\mathfrak{I}[f]\|_{\infty,[-1,1]} \geq \Lambda \|f\|_{\infty,[-1,1]}$. ∎

Auch in Hinblick auf die Stabilität ist die Tschebyscheff-Interpolation vorteilhaft: In [30] wird bewiesen, dass

$$\widehat{\Lambda} := \frac{2}{\pi} \log(m+1) + 1 \tag{3.6}$$

eine Stabilitätskonstante für diesen Interpolationsoperator ist und dass andere Interpolationspunkte zu nur unwesentlich kleineren Stabilitätskonstanten führen können.

Sowohl die Fehlerabschätzung (3.4) als auch die Stabilitätsabschätzung (3.5) übertragen sich auf den Fall des transformierten Interpolationsoperators $\mathfrak{I}_{[a,b]}$:

Lemma 3.7 (Transformierter Interpolationsoperator) *Sei $\mathfrak{I}$ der Tschebyscheff-Interpolationsoperator, und sei $\mathfrak{I}_{[a,b]}$ der zugehörige transformierte Interpolationsoperator auf dem Intervall $[a,b]$. Dann gelten*

$$\|f - \mathfrak{I}_{[a,b]}[f]\|_{\infty,[a,b]} \leq 2\left(\frac{b-a}{4}\right)^{m+1} \frac{\|f^{(m+1)}\|_{\infty,[a,b]}}{(m+1)!} \qquad \text{für alle } f \in C^{m+1}[a,b],$$
$$\|\mathfrak{I}_{[a,b]}[f]\|_{\infty,[a,b]} \leq \Lambda \|f\|_{\infty,[a,b]} \qquad \text{für alle } f \in C[a,b].$$

Beweis. Da die Transformation $\Phi_{[a,b]}$ bijektiv ist, gilt

$$\|f - \mathfrak{I}_{[a,b]}[f]\|_{\infty,[a,b]} = \|f \circ \Phi_{[a,b]} - \mathfrak{I}_{[a,b]}[f] \circ \Phi_{[a,b]}\|_{\infty,[-1,1]} = \|\hat{f} - \mathfrak{I}[\hat{f}]\|_{\infty,[-1,1]}$$

mit $\hat{f} := f \circ \Phi_{[a,b]}$ nach Definition von $\mathfrak{I}_{[a,b]}$. Aus (3.4) folgt mit

$$\|(f \circ \Phi_{[a,b]})'\|_{\infty,[-1,1]} = \|f' \circ \Phi_{[a,b]} \Phi'_{[a,b]}\|_{\infty,[-1,1]} = \frac{b-a}{2} \|f' \circ \Phi_{[a,b]}\|_{\infty,[-1,1]},$$

$$\|(f \circ \Phi_{[a,b]})^{(m+1)}\|_{\infty,[-1,1]} = \left(\frac{b-a}{2}\right)^{m+1} \|f^{(m+1)} \circ \Phi_{[a,b]}\|_{\infty,[-1,1]},$$
$$\|\hat{f}^{(m+1)}\|_{\infty,[-1,1]} = \left(\frac{b-a}{2}\right)^{m+1} \|f^{(m+1)}\|_{\infty,[a,b]}$$

die gesuchte Abschätzung des Fehlers.

Für die Stabilitätsabschätzung erhalten wir

$$\|\mathfrak{I}_{[a,b]}[f]\|_{\infty,[a,b]} = \|\mathfrak{I}[\hat{f}]\|_{\infty,[-1,1]} \leq \Lambda \|\hat{f}\|_{\infty,[-1,1]} = \Lambda \|f\|_{\infty,[a,b]},$$

also das gewünschte Resultat. ∎

Entsprechend können wir nun auch die „partiellen“ Interpolationsoperatoren $\mathfrak{I}_{Q,\iota}$ behandeln:

Lemma 3.8 (Partielle Interpolation) *Sei $\iota \in \{1, \ldots, d\}$ gegeben. Dann gelten*

$$\|f - \mathfrak{I}_{Q,\iota}[f]\|_{\infty,Q} \leq 2\left(\frac{b_\iota - a_\iota}{4}\right)^{m+1} \frac{\|\partial_\iota^{m+1} f\|_{\infty,Q}}{(m+1)!} \qquad \text{für alle } f \in C^{m+1}(Q), \tag{3.7}$$
$$\|\mathfrak{I}_{Q,\iota}[f]\|_{\infty,Q} \leq \Lambda \|f\|_{\infty,Q} \qquad \text{für alle } f \in C(Q). \tag{3.8}$$

Beweis. Sei $f \in C(Q)$ gegeben, und sei $x \in Q$. Wir definieren

$$f_{x,\iota} : [a_\iota, b_\iota] \to \mathbb{K}, \qquad y \mapsto f(x_1, \ldots, x_{\iota-1}, y, x_{\iota+1}, \ldots, x_d),$$

und stellen fest, dass nach unserer Definition

$$f(x) = f_{x,\iota}(x_\iota), \qquad \mathfrak{I}_{Q,\iota}[f](x) = \mathfrak{I}_{[a_\iota,b_\iota]}[f_{x,\iota}](x_\iota)$$

gelten. Aus Lemma 3.7 folgt

$$|\mathfrak{I}_{Q,\iota}[f](x)| = |\mathfrak{I}_{[a_\iota,b_\iota]}[f_{x,\iota}](x_\iota)| \leq \Lambda \|f_{x,\iota}\|_{\infty,[a_\iota,b_\iota]},$$

und dank

$$\begin{aligned}\|f_{x,\iota}\|_{\infty,[a_\iota,b_\iota]} &= \max\{|f_{x,\iota}(y)| \ : \ y \in [a_\iota, b_\iota]\} \\ &= \max\{|f(x_1, \ldots, x_{\iota-1}, y, x_{\iota+1}, \ldots, x_d)| \ : \ y \in [a_\iota, b_\iota]\} \leq \|f\|_{\infty,Q}\end{aligned}$$

erhalten wir die Stabilitätsabschätzung (3.8).

Sei nun $f \in C^{m+1}(Q)$. Dann gilt auch $f_{x,\iota} \in C^{m+1}[a_\iota, b_\iota]$, und die Ableitung ist durch

$$f_{x,\iota}(y) = \partial_\iota^{m+1} f(x_1, \ldots, x_{\iota-1}, y, x_{\iota+1}, \ldots, x_d) \qquad \text{für alle } y \in [a_\iota, b_\iota]$$

gegeben. Aus Lemma 3.7 folgt

$$|f(x) - \mathfrak{I}_{Q,\iota}[f](x)| = |f_{x,\iota}(x_\iota) - \mathfrak{I}_{[a_\iota,b_\iota]}[f_{x,\iota}](x_\iota)| \leq 2\left(\frac{b_\iota - a_\iota}{4}\right)^{m+1} \frac{\|f_{x,\iota}^{(m+1)}\|_{\infty,[a_\iota,b_\iota]}}{(m+1)!},$$

und aus

$$\begin{aligned}\|f_{x,\iota}^{(m+1)}\|_{\infty,[a_\iota,b_\iota]} &= \max\{|f_{x,\iota}^{(m+1)}(y)| \ : \ y \in [a_\iota,b_\iota]\}\\ &= \max\{|\partial_\iota^{m+1} f(x_1,\dots,x_{\iota-1},y,x_{\iota+1},\dots,x_d)| \ : \ y \in [a_\iota,b_\iota]\}\\ &\leq \|\partial_\iota^{m+1} f\|_{\infty,Q}.\end{aligned}$$

ergibt sich die Fehlerabschätzung (3.7). ∎

Aus den partiellen Interpolationsoperatoren setzt sich der gesuchte mehrdimensionale Interpolationsoperator durch Verkettung zusammen, so dass wir die Ergebnisse des Lemmas 3.8 übertragen können:

Satz 3.9 (Tensorinterpolation) *Es gelten*

$$\begin{aligned}\|f - \mathfrak{I}_Q[f]\|_{\infty,Q} &\leq 2\sum_{\iota=1}^{d} \Lambda^{\iota-1}\left(\frac{b_\iota - a_\iota}{4}\right)^{m+1} \frac{\|\partial_\iota^{m+1} f\|_{\infty,Q}}{(m+1)!} && \text{für alle } f \in C^{m+1}(Q),\\ \|\mathfrak{I}_Q[f]\|_{\infty,Q} &\leq \Lambda^d \|f\|_{\infty,Q} && \text{für alle } f \in C(Q).\end{aligned}$$

Beweis. Um den Interpolationsoperator $\mathfrak{I}_Q$ per Induktion behandeln zu können, benötigen wir Hilfsoperatoren: Wir definieren für alle $\iota \in \{0,\dots,d\}$ die Interpolationsoperatoren $P_\iota : C(Q) \to C(Q)$ durch

$$P_\iota[f] := \begin{cases} f & \text{falls } \iota = 0,\\ \mathfrak{I}_{Q,\iota} P_{\iota-1}[f] & \text{ansonsten}\end{cases} \qquad \text{für alle } f \in C(Q).$$

Dank (3.8) folgt mit einer einfachen Induktion

$$\|P_\iota[f]\|_{\infty,Q} \leq \Lambda^\iota \|f\|_{\infty,Q} \qquad \text{für alle } f \in C(Q), \iota \in \{0,\dots,d\},$$

und dank $P_d = \mathfrak{I}_Q$ folgt die gesuchte Stabilitätsaussage.

Da die partiellen Interpolationsoperatoren kommutieren gilt auch

$$P_\iota[f] = \begin{cases} f & \text{falls } \iota = 0,\\ P_{\iota-1}\mathfrak{I}_{Q,\iota}[f] & \text{ansonsten}\end{cases} \qquad \text{für alle } f \in C(Q),$$

und wir finden

$$\begin{aligned}\|f - \mathfrak{I}_Q[f]\|_{\infty,Q} &= \left\|\sum_{\iota=1}^{d} P_{\iota-1}[f] - P_\iota[f]\right\|_{\infty,Q} \leq \sum_{\iota=1}^{d} \|P_{\iota-1}[f] - P_\iota[f]\|_{\infty,Q}\\ &= \sum_{\iota=1}^{d} \|P_{\iota-1}\left[f - \mathfrak{I}_{Q,\iota}[f]\right]\|_{\infty,Q} \leq \sum_{\iota=1}^{d} \Lambda^{\iota-1} \|f - \mathfrak{I}_{Q,\iota}[f]\|_{\infty,Q}\\ &\leq 2\sum_{\iota=1}^{d} \Lambda^{\iota-1}\left(\frac{b_\iota - a_\iota}{4}\right)^{m+1} \frac{\|\partial_\iota^{m+1} f\|_{\infty,Q}}{(m+1)!},\end{aligned}$$

also die erforderliche Fehlerabschätzung. ∎

Um dieses Approximationsresultat auf die Kernfunktion g anwenden zu können benötigen wir Schranken für das Wachstum derer Ableitungen. Derartige Schranken hängen von der Art des zu behandelnden Problems ab, beispielsweise besitzen Kernfunktionen, die im Kontext elliptischer Differentialgleichungen auftreten, häufig die Eigenschaft, *asymptotisch glatt* zu sein:

Definition 3.10 (Asymptotisch glatt) *Seien $C_{as} \in \mathbb{R}_{>0}$, $c_0 \in \mathbb{R}_{\geq 1}$ und $\sigma \in \mathbb{N}_0$ Konstanten. Falls g auf der Menge $\{(x,y) : x,y \in \mathbb{R}^d,\ x \neq y\}$ unendlich oft differenzierbar ist und die Ableitungen sich durch*

$$|\partial_\iota^\nu g(x,y)| \leq C_{as} \frac{(\nu+\sigma-1)! c_0^\nu}{\|x-y\|_2^{\nu+\sigma}} \quad \text{für alle } \nu \in \mathbb{N},\ \iota \in \{1,\dots,2d\},\ x,y \in \mathbb{R}^d \text{ mit } x \neq y$$

abschätzen lassen, nennen wir g eine $(C_{\mathrm{as}}, c_0, \sigma)$-asymptotisch glatte *Kernfunktion.*

Zur Vereinfachung der Notation wird g hier als eine Funktion auf dem Raum $\mathbb{R}^{2d}$ statt $\mathbb{R}^d \times \mathbb{R}^d$ behandelt, die partielle Ableitung ∂_ι^ν für $\iota > d$ bedeutet also, dass nach der $(\iota - d)$-ten Komponenten des Parameters y abgeleitet werden soll.

Das motivierende Beispiel für die Definition einer asymptotisch glatten Funktion ist

$$g : \mathbb{R} \times \mathbb{R} \to \mathbb{R}, \qquad (x,y) \mapsto \begin{cases} 0 & \text{falls } x = y, \\ -\log|x-y| & \text{ansonsten,} \end{cases}$$

denn ihre Ableitungen sind gerade

$$\partial_1^\nu g(x,y) = (-1)^\nu \frac{(\nu-1)!}{(x-y)^\nu} \qquad \text{für alle } \nu \in \mathbb{N}, x,y \in \mathbb{R} \text{ mit } x \neq y,$$
$$\partial_2^\nu g(x,y) = \frac{(\nu-1)!}{(x-y)^\nu} \qquad \text{für alle } \nu \in \mathbb{N}, x,y \in \mathbb{R} \text{ mit } x \neq y.$$

Diese Funktion ist also $(1,1,0)$-asymptotisch glatt. Die Eigenschaft überträgt sich auf ihre Ableitungen: Für beliebige $\alpha, \beta \in \mathbb{N}_0$ ist die Funktion $\partial_1^\alpha \partial_2^\beta g$ gerade $(1,1,\alpha+\beta)$-asymptotisch glatt.

Bei komplizierteren Kernfunktionen können die Konstanten C_{as} und c_0 verwendet werden, um eventuelle zusätzlich auftretende Terme zu beherrschen, sofern sie lediglich exponentiell mit der Ableitungsordnung wachsen, während σ die Ordnung der Singularität in $x = y$ beschreibt.

Infolge der Lage dieser Singularität dürfen wir nur darauf hoffen, die Funktion g in Gebieten approximieren zu können, die hinreichend weit von ihr entfernt sind. Diese Vermutung findet auch in der Fehlerabschätzung ihren Ausdruck: Wenn wir die Approximation

$$\tilde{g}(x,y) = \sum_{\nu \in M} \mathcal{L}_{Q_t,\nu}(x) g(\xi_{Q_t,\nu}, y)$$

verwenden, lässt sie sich als Anwendung des Tensor-Interpolationsoperators $\mathfrak{I}_{Q_t}$ auf die Funktionen

$$g_y : Q_t \to \mathbb{K}, \qquad x \mapsto g(x,y),$$

mit jeweils fixiertem $y \in Q_s$ interpretieren, so dass sich aus Satz 3.9 die Abschätzung

$$\|g - \tilde{g}\|_{\infty,Q_t\times Q_s} \leq 2C_{\mathrm{as}} \sum_{\iota=1}^{d} \Lambda^{\iota-1} \left(\frac{b_{t,\iota} - a_{t,\iota}}{4}\right)^{m+1} \frac{(m+\sigma)! c_0^{m+1}}{(m+1)! \operatorname{dist}(Q_t,Q_s)^{m+1+\sigma}}$$

ergibt, wobei $Q_t = [a_{t,1}, b_{t,1}] \times \ldots \times [a_{t,d}, b_{t,d}]$ gilt. Zur Vereinfachung schätzen wir Λ aus (3.6) durch $\Lambda \leq m+1$ ab und sammeln alle höchstens polynomiell von m abhängenden Terme in einem Polynom

$$C(m) := 2dC_{\mathrm{as}}(m+1)^{d-1} \frac{(m+\sigma)!}{(m+1)!} \geq 2C_{\mathrm{as}} \sum_{\iota=1}^{d} \Lambda^{\iota-1} \frac{(m+\sigma)!}{(m+1)!}.$$

Außerdem verwenden wir wieder die Bezeichnung

$$\operatorname{diam}_\infty(Q_t) = \max\{\|x-y\|_\infty \ : \ x,y \in Q_t\} = \max\{b_{t,\iota} - a_{t,\iota} \ : \ \iota \in \{1,\ldots,d\}\}$$

für den Durchmesser des überdeckenden Quaders Q_t bezüglich der Maximum-Norm und erhalten die vereinfachte Darstellung

$$\|g - \tilde{g}\|_{\infty,Q_t\times Q_s} \leq \frac{C(m)}{\operatorname{dist}(Q_t,Q_s)^\sigma} \left(\frac{c_0 \operatorname{diam}_\infty(Q_t)}{4 \operatorname{dist}(Q_t,Q_s)}\right)^{m+1}.$$

Um exponentielle Konvergenz zu erhalten, um also sicher zu stellen, dass der Fehler um einen gewissen Faktor reduziert wird, wenn wir m erhöhen, müssen wir den rechten Faktor dieser Abschätzung unter Kontrolle bringen. Das gelingt uns durch Wahl der geeigneten Zulässigkeitsbedingung: Wir fordern, dass jedes zulässige Paar von Clustern $t \in \mathcal{T}_\mathcal{I}$, $s \in \mathcal{T}_\mathcal{J}$ die Bedingung

$$\operatorname{diam}_\infty(Q_t) \leq \eta \operatorname{dist}(Q_t,Q_s) \tag{3.9}$$

mit einem Parameter $\eta \in \mathbb{R}_{>0}$ erfüllt. Für Blöcke, die dieser Bedingung genügen, folgt

$$\|g - \tilde{g}\|_{\infty,Q_t\times Q_s} \leq \frac{C(m)}{\operatorname{dist}(Q_t,Q_s)^\sigma} \left(\frac{c_0\eta}{4}\right)^{m+1}, \tag{3.10}$$

durch Wahl eines hinreichend kleinen $\eta < 4/c_0$ erreichen wir also die gewünschte exponentielle Konvergenz. Das Wachstum des Polynoms $C(m)$ spielt für hinreichend hohe Ordnungen m im Vergleich zu der exponentiellen Konvergenz keine große Rolle und kann deshalb bei asymptotischen Betrachtungen vernachlässigt werden.

Falls wir die Interpolation in der Variablen y durchführen, falls wir also

$$\tilde{g}(x,y) = \sum_{\mu\in M} g(x,\xi_{Q_s,\mu}) \mathcal{L}_{Q_s,\mu}(y)$$

verwenden, erhalten wir dieselbe entsprechende Abschätzung für die Zulässigkeitsbedingung

$$\operatorname{diam}_\infty(Q_s) \leq \eta \operatorname{dist}(Q_t, Q_s). \tag{3.11}$$

Wir können beide Zulässigkeitsbedingungen kombinieren, indem wir die Wahl der Kernapproximation von dem Durchmesser der Quader Q_t und Q_s abhängig machen: Falls $\operatorname{diam}_\infty(Q_t) \leq \operatorname{diam}_\infty(Q_s)$ gilt, interpolieren wir in x, anderenfalls in y. Bei dieser Vorgehensweise genügt die symmetrische Zulässigkeitsbedingung

$$\min\{\operatorname{diam}_\infty(Q_t), \operatorname{diam}_\infty(Q_s)\} \leq \eta \operatorname{dist}(Q_t, Q_s), \tag{3.12}$$

die schwächer als die beiden diskutierten ist und trotzdem ausreicht, um eine Approximation mit der Fehlerabschätzung (3.10) praktisch konstruieren zu können.

3.4 Verfeinerte Fehlerabschätzungen *

Die Abschätzung (3.10) ist für unsere Zwecke völlig ausreichend, sofern uns die Konstante c_0 oder zumindest eine obere Schranke dieser Konstante bekannt ist, denn dann können wir den Parameter η der Zulässigkeitsbedingung so wählen, dass die entscheidende Abschätzung $\eta < 4/c_0$ erfüllt ist.

Da die Interpolation ein sehr allgemeines Verfahren ist, wäre es wünschenswert, eine Zulässigkeitsbedingung zu finden, die die exponentielle Konvergenz auch ohne Kenntnis der Konstanten c_0 sicher stellt. Diese Möglichkeit besteht: Man kann nachweisen, dass bei einer asymptotisch glatten Kernfunktion für *jedes* $\eta > 0$ der Interpolant exponentiell konvergiert, wenn die schwache Zulässigkeitsbedingung (3.12) erfüllt ist.

Der erste Schritt dieses Beweises ist einfach: Wir können den Interpolanten einer Funktion mit ihrer bestmöglichen Approximation im Raum der Polynome in Beziehung setzen.

Lemma 3.11 (Bestapproximation) *Sei Λ eine Stabilitätskonstante für $\mathfrak{I}$. Dann gilt*

$$\|f - \mathfrak{I}[f]\|_{\infty,[-1,1]} \leq (\Lambda + 1)\|f - p\|_{\infty,[-1,1]} \qquad \text{für alle } f \in C[-1,1],\ p \in \Pi_m.$$

Beweis. Seien $f \in C[-1,1]$ und $p \in \Pi_m$ gegeben. Da die Polynome p und $\mathfrak{I}[p]$ in den $m+1$ Interpolationspunkten übereinstimmen, muss

$$p = \mathfrak{I}[p]$$

gelten. Aus dieser *Projektionseigenschaft* und aus der Linearität des Interpolationsoperators $\mathfrak{I}$ folgt

$$\begin{aligned}
\|f - \mathfrak{I}[f]\|_{\infty,[-1,1]} &= \|f - p + \mathfrak{I}[p] - \mathfrak{I}[f]\|_{\infty,[-1,1]} \\
&\leq \|f - p\|_{\infty,[-1,1]} + \|\mathfrak{I}[p] - \mathfrak{I}[f]\|_{\infty,[-1,1]} \\
&= \|f - p\|_{\infty,[-1,1]} + \|\mathfrak{I}[p - f]\|_{\infty,[-1,1]} \\
&\leq \|f - p\|_{\infty,[-1,1]} + \Lambda\|f - p\|_{\infty,[-1,1]} = (\Lambda + 1)\|f - p\|_{\infty,[-1,1]},
\end{aligned}$$

also die gesuchte Aussage. ∎

Wir müssen also nicht unbedingt mit Ableitungen arbeiten, um den Interpolationsfehler abzuschätzen, wir können auch versuchen, die Existenz eines Polynoms nachzuweisen, das die zu interpolierende Funktion hinreichend gut approximiert.

Es lässt sich zeigen, dass die asymptotische Glattheit der Kernfunktion impliziert, dass letztere sich zu einer holomorphen Funktion auf einer Teilmenge des $\mathbb{C}^d \times \mathbb{C}^d$ fortsetzen lässt. Es lässt sich sogar das Wachstum dieser Fortsetzung abschätzen.

Mit Hilfe von etwas Funktionentheorie können wir auf dieser Grundlage nun zeigen, dass neben der Abschätzung (3.10) auch eine Abschätzung der Form

$$\|g - \tilde{g}\|_{\infty,Q_t \times Q_s} \leq \frac{C(m)}{\operatorname{dist}(Q_t, Q_s)^\sigma} \left(\frac{c_0\eta}{c_0\eta + 2}\right)^{m+1}$$

gilt, also muss der Fehler für *jedes* $c_0 > 0$ und *jedes* $\eta > 0$ exponentiell konvergieren.

Der Ansatz, Lemma 3.11 zum Nachweis der Existenz einer Approximation zu verwenden, lässt sich auch verwenden, um ein weiteres Problem zu lösen: Häufig sind wir nicht nur an der Approximation der Kernfunktion g, sondern auch an der Approximation ihrer Ableitungen interessiert. Beispielsweise übt das Gravitationsfeld einer Masse m_y im Punkt $y \in \mathbb{R}^3$ auf eine Masse m_x im Punkt $x \in \mathbb{R}^3$ eine Kraft aus, die proportional zu

$$F(x,y) = m_x m_y \frac{y-x}{\|y-x\|_2^3}$$

ist. Es lässt sich relativ leicht nachweisen, dass jede Komponente der Funktion f asymptotisch glatt mit Singularitätsordnung $\sigma = 2$ ist, also können wir jede Komponente einzeln interpolieren, um eine entartete Approximation zu konstruieren.

Eleganter ist es allerdings, die Funktion

$$g : \mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}, \qquad (x,y) \mapsto \begin{cases} \frac{1}{\|y-x\|_2} & \text{falls } x \neq y, \\ 0 & \text{ansonsten,} \end{cases}$$

zu verwenden. Sie lässt sich auch durch

$$g(x,y) = \left(\sum_{\iota=1}^{3} (x_\iota - y_\iota)^2\right)^{-1/2} \qquad \text{für alle } x, y \in \mathbb{R}^d \text{ mit } x \neq y$$

darstellen, und wir können leicht nachprüfen, dass

$$\begin{aligned} \partial_\iota g(x,y) &= -(x_\iota - y_\iota) \left(\sum_{\iota=1}^{3} (x_\iota - y_\iota)^2\right)^{-3/2} \\ &= \frac{y_\iota - x_\iota}{\|y-x\|_2^3} \qquad \text{für alle } x, y \in \mathbb{R}^3 \text{ mit } x \neq y \end{aligned}$$

gilt. Wir können also F aus g rekonstruieren, indem wir die partiellen Ableitungen berechnen, also liegt es nahe, zu hoffen, dass wir eine entartete Approximation der Funktion

F aus einer der Funktion g erhalten können, indem wir letztere differenzieren. Die Multiplikation mit den Massen m_x und m_y ist dabei unkritisch, da sich diese Produkte ohne weiteres in den Termen der entarteten Approximation unterbringen lassen.

Insbesondere stellt sich die Frage, ob die Ableitungen des Interpolanten die Ableitungen der interpolierten Funktion approximieren. Um ähnlich wie zuvor vorgehen zu können benötigen wir eine Stabilitätsaussage für die Ableitungen von Polynomen. Sie lässt sich aus der bereits bekannten Stabilitätsaussage mit Hilfe zweier Teilschritte gewinnen:

Um eine Aussage über die Ableitung von $\mathfrak{I}[f]$ zu erhalten, wäre es hilfreich, die Ableitung des Interpolanten durch den Interpolanten selbst abschätzen zu können. Für allgemeine Funktionen ist es sicherlich nicht möglich, die Maximumnorm der Ableitung durch die Maximumnorm der Funktion zu beschränken, für Polynome dagegen gilt eine sogenannte *inverse Abschätzung* der Form

$$\|p'\|_{\infty,[-1,1]} \leq C_{\mathrm{iv}} \|p\|_{\infty,[-1,1]} \qquad \text{für alle } p \in \Pi_m. \tag{3.13}$$

Letzten Endes ist diese Abschätzung eine Folge des Satzes von Heine-Borel: Da Π_m ein endlich-dimensionaler Banachraum ist, muss jeder lineare Operator auf diesem Raum stetig sein, und C_{iv} ist gerade die Stetigkeitskonstante des Ableitungsoperators. Eine genauere Aussage erhalten wir aus dem *Satz von Markow* [14, Theorem 4.1.4], der besagt, dass $C_{\mathrm{iv}} = m^2$ die bestmögliche Konstante ist.

Wir benötigen auch eine Möglichkeit, um umgekehrt die Maximumnorm der Funktion durch die ihrer Ableitung abschätzen zu können. Wie man am Beispiel der konstanten Funktion (Ableitung null, Funktionswert nicht null) erkennt, kann auch diese Abschätzung nur für bestimmte Funktionen gelingen. In unserem Fall genügt der Satz von Taylor: Für jede Funktion $f \in C^1[-1,1]$ und jedes $x \in [-1,1]$ gilt

$$f(x) = f(0) + f'(\eta)x$$

mit einem $\eta \in [-1,1]$, also insbesondere

$$\|f - f(0)\|_{\infty,[-1,1]} \leq \|f'\|_{\infty,[-1,1]}.$$

Nun können wir die gesuchte Stabilitätsaussage formulieren und beweisen:

Lemma 3.12 (Stabilität der Ableitung) *Seien Konstanten $\Lambda, C_{iv} \in \mathbb{R}_{>0}$ mit*

$$\begin{aligned} \|\mathfrak{I}[f]\|_{\infty,[-1,1]} &\leq \Lambda \|f\|_{\infty,[-1,1]} && \text{für alle } f \in C[-1,1], \\ \|p'\|_{\infty,[-1,1]} &\leq C_{iv} \|p\|_{\infty,[-1,1]} && \text{für alle } p \in \Pi_m \end{aligned}$$

gegeben. Dann gilt auch

$$\|(\mathfrak{I}[f])'\|_{\infty,[-1,1]} \leq \Lambda' \|f'\|_{\infty,[-1,1]} \qquad \text{für alle } f \in C^1[-1,1] \tag{3.14}$$

mit der Stabilitätskonstanten $\Lambda' := C_{iv}\Lambda$.

Beweis. Sei $f \in C^1[-1,1]$. Wir definieren

$$q : [-1,1] \to \mathbb{K}, \qquad x \mapsto f(0),$$

und stellen fest, dass $q' = 0$ und $q \in \Pi_m$ sowie

$$\|f - q\|_{\infty,[-1,1]} \leq \|f'\|_{\infty,[-1,1]}$$

nach dem Satz von Taylor gelten. Nun setzen wir $\hat{f} := f - q$ und finden

$$\begin{aligned}\|(\mathfrak{I}[f])'\|_{\infty,[-1,1]} &= \|(\mathfrak{I}[f] - q)'\|_{\infty,[-1,1]} = \|(\mathfrak{I}[f-q])'\|_{\infty,[-1,1]} = \|(\mathfrak{I}[\hat{f}])'\|_{\infty,[-1,1]} \\ &\leq C_{\mathrm{iv}}\|\mathfrak{I}[\hat{f}]\|_{\infty,[-1,1]} \leq C_{\mathrm{iv}}\Lambda\|\hat{f}\|_{\infty,[-1,1]} = C_{\mathrm{iv}}\Lambda\|f-q\|_{\infty,[-1,1]} \\ &\leq C_{\mathrm{iv}}\Lambda\|(f-q)'\|_{\infty,[-1,1]}. = C_{\mathrm{iv}}\Lambda\|f'\|_{\infty,[-1,1]}.\end{aligned}$$

Das ist die gesuchte Stabilitätsaussage. ∎

Der Beweis lässt sich verallgemeinern: Für höhere Ableitungen müssen wir lediglich q durch ein Polynom entsprechend höherer Ordnung ersetzen, beispielsweise durch den Interpolanten aus (3.4), der es uns ermöglicht, von der Maximumnorm von $f - q$ zu der Maximumnorm der Ableitung zurückzukehren.

Mit Hilfe der Stabilitätsaussage können wir die folgende Fehlerabschätzung für die Ableitung des Interpolanten gewinnen:

Lemma 3.13 (Ableitung des Interpolanten) *Sei $\mathfrak{I}$ wie zuvor der Tschebyscheff-Interpolationsoperator, und sei $\mathfrak{I}_{[a,b]}$ wieder der zugehörige transformierte Interpolationsoperator auf dem Intervall $[a,b]$. Sei Λ' eine Konstante, die (3.14) erfüllt. Dann gilt*

$$\|(f - \mathfrak{I}_{[a,b]}[f])'\|_{\infty,[a,b]} \leq 2(\Lambda'+1)\left(\frac{b-a}{4}\right)^m \frac{\|f^{(m+1)}\|_{\infty,[a,b]}}{(m+1)!} \quad \text{für alle } f \in C^{m+1}[a,b].$$

Beweis. Zunächst verallgemeinern wir die Stabilitätsaussage aus Lemma 3.12: Für $f \in C^1[a,b]$ setzen wir $\hat{f} := f \circ \Phi_{[a,b]} \in C^1[-1,1]$ und erhalten mit der Kettenregel

$$\begin{aligned}\|(\mathfrak{I}_{[a,b]}[f])'\|_{\infty,[a,b]} &= \|(\mathfrak{I}[\hat{f}] \circ \Phi_{[a,b]}^{-1})'\|_{\infty,[a,b]} = \frac{2}{b-a}\|(\mathfrak{I}[\hat{f}])' \circ \Phi_{[a,b]}^{-1}\|_{\infty,[-1,1]} \\ &= \frac{2}{b-a}\|(\mathfrak{I}[\hat{f}])'\|_{\infty,[-1,1]} \leq C_{\mathrm{iv}}\Lambda\frac{2}{b-a}\|\hat{f}'\|_{\infty,[-1,1]} \\ &= C_{\mathrm{iv}}\Lambda\frac{2}{b-a}\frac{b-a}{2}\|f' \circ \Phi_{[a,b]}\|_{\infty,[-1,1]} = C_{\mathrm{iv}}\Lambda\|f'\|_{\infty,[a,b]}.\end{aligned}$$

Wie in Lemma 3.11 folgt daraus bereits

$$\|(f - \mathfrak{I}_{[a,b]}[f])'\|_{\infty,[a,b]} \leq (C_{\mathrm{iv}}\Lambda + 1)\|(f-p)'\|_{\infty,[a,b]}$$

für alle $p \in \Pi_m$, wir müssen also „nur“ noch ein geeignetes Polynom finden.

Dazu greifen wir auf Lemma 3.7 zurück, das wir diesmal auf f' und die Ordnung $m-1$ anwenden, um ein $q \in \Pi_{m-1}$ mit

$$\|f' - q\|_{\infty,[a,b]} \leq 2 \left(\frac{b-a}{4}\right)^m \frac{\|f^{(m+1)}\|_{\infty,[a,b]}}{(m+1)!}$$

zu konstruieren. Das Polynom p definieren wir dann als dessen Stammfunktion

$$p(x) = \int_a^x q(y)\,dy \qquad \text{für alle } x \in [a,b],$$

so dass wir $p' = q$ erhalten. Es folgt

$$\begin{aligned}\|(f - \mathfrak{I}_{[a,b]}[f])'\|_{\infty,[a,b]} &\leq (C_{\mathrm{iv}}\Lambda + 1)\|(f-p)'\|_{\infty,[a,b]} \\ &\leq (C_{\mathrm{iv}}\Lambda + 1)2\left(\frac{b-a}{4}\right)^m \frac{\|f^{(m+1)}\|_{\infty,[a,b]}}{(m+1)!}.\end{aligned}$$

■

Abgesehen von der etwas vergrößerten Stabilitätskonstanten Λ' verlieren wir also lediglich einen Exponenten des Faktors $(b-a)/4$, weil wir f' nur mit einem Polynom der Ordnung $m-1$ statt mit der vollen Ordnung m approximieren.

Gegenüber der direkten Interpolation der Ableitung mit der vollen Ordnung bietet der neue Ansatz in der Praxis, insbesondere bei der Implementierung, in der Regel viele Vorteile, die den Nachteil der etwas geringeren Genauigkeit häufig aufwiegen. Insbesondere können wir bei beiden Verfahren ohnehin die Ordnung m erhöhen, um jede beliebige Genauigkeit zu erreichen.

4 Kreuzapproximation

Die Konstruktion einer Rang-k-Approximation eines Matrixblocks mit Hilfe einer entarteten Näherung der der Matrix zugrunde liegenden Kernfunktion führt zu zuverlässigen und einfach umzusetzenden Verfahren, allerdings berücksichtigt sie nicht Besonderheiten des Matrixblocks oder seiner Entstehung. Falls der Block beispielsweise nur 10×10 Koeffizienten enthält, ließe es sich selbstverständlich mit einem Rang von $k = 10$ exakt darstellen, obwohl ein auf Interpolation aufsetzendes Verfahren auch bei $k = 20$ lediglich eine Approximation bietet. Falls der Block mit einem eindimensionalen Segment einer zweidimensionalen Geometrie zusammenhängt, werden wir mit Interpolation trotzdem eine zweidimensionale Approximation konstruieren, obwohl eine eindimensionale völlig ausreichen würde.

Es stellt sich die Frage, ob es möglich ist, eine Approximation eines Matrixblocks zu konstruieren, die dessen Eigenschaften besser ausnutzt und trotzdem effizient zu berechnen ist. Einen Ansatz bietet die *Kreuzapproximation* [36], bei der aus nur k Zeilen und Spalten des Matrixblocks eine Rang-k-Darstellung gewonnen wird. Entscheidend bei diesem Ansatz ist die Frage, wie diese k Zeilen und Spalten ausgewählt werden sollen. Einen relativ einfachen, von der partiellen Pivotsuche bei der Gauß-Elimination motivierten Ansatz verfolgt die *adaptive Kreuzapproximation* [2, 1, 4], allerdings bestehen Zweifel an ihrer Zuverlässigkeit. In [20] und [3] werden alternative Strategien entwickelt, die diese Probleme zu lösen versuchen, allerdings lassen sich einfache Beispiele finden, bei denen alle bisher bekannten effizienten Strategien versagen.

Diese Schwierigkeiten des rein algebraischen Ansatzes lassen sich beheben, indem man auf die Kernfunktion zurückgreift: Die *hybride Kreuzapproximation* [9] kombiniert Interpolation und Kreuzapproximation, um ein Verfahren zu entwerfen, das die Zuverlässigkeit der auf Interpolation basierenden Techniken erhält und die hohe Effizienz der Kreuzapproximation teilweise sogar noch übertrifft.

4.1 Unvollständige Dreieckszerlegungen

Als motivierendes Beispiel untersuchen wir eine Matrix $\mathbf{G} \in \mathbb{K}^{n \times m}$, bei der bekannt ist, dass sie einen Rang von k besitzt, dass also die Dimension des Bildes gerade gleich k ist. Für derartige Matrizen lässt sich eine spezielle Form der LR-Zerlegung konstruieren:

Satz 4.1 (Dünne LR-Zerlegung) *Sei $\mathbf{G} \in \mathbb{K}^{n \times m}$ eine Matrix von Rang k. Dann existieren Permutationen $\mathbf{P} \in \mathbb{K}^{n \times n}$ und $\mathbf{Q} \in \mathbb{K}^{m \times m}$, eine untere Dreiecksmatrix $\mathbf{L} \in \mathbb{K}^{n \times k}$ und eine obere Dreiecksmatrix $\mathbf{R} \in \mathbb{K}^{k \times m}$ mit*

$$\mathbf{PGQ}^* = \mathbf{LR}.$$

Alle Diagonalelemente beider Dreiecksmatrizen sind von null verschieden, also besitzen beide insbesondere vollen Rang.

Beweis. Per Induktion über $k \in \mathbb{N}_0$. Für $k = 0$ ist die Aussage trivial.

Sei nun $k \in \mathbb{N}_0$ so fixiert, dass die Aussage gilt. Sei $\mathbf{G} \in \mathbb{K}^{n \times m}$ eine Matrix von Rang $k+1$. Dann ist insbesondere $\mathbf{G} \neq \mathbf{0}$, also müssen Permutationen $\mathbf{P}_1 \in \mathbb{K}^{n \times n}$ und $\mathbf{Q}_1 \in \mathbb{K}^{m \times m}$ mit

$$(\mathbf{P}_1 \mathbf{G} \mathbf{Q}_1^*)_{11} \neq 0$$

existieren. Damit können wir nun immerhin die erste Zeile und die erste Spalte einer LR-Zerlegung der Matrix $\mathbf{H} := \mathbf{P}_1 \mathbf{G} \mathbf{Q}_1^*$ konstruieren. Wir setzen

$$\mathbf{H}_{1*} := \begin{pmatrix} H_{12} & \dots & H_{1m} \end{pmatrix}, \qquad \mathbf{H}_{*1} := \begin{pmatrix} H_{21} \\ \vdots \\ H_{n1} \end{pmatrix}, \qquad \mathbf{H}_{**} := \begin{pmatrix} H_{22} & \dots & H_{2m} \\ \vdots & \ddots & \vdots \\ H_{n2} & \dots & H_{nm} \end{pmatrix}$$

und erhalten

$$\mathbf{H} = \begin{pmatrix} 1 & \\ \mathbf{H}_{*1} H_{11}^{-1} & \mathbf{I} \end{pmatrix} \begin{pmatrix} H_{11} & \mathbf{H}_{1*} \\ & \mathbf{H}_{**} - \mathbf{H}_{*1} H_{11}^{-1} \mathbf{H}_{1*} \end{pmatrix} = \begin{pmatrix} 1 & \\ \mathbf{H}_{*1} H_{11}^{-1} & \mathbf{I} \end{pmatrix} \begin{pmatrix} H_{11} & \mathbf{H}_{1*} \\ & \widehat{\mathbf{H}} \end{pmatrix} \tag{4.1}$$

mit der Matrix

$$\widehat{\mathbf{H}} := \mathbf{H}_{**} - \mathbf{H}_{*1} H_{11}^{-1} \mathbf{H}_{1*}.$$

Diese Gleichung lässt sich auch in der Form

$$\mathbf{H} = \begin{pmatrix} 1 \\ \mathbf{H}_{*1} H_{11}^{-1} \end{pmatrix} \begin{pmatrix} H_{11} & \mathbf{H}_{1*} \end{pmatrix} + \begin{pmatrix} 0 & \mathbf{0} \\ \mathbf{0} & \widehat{\mathbf{H}} \end{pmatrix} \tag{4.2}$$

schreiben, wir haben also $\mathbf{H}$ in einen Term von Rang 1 und einen von der Matrix $\widehat{\mathbf{H}}$ bestimmten Rest zerlegt.

Um die Induktionsvoraussetzung auf $\widehat{\mathbf{H}}$ anwenden zu können, müssen wir nachweisen, dass der Rang dieser Matrix gerade k beträgt. Da der erste Summand in (4.2) eine Matrix von Rang 1 ist, muss der Rang von $\widehat{\mathbf{H}}$ mindestens k betragen, damit $\mathbf{H}$ von Rang $k+1$ sein kann.

Nun ist zu zeigen, dass der Rang von $\widehat{\mathbf{H}}$ nicht größer als k ist. Sei dazu $\mathbf{e}_1 \in \mathbb{K}^m$ der erste kanonische Einheitsvektor. Wegen $H_{11} \neq 0$ gilt

$$\mathbf{H}\mathbf{e}_1 = \begin{pmatrix} H_{11} \\ \mathbf{H}_{*1} \end{pmatrix} \notin \operatorname{Bild} \begin{pmatrix} 0 & \mathbf{0} \\ \mathbf{0} & \widehat{\mathbf{H}} \end{pmatrix},$$

also kann wegen (4.2) das Bild der Matrix $\widehat{\mathbf{H}}$ höchstens k-dimensional sein, wenn die Matrix $\mathbf{H}$ die Dimension $k+1$ besitzen soll.

Damit ist $\widehat{\mathbf{H}}$ eine Matrix von Rang k, so dass wir die Induktionsvoraussetzung anwenden können, um Permutationen $\widehat{\mathbf{P}} \in \mathbb{K}^{(n-1)\times(n-1)}$ und $\widehat{\mathbf{Q}} \in \mathbb{K}^{(m-1)\times(m-1)}$ sowie eine

untere Dreiecksmatrix $\widehat{\mathbf{L}} \in \mathbb{K}^{(n-1)\times k}$ und eine obere Dreiecksmatrix $\widehat{\mathbf{R}} \in \mathbb{K}^{k\times(m-1)}$ zu erhalten, die

$$\widehat{\mathbf{P}}\widehat{\mathbf{H}}\widehat{\mathbf{Q}}^* = \widehat{\mathbf{L}}\widehat{\mathbf{R}}$$

erfüllen. Da Permutationsmatrizen insbesondere unitär sind, folgt

$$\widehat{\mathbf{H}} = \widehat{\mathbf{P}}^*\widehat{\mathbf{L}}\widehat{\mathbf{R}}\widehat{\mathbf{Q}},$$

und durch Einsetzen dieser Gleichung in (4.1) ergibt sich

$$\begin{aligned}\mathbf{H} &= \begin{pmatrix} 1 & \\ \mathbf{H}_{*1}H_{11}^{-1} & \mathbf{I}\end{pmatrix}\begin{pmatrix} H_{11} & \mathbf{H}_{1*} \\ & \widehat{\mathbf{P}}^*\widehat{\mathbf{L}}\widehat{\mathbf{R}}\widehat{\mathbf{Q}}\end{pmatrix} = \begin{pmatrix} 1 & \\ \mathbf{H}_{*1}H_{11}^{-1} & \widehat{\mathbf{P}}^*\widehat{\mathbf{L}}\end{pmatrix}\begin{pmatrix} H_{11} & \mathbf{H}_{1*} \\ & \widehat{\mathbf{R}}\widehat{\mathbf{Q}}\end{pmatrix} \\ &= \begin{pmatrix} 1 & \\ & \widehat{\mathbf{P}}^*\end{pmatrix}\begin{pmatrix} 1 & \\ \widehat{\mathbf{P}}\mathbf{H}_{*1}H_{11}^{-1} & \widehat{\mathbf{L}}\end{pmatrix}\begin{pmatrix} H_{11} & \mathbf{H}_{1*}\widehat{\mathbf{Q}}^* \\ & \widehat{\mathbf{R}}\end{pmatrix}\begin{pmatrix} 1 & \\ & \widehat{\mathbf{Q}}\end{pmatrix}.\end{aligned}$$

An dieser Gleichung können wir die gesuchte Zerlegung ablesen: Sowohl der erste als auch der letzte Faktor ist eine Permutationsmatrix, der zweite Faktor ist eine linke untere Dreiecksmatrix, und der dritte Faktor ist eine rechte obere Dreiecksmatrix. Also setzen wir

$$\mathbf{P} := \begin{pmatrix} 1 & \\ & \widehat{\mathbf{P}}\end{pmatrix}\mathbf{P}_1, \qquad \mathbf{Q} := \begin{pmatrix} 1 & \\ & \widehat{\mathbf{Q}}\end{pmatrix}\mathbf{Q}_1,$$

$$\mathbf{L} := \begin{pmatrix} 1 & \\ \widehat{\mathbf{P}}\mathbf{H}_{*1}H_{11}^{-1} & \widehat{\mathbf{L}}\end{pmatrix}, \qquad \mathbf{R} := \begin{pmatrix} H_{11} & \mathbf{H}_{1*}\widehat{\mathbf{Q}}^* \\ & \widehat{\mathbf{R}}\end{pmatrix}$$

und erhalten

$$\begin{aligned}\mathbf{PGQ}^* &= \begin{pmatrix} 1 & \\ & \widehat{\mathbf{P}}\end{pmatrix}\mathbf{P}_1\mathbf{G}\mathbf{Q}_1^*\begin{pmatrix} 1 & \\ & \widehat{\mathbf{Q}}^*\end{pmatrix} = \begin{pmatrix} 1 & \\ & \widehat{\mathbf{P}}\end{pmatrix}\mathbf{H}\begin{pmatrix} 1 & \\ & \widehat{\mathbf{Q}}^*\end{pmatrix} \\ &= \begin{pmatrix} 1 & \\ & \widehat{\mathbf{P}}\end{pmatrix}\begin{pmatrix} 1 & \\ & \widehat{\mathbf{P}}^*\end{pmatrix}\begin{pmatrix} 1 & \\ \widehat{\mathbf{P}}\mathbf{H}_{*1}H_{11}^{-1} & \widehat{\mathbf{L}}\end{pmatrix}\begin{pmatrix} H_{11} & \mathbf{H}_{1*}\widehat{\mathbf{Q}}^* \\ & \widehat{\mathbf{R}}\end{pmatrix}\begin{pmatrix} 1 & \\ & \widehat{\mathbf{Q}}\end{pmatrix}\begin{pmatrix} 1 & \\ & \widehat{\mathbf{Q}}^*\end{pmatrix} \\ &= \begin{pmatrix} 1 & \\ \widehat{\mathbf{P}}\mathbf{H}_{*1}H_{11}^{-1} & \widehat{\mathbf{L}}\end{pmatrix}\begin{pmatrix} H_{11} & \mathbf{H}_{1*}\widehat{\mathbf{Q}}^* \\ & \widehat{\mathbf{R}}\end{pmatrix} = \mathbf{LR}.\end{aligned}$$

Da $\mathbf{P}$ und $\mathbf{Q}$ als Verkettung von Permutationsmatrizen ebenfalls Permutationsmatrizen sind, ist das die gewünschte Zerlegung. ∎

Dieser Beweis ist konstruktiv, denn er folgt der klassischen Gauß-Elimination, die durch eine vollständige Pivot-Suche, also eine Pivot-Suche in Zeilen *und* Spalten, ergänzt wird. Da die Matrix $\mathbf{L}$ in diesem Fall nur k Spalten besitzt und die Matrix $\mathbf{R}$ nur k Zeilen aufweist, berechnet dieser Algorithmus auch bereits eine faktorisierte Rang-k-Darstellung.

Die dünne LR-Zerlegung besitzt den Vorteil, dass zu ihrer Berechnung nur jeweils k Zeilen und k Spalten der Ausgangsmatrix $\mathbf{G}$ benötigt werden, deshalb lässt sie sich auch bei Matrizen mit sehr vielen Zeilen und sehr vielen Spalten noch effizient berechnen.

Folgerung 4.2 (Darstellung durch Zeilen und Spalten) *Sei $\mathbf{G} \in \mathbb{K}^{\mathcal{I}\times\mathcal{J}}$ eine Matrix von Rang k. Dann existieren k-elementige Teilmengen $\mathcal{I}_k \subseteq \mathcal{I}$ und $\mathcal{J}_k \subseteq \mathcal{J}$ so, dass $\mathbf{G}|_{\mathcal{I}_k\times\mathcal{J}_k}$ regulär ist und*

$$\mathbf{G} = \mathbf{G}|_{\mathcal{I}\times\mathcal{J}_k}\mathbf{G}|^{-1}_{\mathcal{I}_k\times\mathcal{J}_k}\mathbf{G}|_{\mathcal{I}_k\times\mathcal{J}}$$

gilt. Für das Bild von $\mathbf{G}$ können wir also eine Basis aus Spalten der Matrix $\mathbf{G}$ finden, und für das orthogonale Komplement des Kerns eine Basis aus Zeilen.

Beweis. Um Satz 4.1 anwenden zu können, müssen wir zunächst von den allgemeinen Indexmengen $\mathcal{I}$ und $\mathcal{J}$ zu durchnumierten Zeilen und Spalten wechseln. Seien $n := \#\mathcal{I}$ und $m := \#\mathcal{J}$, und seien

$$\iota_n : \{1,\dots,n\} \to \mathcal{I}, \qquad \iota_m : \{1,\dots,m\} \to \mathcal{J}$$

bijektive Abbildungen. Wir definieren $\mathbf{H} \in \mathbb{K}^{n\times m}$ durch

$$\mathbf{H}_{ij} = \mathbf{G}_{\iota_n(i),\iota_m(j)} \qquad \text{für alle } i \in \{1,\dots,n\}, j \in \{1,\dots,m\}.$$

Auch $\mathbf{H}$ ist eine Matrix von Rang k, also existieren nach Satz 4.1 Permutationen $\mathbf{P} \in \mathbb{K}^{n\times n}$ und $\mathbf{Q} \in \mathbb{K}^{m\times m}$ sowie eine untere Dreiecksmatrix $\mathbf{L} \in \mathbb{K}^{n\times k}$ und eine obere Dreiecksmatrix $\mathbf{R} \in \mathbb{K}^{m\times k}$ mit

$$\mathbf{PHQ}^* = \mathbf{LR}.$$

Wie setzen $\widehat{\mathbf{G}} := \mathbf{PHQ}^*$. Da $\mathbf{P}$ und $\mathbf{Q}$ Permutationsmatrizen sind, können wir bijektive Abbildungen

$$\hat{\iota}_n : \{1,\dots,n\} \to \mathcal{I}, \qquad \hat{\iota}_m : \{1,\dots,m\} \to \mathcal{J}$$

so finden, dass

$$\widehat{G}_{ij} = G_{\hat{\iota}_n(i),\hat{\iota}_m(j)} \qquad \text{für alle } i \in \{1,\dots,n\}, j \in \{1,\dots,m\}$$

gilt. Wir zerlegen die Matrizen in der Form

$$\mathbf{G}_{11} := \begin{pmatrix} \widehat{G}_{11} & \dots & \widehat{G}_{1k} \\ \vdots & \ddots & \vdots \\ \widehat{G}_{k1} & \dots & \widehat{G}_{kk} \end{pmatrix}, \qquad \mathbf{G}_{12} := \begin{pmatrix} \widehat{G}_{1,k+1} & \dots & \widehat{G}_{1m} \\ \vdots & \ddots & \vdots \\ \widehat{G}_{k,k+1} & \dots & \widehat{G}_{km} \end{pmatrix},$$

$$\mathbf{G}_{21} := \begin{pmatrix} \widehat{G}_{k+1,1} & \dots & \widehat{G}_{k+1,k} \\ \vdots & \ddots & \vdots \\ \widehat{G}_{n1} & \dots & \widehat{G}_{nk} \end{pmatrix}, \qquad \mathbf{G}_{22} := \begin{pmatrix} \widehat{G}_{k+1,k+1} & \dots & \widehat{G}_{k+1,m} \\ \vdots & \ddots & \vdots \\ \widehat{G}_{n,k+1} & \dots & \widehat{G}_{nm} \end{pmatrix},$$

$$\mathbf{L}_1 := \begin{pmatrix} L_{11} & \dots & L_{1k} \\ \vdots & \ddots & \vdots \\ L_{k1} & \dots & L_{kk} \end{pmatrix}, \qquad \mathbf{L}_2 := \begin{pmatrix} L_{k+1,1} & \dots & L_{k+1,k} \\ \vdots & \ddots & \vdots \\ L_{n1} & \dots & L_{nk} \end{pmatrix},$$

$$\mathbf{R}_1 := \begin{pmatrix} R_{11} & \dots & R_{1k} \\ \vdots & \ddots & \vdots \\ R_{k1} & \dots & R_{kk} \end{pmatrix}, \qquad \mathbf{R}_2 := \begin{pmatrix} R_{1,k+1} & \dots & R_{1m} \\ \vdots & \ddots & \vdots \\ R_{k,k+1} & \dots & R_{km} \end{pmatrix},$$

und erhalten

$$\begin{pmatrix} \mathbf{G}_{11} & \mathbf{G}_{12} \\ \mathbf{G}_{21} & \mathbf{G}_{22} \end{pmatrix} = \widehat{\mathbf{G}} = \mathbf{LR} = \begin{pmatrix} \mathbf{L}_1 \\ \mathbf{L}_2 \end{pmatrix} \begin{pmatrix} \mathbf{R}_1 & \mathbf{R}_2 \end{pmatrix} = \begin{pmatrix} \mathbf{L}_1\mathbf{R}_1 & \mathbf{L}_1\mathbf{R}_2 \\ \mathbf{L}_2\mathbf{R}_1 & \mathbf{L}_2\mathbf{R}_2 \end{pmatrix}.$$

Da die Diagonalelemente der Matrizen $\mathbf{L}_1$ und $\mathbf{R}_1$ nach Satz 4.1 nicht null sind, sind beide Matrizen regulär, also folgen

$$\mathbf{L}_2 = \mathbf{G}_{21}\mathbf{R}_1^{-1}, \qquad \mathbf{R}_2 = \mathbf{L}_1^{-1}\mathbf{G}_{12}$$

und damit auch

$$\begin{aligned} \widehat{\mathbf{G}} &= \begin{pmatrix} \mathbf{L}_1 \\ \mathbf{L}_2 \end{pmatrix} \begin{pmatrix} \mathbf{R}_1 & \mathbf{R}_2 \end{pmatrix} = \begin{pmatrix} \mathbf{L}_1\mathbf{R}_1 \\ \mathbf{G}_{21} \end{pmatrix} \mathbf{R}_1^{-1}\mathbf{L}_1^{-1} \begin{pmatrix} \mathbf{L}_1\mathbf{R}_1 & \mathbf{G}_{12} \end{pmatrix} \\ &= \begin{pmatrix} \mathbf{G}_{11} \\ \mathbf{G}_{21} \end{pmatrix} \mathbf{G}_{11}^{-1} \begin{pmatrix} \mathbf{G}_{11} & \mathbf{G}_{12} \end{pmatrix} = \widehat{\mathbf{G}}|_{n\times k}\widehat{\mathbf{G}}|_{k\times k}^{-1}\widehat{\mathbf{G}}|_{k\times m}. \end{aligned}$$

Indem wir

$$\mathcal{I}_k := \hat{\imath}_n(\{1,\ldots,k\}), \qquad \mathcal{J}_k := \hat{\imath}_m(\{1,\ldots,k\})$$

setzen folgt daraus die gewünschte Gleichung. ∎

Die dünne LR-Zerlegung bietet uns also einen sehr einfachen und eleganten Weg, um zu einer Rang-k-Matrix eine kompakte faktorisierte Rang-k-Darstellung zu konstruieren.

4.2 Adaptive Kreuzapproximation

In der Praxis kommen Rang-k-Matrizen eher selten vor, wesentlich häufiger sind Matrizen, die sich durch derartige Matrizen lediglich approximieren lassen. In diesem Fall greift bedauerlicherweise unsere Theorie nicht mehr: Wir können zwar weiterhin Pivot-Elemente wählen, aber es ist nicht garantiert, dass nach k Schritten bereits eine gute Approximation vorliegt. Immerhin gibt es eine Existenzaussage [18], die besagt, dass bei geschickter Wahl der Pivot-Elemente eine passable dünne LR-Zerlegung konstruiert werden kann, allerdings würde der korrespondierende Algorithmus viel zu aufwendig für den praktischen Einsatz sein.

Trotzdem besteht die Möglichkeit, eine Approximation zu konstruieren, indem man *adaptiv* vorgeht: Ausgehen von einem Rang von 1 konstruieren wir eine Folge von Approximationen mit zunehmendem Rang, die wir abbrechen, sobald der verbliebene Fehler klein genug geworden ist. Den Schlüssel zu dieser Vorgehensweise bietet die Gleichung (4.2): Die Zerlegung

$$\mathbf{H} = \underbrace{\begin{pmatrix} 1 \\ \mathbf{H}_{*1}H_{11}^{-1} \end{pmatrix}}_{=:\mathbf{A}_1} \underbrace{\begin{pmatrix} H_{11} & \mathbf{H}_{1*} \end{pmatrix}}_{=:\mathbf{B}_1^*} + \begin{pmatrix} 0 & \mathbf{0} \\ \mathbf{0} & \widehat{\mathbf{H}} \end{pmatrix}$$

```
procedure full_aca(var G, A, B, k);
k ← 0;
repeat
  k ← k + 1;
  Wähle i_k ∈ I und j_k ∈ J mit G_{i_k,j_k} ≠ 0;
  for i ∈ I do
    A_{ik} ← G_{i,j_k} / G_{i_k,j_k}
  end for;
  for j ∈ J do
    B_{jk} ← G_{i_k,j}
  end for;
  for i ∈ I, j ∈ J do
    G_{ij} ← G_{ij} − A_{ik} B_{jk}
  end for
until Restfehler G klein genug
```

Abbildung 4.1: Adaptive Kreuzapproximation für vollständig im Speicher vorliegende Matrizen $\mathbf{G} \in \mathbb{K}^{\mathcal{I}\times\mathcal{J}}$.

impliziert, dass $\mathbf{A}_1\mathbf{B}_1^*$ genau dann eine hinreichend gute Rang-1-Approximation der Matrix $\mathbf{H}$ ist, wenn der verbliebene Fehler $\widehat{\mathbf{H}}$ hinreichend klein ist.

Falls er es ist, können wir die Konstruktion abbrechen und $\mathbf{A}_1\mathbf{B}_1^*$ als Approximation verwenden. Anderenfalls können wir rekursiv eine Approximation $\widehat{\mathbf{A}}\widehat{\mathbf{B}}^*$ des Fehlers $\widehat{\mathbf{H}}$ berechnen, und falls sie hinreichend genau ist, wird

$$\mathbf{A}_1\mathbf{B}_1^* + \begin{pmatrix} 0 & \mathbf{0} \\ \mathbf{0} & \widehat{\mathbf{A}}\widehat{\mathbf{B}}^* \end{pmatrix} = \begin{pmatrix} \mathbf{A}_1 & \mathbf{0} \\ & \widehat{\mathbf{A}} \end{pmatrix} \begin{pmatrix} \mathbf{B}_1 & \mathbf{0} \\ & \widehat{\mathbf{B}} \end{pmatrix}^*$$

eine hinreichend genaue Approximation von $\mathbf{H}$ sein.

In der Praxis verwenden wir die Darstellung aus Satz 4.1 häufig in der Form

$$\mathbf{G} = \underbrace{\mathbf{P}^*\mathbf{L}}_{=:\widehat{\mathbf{L}}} \underbrace{\mathbf{R}\mathbf{Q}}_{=:\widehat{\mathbf{R}}}.$$

Dabei werden dann zwar $\widehat{\mathbf{L}}$ und $\widehat{\mathbf{R}}$ keine unteren beziehungsweise oberen Dreiecksmatrizen mehr sein, aber dafür erhalten wir direkt eine faktorisierte Rang-k-Darstellung in der üblichen Form. Der Verzicht auf die explizite Permutation bedeutet, dass nun nicht mehr die erste Spalte und die erste Zeile eliminiert werden, sondern die Spalte und Zeile, die sich in dem ausgewählten Pivot-Element kreuzen. Dadurch ist die Bezeichnung *Kreuzapproximation* dieser Klasse von Verfahren motiviert.

Der resultierende Algorithmus ist in Abbildung 4.1 dargestellt: Es wird ein Pivot-Element ausgewählt, beispielsweise indem man den betragsgrößten Koeffizienten verwendet, dann werden Spalte und Zeile zu der faktorisierten Rang-k-Darstellung hinzugefügt und der verbliebene Fehler berechnet, indem ihr Produkt von der bisherigen Matrix

```
procedure partial_aca(G, var A, B, k);
k ← 0;
repeat
  k ← k + 1;
  Wähle i_k ∈ I und j_k ∈ J geeignet;
  for j ∈ J do
    B_jk ← G_{i_k,j};
    for ℓ ∈ {1, ..., k − 1} do
      B_jk ← B_jk − B_jℓ A_{i_k,ℓ}
    end for
  end for;
  for i ∈ I do
    A_ik ← G_{i,j_k};
    for ℓ ∈ {1, ..., k − 1} do
      A_ik ← A_ik − A_iℓ B_{j_k,ℓ}
    end for;
    A_ik ← A_ik / B_{j_k,k}
  end for;
until Restfehler klein genug
```

Abbildung 4.2: Adaptive Kreuzapproximation für Matrizen $\mathbf{G} \in \mathbb{K}^{\mathcal{I}\times\mathcal{J}}$, von denen nur einzelne Zeilen und Spalten vorliegen.

subtrahiert wird. Falls der so berechnete Fehler noch zu groß ist, wird die Prozedur wiederholt.

Dieser Algorithmus hat den Vorteil, dass er garantiert, dass der verbliebene Fehler klein genug sein wird. Es wird allerdings nicht garantiert, dass der Rang der resultierenden Approximation in der Nähe des theoretisch optimalen Rangs liegen wird. Die Erfahrung zeigt allerdings, dass der Algorithmus gut funktioniert und keine unangemessen hohen Ränge verwendet.

Ein Nachteil des in Abbildung 4.1 dargestellten Verfahrens liegt darin, dass die gesamte Matrix $\mathbf{X}$ vorliegen muss, bevor der Algorithmus seine Arbeit aufnehmen kann. In der Regel möchte man genau das vermeiden, denn die Berechnung des Fehlers erfordert unter diesen Umständen jeweils $(\#\mathcal{I})(\#\mathcal{J})$ Operationen, so dass der Algorithmus eine quadratische Komplexität aufweist.

Einen Ausweg legt uns Folgerung 4.2 nahe: Für die Berechnung der Approximation sind nur k Zeilen und Spalten erforderlich, warum sollten wir also alle Zeilen und Spalten berechnen? Für den ersten Schritt der Approximation benötigen wir nur eine Zeile und eine Spalte der Ausgangsmatrix, also berechnen wir auch nur diese eine Zeile und eine Spalte. Für den zweiten Schritt benötigen wir je eine weitere Zeile und Spalte, allerdings des Restfehlers, der entsteht, wenn wir die bereits bekannte erste Approximation subtrahieren. Im $(k+1)$-ten Schritt müssen wir eine Zeile und Spalte wählen und die bekannte k-te Approximation abziehen.

Der resultierende Algorithmus ist in Abbildung 4.2 angegeben. Zunächst werden wieder Pivot-Indizes i_k und j_k ausgewählt, dann wird die i_k-te Zeile des Fehlers berechnet, daraufhin die j_k-te Spalte. Da nach der Berechnung der Zeile auch das Diagonalelement bereits in $B_{j_k,k}$ zur Verfügung steht, können wir die Skalierung der neuen Spalte in derselben Schleife bewerkstelligen.

Die partielle adaptive Kreuzapproximation hat den Vorteil, dass im ℓ-ten Schritt nur eine Zeile und eine Spalte der zu approximierenden Matrix benötigt werden, die sich in der Regel mit einem Aufwand proportional zu $\#\mathcal{J}$ beziehungsweise $\#\mathcal{I}$ berechnen lassen. Die Subtraktion der bereits vorhandenen Approximation erfordert $(\#\mathcal{I} + \#\mathcal{J})(\ell - 1)$ Operationen, so dass der gesamte Algorithmus wegen

$$\sum_{\ell=1}^{k} (\#\mathcal{I} + \#\mathcal{J})\ell = (\#\mathcal{I} + \#\mathcal{J}) \sum_{\ell=1}^{k} \ell = (\#\mathcal{I} + \#\mathcal{J}) \frac{k(k+1)}{2}$$

nur eine zu $(\#\mathcal{I} + \#\mathcal{J})k^2$ proportionale Anzahl von Rechenoperationen benötigt und deshalb schon für moderate Matrixgrößen wesentlich schneller als die Variante mit vollständig berechneter Matrix arbeitet.

Allerdings weist die Vorgehensweise mit partieller Matrix auch zwei große Nachteile auf: Erstens liegt der Fehler nicht mehr explizit vor, wir müssen also versuchen, ihn abzuschätzen, und falls unsere Schätzung zu ungenau ist, kann die berechnete Rang-k-Approximation wesentlich ungenauer als verlangt ausfallen. Zweitens müssen wir die Pivot-Indizes i_k und j_k wählen, ohne die gesamte Matrix zu kennen, wir können also beispielsweise nicht mehr den betragsgrößten Koeffizienten verwenden, weil wir ihn nicht finden können, ohne die gesamte Matrix zu durchsuchen.

Beispiel 4.3 (Pivotstrategie) *Als Beispiel für die Schwierigkeiten, die bei der Suche nach einer effizienten Pivotstrategie auftreten können, kann die besonders einfache Kernfunktion*

$$g : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}, \qquad (\mathbf{x}, \mathbf{y}) \mapsto (x_1 - y_1 - 10)(x_1 - y_1 - 9),$$

dienen. Offenbar ist g ein quadratisches Polynom, lässt sich also per Interpolation zweiter Ordnung exakt darstellen.

Wir fixieren $n, m \in \mathbb{N}$ und setzen $\mathcal{I} := \{1, \ldots, n\}$ sowie $\mathcal{J} := \{1, \ldots, m\}$. Nun wählen wir einen beliebigen Zeilenindex $\nu \in \{1, \ldots, n\}$ und einen beliebigen Spaltenindex $\mu \in \{1, \ldots, m\}$ und definieren Punkte

$$\mathbf{x}_i := \begin{cases} \begin{pmatrix} 10 \\ i \end{pmatrix} & \text{falls } i \neq \nu, \\ \begin{pmatrix} 9 \\ i \end{pmatrix} & \text{ansonsten} \end{cases}, \quad \mathbf{y}_j := \begin{cases} \begin{pmatrix} 0 \\ j \end{pmatrix} & \text{falls } j \neq \mu, \\ \begin{pmatrix} 1 \\ j \end{pmatrix} & \text{ansonsten} \end{cases} \qquad \text{für alle } i \in \mathcal{I}, j \in \mathcal{J}.$$

Für die durch

$$G_{ij} := g(\mathbf{x}_i, \mathbf{y}_j) \qquad \text{für alle } i \in \{1, \ldots, n\}, j \in \{1, \ldots, m\}$$

definierte Matrix $\mathbf{G} \in \mathbb{K}^{n \times m}$ *kann man leicht nachrechnen, dass*

$$G_{ij} = \begin{cases} 1 & \text{falls } i = \nu, j = \mu, \\ 0 & \text{ansonsten} \end{cases} \qquad \text{für alle } i \in \mathcal{I},\ j \in \mathcal{J}$$

gilt, die Matrix ist also sicherlich von Rang 1. *Da wir die Position des einzigen von null verschiedenen Eintrags mit Hilfe der Parameter* ν *und* μ *beliebig festlegen können, ist davon auszugehen, dass jede Pivotstrategie, die ausschließlich einzelne Koeffizienten der Matrix abfragen darf, sehr lange brauchen wird, um diesen Eintrag zu finden.*

4.3 Hybride Kreuzapproximation

Wie wir gesehen haben hat die vollständige Pivotsuche den Nachteil, dass sie einen zu hohen Rechenaufwand erfordert, während die partielle Pivotsuche entweder ähnlich aufwendig oder unzuverlässig ist. Befriedigend sind beide Ansätze nicht.

Wir sind daran interessiert, eine zuverlässige und trotzdem effiziente Variante der Kreuzapproximation zu konstruieren. Wie wir gesehen haben, ist das alleine aufgrund der Matrix $\mathbf{G}$ nicht möglich, wir sollten also zusätzliche Informationen über die Herkunft der Matrix hinzunehmen.

Ausgangspunkt unserer Betrachtungen ist die Kreuzapproximation mit vollständiger Pivotsuche. Wir haben bereits gesehen, dass sie es uns ermöglicht, zu jeder gewünschten Genauigkeit $\epsilon_{\text{aca}} \in \mathbb{R}_{>0}$ Teilmengen $\mathcal{I}_k$ und $\mathcal{J}_k$ der Mächtigkeit k zu konstruieren, die

$$\|\mathbf{G} - \mathbf{G}|_{\mathcal{I} \times \mathcal{J}_k} \mathbf{G}|^{-1}_{\mathcal{I}_k \times \mathcal{J}_k} \mathbf{G}|_{\mathcal{I}_k \times \mathcal{J}}\|_2 \leq \epsilon_{\text{aca}}$$

erfüllen (siehe Folgerung 4.2 und die Fehlerdarstellung aus (4.2)). Leider erfordert das Aufstellen der Matrix $\mathbf{G}$ so viele Rechenoperationen, dass dieser Ansatz für größere Matrizen zu aufwendig ist.

Die Idee besteht darin, statt der Matrix $\mathbf{G}$ eine kleinere Matrix zu verwenden, die aus der Interpolation der Kernfunktion entsteht: Wir wählen einen zulässigen Block

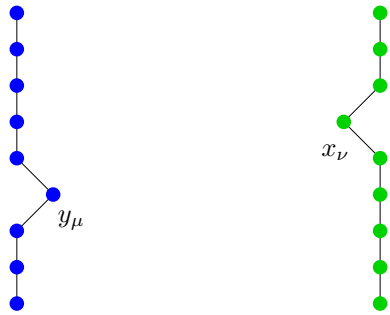

Abbildung 4.3: In Beispiel 4.3 verwendete Geometrie

$b = (t, s)$, der die starke Zulässigkeitsbedingung

$$\max\{\operatorname{diam}(Q_t), \operatorname{diam}(Q_s)\} \leq \eta \operatorname{dist}(Q_t, Q_s) \tag{4.3}$$

erfüllt. Dann können wir eine Approximation der Kernfunktion durch Interpolation *in beiden Variablen* definieren:

$$\hat{g}(\mathbf{x}, \mathbf{y}) := \sum_{\nu \in M} \sum_{\mu \in M} g(\xi_{Q_t,\nu}, \xi_{Q_s,\mu}) \mathcal{L}_{Q_t,\nu}(\mathbf{x}) \mathcal{L}_{Q_s,\mu}(\mathbf{y}) \qquad \text{für alle } \mathbf{x} \in Q_t,\ \mathbf{y} \in Q_s. \tag{4.4}$$

Aus Satz 3.9 folgt für eine asymptotisch glatte Kernfunktion (vgl. Definition 3.10) die Fehlerabschätzung

$$|g(\mathbf{x}, \mathbf{y}) - \hat{g}(\mathbf{x}, \mathbf{y})| \leq \frac{C(m)}{\operatorname{dist}(Q_t, Q_s)^\sigma} \left(\frac{c_0 \eta}{4}\right)^{m+1} \qquad \text{für alle } \mathbf{x} \in Q_t,\ \mathbf{y} \in Q_s,$$

wobei wieder zur Abkürzung das Polynom

$$C(m) := 4dC_{\text{as}}(m+1)^{2d-1} \frac{(m+\sigma)!}{(m+1)!}$$

verwendet wird. Bei geeigneter Wahl von η und m können wir dafür sorgen, dass der Interpolationsfehler eine Gleichung der Form

$$\|g - \hat{g}\|_{\infty, Q_t \times Q_s} \leq \epsilon_{\text{int}}$$

erfüllt. Wir haben bereits gesehen, dass Interpolation zu einer Approximation des Rangs $(m+1)^d$ führt, und man kann sich überlegen, dass dieser Rang in vielen praktischen Fällen deutlich höher als unbedingt nötig ist. Eine eingehende Analyse legt nahe, dass die Kernfunktion g Eigenschaften besitzt, die es möglich machen sollten, sie durch niedrigeren Rang zu approximieren. Da die Kernfunktion bei der Interpolation nur in Gestalt der Interpolationskoeffizienten auftritt, bietet es sich deshalb an, deren Matrix $\mathbf{S} \in \mathbb{K}^{M \times M}$, gegeben durch

$$S_{\nu\mu} := g(\xi_{Q_t,\nu}, \xi_{Q_s,\mu}) \qquad \text{für alle } \nu \in M,\ \mu \in M,$$

genauer zu untersuchen. Diese Matrix besitzt zwar $(m+1)^d$ Zeilen und Spalten, wird aber in der Regel wesentlich kleiner als der zu approximierende Matrixblock sein. Deshalb ist es möglich, sie durch eine Kreuzapproximation mit vollständiger Pivotsuche anzunähern, die dann nur eine zu $k(m+1)^{2d}$ proportionale Anzahl von Rechenoperationen benötigt.

Wir gehen also davon aus, dass wir Indexmengen $M_r, M_c \subseteq M$ der Mächtigkeit k so gefunden haben, dass $\mathbf{S}|_{M_r \times M_c}$ regulär ist und

$$\|\mathbf{S} - \mathbf{S}|_{M \times M_c} \mathbf{S}|_{M_r \times M_c}^{-1} \mathbf{S}|_{M_r \times M}\|_2 \leq \epsilon_{\text{aca}}$$

gilt. Wir bezeichnen den mittleren Faktor mit

$$\widehat{\mathbf{G}} := \mathbf{S}|_{M_r \times M_c}^{-1}$$

und ersetzen in der Gleichung (4.4) nun die exakten Koeffizienten durch die mittels der Kreuzapproximation genäherten:

$$\begin{aligned}
\hat{g}(\mathbf{x},\mathbf{y}) &:= \sum_{\nu\in M}\sum_{\mu\in M} \mathbf{S}_{\nu\mu}\mathcal{L}_{Q_t,\nu}(\mathbf{x})\mathcal{L}_{Q_s,\mu}(\mathbf{y}) \\
&\approx \sum_{\nu\in M}\sum_{\mu\in M} (\mathbf{S}|_{M\times M_c}\widehat{\mathbf{G}}\mathbf{S}|_{M_r\times M})_{\nu\mu}\mathcal{L}_{Q_t,\nu}(\mathbf{x})\mathcal{L}_{Q_s,\mu}(\mathbf{y}) \\
&= \sum_{\nu\in M}\sum_{\mu\in M}\sum_{\alpha\in M_r}\sum_{\beta\in M_c} S_{\nu\beta}\widehat{G}_{\beta\alpha}S_{\alpha\mu}\mathcal{L}_{Q_t,\nu}(\mathbf{x})\mathcal{L}_{Q_s,\mu}(\mathbf{y}) \\
&= \sum_{\alpha\in M_r}\sum_{\beta\in M_c} \widehat{G}_{\beta\alpha}\left(\sum_{\nu\in M} S_{\nu\beta}\mathcal{L}_{Q_t,\nu}(\mathbf{x})\right)\left(\sum_{\mu\in M} S_{\alpha\mu}\mathcal{L}_{Q_s,\mu}(\mathbf{y})\right) =: \check{g}(\mathbf{x},\mathbf{y}).
\end{aligned}$$

Eine genauere Untersuchung der letzten beiden Summen ergibt

$$\begin{aligned}
\sum_{\nu\in M} S_{\nu\beta}\mathcal{L}_{Q_t,\nu}(\mathbf{x}) &= \sum_{\nu\in M} g(\xi_{Q_t,\nu},\xi_{Q_s,\beta})\mathcal{L}_{Q_t,\nu}(\mathbf{x}) \approx g(\mathbf{x},\xi_{Q_s,\beta}), \\
\sum_{\mu\in M} S_{\alpha\mu}\mathcal{L}_{Q_s,\mu}(\mathbf{y}) &= \sum_{\mu\in M} g(\xi_{Q_t,\alpha},\xi_{Q_s,\mu})\mathcal{L}_{Q_s,\mu}(\mathbf{y}) \approx g(\xi_{Q_t,\alpha},\mathbf{y}),
\end{aligned}$$

sie beschreiben also gerade Interpolanten der Kernfunktion. Wenn diese Interpolanten die Kernfunktion gut approximieren, dann wird die Kernfunktion auch die Interpolanten gut approximieren, also dürfen wir hoffen, dass

$$\tilde{g}(\mathbf{x},\mathbf{y}) := \sum_{\alpha\in M_r}\sum_{\beta\in M_c} \widehat{G}_{\beta\alpha}g(\mathbf{x},\xi_{Q_s,\beta})g(\xi_{Q_t,\alpha},\mathbf{y}) \qquad \text{für alle } \mathbf{x}\in Q_t,\ \mathbf{y}\in Q_s$$

eine brauchbare Näherung der Kernfunktion g ist. Da $\#M_r = \#M_c = k \leq (m+1)^d$ gilt, ist der Rang der resultierenden Rang-k-Darstellung jedenfalls nicht größer als für die zugrundeliegende Interpolation. In der Praxis zeigt sich allerdings, dass der Rang häufig wesentlich geringer ist und dass bei gleicher Ordnung m eine wesentlich bessere Genauigkeit als bei der Interpolation erreicht wird.

Es lässt sich leicht nachprüfen, dass

$$\tilde{g}(\xi_{Q_t,\nu},\xi_{Q_s,\mu}) = \check{g}(\xi_{Q_t,\nu},\xi_{Q_s,\mu}) \qquad \text{für alle } \nu,\mu\in M$$

gilt, also ist $\check{g}$ ein Interpolant der approximativen Kernfunktion $\tilde{g}$.

Damit können wir die Fehleranalyse der hybriden Kreuzapproximation auf die Ergebnisse aus Satz 3.9 zurückführen: $\hat{g}$ ist ein Interpolant der Kernfunktion g, und Interpolationsfehler asymptotisch glatter Kernfunktionen können wir auf zulässigen Gebieten beherrschen. $\hat{g}$ und $\check{g}$ sind Interpolanten zu um ϵ_{aca} gestörten Interpolationspunkten, und mit Hilfe der Stabilitätsabschätzung aus dem erwähnten Satz folgt daraus, dass sich die Interpolanten nur um einen Faktor $\Lambda^d\epsilon_{\text{aca}}$ unterscheiden können. $\check{g}$ ist ein Interpolant der Kernfunktion $\tilde{g}$, und es lässt sich nachweisen, dass $\tilde{g}$ von g die asymptotische Glattheit erbt, so dass sich auch hier der Interpolationsfehler steuern lässt.

Kurz gesagt: Wenn wir die Ordnung m und die Genauigkeit ϵ_{aca} der adaptiven Kreuzapproximation geeignet wählen, können wir auch bei der hybriden Kreuzapproximation jede gewünschte Genauigkeit erreichen.

5 Norm- und Fehlerabschätzungen

Wir haben verschiedene Techniken kennen gelernt, mit denen sich in einem zulässigen Block eine Rang-k-Approximation einer Matrix konstruieren lässt, beispielsweise per Interpolation oder Kreuzapproximation. Für die meisten dieser Verfahren (mit Ausnahme der adaptiven Kreuzapproximation mit partieller Pivotsuche) besteht auch die Möglichkeit, den Approximationsfehler innerhalb des betreffenden Blocks abzuschätzen.

In der Praxis sind solche Abschätzungen des blockweisen Fehlers weniger von Interesse, viel wichtiger sind Abschätzungen des Gesamtfehlers für die vollständige approximierte Matrix.

In diesem Kapitel beschäftigen wir uns mit der Frage, wie sich globale Normen aus lokalen zusammensetzen lassen und wie sich Normen hierarchischer Matrizen näherungsweise berechnen lassen.

5.1 Frobenius-Norm

Eine besonders einfach handzuhabende Matrixnorm ist die *Frobenius-Norm*, die der euklidischen Norm entspricht, wenn wir die Matrizen als Vektoren über $\mathcal{I} \times \mathcal{J}$ interpretieren. Zu beachten ist, dass sie sich von der von den euklidischen Normen auf $\mathbb{K}^{\mathcal{I}}$ und $\mathbb{K}^{\mathcal{J}}$ induzierten Matrixnorm unterscheidet.

Definition 5.1 (Frobenius-Norm) *Die Abbildung*

$$\|\cdot\|_F : \mathbb{K}^{\mathcal{I}\times\mathcal{J}} \to \mathbb{R}_{\geq 0}, \qquad \mathbf{X} \mapsto \left(\sum_{i\in\mathcal{I}} \sum_{j\in\mathcal{J}} |X_{ij}|^2 \right)^{1/2},$$

ist eine Norm auf dem Raum $\mathbb{K}^{\mathcal{I}\times\mathcal{J}}$, die wir als die Frobenius-Norm *bezeichnen. Wir notieren die Frobenius-Norm einer Matrix $\mathbf{X} \in \mathbb{K}^{\mathcal{I}\times\mathcal{J}}$ als $\|\mathbf{X}\|_F$.*

Der entscheidende Vorteil der Frobenius-Norm besteht darin, dass sich die Frobenius-Norm der Gesamtmatrix direkt aus den Frobenius-Normen der Teilmatrizen zusammensetzen lässt:

Lemma 5.2 (Globale Frobenius-Norm) *Sei $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Blockbaum mit Blattmenge $\mathcal{L}_{\mathcal{I}\times\mathcal{J}}$. Dann gilt*

$$\|\mathbf{X}\|_F = \left(\sum_{b=(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}} \|\mathbf{X}|_{\hat{t}\times\hat{s}}\|_F^2 \right)^{1/2} \qquad \text{für alle } \mathbf{X} \in \mathbb{K}^{\mathcal{I}\times\mathcal{J}}.$$

Beweis. Aus Folgerung 2.22 wissen wir, dass die Beschriftungen der Blätter des Blockbaums $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ eine disjunkte Partition der Menge $\mathcal{I}\times\mathcal{J}$ bilden. Also gilt

$$\|\mathbf{X}\|_F^2 = \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} |X_{ij}|^2 = \sum_{b=(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}}\sum_{i\in\hat{t}}\sum_{j\in\hat{s}} |X_{ij}|^2 = \sum_{b=(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}} \|\mathbf{X}|_{\hat{t}\times\hat{s}}\|_F^2,$$

und das ist auch schon die gesuchte Gleichung. ∎

Mit Hilfe dieses einfachen Lemmas lassen sich globale Fehlerabschätzungen einfach gewinnen. Als Beispiel untersuchen wir den Fall einer Matrix, die aus der Auswertung einer Kernfunktion

$$g : \Omega\times\Omega \to \mathbb{K}, \qquad \mathbb{K}\in\{\mathbb{R},\mathbb{C}\}$$

in Punkten $(x_i)_{i\in\mathcal{I}}$ und $(y_j)_{j\in\mathcal{J}}$ in Ω entsteht: $\mathbf{G}\in\mathbb{K}^{\mathcal{I}\times\mathcal{J}}$ ist definiert durch

$$G_{ij} := g(x_i, y_j) \qquad \text{für alle } i\in\mathcal{I},\ j\in\mathcal{J}.$$

Um diese Matrix durch eine $\mathcal{H}$-Matrix zu approximieren, wählen wir Clusterbäume $\mathcal{T}_{\mathcal{I}}$ und $\mathcal{T}_{\mathcal{J}}$ sowie einen passenden zulässigen Blockbaum $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$. Für jedes zulässige Blatt $b=(t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ gehen wir davon aus, dass eine entartete Kernfunktion

$$\tilde{g}_b(x,y) = \sum_{\nu=1}^{k} a_{b,\nu}(x)\bar{b}_{b,\nu}(y) \qquad \text{für alle } x\in\Omega_t,\ y\in\Omega_s$$

existiert, die die ursprüngliche Kernfunktion hinreichend gut approximiert, also

$$\|g-\tilde{g}_b\|_{\infty,\Omega_t\times\Omega_s} \leq \epsilon \qquad \text{für alle } b\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$$

mit einem geeigneten $\epsilon\in\mathbb{R}_{\geq 0}$ erfüllt (vgl. (3.10) für die Abschätzung im Fall der Interpolation). Wie zuvor definieren wir die Approximation eines Matrixblocks $\widetilde{\mathbf{G}}|_{\hat{t}\times\hat{s}}$ durch

$$\widetilde{G}_{ij} := \tilde{g}_b(x_i,y_j) = \sum_{\nu=1}^{k} \underbrace{a_{b,\nu}(x_i)}_{=:A_{b,i\nu}} \underbrace{\bar{b}_{b,\nu}(y_j)}_{=:\bar{B}_{b,j\nu}} = (AB^*)_{ij} \qquad \text{für alle } i\in\hat{t},\ j\in\hat{s}$$

und haben unsere $\mathcal{H}$-Matrix-Approximation konstruiert.

Lemma 5.3 (Frobenius-Norm des Fehlers) *Es gilt*

$$\|\mathbf{G}|_{\hat{t}\times\hat{s}} - \widetilde{\mathbf{G}}|_{\hat{t}\times\hat{s}}\|_F \leq (\#\hat{t})^{1/2}(\#\hat{s})^{1/2}\epsilon \qquad \textit{für alle } b=(t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$$

und damit

$$\|\mathbf{G}-\widetilde{\mathbf{G}}\|_F \leq (\#\mathcal{I})^{1/2}(\#\mathcal{J})^{1/2}\epsilon.$$

Beweis. Mit der Definition der Frobenius-Norm erhalten wir direkt

$$\begin{aligned}\|\mathbf{G}|_{\hat{t}\times\hat{s}} - \widetilde{\mathbf{G}}|_{\hat{t}\times\hat{s}}\|_F^2 &= \sum_{i\in\hat{t}}\sum_{j\in\hat{s}} |G_{ij} - \widetilde{G}_{ij}|^2 = \sum_{i\in\hat{t}}\sum_{j\in\hat{s}} |g(x_i,y_j) - \tilde{g}_b(x_i,y_j)|^2 \\ &\leq \sum_{i\in\hat{t}}\sum_{j\in\hat{s}} \epsilon^2 = (\#\hat{t})(\#\hat{s})\epsilon^2.\end{aligned}$$

Mit Lemma 5.2 folgt daraus

$$\|\mathbf{G} - \widetilde{\mathbf{G}}\|_F^2 = \sum_{b=(t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}} \|\mathbf{G}|_{\hat{t}\times\hat{s}} - \widetilde{\mathbf{G}}|_{\hat{t}\times\hat{s}}\|_F^2 \leq \sum_{b=(t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}} (\#\hat{t})(\#\hat{s})\epsilon^2. \tag{5.1}$$

Da nach Folgerung 2.22 die Mengen $\hat{t}\times\hat{s}$ für $(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ eine disjunkte Partition der Menge $\mathcal{I}\times\mathcal{J}$ bilden, erhalten wir

$$\begin{aligned}\|\mathbf{G} - \widetilde{\mathbf{G}}\|_F^2 &\leq \sum_{b=(t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}} (\#\hat{t})(\#\hat{s})\epsilon^2 = \epsilon^2 \sum_{b=(t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}} \#(\hat{t}\times\hat{s}) \\ &= \epsilon^2 \# \bigcup_{b=(t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}} \hat{t}\times\hat{s} \leq \epsilon^2 \#(\mathcal{I}\times\mathcal{J}) = \epsilon^2(\#\mathcal{I})(\#\mathcal{J}).\end{aligned}$$

Die gewünschte Abschätzung folgt, indem wir die Wurzel aus beiden Seiten dieser Ungleichung ziehen. ∎

5.2 Spektralnorm

In vielen Anwendungen sind wir eher an *induzierten* Matrixnormen interessiert, weil sie es uns ermöglichen, Aussagen über die „Wirkung“ einer Matrix auf Vektoren zu treffen. Eine der wichtigsten solchen Normen ist die *Spektralnorm*, die von der euklidischen Norm induziert wird.

Definition 5.4 (Spektralnorm) *Die Abbildung*

$$\|\cdot\|_2 : \mathbb{K}^{\mathcal{I}\times\mathcal{J}} \to \mathbb{R}_{\geq 0}, \qquad \mathbf{X} \mapsto \sup\left\{\frac{\|\mathbf{X}\mathbf{y}\|_2}{\|\mathbf{y}\|_2} \;:\; \mathbf{y}\in\mathbb{K}^{\mathcal{J}}\setminus\{\mathbf{0}\}\right\},$$

ist eine Norm auf dem Raum $\mathbb{K}^{\mathcal{I}\times\mathcal{J}}$, die wir als die Spektralnorm *bezeichnen. Wir notieren die Spektralnorm einer Matrix $\mathbf{X}\in\mathbb{K}^{\mathcal{I}\times\mathcal{J}}$ als $\|\mathbf{X}\|_2$.*

Die Spektralnorm ist mit der euklidischen Norm *verträglich*, es gilt nämlich

$$\|\mathbf{X}\mathbf{y}\|_2 \leq \|\mathbf{X}\|_2\|\mathbf{y}\|_2 \qquad \text{für alle } \mathbf{X}\in\mathbb{K}^{\mathcal{I}\times\mathcal{J}},\ \mathbf{y}\in\mathbb{K}^{\mathcal{J}}, \tag{5.2}$$

falls also $\mathbf{X}$ der Fehler einer $\mathcal{H}$-Matrix-Approximation ist, beschreibt $\|\mathbf{X}\|_2$, wie sehr Matrix-Vektor-Produkte mit der Approximation von solchen mit der ursprünglichen Matrix abweichen.

Die Abschätzung lässt sich auf Produkte von Matrizen übertragen: Indem wir (5.2) in die Definition der Spektralnorm einsetzen, erhalten wir

$$\|\mathbf{XY}\|_2 \leq \|\mathbf{X}\|_2\|\mathbf{Y}\|_2 \qquad \text{für alle } \mathbf{X} \in \mathbb{K}^{\mathcal{I}\times\mathcal{J}},\ \mathbf{Y} \in \mathbb{K}^{\mathcal{J}\times\mathcal{K}}. \tag{5.3}$$

In vielen Anwendungsfällen sind wir daran interessiert, ein lineares Gleichungssystem

$$\mathbf{Gx} = \mathbf{b}$$

zu lösen, müssen aber aus Gründen der Effizienz die Matrix durch eine Approximation $\widetilde{\mathbf{G}}$ ersetzen, so dass wir das gestörte System

$$\widetilde{\mathbf{G}}\widetilde{\mathbf{x}} = \mathbf{b}$$

lösen, das in der Regel eine andere Lösung besitzen wird. Damit stellen sich uns zwei Fragen: Falls $\mathbf{G}$ regulär ist, ist es dann auch $\widetilde{\mathbf{G}}$? Und falls ja, wie sehr unterscheiden sich die exakte Lösung $\mathbf{x}$ und die Näherung $\widetilde{\mathbf{x}}$?

Offenbar hängen die Antworten auf beide Fragen davon ab, wie „nahe“ sich $\mathbf{G}$ und $\widetilde{\mathbf{G}}$ in einem geeigneten Sinn sind. Der naheliegende Ansatz, einfach $\|\mathbf{G}-\widetilde{\mathbf{G}}\|_2$ als Ausgangspunkt zu verwenden, ist wenig elegant, denn er berücksichtigt nicht, dass die Matrix $\mathbf{G}$ unterschiedliche Vektoren sehr unterschiedlich beeinflussen kann. Geschickter ist es, die Größe

$$\|\mathbf{G}^{-1}(\mathbf{G}-\widetilde{\mathbf{G}})\|_2 = \|\mathbf{I}-\mathbf{G}^{-1}\widetilde{\mathbf{G}}\|_2$$

zu untersuchen. Wir verwenden dazu die *Neumannsche Reihe*:

Lemma 5.5 (Neumannsche Reihe) *Sei $\mathbf{X} \in \mathbb{K}^{\mathcal{I}\times\mathcal{I}}$ mit $\|\mathbf{X}\|_2 < 1$ gegeben. Dann ist $\mathbf{I}-\mathbf{X}$ regulär und es gilt*

$$\|(\mathbf{I}-\mathbf{X})^{-1}\|_2 \leq \frac{1}{1-\|\mathbf{X}\|_2}. \tag{5.4}$$

Beweis. Wir untersuchen die Matrizen

$$\mathbf{Y}^{(m)} := \sum_{k=0}^{m} \mathbf{X}^k \qquad \text{für alle } m \in \mathbb{N}_0.$$

Durch Kombination der Abschätzung (5.3) mit der geometrischen Summenformel erhalten wir

$$\sum_{k=0}^{m} \|\mathbf{X}^k\|_2 \leq \sum_{k=0}^{m} \|\mathbf{X}\|_2^k = \frac{1-\|\mathbf{X}\|_2^{m+1}}{1-\|\mathbf{X}\|_2} \leq \frac{1}{1-\|\mathbf{X}\|_2},$$

also ist die *Neumannsche Reihe*

$$\mathbf{Y} = \lim_{m\to\infty} \mathbf{Y}^{(m)} = \sum_{k=0}^{\infty} \mathbf{X}^k \tag{5.5}$$

absolut summierbar, damit insbesondere auch summierbar, und erfüllt

$$\|\mathbf{Y}\|_2 \leq \frac{1}{1-\|\mathbf{X}\|_2}.$$

Nun ist zu zeigen, dass $\mathbf{Y}$ die Inverse der Matrix $\mathbf{I} - \mathbf{X}$ ist. Dazu untersuchen wir

$$(\mathbf{I} - \mathbf{X})\mathbf{Y}^{(m)} = \sum_{k=0}^{m} \mathbf{X}^k - \sum_{k=0}^{m} \mathbf{X}^{k+1} = \sum_{k=0}^{m} \mathbf{X}^k - \sum_{k=1}^{m+1} \mathbf{X}^k = \mathbf{I} - \mathbf{X}^{m+1}. \tag{5.6}$$

Wieder mit (5.3) erhalten wir

$$\lim_{m\to\infty} \|\mathbf{X}^{m+1}\|_2 \leq \lim_{m\to\infty} \|\mathbf{X}\|_2^{m+1} = 0,$$

also folgt $(\mathbf{I} - \mathbf{X})\mathbf{Y} = \mathbf{I}$, indem wir in (5.6) zu dem Grenzwert für $m \to \infty$ übergehen. ∎

Die Neumannsche Reihe (5.5) wird nicht nur für theoretische Untersuchungen eingesetzt, sondern kann auch verwendet werden, um Lösungsverfahren für lineare Gleichungssysteme zu entwickeln.

Wir sind allerdings zunächst daran interessiert, mit ihrer Hilfe eine Aussage über die Lösbarkeit gestörter Gleichungssysteme herzuleiten:

Folgerung 5.6 (Gestörtes Gleichungssystem) *Seien $\mathbf{G}, \widetilde{\mathbf{G}} \in \mathbb{K}^{\mathcal{I}\times\mathcal{I}}$ gegeben. Sei $\mathbf{G}$ regulär und gelte*

$$\|\mathbf{G}^{-1}(\mathbf{G} - \widetilde{\mathbf{G}})\|_2 < 1.$$

Dann ist auch $\widetilde{\mathbf{G}}$ regulär und für jedes $\mathbf{b} \in \mathbb{K}^{\mathcal{I}} \setminus \{\mathbf{0}\}$ erfüllen die Lösungen $\mathbf{x}, \widetilde{\mathbf{x}} \in \mathbb{K}^{\mathcal{I}}$ der Gleichungssysteme

$$\mathbf{G}\mathbf{x} = \mathbf{b}, \qquad \widetilde{\mathbf{G}}\widetilde{\mathbf{x}} = \mathbf{b}$$

die Abschätzung

$$\frac{\|\mathbf{x} - \widetilde{\mathbf{x}}\|_2}{\|\mathbf{x}\|_2} \leq \frac{\|\mathbf{G}^{-1}(\mathbf{G} - \widetilde{\mathbf{G}})\|_2}{1 - \|\mathbf{G}^{-1}(\mathbf{G} - \widetilde{\mathbf{G}})\|_2}.$$

Beweis. Wir wenden Lemma 5.5 auf

$$\mathbf{X} := \mathbf{G}^{-1}(\mathbf{G} - \widetilde{\mathbf{G}}) = \mathbf{I} - \mathbf{G}^{-1}\widetilde{\mathbf{G}}$$

an und erhalten die Aussage, dass

$$\mathbf{I} - \mathbf{X} = \mathbf{G}^{-1}\widetilde{\mathbf{G}}$$

regulär ist, also muss auch $\widetilde{\mathbf{G}}$ regulär sein. Es folgt

$$\mathbf{x} - \widetilde{\mathbf{x}} = \mathbf{G}^{-1}\widetilde{\mathbf{G}}\widetilde{\mathbf{x}} - \widetilde{\mathbf{x}} = -(\mathbf{I} - \mathbf{G}^{-1}\widetilde{\mathbf{G}})\widetilde{\mathbf{x}} = -\mathbf{X}\widetilde{\mathbf{x}} = -\mathbf{X}\widetilde{\mathbf{G}}^{-1}\mathbf{G}\mathbf{x} = -\mathbf{X}(\mathbf{I} - \mathbf{X})^{-1}\mathbf{x},$$

so dass wir mit (5.3) und (5.4) die Abschätzung

$$\|\mathbf{x} - \widetilde{\mathbf{x}}\|_2 \leq \|\mathbf{X}\|_2\|(\mathbf{I} - \mathbf{X})^{-1}\|_2\|\mathbf{x}\|_2 \leq \frac{\|\mathbf{X}\|_2}{1 - \|\mathbf{X}\|_2}\|\mathbf{x}\|_2$$

erhalten, aus der unser Ergebnis folgt. ∎

Die für diese Aussage relevante Größe können wir mit (5.3) nach oben abschätzen, um

$$\|\mathbf{G}^{-1}(\mathbf{G}-\widetilde{\mathbf{G}})\|_2 \leq \|\mathbf{G}^{-1}\|_2\|\mathbf{G}-\widetilde{\mathbf{G}}\|_2 \leq \|\mathbf{G}^{-1}\|_2\|\mathbf{G}\|_2\frac{\|\mathbf{G}-\widetilde{\mathbf{G}}\|_2}{\|\mathbf{G}\|_2} = \kappa_2(\mathbf{G})\frac{\|\mathbf{G}-\widetilde{\mathbf{G}}\|_2}{\|\mathbf{G}\|_2},$$

zu erhalten. Sie kann also beschränkt werden durch das Produkt aus der *Konditionszahl* des ungestörten Gleichungssystems und dem relativen Fehler der Matrix.

Um Aussagen darüber treffen zu können, wie gut die Lösung des approximierten Gleichungssystems die exakte Lösung annähert, benötigen wir also eine Abschätzung der Spektralnorm.

Lemma 5.7 (Darstellung per Skalarprodukt) *Sei $\mathbf{X} \in \mathbb{K}^{\mathcal{I}\times\mathcal{J}}$. Dann gilt*

$$\|\mathbf{X}\|_2 = \sup\left\{\frac{|\langle\mathbf{y},\mathbf{X}\mathbf{z}\rangle_2|}{\|\mathbf{y}\|_2\|\mathbf{z}\|_2} \;:\; \mathbf{y}\in\mathbb{K}^{\mathcal{I}}\setminus\{\mathbf{0}\},\ \mathbf{z}\in\mathbb{K}^{\mathcal{J}}\setminus\{\mathbf{0}\}\right\}.$$

Beweis. Für alle $\mathbf{y} \in \mathbb{K}^{\mathcal{I}} \setminus \{\mathbf{0}\}$ und $\mathbf{z} \in \mathbb{K}^{\mathcal{J}} \setminus \{\mathbf{0}\}$ folgt aus der Cauchy-Schwarz-Ungleichung

$$\frac{|\langle\mathbf{y},\mathbf{X}\mathbf{z}\rangle_2|}{\|\mathbf{y}\|_2\|\mathbf{z}\|_2} \leq \frac{\|\mathbf{y}\|_2\|\mathbf{X}\mathbf{z}\|_2}{\|\mathbf{y}\|_2\|\mathbf{z}\|_2} \leq \frac{\|\mathbf{y}\|_2\|\mathbf{X}\|_2\|\mathbf{z}\|_2}{\|\mathbf{y}\|_2\|\mathbf{z}\|_2} = \|\mathbf{X}\|_2.$$

Umgekehrt können wir für jeden Vektor $\mathbf{z} \in \mathbb{K}^{\mathcal{J}}$ mit $\mathbf{X}\mathbf{z} \neq \mathbf{0}$ auch $\mathbf{y} := \mathbf{X}\mathbf{z}$ setzen und erhalten

$$\frac{|\langle\mathbf{y},\mathbf{X}\mathbf{z}\rangle_2|}{\|\mathbf{y}\|_2\|\mathbf{z}\|_2} = \frac{|\langle\mathbf{X}\mathbf{z},\mathbf{X}\mathbf{z}\rangle_2|}{\|\mathbf{y}\|_2\|\mathbf{X}\mathbf{z}\|_2} = \frac{\|\mathbf{X}\mathbf{z}\|_2^2}{\|\mathbf{z}\|_2\|\mathbf{X}\mathbf{z}\|_2} = \frac{\|\mathbf{X}\mathbf{z}\|_2}{\|\mathbf{z}\|_2},$$

also folgt auch

$$\sup\left\{\frac{|\langle\mathbf{y},\mathbf{X}\mathbf{z}\rangle_2|}{\|\mathbf{y}\|_2\|\mathbf{z}\|_2} \;:\; \mathbf{y}\in\mathbb{K}^{\mathcal{I}}\setminus\{\mathbf{0}\},\ \mathbf{z}\in\mathbb{K}^{\mathcal{J}}\setminus\{\mathbf{0}\}\right\} \geq \sup\left\{\frac{\|\mathbf{X}\mathbf{z}\|_2}{\|\mathbf{z}\|_2} \;:\; \mathbf{z}\in\mathbb{K}^{\mathcal{J}}\setminus\{\mathbf{0}\}\right\},$$

und damit die gesuchte Gleichung. ∎

Die etwa in Folgerung 5.6 benötigte Abschätzung der Spektralnorm lässt sich besonders einfach auf dem Umweg über die bereits diskutierte Frobeniusnorm gewinnen:

Lemma 5.8 (Spektral- und Frobeniusnorm) *Es gilt*

$$\|\mathbf{X}\|_2 \leq \|\mathbf{X}\|_F \leq \sqrt{\#\mathcal{J}}\|\mathbf{X}\|_2 \qquad \text{für alle } \mathbf{X}\in\mathbb{K}^{\mathcal{I}\times\mathcal{J}}.$$

Beweis. Sei $\mathbf{X} \in \mathbb{K}^{\mathcal{I}\times\mathcal{J}}$. Wir wenden die Cauchy-Schwarz-Ungleichung an, um für Vektoren $\mathbf{y} \in \mathbb{K}^{\mathcal{I}}$ und $\mathbf{z} \in \mathbb{K}^{\mathcal{J}}$ die Abschätzung

$$\begin{aligned}|\langle\mathbf{y},\mathbf{X}\mathbf{z}\rangle_2| &= \left|\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}\overline{y}_i X_{ij} z_j\right| \leq \left(\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}|X_{ij}|^2\right)^{1/2}\left(\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}|y_i|^2\,|z_j|^2\right)^{1/2}\\ &= \|\mathbf{X}\|_F\|\mathbf{y}\|_2\|\mathbf{z}\|_2\end{aligned}$$

zu erhalten. Aus Lemma 5.7 folgt die erste Abschätzung $\|\mathbf{X}\|_2 \leq \|\mathbf{X}\|_F$.

Zum Nachweis der zweiten Abschätzung definieren wir die kanonischen Einheitsvektoren $\mathbf{e}^j \in \mathbb{K}^{\mathcal{J}}$ durch

$$e_k^j := \begin{cases} 1 & \text{falls } j = k, \\ 0 & \text{ansonsten} \end{cases} \qquad \text{für alle } j, k \in \mathcal{J}$$

und stellen fest, dass

$$\|\mathbf{X}\mathbf{e}^j\|_2^2 = \sum_{i \in \mathcal{I}} |(\mathbf{X}\mathbf{e}^j)_i|^2 = \sum_{i \in \mathcal{I}} \left| \sum_{k \in \mathcal{J}} X_{ik} e_k^j \right|^2 = \sum_{i \in \mathcal{I}} |X_{ij}|^2$$

und damit auch

$$\|\mathbf{X}\|_F^2 = \sum_{j \in \mathcal{J}} \sum_{i \in \mathcal{I}} |X_{ij}|^2 = \sum_{j \in \mathcal{J}} \|\mathbf{X}\mathbf{e}^j\|_2^2 \leq \sum_{j \in \mathcal{J}} \|\mathbf{X}\|_2^2 \|\mathbf{e}^j\|_2^2 = (\#\mathcal{J}) \|\mathbf{X}\|_2^2$$

gilt. Damit ist die gewünschte Normäquivalenz bewiesen. ∎

Indem wir Lemma 5.8 mit Lemma 5.2 kombinieren, können wir eine Fehlerabschätzung in der Spektralnorm konstruieren, falls uns blockweise Abschätzungen in der Frobeniusnorm zur Verfügung stehen. Handlicher wäre es natürlich, wenn wir ausschließlich mit Abschätzungen der Spektralnorm arbeiten könnten. Als ersten Versuch beweisen wir die folgende Variante des Lemmas 5.8:

Lemma 5.9 (Globale Spektralnorm) *Sei $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein Blockbaum, und seien $\mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ seine Blätter. Dann gilt*

$$\|\mathbf{X}\|_2 \leq \left(\sum_{b=(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}} \|\mathbf{X}|_{\hat{t}\times\hat{s}}\|_2^2 \right)^{1/2} \qquad \textit{für alle } \mathbf{X} \in \mathbb{K}^{\mathcal{I}\times\mathcal{J}}.$$

Beweis. Wir verwenden wieder Lemma 5.7: Seien $\mathbf{y} \in \mathbb{K}^{\mathcal{I}}$ und $\mathbf{z} \in \mathbb{K}^{\mathcal{J}}$ gegeben. Dank Folgerung 2.22 gilt

$$\begin{aligned} |\langle \mathbf{y}, \mathbf{X}\mathbf{z} \rangle_2| &= \left| \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} \bar{y}_i X_{ij} z_j \right| = \left| \sum_{b=(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}} \sum_{i \in \hat{t}} \sum_{j \in \mathcal{J}} \bar{y}_i X_{ij} z_i \right| \\ &= \left| \sum_{b=(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}} \langle \mathbf{y}|_{\hat{t}}, \mathbf{X}|_{\hat{t}\times\hat{s}} \mathbf{z}|_{\hat{s}} \rangle_2 \right| \leq \sum_{b=(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}} \|\mathbf{y}|_{\hat{t}}\|_2 \|\mathbf{X}|_{\hat{t}\times\hat{s}}\|_2 \|\mathbf{z}|_{\hat{s}}\|_2. \end{aligned}$$

Auf diese Summe wenden wir wieder die Cauchy-Schwarz-Ungleichung an, um

$$|\langle \mathbf{y}, \mathbf{X}\mathbf{z} \rangle_2| \leq \left(\sum_{b=(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}} \|\mathbf{X}|_{\hat{t}\times\hat{s}}\|_2^2 \right)^{1/2} \left(\sum_{b=(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}} \|\mathbf{y}|_{\hat{t}}\|_2^2 \|\mathbf{z}|_{\hat{s}}\|_2^2 \right)^{1/2}$$

zu erhalten. Der erste Faktor ist bereits der gewünschte, zur Abschätzung des zweiten verwenden wir erneut Folgerung 2.22, die

$$\sum_{b=(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}} \|\mathbf{y}|_{\hat{t}}\|_2^2 \|\mathbf{z}|_{\hat{s}}\|_2^2 = \sum_{b=(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}} \sum_{i\in\hat{t}} \sum_{j\in\hat{s}} |y_i|^2 |z_j|^2 = \sum_{i\in\mathcal{I}} \sum_{j\in\mathcal{J}} |y_i|^2 |z_j|^2 = \|\mathbf{y}\|_2^2 \|\mathbf{z}\|_2^2$$

impliziert. Insgesamt erhalten wir

$$|\langle \mathbf{y}, \mathbf{X}\mathbf{z}\rangle_2| \leq \left(\sum_{b=(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}} \|\mathbf{X}|_{\hat{t}\times\hat{s}}\|_2^2 \right)^{1/2} \|\mathbf{y}\|_2 \|\mathbf{z}\|_2,$$

und aus Lemma 5.7 folgt die gesuchte Abschätzung. ∎

Diese Abschätzung der Spektralnorm ist zwar relativ handlich, aber auch häufig sehr pessimistisch: Würden wir beispielsweise die Einheitsmatrix aus $\mathbb{K}^{n\times n}$ als Blockmatrix mit $n \times n$ einelementigen Teilmatrizen behandeln und Lemma 5.9 anwenden, erhielten wir eine Abschätzung der Form $1 = \|\mathbf{I}\|_2 \leq \sqrt{n}$, die offenbar den korrekten Wert deutlich überschätzt.

Ein modifizierter Ansatz zur Abschätzung der Spektralnorm findet sich in der grundlegenden Arbeit [19]: Falls der Blockbaum schwachbesetzt ist (vgl. Definition 2.24), können wir die Summe über alle Blätter durch ein geeignetes Maximum ersetzen und damit häufig wesentlich bessere obere Schranken finden.

Satz 5.10 (Globale Spektralnorm) *Sei $\mathbf{X} \in \mathbb{K}^{\mathcal{I}\times\mathcal{J}}$. Sei $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein C_{sp}-schwachbesetzter Blockbaum mit Blattmenge $\mathcal{L}_{\mathcal{I}\times\mathcal{J}}$. Sei p seine Tiefe, und sei $(\epsilon_\ell)_{\ell=0}^p$ eine Familie in $\mathbb{R}_{\geq 0}$, für die*

$$\|\mathbf{X}|_{\hat{t}\times\hat{s}}\|_2 \leq \epsilon_{\mathrm{level}(t)}^{1/2} \epsilon_{\mathrm{level}(s)}^{1/2} \qquad \text{für alle } b = (t,s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$$

gilt. Dann folgt

$$\|\mathbf{X}\|_2 \leq C_{sp} \sum_{\ell=0}^{p} \epsilon_\ell.$$

Beweis. Aus Lemma 2.26 folgt, dass für jeden Block $b = (t,s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ die Beziehungen

$$\mathrm{level}(t) \leq \mathrm{level}(b), \qquad \mathrm{level}(s) \leq \mathrm{level}(b)$$

gelten, also sind Voraussetzung und Abschätzung zumindest wohldefiniert.

Für den Beweis verwenden wir erneut Lemma 5.7: Wir fixieren $\mathbf{y} \in \mathbb{K}^{\mathcal{I}}$ und $\mathbf{z} \in \mathbb{K}^{\mathcal{J}}$ und erhalten mit Hilfe von Folgerung 2.22, der Cauchy-Schwarz-Ungleichung und der Submultiplikativität der Spektralnorm ähnlich wie im vorigen Lemma die Abschätzung

$$|\langle \mathbf{y}, \mathbf{X}\mathbf{z}\rangle_2| \leq \sum_{b=(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}} \|\mathbf{X}|_{\hat{t}\times\hat{s}}\|_2 \|\mathbf{y}|_{\hat{t}}\|_2 \|\mathbf{z}|_{\hat{s}}\|_2.$$

An dieser Stelle können wir nun unsere Voraussetzung anwenden, um

$$|\langle \mathbf{y}, \mathbf{Xz} \rangle_2| \leq \sum_{b=(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}} \epsilon_{\operatorname{level}(t)}^{1/2} \|\mathbf{y}|_{\hat{t}}\|_2 \epsilon_{\operatorname{level}(s)}^{1/2} \|\mathbf{z}|_{\hat{s}}\|_2$$

zu erhalten. Eine Anwendung der Cauchy-Schwarz-Ungleichung auf diese Summe ergibt

$$|\langle \mathbf{y}, \mathbf{Xz} \rangle_2| \leq \left(\sum_{b=(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}} \epsilon_{\operatorname{level}(t)} \|\mathbf{y}|_{\hat{t}}\|_2^2 \right)^{1/2} \left(\sum_{b=(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}} \epsilon_{\operatorname{level}(s)} \|\mathbf{z}|_{\hat{s}}\|_2^2 \right)^{1/2}, \tag{5.7}$$

und da beide Faktoren ähnlich sind, beschränken wir uns darauf, den ersten zu analysieren. Da $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ C_{sp}-schwachbesetzt ist, finden wir

$$\sum_{b=(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}} \epsilon_{\operatorname{level}(t)} \|\mathbf{y}|_{\hat{t}}\|_2^2 = \sum_{t\in\mathcal{T}_{\mathcal{I}}} \sum_{s\in\operatorname{row}(t)} \epsilon_{\operatorname{level}(t)} \|\mathbf{y}|_{\hat{t}}\|_2^2 \leq C_{\mathrm{sp}} \sum_{t\in\mathcal{T}_{\mathcal{I}}} \epsilon_{\operatorname{level}(t)} \|\mathbf{y}|_{\hat{t}}\|_2^2.$$

Wir zerlegen den Clusterbaum $\mathcal{T}_{\mathcal{I}}$ in seine einzelnen Stufen und erhalten dank Lemma 2.19 die Abschätzung

$$\begin{aligned}
\sum_{b=(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}} \epsilon_{\operatorname{level}(t)} \|\mathbf{y}|_{\hat{t}}\|_2^2 &\leq C_{\mathrm{sp}} \sum_{t\in\mathcal{T}_{\mathcal{I}}} \epsilon_{\operatorname{level}(t)} \|\mathbf{y}|_{\hat{t}}\|_2^2 = C_{\mathrm{sp}} \sum_{\ell=0}^{p_{\mathcal{I}}} \epsilon_\ell \sum_{\substack{t\in\mathcal{T}_{\mathcal{I}} \\ \operatorname{level}(t)=\ell}} \|\mathbf{y}|_{\hat{t}}\|_2^2 \\
&= C_{\mathrm{sp}} \sum_{\ell=0}^{p_{\mathcal{I}}} \epsilon_\ell \sum_{\substack{t\in\mathcal{T}_{\mathcal{I}} \\ \operatorname{level}(t)=\ell}} \sum_{i\in\hat{t}} |y_i|^2 \leq C_{\mathrm{sp}} \sum_{\ell=0}^{p_{\mathcal{I}}} \epsilon_\ell \sum_{i\in\mathcal{I}} |y_i|^2 \\
&= C_{\mathrm{sp}} \|\mathbf{y}\|_2^2 \sum_{\ell=0}^{p_{\mathcal{I}}} \epsilon_\ell \leq C_{\mathrm{sp}} \|\mathbf{y}\|_2^2 \sum_{\ell=0}^{p} \epsilon_\ell,
\end{aligned}$$

wobei wir mit $p_{\mathcal{I}} \leq p$ wieder die Tiefe des Clusterbaums $\mathcal{T}_{\mathcal{I}}$ bezeichnen. Mit der zweiten Summe in (5.7) können wir ähnlich verfahren und erhalten

$$|\langle \mathbf{y}, \mathbf{Xz} \rangle_2| \leq C_{\mathrm{sp}} \|\mathbf{y}\|_2 \|\mathbf{z}\|_2 \sum_{\ell=0}^{p} \epsilon_\ell,$$

so dass mit Lemma 5.7 die gewünschte Aussage folgt. ∎

Falls wir beispielsweise den Fehler in jedem Block durch $\epsilon \in \mathbb{R}_{>0}$ beschränken können, folgt aus dieser Abschätzung bereits die Schranke $C_{\mathrm{sp}}(p+1)\epsilon$ für den Gesamtfehler.

Um diese Normabschätzung mit der aus Lemma 5.9 vergleichen zu können, bietet es sich an, eine etwas eingeschränkte Variante zu untersuchen:

Folgerung 5.11 (Globale Spektralnorm) *Sei der Blockbaum C_{sp}-schwachbesetzt sowie* stufentreu*, es gelte also*

$$\operatorname{level}(b) = \operatorname{level}(t) = \operatorname{level}(s) \qquad \textit{für alle } b = (t,s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}.$$

Sei p die Tiefe dieses Baums. Dann folgt

$$\|\mathbf{X}\|_2 \leq C_{sp} \sum_{\ell=0}^{p} \max\{\|\mathbf{X}|_{\hat{t}\times\hat{s}}\|_2 \ : \ b=(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}},\ \operatorname{level}(b)=\ell\} \quad \textit{für alle } \mathbf{X}\in\mathbb{K}^{\mathcal{I}\times\mathcal{J}}.$$

Beweis. Wir definieren

$$\epsilon_\ell := \max\{\|\mathbf{X}|_{\hat{t}\times\hat{s}} \ : \ b=(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}},\ \operatorname{level}(b)=\ell\} \qquad \text{für alle } \ell\in\{0,\ldots,p\}$$

und wenden Satz 5.10 an. ∎

5.3 Anwendung auf Integraloperatoren

Die bisher diskutierten Abschätzungen verhalten sich ungünstig, wenn die Anzahl der Unbekannten wächst: Etwa in Lemma 5.3 gehen $\#\mathcal{I}$ und $\#\mathcal{J}$ in die Abschätzung ein und führen dazu, dass die Norm um so größer wird, je mehr Unbekannte vorliegen, obwohl die zugrundeliegende Approximation der Kernfunktion dieselbe bleibt.

Für eine wichtige Klasse von Problemen lässt sich dieser Effekt ausgleichen: Bei Matrizen, die aus der Diskretisierung eines Integraloperators entstehen, treten zusätzliche Faktoren auf, die bei einer Erhöhung der Anzahl der Unbekannten die Gewichtung des Fehlers günstig beeinflussen und so zu besseren Abschätzungen als den bisher diskutierten führen.

Als Modellbeispiel untersuchen wir die Galerkin-Diskretisierung eines Integraloperators (vgl. etwa [24, 33, 34]) der Form

$$\mathfrak{G}[u](x) := \int_\Omega g(x,y)u(y)\,dy,$$

wobei Ω ein Teilgebiet oder eine Teilmannigfaltigkeit des d-dimensionalen Raums $\mathbb{R}^d$ ist, g eine Kernfunktion und u eine Funktion aus einem geeigneten Raum $\mathcal{V}\subseteq L^2(\Omega)$, für die das Integral in einem geeigneten Sinne existiert (Details finden sich in den erwähnten Büchern).

Für die Galerkin-Diskretisierung wählen wir eine Familie $(\varphi_i)_{i\in\mathcal{I}}$ von linear unabhängigen Funktionen aus $\mathcal{V}$, also eine Basis eines Unterraums von $\mathcal{V}$. Das diskrete Gegenstück dieses Operators ist die Matrix $\mathbf{G}\in\mathbb{K}^{\mathcal{I}\times\mathcal{I}}$, die durch

$$G_{ij} := \int_\Omega \bar{\varphi}_i(x) \int_\Omega g(x,y)\varphi_j(y)\,dy\,dx \qquad \text{für alle } i,j\in\mathcal{I} \tag{5.8}$$

definiert ist. In der Praxis wählt man die Basis $(\varphi_i)_{i\in\mathcal{I}}$ häufig so, dass die *Träger* der Basisfunktionen, also die durch

$$\operatorname{supp}\varphi_i := \overline{\{x\in\Omega \ : \ \varphi_i(x)\neq 0\}} \qquad \text{für alle } i\in\mathcal{I}$$

definierten Teilmengen von Ω, relativ klein sind. Wir kürzen sie mit

$$\Omega_i := \operatorname{supp}\varphi_i \qquad \text{für alle } i\in\mathcal{I}$$

ab. Da der Integrand außerhalb dieser Gebiete verschwindet, erhalten wir die alternative Darstellung

$$G_{ij} = \int_{\Omega_i} \bar{\varphi}_i(x) \int_{\Omega_j} g(x,y)\varphi_j(y)\,dy\,dx,$$

die uns an die bisher untersuchten Punktauswertungen erinnert: Statt die Kernfunktion in Punkten auszuwerten bilden wir bei der Galerkin-Diskretisierung Mittelwerte über kleine Teilgebiete.

Die bisher verwendeten Approximationstechniken übertragen sich direkt auf diesen Fall: Wir wählen für jedes $i \in \mathcal{I}$ einen „charakteristischen Punkt“ $\mathbf{x}_i \in \Omega_i$ und konstruieren den Clusterbaum $\mathcal{T}_\mathcal{I}$ so wie bisher (vgl. Abbildungen 2.2 und 2.3). Wir führen die Mengen

$$\Omega_t := \bigcup_{i \in \hat{t}} \Omega_i \qquad \text{für alle } t \in \mathcal{T}_\mathcal{I}$$

ein und konstruieren die überdeckenden Quader $(Q_t)_{t\in\mathcal{T}_\mathcal{I}}$ so, dass sie

$$\Omega_t \subseteq Q_t \qquad \text{für alle } t \in \mathcal{T}_\mathcal{I}$$

erfüllen. Mit Hilfe der üblichen Zulässigkeitsbedingungen, etwa (3.12) oder (4.3), können wir basierend auf diesen Quadern einen zulässigen Blockbaum $\mathcal{T}_{\mathcal{I}\times\mathcal{I}}$ konstruieren.

Falls nun $b = (t,s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{I}}$ ein zulässiges Blatt dieses Baums ist, können wir wie bisher eine Approximation

$$\tilde{g}_b(x,y) = \sum_{\nu=1}^{k} a_{b,\nu}(x)\bar{b}_{b,\nu}(y) \qquad \text{für alle } x \in \Omega_t,\ y \in \Omega_s \tag{5.9}$$

der Kernfunktion konstruieren und erhalten für $i \in \hat{t}$ und $j \in \hat{s}$ die Näherung

$$\begin{aligned} G_{ij} &= \int_{\Omega_i} \bar{\varphi}_i(x) \int_{\Omega_j} g(x,y)\varphi_j(y)\,dy\,dx \\ &\approx \int_{\Omega_i} \bar{\varphi}_i(x) \int_{\Omega_j} \tilde{g}_b(x,y)\varphi_j(y)\,dy\,dx \\ &= \sum_{\nu=1}^{k} \int_{\Omega_i} a_{b,\nu}(x)\bar{\varphi}_i(x)\,dx \int_{\Omega_j} \bar{b}_{b,\nu}(y)\varphi_j(y)\,dy. \end{aligned}$$

Indem wir $\mathbf{A}_b \in \mathbb{K}^{\hat{t}\times k}$ und $\mathbf{B}_b \in \mathbb{K}^{\hat{s}\times k}$ durch

$$\begin{aligned} A_{b,i\nu} &:= \int_{\Omega_i} a_{b,\nu}(x)\bar{\varphi}_i(x)\,dx && \text{für alle } i \in \hat{t},\ \nu \in \{1,\ldots,k\}, \\ B_{b,j\nu} &:= \int_{\Omega_j} b_{b,\nu}(y)\bar{\varphi}_j(y)\,dy && \text{für alle } j \in \hat{s},\ \nu \in \{1,\ldots,k\} \end{aligned}$$

definieren erhalten wir daraus die Rang-k-Approximation

$$\mathbf{G}|_{\hat{t}\times\hat{s}} = \mathbf{A}_b\mathbf{B}_b^*.$$

Die Konstruktion ist also der sehr ähnlich, die wir bisher für Punktauswertungen kennengelernt haben. Aus diesen Teilmatrizen können wir in der üblichen Weise eine $\mathcal{H}$-Matrix-Approximation $\widetilde{\mathbf{G}} \in \mathbb{K}^{\mathcal{I}\times\mathcal{I}}$ konstruieren, indem wir

$$\widetilde{\mathbf{G}}|_{\hat{t}\times\hat{s}} = \begin{cases} \mathbf{A}_b\mathbf{B}_b^* & \text{falls } b \text{ zulässig,} \\ \mathbf{G}|_{\hat{t}\times\hat{s}} & \text{ansonsten} \end{cases} \qquad \text{für alle } b = (t,s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{I}}$$

setzen und Folgerung 2.22 verwenden.

Bei der Fehleranalyse ergeben sich allerdings subtile Unterschiede: Wir führen für jedes $t \in \mathcal{T}_\mathcal{I}$ den *Koeffizientenisomorphismus*

$$\Psi_t : \mathbb{K}^{\hat{t}} \to \mathcal{V}, \qquad \mathbf{u} \mapsto \sum_{i\in\hat{t}} u_i\varphi_i,$$

ein und bezeichnen mit Ψ den zu der Wurzel $r \in \mathcal{T}_\mathcal{I}$ gehörenden Isomorphismus Ψ_r. Für beliebige $t, s \in \mathcal{T}_\mathcal{I}$ und $\mathbf{v} \in \mathbb{K}^{\hat{t}}$ sowie $\mathbf{u} \in \mathbb{K}^{\hat{s}}$ gilt die Gleichung

$$\begin{aligned} \langle \mathbf{v}, \mathbf{G}\mathbf{u}\rangle_2 &= \sum_{i\in\hat{t}}\sum_{j\in\hat{s}} \bar{v}_i G_{ij} u_j \\ &= \sum_{i\in\hat{t}}\sum_{j\in\hat{s}} \int_\Omega \bar{v}_i\bar{\varphi}_i(x) \int_\Omega g(x,y) u_j\varphi_j(y)\,dy\,dx \\ &= \int_\Omega \left(\sum_{i\in\hat{t}} \overline{v_i\varphi_i(x)}\right) \int_\Omega g(x,y) \left(\sum_{j\in\hat{s}} u_j\varphi_j(y)\right) dy\,dx \\ &= \int_\Omega \overline{\Psi_t[v](x)} \int_\Omega g(x,y)\Psi_s[u](y)\,dy\,dx, \end{aligned} \tag{5.10}$$

also können wir Skalarprodukte mit der Matrix $\mathbf{G}$ auf Doppelintegrale zurückführen.

Um aus dieser Eigenschaft Fehlerabschätzungen gewinnen zu können, benötigen wir eine Stabilitätskonstante für die Operatoren Ψ_t.

Lemma 5.12 (Massematrix) *Wir definieren durch*

$$M_{ij} := \int_\Omega \bar{\varphi}_i(x)\varphi_j(x)\,dx \qquad \textit{für alle } i,j \in \mathcal{I}$$

die Massematrix $\mathbf{M} \in \mathbb{K}^{\mathcal{I}\times\mathcal{I}}$*, und bezeichnen mit* $C_{fe} := \|\mathbf{M}\|_2$ *ihre Spektralnorm. Dann gilt*

$$\|\Psi_t[u]\|_{L^2} \leq C_{fe}^{1/2}\|u\|_2 \qquad \textit{für alle } t \in \mathcal{T}_\mathcal{I},\ u \in \mathbb{K}^{\hat{t}}.$$

Beweis. Sei $t \in \mathcal{T}_\mathcal{I}$, und sei $u \in \mathbb{K}^{\hat{t}}$. Dann gilt

$$\|\Psi_t[u]\|_{L^2}^2 = \int_\Omega \overline{\Psi_t[u](x)}\Psi_t[u](x)\,dx = \sum_{i\in\hat{t}}\sum_{j\in\hat{t}} \int_\Omega \bar{u}_i\bar{\varphi}_i(x)u_j\varphi_j(x)\,dx$$

$$= \sum_{i\in\hat{t}} \sum_{j\in\hat{t}} \bar{u}_i M_{ij} u_j = \langle \mathbf{u}, \mathbf{M}|_{\hat{t}\times\hat{t}}\mathbf{u}\rangle_2 \leq \|\mathbf{u}\|_2 \|\mathbf{M}|_{\hat{t}\times\hat{t}}\mathbf{u}\|_2.$$

Wir definieren $\widehat{\mathbf{u}} \in \mathbb{K}^{\mathcal{I}}$ durch $\widehat{\mathbf{u}}|_{\hat{t}} = \mathbf{u}$ und $\widehat{\mathbf{u}}|_{\mathcal{I}\setminus\hat{t}} = \mathbf{0}$ und erhalten

$$\|\mathbf{M}|_{\hat{t}\times\hat{t}}\mathbf{u}\|_2 = \|\mathbf{M}|_{\hat{t}\times\mathcal{I}}\widehat{\mathbf{u}}\|_2 \leq \|\mathbf{M}\widehat{\mathbf{u}}\|_2 \leq C_{\mathrm{fe}}\|\widehat{\mathbf{u}}\|_2 = C_{\mathrm{fe}}\|\mathbf{u}\|_2,$$

und daraus folgt $\|\Psi_t[u]\|_{L^2}^2 \leq C_{\mathrm{fe}}\|u\|_2^2$, also die gewünschte Abschätzung. ∎

Indem wir dieses Resultat mit (5.10) kombinieren können wir eine obere obere Schranke für den blockweisen Fehler in der Spektralnorm erhalten, die statt $\#\hat{t}$ und $\#\hat{s}$ die *Maße*

$$|\Omega_t| := \int_{\Omega_t} 1\,dx, \qquad\qquad |\Omega_s| := \int_{\Omega_s} 1\,dy$$

enthält, und diese Größen sind unabhängig von der Anzahl der Unbekannten.

Lemma 5.13 (Blockweiser Fehler) *Sei $b = (t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ ein zulässiges Blatt des Blockbaums $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$. Sei $\epsilon_b \in \mathbb{R}_{\geq 0}$ mit*

$$|g(\mathbf{x},\mathbf{y}) - \tilde{g}_b(\mathbf{x},\mathbf{y})| \leq \epsilon_b \qquad\qquad \text{für alle } \mathbf{x} \in Q_t,\ \mathbf{y} \in Q_s$$

gegeben. Dann gilt

$$\|\mathbf{G}|_{\hat{t}\times\hat{s}} - \mathbf{A}_b\mathbf{B}_b^*\|_2 \leq C_{fe}|\Omega_t|^{1/2}|\Omega_s|^{1/2}\epsilon_b.$$

Beweis. Wir verwenden wieder Lemma 5.7. Seien $\mathbf{v} \in \mathbb{K}^{\hat{t}}$ und $\mathbf{u} \in \mathbb{K}^{\hat{s}}$ fixiert. Indem wir (5.10) auf $\mathbf{G}|_{\hat{t}\times\hat{s}}$ und $\widetilde{\mathbf{G}}|_{\hat{t}\times\hat{s}} = \mathbf{A}_b\mathbf{B}_b^*$ anwenden, erhalten wir

$$\begin{aligned}
|\langle \mathbf{v}, (\mathbf{G} - \widetilde{\mathbf{G}})|_{\hat{t}\times\hat{s}}\mathbf{u}\rangle_2| &= |\langle \mathbf{v}, \mathbf{G}|_{\hat{t}\times\hat{s}}\mathbf{u}\rangle_2 - \langle \mathbf{v}, \mathbf{A}_b\mathbf{B}_b^*\mathbf{u}\rangle_2| \\
&= \left| \int_{\Omega_t} \overline{\Psi_t[v](x)} \int_{\Omega_s} g(x,y)\Psi_s[u](y)\,dy\,dx \right. \\
&\quad \left. - \int_{\Omega_t} \overline{\Psi_t[v](x)} \int_{\Omega_s} \tilde{g}_b(x,y)\Psi_s[u](y)\,dy\,dx \right| \\
&= \left| \int_{\Omega_t} \overline{\Psi_t[v](x)} \int_{\Omega_s} (g(x,y) - \tilde{g}_b(x,y))\Psi_s[u](y)\,dy\,dx \right| \\
&\leq \epsilon_b \int_{\Omega_t} |\Psi_t[v](x)|\,dx \int_{\Omega_s} |\Psi_s[u](y)|\,dy.
\end{aligned}$$

Hier treten nun die L^1-Normen der Funktionen $\Psi_t[v]$ und $\Psi_s[u]$ auf, die wir mit Hilfe der Cauchy-Schwarz-Ungleichung (diesmal für Integrale) auf die L^2-Normen zurückführen können, die sich mit Lemma 5.12 abschätzen lassen:

$$\int_{\Omega_t} |\Psi_t[v](x)|\,dx = \int_{\Omega_t} 1|\Psi_t[v](x)|\,dx \leq \left(\int_{\Omega_t} 1\,dx\right)^{1/2} \left(\int_{\Omega_t} |\Psi_t[v](x)|^2\,dx\right)^{1/2}$$

$$= |\Omega_t|^{1/2} \|\Psi_t[v]\|_{L^2} \leq C_{\mathrm{fe}}^{1/2} |\Omega_t|^{1/2} \|v\|_2.$$

Insgesamt erhalten wir so

$$|\langle \mathbf{v}, (\mathbf{G} - \widetilde{\mathbf{G}})|_{\hat{t}\times\hat{s}} \mathbf{u} \rangle_2| \leq C_{\mathrm{fe}} |\Omega_t|^{1/2} |\Omega_s|^{1/2} \epsilon_b \|\mathbf{v}\|_2 \|\mathbf{u}\|_2,$$

und mit Lemma 5.7 folgt die gewünschte Ungleichung. ∎

Wenn wir diese blockweisen Fehlerabschätzungen mit Hilfe des Lemmas 5.9 zu Abschätzungen der gesamten Matrix zusammensetzen wollen, müssen wir untersuchen, wie sich die Maße $|\Omega_t|$ und $|\Omega_s|$ aufaddieren. Entscheidend dafür ist, wie oft einzelne Punkte in den Trägern von Basisfunktionen auftreten.

Definition 5.14 (Überlappende Träger) *Sei $C_{ov} \in \mathbb{N}$. Die Familie $(\varphi_i)_{i\in\mathcal{I}}$ der Basisfunktionen nennen wir C_{ov}-*überlappend*, falls*

$$\#\{i \in \mathcal{I} \ : \ x \in \operatorname{supp} \varphi_i\} \leq C_{ov} \qquad \text{für alle } x \in \Omega$$

gilt, falls also jeder Punkt $x \in \Omega$ in den Trägern von höchstens C_{ov} Basisfunktionen vorkommt.

Mit Hilfe der Konstanten C_{ov} können wir die gesuchte Abschätzung beweisen:

Lemma 5.15 (Überlappende Träger) *Sei $(\varphi_i)_{i\in\mathcal{I}}$ C_{ov}-überlappend. Dann gelten*

$$\sum_{i\in\mathcal{I}} |\Omega_i| \leq C_{ov} |\Omega|, \qquad \sum_{b=(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}} |\Omega_t||\Omega_s| \leq C_{ov}^2 |\Omega|^2.$$

Beweis. Wir definieren die Abbildungen

$$\chi_i : \Omega \to \mathbb{R}, \qquad \mathbf{x} \mapsto \begin{cases} 1 & \text{falls } \mathbf{x} \in \Omega_i, \\ 0 & \text{ansonsten} \end{cases}$$

und stellen fest, dass

$$\sum_{i\in\mathcal{I}} \chi_i(\mathbf{x}) = \#\{i \in \mathcal{I} \ : \ \mathbf{x} \in \Omega_i\} \leq C_{\mathrm{ov}} \qquad \text{für alle } \mathbf{x} \in \Omega$$

gilt, so dass wir

$$\sum_{i\in\mathcal{I}} |\Omega_i| = \sum_{i\in\mathcal{I}} \int_{\Omega_i} 1 \, d\mathbf{x} = \sum_{i\in\mathcal{I}} \int_{\Omega} \chi_i(\mathbf{x}) \, d\mathbf{x} = \int_{\Omega} \sum_{i\in\mathcal{I}} \chi_i(\mathbf{x}) \, d\mathbf{x} \leq \int_{\Omega} C_{\mathrm{ov}} \, d\mathbf{x} = C_{\mathrm{ov}} |\Omega|$$

erhalten. Aus

$$|\Omega_t| = \left| \bigcup_{i\in\hat{t}} \Omega_i \right| \leq \sum_{i\in\hat{t}} |\Omega_i|$$

und Folgerung 2.22 ergibt sich die zweite Abschätzung. ∎

Aus der Kombination der Lemmas 5.13, 5.9 und 5.15 ergibt sich eine Fehlerabschätzung, die nicht mehr von der Anzahl der Unbekannten abhängt:

Lemma 5.16 (Globale Spektralnorm) *Sei $(\varphi_i)_{i\in\mathcal{I}}$ C_{ov}-überlappend und gelte*

$$|g(\mathbf{x},\mathbf{y}) - \tilde{g}_b(\mathbf{x},\mathbf{y})| \leq \epsilon \qquad \text{für alle } b=(t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}},\ \mathbf{x}\in Q_t,\ \mathbf{y}\in Q_s.$$

Dann folgt

$$\|\mathbf{G} - \widetilde{\mathbf{G}}\|_2 \leq C_{fe}C_{ov}|\Omega|\epsilon.$$

Beweis. Für jedes zulässige Blatt $b=(t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ erhalten wir mit Lemma 5.13 die Abschätzung

$$\|\mathbf{G}|_{\hat{t}\times\hat{s}} - \widetilde{\mathbf{G}}|_{\hat{t}\times\hat{s}}\|_2 \leq C_{\mathrm{fe}}|\Omega_t|^{1/2}|\Omega_s|^{1/2}\epsilon.$$

Für unzulässige Blätter gilt diese Abschätzung ohnehin, da bei ihnen keine Approximation stattfindet. Also können wir Lemma 5.9 anwenden, um

$$\|\mathbf{G} - \widetilde{\mathbf{G}}\|_2 \leq \left(\sum_{b=(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}} C_{\mathrm{fe}}^2|\Omega_t|\ |\Omega_s|\ \epsilon^2\right)^{1/2} = C_{\mathrm{fe}}\epsilon\left(\sum_{b=(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}} |\Omega_t|\ |\Omega_s|\right)^{1/2}$$

zu erhalten. Dank Lemma 5.15 lässt sich die Summe durch $C_{\mathrm{ov}}^2|\Omega|^2$ abschätzen, um das gewünschte Resultat zu erreichen. ∎

Auf den ersten Blick ist diese Fehlerabschätzung sehr befriedigend: Außer ϵ treten nur Konstanten auf, und ϵ lässt sich durch Wahl einer hinreichend guten Approximation der Kernfunktion beliebig reduzieren.

Auf den zweiten Blick entdeckt man eine versteckte Abhängigkeit: Wenn wir g mit Hilfe einer Tensor-Interpolation approximieren, erhalten wir die Fehlerabschätzung (3.10) der Form

$$\|g-\tilde{g}_b\|_{\infty,Q_t\times Q_s} \leq \frac{C(m)}{\operatorname{dist}(Q_t,Q_s)^\sigma}\left(\frac{c_0\eta}{4}\right)^{m+1}.$$

Aus der Zulässigkeitsbedingung (4.3) folgt

$$\operatorname{diam}(Q_t)\operatorname{diam}(Q_s) \leq \eta^2\operatorname{dist}^2(Q_t,Q_s), \qquad \frac{\sqrt{\operatorname{diam}(Q_t)\operatorname{diam}(Q_s)}}{\eta} \leq \operatorname{dist}(Q_t,Q_s)$$

und somit

$$\|g-\tilde{g}_b\|_{\infty,Q_t\times Q_s} \leq \epsilon_b := \frac{C(m)\eta^\sigma}{\operatorname{diam}(Q_t)^{\sigma/2}\operatorname{diam}(Q_s)^{\sigma/2}}\left(\frac{c_0\eta}{4}\right)^{m+1}.$$

Die Interpolation wird auf kleinen Blöcken also einen größeren Fehler als auf großen aufweisen.

Glücklicherweise haben wir in Lemma 5.13 Faktoren erhalten, die diesem unerwünschten Effekt entgegenwirken: Es gilt

$$\begin{aligned}\|\mathbf{G}|_{\hat{t}\times\hat{s}} - \mathbf{A}_b\mathbf{B}_b^*\|_2 &\leq C_{\mathrm{fe}}|\Omega_t|^{1/2}|\Omega_s|^{1/2}\epsilon_b\\ &\leq C_{\mathrm{fe}}C(m)\eta^\sigma\left(\frac{|\Omega_t|}{\operatorname{diam}(Q_t)^\sigma}\right)^{1/2}\left(\frac{|\Omega_s|}{\operatorname{diam}(Q_s)^\sigma}\right)^{1/2}\left(\frac{c_0\eta}{4}\right)^{m+1},\end{aligned}$$

wir dürfen also darauf hoffen, dass bei kleinen Clustern t das Maß $|\Omega_t|$ so klein wird, dass es den von der Singularität der Kernfunktion verursachten Faktor $\operatorname{diam}(Q_t)^{-\sigma}$ ausgleichen kann. Für geeignete Geometrien und nicht zu starke Singularitäten ist das tatsächlich der Fall, und dank Satz 5.10 können wir schließlich für typische Anwendungen eine Abschätzung der Form

$$\|\mathbf{G} - \widetilde{\mathbf{G}}\|_2 \leq \widetilde{C}(m) \left(\frac{c_0\eta}{4}\right)^{m+1}$$

oder wenigstens der Form

$$\|\mathbf{G} - \widetilde{\mathbf{G}}\|_2 \leq \widetilde{C}(m)(p+1) \left(\frac{c_0\eta}{4}\right)^{m+1}$$

mit einem nur von m abhängenden Polynom $\widetilde{C}$ und der Tiefe p des Blockbaums erhalten.

Damit ist das Ziel erreicht: Keine oder nur eine schwache Abhängigkeit von der Größe der Matrix, und exponentielle Konvergenz des Fehlers in Abhängigkeit von m, so dass sich theoretisch jede gewünschte Genauigkeit erreichen lässt. Praktisch sind wir natürlich durch die Menge des zur Verfügung stehenden Speichers beschränkt.

Für schwache Singularitätsordnungen lässt sich eine noch bessere Abschätzung konstruieren, wenn wir die Interpolationsordnung abhängig von der Größe des Clusters wählen. Bei diesen *Verfahren variabler Ordnung* [32, 12, 13] verwendet man eine hohe Ordnung auf großen Blöcken und eine niedrige auf kleinen. Dadurch lässt sich erreichen, dass der Fehler *kleiner* wird, wenn die Anzahl der Unbekannten wächst. Da es in der Regel nur wenige große Blöcke gibt, lässt sich der Rechenaufwand bei diesem Ansatz wesentlich besser als bei einer konstanten Ordnung beherrschen.

6 Matrix-Arithmetik

In vielen Anwendungen treten Matrizen auf, die nicht explizit beschrieben sind. Beispielsweise ist die Inverse $\mathbf{X}$ einer Matrix $\mathbf{G}$ durch die Gleichung $\mathbf{GX} = \mathbf{I}$ definiert, und uns stehen die einzelnen Koeffizienten X_{ij} nicht unmittelbar zur Verfügung.

Um derartige *implizit* beschriebenen Matrizen behandeln zu können, müssen wir deshalb auf andere als die bisher beschriebenen Techniken zurückgreifen. Glücklicherweise lassen sich wichtige Berechnungen wie

- die der Inversen $\mathbf{X} = \mathbf{G}^{-1}$ einer Matrix,
- die ihrer LR-Zerlegung $\mathbf{LR} = \mathbf{G}$,
- die der Exponentialfunktion $\mathbf{X} = \exp(\mathbf{G})$ oder
- das Lösen von Matrixgleichungen $\mathbf{AX} + \mathbf{XB} = \mathbf{C}$

auf zwei grundlegende Operationen zurückführen, nämlich auf die Addition und Multiplikation von Matrizen.

Unser Ziel ist es also, Summen und Produkte hierarchischer Matrizen effizient zu approximieren. Indem wir die spezielle Struktur dieser Matrizen ausnutzen, können wir in vielen Fällen sogar erreichen, dass die Anzahl der erforderlichen Rechenoperationen sich proportional zu $n \log^{\alpha} n$ verhält, wobei n die Matrixdimension und α ein (in der Regel kleiner) Exponent ist.

6.1 Auswertung

Eine der grundlegenden Operationen, die wir mit einer Matrix $\mathbf{X}$ durchführen können, ist die Multiplikation mit einem Vektor $\mathbf{y}$, also die Berechnung von $\mathbf{Xy}$. In der Praxis tritt diese Berechnung häufig in Form der Operation

$$\mathbf{z} \leftarrow \mathbf{z} + \alpha \mathbf{Xy} \tag{6.1}$$

auf, es wird also das Produkt mit einem zusätzlichen Faktor $\alpha \in \mathbb{K}$ zu einem gegebenen Vektor $\mathbf{z}$ hinzuaddiert, und der Wert dieses Vektors wird mit dem Ergebnis überschrieben. Derartige Operationen bilden die Grundlage vieler Anwendungen, beispielsweise lässt sich mit Hilfe der sogenannten *Krylow-Verfahren* das Lösen eines linearen Gleichungssystems auf eine Folge von Matrix-Vektor-Multiplikationen zurückführen (vgl. [23, 31])

Im Falle einer hierarchischen Matrix können wir wieder einmal auf Folgerung 2.22 zurückgreifen, um zu folgern, dass wir die Operation (6.1) auf verwandte Operationen

für die Blätter des Blockbaums zurückführen können: Wenn wir in einer beliebigen Reihenfolge die Operationen

$$\mathbf{z}|_{\hat{t}} \leftarrow \mathbf{z}|_{\hat{t}} + \alpha \mathbf{X}|_{\hat{t}\times\hat{s}} \mathbf{y}|_{\hat{s}} \qquad \text{für alle } b = (t,s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$$

durchführen, erhalten wir dasselbe Endergebnis.

Für zulässige Blätter $b = (t,s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ verwenden wir die Rang-k-Darstellung

$$\mathbf{X}|_{\hat{t}\times\hat{s}} = \mathbf{A}_b \mathbf{B}_b^*,$$

indem wir lediglich

$$\widehat{\mathbf{y}} \leftarrow \alpha \mathbf{B}_b^* \mathbf{y}|_{\hat{s}}, \qquad \mathbf{z}|_{\hat{t}} \leftarrow \mathbf{z}|_{\hat{t}} + \mathbf{A}_b \widehat{\mathbf{y}}$$

berechnen und so nicht mehr als $2k(\#\hat{t} + \#\hat{s})$ Rechenoperationen benötigen: $k(2(\#\hat{s}) - 1)$ Operationen für die Berechnung von $\mathbf{B}_b^* \mathbf{y}|_{\hat{s}}$, k Operationen für dessen Skalierung, $\#\hat{t}(2k-1)$ Operationen für die Berechnung von $\mathbf{A}_b \widehat{\mathbf{y}}$ und $\#\hat{t}$ für die Addition zu dem Ergebnisvektor.

```
procedure eval_h(α, b, X, y, var z);
(t, s) ← b;
if b ∈ L⁺_{I×J} then
  ŷ ← αB*_{X,b} y|_ŝ;   z|_t̂ ← z|_t̂ + A_{X,b} ŷ
else if b ∈ L⁻_{I×J} then
  z|_t̂ ← z|_t̂ + αX|_{t̂×ŝ} y|_ŝ
else
  for b' ∈ sons(b) do
    eval_h(α, b', X, y, z)
  end for
end if
```

Abbildung 6.1: Auswertung einer $\mathcal{H}$-Matrix: $\mathbf{z} \leftarrow \mathbf{z} + \alpha \mathbf{X}\mathbf{y}$

Da die Auswertung der Matrix auch als Bestandteil vieler weiterer Algorithmen auftritt, in denen sie häufig auf Teilmatrizen angewandt wird, organisieren wir sie als den in Abbildung 6.1 dargestellten rekursiven Algorithmus.

Unter bestimmten Bedingungen kann es auch nützlich sein, die Adjungierte einer hierarchischen Matrix auswerten zu können, ohne diese Adjungierte zuvor explizit als hierarchische Matrix konstruieren zu müssen. Wir interessieren uns also für die Operation

$$\mathbf{z} \leftarrow \mathbf{z} + \alpha \mathbf{X}^* \mathbf{y},$$

die sich in der bereits zuvor diskutierten Weise in die blockweisen Teiloperationen

$$\mathbf{z}|_{\hat{s}} \leftarrow \mathbf{z}|_{\hat{s}} + \alpha \mathbf{X}|^*_{\hat{t}\times\hat{s}} \mathbf{y}|_{\hat{t}} \qquad \text{für alle } b = (t,s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$$

```
procedure adjeval_h(α, b, X, y, var z);
(t, s) ← b;
if b ∈ L⁺_{I×J} then
  ŷ ← αA*_{X,b} y|_t̂;    z|_ŝ ← z|_ŝ + B_{X,b} ŷ
else if b ∈ L⁻_{I×J} then
  z|_ŝ ← z|_ŝ + αX|*_{t̂×ŝ} y|_t̂
else
  for b' ∈ sons(b) do
    adjeval_h(α, b', X, y, z)
  end for
end if
```

Abbildung 6.2: Auswertung der Adjungierten einer $\mathcal{H}$-Matrix: $\mathbf{z} \leftarrow \mathbf{z} + \alpha \mathbf{X}^* \mathbf{y}$

zerlegen lässt. In zulässigen Blättern $b = (t, s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ können wir die Rang-k-Darstellung

$$\mathbf{X}|^*_{\hat{t}\times\hat{s}} = (\mathbf{A}_b \mathbf{B}_b^*)^* = \mathbf{B}_b \mathbf{A}_b^*$$

verwenden, um die Auswertung durch Verwendung der effizienten Form

$$\widehat{\mathbf{y}} \leftarrow \alpha \mathbf{A}_b^* \mathbf{y}|_{\hat{t}}, \qquad\qquad \mathbf{z}|_{\hat{s}} \leftarrow \mathbf{z}|_{\hat{s}} + \mathbf{B}_b \widehat{\mathbf{y}}$$

zu beschleunigen. Dadurch fallen auch hier nicht mehr als $2k(\#\hat{t} + \#\hat{s})$ Rechenoperationen an.

Auch in diesem Fall ist es eine gute Idee, die einzelnen Blöcke des Blockbaums $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ rekursiv zu durchlaufen, um elegant auch mit Teilmatrizen der Matrix $\mathbf{X}$ arbeiten zu können. Der resultierende Algorithmus ist in Abbildung 6.2 zusammengefasst.

Natürlich sind wir daran interessiert, zu wissen, wie schnell die Auswertung einer hierarchischen Matrix oder ihrer Adjungierten durchgeführt werden kann:

Lemma 6.1 (Aufwand Auswertung) *Sei $\mathbf{X}$ in $\mathcal{H}$-Matrix-Darstellung mit lokalem Rang k, einem C_{sp}-schwachbesetzten zulässigen Blockbaum $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ der Tiefe p zu Clusterbäumen $\mathcal{T}_{\mathcal{I}}$ und $\mathcal{T}_{\mathcal{J}}$ mit Auflösungen $r_{\mathcal{I}}$ und $r_{\mathcal{J}}$ gegeben. Dann benötigen die in Abbildungen 6.1 und 6.2 dargestellten Algorithmen, aufgerufen mit $b = \operatorname{root}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}})$, nicht mehr als*

$$2C_{sp}(p+1)\max\{k, r_{\mathcal{I}}, r_{\mathcal{J}}\}(\#\mathcal{I} + \#\mathcal{J})$$

Rechenoperationen.

Beweis. Sei $b \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$. Falls b zulässig ist, fallen — wie bereits gesehen — nicht mehr als

$$2k(\#\hat{t} + \#\hat{s})$$

Rechenoperationen an.

Falls b unzulässig ist, sind es $(\#\hat{t})(2\#\hat{s} - 1)$ Rechenoperationen für die Berechnung von $\mathbf{X}|_{\hat{t}\times\hat{s}}\mathbf{y}|_{\hat{s}}$ und $\#\hat{t}$ Operationen für die Addition. Die Multiplikation mit α können

wir entweder im Ein- oder Ausgabevektor durchführen, so dass sie mit $\min\{\#\hat{t}, \#\hat{s}\}$ Operationen zu Buche schlägt, insgesamt kommen wir also auf

$$2(\#\hat{t})(\#\hat{s}) + \min\{\#\hat{t}, \#\hat{s}\}$$

Operationen. Da $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein zulässiger Blockbaum ist, muss t oder s ein Blatt sein. Im ersten Fall schätzen wir den ersten Summanden durch $2r_{\mathcal{I}}\#\hat{s}$ ab und den zweiten durch $\#\hat{t}$, so dass sich

$$2r_{\mathcal{I}}\#\hat{s} + \#\hat{t} \leq 2r_{\mathcal{I}}(\#\hat{t} + \#\hat{s})$$

ergibt, im zweiten Fall schätzen wir den ersten Summanden durch $2r_{\mathcal{J}}\#\hat{t}$ und den zweiten durch $\#\hat{s}$ ab, so dass wir

$$2r_{\mathcal{J}}\#\hat{t} + \#\hat{s} \leq 2r_{\mathcal{J}}(\#\hat{t} + \#\hat{s})$$

erhalten. Als Schranke für den Gesamtaufwand erhalten wir so

$$2\max\{k, r_{\mathcal{I}}, r_{\mathcal{J}}\} \sum_{b=(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}} \#\hat{t} + \#\hat{s}.$$

Nun können wir Lemma 2.28 anwenden, um die Anzahl der Rechenoperationen durch

$$C_{\mathrm{sp}}(p+1)\max\{k, r_{\mathcal{I}}, r_{\mathcal{J}}\}(\#\hat{t} + \#\hat{s})$$

abzuschätzen, und das ist das angestrebte Ergebnis. ∎

6.2 Kürzen auf niedrigen Rang

In der Regel werden bei der Durchführung arithmetischer Operationen Matrizen hohen Rangs entstehen: Wenn wir zwei Rang-k-Approximationen addieren erhalten wir wegen

$$AB^* + CD^* = \begin{pmatrix} A & C \end{pmatrix} \begin{pmatrix} B^* \\ D^* \end{pmatrix} = \begin{pmatrix} A & C \end{pmatrix} \begin{pmatrix} B & D \end{pmatrix}^*$$

im Allgemeinen eine Rang-$2k$-Matrix, selbst wenn die Summe sich eigentlich auch durch Rang k approximieren ließe. Unser erstes Ziel sollte also darin bestehen, Verfahren zur systematischen Berechnung von Approximationen niedrigen Ranges zu entwickeln.

Im Prinzip könnte man dafür die Kreuzapproximation (vgl. Kapitel 4) verwenden. Allerdings beruht sie auf einer Dreieckszerlegung der betreffenden Matrix, und es ist bekannt, dass derartige Zerlegungen bei schlecht konditionierten Problemen infolge von Rundungsfehlern zu unbefriedigenden Ergebnissen führen können.

Numerisch stabiler sind orthogonale Transformationen: Es ist möglich eine beliebige Matrix mit derartigen Transformationen numerisch stabil auf Diagonalgestalt zu transformieren, und die so definierte *Singulärwertzerlegung* bietet alle Informationen, die wir für die Konstruktion von Approximationen benötigen.

Definition 6.2 (Singulärwertzerlegung) *Sei $\mathbf{X} \in \mathbb{K}^{n\times m}$ eine beliebige Matrix, und sei $p \in \mathbb{N}$. Falls orthogonale Matrizen $\mathbf{U} \in \mathbb{K}^{n\times p}$ und $\mathbf{V} \in \mathbb{K}^{m\times p}$ und eine reelle Diagonalmatrix $\mathbf{\Sigma} \in \mathbb{R}^{p\times p}$ die Bedingungen*

$$\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^*, \qquad \mathbf{\Sigma} = \begin{pmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_p \end{pmatrix}, \qquad \sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_p \geq 0 \tag{6.2}$$

erfüllen, nennen wir $(\mathbf{\Sigma}, \mathbf{U}, \mathbf{V})$ eine (dünne) Singulärwertzerlegung *der Matrix $\mathbf{X}$. Die Koeffizienten $\sigma_1, \ldots, \sigma_p$ bezeichnen wir als* Singulärwerte *der Matrix $\mathbf{X}$, die Spalten der Matrizen $\mathbf{U}$ und $\mathbf{V}$ als die* linken (bzw. rechten) Singulärvektoren.

Die Singulärwertzerlegung bietet gegenüber der Kreuzapproximation den Vorteil, dass sie auf *orthogonalen* Matrizen $\mathbf{U}$ und $\mathbf{V}$ statt auf Dreiecksmatrizen basiert und deshalb wesentlich bessere numerische Eigenschaften besitzt.

Bemerkung 6.3 (Berechnung) *Für jede Matrix $\mathbf{X} \in \mathbb{K}^{n\times m}$ kann eine Singulärwertzerlegung praktisch berechnet werden (ein Existenzbeweis findet sich in Lemma 6.6).*

Beispielsweise lässt sich diese Aufgabe theoretisch auf das Lösen des symmetrischen Eigenwertproblems für $\mathbf{X}^*\mathbf{X}$ oder $\mathbf{X}\mathbf{X}^*$ zurückführen: Aus (6.2) folgt

$$\mathbf{X}^*\mathbf{X} = \mathbf{V}\mathbf{\Sigma}^2\mathbf{V}^*, \qquad \mathbf{X}\mathbf{X}^* = \mathbf{U}\mathbf{\Sigma}^2\mathbf{U}^*,$$

also stehen die Eigenvektoren in enger Beziehung zu den Matrizen $\mathbf{U}$ und $\mathbf{V}$, während die Eigenwerte gerade die Quadrate der Singulärwerte sind. Mit Hilfe geeigneter Algorithmen [17, 15, 16] kann das Eigenwertproblem effizient gelöst werden, ohne die numerisch instabile Berechnung der Matrix-Produkte explizit durchführen zu müssen. Mit diesem Ansatz genügen in der Regel

$$C_{svd} nm \min\{n, m\}$$

Operationen (mit einer Konstante C_{svd}), um die Singulärwertzerlegung bis auf einen Restfehler in Größenordnung der Maschinengenauigkeit zu berechnen. Die Singulärwerte sind dabei durch (6.2) eindeutig bestimmt.

Die Singulärwertzerlegung kann als Variante eines uns bereits bekannten Ansatzes interpretiert werden: Wenn wir für $\nu \in \{1, \ldots, p\}$ mit $\mathbf{u}_\nu \in \mathbb{K}^n$ und $\mathbf{v}_\nu \in \mathbb{K}^m$ jeweils die ν-te Spalte der Matrizen $\mathbf{U}$ und $\mathbf{V}$ bezeichnen, können wir die Gleichung (6.2) auch in der Form

$$\mathbf{X} = \sum_{\nu=1}^{p} \mathbf{u}_\nu \sigma_\nu \mathbf{v}_\nu^*$$

schreiben, also als Summe von p Rang-1-Matrizen. Diese Darstellung erinnert an die einer entarteten Kernfunktion (vgl. Definition 3.1), denn auch sie entkoppelt Zeilen und Spalten der Matrix voneinander.

Da sowohl die Vektoren $\mathbf{u}_\nu$ als auch die Vektoren $\mathbf{v}_\nu$ orthogonale Basen bilden, liegt es nahe, zu vermuten, dass alleine die Singulärwerte σ_ν darüber entscheiden, wie wichtig

einzelne Terme dieser Summe für die Gesamtmatrix $\mathbf{X}$ sind. Die Singulärwerte sind absteigend sortiert, so dass es sich anbietet,

$$\widetilde{\mathbf{X}} = \sum_{\nu=1}^{k} \mathbf{u}_\nu \sigma_\nu \mathbf{v}_\nu^* \tag{6.3}$$

als Rang-k-Approximation der Matrix $\mathbf{X}$ zu verwenden.

Lemma 6.4 (Rang-k-Approximation) *Die durch (6.3) definierte Rang-k-Approximation der Matrix $\mathbf{X}$ erfüllt die Fehlergleichungen*

$$\|\mathbf{X} - \widetilde{\mathbf{X}}\|_2 = \sigma_{k+1}, \qquad \|\mathbf{X} - \widetilde{\mathbf{X}}\|_F = \left(\sum_{\nu=k+1}^{p} \sigma_\nu^2 \right)^{1/2}.$$

Beweis. Wir definieren die Matrix

$$\widetilde{\mathbf{\Sigma}} := \begin{pmatrix} \sigma_1 & & & & & \\ & \ddots & & & & \\ & & \sigma_k & & & \\ & & & 0 & & \\ & & & & \ddots & \\ & & & & & 0 \end{pmatrix} \in \mathbb{R}^{p \times p}$$

und erhalten

$$\widetilde{\mathbf{X}} = \mathbf{U} \widetilde{\mathbf{\Sigma}} \mathbf{V}^*.$$

Da die euklidische Norm unter orthogonalen Transformationen invariant ist, gilt

$$\|\mathbf{X} - \widetilde{\mathbf{X}}\|_2 = \|\mathbf{U}(\mathbf{\Sigma} - \widetilde{\mathbf{\Sigma}})\mathbf{V}^*\|_2 = \|\mathbf{\Sigma} - \widetilde{\mathbf{\Sigma}}\|_2 = \|\mathbf{E}\|_2$$

mit der Diagonalmatrix

$$\mathbf{E} := \mathbf{\Sigma} - \widetilde{\mathbf{\Sigma}} = \operatorname{diag}(0, \ldots, 0, \sigma_{k+1}, \ldots, \sigma_p) \in \mathbb{R}^{p \times p}.$$

Da die Singulärwerte absteigend sortiert sind, erhalten wir

$$\|\mathbf{E}\mathbf{x}\|_2 = \left(\sum_{i=k+1}^{p} \sigma_i^2 |x_i|^2 \right)^{1/2} \leq \left(\sum_{i=k+1}^{p} \sigma_{k+1}^2 |x_i|^2 \right)^{1/2} = \sigma_{k+1} \|\mathbf{x}\|_2,$$

also folgt $\|\mathbf{E}\|_2 \leq \sigma_{k+1}$. Durch Einsetzen des $(k+1)$-ten kanonischen Einheitsvektors $\delta_{k+1} \in \mathbb{K}^p$ erhalten wir

$$\|\mathbf{E}\delta_{k+1}\|_2 = \|\sigma_{k+1}\delta_{k+1}\|_2 = \sigma_{k+1}\|\delta_{k+1}\|_2,$$

und somit auch $\|\mathbf{E}\|_2 \geq \sigma_{k+1}$.

Auch die Frobenius-Norm ist invariant unter orthogonalen Transformationen, so dass wir direkt

$$\|\mathbf{X} - \widetilde{\mathbf{X}}\|_F = \|\mathbf{\Sigma} - \widetilde{\mathbf{\Sigma}}\|_F = \|\mathbf{E}\|_F = \left(\sum_{\nu=k+1}^{p} \sigma_\nu^2 \right)^{1/2}$$

nachrechnen können. ∎

Mit der Singulärwertzerlegung können wir also nicht nur Rang-k-Approximationen einer beliebigen Matrix konstruieren, uns steht auch der dabei auftretende Fehler *explizit* zur Verfügung.

Das eröffnet uns die Möglichkeit, den Fehler zu steuern: Wir müssen nicht zwingend mit einem festen Rang k arbeiten, wir können den Rang k auch so wählen, dass die von uns berechnete Approximation eine gewisse Genauigkeit $\epsilon \in \mathbb{R}_{>0}$ erreicht. Verwenden wir beispielsweise

$$k := \min\{\nu \in \{1, \ldots, p\} \ : \ \sigma_\nu \leq \epsilon\} - 1,$$

so folgt

$$\|\mathbf{X} - \widetilde{\mathbf{X}}\|_2 \leq \epsilon.$$

Entsprechend erhalten wir für die Frobenius-Norm mit

$$k := \min\{\nu \in \{1, \ldots, p\} \ : \ \sigma_\nu^2 + \ldots + \sigma_p^2 \leq \epsilon^2\} - 1$$

die Abschätzung

$$\|\mathbf{X} - \widetilde{\mathbf{X}}\|_F \leq \epsilon.$$

In der Praxis sind häufig relative Fehlerabschätzungen von Interesse. Indem wir Lemma 6.4 auf $k = 0$ anwenden, erhalten wir $\|\mathbf{X}\|_2 = \sigma_1$, also ergibt sich aus der Wahl

$$k := \min\{\nu \in \{1, \ldots, p\} \ : \ \sigma_\nu \leq \epsilon\sigma_1\} - 1,$$

direkt die relative Fehlerschranke

$$\|\mathbf{X} - \widetilde{\mathbf{X}}\|_2 \leq \epsilon\sigma_1 = \epsilon\|\mathbf{X}\|_2.$$

Entsprechend gilt $\|\mathbf{X}\|_F = \sqrt{\sigma_1^2 + \ldots + \sigma_p^2}$ und die Wahl

$$k := \min\{\nu \in \{1, \ldots, p\} \ : \ \sigma_\nu^2 + \ldots + \sigma_p^2 \leq \epsilon^2(\sigma_1^2 + \ldots + \sigma_p^2)\} - 1$$

führt zu der relativen Fehlerschranke

$$\|\mathbf{X} - \widetilde{\mathbf{X}}\|_F \leq \epsilon \left(\sigma_1^2 + \ldots + \sigma_p^2\right)^{1/2} = \epsilon\|\mathbf{X}\|_F$$

in der Frobenius-Norm.

Man kann beweisen (vgl. Lemma 6.7 und Lemma 6.11), dass die von uns konstruierte Rang-k-Approximation $\widetilde{\mathbf{X}}$ die *bestmögliche* ist, dass also keine andere Rang-k-Matrix einen geringeren Fehler in Spektral- und Frobeniusnorm aufweisen kann.

Leider ist im allgemeinen Fall die Berechnung der für unsere Konstruktion entscheidenden Singulärwertzerlegung für große Matrizen sehr aufwendig und damit unattraktiv.

Der Aufwand des Kürzens einer Matrix auf geringeren Rang lässt sich glücklicherweise deutlich reduzieren, falls wir etwas über die Struktur der vorliegenden Matrix wissen. Von besonderer Bedeutung ist hier der Fall, dass bereits eine, eventuell suboptimale, Niedrigrang-Darstellung der Ausgangsmatrix zur Verfügung steht. Seien also $\mathbf{A} \in \mathbb{K}^{n\times k}$ und $\mathbf{B} \in \mathbb{K}^{m\times k}$ mit

$$\mathbf{X} = \mathbf{A}\mathbf{B}^*$$

gegeben. Wir suchen nach einer Darstellung

$$\widetilde{\mathbf{X}} = \widetilde{\mathbf{A}}\widetilde{\mathbf{B}}^*$$

mit $\widetilde{\mathbf{A}} \in \mathbb{K}^{n\times\tilde{k}}$ und $\widetilde{\mathbf{B}} \in \mathbb{K}^{m\times\tilde{k}}$ für ein $\tilde{k} \leq k$.

Um diese Matrix wie im allgemeinen Fall konstruieren zu können, benötigen wir die Singulärwertzerlegung der Matrix $\mathbf{X}$. Für ihre Berechnung können wir auf die Rang-k-Darstellung zurückgreifen: Wir berechnen eine dünne QR-Zerlegung der Matrix $\mathbf{A}$, also eine orthogonale Matrix $\mathbf{Q} \in \mathbb{K}^{n\times k}$ und eine obere Dreiecksmatrix $\mathbf{R} \in \mathbb{K}^{k\times k}$ mit

$$\mathbf{A} = \mathbf{Q}\mathbf{R}.$$

Die dünne QR-Faktorisierung kann ähnlich der konventionellen mit Hilfe von Householder-Transformationen konstruiert werden, die die Spalten der Matrix der Reihe nach auf obere Dreiecksgestalt bringen: In einem ersten Schritt verwenden wir eine Transformation $\mathbf{Q}_1$, um in der ersten Spalte der Matrix $\mathbf{A}$ alles unterhalb der Diagonalen zu eliminieren. Mit den folgenden Spalten verfahren wir entsprechend und erhalten schließlich

$$\mathbf{Q}_k \dots \mathbf{Q}_1 \mathbf{A} = \begin{pmatrix} \mathbf{R} \\ \mathbf{0} \end{pmatrix} \tag{6.4}$$

mit einer oberen Dreiecksmatrix $\mathbf{R} \in \mathbb{K}^{k\times k}$. Da die Householder-Spiegelungen selbstinvers sind, ist die gesuchte Faktorisierung durch

$$\mathbf{A} = \mathbf{Q}_1 \dots \mathbf{Q}_k \begin{pmatrix} \mathbf{R} \\ \mathbf{0} \end{pmatrix} = \mathbf{Q}_1 \dots \mathbf{Q}_k \begin{pmatrix} \mathbf{I} \\ \mathbf{0} \end{pmatrix} \mathbf{R} = \mathbf{Q}\mathbf{R}, \qquad \mathbf{Q} := \mathbf{Q}_1 \dots \mathbf{Q}_k \begin{pmatrix} \mathbf{I} \\ \mathbf{0} \end{pmatrix},$$

gegeben. Mit Hilfe dieser Zerlegung erhalten wir

$$\mathbf{X} = \mathbf{A}\mathbf{B}^* = \mathbf{Q}\mathbf{R}\mathbf{B}^* = \mathbf{Q}\widehat{\mathbf{X}}, \qquad \widehat{\mathbf{X}} := \mathbf{R}\mathbf{B}^* \in \mathbb{K}^{k\times m}.$$

Da wir davon ausgehen, dass k relativ klein ist, können wir es uns erlauben, die Singulärwertzerlegung

$$\widehat{\mathbf{X}} = \widehat{\mathbf{U}}\mathbf{\Sigma}\mathbf{V}^*$$

der $k \times m$-Matrix $\widehat{\mathbf{X}}$ zu berechnen, denn dafür sind nach Bemerkung 6.3 nicht mehr als $C_{\mathrm{svd}}k^2m$ Operationen erforderlich.

Damit erhalten wir

$$\mathbf{X} = \mathbf{Q}\widehat{\mathbf{X}} = \mathbf{Q}\widehat{\mathbf{U}}\mathbf{\Sigma}\mathbf{V}^* = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^*, \qquad \mathbf{U} := \mathbf{Q}\widehat{\mathbf{U}},$$

procedure trunc_rk($\tilde{k}$, **var** $\mathbf{A}$, $\mathbf{B}$);
if $\#\hat{t} \leq k$ oder $\#\hat{s} \leq k$ **then**
  $\mathbf{X} \leftarrow \mathbf{AB}^*$;
  Berechne die Singulärwertzerlegung $\mathbf{U\Sigma V}^* = \mathbf{X}$;
  $\mathbf{A} \leftarrow \mathbf{U}|_{\hat{t}\times\tilde{k}}$; $\mathbf{B} \leftarrow \mathbf{V}|_{\hat{s}\times\tilde{k}}\mathbf{\Sigma}|^*_{\tilde{k}\times\tilde{k}}$
else
  Berechne $\mathbf{QR} = \mathbf{A}$ mit $\mathbf{Q} \in \mathbb{K}^{\hat{t}\times k}$ orthogonal, $\mathbf{R} \in \mathbb{K}^{k\times k}$;
  $\widehat{\mathbf{X}} \leftarrow \mathbf{RB}^*$;
  Berechne die Singulärwertzerlegung $\widehat{\mathbf{U}}\mathbf{\Sigma V}^* = \widehat{\mathbf{X}}$;
  $\mathbf{A} \leftarrow \mathbf{Q}\widehat{\mathbf{U}}|_{k\times\tilde{k}}$; $\mathbf{B} \leftarrow \mathbf{V}|_{\hat{s}\times\tilde{k}}\mathbf{\Sigma}|^*_{\tilde{k}\times\tilde{k}}$
end if

Abbildung 6.3: Kürzen einer Matrix $\mathbf{AB}^*$ in faktorisierter Rang-k-Darstellung auf eine approximative Rang-$\tilde{k}$-Darstellung.

und da $\mathbf{U}$ als Produkt der orthogonalen Matrizen $\mathbf{Q}$ und $\widehat{\mathbf{U}}$ ebenfalls orthogonal sein muss, haben wir eine Singulärwertzerlegung der Matrix $\mathbf{X}$ konstruiert.

In praktischen Implementierungen kann es passieren, dass eine faktorisierte Rang-k-Darstellung für eine Matrix berechnet wird, deren Rang wegen $\#\hat{t} < k$ oder $\#\hat{s} < k$ niemals den Rang k erreichen kann. In diesem Fall ist es klüger, auf die QR-Zerlegung zu verzichten und mit der Originalmatrix zu rechnen. Der in dieser Weise entstehende vollständige Algorithmus zum Kürzen einer Rang-k-Darstellung ist in Abbildung 6.3 angegeben.

Lemma 6.5 (Aufwand des Kürzens) *Falls* $\mathbf{X} \in \mathbb{K}^{\hat{t}\times\hat{s}}$ *in Rang-k-Darstellung*

$$\mathbf{X} = \mathbf{AB}^*, \qquad \mathbf{A} \in \mathbb{K}^{\hat{t}\times k}, \qquad \mathbf{B} \in \mathbb{K}^{\hat{s}\times k}$$

vorliegt, können wir die optimale Rang-$\tilde{k}$-Approximation

$$\widetilde{\mathbf{X}} = \widetilde{\mathbf{A}}\widetilde{\mathbf{B}}^*, \qquad \widetilde{\mathbf{A}} \in \mathbb{K}^{\hat{t}\times\tilde{k}}, \qquad \widetilde{\mathbf{B}} \in \mathbb{K}^{\hat{s}\times\tilde{k}}$$

für jedes $0 \leq \tilde{k} \leq k$ *in nicht mehr als*

$$C_{svdk}(\#\hat{t} + \#\hat{s})k^2$$

Operationen berechnen, wobei die Konstante durch

$$C_{svdk} := \max\{C_{svd} + 2, 8\}$$

gegeben ist.

Beweis. Wir setzen zur Abkürzung $n = \#\hat{t}$ und $m = \#\hat{s}$ und untersuchen zunächst den interessanten Fall, also $k < \min\{n, m\}$.

Die Konstruktion einer Householder-Transformation für eine Spalte und ihre Anwendung auf diese Spalte erfordert nicht mehr als $3n+2$ Operationen, die Anwendung auf die weiteren Spalten nicht mehr als $4n$ Operationen, so dass wir wegen $n > 1$ von $4n$ Operationen pro Spalte ausgehen können. Im ℓ-ten Schritt des Verfahrens sind $k-\ell+1$ Spalten der Matrix betroffen, also erfordert die Konstruktion der QR-Zerlegung insgesamt nicht mehr als

$$\sum_{\ell=1}^{k} 4n(k-\ell+1) = 4n\sum_{\ell=1}^{k} k-\ell+1 = 4n\sum_{\ell=1}^{k} \ell = 4n\frac{k(k+1)}{2} = 2nk(k+1)$$

Operationen. Die Berechnung des Produkts $\widehat{X} = \mathbf{R}\mathbf{B}^*$ lässt sich zeilenweise durchführen: Der Eintrag $\widehat{X}_{ij}$ ergibt sich als das Skalarprodukt der i-ten Zeile der Matrix $\mathbf{R}$ mit der j-ten Spalte der Matrix $\mathbf{B}^*$. Infolge der Dreiecksstruktur der Matrix sind die ersten $i-1$ Einträge gleich null, so dass lediglich $k-i+1$ Einträge zu berücksichtigen sind und der Eintrag mit $2(k-i+1)-1$ Operationen berechnet werden kann. Insgesamt benötigen wir deshalb

$$m\sum_{i=1}^{k}(2(k-i+1)-1) = m\sum_{i=1}^{k}(2i-1) = mk^2$$

Operationen für die Konstruktion der Matrix $\widehat{\mathbf{X}}$.

Für die Berechnung der Singulärwertzerlegung stellt uns bereits Bemerkung 6.3 eine Abschätzung des zu erwartenden Rechenaufwands zur Verfügung: Es sind nicht mehr als

$$C_{\mathrm{svd}}k^2m$$

Operationen erforderlich, um $\widehat{\mathbf{U}}$, $\mathbf{\Sigma}$ und $\mathbf{V}$ zu finden.

Um schließlich die Matrix $\mathbf{U} = \mathbf{Q}\widehat{\mathbf{U}}$ zu berechnen, wenden wir die Householder-Transformationen aus (6.4) in umgekehrter Reihenfolge an:

$$\mathbf{U} = \mathbf{Q}_1 \dots \mathbf{Q}_k \begin{pmatrix} \widehat{\mathbf{U}} \\ \mathbf{0} \end{pmatrix}.$$

Jede Transformation erfordert nicht mehr als $4nk$ Operationen, so dass insgesamt $4nk^2$ Operationen ausreichen. Schließlich müssen noch alle Spalten der Matrix $\mathbf{V}$ mit den Faktoren $\sigma_1, \dots, \sigma_{\tilde{k}}$ multipliziert werden, und diese Aufgabe lässt sich in mk Operationen bewältigen.

Der Aufwand für die gesamte Berechnung ist damit in diesem Fall durch

$$\begin{aligned} 2nk(k+1) + mk^2 + C_{\mathrm{svd}}k^2m + 4nk^2 + mk &= 6nk^2 + 2nk + (C_{\mathrm{svd}}+1)mk^2 + mk \\ &\leq 8nk^2 + (C_{\mathrm{svd}}+2)mk^2 \leq C_{\mathrm{svdk}}(n+m)k^2 \end{aligned}$$

beschränkt.

Wir müssen auch den Sonderfall $\min\{n,m\} \leq k$ untersuchen. In diesem Fall wird $\mathbf{X}$ durch $nm(2k-1)$ Operationen berechnet. Die Singulärwertzerlegung benötigt gemäß Bemerkung 6.3 dann nicht mehr als $C_{\mathrm{svd}}nm\min\{n,m\} \leq C_{\mathrm{svd}}nmk$ Operationen. Die

Bestimmung von $\mathbf{B}$ erfordert es schließlich noch, die ersten $\tilde{k}$ Spalten der Matrix $\mathbf{V}$ mit jeweils einem Faktor zu multiplizieren. Da $\mathbf{V}$ nicht mehr als n Spalten haben kann, fallen dafür nicht mehr als nm Operationen an, so dass insgesamt

$$\begin{aligned} nm(2k-1) + C_{\text{svd}} nmk + nm &= (C_{\text{svd}} + 2)nmk \\ &\leq C_{\text{svdk}} \max\{n,m\} \min\{n,m\} \leq C_{\text{svdk}}(n+m)k^2 \end{aligned}$$

Operationen ausreichen. ∎

6.3 Theoretische Grundlagen des Kürzungsalgorithmus

Da die Singulärwertzerlegung zu den zentralen Hilfsmitteln bei der Konstruktion hierarchischer Matrizen gehört, empfiehlt es sich, kurz ihre für uns relevanten Eigenschaften zu rekapitulieren. Von Interesse ist dabei natürlich zunächst die Frage, ob sich überhaupt für jede Matrix eine Singulärwertzerlegung finden lässt.

Lemma 6.6 (Existenz) *Jede beliebige Matrix $\mathbf{X} \in \mathbb{K}^{n\times m}$ besitzt eine Singulärwertzerlegung $(\mathbf{\Sigma}, \mathbf{U}, \mathbf{V})$.*

Beweis. Wir stellen zunächst fest, dass es ausreicht, quadratische Matrizen zu untersuchen: Falls $n > m$ gelten sollte, können wir die dünne QR-Zerlegung $\mathbf{QR} = \mathbf{X}$ der Matrix mit einer orthogonalen Matrix $\mathbf{Q} \in \mathbb{K}^{n\times m}$ und einer *quadratischen* Rechtecksmatrix $\mathbf{R} \in \mathbb{K}^{m\times m}$ verwenden. Aus einer Singulärwertzerlegung der Matrix $\mathbf{R}$ ergibt sich durch Multiplikation mit $\mathbf{Q}$ unmittelbar eine der Matrix $\mathbf{X}$. Falls $n < m$ gilt, können wir entsprechend die Zerlegung der adjungierten Matrix $\mathbf{QR} = \mathbf{X}^*$ berechnen und entsprechend vorgehen.

Sei also nun $\mathbf{X} \in \mathbb{K}^{n\times n}$ eine quadratische Matrix. Wir beweisen die Existenz einer Singulärwertzerlegung per Induktion über n.

Für $n = 1$ ist die Aussage trivial.

Gelte nun die Aussage für ein $n \in \mathbb{N}$, und sei $\mathbf{X} \in \mathbb{K}^{(n+1)\times(n+1)}$ gegeben. Die stetige Abbildung

$$s : (\mathbb{K}^{n+1} \setminus \{\mathbf{0}\}) \times (\mathbb{K}^{n+1} \setminus \{\mathbf{0}\}) \to \mathbb{R}, \qquad (\mathbf{y}, \mathbf{z}) \mapsto \frac{|\langle \mathbf{y}, \mathbf{Xz}\rangle_2|}{\|\mathbf{y}\|_2 \|\mathbf{z}\|_2} = \frac{|\langle \mathbf{X}^*\mathbf{y}, \mathbf{z}\rangle_2|}{\|\mathbf{y}\|_2 \|\mathbf{z}\|_2},$$

nimmt auf der kompakten Menge

$$K := \{(\mathbf{y}, \mathbf{z}) \in \mathbb{K}^{n+1} \times \mathbb{K}^{n+1} \ : \ \|\mathbf{y}\|_2 = 1, \ \|\mathbf{z}\|_2 = 1\}$$

nach dem Extremwertsatz von Weierstraß ihr Maximum σ_1 in einem Paar $(\mathbf{y}, \mathbf{z}) \in K$ an. Da die Skalierung der Vektoren für s keine Rolle spielt, ist σ_1 auch das Maximum auf dem gesamten Definitionsbereich der Funktion.

Aus der Cauchy-Schwarz-Ungleichung folgt, dass die Abbildung

$$\mathbf{y} \mapsto \frac{|\langle \mathbf{y}, \mathbf{Xz}\rangle_2|}{\|\mathbf{y}\|_2}$$

ihr Maximum $\|\mathbf{Xz}\|_2$ genau dann annimmt, wenn $\mathbf{y}$ und $\mathbf{Xz}$ voneinander linear abhängig sind, also muss ein $\alpha \in \mathbb{K}$ mit

$$\mathbf{Xz} = \alpha\mathbf{y}$$

existieren. Entsprechend können wir schließen, dass auch ein $\beta \in \mathbb{K}$ mit

$$\mathbf{X}^*\mathbf{y} = \beta\mathbf{z}$$

existieren muss. Indem wir $\mathbf{y}$ oder $\mathbf{z}$ geeignet skalieren, können wir dafür sorgen, dass $\alpha \in \mathbb{R}_{\geq 0}$ gilt.

Wir wählen Householder-Spiegelungen $\mathbf{U}_1$ und $\mathbf{V}_1$ mit

$$\mathbf{U}_1\delta_1 = \mathbf{y}, \qquad \mathbf{V}_1\delta_1 = \mathbf{z}$$

und erhalten

$$\begin{aligned}\mathbf{U}_1^*\mathbf{X}\mathbf{V}_1\delta_1 &= \mathbf{U}_1^*\mathbf{X}\mathbf{z} = \alpha\mathbf{U}_1^*\mathbf{y} = \alpha\delta_1,\\ \mathbf{V}_1^*\mathbf{X}^*\mathbf{U}_1\delta_1 &= \mathbf{V}_1^*\mathbf{X}^*\mathbf{y} = \beta\mathbf{V}_1^*\mathbf{z} = \beta\delta_1,\end{aligned}$$

also insgesamt

$$\mathbf{U}_1^*\mathbf{X}\mathbf{V}_1 = \begin{pmatrix}\alpha & \mathbf{0}\\ \mathbf{0} & \widehat{\mathbf{X}}\end{pmatrix}$$

mit einer Matrix $\widehat{\mathbf{X}} \in \mathbb{K}^{n\times n}$, auf die wir die Induktionsvoraussetzung $\widehat{\mathbf{X}}$ anwenden können, um den Induktionsschritt abzuschließen. ∎

Wie wir gesehen haben, erlaubt es uns die Singulärwertzerlegung, sehr einfach Approximationen jeder gewünschten Genauigkeit zu konstruieren. Unser Ansatz besitzt allerdings noch einen zweiten Vorteil: Er berechnet die *optimale* Approximation bezüglich der Spektral- und der Frobenius-Norm:

Lemma 6.7 (Bestapproximation Spektralnorm) *Sei $\mathbf{X} \in \mathbb{K}^{n\times m}$ eine Matrix von Rang $p \in \mathbb{N}$, sei $\mathbf{R} \in \mathbb{K}^{n\times m}$ eine Matrix von Rang $k \in \{0,\ldots,p-1\}$. Dann existiert ein Vektor $\mathbf{z} \in \operatorname{Kern}\mathbf{R} \subseteq \mathbb{K}^m$ mit*

$$\|(\mathbf{X}-\mathbf{R})\mathbf{z}\|_2 \geq \sigma_{k+1}, \qquad \|\mathbf{z}\|_2 = 1,$$

und damit gilt insbesondere auch

$$\|\mathbf{X}-\mathbf{R}\|_2 \geq \sigma_{k+1}.$$

Beweis. Da $\mathbf{R}$ eine Matrix von Rang k ist, ist ihr Bild höchstens k-dimensional. Aus dem Dimensionssatz folgt, dass ihr Kern mindestens $(m-k)$-dimensional sein muss. Dank $k < p \leq \min\{n,m\}$ gilt $m-k > m-p \geq 0$. Wir definieren den Raum

$$\mathcal{V}_{k+1} := \operatorname{span}\{\mathbf{v}_\nu \ : \ \nu \in \{1,\ldots,k+1\}\},$$

der gerade die Dimension $k+1$ besitzt. Da sowohl $\mathcal{V}_{k+1}$ als auch der Kern der Matrix $\mathbf{R}$ Teilräume des m-dimensionalen Raums $\mathbb{K}^m$ sind, kann ihr Schnitt nicht nur den Nullvektor enthalten. Wir können also einen Vektor

$$\mathbf{z} \in (\mathcal{V}_{k+1} \cap \operatorname{Kern} \mathbf{R}) \setminus \{\mathbf{0}\}$$

finden, und wir können diesen Vektor auch so skalieren, dass $\|\mathbf{z}\|_2 = 1$ gilt. Aus $\mathbf{z} \in \mathcal{V}_{k+1}$ folgt, dass es Koeffizienten $(\alpha_\nu)_{\nu=1}^{k+1}$ mit

$$\mathbf{z} = \sum_{\nu=1}^{k+1} \alpha_\nu \mathbf{v}_\nu$$

geben muss, und aus der Orthogonalität der Vektoren $\mathbf{v}_\nu$ erhalten wir

$$\begin{aligned}(\mathbf{V}^*\mathbf{z})_\mu = \langle \mathbf{v}_\mu, \mathbf{z}\rangle_2 &= \sum_{\nu=1}^{k+1} \alpha_\nu \langle \mathbf{v}_\mu, \mathbf{v}_\nu\rangle_2 \\ &= \begin{cases} \alpha_\mu & \text{falls } \mu \le k+1, \\ 0 & \text{ansonsten} \end{cases} \qquad \text{für alle } \mu \in \{1,\ldots,m\},\end{aligned}$$

und damit wegen $\mathbf{z} \in \operatorname{Kern} \mathbf{R}$ auch

$$\begin{aligned}\|(\mathbf{X}-\mathbf{R})\mathbf{z}\|_2 = \|\mathbf{X}\mathbf{z}\|_2 = \|\mathbf{U}\mathbf{\Sigma}\mathbf{V}^*\mathbf{z}\|_2 = \|\mathbf{\Sigma}\mathbf{V}^*\mathbf{z}\|_2 &= \left(\sum_{\mu=1}^{k+1} \sigma_\mu^2 |\alpha_\mu|^2\right)^{1/2} \\ \ge \left(\sum_{\mu=1}^{k+1} \sigma_{k+1}^2 |\alpha_\mu|^2\right)^{1/2} &= \sigma_{k+1} \left(\sum_{\mu=1}^{k+1} |\alpha_\mu|^2\right)^{1/2} = \sigma_{k+1}\|\mathbf{z}\|_2.\end{aligned}$$

Dank $\|\mathbf{z}\|_2 = 1$ folgt daraus $\|\mathbf{X}-\mathbf{R}\|_2 \ge \sigma_{k+1}$, also die gewünschte Bestapproximationsaussage in der Spektralnorm. ∎

Um ein entsprechendes Resultat für die Frobenius-Norm zu beweisen, benötigen wir als Hilfsmittel das dieser Norm zugeordnete Skalarprodukt auf dem Raum der Matrizen:

Definition 6.8 (Frobenius-Skalarprodukt) *Die Sesquilinearform*

$$s_F : \mathbb{K}^{\mathcal{I}\times\mathcal{J}} \times \mathbb{K}^{\mathcal{I}\times\mathcal{J}} \to \mathbb{K}, \qquad (\mathbf{X},\mathbf{Y}) \mapsto \sum_{i\in\mathcal{I}} \sum_{j\in\mathcal{J}} \bar{X}_{ij} Y_{ij},$$

ist ein Skalarprodukt auf dem Raum $\mathbb{K}^{\mathcal{I}\times\mathcal{J}}$, das wir als das Frobenius-Skalarprodukt *bezeichnen und als $\langle \mathbf{X},\mathbf{Y}\rangle_F = s_F(\mathbf{X},\mathbf{Y})$ notieren.*

Ebenso wie im Falle des Euklidischen Skalarprodukts ermöglicht es uns auch das Frobenius-Skalarprodukt, Matrizen von einem Argument in das andere mit Hilfe der adjungierten Matrix zu verschieben:

Lemma 6.9 (Frobenius-Skalarprodukt) *Es gilt*

$$\langle \mathbf{Y}, \mathbf{X}^*\mathbf{Z}\rangle_F = \langle \mathbf{XY}, \mathbf{Z}\rangle_F = \langle \mathbf{X}, \mathbf{ZY}^*\rangle_F \quad \text{für alle } \mathbf{X} \in \mathbb{K}^{\mathcal{I}\times\mathcal{J}},\ \mathbf{Y} \in \mathbb{K}^{\mathcal{J}\times\mathcal{K}},\ \mathbf{Z} \in \mathbb{K}^{\mathcal{I}\times\mathcal{K}}.$$

Beweis. Aus der Definition folgt

$$\begin{aligned}
\langle \mathbf{XY}, \mathbf{Z}\rangle_F &= \sum_{i\in\mathcal{I}}\sum_{k\in\mathcal{K}} \overline{(\mathbf{XY})}_{ik} Z_{ik} = \sum_{i\in\mathcal{I}}\sum_{k\in\mathcal{K}}\sum_{j\in\mathcal{J}} \bar{X}_{ij}\bar{Y}_{jk}Z_{ik},\\
\langle \mathbf{Y}, \mathbf{X}^*\mathbf{Z}\rangle_F &= \sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}} \bar{Y}_{jk}(\mathbf{X}^*\mathbf{Z})_{jk} = \sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{I}} \bar{Y}_{jk}\bar{X}_{ij}Z_{ik},\\
\langle \mathbf{X}, \mathbf{ZY}^*\rangle_F &= \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} \bar{X}_{ij}(\mathbf{ZY}^*)_{ij} = \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}} \bar{X}_{ij}Z_{ik}\bar{Y}_{jk},
\end{aligned}$$

und alle Summen sind offenbar identisch. ∎

Im folgenden Beweis werden wir Vektoren mit Matrizen mit einer Spalte identifizieren, also keinen Unterschied zwischen dem Vektor $\mathbf{x} \in \mathbb{K}^{\mathcal{I}}$ und der Matrix $\widehat{\mathbf{x}} \in \mathbb{K}^{\mathcal{I}\times 1}$ mit

$$\widehat{x}_{i1} = x_i \qquad \text{für alle } i \in \mathcal{I}$$

machen. So lässt sich beispielsweise eine Rang-1-Matrix bequem in der Form $\mathbf{R} = \mathbf{ab}^*$ schreiben, statt umständlicher $R_{ij} = a_i b_j$ für alle $i \in \mathcal{I}$ und $j \in \mathcal{J}$ definieren zu müssen. Diese Konvention ist verträglich mit dem Frobenius-Skalarprodukt:

Lemma 6.10 (Spaltenvektoren) *Es gelten*

$$\begin{aligned}
\langle \mathbf{a}, \mathbf{b}\rangle_F &= \langle \mathbf{a}, \mathbf{b}\rangle_2 && \text{für alle } \mathbf{a}, \mathbf{b} \in \mathbb{K}^{\mathcal{I}},\\
\|\mathbf{ab}^*\|_F &= \|\mathbf{a}\|_2\|\mathbf{b}\|_2 && \text{für alle } \mathbf{a} \in \mathbb{K}^{\mathcal{I}},\ \mathbf{b} \in \mathbb{K}^{\mathcal{J}}.
\end{aligned}$$

Beweis. Die erste Gleichung folgt direkt aus der Definition.

Zum Nachweis der zweiten Gleichung wählen wir $\mathbf{a} \in \mathbb{K}^{\mathcal{I}}$ und $\mathbf{b} \in \mathbb{K}^{\mathcal{J}}$ und finden

$$\|\mathbf{ab}^*\|_F^2 = \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} |a_i|^2|b_j|^2 = \sum_{i\in\mathcal{I}} |a_i|^2 \sum_{j\in\mathcal{J}} |b_j|^2 = \|\mathbf{a}\|_2^2\|\mathbf{b}\|_2^2.$$

Das ist die gesuchte Gleichung. ∎

Jetzt stehen uns alle Hilfsmittel zur Verfügung, die wir brauchen, um zu zeigen, dass auch in der Frobenius-Norm keine bessere Rang-k-Approximation als $\widetilde{\mathbf{X}}$ möglich ist.

Lemma 6.11 (Bestapproximation Frobenius-Norm) *Sei $\mathbf{X} \in \mathbb{K}^{n\times m}$ eine Matrix von Rang $p \in \mathbb{N}$, sei $\mathbf{R} \in \mathbb{K}^{n\times m}$ eine Matrix von Rang $k \in \{0, \ldots, p-1\}$. Dann gilt*

$$\|\mathbf{X} - \mathbf{R}\|_F \geq \left(\sum_{\nu=k+1}^{p} \sigma_\nu^2\right)^{1/2}.$$

Beweis. Wir beweisen nun induktiv für alle $\ell \in \mathbb{N}_0$, dass für eine Matrix $\mathbf{R} \in \mathbb{K}^{n\times m}$, deren Kern höchstens ℓ-dimensional ist, gerade

$$\|\mathbf{X} - \mathbf{R}\|_F^2 \geq \sum_{\nu=m-\ell+1}^{p} \sigma_\nu^2$$

gilt. Der Induktionsanfang ist einfach: Falls der Kern die Dimension $\ell = 0$ besitzt, ist die Summe leer, also gleich null, und damit sicherlich nicht größer als die Norm des Fehlers.

Gelte nun die Aussage für $\ell \in \mathbb{N}_0$. Sei $\mathbf{R} \in \mathbb{K}^{n\times m}$ eine Matrix, deren Kern höchstens die Dimension $\ell + 1$ besitzt. Nach Induktionsvoraussetzung genügt es, den Fall zu untersuchen, dass die Dimension des Kerns *genau* $\ell + 1$ beträgt. Dann muss insbesondere $\ell + 1 \leq m$ gelten, und nach dem Dimensionssatz muss das Bild die Dimension $k := m - (\ell + 1)$ besitzen. Unser Ziel ist es nun, die Abschätzung

$$\|\mathbf{X} - \mathbf{R}\|_F^2 \geq \sum_{\nu=m-\ell}^{p} \sigma_\nu^2 = \sum_{\nu=k+1}^{p} \sigma_\nu^2 \tag{6.5}$$

zu beweisen. Falls $k \geq p$ gelten sollte, ist die Summe auf der rechten Seite der Ungleichung (6.5) leer, also gilt die Aussage infolge der Positivität der Norm.

Sei also nun $k < p$. Nach Lemma 6.7 existiert dann ein Vektor $\mathbf{z} \in \operatorname{Kern} \mathbf{R}$ mit $\|\mathbf{z}\|_2 = 1$ und

$$\|(\mathbf{X} - \mathbf{R})\mathbf{z}\|_2 = \|\mathbf{X}\mathbf{z}\|_2 \geq \sigma_{k+1} = \sigma_{m-\ell} > 0.$$

Um die Induktionsvoraussetzung anwenden zu können, setzen wir

$$\widetilde{\mathbf{R}} := \mathbf{R} + \mathbf{X}\mathbf{z}\mathbf{z}^*,$$

wir erweitern also $\mathbf{R}$ um eine Rang-1-Matrix, die dafür sorgt, dass

$$\widetilde{\mathbf{R}}\mathbf{z} = \mathbf{R}\mathbf{z} + \mathbf{X}\mathbf{z} = \mathbf{X}\mathbf{z}$$

gilt, dass die Matrix also den Vektor $\mathbf{z}$ auf denselben Vektor wie die Matrix $\mathbf{X}$ abbildet.

Da $\mathbf{R}$ von Rang k ist, können wir linear unabhängige Vektoren $\mathbf{b}_1, \ldots, \mathbf{b}_k \in \mathbb{K}^n$ im Bild der Matrix $\mathbf{R}$ finden. Wir wählen Urbilder $\mathbf{c}_1, \ldots, \mathbf{c}_k \in \mathbb{K}^m$ mit $\mathbf{R}\mathbf{c}_i = \mathbf{b}_i$, die ebenfalls linear unabhängig sein müssen. Da wir Vielfache von $\mathbf{z}$ zu den $\mathbf{c}_i$ hinzuaddieren können, ohne diese Eigenschaft zu verlieren, dürfen wir ohne Beschränkung der Allgemeinheit $\langle \mathbf{c}_i, \mathbf{z}\rangle_2 = 0$ für alle $i \in \{1, \ldots, k\}$ voraussetzen. Damit haben wir $k+1$ linear unabhängige Vektoren $\{\mathbf{c}_1, \ldots, \mathbf{c}_k, \mathbf{z}\}$ gefunden, die wegen

$$\widetilde{\mathbf{R}}\mathbf{c}_i = \mathbf{R}\mathbf{c}_i + \mathbf{X}\mathbf{z}\mathbf{z}^*\mathbf{c}_i = \mathbf{b}_i \qquad \text{für alle } i \in \{1, \ldots, k\}$$

nicht im Kern der Matrix $\widetilde{\mathbf{R}}$ liegen, also kann der Kern höchstens die Dimension $m - (k + 1) = \ell$ besitzen, und wir können die Induktionsvoraussetzung anwenden, um

$$\|\mathbf{X} - \widetilde{\mathbf{R}}\|_F^2 \geq \sum_{\nu=m-\ell+1}^{p} \sigma_\nu^2$$

zu erhalten. Nach Definition des Frobenius-Skalarprodukts und mit Hilfe der Lemmas 6.9 und 6.10 folgt

$$\begin{aligned}\|\mathbf{X}-\widetilde{\mathbf{R}}\|_F^2 &= \|\mathbf{X}-\mathbf{R}-\mathbf{X}\mathbf{z}\mathbf{z}^*\|_F^2\\ &= \|\mathbf{X}-\mathbf{R}\|_F^2 - 2\operatorname{Re}\langle\mathbf{X}\mathbf{z}\mathbf{z}^*,\mathbf{X}-\mathbf{R}\rangle_F + \|\mathbf{X}\mathbf{z}\mathbf{z}^*\|_F^2\\ &= \|\mathbf{X}-\mathbf{R}\|_F^2 - 2\operatorname{Re}\langle\mathbf{X}\mathbf{z},(\mathbf{X}-\mathbf{R})\mathbf{z}\rangle_F + \|\mathbf{X}\mathbf{z}\|_2^2\|\mathbf{z}\|_2^2\\ &= \|\mathbf{X}-\mathbf{R}\|_F^2 - 2\operatorname{Re}\langle\mathbf{X}\mathbf{z},\mathbf{X}\mathbf{z}\rangle_2 + \|\mathbf{X}\mathbf{z}\|_2^2\\ &= \|\mathbf{X}-\mathbf{R}\|_F^2 - \|\mathbf{X}\mathbf{z}\|_2^2.\end{aligned}$$

Aus dieser Gleichung folgt insbesondere

$$\|\mathbf{X}-\mathbf{R}\|_F^2 = \|\mathbf{X}-\widetilde{\mathbf{R}}\|_F^2 + \|\mathbf{X}\mathbf{z}\|_2^2 \geq \sum_{\nu=m-\ell+1}^{m}\sigma_\nu^2 + \sigma_{m-\ell}^2 = \sum_{\nu=m-\ell}^{m}\sigma_\nu^2.$$

Damit ist die Induktion vollständig. ∎

6.4 Addition hierarchischer Matrizen

Als erste arithmetische Operation auf der Menge der hierarchischen Matrizen untersuchen wir die Addition. Seien dazu zwei hierarchische Matrizen $\mathbf{X}\in\mathbb{K}^{\mathcal{I}\times\mathcal{J}}$ und $\mathbf{Y}\in\mathbb{K}^{\mathcal{I}\times\mathcal{J}}$ gegeben, die mit Hilfe desselben Blockbaums $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ definiert sind und deren lokaler Rang jeweils k beträgt.

Unser Ziel ist es, die Summe $\mathbf{X}+\mathbf{Y}$ durch eine hierarchische Matrix zu approximieren. Nach Definition ist die exakte Summe durch

$$(\mathbf{X}+\mathbf{Y})|_{\hat{t}\times\hat{s}} = \mathbf{X}|_{\hat{t}\times\hat{s}} + \mathbf{Y}|_{\hat{t}\times\hat{s}} \qquad \text{für alle } b=(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}$$

gegeben. Für unzulässige Blöcke können wir die Summe direkt berechnen. Für zulässige Blöcke $b=(t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ stehen uns die faktorisierten Rang-k-Darstellungen

$$\mathbf{X}|_{\hat{t}\times\hat{s}} = \mathbf{A}_{X,b}\mathbf{B}_{X,b}^*, \qquad \mathbf{Y}|_{\hat{t}\times\hat{s}} = \mathbf{A}_{Y,b}\mathbf{B}_{Y,b}^*,$$

mit $\mathbf{A}_{X,b},\mathbf{A}_{Y,b}\in\mathbb{K}^{\hat{t}\times k}$ und $\mathbf{B}_{X,b},\mathbf{B}_{Y,b}\in\mathbb{K}^{\hat{s}\times k}$ zur Verfügung, so dass wir

$$\begin{aligned}(\mathbf{X}+\mathbf{Y})|_{\hat{t}\times\hat{s}} &= \mathbf{A}_{X,b}\mathbf{B}_{X,b}^* + \mathbf{A}_{Y,b}\mathbf{B}_{Y,b}^* = \begin{pmatrix}\mathbf{A}_{X,b} & \mathbf{A}_{Y,b}\end{pmatrix}\begin{pmatrix}\mathbf{B}_{X,b}^*\\ \mathbf{B}_{Y,b}^*\end{pmatrix}\\ &= \begin{pmatrix}\mathbf{A}_{X,b} & \mathbf{A}_{Y,b}\end{pmatrix}\begin{pmatrix}\mathbf{B}_{X,b} & \mathbf{B}_{Y,b}\end{pmatrix}^*\end{aligned}$$

erhalten. Mit den Matrizen

$$\mathbf{A}_{X+Y,b} := \begin{pmatrix}\mathbf{A}_{X,b} & \mathbf{A}_{Y,b}\end{pmatrix}\in\mathbb{K}^{\hat{t}\times 2k}, \qquad \mathbf{B}_{X+Y,b} := \begin{pmatrix}\mathbf{B}_{X,b} & \mathbf{B}_{Y,b}\end{pmatrix}\in\mathbb{K}^{\hat{s}\times 2k}$$

erhalten wir so die Darstellung

$$(\mathbf{X}+\mathbf{Y})|_{\hat{t}\times\hat{s}} = \mathbf{A}_{X+Y,b}\mathbf{B}_{X+Y,b}^*,$$

```
procedure add_rk_rk(C, D, var A, B);
A ← (A  C);
B ← (B  D);
trunc_rk(k, A, B)
```

Abbildung 6.4: Effiziente gekürzte Addition zweier Matrizen $\mathbf{AB}^*$ und $\mathbf{CD}^*$ in Rang-k-Darstellung.

```
procedure add_h_h(b, Y, var X);
if b ∈ L⁺_{I×J} then
  add_rk_rk(A_{Y,b}, B_{Y,b}, A_{X,b}, B_{X,b})
else if b ∈ L⁻_{I×J} then
  X|_{t̂×ŝ} ← X|_{t̂×ŝ} + Y|_{t̂×ŝ}
else
  for b' ∈ sons(b) do
    add_h_h(b', Y, X)
  end for
end if
```

Abbildung 6.5: Effiziente gekürzte Addition zweier hierarchischer Matrizen.

also eine faktorisierte Rang-$2k$-Darstellung der Teilmatrix, die zu dem zulässigen Blatt $b = (t, s)$ gehört.

Offenbar kann der Rang dieser Summe bis zu $2k$ betragen, und das ist mehr, als wir für unsere Approximation der Summe vorgesehen haben. Deshalb bietet es sich an, die Rang-$2k$-Darstellung zu kürzen, um den gewünschten lokalen Rang k zu erreichen. Wie wir bereits gesehen haben erfordert diese Aufgabe höchstens $C_{\text{svdk}}(\#\hat{t} + \#\hat{s})(2k)^2 = 4C_{\text{svdk}}(\#\hat{t} + \#\hat{s})k^2$ Operationen, lässt sich also relativ effizient durchführen.

Es stellt sich natürlich die Frage, ob wir durch das Kürzen wesentliche Informationen verlieren. Am Beispiel der Integraloperatoren sieht man, dass das nicht der Fall sein muss: Falls $\mathbf{X}$ und $\mathbf{Y}$ aus der Diskretisierung asymptotisch glatter Kernfunktionen entstanden sind, wird auch die Summe $\mathbf{X} + \mathbf{Y}$ diese Eigenschaft teilen. Insofern wissen wir bereits, dass die Summe sich blockweise gut durch denselben Rang approximieren lassen wird, den wir auch für die Summanden verwendet haben. Aus den Lemmas 6.7 und 6.11 wissen wir auch, dass unser Algorithmus die bestmögliche Approximation berechnet, also folgt aus der bereits festgestellten guten Approximierbarkeit der Summe, dass auch unser Algorithmus eine gute Approximation finden wird. Diejenigen Singulärwerte, die wir durch das Kürzen eliminieren, werden deshalb solche sein, die lediglich den Fehler beschreiben, den wir bei der Approximation der Matrizen $\mathbf{X}$ und $\mathbf{Y}$ in Kauf genommen haben, so dass wir nicht viel verlieren, wenn wir auf sie verzichten.

In der beschriebenen Weise können wir jede Teilmatrix der Summe einzeln approximieren und erhalten so eine hierarchische Matrix, die $\mathbf{X} + \mathbf{Y}$ approximiert. Natürlich stellt sich die Frage nach dem für diese Berechnung erforderlichen Aufwand, die sich

mit dem uns bereits vertrauten Konzept des schwachbesetzten Blockbaums beantworten lässt:

Lemma 6.12 (Aufwand der Matrixaddition) *Seien $\mathbf{X}, \mathbf{Y} \in \mathbb{K}^{\mathcal{I}\times\mathcal{J}}$ hierarchische Matrizen mit lokalem Rang k für den Blockbaum $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$. Sei $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ zulässig und C_{sp}-schwachbesetzt, und sei p die Tiefe dieses Blockbaums. Seien $r_{\mathcal{I}}$ und $r_{\mathcal{J}}$ die Auflösungen der Clusterbäume $\mathcal{T}_{\mathcal{I}}$ und $\mathcal{T}_{\mathcal{J}}$. Dann können wir die blockweise Bestapproximation der Summe der Matrizen mit nicht mehr als*

$$C_{sp}\max\{4C_{svdk}k^2, r_{\mathcal{I}}, r_{\mathcal{J}}\}(p+1)(\#\mathcal{I}+\#\mathcal{J})$$

Operationen berechnen.

Beweis. Übungsaufgabe ∎

6.5 Multiplikation hierarchischer Matrizen

Wie wir gesehen haben, lässt sich die Addition zweier hierarchischer Matrizen relativ einfach auf blockweise Additionen zurückführen, denn es gilt

$$(\mathbf{X}+\mathbf{Y})|_{\hat{t}\times\hat{s}} = \mathbf{X}|_{\hat{t}\times\hat{s}} + \mathbf{Y}|_{\hat{t}\times\hat{s}}.$$

Für die Multiplikation zweier Matrizen $\mathbf{X} \in \mathbb{K}^{\mathcal{I}\times\mathcal{J}}$ und $\mathbf{Y} \in \mathbb{K}^{\mathcal{J}\times\mathcal{K}}$ ist die Situation komplizierter: Ein einzelner Eintrag des Produkts ergibt sich aus

$$(\mathbf{XY})_{ik} = \sum_{j\in\mathcal{J}} X_{ij}Y_{jk}, \tag{6.6}$$

und entsprechend besitzt eine Teilmatrix des Produkts die Gestalt

$$(\mathbf{XY})|_{\hat{t}\times\hat{r}} = \mathbf{X}|_{\hat{t}\times\mathcal{J}}\mathbf{Y}|_{\mathcal{J}\times\hat{r}}.$$

In typischen Anwendungsfällen werden die Streifen $\hat{t}\times\mathcal{J}$ und $\mathcal{J}\times\hat{r}$ nicht mit Blättern des Blockbaums übereinstimmen, wir müssen stattdessen davon ausgehen, dass Blätter auf verschiedenen Stufen des Blockbaums miteinander interagieren. Dadurch wird es *wesentlich* schwieriger, eine Approximation des Produkts zu berechnen, und eine Bestapproximation, wie im Falle der Addition, ist nur mit erheblichem Aufwand zu konstruieren und deshalb für praktische Anwendungen eher uninteressant.

Wir konzentrieren uns deshalb darauf, einen Algorithmus zu finden, der möglichst schnell arbeitet, auch wenn wir dabei auf die Bestapproximationseigenschaft verzichten müssen. Dabei gehen wir davon aus, dass $\mathbf{X} \in \mathbb{K}^{\mathcal{I}\times\mathcal{J}}$ eine hierarchische Matrix für den Blockbaum $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$, $\mathbf{Y} \in \mathbb{K}^{\mathcal{J}\times\mathcal{K}}$ eine für den Baum $\mathcal{T}_{\mathcal{J}\times\mathcal{K}}$, und $\mathbf{Z} \in \mathbb{K}^{\mathcal{I}\times\mathcal{K}}$ eine für den Baum $\mathcal{T}_{\mathcal{I}\times\mathcal{K}}$ ist. Wir stellen uns die Aufgabe, die Operation

$$\mathbf{Z}|_{\hat{t}\times\hat{r}} \leftarrow \mathbf{Z}|_{\hat{t}\times\hat{r}} + \mathbf{X}|_{\hat{t}\times\hat{s}}\mathbf{Y}|_{\hat{s}\times\hat{r}}$$

für Cluster $t \in \mathcal{T}_{\mathcal{I}}$, $s \in \mathcal{T}_{\mathcal{J}}$ und $r \in \mathcal{T}_{\mathcal{K}}$ möglichst effizient durchzuführen, für zulässige Blätter (t, r) selbstverständlich wieder approximativ.

Fall 1: (t, s) ist ein zulässiges Blatt

Wir untersuchen zunächst den Spezialfall, dass $(t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ gilt, dass wir also mit einem zulässigen Blatt des Blockbaums $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ arbeiten. Da $\mathbf{X}$ eine hierarchische Matrix ist, stehen uns dann Matrizen $\mathbf{A} \in \mathbb{K}^{\hat{t}\times k}$ und $\mathbf{B} \in \mathbb{K}^{\hat{s}\times k}$ zur Verfügung, die

$$\mathbf{X}|_{\hat{t}\times\hat{s}} = \mathbf{A}\mathbf{B}^*$$

erfüllen. Daraus folgt

$$\mathbf{X}|_{\hat{t}\times\hat{s}}\mathbf{Y}|_{\hat{s}\times\hat{r}} = \mathbf{A}\mathbf{B}^*\mathbf{Y}|_{\hat{s}\times\hat{r}} = \mathbf{A}(\mathbf{Y}|^*_{\hat{s}\times\hat{r}}\mathbf{B})^* = \mathbf{A}\widehat{\mathbf{B}}^*$$

mit der Matrix

$$\widehat{\mathbf{B}} := \mathbf{Y}|^*_{\hat{s}\times\hat{r}}\mathbf{B},$$

die sich einfach berechnen lässt, indem man die Teilmatrix $\mathbf{Y}|^*_{\hat{s}\times\hat{r}}$ mit den k Spalten der Matrix $\mathbf{B}$ multipliziert und so die Spalten der gesuchten Matrix $\widehat{\mathbf{B}}$ erhält. Wie wir bereits gesehen haben, kann der in Abbildung 6.2 dargestellte Algorithmus diese Multiplikationen effizient durchführen.

Damit reduziert sich unsere Aufgabe darauf, die Berechnung

$$\mathbf{Z}|_{\hat{t}\times\hat{r}} \leftarrow \mathbf{Z}|_{\hat{t}\times\hat{r}} + \mathbf{A}\widehat{\mathbf{B}}^*$$

durchzuführen. Falls (t, r) ein Blatt des betreffenden Baums $\mathcal{T}_{\mathcal{I}\times\mathcal{K}}$ ist, lässt sich diese Aufgabe einfach durchführen: Bei zulässigen Blättern durch die gekürzte Addition, bei unzulässigen Blättern direkt.

Anderenfalls gehen wir zu den Söhnen (t', r') des Knoten (t, r) über und nutzen aus, dass auch Teilmatrizen der Rang-k-Matrix $\mathbf{A}\widehat{\mathbf{B}}^*$ wieder Rang-k-Matrizen mit der Darstellung

$$(\mathbf{A}\widehat{\mathbf{B}}^*)|_{\hat{t}'\times\hat{r}'} = \mathbf{A}|_{\hat{t}'\times k}\widehat{\mathbf{B}}|^*_{\hat{r}'\times k}$$

sind. In dieser Weise können wir rekursiv fortfahren, bis die Blätter des Baums $\mathcal{T}_{\mathcal{I}\times\mathcal{K}}$ erreicht sind. Der entsprechende Algorithmus ist in Abbildung 6.6 zusammengefasst.

Fall 2: (s, r) ist ein zulässiges Blatt

Nun untersuchen wir den Fall, dass $(s, r) \in \mathcal{L}^+_{\mathcal{J}\times\mathcal{K}}$ gilt, dass also ein zulässiges Blatt des Blockbaums $\mathcal{T}_{\mathcal{J}\times\mathcal{K}}$ vorliegt. Diesmal können wir ausnutzen, dass $\mathbf{Y}$ eine hierarchische Matrix ist und uns deshalb Matrizen $\mathbf{A} \in \mathbb{K}^{\hat{s}\times k}$ und $\mathbf{B} \in \mathbb{K}^{\hat{r}\times k}$ zur Verfügung stehen, für die

$$\mathbf{Y}|_{\hat{s}\times\hat{r}} = \mathbf{A}\mathbf{B}^*$$

gilt. Wie im vorangehenden Fall können wir auch hier das Produkt zweier Teilmatrizen effizient darstellen: Wir erhalten

$$\mathbf{X}|_{\hat{t}\times\hat{s}}\mathbf{Y}|_{\hat{s}\times\hat{r}} = \mathbf{X}|_{\hat{t}\times\hat{s}}\mathbf{A}\mathbf{B}^* = (\mathbf{X}|_{\hat{t}\times\hat{s}}\mathbf{A})\mathbf{B}^* = \widehat{\mathbf{A}}\mathbf{B}^*$$

mit der Matrix

$$\widehat{\mathbf{A}} = \mathbf{X}|_{\hat{t}\times\hat{s}}\mathbf{A},$$

```
procedure add_rk_h(b, Â, B̂, var Z);
(t, r) ← b;
if (t, r) ∈ L⁺_{I×K} then
  add_rk_rk(Â, B̂, A_{Z,b}, B_{Z,b})
else if (t, r) ∈ L⁻_{I×K} then
  Z|_{t̂×r̂} ← Z|_{t̂×r̂} + ÂB̂*
else
  for b' ∈ sons(b) do
    (t', r') ← b';
    add_rk_h(b', Â|_{t̂'×k}, B̂|_{r̂'×k}, Z)
  end for
end if
```

Abbildung 6.6: Addition einer Rang-k-Matrix zu einer $\mathcal{H}$-Matrix: $\mathbf{Z} \leftarrow \mathbf{Z} + \widehat{\mathbf{A}}\widehat{\mathbf{B}}^*$

die sich effizient bestimmen lässt, indem wir die Teilmatrix $\mathbf{X}|_{\hat{t}\times\hat{s}}$ mit den k Spalten der Matrix $\mathbf{A}$ multiplizieren und damit die Spalten der Matrix $\widehat{\mathbf{A}}$ berechnen. Dazu kann der Algorithmus aus Abbildung 6.1 verwendet werden.

Mit Hilfe dieses Zwischenergebnisses können wir nun die Berechnung

$$\mathbf{Z}|_{\hat{t}\times\hat{r}} \leftarrow \mathbf{Z}|_{\hat{t}\times\hat{r}} + \widehat{\mathbf{A}}\mathbf{B}^*$$

wie bereits diskutiert rekursiv durchführen.

Fall 3: (t, s) oder (s, r) ist ein unzulässiges Blatt

Unzulässige Blätter können wir bei Bedarf wie Rang-k-Matrizen behandeln: Falls etwa (t, s) ein unzulässiges Blatt ist, muss nach Definition 2.9 t oder s ein Blatt sein, also können wir davon ausgehen, dass $\#\hat{t} \leq r_{\mathcal{I}}$ oder $\#\hat{s} \leq r_{\mathcal{J}}$ gilt.

Im ersten Fall können wir $\mathbf{X}|_{\hat{t}\times\hat{s}}$ als Rang-$r_{\mathcal{I}}$-Matrix schreiben:

$$\mathbf{X}|_{\hat{t}\times\hat{s}} = \mathbf{I}(\mathbf{X}|^*_{\hat{t}\times\hat{s}})^*$$

gilt mit der Einheitsmatrix aus $\mathbb{K}^{\hat{t}\times\hat{t}}$, und wir erhalten

$$\mathbf{X}|_{\hat{t}\times\hat{s}}\mathbf{Y}|_{\hat{s}\times\hat{r}} = \mathbf{I}(\mathbf{X}|^*_{\hat{t}\times\hat{s}})^*\mathbf{Y}|_{\hat{s}\times\hat{r}} = \mathbf{I}(\mathbf{Y}|^*_{\hat{s}\times\hat{r}}\mathbf{X}|^*_{\hat{t}\times\hat{s}})^* = \widehat{\mathbf{A}}\widehat{\mathbf{B}}^*$$

mit den Hilfsmatrizen

$$\widehat{\mathbf{A}} = \mathbf{I}, \qquad \widehat{\mathbf{B}} = \mathbf{Y}|^*_{\hat{s}\times\hat{r}}\mathbf{X}|^*_{\hat{t}\times\hat{s}},$$

so dass wir wie im ersten und zweiten Fall verfahren können.

Falls dagegen s ein Blatt ist und deshalb $\#\hat{s} \leq r_{\mathcal{J}}$ gilt, ist

$$\mathbf{X}|_{\hat{t}\times\hat{s}}\mathbf{Y}|_{\hat{s}\times\hat{r}} = \widehat{\mathbf{A}}\widehat{\mathbf{B}}^*$$

mit den Matrizen

$$\widehat{\mathbf{A}} = \mathbf{X}|_{\hat{t}\times\hat{s}}, \qquad \widehat{\mathbf{B}} = \mathbf{Y}|^{*}_{\hat{s}\times\hat{r}}$$

bereits eine Rang-$r_{\mathcal{J}}$-Darstellung des Produkts, die wir direkt verwenden können.

Fall 4: (t,s) und (s,r) sind keine Blätter

In diesem Fall bleibt uns keine andere Wahl, als rekursiv vorzugehen: Falls beispielsweise

$$\operatorname{sons}(t) = \{t_1, t_2\}, \qquad \operatorname{sons}(s) = \{s_1, s_2\}, \qquad \operatorname{sons}(r) = \{r_1, r_2\}$$

gelten, können wir die Matrizen

$$\mathbf{X}_{ij} := \mathbf{X}|_{\hat{t}_i\times\hat{s}_j}, \qquad \mathbf{Y}_{ij} := \mathbf{Y}|_{\hat{s}_i\times\hat{r}_j}, \qquad \mathbf{Z}_{ij} := \mathbf{Z}|_{\hat{t}_i\times\hat{r}_j} \qquad \text{für alle } i,j \in \{1,2\}$$

einführen und erhalten

$$\begin{aligned}\mathbf{Z}|_{\hat{t}\times\hat{r}} + \mathbf{X}|_{\hat{t}\times\hat{s}}\mathbf{Y}|_{\hat{s}\times\hat{r}} &= \begin{pmatrix}\mathbf{Z}_{11} & \mathbf{Z}_{12}\\ \mathbf{Z}_{21} & \mathbf{Z}_{22}\end{pmatrix} + \begin{pmatrix}\mathbf{X}_{11} & \mathbf{X}_{12}\\ \mathbf{X}_{21} & \mathbf{X}_{22}\end{pmatrix}\begin{pmatrix}\mathbf{Y}_{11} & \mathbf{Y}_{12}\\ \mathbf{Y}_{21} & \mathbf{Y}_{22}\end{pmatrix}\\ &= \begin{pmatrix}\mathbf{Z}_{11} + \mathbf{X}_{11}\mathbf{Y}_{11} + \mathbf{X}_{12}\mathbf{Y}_{21} & \mathbf{Z}_{12} + \mathbf{X}_{11}\mathbf{Y}_{12} + \mathbf{X}_{12}\mathbf{Y}_{22}\\ \mathbf{Z}_{21} + \mathbf{X}_{21}\mathbf{Y}_{11} + \mathbf{X}_{22}\mathbf{Y}_{21} & \mathbf{Z}_{22} + \mathbf{X}_{21}\mathbf{Y}_{12} + \mathbf{X}_{22}\mathbf{Y}_{22}\end{pmatrix}.\end{aligned}$$

Diese Blockmatrix lässt sich mit acht Teilschritten berechnen:

$$\begin{aligned}\mathbf{Z}_{11} &\leftarrow \mathbf{Z}_{11} + \mathbf{X}_{11}\mathbf{Y}_{11}, & \mathbf{Z}_{11} &\leftarrow \mathbf{Z}_{11} + \mathbf{X}_{12}\mathbf{Y}_{21},\\ \mathbf{Z}_{12} &\leftarrow \mathbf{Z}_{12} + \mathbf{X}_{11}\mathbf{Y}_{12}, & \mathbf{Z}_{12} &\leftarrow \mathbf{Z}_{12} + \mathbf{X}_{12}\mathbf{Y}_{22},\\ \mathbf{Z}_{21} &\leftarrow \mathbf{Z}_{21} + \mathbf{X}_{21}\mathbf{Y}_{11}, & \mathbf{Z}_{21} &\leftarrow \mathbf{Z}_{21} + \mathbf{X}_{22}\mathbf{Y}_{21},\\ \mathbf{Z}_{22} &\leftarrow \mathbf{Z}_{22} + \mathbf{X}_{21}\mathbf{Y}_{12}, & \mathbf{Z}_{22} &\leftarrow \mathbf{Z}_{22} + \mathbf{X}_{22}\mathbf{Y}_{22},\end{aligned}$$

und jeder dieser Teilschritte besitzt die Form der ursprünglich auszuführenden Operation, so dass wir direkt mit einer Rekursion arbeiten können.

Um den allgemeinen Fall einfach darstellen zu können, führen wir

$$\operatorname{sons}^+(t) := \begin{cases}\operatorname{sons}(t) & \text{falls } \operatorname{sons}(t) \neq \emptyset,\\ \{t\} & \text{ansonsten,}\end{cases} \qquad \operatorname{sons}^+(s) := \begin{cases}\operatorname{sons}(s) & \text{falls } \operatorname{sons}(s) \neq \emptyset,\\ \{s\} & \text{ansonsten,}\end{cases}$$

$$\operatorname{sons}^+(r) := \begin{cases}\operatorname{sons}(r) & \text{falls } \operatorname{sons}(r) \neq \emptyset,\\ \{r\} & \text{ansonsten,}\end{cases}$$

ein, so dass sich die Söhne der beteiligten Blöcke einfach als

$$\operatorname{sons}(t,s) = \operatorname{sons}^+(t) \times \operatorname{sons}^+(s), \qquad \operatorname{sons}(s,r) = \operatorname{sons}^+(s) \times \operatorname{sons}^+(r),$$
$$\operatorname{sons}(t,r) = \operatorname{sons}^+(t) \times \operatorname{sons}^+(r)$$

schreiben lassen und die Operation

$$\mathbf{Z}|_{\hat{t}\times\hat{r}} \leftarrow \mathbf{Z}|_{\hat{t}\times\hat{r}} + \mathbf{X}|_{\hat{t}\times\hat{s}}\mathbf{Y}|_{\hat{s}\times\hat{r}}$$

sich auch in der Form

$$\mathbf{Z}|_{\hat{t}'\times\hat{r}'} \leftarrow \mathbf{Z}|_{\hat{t}'\times\hat{r}'} + \mathbf{X}|_{\hat{t}'\times\hat{s}'}\mathbf{Y}|_{\hat{s}'\times\hat{r}'} \qquad \text{für alle } t' \in \operatorname{sons}^+(t),\ s' \in \operatorname{sons}^+(s),\ r' \in \operatorname{sons}^+(r)$$

ausführen lässt. Das ist die verallgemeinerte Form des rekursiven Ansatzes.

Leider besitzt dieser allgemeine Zugang einen kleinen Nachteil: Es könnte passieren, dass zwar (t, s) und (s, r) keine Blätter der Bäume $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und $\mathcal{T}_{\mathcal{J}\times\mathcal{K}}$ sind, dass aber (t, r) sehr wohl ein Blatt des Baums $\mathcal{T}_{\mathcal{I}\times\mathcal{K}}$ ist. Falls es sich um ein unzulässiges Blatt handeln sollte, entsteht dabei kein Problem, denn dann können wir Teilmatrizen direkt ansprechen.

Schwieriger ist die Situation bei einem zulässigen Blatt, denn eine Matrix, deren Teilmatrizen von Rang k sind, muss insgesamt keineswegs denselben Rang besitzen. Deshalb führen wir in diesem Fall Hilfsmatrizen

$$\widehat{\mathbf{A}}_{t',r'} \in \mathbb{K}^{\hat{t}'\times k}, \qquad \widehat{\mathbf{B}}_{t',r'} \in \mathbb{K}^{\hat{r}'\times k} \qquad \text{für alle } t' \in \operatorname{sons}^+(t),\ r' \in \operatorname{sons}^+(r)$$

ein, die die Rang-k-Darstellungen der Ergebnisse der Rekursion aufnehmen.

Sobald alle Hilfsmatrizen gefüllt sind, können wir sie geeignet mit Nullen fortsetzen und zu dem Ergebnis $\mathbf{Z}|_{\hat{t}\times\hat{r}}$ hinzuaddieren, selbstverständlich wieder approximativ, um die gesuchte Rang-k-Darstellung zu erhalten.

Damit sind alle möglichen Fälle abgehandelt. Eine Übersicht über den soeben hergeleiteten Algorithmus findet sich in Abbildung 6.7.

Es stellt sich natürlich die Frage nach der Genauigkeit, die dieser Ansatz erreichen kann: An verschiedenen Stellen werden Rang-k-Matrizen addiert, und dabei wird jeweils gekürzt, also können sich Fehler ansammeln.

Anders als bei der Addition dürfen wir bei dem beschriebenen Algorithmus zur Approximation des Matrixprodukts nicht darauf hoffen, die Bestapproximation zu erhalten: Es treten Summen mit potentiell vielen Termen auf, die nicht insgesamt durch niedrigen Rang approximiert werden, sondern mit Hilfe einer Folge von einzelnen gekürzten Additionen. Dadurch kann sich das Ergebnis von der Bestapproximation entfernen, allerdings nicht allzu weit (vgl. [19, Satz 4.7]).

Immerhin können wir bei adaptiver Wahl des Rangs k garantieren, dass auch bei langen Summen eine vorgegebene Genauigkeit erreicht wird, indem wir mit Hilfe der Dreiecksungleichung ermitteln, wie genau die einzelnen Teilsummen berechnet werden müssen, um eine hinreichend gute Näherung der Gesamtsumme zu erhalten.

Bemerkung 6.13 (Bestapproximation) *Es ist auch möglich, einen Multiplikationsalgorithmus zu entwerfen, der die Bestapproximation berechnet, allerdings ist dieser Algorithmus sehr aufwendig: Bevor ein zulässiger Block berechnet wird, wird eine Liste aller Blockpaare aufgestellt, die etwas zu diesem Block beitragen. Anhand dieser Liste kann dann das Ergebnis als Summe vieler Teilmatrizen dargestellt und direkt mit Hilfe eines entsprechend modifizierten Kürzungsalgorithmus behandelt werden. Der Rechenaufwand (und auch der Implementierungsaufwand) dieser Variante ist sehr hoch, und da wir in*

procedure mul_h_h(α, t, s, r, $\mathbf{X}$, $\mathbf{Y}$, **var** $\mathbf{Z}$);
$b_X \leftarrow (t, s)$; $b_Y \leftarrow (s, r)$; $b_Z \leftarrow (t, r)$;
if $b_X \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ **then**
  $\widehat{\mathbf{B}} \leftarrow \mathbf{0} \in \mathbb{K}^{\hat{r}\times k}$; adjeval_h($\bar{\alpha}$, b_Y, $\mathbf{Y}$, $\mathbf{B}_{X,b_X}$, $\widehat{\mathbf{B}}$); {für Spaltenvektoren}
  add_rk_h(b_Z, $\mathbf{A}_{X,b_X}$, $\widehat{\mathbf{B}}$, $\mathbf{Z}$)
else if $b_Y \in \mathcal{L}^+_{\mathcal{J}\times\mathcal{K}}$ **then**
  $\widehat{\mathbf{A}} \leftarrow \mathbf{0} \in \mathbb{K}^{\hat{t}\times k}$; eval_h($\alpha$, b_X, $\mathbf{X}$, $\mathbf{A}_{Y,b_Y}$, $\widehat{\mathbf{A}}$); {für Spaltenvektoren}
  add_rk_h(b_Z, $\widehat{\mathbf{A}}$, $\mathbf{B}_{Y,b_Y}$, $\mathbf{Z}$)
else if $b_X \in \mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}$ **then**
  if $\#\hat{t} \leq \#\hat{s}$ **then**
    $\widehat{\mathbf{A}} \leftarrow \mathbf{I}_{\hat{t}}$; $\widehat{\mathbf{B}} \leftarrow \mathbf{0} \in \mathbb{K}^{\hat{r}\times\hat{t}}$; adjeval_h($\bar{\alpha}$, b_Y, $\mathbf{Y}$, $\mathbf{X}|^*_{\hat{t}\times\hat{s}}$, $\widehat{\mathbf{B}}$)
  else
    $\widehat{\mathbf{A}} \leftarrow \mathbf{X}|_{\hat{t}\times\hat{s}}$; $\widehat{\mathbf{B}} \leftarrow \mathbf{0} \in \mathbb{K}^{\hat{r}\times\hat{s}}$; adjeval_h($\bar{\alpha}$, b_Y, $\mathbf{Y}$, $\mathbf{I}_{\hat{s}}$, $\widehat{\mathbf{B}}$)
  end if;
  add_rk_h(b_Z, $\widehat{\mathbf{A}}$, $\widehat{\mathbf{B}}$, $\mathbf{Z}$)
else if $b_Y \in \mathcal{L}^-_{\mathcal{J}\times\mathcal{K}}$ **then**
  if $\#\hat{s} \leq \#\hat{r}$ **then**
    $\widehat{\mathbf{A}} \leftarrow \mathbf{0} \in \mathbb{K}^{\hat{t}\times\hat{s}}$; eval_h($\alpha$, b_X, $\mathbf{X}$, $\mathbf{I}_{\hat{s}}$, $\widehat{\mathbf{A}}$); $\widehat{\mathbf{B}} \leftarrow \mathbf{Y}|^*_{\hat{s}\times\hat{r}}$
  else
    $\widehat{\mathbf{A}} \leftarrow \mathbf{0} \in \mathbb{K}^{\hat{t}\times\hat{r}}$; eval_h($\alpha$, b_X, $\mathbf{X}$, $\mathbf{Y}|_{\hat{s}\times\hat{r}}$, $\widehat{\mathbf{A}}$); $\widehat{\mathbf{B}} \leftarrow \mathbf{I} \in \mathbb{K}^{\hat{r}\times\hat{r}}$
  end if;
  add_rk_h(b_Z, $\widehat{\mathbf{A}}$, $\widehat{\mathbf{B}}$, $\mathbf{Z}$)
else
  for $t' \in \mathrm{sons}^+(t)$, $r' \in \mathrm{sons}^+(r)$ **do**
    if $b_Z \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{K}}$ oder $b_Z \notin \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ **then**
      $b' \leftarrow (t', r')$; $\mathbf{A}_{Z,b'} \leftarrow \mathbf{0} \in \mathbb{K}^{t'\times k}$; $\mathbf{B}_{Z,b'} \leftarrow \mathbf{0} \in \mathbb{K}^{\hat{r}'\times k}$
    end if;
    for $s' \in \mathrm{sons}^+(s)$ **do**
      mul_h_h(α, t', s', r', $\mathbf{X}$, $\mathbf{Y}$, $\mathbf{Z}$)
    end for;
    if $b_Z \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{K}}$ oder $b_Z \notin \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ **then**
      add_rk_rk($\mathbf{A}_{Z,b'}$, $\mathbf{B}_{Z,b'}$, $\mathbf{A}_{Z,b_Z}$, $\mathbf{B}_{Z,b_Z}$)
    end if
  end for
end if

Abbildung 6.7: Multiplikation zweier $\mathcal{H}$-Matrizen: $\mathbf{Z} \leftarrow \mathbf{Z} + \alpha\mathbf{X}\mathbf{Y}$

der Regel die Multiplikation nur als Teil einer größeren Berechnung einsetzen, die ohnehin nicht mehr die Bestapproximation bestimmen wird, verzichten wir hier darauf, sie weiter zu analysieren.

Neben der Genauigkeit interessiert uns natürlich auch die Effizienz des vorgestellten

Algorithmus zur Matrixmultiplikation. Zur Vereinfachung der Aufwandsabschätzung setzen wir

$$\hat{k} := \max\{k, r_{\mathcal{I}}, r_{\mathcal{J}}, r_{\mathcal{K}}\}, \qquad \hat{p} := \max\{\operatorname{depth}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}), \operatorname{depth}(\mathcal{T}_{\mathcal{J}\times\mathcal{K}}), \operatorname{depth}(\mathcal{T}_{\mathcal{I}\times\mathcal{K}})\}$$

und nehmen an, dass $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$, $\mathcal{T}_{\mathcal{J}\times\mathcal{K}}$ und $\mathcal{T}_{\mathcal{I}\times\mathcal{K}}$ C_{sp}-schwachbesetzt und zulässig sind. Außerdem nehmen wir an, dass es eine Konstante C_{sn} gibt, für die

$$\#\operatorname{sons}(t) \leq C_{\mathrm{sn}}, \quad \#\operatorname{sons}(s) \leq C_{\mathrm{sn}}, \quad \#\operatorname{sons}(r) \leq C_{\mathrm{sn}} \quad \text{für alle } t \in \mathcal{T}_{\mathcal{I}},\ s \in \mathcal{T}_{\mathcal{J}},\ r \in \mathcal{T}_{\mathcal{K}}$$

gilt. Diese Annahme ist in der Regel einfach zu erfüllen, indem wir die Konstruktion der Clusterbäume geeignet organisieren.

Die Analyse des Aufwands zerlegen wir in drei Teile: Wir schätzen die Anzahl der Rechenoperationen für den Fall ab, dass b_X oder b_Y ein Blatt ist, wir schätzen die Anzahl der Operationen mit den Hilfsmatrizen $\mathbf{A}_{Z,b'}$ und $\mathbf{B}_{Z,b'}$ ab, und wir schätzen ab, wieviele Operationen für ein Clustertripel (t, s, r) anfallen.

Für die Blätter benötigen wir zunächst eine Aussage über die Addition einer Rang-k-Darstellung zu einer hierarchischen Matrix:

Lemma 6.14 ($\mathcal{H}$-Matrix plus Rang k) *Sei $b_0 = (t_0, s_0) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$. Der in Abbildung 6.6 dargestellte Algorithmus benötigt, aufgerufen mit $b = b_0$, nicht mehr als*

$$C_{addrkh}\hat{k}^2(\hat{p} + 1 - \operatorname{level}(b_0))(\#\hat{t}_0 + \#\hat{s}_0)$$

Operationen, wobei die Konstante durch

$$C_{addrkh} := C_{sp} \max\{4C_{svdk}, 2\}$$

gegeben ist.

Beweis. Solange der Algorithmus sich rekursiv in Richtung der Blätter des Blockbaums $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ vorarbeitet, fallen keine Rechenoperationen an.

Sobald er ein zulässiges Blatt $(t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ erreicht, wird eine gekürzte Addition durchgeführt, die wegen Lemma 6.5 nicht mehr als

$$4C_{\mathrm{svdk}}(\#\hat{t} + \#\hat{s})\hat{k}^2$$

Operationen erfordert.

Für ein unzulässiges Blatt $(t, s) \in \mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}$ dagegen wird das Produkt $\widehat{\mathbf{A}}|_{\hat{t}\times k}\widehat{\mathbf{B}}|^*_{\hat{s}\times k}$ in $(\#\hat{t})(\#\hat{s})(2k-1)$ Operationen berechnet. Da $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ zulässig ist, muss t oder s ein Blatt sein, also muss $\#\hat{t} \leq r_{\mathcal{I}} \leq \hat{k}$ oder $\#\hat{s} \leq r_{\mathcal{J}} \leq \hat{k}$ gelten, so dass wir eine Schranke von

$$2(\#\hat{t} + \#\hat{s})\hat{k}^2$$

erhalten.

Der Gesamtaufwand ist damit beschränkt durch

$$\sum_{b=(t,s)\in\operatorname{sons}^*(b_0)} \max\{4C_{\mathrm{svdk}}, 2\}(\#\hat{t} + \#\hat{s})k^2,$$

und wir erhalten dank Lemma 2.28

$$\sum_{b=(t,s)\in\operatorname{sons}^*(b_0)} \#\hat{t} \leq C_{\mathrm{sp}}(\hat{p}+1-\operatorname{level}(b_0))\#\hat{t}_0,$$
$$\sum_{b=(t,s)\in\operatorname{sons}^*(b_0)} \#\hat{s} \leq C_{\mathrm{sp}}(\hat{p}+1-\operatorname{level}(b_0))\#\hat{s}_0,$$

also die angestrebte Abschätzung. ∎

Indem wir diese Abschätzung des Aufwands des Algorithmus für das „Verteilen“ einer Rang-k-Matrix auf die Blätter eines Teil-Blockbaums mit einer Abschätzung für den Aufwand der Auswertung einer hierarchischen Matrix kombinieren, erhalten wir eine Schranke für den Rechenaufwand in Situationen, in denen b_X oder b_Y ein Blatt ist:

Lemma 6.15 (Aufwand für Blätter) *Falls in dem Algorithmus aus Abbildung 6.7 $b_X \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ oder $b_Y \in \mathcal{L}_{\mathcal{J}\times\mathcal{K}}$ gelten, benötigt er nicht mehr als*

$$C_{mul,lf}\hat{k}^2(\hat{p}+1)(\#\hat{t}+\#\hat{s}+\#\hat{r})$$

Operationen, wobei die Konstante durch

$$C_{mul,lf} := 2C_{sp} + C_{addrkh}$$

gegeben ist.

Beweis. Sei $b_X \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ oder $b_Y \in \mathcal{L}_{\mathcal{J}\times\mathcal{K}}$. Dann berechnet unser Algorithmus die Hilfsmatrizen $\widehat{\mathbf{A}}$ und $\widehat{\mathbf{B}}$ im Fall eines zulässigen Blatts mit k Matrix-Vektor-Multiplikationen, im Fall eines unzulässigen Blatts mit $r_{\mathcal{I}}$ oder $r_{\mathcal{J}}$ solchen Operationen, so dass nach Lemma 6.1 insgesamt nicht mehr als

$$2C_{\mathrm{sp}}\hat{k}^2(\hat{p}+1)(\#\hat{t}+\#\hat{s}+\#\hat{r})$$

Operationen anfallen. Die Hilfsmatrizen werden dann mit Hilfe des Algorithmus aus Abbildung 6.6 auf die Matrix $\mathbf{Z}|_{\hat{t}\times\hat{r}}$ verteilt, wofür nach Lemma 6.14 nicht mehr als

$$C_{\mathrm{addrkh}}\hat{k}^2(\hat{p}+1)(\#\hat{t}+\#\hat{r})$$

Operationen nötig sind. Indem wir beide Schranken addieren erhalten wir das gewünschte Ergebnis. ∎

Wenden wir uns nun dem Aufwand zu, der für die Hilfsmatrizen $\mathbf{A}_{Z,b'}$ und $\mathbf{B}_{Z,b'}$ anfällt, die wir einsetzen, falls b_Z zulässig ist, es b_X und b_Y aber nicht sind.

Lemma 6.16 (Aufwand für Hilfsmatrizen) *Sei $b_Z = (t,r)$ mit $t \in \mathcal{T}_{\mathcal{I}}$ und $r \in \mathcal{T}_{\mathcal{K}}$. Das Aufaddieren der Hilfsmatrizen $\mathbf{A}_{Z,b'}\mathbf{B}_{Z,b'}$ für $b' \in \operatorname{sons}^+(t)\times\operatorname{sons}^+(r)$ erfordert insgesamt nicht mehr als*

$$C_{mul,aux}\hat{k}^2(\#\hat{t}+\#\hat{r})$$

Operationen, wobei die Konstante durch

$$C_{mul,aux} := 4C_{sn}^2C_{svdk}$$

gegeben ist.

Beweis. Nach Voraussetzung gelten $\#\operatorname{sons}^+(t) \leq C_{\mathrm{sn}}$ und $\#\operatorname{sons}^+(s) \leq C_{\mathrm{sn}}$, also müssen höchstens C_{sn}^2 Additionen ausgeführt werden.

Eine einzelne Addition wird wieder auf das Rang-$2k$-Kürzen zurückgeführt und benötigt deshalb nicht mehr als

$$4C_{\mathrm{svdk}}k^2(\#\hat{t} + \#\hat{s}) \leq 4C_{\mathrm{svdk}}\hat{k}^2(\#\hat{t} + \#\hat{s})$$

Operationen. Damit erhalten wir die angestrebte Abschätzung. ∎

Nun stehen uns Abschätzungen für den Aufwand sämtlicher Rechenoperationen innerhalb des Algorithmus zur Verfügung, es fehlt „nur“ noch die Abschätzung des gesamten rekursiven Algorithmus. Dieser Aufgabe nähern wir uns, indem wir die Tripel (t, s, r), für die der Algorithmus Berechnungen durchführt, in einem Baum zusammenfassen:

Definition 6.17 (Produktbaum) *Wir definieren $\mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}$ induktiv: Die Wurzel dieses Baums ist*

$$\operatorname{root}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}) = (\operatorname{root}(\mathcal{T}_{\mathcal{I}}), \operatorname{root}(\mathcal{T}_{\mathcal{J}}), \operatorname{root}(\mathcal{T}_{\mathcal{K}})),$$

und die Söhne seiner Knoten sind durch

$$\operatorname{sons}(t, s, r) = \begin{cases} \emptyset & \text{falls } (t, s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}} \\ & \quad \text{oder } (s, r) \in \mathcal{L}_{\mathcal{J}\times\mathcal{K}}, \\ \operatorname{sons}^+(t) \times \operatorname{sons}^+(s) \times \operatorname{sons}^+(r) & \text{ansonsten} \end{cases}$$

für alle $(t, s, r) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}$ gegeben. Diesen Baum bezeichnen wir als Produktbaum *zu den Clusterbäumen $\mathcal{T}_{\mathcal{I}}$, $\mathcal{T}_{\mathcal{J}}$ und $\mathcal{T}_{\mathcal{K}}$ sowie den Blockbäumen $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und $\mathcal{T}_{\mathcal{J}\times\mathcal{K}}$.*

Der so definierte Produktbaum beschreibt gerade die rekursiven Aufrufe, die der Algorithmus aus Abbildung 6.7 verwendet, um die Berechnung durchzuführen: Falls (t', s', r') ein Sohn des Tripels (t, s, r) ist, bedeutet das gerade, dass der Algorithmus, aufgerufen mit dem Tripel (t, s, r), sich selbst erneut mit dem Tripel (t', s', r') aufruft.

Die für uns entscheidende Eigenschaft des Produktbaums besteht darin, dass er, in einem geeignet verallgemeinerten Sinn, die schwachbesetzte Struktur der Blockbäume $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und $\mathcal{T}_{\mathcal{J}\times\mathcal{K}}$ erbt:

Lemma 6.18 (Schwachbesetzt) *Für alle $(t, s, r) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}$ gelten $(t, s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und $(s, r) \in \mathcal{T}_{\mathcal{J}\times\mathcal{K}}$. Daraus folgt*

$$\begin{aligned} \#\{(s, r) \in \mathcal{T}_{\mathcal{J}} \times \mathcal{T}_{\mathcal{K}} \;:\; (t, s, r) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}\} &\leq C_{sp}^2 && \text{für alle } t \in \mathcal{T}_{\mathcal{I}}, \\ \#\{(t, r) \in \mathcal{T}_{\mathcal{I}} \times \mathcal{T}_{\mathcal{K}} \;:\; (t, s, r) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}\} &\leq C_{sp}^2 && \text{für alle } s \in \mathcal{T}_{\mathcal{J}}, \\ \#\{(t, s) \in \mathcal{T}_{\mathcal{I}} \times \mathcal{T}_{\mathcal{J}} \;:\; (t, s, r) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}\} &\leq C_{sp}^2 && \text{für alle } r \in \mathcal{T}_{\mathcal{K}}. \end{aligned}$$

Beweis. Wir beweisen zunächst, dass für alle $(t, s, r) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}$ auch $(t, s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und $(s, r) \in \mathcal{T}_{\mathcal{J}\times\mathcal{K}}$ gelten.

Diesen Beweis führen wir induktiv über den Aufbau des Baums: Offenbar besitzt $\operatorname{root}(\mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}})$ die Eigenschaft, denn nach Definition 2.7 sind $(\operatorname{root}(\mathcal{T}_{\mathcal{I}}), \operatorname{root}(\mathcal{T}_{\mathcal{J}}))$ und $(\operatorname{root}(ctJ), \operatorname{root}(\mathcal{T}_{\mathcal{K}}))$ die Wurzeln der Blockbäume $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und $\mathcal{T}_{\mathcal{J}\times\mathcal{K}}$.

Sei nun $(t,s,r) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}$ mit $(t,s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und $(s,r) \in \mathcal{T}_{\mathcal{J}\times\mathcal{K}}$ gegeben. Falls $(t,s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}$ oder $(s,r) \in \mathcal{L}_{\mathcal{J}\times\mathcal{K}}$ gelten sollte, falls also einer der Blöcke ein Blatt sein sollte, besitzt (t,s,r) keine Söhne, also ist auch nichts zu beweisen.

Anderenfalls gelten nach Definition 2.7 gerade

$$\operatorname{sons}(t,s) = \operatorname{sons}^+(t) \times \operatorname{sons}^+(s), \qquad \operatorname{sons}(s,r) = \operatorname{sons}^+(s) \times \operatorname{sons}^+(r),$$

und damit folgt für jedes Tripel $(t',s',r') \in \operatorname{sons}(t,s,r)$ aus $t' \in \operatorname{sons}^+(t)$, $s' \in \operatorname{sons}^+(s)$ und $r' \in \operatorname{sons}^+(r)$ bereits $(t',s') \in \operatorname{sons}(t,s)$ sowie $(s',r') \in \operatorname{sons}(s,r)$ und somit insbesondere $(t',s') \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und $(s',r') \in \mathcal{T}_{\mathcal{J}\times\mathcal{K}}$. Damit ist die Induktion vollständig.

Sei nun $t \in \mathcal{T}_{\mathcal{I}}$. Für alle $(t,s,r) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}$ müssen dann $(t,s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ und $(s,r) \in \mathcal{T}_{\mathcal{J}\times\mathcal{K}}$ gelten, also $s \in \operatorname{row}(t)$ und $r \in \operatorname{row}(s)$, so dass wir

$$\begin{aligned}\#\{(s,r) \in \mathcal{T}_{\mathcal{J}} \times \mathcal{T}_{\mathcal{K}} \ : \ (t,s,r) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}\} &= \# \bigcup_{s\in\operatorname{row}(t)} \{(s,r) \ : \ r \in \operatorname{row}(s)\} \\ &\leq \sum_{s\in\operatorname{row}(t)} \#\operatorname{row}(s) \leq \sum_{s\in\operatorname{row}(t)} C_{\mathrm{sp}} \leq C_{\mathrm{sp}}^2\end{aligned}$$

erhalten. Entsprechend können wir auch die Mengen für s und r behandeln. ∎

Nun stehen uns alle Hilfsmittel zur Verfügung, die wir benötigen, um den Rechenaufwand der Multiplikation hierarchischer Matrizen abzuschätzen:

Satz 6.19 (Aufwand Matrix-Multiplikation) *Der in Abbildung 6.7 angegebene Algorithmus benötigt, aufgerufen mit $(t,s,r) = (t_0,s_0,r_0) \in \mathcal{T}_{\mathcal{I}} \times \mathcal{T}_{\mathcal{J}} \times \mathcal{T}_{\mathcal{K}}$, nicht mehr als*

$$C_{mul}\hat{k}^2(\hat{p}+1)^2(\#\hat{t}_0 + \#\hat{s}_0 + \#\hat{r}_0)$$

Rechenoperationen. Für die approximative Berechnung von $\mathbf{Z} \leftarrow \mathbf{Z} + \mathbf{XY}$ fallen also insbesondere nicht mehr als

$$C_{mul}\hat{k}^2(\hat{p}+1)^2(\#\mathcal{I} + \#\mathcal{J} + \#\mathcal{K})$$

Operationen an, wobei die Konstante durch

$$C_{mul} := C_{sp}^2 \max\{C_{mul,lf}, C_{mul,aux}\}$$

gegeben ist.

Beweis. Aus Lemma 6.15 wissen wir, dass nicht mehr als

$$C_{\mathrm{mul,lf}}\hat{k}^2(\hat{p}+1)(\#\hat{t} + \#\hat{s} + \#\hat{r})$$

Operationen anfallen, falls (t,s) oder (s,r) ein Blatt ist.

Aus Lemma 6.16 wissen wir, dass anderenfalls nicht mehr als

$$C_{\mathrm{mul,aux}}\hat{k}^2(\#\hat{t} + \#\hat{r})$$

Operationen durchgeführt werden.

Insgesamt ist damit der Aufwand *pro Tripel* $(t,s,r) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}\times\mathcal{K}}$ durch

$$C_{\mathrm{tmul}}\hat{k}^2(\hat{p}+1)(\#\hat{t}+\#\hat{s}+\#\hat{r}) \qquad \text{mit} \qquad C_{\mathrm{tmul}} := \max\{C_{\mathrm{mul,lf}}, C_{\mathrm{mul,aux}}\}$$

beschränkt. Damit erhalten wir nun für den Gesamtaufwand die Schranke

$$\begin{aligned}\sum_{(t,s,r)\in\operatorname{sons}^*(t_0,s_0,r_0)} & C_{\mathrm{tmul}}\hat{k}^2(\hat{p}+1)(\#\hat{t}+\#\hat{s}+\#\hat{r}) \\ &= C_{\mathrm{tmul}}\hat{k}^2(\hat{p}+1) \sum_{(t,s,r)\in\operatorname{sons}^*(t_0,s_0,r_0)} \#\hat{t}+\#\hat{s}+\#\hat{r}.\end{aligned}$$

Dank Lemma 6.18 und Lemma 2.28 können wir

$$\begin{aligned}\sum_{(t,s,r)\in\operatorname{sons}^*(t_0,s_0,r_0)} \#\hat{t} &\leq \sum_{(t,s)\in\operatorname{sons}^*(t_0,s_0)} \sum_{r\in\operatorname{row}(s)} \#\hat{t} \leq C_{\mathrm{sp}} \sum_{(t,s)\in\operatorname{sons}^*(t_0,s_0)} \#\hat{t} \\ &\leq C_{\mathrm{sp}}^2(\hat{p}+1)\#\hat{t}_0 \end{aligned} \tag{6.7}$$

erhalten und ähnliche Schranken auch für die Summen über s und r gewinnen. Durch Einsetzen in die vorangehende Abschätzung folgt die gewünschte Aussage. ∎

6.6 Inversion hierarchischer Matrizen

Die Matrix-Multiplikation ist ein wichtiges Hilfsmittel bei der Durchführung anspruchsvollerer Matrixoperationen. Als erstes wichtiges Beispiel untersuchen wir die Matrix-Inversion, also die Berechnung von $\mathbf{X}^{-1}$ für eine reguläre Matrix $\mathbf{X} \in \mathbb{K}^{\mathcal{I}\times\mathcal{I}}$.

Wir gehen davon aus, dass $\mathbf{X}$ eine hierarchische Matrix ist, und dass der Blockbaum $\mathcal{T}_{\mathcal{I}\times\mathcal{I}}$ denselben Clusterbaum $\mathcal{T}_{\mathcal{I}}$ für Zeilen und Spalten verwendet.

Der zu konstruierende Algorithmus wird rekursiv arbeiten. Zur Motivation untersuchen wir zunächst den Fall einer regulären Blockmatrix $\mathbf{Y} \in \mathbb{K}^{n\times n}$ der Form

$$\mathbf{Y} = \begin{pmatrix} \mathbf{Y}_{11} & \mathbf{Y}_{12} \\ \mathbf{Y}_{21} & \mathbf{Y}_{22} \end{pmatrix}$$

mit den Teilmatrizen

$$\mathbf{Y}_{11} \in \mathbb{K}^{m\times m}, \quad \mathbf{Y}_{12} \in \mathbb{K}^{m\times(n-m)}, \quad \mathbf{Y}_{21} \in \mathbb{K}^{(n-m)\times m}, \quad \mathbf{Y}_{22} \in \mathbb{K}^{(n-m)\times(n-m)}.$$

Falls $\mathbf{Y}_{11}$ regulär ist, erhalten wir mit Hilfe der Gauß-Elimination die Gleichung

$$\begin{pmatrix} \mathbf{I} & \\ -\mathbf{Y}_{21}\mathbf{Y}_{11}^{-1} & \mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{Y}_{11} & \mathbf{Y}_{12} \\ \mathbf{Y}_{21} & \mathbf{Y}_{22} \end{pmatrix} = \begin{pmatrix} \mathbf{Y}_{11} & \mathbf{Y}_{12} \\ & \mathbf{Y}_{22} - \mathbf{Y}_{21}\mathbf{Y}_{11}^{-1}\mathbf{Y}_{12} \end{pmatrix}.$$

Da beide Faktoren auf der linken Seite regulär sind, muss auch die Matrix auf der rechten Seite regulär sein. Das ist nur möglich, falls auch das *Schur-Komplement*

$$\mathbf{S} := \mathbf{Y}_{22} - \mathbf{Y}_{21}\mathbf{Y}_{11}^{-1}\mathbf{Y}_{12}$$

regulär ist. Dank der Gleichung

$$\begin{pmatrix} \mathbf{Y}_{11}^{-1} & -\mathbf{Y}_{11}^{-1}\mathbf{Y}_{12}\mathbf{S}^{-1} \\ & \mathbf{S}^{-1} \end{pmatrix} \begin{pmatrix} \mathbf{Y}_{11} & \mathbf{Y}_{12} \\ & \mathbf{S} \end{pmatrix} = \begin{pmatrix} \mathbf{I} & \\ & \mathbf{I} \end{pmatrix}$$

kennen wir die Inverse der rechten oberen Dreiecksmatrix und erhalten

$$\begin{pmatrix} \mathbf{Y}_{11}^{-1} & -\mathbf{Y}_{11}^{-1}\mathbf{Y}_{12}\mathbf{S}^{-1} \\ & \mathbf{S}^{-1} \end{pmatrix} \begin{pmatrix} \mathbf{I} & \\ -\mathbf{Y}_{21}\mathbf{Y}_{11}^{-1} & \mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{Y}_{11} & \mathbf{Y}_{12} \\ \mathbf{Y}_{21} & \mathbf{Y}_{22} \end{pmatrix} = \begin{pmatrix} \mathbf{I} & \\ & \mathbf{I} \end{pmatrix},$$

also folgt insbesondere

$$\begin{aligned} \mathbf{Y}^{-1} &= \begin{pmatrix} \mathbf{Y}_{11}^{-1} & -\mathbf{Y}_{11}^{-1}\mathbf{Y}_{12}\mathbf{S}^{-1} \\ & \mathbf{S}^{-1} \end{pmatrix} \begin{pmatrix} \mathbf{I} & \\ -\mathbf{Y}_{21}\mathbf{Y}_{11}^{-1} & \mathbf{I} \end{pmatrix} \\ &= \begin{pmatrix} \mathbf{Y}_{11}^{-1} + \mathbf{Y}_{11}^{-1}\mathbf{Y}_{12}\mathbf{S}^{-1}\mathbf{Y}_{21}\mathbf{Y}_{11}^{-1} & -\mathbf{Y}_{11}^{-1}\mathbf{Y}_{12}\mathbf{S}^{-1} \\ -\mathbf{S}^{-1}\mathbf{Y}_{21}\mathbf{Y}_{11}^{-1} & \mathbf{S}^{-1} \end{pmatrix}. \end{aligned} \tag{6.8}$$

Damit steht uns eine explizite Formel zur Berechnung der Inversen einer 2×2-Blockmatrix zur Verfügung.

Dieses Ergebnis wenden wir nun auf den Fall einer $\mathcal{H}$-Matrix an. Sei $t \in \mathcal{T}_{\mathcal{I}}$ ein Cluster. Wir untersuchen die Inverse einer Teilmatrix $\mathbf{X}|_{\hat{t}\times\hat{t}}$. Falls t ein Blatt ist, dürfen wir davon ausgehen, dass $\#\hat{t}$ klein genug ist, um die Inverse direkt berechnen zu können, etwa mit einer pivotisierten Gauß-Elimination.

Falls t kein Blatt ist, setzen wir $\tau := \#\operatorname{sons}(t)$ und wählen $t_1, \ldots, t_\tau$ mit

$$\operatorname{sons}(t) = \{t_1, \ldots, t_\tau\}.$$

Nun zerlegen wir

$$\widehat{\mathbf{X}} := \mathbf{X}|_{\hat{t}\times\hat{t}}$$

in Teilmatrizen

$$\widehat{\mathbf{X}}_{ij} := \mathbf{X}|_{\hat{t}_i\times\hat{t}_j} \qquad \text{für alle } i,j \in \{1,\ldots,\tau\},$$

mit denen wir die Darstellung

$$\widehat{\mathbf{X}} = \begin{pmatrix} \widehat{\mathbf{X}}_{11} & \ldots & \widehat{\mathbf{X}}_{1\tau} \\ \vdots & \ddots & \vdots \\ \widehat{\mathbf{X}}_{\tau 1} & \ldots & \widehat{\mathbf{X}}_{\tau\tau} \end{pmatrix}$$

erhalten. Entsprechend zerlegen wir die Inverse $\mathbf{Z} := \widehat{\mathbf{X}}^{-1}$ in Teilmatrizen

$$\mathbf{Z}_{ij} := \mathbf{Z}|_{\hat{t}_i\times\hat{t}_j} \qquad \text{für alle } i,j \in \{1,\ldots,\tau\},$$

um die Darstellung

$$\widehat{\mathbf{X}}^{-1} = \mathbf{Z} = \begin{pmatrix} \mathbf{Z}_{11} & \ldots & \mathbf{Z}_{1\tau} \\ \vdots & \ddots & \vdots \\ \mathbf{Z}_{\tau 1} & \ldots & \mathbf{Z}_{\tau\tau} \end{pmatrix}$$

verwenden zu können. Damit sich unsere für den Fall der 2×2-Blockmatrix gewonnenen Erkenntnisse einsetzen lassen, setzen wir

$$\widehat{\mathbf{X}}_{*1} := \begin{pmatrix} \widehat{\mathbf{X}}_{21} \\ \vdots \\ \widehat{\mathbf{X}}_{\tau 1} \end{pmatrix}, \qquad \widehat{\mathbf{X}}_{1*} := \begin{pmatrix} \widehat{\mathbf{X}}_{12} & \dots & \widehat{\mathbf{X}}_{1\tau} \end{pmatrix}, \qquad \widehat{\mathbf{X}}_{**} := \begin{pmatrix} \widehat{\mathbf{X}}_{22} & \dots & \widehat{\mathbf{X}}_{2\tau} \\ \vdots & \ddots & \vdots \\ \widehat{\mathbf{X}}_{\tau 2} & \dots & \widehat{\mathbf{X}}_{\tau\tau} \end{pmatrix},$$

$$\mathbf{Z}_{*1} := \begin{pmatrix} \mathbf{Z}_{21} \\ \vdots \\ \mathbf{Z}_{\tau 1} \end{pmatrix}, \qquad \mathbf{Z}_{1*} := \begin{pmatrix} \mathbf{Z}_{12} & \dots & \mathbf{Z}_{1\tau} \end{pmatrix}, \qquad \mathbf{Z}_{**} := \begin{pmatrix} \mathbf{Z}_{22} & \dots & \mathbf{Z}_{2\tau} \\ \vdots & \ddots & \vdots \\ \mathbf{Z}_{\tau 2} & \dots & \mathbf{Z}_{\tau\tau} \end{pmatrix}$$

und finden

$$\widehat{\mathbf{X}} = \begin{pmatrix} \widehat{\mathbf{X}}_{11} & \widehat{\mathbf{X}}_{1*} \\ \widehat{\mathbf{X}}_{*1} & \widehat{\mathbf{X}}_{**} \end{pmatrix}, \qquad \mathbf{Z} = \begin{pmatrix} \mathbf{Z}_{11} & \mathbf{Z}_{1*} \\ \mathbf{Z}_{*1} & \mathbf{Z}_{**} \end{pmatrix}.$$

Mit diesen Matrizen können wir wie im 2×2-Fall verfahren: Zunächst berechnen wir $\widehat{\mathbf{X}}_{11}^{-1}$, also die Inverse der Teilmatrix zu dem Cluster t_1, durch eine Rekursion.

Sobald das Ergebnis vorliegt, können wir die Hilfsmatrix

$$\mathbf{L}_{*1} := \widehat{\mathbf{X}}_{*1}\widehat{\mathbf{X}}_{11}^{-1} = \begin{pmatrix} \mathbf{L}_{21} \\ \vdots \\ \mathbf{L}_{\tau 1} \end{pmatrix}, \qquad \mathbf{L}_{i1} := \widehat{\mathbf{X}}_{i1}\widehat{\mathbf{X}}_{11}^{-1} \qquad \text{für alle } i \in \{2, \dots, \tau\}$$

bestimmen und das Schur-Komplement

$$\mathbf{S}_{**} := \widehat{\mathbf{X}}_{**} - \mathbf{L}_{*1}\widehat{\mathbf{X}}_{1*} = \begin{pmatrix} \mathbf{S}_{22} & \dots & \mathbf{S}_{2\tau} \\ \vdots & \ddots & \vdots \\ \mathbf{S}_{\tau 2} & \dots & \mathbf{S}_{\tau\tau} \end{pmatrix},$$

$$\mathbf{S}_{ij} := \widehat{\mathbf{X}}_{ij} - \mathbf{L}_{i1}\widehat{\mathbf{X}}_{1j} \qquad \text{für alle } i, j \in \{2, \dots, \tau\}$$

aufstellen. Nun benötigen wir die Inverse des Schur-Komplements $\mathbf{S}_{**}$. Falls $\tau = 2$ gelten sollte, können wir sie direkt berechnen, anderenfalls nutzen wir aus, dass die Matrix eine $(\tau - 1) \times (\tau - 1)$-Blockmatrix ist, so dass wir rekursiv ihre Inverse auf die Inversen kleinerer Teilmatrizen zurückführen können.

Die Inverse des Schur-Komplements ist nach (6.8) bereits der rechte untere Diagonalblock der Inversen $\mathbf{Z}$, sobald wir sie berechnet haben gilt also

$$\mathbf{Z}_{**} := \mathbf{S}_{**}^{-1} = \begin{pmatrix} \mathbf{Z}_{22} & \dots & \mathbf{Z}_{2\tau} \\ \vdots & \ddots & \vdots \\ \mathbf{Z}_{\tau 2} & \dots & \mathbf{Z}_{\tau\tau} \end{pmatrix}.$$

Nach Gleichung (6.8) können wir damit durch Multiplikation mit $\mathbf{L}_{*1}$ die erste Spalte der Inversen berechnen, denn es gilt

$$\mathbf{Z}_{*1} = -\mathbf{S}_{**}^{-1}\widehat{\mathbf{X}}_{*1}\widehat{\mathbf{X}}_{11}^{-1} = -\mathbf{S}_{**}^{-1}\mathbf{L}_{*1} = -\begin{pmatrix} \mathbf{Z}_{22} & \dots & \mathbf{Z}_{2\tau} \\ \vdots & \ddots & \vdots \\ \mathbf{Z}_{\tau 2} & \dots & \mathbf{Z}_{\tau\tau} \end{pmatrix} \begin{pmatrix} \mathbf{L}_{21} \\ \vdots \\ \mathbf{L}_{\tau 1} \end{pmatrix},$$

$$\mathbf{Z}_{i1} = -\sum_{j=2}^{\tau} \mathbf{Z}_{ij}\mathbf{L}_{j1} \qquad \text{für alle } i \in \{2,\dots,\tau\}.$$

Zur Bestimmung der ersten Zeile der Inversen benutzen wir eine zweite Hilfsmatrix

$$\mathbf{R}_{1*} := \widehat{\mathbf{X}}_{11}^{-1}\widehat{\mathbf{X}}_{1*} = \begin{pmatrix}\mathbf{R}_{12} & \dots & \mathbf{R}_{1\tau}\end{pmatrix}, \quad \mathbf{R}_{1j} := \widehat{\mathbf{X}}_{11}^{-1}\widehat{\mathbf{X}}_{1j} \quad \text{für alle } j \in \{2,\dots,\tau\},$$

die es uns ermöglicht, die erste Zeile dank (6.8) in der Form

$$\mathbf{Z}_{1*} = -\widehat{\mathbf{X}}_{11}^{-1}\widehat{\mathbf{X}}_{1*}\mathbf{S}_{**}^{-1} = -\mathbf{R}_{1*}\mathbf{S}_{**}^{-1} = -\begin{pmatrix}\mathbf{R}_{12} & \dots & \mathbf{R}_{1\tau}\end{pmatrix}\begin{pmatrix}\mathbf{Z}_{22} & \dots & \mathbf{Z}_{2\tau}\\ \vdots & \ddots & \vdots\\ \mathbf{Z}_{\tau 2} & \dots & \mathbf{Z}_{\tau\tau}\end{pmatrix},$$

$$\mathbf{Z}_{1j} = -\sum_{i=2}^{\tau} \mathbf{R}_{1i}\mathbf{Z}_{ij} \qquad \text{für alle } j \in \{2,\dots,\tau\}$$

zu berechnen. Es fehlt noch der linke obere Eintrag der Inversen, den wir gemäß (6.8) mit einer der Gleichungen

$$\mathbf{Z}_{11} = \widehat{\mathbf{X}}_{11}^{-1} + \widehat{\mathbf{X}}_{11}^{-1}\widehat{\mathbf{X}}_{1*}\mathbf{S}_{**}^{-1}\widehat{\mathbf{X}}_{*1}\widehat{\mathbf{X}}_{11}^{-1} = \widehat{\mathbf{X}}_{11}^{-1} - \mathbf{Z}_{1*}\mathbf{L}_{*1} = \widehat{\mathbf{X}}_{11}^{-1} - \sum_{i=2}^{\tau}\mathbf{Z}_{1i}\mathbf{L}_{i1},$$

$$\mathbf{Z}_{11} = \widehat{\mathbf{X}}_{11}^{-1} + \widehat{\mathbf{X}}_{11}^{-1}\widehat{\mathbf{X}}_{1*}\mathbf{S}_{**}^{-1}\widehat{\mathbf{X}}_{*1}\widehat{\mathbf{X}}_{11}^{-1} = \widehat{\mathbf{X}}_{11}^{-1} - \mathbf{R}_{1*}\mathbf{Z}_{*1} = \widehat{\mathbf{X}}_{11}^{-1} - \sum_{j=2}^{\tau}\mathbf{R}_{1j}\mathbf{Z}_{j1}$$

berechnen können. Damit sind $\mathbf{Z}_{11}$, $\mathbf{Z}_{1*}$, $\mathbf{Z}_{*1}$ sowie $\mathbf{Z}_{**}$ bestimmt, also liegt die vollständige Inverse vor.

Die Rekursion für $\tau > 2$ führen wir nicht explizit durch, sondern wir implementieren sie als eine Schleife, die für $k \in \{1,\dots,\tau\}$ jeweils auf den Teilmatrizen

$$\begin{pmatrix}\widehat{\mathbf{X}}_{kk} & \dots & \widehat{\mathbf{X}}_{k\tau}\\ \vdots & \ddots & \vdots\\ \widehat{\mathbf{X}}_{\tau k} & \dots & \widehat{\mathbf{X}}_{\tau\tau}\end{pmatrix} \qquad \begin{pmatrix}\mathbf{Z}_{kk} & \dots & \mathbf{Z}_{k\tau}\\ \vdots & \ddots & \vdots\\ \mathbf{Z}_{\tau k} & \dots & \mathbf{Z}_{\tau\tau}\end{pmatrix}$$

rechnet. Für $k = 1$ erhalten wir die Ausgangsmatrix, sobald das Schur-Komplement berechnet wurde, gehen wir zu $k = 2$ über, und so weiter, bis wir für $k = \tau$ die Teilmatrix $\widehat{\mathbf{X}}_{\tau\tau}$ invertieren und die Schleife endet. Damit enthält $\mathbf{Z}_{\tau\tau}$ bereits den korrekten Teil der endgültigen Inversen. Dann beginnt die gegenläufige Schleife, in der wir zunächst für $k = \tau - 1$ die Inverse der rechten unteren 2×2-Matrix bestimmen und uns dann bis zu $k = 1$ vorarbeiten und dann die vollständige Inverse berechnet haben. Der resultierende Algorithmus findet sich in Abbildung 6.8.

Unser Ziel ist es nun, seinen Rechenaufwand abzuschätzen. Dazu nutzen wir aus, dass die Inversion intensiv von der Matrix-Multiplikation Gebrauch macht: Abgesehen von der Inversion für Blätter kommen in unserem Algorithmus sämtliche Rechenoperationen nur in Matrix-Multiplikationen vor. Also bietet es sich an, den Rechenaufwand ähnlich

```
procedure inv_h(t, var X, L, R);
if sons(t) = ∅ then
  X|_{t̂×t̂} ← X|_{t̂×t̂}^{-1}
else
  τ ← #sons(t);   {t_1, ..., t_τ} ← sons(t);
  for k = 1, ..., τ do
    inv_h(t_k, X, L, R);                                   {X̂_11 ← X̂_11^{-1}}
    for i ∈ {k+1, ..., τ} do
      L|_{t̂_i×t̂_k} ← 0;   mul_h_h(1, t_i, t_k, t_k, X, X, L);   {L_i1 ← X̂_i1 X̂_11^{-1}}
      R|_{t̂_k×t̂_i} ← 0;   mul_h_h(1, t_k, t_k, t_i, X, X, R)    {R_1i ← X̂_11^{-1} X̂_1i}
    end for;
    for i, j ∈ {k+1, ..., τ} do
      mul_h_h(−1, t_i, t_k, t_j, L, X, X)                  {S_ij ← X̂_ij − L_i1 X̂_1j}
    end for
  end for;
  for k = τ − 1, ..., 1 do
    for i ∈ {k+1, ..., τ} do
      X|_{t̂_i×t̂_k} ← 0;   X|_{t̂_k×t̂_i} ← 0;
      for j ∈ {k+1, ..., τ} do
        mul_h_h(−1, t_i, t_j, t_k, X, L, X);               {Z_i1 ← Z_i1 − Z_ij L_j1}
        mul_h_h(−1, t_k, t_j, t_i, R, X, X)                {Z_1i ← Z_1i − R_1j Z_ji}
      end for;
      mul_h_h(−1, t_k, t_i, t_k, X, L, X)                  {Z_11 ← Z_11 − Z_1i L_i1}
    end for
  end for
end if
```

Abbildung 6.8: Inversion einer $\mathcal{H}$-Matrix: $\mathbf{X} \leftarrow \mathbf{X}^{-1}$, $\mathbf{L}$ und $\mathbf{R}$ dienen als Hilfsspeicher für eine untere und eine obere strikte Block-Dreiecksmatrix

wie zuvor abzuschätzen, indem wir die Struktur der rekursiven Aufrufe der Funktionen inv_h und mul_h_h analysieren. Für den letzteren Fall haben wir das bereits getan, für die Inversion stellen wir fest, dass für $k \in \{1, \ldots, \tau\}$ und $i, j \in \{k+1, \ldots, \tau\}$ Funktionsaufrufe je einmal für

- t_k (Inversion zur Berechnung von $\mathbf{X}_{kk}^{-1}$),
- (t_i, t_k, t_k) (Aufstellen von $\mathbf{L}_{ik}$),
- (t_k, t_k, t_i) (Aufstellen von $\mathbf{R}_{ki}$),
- (t_i, t_k, t_j) (Aufstellen von $\mathbf{S}_{ij}$),
- (t_i, t_j, t_k) (Aufstellen von $\mathbf{Z}_{ik}$),
- (t_k, t_i, t_j) (Aufstellen von $\mathbf{Z}_{ki}$),

- (t_k, t_i, t_k) (Aufstellen von $\mathbf{Z}_{kk}$)

erfolgen. Wenn wir den Aufruf für t_k als Aufruf mit dem Tripel (t_k, t_k, t_k) interpretieren, sehen wir, dass jede Kombination der Söhne von t genau einmal vorkommt, es werden also rekursive Aufrufe für alle Tripel $(t_i, t_j, t_k) \in \operatorname{sons}(t) \times \operatorname{sons}(t) \times \operatorname{sons}(t)$ durchgeführt, falls t kein Blatt ist. Das entspricht gerade den Söhnen des Tripels (t, t, t) in dem für die Analyse der Multiplikation verwendeten Tripelbaum $\mathcal{T}_{\mathcal{I}\times\mathcal{I}\times\mathcal{I}}$.

Also wird bei der Inversion derselbe Baum durchlaufen wie bei der Multiplikation, allerdings wahrscheinlich in einer anderen Reihenfolge. Wenn wir den Rechenaufwand pro Tripel abschätzen können, erhalten wir mit derselben Vorgehensweise wie bei der Multiplikation auch eine Aufwandsabschätzung für die Inversion.

Für die Multiplikation wissen wir dank der Lemmas 6.15 und 6.16 bereits, dass eine Konstante $C_{\text{tmul}} \in \mathbb{N}$ so existiert, dass der Aufwand für einen Tripel $(t, s, r) \in \mathcal{T}_{\mathcal{I}\times\mathcal{I}\times\mathcal{I}}$ durch

$$C_{\text{tmul}}\hat{k}^2(\hat{p}+1)(\#\hat{t} + \#\hat{s} + \#\hat{r})$$

beschränkt ist. Bei der Inversion stellen wir fest, dass der Algorithmus inv_h nur in einem einzigen Fall überhaupt rechnet, nämlich falls t ein Blatt ist und die Inverse des korrespondierenden Diagonalblocks zu bestimmen ist. Wenn wir diese Berechnung etwa per Gauß-Elimination durchführen, existiert eine Konstante $C_{\text{invb}} \in \mathbb{N}$ derart, dass die Inversion einer Teilmatrix $\mathbf{Y}|_{\hat{t}\times\hat{t}}$ nicht mehr als $C_{\text{invb}}(\#\hat{t})^3$ Operationen erfordert. Um eine ähnliche Abschätzung wie im Fall der Multiplikation zu erhalten, nutzen wir aus, dass t ein Blatt ist und also $\#\hat{t} \leq r_{\mathcal{I}}$ gilt, um

$$C_{\text{invb}}(\#\hat{t})^3 \leq C_{\text{invb}} r_{\mathcal{I}}^2 \#\hat{t} \leq C_{\text{invb}}\hat{k}^2\#\hat{t} \leq \frac{C_{\text{invb}}}{3}\hat{k}^2(\hat{p}+1)(\#\hat{t} + \#\hat{t} + \#\hat{t})$$

zu erhalten. Daraus folgt, dass der Aufwand pro Tripel $(t, s, r) \in \mathcal{T}_{\mathcal{I}\times\mathcal{I}\times\mathcal{I}}$ bei der Inversion nicht

$$C_{\text{tinv}}\hat{k}^2(\hat{p}+1)(\#\hat{t} + \#\hat{s} + \#\hat{r}), \qquad C_{\text{tinv}} := \max\left\{\frac{C_{\text{invb}}}{3}, C_{\text{tmul}}\right\}$$

Operationen überschreitet. Nun können wir wie bei der Matrix-Multiplikation verfahren, um eine Abschätzung des Aufwands zu finden:

Satz 6.20 (Aufwand Matrix-Inversion) *Der in Abbildung 6.8 angegebene Algorithmus benötigt, aufgerufen mit $t = t_0 \in \mathcal{T}_{\mathcal{I}}$, nicht mehr als*

$$C_{inv}\hat{k}^2(\hat{p}+1)^2\#\hat{t}_0$$

Rechenoperationen. Für die approximative Berechnung von $\mathbf{Z} \leftarrow \mathbf{X}^{-1}$ fallen also insbesondere nicht mehr als

$$C_{inv}\hat{k}^2(\hat{p}+1)^2\#\mathcal{I}$$

Operationen an, wobei die Konstante in beiden Fällen durch

$$C_{inv} := 3C_{sp}^2 C_{tinv}$$

gegeben ist.

Beweis. Wie wir bereits gesehen haben, sind für ein Tripel $(t, s, r) \in \mathcal{T}_{\mathcal{I}\times\mathcal{I}\times\mathcal{I}}$ nicht mehr als

$$C_{\mathrm{tinv}}\hat{k}^2(\hat{p}+1)(\#\hat{t}+\#\hat{s}+\#\hat{r})$$

Rechenoperationen erforderlich. Da wir auch schon wissen, dass im Zuge der Inversion sämtliche Tripel aus dem Teilbaum $\mathrm{sons}^*(t_0, t_0, t_0)$ vorkommen, aber auch keine anderen, erhalten wir die obere Schranke

$$C_{\mathrm{tinv}}\hat{k}^2(\hat{p}+1) \sum_{(t,s,r)\in\mathrm{sons}^*(t_0,t_0,t_0)} \#\hat{t}+\#\hat{s}+\#\hat{r}.$$

Dank (6.7) lässt sich dieser Term durch

$$C_{\mathrm{tinv}}C_{\mathrm{sp}}^2\hat{k}^2(\hat{p}+1)^2(\#\hat{t}_0+\#\hat{t}_0+\#\hat{t}_0)$$

abschätzen, und das ist das gewünschte Ergebnis. ∎

Bemerkung 6.21 (Fehlersteuerung) *Bei der Inversion ist es wesentlich schwieriger als bei der Multiplikation, eine gegebene Genauigkeit zu garantieren, da jede Inverse einer Teilmatrix in die Berechnung anderer Teilmatrizen eingeht, die in der Regel auch wieder invertiert werden.*

Dieses Problem vermeiden alternative Verfahren, etwa solche, die auf der Newton-Iteration aufbauen [19], allerdings sind sie häufig auch wesentlich rechenintensiver.

6.7 Dreieckszerlegungen

In der Praxis sind wir häufig nur daran interessiert, Gleichungssysteme mit einer Matrix $\mathbf{G}$ zu lösen, die vollständige Inverse wird nur selten benötigt. Für Aufgaben dieser Art sind Dreiecksfaktorisierungen häufig nützlich: Wir suchen nach einer unteren Dreiecksmatrix $\mathbf{L}$ und einer oberen Dreiecksmatrix $\mathbf{R}$ mit

$$\mathbf{LR} = \mathbf{G},$$

einer sogenannten *LR-Zerlegung*, denn mit Hilfe dieser Darstellung lässt sich das System

$$\mathbf{Gx} = \mathbf{b}$$

auch in der Form

$$\mathbf{Ly} = \mathbf{b}, \qquad \mathbf{Rx} = \mathbf{y}$$

schreiben, und aus dieser Form lässt sich durch Vorwärtseinsetzen in $\mathbf{L}$ und Rückwärtseinsetzen in $\mathbf{R}$ die Lösung $\mathbf{x}$ berechnen.

Unser Ziel ist es also nun, eine LR-Zerlegung einer hierarchischen Matrix zu berechnen, selbstverständlich wieder approximativ, um Rechenzeit und Speicherplatz zu sparen. Den Ansätzen aus [29, 22] folgend gehen wir ähnlich wie im Fall der Inversion rekursiv vor:

Wir untersuchen die Berechnung der LR-Zerlegungen von Teilmatrizen $\mathbf{G}|_{\hat{t}\times\hat{t}}$. Falls t ein Blatt des Clusterbaums ist, können wir die Zerlegung in der für vollbesetzte Matrizen üblichen Weise bestimmen.

Anderenfalls setzen wir wieder $\tau := \#\operatorname{sons}(t)$ und wählen $t_1, \ldots, t_\tau \in \mathcal{T}_\mathcal{I}$ mit

$$\operatorname{sons}(t) = \{t_1, \ldots, t_\tau\}.$$

Die Matrix

$$\widehat{\mathbf{G}} := \mathbf{G}|_{\hat{t}\times\hat{t}}$$

zerlegen wir entsprechend in Teilmatrizen

$$\widehat{\mathbf{G}}_{ij} := \mathbf{G}|_{\hat{t}_i\times\hat{t}_j} \qquad \text{für alle } i,j \in \{1,\ldots,\tau\},$$

um die Darstellung

$$\widehat{\mathbf{G}} = \begin{pmatrix} \widehat{\mathbf{G}}_{11} & \ldots & \widehat{\mathbf{G}}_{1\tau} \\ \vdots & \ddots & \vdots \\ \widehat{\mathbf{G}}_{\tau 1} & \ldots & \widehat{\mathbf{G}}_{\tau\tau} \end{pmatrix}$$

zu erhalten. Wie im Fall der Inversion führen wir den allgemeinen Fall auf den Fall einer 2×2-Blockmatrix zurück, indem wir

$$\widehat{\mathbf{G}}_{*1} := \begin{pmatrix} \widehat{\mathbf{G}}_{21} \\ \vdots \\ \widehat{\mathbf{G}}_{\tau 1} \end{pmatrix}, \qquad \widehat{\mathbf{G}}_{1*} := \begin{pmatrix} \widehat{\mathbf{G}}_{12} & \ldots & \widehat{\mathbf{G}}_{1\tau} \end{pmatrix}, \qquad \widehat{\mathbf{G}}_{**} := \begin{pmatrix} \widehat{\mathbf{G}}_{22} & \ldots & \widehat{\mathbf{G}}_{2\tau} \\ \vdots & \ddots & \vdots \\ \widehat{\mathbf{G}}_{\tau 2} & \ldots & \widehat{\mathbf{G}}_{\tau\tau} \end{pmatrix}$$

einführen und

$$\widehat{\mathbf{G}} = \begin{pmatrix} \widehat{\mathbf{G}}_{11} & \widehat{\mathbf{G}}_{1*} \\ \widehat{\mathbf{G}}_{*1} & \widehat{\mathbf{G}}_{**} \end{pmatrix}$$

erhalten. Mit den zu berechnenden Dreiecksfaktoren $\mathbf{L}$ und $\mathbf{R}$ verfahren wir entsprechend: Wir setzen

$$\widehat{\mathbf{L}} := \mathbf{L}|_{\hat{t}\times\hat{t}}, \qquad \widehat{\mathbf{L}}_{ij} := \mathbf{L}|_{\hat{t}_i\times\hat{t}_j} \qquad \text{für alle } i,j \in \{1,\ldots,\tau\}, \tag{6.9}$$

$$\widehat{\mathbf{R}} := \mathbf{R}|_{\hat{t}\times\hat{t}}, \qquad \widehat{\mathbf{R}}_{ij} := \mathbf{R}|_{\hat{t}_i\times\hat{t}_j} \qquad \text{für alle } i,j \in \{1,\ldots,\tau\}. \tag{6.10}$$

Zur Reduktion auf 2×2-Blockmatrizen verwenden wir auch hier

$$\widehat{\mathbf{L}}_{*1} := \begin{pmatrix} \widehat{\mathbf{L}}_{21} \\ \vdots \\ \widehat{\mathbf{L}}_{\tau 1} \end{pmatrix}, \qquad \widehat{\mathbf{L}}_{**} := \begin{pmatrix} \widehat{\mathbf{L}}_{22} & & \\ \vdots & \ddots & \\ \widehat{\mathbf{L}}_{\tau 2} & \ldots & \widehat{\mathbf{L}}_{\tau\tau} \end{pmatrix},$$

$$\widehat{\mathbf{R}}_{1*} := \begin{pmatrix} \widehat{\mathbf{R}}_{12} & \ldots & \widehat{\mathbf{R}}_{1\tau} \end{pmatrix}, \qquad \widehat{\mathbf{R}}_{**} := \begin{pmatrix} \widehat{\mathbf{R}}_{22} & \ldots & \widehat{\mathbf{R}}_{2\tau} \\ & \ddots & \vdots \\ & & \widehat{\mathbf{R}}_{\tau\tau} \end{pmatrix},$$

so dass wir insgesamt zu der Gleichung

$$\begin{pmatrix} \widehat{\mathbf{L}}_{11} & \\ \widehat{\mathbf{L}}_{*1} & \widehat{\mathbf{L}}_{**} \end{pmatrix} \begin{pmatrix} \widehat{\mathbf{R}}_{11} & \widehat{\mathbf{R}}_{1*} \\ & \widehat{\mathbf{R}}_{**} \end{pmatrix} = \widehat{\mathbf{L}}\widehat{\mathbf{R}} = \widehat{\mathbf{G}} = \begin{pmatrix} \widehat{\mathbf{G}}_{11} & \widehat{\mathbf{G}}_{1*} \\ \widehat{\mathbf{G}}_{*1} & \widehat{\mathbf{G}}_{**} \end{pmatrix}$$

gelangen. Durch Ausmultiplizieren der linken Seite folgt

$$\widehat{\mathbf{L}}_{11}\widehat{\mathbf{R}}_{11} = \widehat{\mathbf{G}}_{11}, \qquad \widehat{\mathbf{L}}_{11}\widehat{\mathbf{R}}_{1*} = \widehat{\mathbf{G}}_{1*}, \tag{6.11a}$$

$$\widehat{\mathbf{L}}_{*1}\widehat{\mathbf{R}}_{11} = \widehat{\mathbf{G}}_{*1}, \qquad \widehat{\mathbf{L}}_{*1}\widehat{\mathbf{R}}_{1*} + \widehat{\mathbf{L}}_{**}\widehat{\mathbf{R}}_{**} = \widehat{\mathbf{G}}_{**}, \tag{6.11b}$$

und diese Gleichungen können wir für die rekursive Berechnung der Matrizen $\widehat{\mathbf{L}}$ und $\widehat{\mathbf{R}}$ verwenden: Die erste Gleichung erlaubt es uns $\widehat{\mathbf{L}}_{11}$ und $\widehat{\mathbf{R}}_{11}$ mittels der LR-Zerlegung der Matrix $\widehat{\mathbf{G}}_{11}$ zu gewinnen.

Sobald diese Matrizen bekannt sind, können wir mit Hilfe der zweiten und dritten Gleichung die Matrizen $\widehat{\mathbf{R}}_{1*}$ und $\widehat{\mathbf{L}}_{*1}$ durch Vorwärtseinsetzen in die Matrizen $\widehat{\mathbf{L}}_{11}$ beziehungsweise $\widehat{\mathbf{R}}_{11}^*$ berechnen. Die genaue Umsetzung dieser Berechnung im Rahmen der approximativen $\mathcal{H}$-Matrix-Arithmetik werden wir später diskutieren.

Wenn $\widehat{\mathbf{L}}_{*1}$ und $\widehat{\mathbf{R}}_{1*}$ bekannt sind, folgt aus der vierten Gleichung schließlich

$$\widehat{\mathbf{L}}_{**}\widehat{\mathbf{R}}_{**} = \widehat{\mathbf{G}}_{**} - \widehat{\mathbf{L}}_{*1}\widehat{\mathbf{R}}_{1*},$$

so dass wir $\widehat{\mathbf{L}}_{**}$ und $\widehat{\mathbf{R}}_{**}$ als LR-Zerlegung der rechten Seite dieser Gleichung rekursiv konstruieren können.

Es bleibt nur noch zu diskutieren, wie sich die Gleichungen $\widehat{\mathbf{L}}_{11}\widehat{\mathbf{R}}_{1*} = \widehat{\mathbf{G}}_{1*}$ und $\widehat{\mathbf{L}}_{*1}\widehat{\mathbf{R}}_{11} = \widehat{\mathbf{G}}_{*1}$ lösen lassen. Dank der Gleichung

$$\widehat{\mathbf{G}}_{*1}^* = (\widehat{\mathbf{L}}_{*1}\widehat{\mathbf{R}}_{11})^* = \widehat{\mathbf{R}}_{11}^*\widehat{\mathbf{L}}_{*1}^*$$

können wir die zweite auf die erste zurückführen, denn hier ist $\widehat{\mathbf{R}}_{11}^*$ wieder eine untere Dreiecksmatrix.

Wir stehen also vor der Aufgabe, ein System der Form

$$\widehat{\mathbf{L}}\widehat{\mathbf{X}} = \widehat{\mathbf{Y}}$$

zu lösen. Zunächst untersuchen wir den Spezialfall, dass $\widehat{\mathbf{X}}$ und $\widehat{\mathbf{Y}}$ Vektoren sind, dass also $\widehat{\mathbf{y}} \in \mathbb{K}^{\hat{t}}$ gegeben und $\widehat{\mathbf{x}} \in \mathbb{K}^{\hat{t}}$ gesucht ist. Falls $\hat{t}$ ein Blatt ist, können wir die Aufgabe direkt lösen. Anderenfalls verfahren wir wieder rekursiv und zerlegen dazu die Vektoren $\widehat{\mathbf{x}}, \widehat{\mathbf{y}} \in \mathbb{K}^{\hat{t}}$ in der üblichen Form

$$\widehat{\mathbf{x}} = \begin{pmatrix} \widehat{\mathbf{x}}_1 \\ \vdots \\ \widehat{\mathbf{x}}_\tau \end{pmatrix}, \quad \widehat{\mathbf{x}}_i = \widehat{\mathbf{x}}|_{\hat{t}_i}, \quad \widehat{\mathbf{y}} = \begin{pmatrix} \widehat{\mathbf{y}}_1 \\ \vdots \\ \widehat{\mathbf{y}}_\tau \end{pmatrix}, \quad \widehat{\mathbf{y}}_i = \widehat{\mathbf{y}}|_{\hat{t}_i} \qquad \text{für alle } i \in \{1, \ldots, \tau\},$$

um die Gleichung

$$\begin{pmatrix} \widehat{\mathbf{L}}_{11} & & \\ \vdots & \ddots & \\ \widehat{\mathbf{L}}_{\tau 1} & \ldots & \widehat{\mathbf{L}}_{\tau\tau} \end{pmatrix} \begin{pmatrix} \widehat{\mathbf{x}}_1 \\ \vdots \\ \widehat{\mathbf{x}}_\tau \end{pmatrix} = \widehat{\mathbf{L}}\widehat{\mathbf{x}} = \widehat{\mathbf{y}} = \begin{pmatrix} \widehat{\mathbf{y}}_1 \\ \vdots \\ \widehat{\mathbf{y}}_\tau \end{pmatrix}$$

zu erhalten. Genau wie jedes andere Gleichungssystem in unterer Dreiecksgestalt lässt sich auch dieses durch Vorwärtseinsetzen lösen, also in unserem Fall durch rekursives Auflösen der Gleichungen

$$\begin{aligned}
\widehat{\mathbf{L}}_{11}\widehat{\mathbf{x}}_1 = \widehat{\mathbf{y}}_1 \qquad &\Longleftrightarrow \qquad \widehat{\mathbf{L}}_{11}\widehat{\mathbf{x}}_1 = \widehat{\mathbf{y}}_1,\\
\widehat{\mathbf{L}}_{21}\widehat{\mathbf{x}}_1 + \widehat{\mathbf{L}}_{22}\widehat{\mathbf{x}}_2 = \widehat{\mathbf{y}}_2 \qquad &\Longleftrightarrow \qquad \widehat{\mathbf{L}}_{22}\widehat{\mathbf{x}}_2 = \widehat{\mathbf{y}}_2 - \widehat{\mathbf{L}}_{21}\widehat{\mathbf{x}}_1,\\
\vdots \qquad &\qquad \vdots\\
\sum_{k=1}^{\tau-1} \widehat{\mathbf{L}}_{\tau k}\widehat{\mathbf{x}}_k + \widehat{\mathbf{L}}_{\tau\tau}\widehat{\mathbf{x}}_\tau = \widehat{\mathbf{y}}_\tau \qquad &\Longleftrightarrow \qquad \widehat{\mathbf{L}}_{\tau\tau}\widehat{\mathbf{x}}_\tau = \widehat{\mathbf{y}}_\tau - \sum_{k=1}^{\tau-1} \widehat{\mathbf{L}}_{\tau k}\widehat{\mathbf{x}}_k
\end{aligned}$$

nach den Teilvektoren $\widehat{\mathbf{x}}_1, \ldots, \widehat{\mathbf{x}}_\tau$. Der resultierende Algorithmus ist in Abbildung 6.9 zusammengefasst.

```
procedure forward_subst_h(t, L, var x);
if sons(t) = ∅ then
  x|_t̂ ← L|^{-1}_{t̂×t̂} x|_t̂
else
  τ ← #sons(t);    {t_1, ..., t_τ} ← sons(t)
  for i = 1, ..., τ do
    for k ∈ {1, ..., i − 1} do
      b' ← (t_i, t_k);    eval_h(−1, b', L, x, x)
    end for;
    forward_subst_h(t_i, L, x)
  end for
end if
```

Abbildung 6.9: Vorwärtseinsetzen für eine $\mathcal{H}$-Matrix $\mathbf{L}$ in unterer Blockdreiecksform zur Berechnung von $\mathbf{x} \leftarrow \mathbf{L}^{-1}\mathbf{x}$

Entsprechend können wir selbstverständlich auch das Rückwärtseinsetzen für eine $\mathcal{H}$-Matrix in oberer Blockdreiecksform als Rekursion über Teilblöcke formulieren und erhalten den in Abbildung 6.10 angegebenen Algorithmus.

Schließlich benötigen wir für die Berechnung der LR-Zerlegung auch noch die Möglichkeit, Gleichungen der Form $\mathbf{R}^*\mathbf{x} = \mathbf{y}$ aufzulösen. Da $\mathbf{R}$ eine obere Dreiecksmatrix ist, ist $\mathbf{R}^*$ eine untere Dreiecksmatrix, so dass wir die Aufgabe wieder durch rekursives Vorwärtseinsetzen lösen können. Der korrespondierende Algorithmus ist in (6.11) zusammengefasst.

Es bietet sich an, das Vorwärtseinsetzen auch auf Matrixblöcke zu übertragen, um auch das ursprüngliche Problem $\widehat{\mathbf{L}}\widehat{\mathbf{X}} = \widehat{\mathbf{Y}}$ behandeln zu können. Um ähnlich wie im vorigen Fall die rechte Seite mit der Lösung überschreiben zu können verlangen wir, dass die $\mathcal{H}$-Matrizen $\widehat{\mathbf{X}}$ und $\widehat{\mathbf{Y}}$ dieselbe Struktur aufweisen, so dass ein Block $b = (t, s)$ genau dann ein zulässiges oder unzulässiges Blatt des zu $\widehat{\mathbf{X}}$ gehörenden Blockbaums ist, wenn er es bezüglich des zu $\widehat{\mathbf{Y}}$ gehörenden Baums ist.

```
procedure backward_subst_h(t, R, var x);
if sons(t) = ∅ then
  x|_t̂ ← R|^{-1}_{t̂×t̂} x|_t̂
else
  τ ← #sons(t);   {t_1, ..., t_τ} ← sons(t)
  for i = τ, ..., 1 do
    for k ∈ {i+1, ..., τ} do
      b' ← (t_i, t_k);   eval_h(−1, b', R, x, x)
    end for;
    backward_subst_h(t_i, R, x)
  end for
end if
```

Abbildung 6.10: Rückwärtseinsetzen für eine $\mathcal{H}$-Matrix $\mathbf{R}$ in oberer Blockdreiecksform zur Berechnung von $\mathbf{x} \leftarrow \mathbf{R}^{-1}\mathbf{x}$

```
procedure adjforward_subst_h(t, R, var x);
if sons(t) = ∅ then
  x|_t̂ ← (R|^*_{t̂×t})^{-1} x|_t̂
else
  τ ← #sons(t);   {t_1, ..., t_τ} ← sons(t)
  for i = 1, ..., τ do
    for k ∈ {1, ..., i−1} do
      b' ← (t_k, t_i);   adjeval_h(−1, b', R, x, x)
    end for;
    adjforward_subst_h(t_i, R, x)
  end for
end if
```

Abbildung 6.11: Vorwärtseinsetzen in die Adjungierte einer $\mathcal{H}$-Matrix $\mathbf{R}$ in oberer Blockdreiecksform zur Berechnung von $\mathbf{x} \leftarrow (\mathbf{R}^*)^{-1}\mathbf{x}$

Mit dieser Vereinfachung erhalten wir die folgende Problemformulierung: Gegeben $b = (t, s) \in \mathcal{T}_{\mathcal{I}\times\mathcal{J}}$, bestimme $\mathbf{X}|_{\hat{t}\times\hat{s}}$ so, dass

$$\mathbf{L}|_{\hat{t}\times\hat{t}}\mathbf{X}|_{\hat{t}\times\hat{s}} = \mathbf{Y}|_{\hat{t}\times\hat{s}}$$

gilt. Bei der Lösung dieses Problems müssen wir die üblichen drei Fälle unterscheiden: b kann ein zulässiges oder unzulässiges Blatt oder kein Blatt sein.

Falls b ein zulässiges Blatt ist, kennen wir eine Rang-k-Darstellung

$$\mathbf{Y}|_{\hat{t}\times\hat{s}} = \mathbf{A}\mathbf{B}^* \qquad \text{mit } \mathbf{A} \in \mathbb{K}^{\hat{t}\times k},\ \mathbf{B} \in \mathbb{K}^{\hat{s}\times k}$$

und können

$$\widehat{\mathbf{A}} := \mathbf{L}|^{-1}_{\hat{t}\times\hat{t}}\mathbf{A}$$

berechnen, indem wir den bereits diskutierten Algorithmus für das Vorwärtseinsetzen (vgl. Abbildung 6.9) auf die k Spalten der Matrix $\mathbf{A}$ anwenden. Damit folgt

$$\mathbf{X}|_{\hat{t}\times\hat{s}} = \mathbf{L}|_{\hat{t}\times\hat{t}}^{-1}\mathbf{Y}|_{\hat{t}\times\hat{s}} = \mathbf{L}|_{\hat{t}\times\hat{t}}^{-1}\mathbf{A}\mathbf{B}^* = \widehat{\mathbf{A}}\mathbf{B}^*,$$

wir haben also, wie schon im Fall der Matrix-Multiplikation, ausgenutzt, dass das Produkt einer beliebigen Matrix mit einer Rang-k-Matrix wieder eine Rang-k-Matrix ist, deren Darstellung wir aus einer Darstellung der ursprünglichen Matrix einfach gewinnen können.

Falls b ein unzulässiges Blatt ist, folgt aus der Voraussetzung, dass $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein zulässiger Blockbaum ist, dass t oder s ein Blatt des jeweiligen Clusterbaums sein muss. Ist t ein Blatt, so ist (t,t) ebenfalls ein Blatt, so dass wir

$$\mathbf{X}|_{\hat{t}\times\hat{s}} = \mathbf{L}|_{\hat{t}\times\hat{t}}^{-1}\mathbf{Y}|_{\hat{t}\times\hat{s}}$$

direkt bestimmen können. Ist dagegen s ein Blatt, so enthält die zugehörige Indexmenge $\#\hat{s} \leq r_{\mathcal{J}}$ nur wenige Elemente, so dass wir $\mathbf{X}|_{\hat{t}\times\hat{s}}$ berechnen können, indem wir den Algorithmus für das Vorwärtseinsetzen (vgl. Abbildung 6.9) auf jede Spalte anwenden.

Interessant wird es, falls b kein Blatt ist und wir deshalb eine Rekursion verwenden müssen. Für $\widehat{\mathbf{L}} = \mathbf{L}|_{\hat{t}\times\hat{t}}$ haben wir bereits in (6.9) eine Zerlegung in Teilmatrizen fixiert, so dass wir nur noch eine passende Zerlegung für $\mathbf{Y}$ benötigen. Wir setzen $\sigma := \#\operatorname{sons}(s)$ und wählen $s_1,\ldots,s_\sigma \in \mathcal{T}_{\mathcal{J}}$ mit $\operatorname{sons}(s) = \{s_1,\ldots,s_\sigma\}$. Wie üblich verwenden wir zur Abkürzung die Notationen

$$\begin{aligned}
\widehat{\mathbf{Y}} &:= \mathbf{Y}|_{\hat{t}\times\hat{s}}, & \widehat{\mathbf{Y}}_{ij} &:= \mathbf{Y}|_{\hat{t}_i\times\hat{s}_j}, & &\\
\widehat{\mathbf{X}} &:= \mathbf{X}|_{\hat{t}\times\hat{s}}, & \widehat{\mathbf{X}}_{ij} &:= \mathbf{X}|_{\hat{t}_i\times\hat{s}_j} & &\text{für alle } i\in\{1,\ldots,\tau\},\ j\in\{1,\ldots\sigma\},
\end{aligned}$$

die zu den Blockdarstellungen

$$\widehat{\mathbf{Y}} = \begin{pmatrix} \widehat{\mathbf{Y}}_{11} & \ldots & \widehat{\mathbf{Y}}_{1\sigma} \\ \vdots & \ddots & \vdots \\ \widehat{\mathbf{Y}}_{\tau 1} & \ldots & \widehat{\mathbf{Y}}_{\tau\sigma} \end{pmatrix}, \qquad \widehat{\mathbf{X}} = \begin{pmatrix} \widehat{\mathbf{X}}_{11} & \ldots & \widehat{\mathbf{X}}_{1\sigma} \\ \vdots & \ddots & \vdots \\ \widehat{\mathbf{X}}_{\tau 1} & \ldots & \widehat{\mathbf{X}}_{\tau\sigma} \end{pmatrix}$$

führt. Für eine beliebige Spalte $j\in\{1,\ldots,\sigma\}$ können wir die Gleichung

$$\begin{pmatrix} \widehat{\mathbf{L}}_{11} & & \\ \vdots & \ddots & \\ \widehat{\mathbf{L}}_{\tau 1} & \ldots & \widehat{\mathbf{L}}_{\tau\tau} \end{pmatrix} \begin{pmatrix} \widehat{\mathbf{X}}_{1j} \\ \vdots \\ \widehat{\mathbf{X}}_{\tau j} \end{pmatrix} = \begin{pmatrix} \widehat{\mathbf{Y}}_{1j} \\ \vdots \\ \widehat{\mathbf{y}}_{\tau j} \end{pmatrix}$$

wieder durch Vorwärtseinsetzen auf die Gleichungen

$$\begin{aligned}
\widehat{\mathbf{L}}_{11}\widehat{\mathbf{X}}_{1j} = \widehat{\mathbf{Y}}_{1j} \qquad &\Longleftrightarrow \qquad \widehat{\mathbf{L}}_{11}\widehat{\mathbf{X}}_{1j} = \widehat{\mathbf{Y}}_{1j},\\
\widehat{\mathbf{L}}_{21}\widehat{\mathbf{X}}_{1j} + \widehat{\mathbf{L}}_{22}\widehat{\mathbf{X}}_{2j} = \widehat{\mathbf{Y}}_{2j} \qquad &\Longleftrightarrow \qquad \widehat{\mathbf{L}}_{22}\widehat{\mathbf{X}}_{2j} = \widehat{\mathbf{Y}}_{2j} - \widehat{\mathbf{L}}_{21}\widehat{\mathbf{Y}}_{1j},\\
\vdots \qquad &\qquad\qquad\quad \vdots
\end{aligned}$$

```
procedure forward_blocksubst_h(b, L, var X);
(t, s) ← b;
if b ∈ L⁺_{I×J} then
  forward_subst_h(t, L, A_{X,b})          {Algorithmus (6.9) für jede Spalte von A_{X,b}}
else if b ∈ L⁻_{I×J} then
  forward_subst_h(t, L, X|_{t̂×ŝ})         {Algorithmus (6.9) für jede Spalte von X|_{t̂×ŝ}}
else
  τ ← #sons(t);   {t_1, ..., t_τ} ← sons(t);
  σ ← #sons(s);   {s_1, ..., s_σ} ← sons(s);
  for j ∈ {1, ..., σ} do
    for i = 1, ..., τ do
      for k = {1, ..., i − 1} do
        mul_h_h(−1, t_i, t_k, s_j, L, X, X)
      end for;
      b' ← (t_i, s_j);   forward_blocksubst_h(b', L, X)
    end for
  end for
end if
```

Abbildung 6.12: Block-Vorwärtseinsetzen für eine $\mathcal{H}$-Matrix $\mathbf{L}$ in unterer Blockdreiecksform zur Berechnung von $\mathbf{X} \leftarrow \mathbf{L}^{-1}\mathbf{X}$

$$\sum_{k=1}^{\tau-1} \widehat{\mathbf{L}}_{\tau j}\widehat{\mathbf{X}}_{kj} + \widehat{\mathbf{L}}_{\tau\tau}\widehat{\mathbf{X}}_{\tau j} = \widehat{\mathbf{Y}}_{\tau j} \qquad \Longleftrightarrow \qquad \widehat{\mathbf{L}}_{\tau\tau}\widehat{\mathbf{X}}_{\tau j} = \widehat{\mathbf{Y}}_{\tau j} - \sum_{k=1}^{\tau-1} \widehat{\mathbf{L}}_{\tau k}\widehat{\mathbf{X}}_{kj}$$

zurückführen, die sich wieder rekursiv auflösen lassen.

Der resultierende Algorithmus ist in Abbildung 6.12 dargestellt. Wie schon im Fall der Inversion ist nicht vorgeschrieben, in welcher Reihenfolge die Söhne eines Clusters durchlaufen werden, sofern diese Reihenfolge über den gesamten Algorithmus hinweg einheitlich ist: Das Vorwärtseinsetzen funktioniert selbstverständlich nur dann, wenn auch tatsächlich eine untere Dreiecksmatrix bezüglich der gewählten Reihenfolge vorliegt.

Um eine Gleichung der Form $\mathbf{XR} = \mathbf{Y}$ mit einer rechten oberen Dreiecksmatrix $\mathbf{R}$ aufzulösen, können wir entsprechend vorgehen: Für Blätter des Blockbaums wenden wir den Algorithmus aus Abbildung 6.11 im zulässigen Fall auf alle Spalten der Matrix $\mathbf{B}_{X,b}$ oder im unzulässigen Fall auf alle Zeilen der Matrix $\mathbf{X}|_{\hat{t}\times\hat{s}}$ an. Falls $b = (t, s)$ kein Blatt ist, setzen wir $\sigma := \#\operatorname{sons}(s)$ und wählen $s_1, \ldots, s_\sigma \in \mathcal{T}_\mathcal{J}$ mit $\operatorname{sons}(s) = \{s_1, \ldots, s_\sigma\}$, um dann

$$\widehat{\mathbf{R}} := \mathbf{R}|_{\hat{s}\times\hat{s}}, \qquad \widehat{\mathbf{R}}_{ij} := \mathbf{R}|_{\hat{s}_i\times\hat{s}_j} \qquad \text{für alle } i, j \in \{1, \ldots, \sigma\}$$

und die Blockdarstellung

$$\widehat{\mathbf{R}} = \begin{pmatrix} \widehat{\mathbf{R}}_{11} & \ldots & \widehat{\mathbf{R}}_{1\sigma} \\ & \ddots & \vdots \\ & & \widehat{\mathbf{R}}_{\sigma\sigma} \end{pmatrix}$$

zu erhalten. Für eine beliebige Zeile $i \in \{1, \ldots, \tau\}$ können wir die Gleichung

$$\begin{pmatrix} \widehat{\mathbf{X}}_{i1} & \ldots & \widehat{\mathbf{X}}_{i\sigma} \end{pmatrix} \begin{pmatrix} \widehat{\mathbf{R}}_{11} & \ldots & \widehat{\mathbf{R}}_{1\sigma} \\ & \ddots & \vdots \\ & & \widehat{\mathbf{R}}_{\sigma\sigma} \end{pmatrix} = \begin{pmatrix} \widehat{\mathbf{Y}}_{i1} & \ldots & \widehat{\mathbf{Y}}_{i\sigma} \end{pmatrix}$$

ebenfalls durch Vorwärtseinsetzen in die Gleichungen

$$\begin{aligned} \widehat{\mathbf{X}}_{i1}\widehat{\mathbf{R}}_{11} = \widehat{\mathbf{Y}}_{i1} \qquad &\Longleftrightarrow \qquad \widehat{\mathbf{X}}_{i1}\widehat{\mathbf{R}}_{11} = \widehat{\mathbf{Y}}_{i1}, \\ \widehat{\mathbf{X}}_{i1}\widehat{\mathbf{R}}_{12} + \widehat{\mathbf{X}}_{i2}\widehat{\mathbf{R}}_{22} = \widehat{\mathbf{Y}}_{i2} \qquad &\Longleftrightarrow \qquad \widehat{\mathbf{X}}_{i2}\widehat{\mathbf{R}}_{22} = \widehat{\mathbf{Y}}_{i2} - \widehat{\mathbf{X}}_{i1}\widehat{\mathbf{R}}_{12}, \\ \vdots \qquad &\qquad \vdots \\ \sum_{k=1}^{\sigma-1} \widehat{\mathbf{X}}_{ik}\widehat{\mathbf{R}}_{k\sigma} + \widehat{\mathbf{X}}_{i\sigma}\widehat{\mathbf{R}}_{\sigma\sigma} = \widehat{\mathbf{Y}}_{i\sigma} \qquad &\Longleftrightarrow \qquad \widehat{\mathbf{X}}_{i\sigma}\widehat{\mathbf{R}}_{\sigma\sigma} = \widehat{\mathbf{Y}}_{i\sigma} - \sum_{k=1}^{\sigma-1} \widehat{\mathbf{X}}_{ik}\widehat{\mathbf{R}}_{k\sigma} \end{aligned}$$

auflösen, indem wir sie auf eine Rekursion und Matrix-Multiplikationen zurückführen.

```
procedure adjforward_blocksubst_h(b, R, var X);
(t, s) ← b;
if b ∈ L⁺_{I×J} then
  adjforward_subst_h(s, R, B_{X,b})   {Algorithmus (6.11) für jede Spalte von B_{X,b}}
else if b ∈ L⁻_{I×J} then
  adjforward_subst_h(s, R, X|_{t̂×ŝ})  {Algorithmus (6.11) für jede Spalte von X|_{t̂×ŝ}}
else
  τ ← #sons(t);   {t_1, ..., t_τ} ← sons(t);
  σ ← #sons(s);   {s_1, ..., s_σ} ← sons(s);
  for i ∈ {1, ..., τ} do
    for j = 1, ..., σ do
      for k = {1, ..., j − 1} do
        mul_h_h(−1, t_i, s_k, s_j, X, R, X)
      end for;
      b' ← (t_i, s_j);   adjforward_blocksubst_h(b', R, X)
    end for
  end for
end if
```

Abbildung 6.13: Block-Vorwärtseinsetzen für die Adjungierte einer $\mathcal{H}$-Matrix $\mathbf{R}$ in oberer Blockdreiecksform zur Berechnung von $\mathbf{X} \leftarrow \mathbf{X}\mathbf{R}^{-1}$

Nun stehen uns alle Algorithmen zur Verfügung, die wir benötigen, um aus den Gleichungen (6.11) den Algorithmus zur Berechnung der LR-Zerlegung zu konstruieren: Die Gleichung

$$\widehat{\mathbf{L}}_{11}\widehat{\mathbf{R}}_{11} = \widehat{\mathbf{G}}_{11}$$

bedeutet, dass wir $\widehat{\mathbf{L}}_{11}$ und $\widehat{\mathbf{R}}_{11}$ aus einer LR-Zerlegung der Teilmatrix $\widehat{\mathbf{G}}_{11}$ gewinnen können, also mit Hilfe eines rekursiven Aufrufs.

Sobald uns diese Matrizen zu Verfügung stehen, können wir

$$\widehat{\mathbf{L}}_{11}\widehat{\mathbf{R}}_{1*} = \widehat{\mathbf{G}}_{1*}, \qquad \widehat{\mathbf{L}}_{11}\widehat{\mathbf{R}}_{1j} = \widehat{\mathbf{G}}_{1j} \qquad \text{für alle } j \in \{2, \ldots, \tau\}$$

durch Vorwärtseinsetzen mit Hilfe des in Abbildung 6.12 dargestellten Algorithmus auflösen, um $\widehat{\mathbf{R}}_{1j}$ für alle $j \in \{2, \ldots, \tau\}$ zu berechnen.

Entsprechend können wir

$$\widehat{\mathbf{L}}_{*1}\widehat{\mathbf{R}}_{11} = \widehat{\mathbf{G}}_{*1}, \qquad \widehat{\mathbf{L}}_{i1}\widehat{\mathbf{R}}_{11} = \widehat{\mathbf{G}}_{i1} \qquad \text{für alle } i \in \{2, \ldots, \tau\}$$

durch Vorwärtseinsetzen in die Adjungierte der Matrix $\widehat{\mathbf{R}}_{11}$ mit Hilfe des Algorithmus aus Abbildung 6.13 auflösen, um $\widehat{\mathbf{L}}_{i1}$ für alle $i \in \{2, \ldots, \tau\}$ zu bestimmen.

Schließlich müssen wir

$$\widehat{\mathbf{L}}_{*1}\widehat{\mathbf{R}}_{1*} + \widehat{\mathbf{L}}_{**}\widehat{\mathbf{R}}_{**} = \widehat{\mathbf{G}}_{**}, \qquad \widehat{\mathbf{L}}_{**}\widehat{\mathbf{R}}_{**} = \widehat{\mathbf{G}}_{**} - \widehat{\mathbf{L}}_{*1}\widehat{\mathbf{R}}_{1*}$$

auflösen. Das Schur-Komplement

$$\widehat{\mathbf{S}}_{**} := \widehat{\mathbf{G}}_{**} - \widehat{\mathbf{L}}_{*1}\widehat{\mathbf{R}}_{1*}, \qquad \widehat{\mathbf{S}}_{ij} = \widehat{\mathbf{G}}_{ij} - \widehat{\mathbf{L}}_{i1}\widehat{\mathbf{R}}_{1j} \qquad \text{für alle } i, j \in \{2, \ldots, \tau\}$$

können wir mit der bereits bekannten Matrix-Multiplikation aufstellen, seine LR-Zerlegung erfolgt dann wieder rekursiv.

Bei der praktischen Umsetzung können wir ausnutzen, dass in den Algorithmen für das Vorwärtseinsetzen die rechte Seite mit der Lösung überschrieben wird, so dass ohne weiteres Zutun jeweils $\widehat{\mathbf{G}}_{1j}$ durch $\widehat{\mathbf{R}}_{1j}$ und $\widehat{\mathbf{G}}_{i1}$ durch $\widehat{\mathbf{L}}_{i1}$ ersetzt wird. Da wir die Matrix-Multiplikation in weiser Voraussicht so formuliert haben, dass sie das Produkt zu der Ergebnismatrix addiert, können wir elegant $\widehat{\mathbf{G}}_{**}$ mit dem Schur-Komplement $\widehat{\mathbf{S}}_{**}$ überschreiben, so dass dessen LR-Zerlegung genau dort gespeichert wird, wo wir sie haben wollen.

Die Analyse der Komplexität können wir ähnlich wie im Fall der Inversion durchführen: Den Aufruf der Prozedur lr_decomp_h für den Cluster t zählen wir als Aufruf mit dem Tripel (t, t, t) im Tripelbaum, den Aufruf der Prozedur forward_blocksubst_h für den Block $b = (t, s)$ zählen wir als Aufruf mit dem Tripel (t, t, s) im Tripelbaum, den Aufruf der Prozedur adjforward_blocksubst_h für den Block $b = (s, t)$ als Aufruf mit dem Tripel (s, t, t).

Dann erfolgen in der Funktion lr_decomp_h Aufrufe für (t_k, t_k, t_k), (t_k, t_k, t_i), (t_i, t_k, t_k) und (t_i, t_k, t_j), letztere jeweils für alle $i, j \in \{k+1, \ldots, \tau\}$. In forward_blocksubst_h erfolgen Aufrufe für (t_i, t_k, s_j) mit $i \in \{k+1, \ldots, \tau\}$ sowie (t_i, t_i, s_j), in adjforward_blocksubst_h dagegen für (t_i, s_k, s_j) mit $j \in \{k+1, \ldots, \sigma\}$ sowie (t_i, s_j, s_j). Damit entspricht die rekursive Aufrufstruktur einer Teilmenge derer, die der Tripelbaum beschreibt, und wir können den Gesamtaufwand durch den Aufwand für den vollständigen Tripelbaum beschränken, um eine ähnliche Aufwandsabschätzung wie in Satz 6.20 zu erhalten.

```
procedure lr_decomp_h(t, var G);
if sons(t) = ∅ then
  Überschreibe G|_{t̂×t̂} mit seiner LR-Zerlegung
else
  τ ← #sons(t);    {t_1, ..., t_τ} ← sons(t);
  for k = 1, ..., τ do
    lr_decomp_h(t_k, G);
    for i ∈ {k+1, ..., τ} do
      b' ← (t_k, t_i);    forward_blocksubst_h(b', G, G);
      b' ← (t_i, t_k);    adjforward_blocksubst_h(b', G, G);
    end for;
    for i, j ∈ {k+1, ..., τ} do
      mul_h_h(−1, t_i, t_k, t_j, G, G, G)
    end for
  end for
end if
```

Abbildung 6.14: Berechnung der LR-Zerlegung einer $\mathcal{H}$-Matrix $\mathbf{G}$. Die linke untere Hälfte wird mit dem Faktor $\mathbf{L}$ überschrieben, die rechte obere mit $\mathbf{R}$.

Bemerkung 6.22 (Praxis) *In der Praxis wird die approximative LR-Zerlegung in wesentlich kürzerer Zeit als die Inverse berechnet, und sie ist auch in vielen Anwendungen wesentlich genauer.*

Ersteres kann man dadurch erklären, dass die LR-Zerlegung der „ersten Hälfte“ der Inversion entspricht, in der die Schur-Komplemente berechnet werden, dass aber die „zweite Hälfte“, in der sie invertiert und zur Aktualisierung des Rests der Matrix verwendet werden, fehlt. Außerdem wird die Struktur der LR-Faktoren „näher“ an der Struktur der Ausgangsmatrix $\mathbf{G}$ liegen, beispielsweise vererben sich Nulleinträge innerhalb gewisser Grenzen.

Die höhere Genauigkeit lässt sich ebenfalls dadurch erklären, dass die Fehler sich nur „in einer Richtung“ fortpflanzen: Fehler, die wir bei der Berechnung von „links oben“ stehenden Blöcken einführen, beeinflussen nur „rechts unten“ stehende Blöcke, werden aber nicht von ihnen beeinflusst.

7 Inverse partieller Differentialgleichungen

Neben der Behandlung von Integralgleichungen ist das Lösen partieller Differentialgleichungen eine wichtige Anwendung hierarchischer Matrizen. Anders als bei Integralgleichungen ist die Darstellung des zu lösenden Gleichungssystems hier kein Problem, denn die bei den typischen Finite-Elemente-Diskretisierungen auftretenden Matrizen sind schwachbesetzt, besitzen also nur wenige von null verschiedene Einträge pro Zeile, so dass sie sich elegant in einer entsprechenden Datenstruktur abspeichern lassen.

Wesentlich interessanter ist die Frage nach effizienten Lösungsverfahren für derartige Systeme: In der Regel sind die Systeme sehr groß, so dass direkte Lösungsalgorithmen wie die (exakte) LR-Zerlegung nicht gut einsetzbar sind, und sie sind schlecht konditioniert, so dass iterative Verfahren zwar wenig Speicher, aber sehr viele Schritte benötigen.

Unter bestimmten Bedingungen lassen sich zwar noch entsprechend angepasste direkte Verfahren einsetzen, und speziell entworfene iterative Verfahren (insbesondere Mehrgitteriterationen) können in bestimmten Situationen sehr effizient sein, aber beide Techniken stoßen an Grenzen.

Eine Alternative bieten hierarchische Matrizen: Es lässt sich leicht nachprüfen, dass die bei der Finite-Elemente-Diskretisierung auftretenden Matrizen auch hierarchische Matrizen mit lokalem Rang 0 sind, wenn wir eine der üblichen Zulässigkeitsbedingungen verwenden. Demzufolge können wir die Matrizen auch mit den Algorithmen aus Kapitel 6 behandeln, um ihre Inversen oder ihre LR-Faktorisierungen approximativ zu bestimmen. Beide Varianten ermöglichen es uns, das lineare Gleichungssystem durch ein äquivalentes System mit wesentlich besserer Kondition zu ersetzen, das sich sehr effizient lösen lässt.

Die zentrale Frage in diesem Kontext lautet, ob sich die Inverse einer partiellen Differentialgleichung sinnvoll durch eine hierarchische Matrix approximieren lässt. Sofern das der Fall ist, können wir die approximative Inverse berechnen und darauf hoffen, sie auch effizient abspeichern zu können.

7.1 Variationelle Formulierung

Wir fixieren ein offenes beschränktes Normalgebiet $\Omega \subseteq \mathbb{R}^d$ und eine differenzierbare Abbildung

$$\mathbf{C} : \Omega \to \mathbb{R}^{d \times d},$$

die jedem Punkt eine $d \times d$-Matrix zuordnet. Unser Interesse gilt dem partiellen Differentialoperator

$$\mathcal{L}[u] := -\sum_{i,j=1}^{d} \partial_i \left(C_{ij} \partial_j u \right),$$

der zumindest für alle $u \in C^2(\overline{\Omega})$ definiert ist.

Wir stellen uns die Aufgabe, partielle Differentialgleichungen der Form

$$\mathcal{L}[u](x) = F(x) \qquad \text{für alle } x \in \Omega$$

zu lösen. Um die Eindeutigkeit sicher zu stellen, fordern wir außerdem, dass u auf dem Rand $\partial\Omega$ des Gebiets verschwindet, dass also

$$u(x) = 0 \qquad \text{für alle } x \in \partial\Omega$$

gilt. Indem wir mit einer Funktion $v \in C^1(\overline{\Omega})$ multiplizieren, die ebenfalls auf dem Rand verschwindet, und partiell integrieren, erhalten wir die Gleichung

$$\sum_{i,j=1}^{d} \int_\Omega C_{ij}(x)\partial_i v(x)\partial_j u(x)\,dx = \int_\Omega v(x)F(x)\,dx \quad \text{für alle } v \in C^1(\overline{\Omega}) \text{ mit } v|_{\partial\Omega} = 0.$$

Diese Formulierung dient als Grundlage unserer Untersuchung. Wir vereinfachen sie zunächst, indem wir den *Gradienten*

$$\nabla u(x) := \begin{pmatrix} \partial_1 u(x) \\ \vdots \\ \partial_d u(x) \end{pmatrix} \qquad \text{für alle } x \in \Omega$$

einführen und

$$\begin{aligned}\sum_{i,j=1}^{d} C_{ij}(x)\partial_i v(x)\partial_j u(x)\,dx &= \sum_{i=1}^{d} (\nabla v(x))_i \left(\sum_{j=1}^{d} C_{ij}(x)(\nabla u(x))_j \right) \\ &= \langle \nabla v(x), \mathbf{C}(x)\nabla u(x)\rangle_2\end{aligned}$$

ausnutzen, um die kompaktere Darstellung

$$\int_\Omega \langle \nabla v(x), \mathbf{C}(x)\nabla u(x)\rangle_2\,dx = \int_\Omega v(x)F(x)\,dx \quad \text{für alle } v \in C^1(\overline{\Omega}) \text{ mit } v|_{\partial\Omega} = 0 \quad (7.1)$$

zu erhalten. In dieser Gleichung kommen wir mit schwächeren Voraussetzungen an die Differenzierbarkeit aus: Erstens treten nur die ersten Ableitungen von u und v auf, zweitens werden diese Ableitungen nicht punktweise ausgewertet, sondern aufintegriert, so dass wir auf die Voraussetzung der Stetigkeit verzichten können.

Wir werden deshalb mit *schwachen Ableitungen* arbeiten, die mit Hilfe des Raums der *Testfunktionen*

$$C_0^\infty(\Omega) := \{\varphi \in C^\infty(\Omega) \ : \ \text{der Träger von } \varphi \text{ ist kompakte Teilmenge von } \Omega\}$$

definiert werden. Insbesondere gilt für jede Testfunktion $\varphi \in C_0^\infty(\Omega)$ die Gleichung $\varphi|_{\partial\Omega} = 0$, so dass wir für ein $k \in \mathbb{N}_0$ mit partieller Integration

$$\int_\Omega u(x)\partial^\alpha \varphi(x)\,dx = (-1)^{|\alpha|} \int_\Omega \partial^\alpha u(x)\varphi(x)\,dx \qquad \text{für alle } u \in C^k(\Omega),\ \varphi \in C_0^\infty(\Omega)$$

erhalten, wobei $\alpha \in \mathbb{N}_0^d$ ein Multiindex mit Betrag $|\alpha| = \alpha_1 + \ldots + \alpha_d \leq k$ ist und $\partial^\alpha = \partial_1^{\alpha_1} \ldots \partial_d^{\alpha_d}$ den korrespondierenden Differentialoperator bezeichnet.

Diese Formel dient als Motivation für den Begriff der schwachen Ableitung:

Definition 7.1 (Schwache Ableitung) *Sei $u \in L^2(\Omega)$, sei $\alpha \in \mathbb{N}_0^d$ ein Multiindex. Falls ein $v \in L^2(\Omega)$ existiert, dass*

$$\int_\Omega u(x)\partial^\alpha \varphi(x)\,dx = (-1)^{|\alpha|}\int_\Omega v(x)\varphi(x)\,dx \qquad \text{für alle } \varphi \in C_0^\infty(\Omega)$$

erfüllt, bezeichnen wir v als die schwache α-te Ableitung von u *und notieren die Funktion als $\partial^\alpha u$.*

Diese Notation führt nicht zu Missverständnissen, da sie, wie wir bereits gesehen haben, für klassisch differenzierbare Funktionen mit den klassischen Ableitungen übereinstimmt.

Da Linearkombinationen schwach differenzierbarer Funktionen wieder schwach differenzierbar sind, können wir entsprechend $C^k(\Omega)$ auch Räume schwach differenzierbarer Funktionen definieren:

Definition 7.2 (Sobolew-Raum) *Sei $k \in \mathbb{N}_0$. Dann ist*

$$H^k(\Omega) := \{u \in L^2(\Omega) \;:\; \text{für alle } \alpha \in \mathbb{N}_0^d \text{ mit } |\alpha| \leq k \text{ existiert } \partial^\alpha u \in L^2(\Omega)\}$$

ein Hilbertraum mit dem Skalarprodukt

$$\langle u, v\rangle_{H^k(\Omega)} := \sum_{|\alpha|\leq k} \langle \partial^\alpha u, \partial^\alpha v\rangle_{L^2(\Omega)} \qquad \text{für alle } u, v \in H^k(\Omega)$$

und der korrespondierenden Hilbertraum-Norm

$$\|u\|_{H^k(\Omega)} := \sqrt{\langle u, u\rangle_{H^k(\Omega)}} \qquad \text{für alle } u \in H^k(\Omega).$$

Diesen Raum nennen wir den k-ten Sobolew-Raum.

Nun können wir die Gleichung (7.1) in eine für unsere Untersuchung geeignete Form bringen: Wir setzen vorläufig $V := H^1(\Omega)$, definieren

$$a(v,u) := \int_\Omega \langle \nabla v(x), \mathbf{C}(x)\nabla u(x)\rangle_2\,dx \qquad \text{für alle } u, v \in V, \tag{7.2a}$$

$$f(v) := \int_\Omega v(x)F(x)\,dx \qquad \text{für alle } v \in V \tag{7.2b}$$

und suchen nach einem $u \in V$, dass die *Variationsgleichung*

$$a(v,u) = f(v) \qquad \text{für alle } v \in V \tag{7.3}$$

erfüllt. Zu klären sind Lösbarkeit sowie Eindeutigkeit und Stabilität der Lösung.

Da $H^k(\Omega)$ ein Hilbertraum ist, können wir den Satz von Riesz verwenden, um die Lösung von Variationsaufgaben zu berechnen. Allerdings müssen wir dazu nachweisen, dass die von uns untersuchte Bilinearform $a(\cdot,\cdot)$ ein Skalarprodukt ist, das eine zu der Sobolew-Norm $\|\cdot\|_{H^k(\Omega)}$ äquivalente Hilbertraum-Norm induziert.

Es lässt sich einfach feststellen, dass das nur möglich ist, falls die Koeffizientenmatrizen $\mathbf{C}(x)$ in jedem Punkt des Gebiets mindestens positiv definit sind. Tatsächlich müssen sie sogar gleichmäßig positiv definit sein, es müssen also $\alpha, \beta \in \mathbb{R}_{>0}$ mit

$$\alpha\|\mathbf{y}\|_2^2 \leq \langle \mathbf{y}, \mathbf{C}(x)\mathbf{y}\rangle_2 \leq \beta\|\mathbf{y}\|_2^2 \qquad \text{für alle } \mathbf{y} \in \mathbb{R}^d,\ x \in \Omega \tag{7.4}$$

existieren. Damit $a(\cdot,\cdot)$ auch symmetrisch ist, muss diese Eigenschaft von den Koeffizientenmatrizen geteilt werden, es muss also

$$\mathbf{C}(x) = \mathbf{C}(x)^* \qquad \text{für alle } x \in \Omega \tag{7.5}$$

gelten. Unter diesen beiden Voraussetzungen induziert $a(\cdot,\cdot)$ immerhin schon eine *Halbnorm*, die zu dem Spezialfall $k = 1$ der Halbnorm

$$|u|_{H^k(\Omega)} := \sqrt{\sum_{|\alpha|=k} \langle \partial^\alpha u, \partial^\alpha u\rangle_{L^2(\Omega)}} \qquad \text{für alle } u \in H^k(\Omega)$$

äquivalent ist:

Lemma 7.3 (Normäquivalenz) *Es gilt*

$$\alpha|u|_{H^1(\Omega)}^2 \leq a(u,u) \leq \beta|u|_{H^1(\Omega)}^2 \qquad \textit{für alle } u \in H^1(\Omega).$$

Beweis. Sei $u \in H^1(\Omega)$. Es gilt offenbar

$$|u|_{H^1(\Omega)}^2 = \sum_{|\alpha|=1} \|\partial^\alpha u\|_{L^2(\Omega)}^2 = \int_\Omega \sum_{|\alpha|=1} |\partial^\alpha u(x)|^2\,dx = \int_\Omega \|\nabla u(x)\|_2^2\,dx,$$

also erhalten wir aus (7.4)

$$\begin{aligned} a(u,u) &= \int_\Omega \langle \nabla u(x), \mathbf{C}(x)\nabla u(x)\rangle_2\,dx \geq \int_\Omega \alpha\|\nabla u(x)\|_2^2\,dx = \alpha|u|_{H^1(\Omega)}^2, \\ a(u,u) &= \int_\Omega \langle \nabla u(x), \mathbf{C}(x)\nabla u(x)\rangle_2\,dx \leq \int_\Omega \beta\|\nabla u(x)\|_2^2\,dx = \beta|u|_{H^1(\Omega)}^2, \end{aligned}$$

und das sind die gewünschten Abschätzungen. ∎

Offenbar können wir die Halbnorm $|\cdot|_{H^k(\Omega)}$ problemlos nach oben durch $\|\cdot\|_{H^k(\Omega)}$ abschätzen, eine Abschätzung nach unten ist dagegen schwieriger, weil beispielsweise eine von null verschiedene konstante Funktion im Kern der Halbnorm liegt, aber nicht im Kern der Norm. Hier helfen uns die Randbedingungen: Die Forderung $u|_{\partial\Omega} = 0$ lässt sich im Fall der schwach differenzierbaren Funktionen wie folgt fassen:

Definition 7.4 (Nullrandbedingungen) *Sei $k \in \mathbb{N}_0$. Wir definieren*

$$H_0^k(\Omega) := \overline{C_0^\infty(\Omega)}^{H^k(\Omega)},$$

der Raum $H_0^k(\Omega)$ ist also der Abschluss des Raums der Testfunktionen unter der Norm $H^k(\Omega)$.

Diese Definition ist naheliegend: Die Testfunktionen verschwinden auf dem Rand $\partial\Omega$, und wir dürfen davon ausgehen, dass sich diese Eigenschaft unter dem Abschluss in geeigneter Form erhält.

Auf dem Teilraum $H_0^k(\Omega)$ sind die Halbnorm $|\cdot|_{H^k(\Omega)}$ und die Norm $\|\cdot\|_{H^k(\Omega)}$ äquivalent, denn aus der *Poincaré-Friedrichs-Ungleichung* folgt

$$C_\Omega \|u\|_{H^k(\Omega)}^2 \leq |u|_{H^k(\Omega)}^2 \qquad \text{für alle } u \in H_0^k(\Omega)$$

mit einer nur von Ω abhängenden Konstanten $C_\Omega \in \mathbb{R}_{>0}$.

Indem wir die Variationsgleichung (7.3) auf dem Raum $V := H_0^1(\Omega)$ statt auf $H^1(\Omega)$ untersuchen, können wir diese Ungleichung ausnutzen, um

$$\alpha C_\Omega \|u\|_{H^1(\Omega)}^2 \leq \alpha |u|_{H^1(\Omega)}^2 \leq a(u,u) \leq \beta |u|_{H^1(\Omega)}^2 \leq \beta \|u\|_{H^1(\Omega)}^2 \qquad \text{für alle } u \in H_0^1(\Omega)$$

zu erhalten, also induziert die Bilinearform $a(\cdot,\cdot)$ auf dem Teilraum $H_0^1(\Omega)$ eine zu $\|\cdot\|_{H^1(\Omega)}$ äquivalente Norm.

Nun können wir uns wieder der Frage der Lösbarkeit der Differentialgleichung zuwenden: Wie wir bereits gesehen haben, erfüllt eine klassische Lösung der partiellen Differentialgleichung auch diese Gleichung, falls keine klassische Lösung existieren sollte, erhalten wir eine *schwache Lösung* im Sinne der schwachen Ableitungen, indem wir den Satz von Riesz anwenden. Wir bezeichnen mit V' den Dualraum von V, also den Raum der stetigen Funktionale, und mit

$$\|g\|_{V'} := \sup_{v \in V \setminus \{0\}} \frac{|g(v)|}{\|v\|_V} \qquad \text{für alle } g \in V'$$

die auf ihm definierte Dualnorm. Damit erhalten wir die folgende Aussage:

Satz 7.5 (Lösbarkeit) *Unter den Voraussetzungen (7.4) und (7.5) existiert für jedes Funktional $F \in V'$ genau ein $u \in V$, das (7.3) erfüllt. Für dieses u gilt*

$$\|u\|_V \leq \frac{1}{C_\Omega \alpha} \|f\|_{V'}.$$

Beweis. Sei $f \in V'$. Wir definieren die *Energienorm*

$$\|u\|_A := \sqrt{a(u,u)} \qquad \text{für alle } u \in V$$

und erinnern uns daran, dass wir bereits

$$\alpha C_\Omega \|u\|_V^2 \leq \|u\|_A^2 \leq \beta \|u\|_V^2 \qquad \text{für alle } u \in V$$

bewiesen haben. Also ist insbesondere V mit dem Skalarprodukt $a(\cdot,\cdot)$ ein Hilbertraum, und wir können den Satz von Riesz anwenden, um ein $u \in V$ mit

$$a(v,u) = f(v) \qquad \text{für alle } v \in V$$

zu finden. Nach Satz ist dieses u eindeutig bestimmt und erfüllt die Gleichung

$$\|u\|_A = \sup_{v \in V\setminus\{0\}} \frac{|f(v)|}{\|v\|_A}.$$

Mit der Normäquivalenz erhalten wir

$$\|u\|_V^2 \leq \frac{1}{\alpha C_\Omega}\|u\|_A^2 = \frac{1}{\alpha C_\Omega} \sup_{v\in V\setminus\{0\}} \frac{|f(v)|^2}{\|v\|_A^2} \leq \frac{1}{\alpha C_\Omega} \sup_{v\in V\setminus\{0\}} \frac{|f(v)|^2}{\alpha C_\Omega \|v\|_V^2} = \frac{1}{\alpha^2 C_\Omega^2}\|f\|_{V'}^2,$$

also die gesuchte obere Schranke für u. ∎

Die Aussage dieses Satzes lässt sich kompakter formulieren, indem wir den Operator

$$L : V \to V', \qquad u \mapsto a(\cdot,u),$$

einführen: Die Variationsgleichung (7.3) ist äquivalent zu

$$Lu = f,$$

und Satz 7.5 impliziert gerade, dass

$$L^{-1} : V' \to V$$

existiert und wegen

$$\|L^{-1}f\|_V = \|u\|_V \leq \frac{1}{\alpha C_\Omega}\|f\|_{V'} \qquad \text{für alle } F \in V'$$

auch stetig ist. Mit Hilfe der Cauchy-Schwarz-Ungleichung erhalten wir auch

$$a(v,u) \leq \sqrt{a(v,v)}\sqrt{a(u,u)} \leq \beta\|v\|_V\|u\|_V \qquad \text{für alle } u,v \in H_0^1(\Omega),$$

also insbesondere

$$\|Lu\|_{V'} = \sup_{v\in V\setminus\{0\}} \frac{|Lu(v)|}{\|v\|_V} = \sup_{v\in V\setminus\{0\}} \frac{|a(v,u)|}{\|v\|_V} \leq \beta\|u\|_V \qquad \text{für alle } u \in V,$$

so dass wir zusammenfassend

$$\|L\|_{V'\leftarrow V} \leq \beta, \qquad \|L^{-1}\|_{V\leftarrow V'} \leq \frac{1}{\alpha C_\Omega} \tag{7.6}$$

feststellen können. L ist also ein stetiger linearer Operator mit stetiger Inversen, kurz ein Isomorphismus zwischen dem Raum V und seinem Dualraum V'.

7.2 Diskretisierung

Um die Lösung u der Variationsgleichung (7.3) zu approximieren, kommen häufig *Galerkin-Verfahren* zum Einsatz. Sie beruhen auf der Wahl eines endlich-dimensionalen Teilraums $V_h \subseteq V$, auf den die Variationsgleichung eingeschränkt wird: Wir suchen nun ein $u_h \in V_h$, das

$$a(v_h, u_h) = f(v_h) \qquad \text{für alle } v_h \in V_h$$

erfüllt. Die Bilinearform a erfüllt auf dem Teilraum V_h von V selbstverständlich dieselben Abschätzungen, die sie auf dem größeren Raum erfüllt, so dass auch diese eingeschränkte Variationsgleichung eine eindeutig bestimmte Lösung u_h besitzt.

Um den Raum V_h konkret handhaben zu können, wählen wir eine Basis $(\varphi_i)_{i\in\mathcal{I}}$. Dann ist die obige Variationsgleichung äquivalent zu

$$a(\varphi_i, u_h) = f(\varphi_i) \qquad \text{für alle } i \in \mathcal{I}.$$

Die Lösung u_h stellen wir in der Basis dar, wir verwenden also einen Koeffizientenvektor $\mathbf{x} \in \mathbb{R}^{\mathcal{I}}$ mit

$$u_h = \sum_{j\in\mathcal{I}} x_j \varphi_j.$$

Mit dieser Gleichung lässt sich die Variationsgleichung als

$$\sum_{j\in\mathcal{I}} x_j a(\varphi_i, \varphi_j) = a(\varphi_i, u_h) = f(\varphi_i) \qquad \text{für alle } i \in \mathcal{I}$$

schreiben, so dass wir mit der durch

$$A_{ij} := a(\varphi_i, \varphi_j) \qquad \text{für alle } i, j \in \mathcal{I}$$

definierten Matrix $\mathbf{A} \in \mathbb{R}^{\mathcal{I}\times\mathcal{I}}$ und dem durch

$$b_i := f(\varphi_i) \qquad \text{für alle } i \in \mathcal{I}$$

definierten Vektor $\mathbf{b} \in \mathbb{R}^{\mathcal{I}}$ schließlich das lineare Gleichungssystem

$$\mathbf{A}\mathbf{x} = \mathbf{b} \tag{7.7}$$

erhalten. Aus (7.4) folgt, dass $\mathbf{A}$ positiv definit ist, dass sich also dieses System für alle $\mathbf{b}$ eindeutig lösen lässt, um $\mathbf{x}$ und damit auch u_h zu bestimmen.

Es gibt eine große Anzahl von Verfahren, mit denen sich derartige lineare Gleichungssystem lösen lassen, die meisten unter ihnen stoßen allerdings an Grenzen, falls die Dimension $\#\mathcal{I}$ sehr groß wird oder falls die Koeffizientenmatrix $\mathbf{C}(x)$ sehr unterschiedliche Eigenwerte besitzt oder sich in Abhängigkeit von der Ortsvariablen x schnell ändert.

Hierarchische Matrizen bieten eine Möglichkeit, diese Probleme zu umgehen: Nach Definition gilt $a(\varphi_i, \varphi_j) = 0$, sofern die Träger der Basisfunktionen φ_i und φ_j disjunkt

sind. Falls wir in der üblichen Weise einen Clusterbaum $\mathcal{T}_\mathcal{I}$ und einen Blockbaum $\mathcal{T}_{\mathcal{I}\times\mathcal{I}}$ konstruieren, folgt bei zulässigen Blöcken $b = (t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{I}}$ aus

$$\operatorname{dist}(\Omega_t, \Omega_s) \geq \operatorname{diam}(\Omega_t) > 0$$

insbesondere auch $\mathbf{A}|_{\hat{t}\times\hat{s}} = \mathbf{0}$, jeder zulässige Block der Matrix $\mathbf{A}$ besitzt also Rang 0. Also können wir $\mathbf{A}$ exakt als hierarchische Matrix mit lokalem Rang 0 darstellen.

Es liegt nun nahe, zur Lösung der Gleichung (7.7) auf die $\mathcal{H}$-Matrix-Arithmetik zurückzugreifen, indem wir die Inverse $\mathbf{A}^{-1}$ oder die LR-Zerlegung $\mathbf{A} = \mathbf{LR}$ approximieren. Dazu müssen wir klären, ob die Inverse sich überhaupt durch eine hierarchische Matrix approximieren lässt, ob also $\mathbf{A}^{-1}|_{\hat{t}\times\hat{s}}$ durch eine Matrix niedrigen Rangs angenähert werden kann, wenn $b = (t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{I}}$ ein zulässiger Block ist.

7.3 Blockweise Approximation des Lösungsoperators

Unsere Aufgabe besteht darin, zu beweisen, dass sich für jedes $b = (t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{I}}$ der Block $\mathbf{A}^{-1}|_{\hat{t}\times\hat{s}}$ durch eine Matrix niedrigen Rangs approximieren lässt, dass also für jedes $\epsilon \in \mathbb{R}_{>0}$ ein möglichst kleines $k \in \mathbb{N}$ sowie Matrizen $\mathbf{X} \in \mathbb{K}^{\hat{t}\times k}$, $\mathbf{Y} \in \mathbb{K}^{\hat{s}\times k}$ mit

$$\|\mathbf{A}^{-1}|_{\hat{t}\times\hat{s}} - \mathbf{X}\mathbf{Y}^*\|_2 \leq \epsilon$$

existieren. Nach Definition der Spektralnorm genügt es dazu, die Abschätzung

$$\|\mathbf{A}^{-1}|_{\hat{t}\times\hat{s}}\mathbf{b} - \mathbf{X}\mathbf{Y}^*\mathbf{b}\|_2 \leq \epsilon\|\mathbf{b}\|_2 \qquad \text{für alle } \mathbf{b} \in \mathbb{K}^{\hat{s}}$$

zu zeigen. Wir können $\mathbf{b}$ durch null fortsetzen, um die äquivalente Bedingung

$$\|(\mathbf{A}^{-1}\mathbf{b})|_{\hat{t}} - \mathbf{X}\mathbf{Y}^*\mathbf{b}|_{\hat{s}}\|_2 \leq \epsilon\|\mathbf{b}\|_2 \qquad \text{für alle } \mathbf{b} \in \mathbb{K}^{\mathcal{I}} \text{ mit } \operatorname{supp}\mathbf{b} \subseteq \hat{s}$$

zu erhalten. Unsere Aufgabe besteht also darin, die Lösung $\mathbf{A}^{-1}\mathbf{b}$ in den zu $\hat{t}$ gehörenden Indizes zu approximieren.

Zur weiteren Vereinfachung formulieren wir das Problem als Approximationsaufgabe in Vektorräumen:

Lemma 7.6 (Teilräume) *Sei $\mathbf{G} \in \mathbb{K}^{\mathcal{I}\times\mathcal{J}}$, sei $b = (t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$. Sei $\mathcal{V} \subseteq \mathbb{K}^{\hat{t}}$ ein k-dimensionaler Teilraum, der die Eigenschaft*

$$\min_{\mathbf{y}\in\mathcal{V}} \|(\mathbf{G}\mathbf{x})|_{\hat{t}} - \mathbf{y}\|_2 \leq \epsilon\|\mathbf{x}\|_2 \qquad \textit{für alle } \mathbf{x} \in \mathbb{K}^{\mathcal{J}} \textit{ mit } \operatorname{supp}\mathbf{x} \subseteq \hat{s} \tag{7.8}$$

besitzt. Dann existieren Matrizen $\mathbf{X} \in \mathbb{K}^{\hat{t}\times k}$ und $\mathbf{Y} \in \mathbb{K}^{\hat{s}\times k}$ mit

$$\|\mathbf{G}|_{\hat{t}\times\hat{s}} - \mathbf{X}\mathbf{Y}^*\|_2 \leq \epsilon.$$

Beweis. Wir wählen eine orthonormale Basis des Raums $\mathcal{V}$ und fassen die k Basisvektoren in den Spalten einer Matrix $\mathbf{X} \in \mathbb{K}^{\hat{t} \times k}$ zusammen. Dann gelten $\operatorname{Bild} \mathbf{X} = \mathcal{V}$ und $\mathbf{X}^* \mathbf{X} = \mathbf{I}$. Insbesondere ist $\mathbf{X}\mathbf{X}^*$ eine orthogonale Projektion auf $\mathcal{V}$, so dass

$$\|(\mathbf{G}\mathbf{x})|_{\hat{t}} - \mathbf{X}\mathbf{X}^*(\mathbf{G}\mathbf{x})|_{\hat{t}}\|_2 = \min_{\mathbf{y} \in \mathcal{V}} \|(\mathbf{G}\mathbf{x})|_{\hat{t}} - \mathbf{y}\|_2 \qquad \text{für alle } \mathbf{x} \in \mathbb{K}^{\mathcal{J}} \text{ mit } \operatorname{supp} \mathbf{x} \subseteq \hat{s}$$

gilt. Aus $\operatorname{supp} \mathbf{x} \subseteq \hat{s}$ in der obigen Gleichung folgt

$$\|\mathbf{G}|_{\hat{t} \times \hat{s}} \mathbf{x} - \mathbf{X}\mathbf{X}^* \mathbf{G}|_{\hat{t} \times \hat{s}} \mathbf{x}\|_2 \leq \epsilon \|\mathbf{x}\|_2 \qquad \text{für alle } \mathbf{x} \in \mathbb{K}^{\hat{s}},$$

also insbesondere auch

$$\|\mathbf{G}|_{\hat{t} \times \hat{s}} - \mathbf{X}\mathbf{X}^* \mathbf{G}|_{\hat{t} \times \hat{s}}\|_2 \leq \epsilon.$$

Indem wir $\mathbf{Y} := \mathbf{G}|^*_{\hat{t} \times \hat{s}} \mathbf{X}$ setzen erhalten wir die gewünschte Aussage. ∎

Damit haben wir eine Formulierung des Problems gefunden, die sich für eine Analyse eignet: Wir suchen nach einem Teilraum $\mathcal{V} \subseteq \mathbb{K}^{\hat{t}}$ möglichst niedriger Dimension $k \in \mathbb{N}$ derart, dass

$$\min_{\mathbf{y} \in \mathcal{V}} \|(\mathbf{A}^{-1}\mathbf{b})|_{\hat{t}} - \mathbf{y}\|_2 \leq \epsilon \|\mathbf{b}\|_2 \qquad \text{für alle } \mathbf{b} \in \mathbb{K}^{\mathcal{J}} \text{ mit } \operatorname{supp} \mathbf{b} \subseteq \hat{s} \tag{7.9}$$

gilt. Von Interesse ist also das Verhalten der Lösung zu einer rechten Seite $\mathbf{b}$, die nur auf $\hat{s}$ von null verschieden ist, auf einer Indexmenge $\hat{t}$.

Beispiel 7.7 (Tridiagonalmatrizen) *Besonders einfach lässt sich das Verhalten der Lösung auf niedrig-dimensionalen Teilräumen für den Spezialfall analysieren, dass* $\mathbf{A}$ *eine Tridiagonalmatrix ist:*

Sei $n \in \mathbb{N}$ *und* $\mathcal{I} := \{1, \ldots, n\}$. *Wir untersuchen eine positiv definite Tridiagonalmatrix*

$$\mathbf{A} = \begin{pmatrix} d_1 & u_1 & & \\ l_1 & \ddots & \ddots & \\ & \ddots & \ddots & u_{n-1} \\ & & l_{n-1} & d_n \end{pmatrix}.$$

Seien $\alpha, \beta \in \{1, \ldots, n\}$ *mit* $\alpha \leq \beta$ *fixiert. Um eine Sonderbehandlung von Randpunkten zu vermeiden setzen wir* $1 < \alpha$ *und* $\beta < n$ *voraus.*

Wir untersuchen die zusammenhängende Teilmenge $\hat{t} := \{\alpha, \alpha + 1, \ldots, \beta - 1, \beta\} \subseteq \mathcal{I}$ *und eine beliebige zweite Teilmenge* $\hat{s} \subseteq \mathcal{I}$, *die die (sehr schwache) Zulässigkeitsbedingung* $\hat{t} \cap \hat{s} = \emptyset$ *erfüllen soll.*

Sei ein Vektor $\mathbf{b} \in \mathbb{K}^{\mathcal{I}}$ *mit* $\operatorname{supp} \mathbf{b} \subseteq \hat{s}$ *fixiert. Sei* $\mathbf{x} = \mathbf{A}^{-1}\mathbf{b} \in \mathbb{K}^{\mathcal{I}}$ *die Lösung des Gleichungssystems* $\mathbf{A}\mathbf{x} = \mathbf{b}$. *Indem wir uns die einzelnen Zeilen dieses Systems näher anschauen erhalten wir*

$$b_i = l_{i-1} x_{i-1} + d_i x_i + u_i x_{i+1} \qquad \text{für alle } i \in \hat{t},$$

und aus diesen Gleichungen können wir das Gleichungssystem

$$\begin{pmatrix} d_\alpha & u_\alpha & & \\ l_\alpha & \ddots & \ddots & \\ & \ddots & \ddots & u_{\beta-1} \\ & & l_{\beta-1} & d_\beta \end{pmatrix} \begin{pmatrix} x_\alpha \\ x_{\alpha+1} \\ \vdots \\ x_{\beta-1} \\ x_\beta \end{pmatrix} = \begin{pmatrix} b_\alpha - l_{\alpha-1}x_{\alpha-1} \\ b_{\alpha+1} \\ \vdots \\ b_{\beta-1} \\ b_\beta - u_\beta x_{\beta+1} \end{pmatrix} = \begin{pmatrix} -l_{\alpha-1}x_{\alpha-1} \\ 0 \\ \vdots \\ 0 \\ -u_\beta x_{\beta+1} \end{pmatrix}$$

gewinnen, da $b_i = 0$ für alle $i \in \hat{t}$ nach Voraussetzung gilt.

Da $\mathbf{A}$ als positiv definit vorausgesetzt ist, ist dieses System eindeutig lösbar, also ist $\mathbf{x}|_{\hat{t}}$ bereits durch die beiden Zahlen $x_{\alpha-1}$ und $x_{\beta+1}$ eindeutig bestimmt. Also liegt $\mathbf{x}|_{\hat{t}}$ in einem höchstens zweidimensionalen Raum, der durch Vektoren $\mathbf{v}, \mathbf{w} \in \mathbb{K}^{\hat{t}}$ aufgespannt wird, die durch

$$\begin{pmatrix} d_\alpha & u_\alpha & & \\ l_\alpha & \ddots & \ddots & \\ & \ddots & \ddots & u_{\beta-1} \\ & & l_{\beta-1} & d_\beta \end{pmatrix} \begin{pmatrix} v_\alpha \\ v_{\alpha+1} \\ \vdots \\ v_{\beta-1} \\ v_\beta \end{pmatrix} = \begin{pmatrix} -l_{\alpha-1} \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix},$$

$$\begin{pmatrix} d_\alpha & u_\alpha & & \\ l_\alpha & \ddots & \ddots & \\ & \ddots & \ddots & u_{\beta-1} \\ & & l_{\beta-1} & d_\beta \end{pmatrix} \begin{pmatrix} w_\alpha \\ w_{\alpha+1} \\ \vdots \\ w_{\beta-1} \\ w_\beta \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ -u_\beta \end{pmatrix}$$

eindeutig definiert sind. Damit haben wir bewiesen, dass die Inverse jeder positiv definiten Tridiagonalmatrix sich exakt *durch eine $\mathcal{H}$-Matrix mit einem lokalem Rang von höchstens $k = 2$ darstellen lässt.*

Dieses Beispiel lässt sich erweitern: Sei $\mathbf{A} \in \mathbb{K}^{\mathcal{I}\times\mathcal{I}}$ eine beliebige positiv definite Matrix. Wenn wir den *Rand* $\partial\hat{t}$ einer Menge $\hat{t}$ durch

$$\partial\hat{t} := \{i \in \mathcal{I} \setminus \hat{t} \ : \ \text{es existiert } j \in \hat{t} \text{ mit } A_{ij} \neq 0 \text{ oder } A_{ji} \neq 0\}$$

definieren, können wir uns in ähnlicher Weise wie oben überlegen, dass $\mathbf{x}|_{\hat{t}}$ in einem Raum der Dimension $\#\partial\hat{t}$ liegen muss.

Unglücklicherweise wird $\#\partial\hat{t}$ gerade bei großen Clustern zu groß sein, um sich praktisch anwenden zu lassen. Einen Ausweg können wir allerdings finden, wenn wir statt der sehr schwachen Zulässigkeitsbedingung $\hat{t} \cap \hat{s} = \emptyset$ die gängigere Bedingung

$$\operatorname{diam}(\tau) \leq 2\eta \operatorname{dist}(\tau, \sigma)$$

verwenden, wobei $\tau, \sigma \subseteq \Omega$ konvexe Teilgebiete sind, die

$$\operatorname{supp}\varphi_i \subseteq \tau, \qquad \operatorname{supp}\varphi_j \subseteq \sigma \qquad \text{für alle } i \in \hat{t},\ j \in \hat{s}$$

erfüllen. Eine naheliegende Wahl ist es, die überdeckenden Quader (vgl. Definition 2.11) als τ und σ zu verwenden, die wir ohnehin für die effiziente Überprüfung der Zulässigkeitsbedingung benötigen.

In diesem Fall können wir uns nämlich auf die Analyse des ursprünglichen unendlichdimensionalen Problems zurückziehen: Wir übersetzen (7.9) in die Aufgabe, einen Raum $\mathcal{V} \subseteq L^2(\tau)$ möglichst niedriger Dimension zu finden, der

$$\min_{w \in \mathcal{V}} \|(L^{-1}f)|_\tau - w\|_{L^2(\tau)} \leq \epsilon \|f\|_{V'} \qquad \text{für alle } f \in V' \text{ mit } \operatorname{supp} f \subseteq \sigma \tag{7.10}$$

erfüllt. Dabei ist der Träger $\operatorname{supp} f$ eines Funktionals $f \in V'$ definiert als die größte abgeschlossene Teilmenge ω des Gebiets Ω, die die Eigenschaft

$$f(v) = 0 \qquad \text{für alle } v \in V \text{ mit } \operatorname{supp} v \cap \omega = \emptyset$$

besitzt. Falls f wie in (7.2b) aus einer Funktion F hervorgeht, stimmt der Träger des Funktionals f mit dem der Funktion F überein.

Sei $f \in V'$, und sei $u := L^{-1}f$. Unsere Aufgabe ist es, $(L^{-1}f)|_\tau = u|_\tau$ durch ein Element eines Raums möglichst niedriger Dimension zu approximieren. Wir wissen, dass $u \in V \subseteq H^1(\Omega)$ gilt, dass also insbesondere auch $u|_\tau \in H^1(\tau)$ erfüllt ist. Für derartige Funktionen lässt sich eine einfache Approximationsaussage mit Hilfe des *Lemmas von Poincaré* entwickeln.

Lemma 7.8 (Poincaré) *Es existiert eine Konstante $C_{apx} \in \mathbb{R}_{>0}$ so, dass für alle konvexen Gebiete $\omega \subseteq \mathbb{R}^d$ und alle Funktionen $u \in H^1(\omega)$ die Abschätzung*

$$\|u - u_0\|_{L^2(\omega)} \leq C_{apx} \operatorname{diam}(\omega) \|\nabla u\|_{L^2(\omega)}, \qquad u_0 := \frac{1}{|\omega|} \int_\omega u(x)\, dx$$

gilt, sich also u durch seinen Mittelwert u_0 approximieren lässt.

Beweis. siehe etwa [37, Theorem 7.6] ∎

Um mit Hilfe dieses Lemmas eine hinreichende Genauigkeit zu erreichen, muss der Durchmesser des Gebiets ω klein im Vergleich zu dem Gradienten der Funktion u sein. Diese Bedingung stellt man in der Regel sicher, indem man zu Teilgebieten übergeht: Wir wählen $\ell \in \mathbb{N}$ und schließen ω in einen achsenparallelen Quader $Q = [a_1, b_1] \times \ldots \times [a_d, b_d]$ ein. Mit

$$c_{\iota\nu} := \frac{\ell - \nu}{\ell} a_\iota + \frac{\nu}{\ell} b_\iota \qquad \text{für alle } \iota \in \{1, \ldots, d\},\ \nu \in \{0, \ldots, \ell\}$$

können wir Q in Teilquader

$$Q_\nu := (c_{1,\nu_1 - 1}, c_{1,\nu_1}) \times \ldots \times (c_{d,\nu_d - 1}, c_{d,\nu_d}) \qquad \text{für alle } \nu \in \{1, \ldots, \ell\}^d$$

zerlegen, die

$$Q = \bigcup_{\nu \in \{1,\ldots,\ell\}^d} \overline{Q_\nu}$$

erfüllen, also abgesehen von einer Nullmenge eine disjunkte Partition des Quaders Q bilden. Nach Konstruktion gilt

$$c_{\iota,\nu_\iota} - c_{\iota,\nu_\iota-1} = \frac{b_\iota - a_\iota}{\ell} \qquad \text{für alle } \iota \in \{1,\dots,d\},\ \nu \in \{1,\dots,\ell\}^d,$$
$$\operatorname{diam}(Q_\nu) = \frac{\operatorname{diam}(Q)}{\ell} \qquad \text{für alle } \nu \in \{1,\dots,\ell\}^d,$$

so dass wir durch Wahl eines hinreichend großen Werts für ℓ die Durchmesser der Teilgebiete beliebig verkleinern können.

Wir sind an einer Approximation einer Funktion $u \in H^1(\omega)$ interessiert. Zu ihrer Konstruktion benutzen wir die Teilmengen

$$\omega_\nu := \omega \cap Q_\nu \qquad \text{für alle } \nu \in \{1,\dots,\ell\}^d,$$

die als Schnitte zweier konvexen Mengen wieder konvex sind und als Teilmengen von Q_ν die Eigenschaft

$$\operatorname{diam}(\omega_\nu) \leq \frac{\operatorname{diam}(Q)}{\ell} \qquad \text{für alle } \nu \in \{1,\dots,\ell\}^d$$

besitzen. Um Lemma 7.8 auf diese Teilgebiete anwenden zu können, müssen wir sicherstellen, dass $|\omega_\nu| > 0$ gilt, also setzen wir

$$N := \{\nu \in \{1,\dots,\ell\}^d \ :\ |\omega_\nu| > 0\}$$

und stellen fest, dass

$$\overline{\omega} = \bigcup_{\nu\in N} \overline{\omega_\nu}$$

gilt, wir also wieder, abgesehen von einer Nullmenge, eine disjunkte Partition konstruiert haben. Mit Hilfe dieser Partition können wir nun eine erste Approximation der Funktion u konstruieren:

Lemma 7.9 (Stückweise konstante Approximation) *Für jede konvexe Menge $\omega \subseteq \mathbb{R}^d$, jedes $u \in H^1(\omega)$ und jedes $\ell \in \mathbb{N}$ erfüllt die durch*

$$v|_{\omega_\nu} = \frac{1}{|\omega_\nu|}\int_{\omega_\nu} u(x)\,dx \qquad \textit{für alle } \nu \in N$$

bis auf Nullmengen definierte stückweise konstante *Funktion die Abschätzung*

$$\|u - v\|_{L^2(\omega)} \leq C_{apx}\frac{\operatorname{diam}(Q)}{\ell}\|\nabla u\|_{L^2(\omega)}.$$

Beweis. Wir zerlegen das Integral über ω in die Summe der Teilintegrale über ω_ν:

$$\|u - v\|^2_{L^2(\omega)} = \int_\omega |u(x) - v(x)|^2\,dx = \sum_{\nu\in N}\int_{\omega_\nu} |u(x) - v(x)|^2\,dx.$$

Nach Definition ist $v|_{\omega_\nu}$ gerade der Mittelwert der Funktion u auf diesem Teilgebiet, also folgt aus Lemma 7.8

$$\begin{aligned}\|u-v\|_{L^2(\omega)}^2 &= \sum_{\nu\in N}\int_{\omega_\nu}|u(x)-v(x)|^2\,dx \leq \sum_{\nu\in N}C_{\mathrm{apx}}^2\operatorname{diam}(\omega_\nu)^2\int_{\omega_\nu}\|\nabla u(x)\|_2^2\,dx\\ &\leq C_{\mathrm{apx}}^2\frac{\operatorname{diam}(Q)^2}{\ell^2}\sum_{\nu\in N}\int_{\omega_\nu}\|\nabla u(x)\|_2^2\,dx\\ &= C_{\mathrm{apx}}^2\frac{\operatorname{diam}(Q)^2}{\ell^2}\int_{\omega}\|\nabla u(x)\|_2^2\,dx = C_{\mathrm{apx}}^2\frac{\operatorname{diam}(Q)^2}{\ell^2}\|\nabla u\|_{L^2(\omega)}^2.\end{aligned}$$

Indem wir die Wurzel aus dieser Abschätzung ziehen erhalten wir das gewünschte Resultat. ∎

Aus diesem Lemma ergibt sich bereits eine erste Approximationsaussage: Da v aus dem Raum der stückweise konstanten Funktionen auf der Partition $(\omega_\nu)_{\nu\in N}$ stammt und diese Raum höchstens die Dimension $\#N \leq \ell^d$ besitzen kann, können wir u mit einer zu $1/\ell$ proportionalen Genauigkeit in einem höchstens ℓ^d-dimensionalen Raum annähern.

Leider ist diese Aussage in der Praxis wenig nützlich, weil die Dimension des Unterraums, und damit auch der lokale Rang der resultierenden hierarchischen Matrix, in der Regel sehr hoch sein muss, um eine gewisse Genauigkeit zu erreichen, da der Fehler nicht exponentiell konvergiert.

Allerdings haben wir bisher auch nur ausgenutzt, dass $u \in H^1(\omega)$ gilt, während seine Beziehung zu der Variationsgleichung außer Acht gelassen wurde. Das holen wir nun nach:

Definition 7.10 (L-harmonische Funktionen) *Sei $\omega \subseteq \mathbb{R}^d$ ein offenes Gebiet. Eine Funktion $u \in L^2(\omega)$ nennen wir* (lokal) L-harmonisch *auf ω, falls für alle $\tilde{\omega} \subseteq \omega$ mit $\operatorname{dist}(\tilde{\omega}, \partial\omega) > 0$ die folgenden Bedingungen gelten:*

$$u|_{\tilde{\omega}} \in H^1(\tilde{\omega}), \tag{7.11a}$$

$$a(v, u|_\Omega) = 0 \qquad \text{für alle } v \in H_0^1(\Omega) \text{ mit } \operatorname{supp} v \subseteq \tilde{\omega}, \tag{7.11b}$$

$$u|_{\omega\setminus\Omega} = 0. \tag{7.11c}$$

Der Raum aller (lokal) L-harmonischen Funktionen auf ω wird mit $\mathcal{H}(\omega)$ bezeichnet.

Die Bedingung (7.11b) ist formal nicht ganz sauber, weil $u|_\Omega$ nicht unbedingt auf dem ganzen Gebiet Ω definiert ist. Da aber $\operatorname{supp} v \subseteq \tilde{\omega}$ gilt, ist für die Auswertung der Bilinearform $a(\cdot,\cdot)$ nur das Integral über $\tilde{\omega}$ relevant, und auf diesem Gebiet ist $u|_\Omega$ definiert und nach (7.11a) auch schwach differenzierbar.

Für L-harmonische Funktionen ist es möglich, die H^1-Norm durch die L^2-Norm auf einem größeren Gebiet abzuschätzen:

Lemma 7.11 (Cacciopoli-Ungleichung) *Sei $u \in \mathcal{H}(\omega)$, und sei $\tilde{\omega} \subseteq \omega$ ein Gebiet mit $\operatorname{dist}(\tilde{\omega}, \partial\omega) > 0$. Dann gelten $u|_{\tilde{\omega}} \in H^1(\tilde{\omega})$ und die Abschätzung*

$$\|\nabla u\|_{L^2(\tilde{\omega})} \leq \frac{C_{reg}}{\operatorname{dist}(\tilde{\omega}, \partial\omega)}\|u\|_{L^2(\omega)}, \qquad C_{reg} := 4\sqrt{\beta/\alpha} \geq 4.$$

Hier sind $\alpha \le \beta$ die Konstanten aus (7.4).

Beweis. Sei $\delta := \operatorname{dist}(\tilde{\omega}, \partial\omega) > 0$ der Abstand von $\tilde{\omega}$ zu dem Rand des Gebiets ω. Wir können eine Funktion $\eta \in C^1(\Omega \cap \omega)$ finden, die

$$\begin{aligned} 0 \le \eta \le 1, \qquad \eta|_{\tilde{\omega}} = 1, \qquad \|\nabla\eta\|_2 \le 2/\delta \\ \eta(x) = 0 \quad \text{für alle } x \in \omega \text{ mit } \operatorname{dist}(x, \partial\omega) < \delta/4 \end{aligned}$$

erfüllt. Für das Gebiet

$$\hat{\omega} := \{x \in \omega \ : \ \operatorname{dist}(x, \partial\omega) > \delta/8\}$$

gilt nach Definition $\operatorname{dist}(\hat{\omega}, \partial\omega) \ge \delta/8 > 0$, also folgt aus (7.11a) bereits $u|_{\hat{\omega}} \in H^1(\hat{\omega})$. Da η stetig differenzierbar und beschränkt ist, muss dann auch $\hat{v} := \eta^2 u \in H_0^1(\omega)$ gelten, und wir können diese Funktion durch null zu einer Funktion $v \in H_0^1(\Omega \cup \omega)$ fortsetzen. Aus (7.11c) folgt $v|_{\omega\setminus\Omega} = 0$, also auch $v|_\Omega \in H_0^1(\Omega)$. Nach Voraussetzung gilt $\operatorname{dist}(\operatorname{supp}\eta, \partial\omega) \ge \delta/4 > \delta/8$, also muss auch $\operatorname{supp} v \subseteq \hat{\omega}$ gelten, so dass wir (7.11b) auf v anwenden können, um

$$\begin{aligned} 0 = a(v|_\Omega, u|_\Omega) &= \int_{\hat{\omega}\cap\Omega} \langle \nabla v(x), \mathbf{C}(x)\nabla u(x)\rangle_2 \, dx = \int_{\hat{\omega}\cap\Omega} \langle \nabla(\eta^2 u)(x), \mathbf{C}(x)\nabla u(x)\rangle_2 \, dx \\ &= \int_{\hat{\omega}\cap\Omega} \langle 2\eta(x)u(x)\nabla\eta(x) + \eta^2(x)\nabla u(x), \mathbf{C}(x)\nabla u(x)\rangle_2 \, dx \end{aligned}$$

mit der Produktregel zu erhalten, also

$$\int_{\hat{\omega}\cap\Omega} \eta^2(x)\langle \nabla u(x), \mathbf{C}\nabla u(x)\rangle_2 \, dx = -2\int_{\hat{\omega}\cap\Omega} \eta(x)u(x)\langle \nabla\eta(x), \mathbf{C}(x)\nabla u(x)\rangle_2 \, dx.$$

Aus dieser Gleichung folgt

$$\begin{aligned} \int_{\hat{\omega}\cap\Omega} \eta^2(x)\|\mathbf{C}(x)^{1/2}\nabla u(x)\|_2^2 \, dx &= \int_{\hat{\omega}\cap\Omega} \eta^2(x)\langle \nabla u(x), \mathbf{C}(x)\nabla u(x)\rangle_2^2 \, dx \\ &= -2\int_{\hat{\omega}\cap\Omega} \eta(x)u(x)\langle \nabla\eta(x), \mathbf{C}(x)\nabla u(x)\rangle_2 \, dx \\ &\le 2\int_{\hat{\omega}\cap\Omega} \eta(x)|u(x)| \ \|\mathbf{C}(x)^{1/2}\nabla\eta(x)\|_2 \|\mathbf{C}(x)^{1/2}\nabla u(x)\|_2 \, dx \\ &\le 2\beta^{1/2}\int_{\hat{\omega}\cap\Omega} \eta(x)|u(x)| \ \|\nabla\eta(x)\|_2 \|\mathbf{C}(x)^{1/2}\nabla u(x)\|_2 \, dx \\ &\le 4\frac{\beta^{1/2}}{\delta}\int_{\hat{\omega}\cap\Omega} \eta(x)|u(x)| \ \|\mathbf{C}(x)^{1/2}\nabla u(x)\|_2 \, dx \\ &\le 4\frac{\beta^{1/2}}{\delta}\|u\|_{L^2(\hat{\omega}\cap\Omega)} \left(\int_{\hat{\omega}\cap\Omega} \eta^2(x)\|\mathbf{C}(x)^{1/2}\nabla u(x)\|_2^2 \, dx\right)^{1/2}. \end{aligned}$$

Da auf der rechten Seite dieser Abschätzung die Wurzel der linken Seite auftritt, können wir

$$\left(\int_{\hat{\omega}\cap\Omega} \eta^2(x)\|\mathbf{C}(x)^{1/2}\nabla u(x)\|_2^2 \, dx\right)^{1/2} \le 4\frac{\sqrt{\beta}}{\delta}\|u\|_{L^2(\hat{\omega}\cap\Omega)}$$

folgern, abgesehen von den noch nicht ganz passenden Integrationsgebieten und dem Faktor $\mathbf{C}(x)^{1/2}$ liegt also bereits eine Abschätzung des Gradienten vor.

Indem wir (7.11c) und $\eta|_{\tilde{\omega}} = 1$ ausnutzen erhalten wir

$$\begin{aligned}
\|\nabla u\|_{L^2(\tilde{\omega})} = \|\nabla u\|_{L^2(\tilde{\omega}\cap\Omega)} &= \left(\int_{\tilde{\omega}\cap\Omega} \eta^2(x)\|\nabla u(x)\|_2^2\,dx\right)^{1/2} \\
&\leq \frac{1}{\sqrt{\alpha}}\left(\int_{\tilde{\omega}\cap\Omega} \eta^2(x)\|\mathbf{C}^{1/2}(x)\nabla u(x)\|_2^2\,dx\right)^{1/2} \\
&\leq \frac{1}{\sqrt{\alpha}}\left(\int_{\hat{\omega}\cap\Omega} \eta^2(x)\|\mathbf{C}^{1/2}(x)\nabla u(x)\|_2^2\,dx\right)^{1/2} \\
&\leq \frac{4}{\delta}\sqrt{\frac{\beta}{\alpha}}\|u\|_{L^2(\hat{\omega}\cap\Omega)} \leq \frac{4}{\delta}\sqrt{\frac{\beta}{\alpha}}\|u\|_{L^2(\omega)},
\end{aligned}$$

und das ist die gewünschte Abschätzung. ∎

Die Cacciopoli-Ungleichung bietet uns die Möglichkeit, die Approximationsaussage des Lemmas 7.9 wiederholt anzuwenden: Das letztere Lemma bietet uns eine Abschätzung für die L^2-Norm des Fehlers, und die Cacciopoli-Ungleichung ermöglicht es uns, daraus auf einem Teilgebiet eine Abschätzung der H^1-Norm des Fehlers zu gewinnen. Das ist allerdings nur möglich, sofern der Fehler L-harmonisch ist. Um diese Eigenschaft sicherstellen zu können, benötigen wir das folgende vorbereitende Lemma:

Lemma 7.12 (Abgeschlossener Teilraum) *Die Menge $\mathcal{H}(\omega)$ ist ein abgeschlossener Teilraum des Raums $L^2(\omega)$.*

Beweis. Sei $(u_n)_{n\in\mathbb{N}}$ eine Folge in $\mathcal{H}(\omega)$, die bezüglich der L^2-Norm gegen ein $u \in L^2(\omega)$ konvergiert. Wir wollen nachweisen, dass $u \in \mathcal{H}(\omega)$ gilt.

Sei dazu ein $\tilde{\omega} \subseteq \omega$ mit $\operatorname{dist}(\tilde{\omega}, \partial\omega) > 0$ fixiert. Aus Lemma 7.11 folgt, dass $(u_n|_{\tilde{\omega}})_{n\in\mathbb{N}}$ eine Cauchy-Folge bezüglich der H^1-Norm ist, also gegen einen Grenzwert $v \in H^1(\tilde{\omega})$ konvergieren muss. Da die Folge auch in der L^2-Norm gegen $u|_{\tilde{\omega}}$ konvergiert, müssen $u|_{\tilde{\omega}}$ und $v|_{\tilde{\omega}}$ bis auf Nullmengen identisch sein, also folgt $u|_{\tilde{\omega}} \in H^1(\tilde{\omega})$.

Damit ist (7.11a) für u nachgewiesen, die Bedingungen (7.11b) und (7.11c) folgen aus der H^1-Konvergenz per Stetigkeit. ∎

Da also $\mathcal{H}(\omega)$ ein bezüglich der L^2-Norm abgeschlossener Teilraum des Hilbertraums $L^2(\omega)$ ist, können wir eine orthogonale Projektion auf diesen Raum verwenden, um die folgende verfeinerte Variante des Lemmas 7.9 zu gewinnen:

Lemma 7.13 (Approximierende Teilräume) *Für jede konvexe Menge $\omega \subseteq \mathbb{R}^d$, die in einem achsenparallelen Quader Q enthalten ist, und für jedes $\ell \in \mathbb{N}$ existiert ein Raum $\mathcal{W} \subseteq \mathcal{H}(\omega)$ derart, dass*

$$\min_{w\in\mathcal{W}} \|u - w\|_{L^2(\omega)} \leq C_{apx}\frac{\operatorname{diam}(Q)}{\ell}\|\nabla u\|_{L^2(\omega)} \qquad \text{für alle } u \in \mathcal{H}(\omega)\cap H^1(\omega)$$

gilt, dass wir also L-harmonische Funktionen bei hinreichend großem ℓ beliebig gut approximieren können. Die Dimension dieses Raums ist durch ℓ^d beschränkt.

Beweis. Sei $\omega \subseteq \mathbb{R}^d$ eine konvexe Menge, sei Q ein sie enthaltender achsenparalleler Quader, sei $\ell \in \mathbb{N}$.

Sei Ψ der Riesz-Isomorphismus auf $\mathcal{H}(\omega)$, und sei

$$\mathfrak{E} : L^2(\omega) \to \mathcal{H}(\omega)', \qquad u \mapsto (v \mapsto \langle v, u \rangle_2),$$

die Einbettung von $L^2(\omega)$ in den Dualraum des Raums $\mathcal{H}(\omega)$. Dann ist $\Pi := \Psi^{-1}\mathfrak{E}$ eine lineare stetige Abbildung von $L^2(\omega)$ in den Raum $\mathcal{H}(\omega)$, die

$$\langle v, \Pi[u] \rangle_{L^2(\omega)} = \mathfrak{E}[u](v) = \langle v, u \rangle_{L^2(\omega)} \qquad \text{für alle } u \in L^2(\omega),\ v \in \mathcal{H}(\omega) \tag{7.12}$$

erfüllt. Für $u \in \mathcal{H}(\omega)$ folgt daraus mit der Testfunktion $v := u - \Pi[u] \in \mathcal{H}(\omega)$ die Gleichung

$$\|u - \Pi[u]\|_{L^2(\omega)}^2 = \langle v, u - \Pi[u] \rangle_{L^2(\omega)} = \langle v, u \rangle_{L^2(\omega)} - \langle v, \Pi[u] \rangle_{L^2(\omega)} = 0,$$

also ist Π eine Projektion auf $\mathcal{H}(\omega)$. Außerdem folgt aus (7.12) für $u \in L^2(\omega)$ und die Testfunktion $v := \Pi[u]$ auch

$$\|\Pi[u]\|_{L^2(\omega)}^2 = \langle v, \Pi[u] \rangle_{L^2(\omega)} = \langle v, u \rangle_{L^2(\omega)} = \langle \Pi[u], u \rangle_{L^2(\omega)} \leq \|\Pi[u]\|_{L^2(\omega)} \|u\|_{L^2(\omega)},$$

also insbesondere $\|\Pi[u]\|_{L^2(\omega)} \leq \|u\|_{L^2(\omega)}$.

Aus Lemma 7.9 wissen wir, dass es einen Raum $\mathcal{V} \subseteq L^2(\omega)$ der Dimension $\leq \ell^d$, nämlich den der stückweise konstanten Funktionen, gibt, der

$$\min_{v \in \mathcal{V}} \|u - v\|_{L^2(\omega)} \leq C_{\mathrm{apx}} \frac{\operatorname{diam}(Q)}{\ell} \|\nabla u\|_{L^2(\omega)} \qquad \text{für alle } u \in H^1(\omega)$$

erfüllt. Sei nun $u \in \mathcal{H}(\omega) \cap H^1(\omega)$. Wie bereits gesehen finden wir dann ein $v \in \mathcal{V}$ mit

$$\|u - v\|_{L^2(\omega)} \leq C_{\mathrm{apx}} \frac{\operatorname{diam}(Q)}{\ell} \|\nabla u\|_{L^2(\omega)}.$$

Aus $u \in \mathcal{H}(\omega)$ folgt $u = \Pi[u]$, und indem wir $w := \Pi[v]$ setzen, erhalten wir

$$\|u - w\|_{L^2(\omega)} = \|\Pi[u - v]\|_{L^2(\omega)} \leq \|u - v\|_{L^2(\omega)} \leq C_{\mathrm{apx}} \frac{\operatorname{diam}(Q)}{\ell} \|\nabla u\|_{L^2(\omega)}.$$

Damit ist klar, dass die Wahl

$$\mathcal{W} := \{\Pi[z] \ : \ z \in \mathcal{V}\}$$

zu einem Raum führt, dessen Dimension nicht größer als die des Raums $\mathcal{V}$ ist. Nach Definition der Projektion Π muss $\mathcal{W}$ auch ein Teilraum der L-harmonischen Funktionen $\mathcal{H}(\omega)$ sein. Offenbar gilt auch $w = \Pi[v] \in \mathcal{W}$, so dass der Beweis vollständig ist. ∎

Indem wir dieses Lemma mit der Cacciopoli-Ungleichung kombinieren, erhalten wir die folgende zentrale Aussage:

Lemma 7.14 (Approximationsschritt) *Sei $\omega \subseteq \mathbb{R}^d$ ein konvexes Gebiet und sei $\tilde{\omega} \subseteq \omega$ ein konvexes Teilgebiet mit $\operatorname{dist}(\tilde{\omega}, \partial\omega) \geq \delta > 0$. Sei $\widetilde{Q}$ ein achsenparalleler Quader, der $\tilde{\omega}$ enthält, und sei $\ell \in \mathbb{N}$. Dann existiert ein Teilraum $\mathcal{W} \subseteq \mathcal{H}(\tilde{\omega})$, dessen Dimension ℓ^d nicht überschreitet und für den*

$$\min_{w \in \mathcal{W}} \|u - w\|_{L^2(\tilde{\omega})} \leq C_{apx} C_{reg} \frac{\operatorname{diam}(\widetilde{Q})}{\delta \ell} \|u\|_{L^2(\omega)} \qquad \text{für alle } u \in \mathcal{H}(\omega)$$

gilt. Da $u \in \mathcal{H}(\tilde{\omega})$ gilt, folgt $u - w \in \mathcal{H}(\tilde{\omega})$.

Beweis. Indem wir Lemma 7.13 auf das Teilgebiet $\tilde{\omega}$ anwenden, erhalten wir einen Teilraum $\mathcal{W} \subseteq L^2(\tilde{\omega})$.

Sei $u \in \mathcal{H}(\omega)$. Aus Lemma 7.11 erhalten wir die Aussage, dass $u|_{\tilde{\omega}} \in H^1(\tilde{\omega})$ mit

$$\|\nabla u\|_{L^2(\tilde{\omega})} \leq \frac{C_{\mathrm{reg}}}{\delta} \|u\|_{L^2(\omega)}$$

gilt. Also können wir nun Lemma 7.13 anwenden, um

$$\min_{w \in \mathcal{W}} \|u - w\|_{L^2(\tilde{\omega})} \leq C_{\mathrm{apx}} \frac{\operatorname{diam}(\widetilde{Q})}{\ell} \|\nabla u\|_{L^2(\tilde{\omega})} \leq C_{\mathrm{apx}} C_{\mathrm{reg}} \frac{\operatorname{diam}(\widetilde{Q})}{\delta \ell} \|u\|_{L^2(\omega)}$$

zu folgern. Das ist die gesuchte Aussage. ∎

Unser Plan besteht nun darin, eine Folge

$$\omega_0 \supseteq \omega_1 \supseteq \ldots \supseteq \omega_p$$

von $p \in \mathbb{N}$ konvexen Teilmengen zu konstruieren, die

$$\operatorname{dist}(\omega_i, \partial\omega_{i-1}) \geq \delta > 0 \qquad \text{für alle } i \in \{1, \ldots, p\}$$

mit einem $\delta > 0$ und $L^{-1} f|_{\omega_0} \in \mathcal{H}(\omega_0)$ sowie $\tau \subseteq \omega_p$ (vgl. (7.10) für die Definition von τ) erfüllen.

Sei Q ein achsenparalleler Quader, der ω_1 und damit auch $\omega_2, \ldots, \omega_p$ enthält. Sei $u_0 := L^{-1} f|_{\omega_0} \in \mathcal{H}(\omega_0)$. Dank Lemma 7.14 finden wir einen Raum $\mathcal{W}_1 \subseteq L^2(\omega_1)$ und ein $w_1 \in \mathcal{W}_1$ mit

$$u_1 := u_0|_{\omega_1} - w_1 \in \mathcal{H}(\omega_1), \qquad \|u_1\|_{L^2(\omega_1)} \leq C_{\mathrm{apx}} C_{\mathrm{reg}} \frac{\operatorname{diam}(Q)}{\delta \ell} \|u_0\|_{L^2(\omega_0)}.$$

Induktiv konstruieren wir nun für jedes $i \in \{1, \ldots, p\}$ einen Raum $\mathcal{W}_i$ und ein $w_i \in \mathcal{W}_i$ mit

$$u_i := u_{i-1}|_{\omega_i} - w_i, \qquad \|u_i\|_{L^2(\omega_i)} \leq C_{\mathrm{apx}} C_{\mathrm{reg}} \frac{\operatorname{diam}(Q)}{\delta \ell} \|u_{i-1}\|_{L^2(\omega_{i-1})}$$

und erhalten schließlich

$$\|u_p\|_{L^2(\omega_p)} \leq \left(C_{\mathrm{apx}} C_{\mathrm{reg}} \frac{\operatorname{diam}(Q)}{\delta \ell} \right)^p \|u_0\|_{L^2(\omega_0)}. \tag{7.13}$$

Indem wir ℓ und p hinreichend groß wählen, können wir

$$u_p = u_{p-1}|_{\omega_p} - w_p = u_{p-2}|_{\omega_p} - w_{p-1}|_{\omega_p} - w_p = \ldots = u_0|_{\omega_p} - w_1|_{\omega_p} - w_2|_{\omega_p} - \ldots - w_p$$

beliebig weit reduzieren, also jede gewünschte Genauigkeit erreichen. Dabei gilt

$$w := w_1|_{\omega_p} + \ldots + w_p|_{\omega_p} \in \mathcal{W} := \mathcal{W}_1|_{\omega_p} + \ldots + \mathcal{W}_p|_{\omega_p},$$

wir haben also eine Funktion w aus einem Raum mit Dimension $\leq p\ell^d$ gefunden, die $u_0|_{\omega_p}$ beliebig gut approximiert.

Für die Konstruktion der Gebiete $\omega_0, \ldots, \omega_p$ greifen wir auf die uns wohlbekannten überdeckenden Quader zurück: Sei

$$Q_\tau = [a_{\tau,1}, b_{\tau,1}] \times \ldots \times [a_{\tau,d}, b_{\tau,d}] \supseteq \tau$$

ein überdeckender Quader des Gebiets τ, und gelte die Zulässigkeitsbedingung

$$\operatorname{diam}(Q_\tau) \leq 2\eta \operatorname{dist}(Q_\tau, \sigma).$$

Den Abstand zwischen Q_τ und σ zerlegen wir in gleich große Abschnitte, indem wir

$$\delta := \frac{\operatorname{dist}(Q_\tau, \sigma)}{\sqrt{d}p}$$

setzen. Nun definieren wir die Teilgebiete durch

$$\omega_i := [a_{\tau,1} - (p-i)\delta, b_{\tau,1} + (p-i)\delta] \times \ldots \times [a_{\tau,d} - (p-i)\delta, b_{\tau,d} + (p-i)\delta]$$
$$\text{für alle } i \in \{0, \ldots, p\}.$$

Damit gelten offenbar $\omega_p = Q_\tau$ und

$$\operatorname{dist}(\omega_i, \partial\omega_{i-1}) = \delta, \qquad \text{für alle } i \in \{1, \ldots, p\}.$$

Für jedes $x \in \omega_0$ haben wir

$$\begin{aligned}\operatorname{dist}(x, Q_\tau)^2 = \operatorname{dist}(x, \omega_p)^2 = \sum_{\iota=1}^{d} \operatorname{dist}(x_\iota, [a_{\tau,\iota}, b_{\tau,\iota}])^2 \leq \sum_{\iota=1}^{d} (p\delta)^2 = d(p\delta)^2 \\ = d\frac{p^2 \operatorname{dist}(Q_\tau, \sigma)^2}{dp^2} = \operatorname{dist}(Q_\tau, \sigma)^2,\end{aligned}$$

also gilt insbesondere $\omega_0 \cap \sigma = \emptyset$. Für den $\omega_0, \ldots, \omega_p$ überdeckenden Quader verwenden wir der Einfachheit halber $Q = \omega_0$ und finden per Dreiecksungleichung

$$\operatorname{diam}(Q) \leq \operatorname{diam}(Q_\tau) + 2\operatorname{dist}(Q_\tau, \sigma) \leq 2(\eta + 1)\operatorname{dist}(Q_\tau, \sigma).$$

Damit nimmt (7.13) die Form

$$\|u_p\|_{L^2(\omega_p)} \leq \left(2C_{\mathrm{apx}}C_{\mathrm{reg}}(\eta+1)\frac{\operatorname{dist}(Q_\tau, \sigma)\sqrt{d}p}{\ell \operatorname{dist}(Q_\tau, \sigma)}\right)^p \|u_0\|_{L^2(\omega_0)}$$

an, die wir durch Wahl von

$$c := 2C_{\mathrm{apx}}C_{\mathrm{reg}}(\eta+1)\sqrt{d}$$

auf die einfache Formel

$$\|u_p\|_{L^2(\omega_p)} \leq \left(c\frac{p}{\ell}\right)^p \|u_0\|_{L^2(\omega_0)}$$

reduzieren können. Wir können *jede beliebige* gewünschte Konvergenzrate $q \in (0,1)$ erreichen, indem wir

$$\ell := \left\lceil \frac{cp}{q} \right\rceil$$

setzen, denn dann gilt

$$c\frac{p}{\ell} \leq c\frac{pq}{cp} = q,$$

also auch

$$\|u_p\|_{L^2(\omega_p)} \leq q^p \|u_0\|_{L^2(\omega_0)}, \tag{7.14}$$

wir erreichen also *exponentielle Konvergenz* in p. Die Dimension des Raums $\mathcal{W}$ ist in diesem Fall durch

$$p\ell^d \leq p\left(\frac{cp}{q}+1\right)^d \leq p\left(\frac{c}{q}p+p\right)^d = p^{d+1}\left(\frac{c}{q}+1\right)^d = C_{\mathrm{dim}}p^{d+1} \tag{7.15}$$

mit der Konstanten

$$C_{\mathrm{dim}} := \left(\frac{c}{q}+1\right)^d$$

beschränkt, die Dimension des Teilraums, und damit der Rang der resultierenden hierarchischen Matrix, wächst also polynomiell in p, während der Fehler exponentiell fällt.

Damit ist das folgende zentrale Resultat bewiesen:

Satz 7.15 (Approximation) *Sei $q \in (0,1)$. Dann existiert eine Konstante $C_{dim} \in \mathbb{R}_{\geq 0}$ so, dass für jedes $\tau \subseteq Q_\tau$, jedes $\sigma \subseteq \Omega$ mit*

$$\operatorname{diam}(Q_\tau) \leq 2\eta \operatorname{dist}(Q_\tau, \sigma),$$

jedes $\ell \in \mathbb{N}$ ein Raum $\mathcal{W} \subseteq L^2(\tau)$ existiert, dessen Dimension $C_{dim}\ell^{d+1}$ nicht überschreitet und für den

$$\begin{aligned}\min_{w\in\mathcal{W}} \|(L^{-1}f)|_\tau - w\|_{L^2(\tau)} &\leq q^\ell \|L^{-1}f\|_{L^2(\tau)} \\ &\leq \frac{q^\ell}{\alpha C_\Omega}\|f\|_{V'} \qquad \text{für alle } f \in V' \text{ mit } \operatorname{supp} f \subseteq \sigma\end{aligned}$$

gilt. Für derartige $f \in V'$ können wir also $(L^{-1}f)|_\tau$ sehr gut durch niedrigen Rang approximieren.

Beweis. Siehe (7.14) und (7.15) in Kombination mit $u_0|_\tau = (L^{-1}f)|_\tau$ und der Stetigkeitsabschätzung (7.6). ∎

7.4 Zusammenfassung

Das Ziel dieses Kapitels besteht darin, zu demonstrieren, dass sich nicht nur Integral-, sondern auch Differentialgleichungen mit Hilfe hierarchischer Matrizen behandeln lassen. Genauer gesagt lässt sich für Differentialoperatoren der Bauart

$$\mathcal{L}[u] := -\sum_{i,j=1}^{d} \partial_i \left(C_{ij} \partial_j u \right),$$

die *stark elliptisch* sind, also die Bedingungen

$$\mathbf{C}(x) = \mathbf{C}^*(x), \qquad \alpha \mathbf{I} \leq \mathbf{C}(x) \leq \beta \mathbf{I}, \qquad \text{für alle } x \in \Omega$$

erfüllen, nachweisen, dass sogar ihre Inverse gut durch eine hierarchische Matrix approximiert werden kann.

Dazu überführt man die Gleichung

$$\mathcal{L}[u] = F$$

durch Multiplikation mit einer Testfunktion v und Integration über Ω in die *Variationsformulierung*

$$a(v,u) = f(v) \qquad \text{für alle } v \in V,$$

wobei Bilinearform und Funktional durch

$$a(v,u) := \int_\Omega \langle \nabla v(x), \mathbf{C}(x) \nabla u(x) \rangle_2 \, dx \qquad \text{für alle } u, v \in V,$$
$$f(v) := \int_\Omega v(x) F(x) \, dx \qquad \text{für alle } v \in V$$

gegeben sind. Als Vektorraum V verwendet man den *Sobolew*-Raum $H_0^1(\Omega)$ aller einmal schwach differenzierbaren Funktionen, die auf dem Rand des Gebiets (in einem geeigneten Sinn) gleich null sind.

Der Sobolew-Raum ist nützlich, weil sich nachweisen lässt, dass er ein Hilbert-Raum mit dem aus der Differentialgleichung entstandenen Skalarprodukt[1] $a(\cdot,\cdot)$ ist. Für Hilbert-Räume besagt der *Satz von Riesz* gerade, dass sich jedes Funktional durch das Skalarprodukt mit einem geeigneten Vektor darstellen lässt, und indem wir diesen Satz auf unseren Fall anwenden erhalten wir direkt ein $u \in V$, das $a(v,u) = f(v)$ für alle $v \in V$ erfüllt, also unsere Variationsgleichung löst.

Mit Hilfe des Operators

$$L : V \to V', \qquad u \mapsto a(\cdot, u)$$

[1] Die Bilinearform $a(\cdot,\cdot)$ wird häufig als *Energie-Skalarprodukt* bezeichnet und erfüllt nach Definition $a(v,u) = \langle v, \mathcal{L}[u] \rangle_{L^2(\Omega)}$ für hinreichend glatte u.

lässt sich die Variationsgleichung kurz als $Lu = f$ schreiben, und aus dem Satz von Riesz folgt, dass L^{-1} existiert und sogar stetig ist.

Unser Ziel besteht darin, nachzuweisen, dass L^{-1} sich mit Hilfe einer hierarchischen Matrix approximieren lässt. Wäre L^{-1} eine Matrix, könnten wir fordern, dass „$L^{-1}|_{\hat{t}\times\hat{s}}$“ für alle zulässigen Blöcke $b = (t, s)$ durch niedrigen Rang approximiert werden kann.

Da L^{-1} keine Matrix ist, müssen wir diese Forderung etwas umformulieren: Die Rollen der Cluster $\hat{t}$ und $\hat{s}$ übernehmen zulässige Teilgebiete $\tau, \sigma \subseteq \Omega$. Statt L^{-1} auf die Spalten aus $\hat{s}$ einzuschränken, schränken wir auf die Funktionale $f \in V'$ ein, die außerhalb von σ verschwinden. Statt L^{-1} auf die Zeilen aus $\hat{t}$ einzuschränken, schränken wir das Ergebnis des Operators auf das Teilgebiet τ ein. Statt zu fordern, dass der Block der Matrix niedrigen Rang hat, fordern wir, dass jeder Vektor aus seinem Bild durch einen Vektor aus einem niedrigdimensionalen Teilraum approximieren lässt. Zu zeigen ist jetzt also

$$\inf_{w\in\mathcal{W}} \|(L^{-1}f)|_\tau - w\|_{L^2(\tau)} \leq \epsilon\|f\|_{V'} \qquad \text{für alle } f \in V' \text{ mit } \operatorname{supp} f \subseteq \sigma,$$

wobei $\mathcal{W}$ ein möglichst niedrigdimensionaler Raum sein soll.

Den Raum $\mathcal{W}$ konstruieren wir mit Hilfe einer Folge von Teilgebieten

$$\omega_0 \supseteq \omega_1 \supseteq \ldots \supseteq \omega_p,$$

die der Einfachheit halber achsenparallele Quader sind und deren kleinstes τ enthält, also $\omega_p \supseteq \tau$ erfüllt. Aus Gründen, die gleich erklärt werden, darf der größte Quader σ nicht schneiden, muss also $\omega_0 \cap \sigma = \emptyset$ erfüllen.

Wir untersuchen nun die Funktion $u_0 := u|_{\omega_0}$. Da ω_0 den Träger σ der rechten Seite nicht schneidet, gilt (in einem geeigneten Sinn) $Lu_0 = 0$, und für derartige Funktionen besagt die *Cacciopoli-Ungleichung*, dass sich die H^1-Norm auf dem Teilgebiet ω_1 durch die L^2-Norm beschränken lässt.

Mit Hilfe dieser Schranke können wir die *Poincaré-Ungleichung* anwenden, um eine Approximation in einem höchstens ℓ^d-dimensionalen Raum zu konstruieren. Insgesamt erhalten wir so

$$\|u_0|_{\omega_1} - v_1\|_{L^2(\omega_1)} \leq \widetilde{C}_{\mathrm{apx}} \frac{\operatorname{dist}(\omega_p, \sigma)}{\operatorname{diam}(\omega_p)} \frac{p}{\ell} \|u_0\|_{L^2(\omega_0)}.$$

Durch einen kleinen Trick können wir dafür sorgen, dass der Fehler $u_1 := u_0|_{\omega_1} - v_1$ auch (wieder in einem geeigneten Sinn) $Lu_1 = 0$ erfüllt, so dass sich das Argument wiederholen lässt, um eine Approximation v_2 auf ω_2 zu erhalten.

Schließlich erhalten wir so $v_1, \ldots, v_p$ mit

$$\|u|_{\omega_p} - v_1|_{\omega_p} - v_2|_{\omega_p} - \ldots - v_p\|_{L^2(\omega_p)} \leq \left(\widetilde{C}_{\mathrm{apx}} \frac{\operatorname{dist}(\omega_p, \sigma)}{\operatorname{diam}(\omega_p)} \frac{p}{\ell}\right)^p \|u_0\|_{L^2(\omega_0)}.$$

Indem wir die Zulässigkeitsbedingung ausnutzen und ℓ hinreichend groß wählen erhalten wir ein ähnliches Konvergenzresultat wie im Fall der Integralgleichungen: Ein Fehler der Größenordnung q^p bei einem Rang proportional zu p^{d+1}, wobei $q \in (0, 1)$ eine beliebig wählbare Konvergenzrate ist.

Wir können $\|u_0\|_{L^2(\omega_0)}$ durch $\|u\|_V$ abschätzen, und da L^{-1} stetig ist, lässt sich diese Norm durch $\|f\|_{V'}$ beschränken und so das gewünschte Ergebnis erreichen.

8 $\mathcal{H}^2$-Matrizen

In vielen praktischen Anwendungen ist man daran interessiert, große Probleme zu lösen: Man möchte beispielsweise komplizierte Geometrien auflösen oder hohe Genauigkeiten erreichen.

Das bedeutet in der Regel, dass man nach der Diskretisierung mit Matrizen arbeiten muss, die sehr viele Zeilen und Spalten aufweisen, in der Regel mehrere Hunderttausend, je nach Anwendung auch Millionen oder Milliarden. Bei Matrizen dieser Größe wird der Speicherbedarf ein ernstes Problem: In der Regel ist der in einem Computer zur Verfügung stehende Speicher beschränkt, und Probleme, die mehr Speicher benötigen, lassen sich überhaupt nicht behandeln.

Also lohnt es sich, nach Techniken zu suchen, mit denen sich der Speicherbedarf für eine Matrix möglichst weit reduzieren lässt.

Eine Möglichkeit besteht darin, die Matrix nicht oder wenigstens nicht vollständig zu speichern, und dann die relevanten Operationen, wie beispielsweise die sehr wichtige Matrix-Vektor-Multiplikation, durchzuführen, indem nur die gerade benötigten Teile der Matrix aufgestellt, verwendet, und wieder gelöscht werden. In dieser Weise lässt sich der Speicherbedarf erheblich reduzieren, allerdings werden dadurch die Matrix-Operationen auch wesentlich langsamer oder lassen sich, wie beispielsweise die Matrix-LR-Zerlegung oder die Matrix-Inversion, nicht mehr durchführen.

Falls wir gleichzeitig hohe Geschwindigkeit und geringen Speicherbedarf anstreben, bietet es sich an, die Struktur der zu approximierenden Matrix etwas genauer zu untersuchen, um festzustellen, ob sie Eigenschaften aufweist, die sich algorithmisch ausnutzen lassen. $\mathcal{H}^2$-Matrizen [28, 10, 6] entstehen aus diesem Ansatz: Aus einer kleinen Modifikation der bisher verwendeten Niedrigrang-Darstellungen ergibt sich eine neue Darstellung, die nicht nur wesentlich weniger Speicher benötigt, sondern mit der es sich auch noch wesentlich effizienter rechnen lässt.

8.1 Struktur

Für die Herleitung der Struktur untersuchen wir wieder das Beispielproblem einer Matrix $\mathbf{G} \in \mathbb{K}^{\mathcal{I}\times\mathcal{J}}$, die durch eine Kernfunktion $g : \Omega \times \Omega \to \mathbb{K}$ und Punkte $(x_i)_{i\in\mathcal{I}}$, $(y_j)_{j\in\mathcal{J}}$ mittels

$$G_{ij} = g(x_i, y_j) \qquad \text{für alle } i \in \mathcal{I},\ j \in \mathcal{J}$$

definiert ist. Die Konstruktion einer hierarchischen Matrix beruht darauf, einen zulässigen Block $b = (t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ zu approximieren, indem g durch die Taylor-Entwicklung

$$g(x, y) \approx \tilde{g}(x, y) := \sum_{\nu=0}^{k-1} \frac{(x - x_t)^\nu}{\nu!} \frac{\partial^\nu g}{\partial x^\nu}(x_t, y)$$

ersetzt wird, so dass wir

$$G_{ij} \approx \widetilde{G}_{ij} := \tilde{g}(x_i, y_j) = \sum_{\nu=0}^{k-1} \frac{(x_i - x_t)^\nu}{\nu!} \frac{\partial^\nu g}{\partial x^\nu}(x_t, y_j) \qquad \text{für alle } i \in \hat{t},\ j \in \hat{s}$$

erhalten. Mit Hilfe der durch

$$(A_b)_{i\nu} := \frac{(x_i - x_t)^\nu}{\nu!}, \quad (B_b)_{j\nu} := \overline{\frac{\partial^\nu g}{\partial x^\nu}}(x_t, y_j) \quad \text{für } \nu \in \{0, \ldots, k-1\},\ i \in \hat{t},\ j \in \hat{s}$$

definierten Matrizen $\mathbf{A}_b \in \mathbb{K}^{\hat{t}\times k}$, $\mathbf{B}_b \in \mathbb{K}^{\hat{s}\times k}$ führt das zu der Darstellung

$$\widetilde{\mathbf{G}}|_{\hat{t}\times\hat{s}} = \mathbf{A}_b \mathbf{B}_b^*,$$

also zu einer faktorisierten Darstellung einer Rang-k-Matrix.

Unser Interesse gilt der Matrix $\mathbf{A}_b$. Ein Blick auf die Definition legt nahe, dass sie nicht von s abhängt, deshalb können wir sie auch als eine Matrix $\mathbf{V}_t \in \mathbb{K}^{\hat{t}\times k}$ interpretieren, die durch

$$(V_t)_{i\nu} := \frac{(x_i - x_t)^\nu}{\nu!} \qquad \text{für alle } i \in \hat{t},\ \nu \in \{0, \ldots, k-1\}$$

gegeben ist und für *alle* Blöcke $b = (t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ identisch ist. Insbesondere müssen wir sie auch nur einmal berechnen und sparen so einen erheblichen Teil des Rechenaufwands für die Konstruktion der hierarchischen Matrix.

Es gibt allerdings noch weitere Vorteile: Wenn wir die Matrix-Vektor-Multiplikation $\mathbf{y} \leftarrow \mathbf{y} + \widetilde{\mathbf{G}}\mathbf{x}$ durchführen, müssen wir unter anderem die Operation

$$\mathbf{y}|_{\hat{t}} \leftarrow \mathbf{y}|_{\hat{t}} + \sum_{\substack{s\in\operatorname{row}(t)\\(t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}} \widetilde{\mathbf{G}}|_{\hat{t}\times\hat{s}}\mathbf{x}|_{\hat{s}} = \mathbf{y}|_{\hat{t}} + \sum_{\substack{s\in\operatorname{row}(t)\\(t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}} \mathbf{A}_b\mathbf{B}_b^*\mathbf{x}|_{\hat{s}}$$

durchführen. Das erfordert offenbar gerade

$$\sum_{\substack{s\in\operatorname{row}(t)\\(t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}} 2k(\#\hat{s} + \#\hat{t}) \qquad \text{Operationen.}$$

Wenn wir nun $\mathbf{V}_t = \mathbf{A}_b$ ausnutzen und diese Matrix aus der Summe herausziehen, erhalten wir die äquivalente Operation

$$\mathbf{y}|_{\hat{t}} \leftarrow \mathbf{y}|_{\hat{t}} + \mathbf{V}_t \sum_{\substack{s\in\operatorname{row}(t)\\(t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}} \mathbf{B}_b^*\mathbf{x}|_{\hat{s}},$$

die lediglich

$$2(\#\hat{t})k + \sum_{\substack{s\in\operatorname{row}(t)\\(t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}} 2k(\#\hat{s}) \qquad \text{Operationen}$$

benötigt, sich also *wesentlich* schneller durchführen lässt, insbesondere falls viele Blöcke sich die Matrix $\mathbf{V}_t$ teilen.

Durch Ausnutzung der speziellen Struktur von $\mathbf{A}_b = \mathbf{V}_t$ gewinnen wir also sowohl während des Aufstellens der Matrix als auch während der Durchführung der Matrix-Vektor-Multiplikation an Effizienz.

Wir können sogar noch bessere Ergebnisse erzielen, wenn wir die Matrizen $\mathbf{V}_t$ nicht isoliert betrachten, sondern die Familie $(\mathbf{V}_t)_{t\in\mathcal{T}_\mathcal{I}}$ als Einheit untersuchen. Falls $t \in \mathcal{T}_\mathcal{I}$ und $t' \in \operatorname{sons}(t)$ gelten, erhalten wir die Gleichung

$$\begin{aligned}\frac{(x-x_t)^\nu}{\nu!} &= \frac{1}{\nu!}(x-x_{t'}+x_{t'}-x_t)^\nu = \frac{1}{\nu!}\sum_{\nu'=0}^{\nu}\binom{\nu}{\nu'}(x-x_{t'})^{\nu'}(x_{t'}-x_t)^{\nu-\nu'}\\ &= \frac{1}{\nu!}\sum_{\nu'=0}^{\nu}\frac{\nu!}{\nu'!(\nu-\nu')!}(x-x_{t'})^{\nu'}(x_{t'}-x_t)^{\nu-\nu'}\\ &= \sum_{\nu'=0}^{\nu}\frac{(x-x_{t'})^{\nu'}}{\nu'!}\frac{(x_{t'}-x_t)^{\nu-\nu'}}{(\nu-\nu')!}.\end{aligned}$$

Wie man sieht entspricht der erste Faktor jedes Summanden jeweils dem Taylor-Monom auf der linken Seite, allerdings mit dem Entwicklungszentrum $x_{t'}$ anstelle von x_t.

Indem wir die Koeffizienten in einer Matrix $\mathbf{E}_{t'} \in \mathbb{K}^{k\times k}$ mit

$$(E_{t'})_{\nu'\nu} = \begin{cases}\frac{(x_{t'}-x_t)^{\nu-\nu'}}{(\nu-\nu')!} & \text{falls } \nu' \le \nu,\\ 0 & \text{ansonsten}\end{cases} \qquad \text{für alle } \nu,\nu' \in \{0,\dots,k-1\}$$

sammeln, erhalten wir die Gleichung

$$(V_t)_{i\nu} = \sum_{\nu'=0}^{k-1}(V_{t'})_{i\nu'}(E_{t'})_{\nu'\nu} \qquad \text{für alle } i \in \hat{t}' \subseteq \hat{t},\ \nu \in \{0,\dots,k-1\},$$

beziehungsweise kürzer

$$\mathbf{V}_t|_{\hat{t}'\times k} = \mathbf{V}_{t'}\mathbf{E}_{t'}.$$

Da wir nach Definition $\hat{t}$ als Vereinigung der Mengen $\hat{t}'$ zu den Söhnen $t' \in \operatorname{sons}(t)$ darstellen können, ist $\mathbf{V}_t$ durch diese Gleichung bereits vollständig definiert. Insbesondere müssen wir die Matrix nicht explizit berechnen, sondern können sie implizit mit Hilfe der *Transfermatrizen* $\mathbf{E}_{t'}$ für $t' \in \operatorname{sons}(t)$ darstellen. Diese Matrizen besitzen nur k^2 Einträge, werden sich also in der Regel wesentlich sparsamer als die ursprüngliche Matrix $\mathbf{V}_t$ abspeichern lassen.

Bedauerlicherweise nutzen uns die zahlreichen erstrebenswerten Eigenschaften der Matrizen $\mathbf{V}_t$ nicht viel, solange wir für die Approximation der Matrixblöcke auch die Matrizen $\mathbf{B}_b$ benötigen. Diese Matrizen teilen die günstigen Eigenschaften der Matrizen $\mathbf{V}_t$ nicht, so dass wir sie nicht effizienter handhaben können.

Allerdings können wir die Approximation der Kernfunktion so modifizieren, dass sie zu besser handhabbaren Matrizen führt: Die Matrizen $\mathbf{V}_t$ entstehen aus den polynomiellen Faktoren der Taylor-Entwicklung, also bietet es sich an, eine Taylor-Entwicklung in beiden Variablen durchzuführen, um eine für unsere Zwecke geeignetere Darstellung zu erreichen:

$$\begin{aligned} g(x,y) &\approx \sum_{\nu=0}^{k-1} \frac{(x-x_t)^\nu}{\nu!} \frac{\partial^\nu g}{\partial x^\nu}(x_t,y) \\ &\approx \sum_{\nu=0}^{k-1} \sum_{\mu=0}^{k-1} \frac{(x-x_t)^\nu}{\nu!} \frac{\partial^{\nu+\mu} g}{\partial x^\nu \partial y^\mu}(x_t,y_s) \frac{(y-y_s)^\mu}{\mu!}. \end{aligned}$$

Die weder von x noch von y, sondern nur von den Entwicklungszentren x_t und y_s abhängenden Koeffizienten fassen wir in einer Matrix $\mathbf{S}_b \in \mathbb{K}^{k\times k}$ zusammen, die durch

$$(S_b)_{\nu\mu} := \frac{\partial^{\nu+\mu} g}{\partial x^\nu \partial y^\mu}(x_t,y_s) \qquad \text{für alle } \nu,\mu \in \{0,\ldots,k-1\}$$

definiert ist. Die Abhängigkeit von y ist nun ebenfalls durch Taylor-Monome ausgedrückt, die wir durch die Matrix $\mathbf{W}_s \in \mathbb{K}^{\hat{s}\times k}$ beschreiben, die durch

$$(W_s)_{j\mu} := \frac{(y_j-y_s)^\mu}{\mu!} \qquad \text{für alle } j \in \hat{s},\ \mu \in \{0,\ldots,k-1\}$$

gegeben ist. Insgesamt erhalten wir so die Approximation

$$\mathbf{G}|_{\hat{t}\times\hat{s}} \approx \widetilde{\mathbf{G}}|_{\hat{t}\times\hat{s}} = \mathbf{V}_t \mathbf{S}_b \mathbf{W}_s^*,$$

die nun aus den effizient handhabbaren Matrizen $\mathbf{V}_t$ und $\mathbf{W}_s$ und der kleinen Matrix $\mathbf{S}_b$ besteht, die die Kopplung zwischen den Clustern t und s beschreibt und sich aufgrund ihrer geringen Größe ebenfalls effizient verarbeiten lässt.

Die für die Speicherersparnis entscheidende Idee, die Matrizen $(\mathbf{V}_t)_{t\in\mathcal{T}_\mathcal{I}}$ und $(\mathbf{W}_s)_{s\in\mathcal{T}_\mathcal{J}}$ als Einheit zu behandeln, finden ihren Ausdruck in dem Begriff der *Clusterbasis*:

Definition 8.1 (Clusterbasis) *Sei $\mathcal{T}_\mathcal{I}$ ein Clusterbaum, und sei $\mathbf{V} = (\mathbf{V}_t)_{t\in\mathcal{T}_\mathcal{I}}$ eine Familie von Matrizen, die*

$$\mathbf{V}_t \in \mathbb{K}^{\hat{t}\times k} \qquad \textit{für alle } t \in \mathcal{T}_\mathcal{I}$$

erfüllt. Falls eine Familie $(\mathbf{E}_t)_{t\in\mathcal{T}_\mathcal{I}}$ existiert, die die Eigenschaft

$$\mathbf{V}_t|_{\hat{t}'\times k} = \mathbf{V}_{t'}\mathbf{E}_{t'} \qquad \textit{für alle } t \in \mathcal{T}_\mathcal{I},\ t' \in \operatorname{sons}(t)$$

besitzt, nennen wir $\mathbf{V}$ eine (geschachtelte) Clusterbasis *des Rangs k für den Clusterbaum $\mathcal{T}_\mathcal{I}$ und die Familie $(\mathbf{E}_t)_{t\in\mathcal{T}_\mathcal{I}}$ die korrespondierenden* Transfermatrizen.

Definition 8.2 (Geschachtelte Darstellung) *Sei $\mathbf{V} = (\mathbf{V}_t)_{t\in\mathcal{T}_\mathcal{I}}$ eine Clusterbasis. Nach Voraussetzung können wir $\mathbf{V}_t$ für $t \in \mathcal{T}_\mathcal{I}$ mit $\operatorname{sons}(t) \neq \emptyset$ mit Hilfe der Transfermatrizen rekonstruieren. Deshalb genügt es, die Matrizen $\mathbf{V}_t$ nur für Blattcluster $t \in \mathcal{L}_\mathcal{I}$ zu speichern.*

Das Paar $((\mathbf{V}_t)_{t\in\mathcal{L}_\mathcal{I}}, (\mathbf{E}_t)_{t\in\mathcal{T}_\mathcal{I}})$ bezeichnen wir als geschachtelte Darstellung *der Clusterbasis $\mathbf{V}$.*

Auf der Grundlage des Begriffs der Clusterbasis können wir nun auch die passende Matrixdarstellung definieren:

Definition 8.3 ($\mathcal{H}^2$-Matrix) *Sei $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ ein streng zulässiger Blockbaum, sei $k \in \mathbb{N}_0$, und seien $\mathbf{V}$ und $\mathbf{W}$ Clusterbasen des Rangs k für die Bäume $\mathcal{T}_\mathcal{I}$ und $\mathcal{T}_\mathcal{J}$. Eine Matrix $\mathbf{X} \in \mathbb{K}^{\mathcal{I}\times\mathcal{J}}$ bezeichnen wir als* $\mathcal{H}^2$-Matrix *mit Zeilenbasis $\mathbf{V}$ und Spaltenbasis $\mathbf{W}$, falls für alle $b = (t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ eine Matrix $\mathbf{S}_b \in \mathbb{K}^{k\times k}$ mit*

$$\mathbf{X}|_{\hat{t}\times\hat{s}} = \mathbf{V}_t\mathbf{S}_b\mathbf{W}_s^* \tag{8.1}$$

existiert. Die Familie $(\mathbf{S}_b)_{b\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}$ bezeichnen wir als Familie der Kopplungsmatrizen. *Die Menge aller $\mathcal{H}^2$-Matrizen bezeichnen wir mit*

$$\mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \mathbf{V}, \mathbf{W}) = \{\mathbf{X} \in \mathbb{K}^{\mathcal{I}\times\mathcal{J}} \ : \ \mathbf{X} \text{ ist } \mathcal{H}^2\text{-Matrix mit Zeilenbasis } \mathbf{V} \text{ und Spaltenbasis } \mathbf{W}\}.$$

Bemerkung 8.4 (Teilraum) *Seien $\mathbf{X}, \mathbf{Y} \in \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \mathbf{V}, \mathbf{W})$, sei $\alpha \in \mathbb{K}$. Dann gilt auch $\mathbf{X} + \alpha\mathbf{Y} \in \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \mathbf{V}, \mathbf{W})$, denn die Linearkombination der Kopplungsmatrizen ist wieder eine Kopplungsmatrix.*

Im Gegensatz zu $\mathcal{H}$-Matrizen bilden also die $\mathcal{H}^2$-Matrizen einen Teilraum des Vektorraums $\mathbb{K}^{\mathcal{I}\times\mathcal{J}}$ der Matrizen.

Definition 8.5 ($\mathcal{H}^2$-Matrix-Darstellung) *Sei $\mathbf{X} \in \mathbb{K}^{\mathcal{I}\times\mathcal{J}}$ eine $\mathcal{H}^2$-Matrix mit Spaltenbasis $\mathbf{V}$ und Zeilenbasis $\mathbf{W}$. Die zulässigen Blöcke sind durch die Kopplungsmatrizen $(\mathbf{S}_b)_{b\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}$ eindeutig beschrieben, die unzulässigen Blöcke durch die Familie $(\mathbf{N}_b)_{b\in\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}}$ mit*

$$\mathbf{N}_b := \mathbf{X}|_{\hat{t}\times\hat{s}} \qquad \text{für alle } b = (t, s) \in \mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}.$$

Falls $((\mathbf{V}_t)_{t\in\mathcal{L}_\mathcal{I}}, (\mathbf{E}_t)_{t\in\mathcal{T}_\mathcal{I}})$ und $((\mathbf{W}_s)_{s\in\mathcal{L}_\mathcal{J}}, (\mathbf{F}_s)_{s\in\mathcal{T}_\mathcal{J}})$ geschachtelte Darstellungen der Basen $\mathbf{V}$ und $\mathbf{W}$ sind, bezeichnen wir das Tupel

$$((\mathbf{S}_b)_{b\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}, (\mathbf{N}_b)_{b\in\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}}, (\mathbf{V}_t)_{t\in\mathcal{L}_\mathcal{I}}, (\mathbf{E}_t)_{t\in\mathcal{T}_\mathcal{I}}, (\mathbf{W}_s)_{s\in\mathcal{L}_\mathcal{J}}, (\mathbf{F}_s)_{s\in\mathcal{T}_\mathcal{J}})$$

als $\mathcal{H}^2$-Matrix-Darstellung der Matrix $\mathbf{X}$.

Unser Ziel ist es, die Effizienz der Darstellung einer Matrix zu verbessern, also sollten wir untersuchen, wie hoch der Speicherbedarf ausfällt.

Lemma 8.6 (Speicherbedarf der Clusterbasis) *Sei* $\mathbf{V}$ *eine Clusterbasis. Ihre geschachtelte Darstellung* $((\mathbf{V}_t)_{t\in\mathcal{L}_\mathcal{I}}, (\mathbf{E}_t)_{t\in\mathcal{T}_\mathcal{I}})$ *benötigt*

$$k(\#\mathcal{I}) + k^2(\#\mathcal{T}_\mathcal{I}) \quad \textit{Speicherplätze.}$$

Beweis. Aus Folgerung 2.21 erhalten wir, dass die Matrizen $(\mathbf{V}_t)_{t\in\mathcal{L}_\mathcal{I}}$ zusammen

$$\sum_{t\in\mathcal{L}_\mathcal{I}} k(\#\hat{t}) = k\sum_{t\in\mathcal{L}_\mathcal{I}} \#\hat{t} = k\#\bigcup_{t\in\mathcal{L}_\mathcal{I}} \hat{t} = k\#\mathcal{I}$$

Speicherplätze erfordern. Für die Transfermatrizen ergibt sich trivial

$$\sum_{t\in\mathcal{T}_\mathcal{I}} k^2 = k^2\#\mathcal{T}_\mathcal{I},$$

und der Beweis ist abgeschlossen. ∎

Indem wir dieses Lemma auf die Clusterbasen $\mathbf{V}$ und $\mathbf{W}$ anwenden, können wir „zwei Drittel“ der $\mathcal{H}^2$-Matrix-Darstellung bereits abschätzen. Es fehlen noch die Kopplungs- und die Nahfeldmatrizen:

Lemma 8.7 (Speicherbedarf der Blöcke) *Sei* $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ C_{sp}*-schwachbesetzt, und seien* $r_\mathcal{I}$ *und* $r_\mathcal{J}$ *die Auflösungen der Clusterbäume* $\mathcal{T}_\mathcal{I}$ *und* $\mathcal{T}_\mathcal{J}$*.*

Die Familie $(\mathbf{S}_b)_{b\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}$ *der Kopplungsmatrizen erfordert nicht mehr als*

$$k^2\#\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}} \leq C_{sp}k^2\min\{\#\mathcal{T}_\mathcal{I}, \#\mathcal{T}_\mathcal{J}\} \quad \textit{Speicherplätze.}$$

Die Familie $(\mathbf{N}_b)_{b\in\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}}$ *der Nahfeldmatrizen erfordert nicht mehr als*

$$C_{sp}\min\{r_\mathcal{I}\#\mathcal{J}, r_\mathcal{J}\#\mathcal{I}\} \quad \textit{Speicherplätze.}$$

Beweis. Für die Kopplungsmatrizen genügt es, die Abschätzung

$$\#\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}} = \sum_{t\in\mathcal{T}_\mathcal{I}} \#\{s\in\operatorname{row}(t) \ : \ (t,s)\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}\} \leq \sum_{t\in\mathcal{T}_\mathcal{I}} C_{\mathrm{sp}} = C_{\mathrm{sp}}\#\mathcal{T}_\mathcal{I}$$

aus der Definition 2.24 zu gewinnen.

Für die Nahfeldmatrizen können wir ausnutzen, dass $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ *streng* zulässig ist, dass also für jeden Block $b=(t,s)\in\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}$ sowohl t als auch s Blätter der Bäume $\mathcal{T}_\mathcal{I}$ und $\mathcal{T}_\mathcal{J}$ sein müssen. Aus Folgerung 2.21 ergibt sich so für den Speicherbedarf die Schranke

$$\begin{aligned}\sum_{b=(t,s)\in\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}} (\#\hat{t})(\#\hat{s}) &\leq \sum_{b=(t,s)\in\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}} r_\mathcal{J}(\#\hat{t}) = r_\mathcal{J}\sum_{t\in\mathcal{L}_\mathcal{I}}\sum_{\substack{s\in\operatorname{row}(t)\\(t,s)\in\mathcal{L}^-_{\mathcal{I}\times\mathcal{J}}}} \#\hat{t}\\ &\leq C_{\mathrm{sp}}r_\mathcal{J}\sum_{t\in\mathcal{L}_\mathcal{I}}\#\hat{t} = C_{\mathrm{sp}}r_\mathcal{J}\#\bigcup_{t\in\mathcal{L}_\mathcal{I}}\hat{t} = C_{\mathrm{sp}}r_\mathcal{J}\#\mathcal{I}.\end{aligned}$$

Wir können die Rollen der Cluster t und s in beiden Abschätzung vertauschen, um das gewünschte Minimum zu erreichen. ∎

Aus der Kombination beider Abschätzungen erhalten wir eine erfreuliche Aussage über den Speicherbedarf von $\mathcal{H}^2$-Matrizen:

Satz 8.8 (Speicherbedarf einer $\mathcal{H}^2$-Matrix) *Sei $\mathcal{T}_{\mathcal{I}\times\mathcal{J}}$ C_{sp}-schwachbesetzt, seien $r_{\mathcal{I}}$ und $r_{\mathcal{J}}$ die Auflösungen der Clusterbäume $\mathcal{T}_{\mathcal{I}}$ und $\mathcal{T}_{\mathcal{J}}$, und seien $\mathbf{V}$ und $\mathbf{W}$ Clusterbasen des Rangs k.*

Dann benötigt die $\mathcal{H}^2$-Matrix-Darstellung einer Matrix $\mathbf{X} \in \mathcal{H}^2(\mathcal{T}_{\mathcal{I}\times\mathcal{J}}, \mathbf{V}, \mathbf{W})$ nicht mehr als

$$\left(\frac{C_{sp}}{2} + 1\right) k^2(\#\mathcal{T}_{\mathcal{I}} + \#\mathcal{T}_{\mathcal{J}}) + (C_{sp} \min\{r_{\mathcal{I}}, r_{\mathcal{J}}\} + k)\,(\#\mathcal{I} + \#\mathcal{J}) \qquad \textit{Speicherplätze.}$$

Beweis. Aus der Abschätzung des Lemmas 8.7 gewinnen wir die handlichere Schranke

$$\frac{C_{\mathrm{sp}}}{2} k^2(\#\mathcal{T}_{\mathcal{I}} + \#\mathcal{T}_{\mathcal{J}}) + C_{\mathrm{sp}} \min\{r_{\mathcal{I}}, r_{\mathcal{J}}\}(\#\mathcal{I} + \#\mathcal{J})$$

für den Speicherbedarf der Kopplungs- und Nahfeldmatrizen. Mit Hilfe des Lemmas 8.6 folgt das gewünschte Ergebnis. ∎

Bemerkung 8.9 (Optimale Ordnung) *Sehr wichtig an der Abschätzung des Satzes 8.8 ist die Tatsache, dass die Tiefen der Clusterbäume und des Blockbaums nicht auftreten. Unter den vereinfachenden Annahmen*

$$\#\mathcal{T}_{\mathcal{I}} \sim \#\mathcal{I}/k, \qquad\qquad \#\mathcal{T}_{\mathcal{J}} \sim \#\mathcal{J}/k$$

erhalten wir, dass der Speicherbedarf der $\mathcal{H}^2$-Matrix proportional zu

$$\max\{k, r_{\mathcal{I}}, r_{\mathcal{J}}\}(\#\mathcal{I} + \#\mathcal{J})$$

ist, also linear mit der Anzahl der Unbekannten wächst. Da nach Lemma 8.7 alleine das Nahfeld schon proportional zu $\#\mathcal{I}$ beziehungsweise $\#\mathcal{J}$ wächst, ist der Speicherbedarf der $\mathcal{H}^2$-Matrix von optimaler Ordnung.

8.2 Konstruktion per Interpolation

Wie bereits in Kapitel 3 diskutiert empfiehlt es sich in der Regel nicht, eine Approximation mit Hilfe der Taylor-Entwicklung zu konstruieren. Wesentlich handlicher und in der Regel auch wesentlich genauer ist ein Ansatz auf Grundlage der Interpolation.

Dieser Ansatz lässt sich direkt auf $\mathcal{H}^2$-Matrizen übertragen [11, 12]: Wir wählen Interpolationspunkte $(\xi_{t,\nu})_{\nu\in M}$ für den Cluster t und $(\xi_{s,\mu})_{\mu\in M}$ für den Cluster s und interpolieren die Kernfunktion g in *beiden* Variablen, um

$$g(x,y) \approx \tilde{g}(x,y) := \sum_{\nu\in M} \sum_{\mu\in M} \mathcal{L}_{t,\nu}(x) g(\xi_{t,\nu}, \xi_{s,\mu}) \mathcal{L}_{s,\mu}(y)$$

zu erhalten. Für die Matrix ergibt sich so die Approximation

$$\mathbf{G}_{ij} \approx \widetilde{\mathbf{G}}_{ij} := \tilde{g}(x_i, y_j) = \sum_{\nu\in M} \sum_{\mu\in M} \mathcal{L}_{t,\nu}(x_i) g(\xi_{t,\nu}, \xi_{s,\mu}) \mathcal{L}_{s,\mu}(y_j) \quad \text{für alle } i \in \hat{t},\ j \in \hat{s},$$

die wir mit Hilfe der Matrizen $\mathbf{V}_t \in \mathbb{K}^{\hat{t}\times k}$, $\mathbf{W}_s \in \mathbb{K}^{\hat{s}\times k}$ und $\mathbf{S}_b \in \mathbb{K}^{k\times k}$, gegeben durch

$$\begin{aligned}(V_t)_{i\nu} &= \mathcal{L}_{t,\nu}(x_i) && \text{für alle } i \in \hat{t},\ \nu \in M,\\ (W_s)_{j\mu} &= \mathcal{L}_{s,\mu}(y_j) && \text{für alle } j \in \hat{s},\ \mu \in M,\\ (S_b)_{\nu\mu} &= g(\xi_{t,\nu}, \xi_{s,\mu}) && \text{für alle } \nu,\mu \in M,\end{aligned}$$

in der kompakten Form

$$\widetilde{\mathbf{G}}|_{\hat{t}\times\hat{s}} = \mathbf{V}_t \mathbf{S}_b \mathbf{W}_s^*$$

schreiben können, also in der Form, die wir für die Darstellung als $\mathcal{H}^2$-Matrix benötigen.

Wir müssen noch überprüfen, ob die so definierten Familien $\mathbf{V} = (\mathbf{V}_t)_{t\in\mathcal{T}_\mathcal{I}}$ und $\mathbf{W} = (\mathbf{W}_s)_{s\in\mathcal{T}_\mathcal{J}}$ Clusterbasen sind, ob wir also passende Transfermatrizen finden können. Da wir für sämtliche Cluster mit Polynomen derselben Ordnung interpolieren, folgt aus der Projektionseigenschaft der Interpolationsoperatoren

$$\mathcal{L}_{t,\nu} = \mathfrak{I}_{Q_{t'}}[\mathcal{L}_{t,\nu}] = \sum_{\nu'\in M} \mathcal{L}_{t,\nu}(\xi_{t',\nu'})\mathcal{L}_{t',\nu'} \qquad \text{für alle } t \in \mathcal{T}_\mathcal{I},\ t' \in \operatorname{sons}(t),\ \nu \in M,$$

und indem wir diese Gleichung in die Definition der Matrix $\mathbf{V}_t$ einsetzen, folgt

$$(V_t)_{i\nu} = \mathcal{L}_{t,\nu}(x_i) = \sum_{\nu'\in M} \mathcal{L}_{t,\nu}(\xi_{t',\nu'})\mathcal{L}_{t',\nu'}(x_i) \qquad \text{für alle } i \in \hat{t}',\ \nu \in M.$$

Mit Hilfe der durch

$$(E_{t'})_{\nu'\nu} := \mathcal{L}_{t,\nu}(\xi_{t',\nu'}) \qquad \text{für alle } \nu,\nu' \in M$$

definierten Transfermatrix $\mathbf{E}_{t'} \in \mathbb{K}^{k\times k}$ lässt sich diese Gleichung in der Form

$$\mathbf{V}_t|_{\hat{t}'\times k} = \mathbf{V}_{t'}\mathbf{E}_{t'}$$

zusammenfassen, also erfüllen die so definierten Transfermatrizen die erforderlichen Bedingungen.

Bei der Analyse des Interpolationsfehlers verwenden wir Satz 3.9, den wir allerdings auf den Interpolationsoperator $\mathfrak{I}_{Q_t\times Q_s}$ anstelle des Operators $\mathfrak{I}_{Q_t}$ anwenden. Wir setzen wieder voraus, dass die Kernfunktion g asymptotisch glatt (vgl. Definition 3.10) ist, und erhalten wie in Kapitel 3 die Abschätzung

$$\|g - \tilde{g}\|_{\infty,Q_t\times Q_s} \leq \frac{C(m)}{\operatorname{dist}(Q_t,Q_s)^\sigma}\left(\frac{c_0 \operatorname{diam}_\infty(Q_t\times Q_s)}{4\operatorname{dist}(Q_t,Q_s)}\right)^{m+1}$$

mit dem Polynom

$$C(m) := 4dC_{\mathrm{as}}(m+1)^{2d-1}\frac{(m+\sigma)!}{(m+1)!}.$$

Da nun der Durchmesser des $2d$-dimensionalen Quaders $Q_t \times Q_s$ im Zähler des für die Konvergenz entscheidenden Terms auftritt, müssen wir die stärkere Zulässigkeitsbedingung

$$\operatorname{diam}_\infty(Q_t\times Q_s) \leq \eta \operatorname{dist}(Q_t,Q_s) \tag{8.2}$$

verwenden, die mit der bereits in (4.3) für die hybride Kreuzapproximation eingesetzten übereinstimmt. Hier ist $\eta \in \mathbb{R}_{>0}$ wieder ein Parameter, den wir geeignet wählen müssen.

Für $t \in \mathcal{T}_\mathcal{I}$ und $s \in \mathcal{T}_\mathcal{J}$, die diese Zulässigkeitsbedingung erfüllen, erhalten wir die Fehlerabschätzung

$$\|g - \tilde{g}\|_{\infty,Q_t\times Q_s} \leq \frac{C(m)}{\operatorname{dist}(Q_t,Q_s)^\sigma}\left(\frac{c_0\eta}{4}\right)^{m+1},$$

dürfen also wieder folgern, dass für $\eta < 4/c_0$ exponentielle Konvergenz des Interpolanten zu erwarten ist.

8.3 Strukturelle Unterschiede zu $\mathcal{H}$-Matrizen

Eine hierarchische Matrix ist dadurch charakterisiert, dass jeder zulässige Block niedrigen Rang besitzt. Bei einer $\mathcal{H}^2$-Matrix reicht diese Eigenschaft nicht aus, die Blöcke müssen auch noch die spezielle Form (8.1) aufweisen. Direkt lässt sich das nur nachweisen, indem wir Clusterbasen und Kopplungsmatrizen konstruieren, die die Gleichung (8.1) erfüllen. Da dieser Weg häufig zu aufwendig ist, suchen wir nach einer indirekten Charakterisierung, die sich leichter nachprüfen lässt. Einen Ansatz bietet die in [8] entwickelte Theorie, die hier kurz zusammengefasst werden soll.

Lemma 8.10 (Einschränkungen) *Sei $(\mathbf{V}_t)_{t\in\mathcal{T}_\mathcal{I}}$ eine Clusterbasis mit Transfermatrizen $(\mathbf{E}_t)_{t\in\mathcal{T}_\mathcal{I}}$. Es gilt*

$$\mathbf{V}_t|_{\hat{r}\times k} = \mathbf{V}_r\mathbf{E}_{r,t} \qquad \text{für alle } t \in \mathcal{T}_\mathcal{I},\ r \in \operatorname{sons}^*(t) \tag{8.3}$$

mit den durch

$$\mathbf{E}_{r,t} := \begin{cases} \mathbf{E}_{r,t'}\mathbf{E}_{t'} & \text{falls } r \in \operatorname{sons}^*(t'),\ t' \in \operatorname{sons}(t), \\ \mathbf{I} & \text{ansonsten, also falls } t = r \end{cases} \qquad \text{für alle } t \in \mathcal{T}_\mathcal{I},\ r \in \operatorname{sons}^*(t)$$

induktiv definierten erweiterten Tranfermatrizen.

Beweis. Wir führen den Beweis per Induktion über $\#\operatorname{sons}^*(t) \in \mathbb{N}$.

Seien zunächst $t \in \mathcal{T}_\mathcal{I}$ mit $\#\operatorname{sons}^*(t) = 1$ und $r \in \operatorname{sons}^*(t)$ gegeben. Nach Definition folgt dann $\operatorname{sons}(t) = \emptyset$ und damit $r = t$, so dass die Gleichung (8.3) trivial erfüllt ist.

Sei nun $n \in \mathbb{N}$ so gewählt, dass (8.3) für alle $t \in \mathcal{T}_\mathcal{I}$ mit $\#\operatorname{sons}^*(t) \leq n$ und alle $r \in \operatorname{sons}^*(t)$ gilt. Sei $t \in \mathcal{T}_\mathcal{I}$ mit $\#\operatorname{sons}^*(t) = n+1$ fixiert, und sei $r \in \operatorname{sons}^*(t)$. Falls $t = r$ gelten sollte, ist (8.3) trivial. Anderenfalls existiert genau ein $t' \in \operatorname{sons}(t)$ mit $r \in \operatorname{sons}^*(t')$. Aus $\operatorname{sons}^*(t') \subseteq \operatorname{sons}^*(t)$ und $t \notin \operatorname{sons}^*(t')$ folgt $\#\operatorname{sons}^*(t') \leq n$, also können wir die Induktionsvoraussetzung anwenden, um

$$\mathbf{V}_{t'}|_{\hat{r}\times k} = \mathbf{V}_r\mathbf{E}_{r,t'}$$

zu folgern. Da die Clusterbasis geschachtelt ist und $\hat{r} \subseteq \hat{t}' \subseteq \hat{t}$ gilt, erhalten wir daraus

$$\mathbf{V}_t|_{\hat{r}\times k} = (\mathbf{V}_t|_{\hat{t}'\times k})|_{\hat{r}\times k} = (\mathbf{V}_{t'}\mathbf{E}_{t'})|_{\hat{r}\times k} = \mathbf{V}_{t'}|_{\hat{r}\times k}\mathbf{E}_{t'} = \mathbf{V}_r\mathbf{E}_{r,t'}\mathbf{E}_{t'} = \mathbf{V}_r\mathbf{E}_{r,t},$$

und das ist die zu zeigende Gleichung. ∎

Mit Hilfe dieses Lemmas können wir nun die Struktur von $\mathcal{H}^2$-Matrizen genauer untersuchen. Sei dazu $\mathbf{X} \in \mathbb{K}^{\mathcal{I}\times\mathcal{J}}$ eine $\mathcal{H}^2$-Matrix mit Zeilenbasis $(\mathbf{V}_t)_{t\in\mathcal{T}_\mathcal{I}}$, Spaltenbasis $(\mathbf{W}_s)_{s\in\mathcal{T}_\mathcal{J}}$ und Kopplungsmatrizen $(\mathbf{S}_b)_{b\in\mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}}$.

Sei $t \in \mathcal{T}_\mathcal{I}$. Für jeden Block $b = (t^*, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ mit $t \in \operatorname{sons}^*(t^*)$ folgt aus Lemma 8.10 die Gleichung

$$\mathbf{X}|_{\hat{t}\times\hat{s}} = (\mathbf{X}|_{\hat{t}^*\times\hat{s}})|_{\hat{t}\times\hat{s}} = (\mathbf{V}_{t^*}\mathbf{S}_b\mathbf{W}_s^*)|_{\hat{t}\times\hat{s}} = \mathbf{V}_{t^*}|_{\hat{t}\times k}\mathbf{S}_b\mathbf{W}_s^* = \mathbf{V}_t\mathbf{E}_{t,t^*}\mathbf{S}_b\mathbf{W}_s^*, \tag{8.4}$$

die Einschränkung der Matrix $\mathbf{X}$ auf den Block $\hat{t}\times\hat{s}$ lässt sich also mit Hilfe der Matrix $\mathbf{V}_t$ ausdrücken.

Zur Abkürzung führen wir die Mengen

$$\operatorname{row}^*(t) := \{s \in \mathcal{T}_\mathcal{J} \ : \ \text{es existiert } t^* \in \mathcal{T}_\mathcal{I} \text{ mit } (t^*, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}},\ t \in \operatorname{sons}^*(t^*)\},$$
$$\operatorname{col}^*(s) := \{t \in \mathcal{T}_\mathcal{I} \ : \ \text{es existiert } s^* \in \mathcal{T}_\mathcal{J} \text{ mit } (t, s^*) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}},\ s \in \operatorname{sons}^*(s^*)\}$$

für alle $t \in \mathcal{T}_\mathcal{I}$ und $s \in \mathcal{T}_\mathcal{J}$ ein und formulieren unsere Beobachtung allgemeiner und präziser wie folgt:

Lemma 8.11 (Teilmatrizen) *Für jedes $t \in \mathcal{T}_\mathcal{I}$ und jedes $s \in \operatorname{row}^*(t)$ existiert eine Matrix $\mathbf{B}_{t,s} \in \mathbb{K}^{\hat{s}\times k}$ mit*

$$\mathbf{X}|_{\hat{t}\times\hat{s}} = \mathbf{V}_t\mathbf{B}_{t,s}^*.$$

Entsprechend exisitiert für jedes $s \in \mathcal{T}_\mathcal{J}$ und jedes $t \in \operatorname{col}^*(s)$ eine Matrix $\mathbf{A}_{t,s} \in \mathbb{K}^{\hat{t}\times k}$ mit

$$\mathbf{X}|_{\hat{t}\times\hat{s}} = \mathbf{A}_{t,s}\mathbf{W}_s^*.$$

Beweis. Seien $t \in \mathcal{T}_\mathcal{I}$ und $s \in \operatorname{row}^*(t)$ gegeben. Nach Definition existiert ein $t^* \in \mathcal{T}_\mathcal{I}$ mit $(t^*, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ und $t \in \operatorname{sons}^*(t^*)$, und gemäß (8.4) folgt

$$\mathbf{X}|_{\hat{t}\times\hat{s}} = \mathbf{V}_t\mathbf{E}_{t,t^*}\mathbf{S}_b\mathbf{W}_s^* = \mathbf{V}_t\mathbf{B}_{t,s}^*$$

mit der Matrix

$$\mathbf{B}_{t,s} := \mathbf{W}_s\mathbf{S}_b^*\mathbf{E}_{t,t^*}^* \in \mathbb{K}^{\hat{s}\times k}.$$

Seien nun $s \in \mathcal{T}_\mathcal{J}$ und $t \in \operatorname{col}^*(s)$ gegeben. Wir bezeichnen mit $(\mathbf{F}_s)_{s\in\mathcal{T}_\mathcal{J}}$ die Transfermatrizen der Spaltenbasis und entsprechend mit $\mathbf{F}_{r,s}$ die erweiterten Transfermatrizen für $s \in \mathcal{T}_\mathcal{J}$ und $r \in \operatorname{sons}^*(r)$. Nach Definition finden wir ein $s^* \in \mathcal{T}_\mathcal{J}$ mit $b := (t, s^*) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ und $s \in \operatorname{sons}^*(s^*)$, und aus Lemma 8.10 folgt

$$\begin{aligned}\mathbf{X}|_{\hat{t}\times\hat{s}} &= (\mathbf{X}|_{\hat{t}\times\hat{s}^*})|_{\hat{t}\times\hat{s}} = (\mathbf{V}_t\mathbf{S}_b\mathbf{W}_{s^*}^*)|_{\hat{t}\times\hat{s}} = \mathbf{V}_t\mathbf{S}_b(\mathbf{W}_{s^*}^*)|_{k\times\hat{s}} = \mathbf{V}_t\mathbf{S}_b(\mathbf{W}_{s^*}|_{\hat{s}\times k})^* \\ &= \mathbf{V}_t\mathbf{S}_b(\mathbf{W}_s\mathbf{F}_{s,s^*})^* = \mathbf{V}_t\mathbf{S}_b\mathbf{F}_{s,s^*}^*\mathbf{W}_s^* = \mathbf{A}_{t,s}\mathbf{W}_s^*\end{aligned}$$

mit der Matrix

$$\mathbf{A}_{t,s} := \mathbf{V}_t\mathbf{S}_b\mathbf{F}_{s,s^*}^* \in \mathbb{K}^{\hat{t}\times k}.$$

Das ist das gewünschte Ergebnis. ∎

Wir können dieses Resultat noch weiter vereinfachen, indem wir die Eigenschaften der Mengen $\operatorname{row}^*(t)$ und $\operatorname{col}^*(s)$ ausnutzen:

Lemma 8.12 (Erweiterte Blockzeilen und -spalten) *Sei $t \in \mathcal{T}_\mathcal{I}$. Für jedes $s \in \operatorname{row}^*(t)$ existiert genau ein $t^* \in \mathcal{T}_\mathcal{I}$ mit $(t^*, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ und $t \in \operatorname{sons}^*(t^*)$. Alle Cluster $s_1, s_2 \in \operatorname{row}^*(t)$ mit $s_1 \neq s_2$ sind disjunkt.*

Sei $s \in \mathcal{T}_\mathcal{J}$. Für jedes $t \in \operatorname{col}^*(t)$ existiert genau ein $s^* \in \mathcal{T}_\mathcal{J}$ mit $(t, s^*) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ und $s \in \operatorname{sons}^*(s^*)$. Alle Cluster $t_1, t_2 \in \operatorname{col}^*(s)$ mit $t_1 \neq s_2$ sind disjunkt.

Beweis. Wir führen den Beweis per Kontraposition: Sei $t \in \mathcal{T}_\mathcal{I}$. Seien $s_1, s_2 \in \operatorname{row}^*(t)$ mit $\hat{s}_1 \cap \hat{s}_2 \neq \emptyset$ gegeben. Dann existieren $t_1^*, t_2^* \in \mathcal{T}_\mathcal{I}$ mit $b_1 := (t_1^*, s_1) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$, $t \in \operatorname{sons}^*(t_1^*)$, $b_2 := (t_2^*, s_2) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ sowie $t \in \operatorname{sons}^*(t_2^*)$. Es folgt

$$\hat{b}_1 \cap \hat{b}_2 \supseteq (\hat{t}_1^* \times \hat{s}_1) \cap (\hat{t}_2^* \times \hat{s}_2) = (\hat{t}_1^* \cap \hat{t}_2^*) \times (\hat{s}_1 \cap \hat{s}_2) \supseteq \hat{t} \times (\hat{s}_1 \cap \hat{s}_2) \neq \emptyset.$$

Da nach Folgerung 2.21 die Blätter des Blockbaums disjunkt sind, folgt $b_1 = b_2$ und damit auch $t_1^* = t_2^*$ sowie $s_1 = s_2$.

Entsprechend können wir auch mit $s \in \mathcal{T}_\mathcal{J}$ und $\operatorname{col}^*(s)$ verfahren. ∎

Aus dieser Eindeutigkeitsaussage ergibt sich, dass wir die in Lemma 8.11 definierten Teilmatrizen zu größeren Blöcken zusammensetzen können: Wir definieren dazu

$$R_t := \bigcup_{r\in\operatorname{row}^*(t)} \hat{r}, \qquad C_s := \bigcup_{r\in\operatorname{col}^*(s)} \hat{r} \qquad \text{für alle } t \in \mathcal{T}_\mathcal{I},\ s \in \mathcal{T}_\mathcal{J}$$

und erhalten die folgende kompakte Notation:

Folgerung 8.13 (Vollständige Clusterbasis) *Sei $\mathbf{X} \in \mathbb{K}^{\mathcal{I}\times\mathcal{J}}$ eine $\mathcal{H}^2$-Matrix.*

Für jedes $t \in \mathcal{T}_\mathcal{I}$ existiert ein $\mathbf{B}_t \in \mathbb{K}^{R_t\times k}$ mit

$$\mathbf{X}|_{\hat{t}\times R_t} = \mathbf{V}_t \mathbf{B}_t^*.$$

Für jedes $s \in \mathcal{T}_\mathcal{J}$ existiert ein $\mathbf{A}_s \in \mathbb{K}^{C_s\times k}$ mit

$$\mathbf{X}|_{C_s\times\hat{s}} = \mathbf{A}_s \mathbf{W}_s^*.$$

Beweis. Sei $t \in \mathcal{T}_\mathcal{I}$. Wir definieren $\mathbf{B}_t$ blockweise mit Hilfe des Lemmas 8.11: Für jedes $s \in \operatorname{row}^*(t)$ finden wir ein $\mathbf{B}_{t,s} \in \mathbb{K}^{\hat{s}\times k}$ mit

$$\mathbf{X}|_{\hat{t}\times\hat{s}} = \mathbf{V}_t \mathbf{B}_{t,s}^*,$$

und da diese s nach Lemma 8.12 disjunkt sind, ist $\mathbf{B}_t$ durch

$$\mathbf{B}_t|_{\hat{s}\times k} = \mathbf{B}_{t,s} \qquad \text{für alle } s \in \operatorname{row}^*(t)$$

eindeutig und wohldefiniert.

Entsprechend können wir auch mit der Spaltenbasis verfahren. ∎

Für uns von entscheidender Bedeutung ist die Aussage, die wir dieser Folgerung für den Rang der Teilmatrizen einer $\mathcal{H}^2$-Matrix entnehmen können: Während bei einer $\mathcal{H}$-Matrix lediglich die zu jedem zulässigen Block $b = (t, s) \in \mathcal{L}^+_{\mathcal{I}\times\mathcal{J}}$ gehörende Teilmatrix $\mathbf{X}|_{\hat{t}\times\hat{s}}$ von niedrigem Rang sein muss, müssen bei einer $\mathcal{H}^2$-Matrix die wesentlich größeren Teilmatrizen $\mathbf{X}|_{\hat{t}\times R_t}$ und $\mathbf{X}|_{C_s\times\hat{s}}$ für alle Cluster $t \in \mathcal{T}_\mathcal{I}$ und $s \in \mathcal{T}_\mathcal{J}$ von niedrigem Rang sein. Der Unterschied ist in Abbildung 8.1 illustriert.

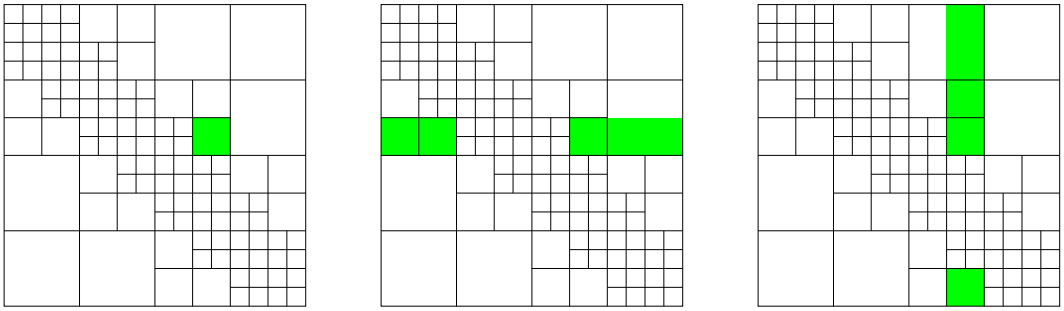

Abbildung 8.1: Vergleich zwischen $\mathcal{H}$- und $\mathcal{H}^2$-Matrix: Bei ersterer müssen nur einzelne Teilmatrizen niedrigen Rang besitzen, bei letzterer ganze Zeilen- und Spaltenblöcke.

8.4 Kompression

Wir haben bereits gesehen, dass bei einer $\mathcal{H}^2$-Matrix $\mathbf{X}$ die Teilmatrizen $\mathbf{X}|_{\hat{t}\times R_t}$ und $\mathbf{X}|_{C_s\times\hat{s}}$ für alle $t \in \mathcal{T}_\mathcal{I}$ und $s \in \mathcal{T}_\mathcal{J}$ höchstens von Rang k sind. Nun soll es darum gehen, zu beweisen, dass diese Eigenschaft schon ausreicht, um eine $\mathcal{H}^2$-Matrix-Darstellung für $\mathbf{X}$ zu finden.

Damit haben wir dann eine vollständige Charakterisierung der Menge der $\mathcal{H}^2$-Matrizen erreicht: Eine Matrix $\mathbf{X}$ kann genau dann als $\mathcal{H}^2$-Matrix dargestellt werden, falls $\mathbf{X}|_{\hat{t}\times R_t}$ und $\mathbf{X}|_{C_s\times\hat{s}}$ für alle $t \in \mathcal{T}_\mathcal{I}$ und $s \in \mathcal{T}_\mathcal{J}$ von niedrigem Rang sind.

Wir führen den Nachweis, indem wir eine praktisch durchführbare Konstruktion geeigneter Clusterbasen $\mathbf{V}$ und $\mathbf{W}$ entwickeln. Die Konstruktion [10] ist so allgemein, dass wir sie auch verwenden können, um eine Approximation statt einer exakten Darstellung der Matrix $\mathbf{X}$ zu konstruieren. Der Einfachheit halber beschränken wir uns dabei darauf, nur die Frobenius-Norm des Approximationsfehlers zu diskutieren.

Für die Konstruktion ist es ausreichend, lediglich die Zeilenbasis $\mathbf{V}$ zu untersuchen, denn die Spaltenbasis lässt sich konstruieren, indem wir eine Zeilenbasis für die adjungierte Matrix $\mathbf{X}^*$ mit dem entsprechend adjungierten Blockbaum suchen.

Unser Ziel ist es also nun, eine geschachtelte Clusterbasis $\mathbf{V}$ so zu finden, dass für jedes $t \in \mathcal{T}_\mathcal{I}$ eine Matrix $\mathbf{B}_t \in \mathbb{K}^{R_t\times k}$ mit

$$\|\mathbf{X}|_{\hat{t}\times R_t} - \mathbf{V}_t\mathbf{B}_t^*\|_F \leq \epsilon_t \tag{8.5}$$

für geeignete $\epsilon_t \in \mathbb{R}_{\geq 0}$ existiert. Wir führen die Abkürzung

$$\mathbf{X}_t := \mathbf{X}|_{\hat{t}\times R_t} \qquad \text{für alle } t \in \mathcal{T}_\mathcal{I}$$

ein und verlangen zur Vereinfachung der Rechnungen, dass die Matrizen $\mathbf{V}_t$ orthogonal sein sollen, dass also

$$\mathbf{V}_t^*\mathbf{V}_t = \mathbf{I} \qquad \text{für alle } t \in \mathcal{T}_\mathcal{I}$$

gelten soll, denn dann ist $\mathbf{V}_t\mathbf{V}_t^*$ eine orthogonale Projektion, die die bestmögliche Näherung von $\mathbf{X}_t$ in der Form (8.5) zur Verfügung stellt: Es gilt

$$\|\mathbf{X}_t - \mathbf{V}_t\mathbf{V}_t^*\mathbf{X}_t\|_F \leq \|\mathbf{X}_t - \mathbf{V}_t\mathbf{B}_t^*\|_F \qquad \text{für alle } \mathbf{B}_t \in \mathbb{K}^{R_t \times k}.$$

Zunächst untersuchen wir die Konstruktion der Matrizen $\mathbf{V}_t$ für Blattcluster. Sei dazu $t \in \mathcal{T}_\mathcal{I}$ ein Blatt des Clusterbaums. Da (8.5) eine Rang-k-Approximation der Matrix $\mathbf{X}_t$ beschreibt, bietet es sich an, ähnlich wie schon bei den $\mathcal{H}$-Matrizen auf die Singulärwertzerlegung zurückzugreifen. Sei also $(\Sigma, \widehat{\mathbf{U}}, \widehat{\mathbf{V}})$ eine Singulärwertzerlegung der Matrix $\mathbf{X}_t \in \mathbb{K}^{\hat{t}\times R_t}$, und seien $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_p > 0$ ihre Singulärwerte. Wir setzen nun $\mathbf{V}_t \in \mathbb{K}^{\hat{t}\times k}$ gleich den ersten k Spalten der Matrix $\widehat{\mathbf{U}}$, denn dann gilt

$$\mathbf{V}_t = \widehat{\mathbf{U}} \begin{pmatrix} 1 & & \\ & \ddots & \\ & & 1 \\ 0 & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & 0 \end{pmatrix}$$

und wir erhalten

$$\mathbf{V}_t\mathbf{V}_t^*\mathbf{X}_t = \widehat{\mathbf{U}} \begin{pmatrix} 1 & & & & & \\ & \ddots & & & & \\ & & 1 & & & \\ & & & 0 & & \\ & & & & \ddots & \\ & & & & & 0 \end{pmatrix} \widehat{\mathbf{U}}^*\widehat{\mathbf{U}}\boldsymbol{\Sigma}\widehat{\mathbf{V}}^* = \widehat{\mathbf{U}} \begin{pmatrix} \sigma_1 & & & & & \\ & \ddots & & & & \\ & & \sigma_k & & & \\ & & & 0 & & \\ & & & & \ddots & \\ & & & & & 0 \end{pmatrix} \widehat{\mathbf{V}}^*,$$

also stimmt $\mathbf{V}_t\mathbf{V}_t^*\mathbf{X}_t$ gerade mit der Rang-k-Bestapproximation aus Lemma 6.3 überein. Mit Hilfe von Lemma 6.4 gewinnen wir so die Abschätzung

$$\|\mathbf{X}_t - \mathbf{V}_t\mathbf{V}_t^*\mathbf{X}_t\|_F \leq \left(\sum_{\nu=k+1}^{p} \sigma_\nu^2\right)^{1/2}.$$

Durch geeignete Wahl von k können wir also wie schon im Fall der $\mathcal{H}$-Matrizen sicherstellen, dass

$$\|\mathbf{X}_t - \mathbf{V}_t\mathbf{V}_t^*\mathbf{X}_t\|_F \leq \hat{\epsilon}_t \tag{8.6}$$

gilt, dass der Fehler in der Frobenius-Norm somit durch ein gegebenes $\hat{\epsilon}_t \in \mathbb{R}_{\geq 0}$ beschränkt ist.

Damit haben wir unser Ziel für Blätter des Clusterbaums erreicht. Sei nun $t \in \mathcal{T}_\mathcal{I}$ ein Cluster, der kein Blatt ist. Zur Vereinfachung nehmen wir an, dass $\#\operatorname{sons}(t) = 2$ gilt und bezeichnen die beiden Söhne mit $\{t_1, t_2\} = \operatorname{sons}(t)$.

Wir suchen wieder eine Matrix $\mathbf{V}_t$, die (8.5) erfüllt, allerdings müssen wir berücksichtigen, dass die Clusterbasis geschachtelt ist, dass also

$$\mathbf{V}_t = \begin{pmatrix} \mathbf{V}_{t_1}\mathbf{E}_{t_1} \\ \mathbf{V}_{t_2}\mathbf{E}_{t_2} \end{pmatrix}$$

für geeignete Transfermatrizen $\mathbf{E}_{t_1}$ und $\mathbf{E}_{t_2}$ gelten muss. Da $\mathbf{V}_{t_1}$ und $\mathbf{V}_{t_2}$ orthogonal sind, ist auch

$$\mathbf{Q}_t := \begin{pmatrix} \mathbf{V}_{t_1} & \\ & \mathbf{V}_{t_2} \end{pmatrix}$$

eine orthogonale Matrix, und die Schachtelungseigenschaft lässt sich in der Form

$$\mathbf{V}_t = \begin{pmatrix} \mathbf{V}_{t_1} & \\ & \mathbf{V}_{t_2} \end{pmatrix} \begin{pmatrix} \mathbf{E}_{t_1} \\ \mathbf{E}_{t_2} \end{pmatrix} = \mathbf{Q}_t \widehat{\mathbf{V}}_t, \qquad \widehat{\mathbf{V}}_t := \begin{pmatrix} \mathbf{E}_{t_1} \\ \mathbf{E}_{t_2} \end{pmatrix} = \mathbf{Q}_t^* \mathbf{V}_t \in \mathbb{K}^{(2k)\times k}$$

schreiben. Die entscheidende Idee des Algorithmus besteht darin, den Fehler geeignet zu zerlegen: Aus der Orthogonalität der Matrix $\mathbf{Q}_t$ folgt

$$\begin{aligned} \|\mathbf{X}_t - \mathbf{V}_t\mathbf{V}_t^*\mathbf{X}_t\|_F^2 &= \|\mathbf{X}_t - \mathbf{Q}_t\mathbf{Q}_t^*\mathbf{X}_t + \mathbf{Q}_t\mathbf{Q}_t^*\mathbf{X}_t - \mathbf{Q}_t\widehat{\mathbf{V}}_t\widehat{\mathbf{V}}_t^*\mathbf{Q}_t^*\mathbf{X}_t\|_F^2 \\ &= \|\mathbf{X}_t - \mathbf{Q}_t\mathbf{Q}_t^*\mathbf{X}_t\|_F^2 + \|\mathbf{Q}_t(\mathbf{I} - \widehat{\mathbf{V}}_t\widehat{\mathbf{V}}_t^*)\widehat{\mathbf{X}}_t\|_F^2 \\ &\quad + 2\langle \mathbf{X}_t - \mathbf{Q}_t\mathbf{Q}_t^*\mathbf{X}_t, \mathbf{Q}_t(\mathbf{I} - \widehat{\mathbf{V}}_t\widehat{\mathbf{V}}_t^*)\widehat{\mathbf{X}}_t\rangle_F \end{aligned}$$

mit der Matrix

$$\widehat{\mathbf{X}}_t := \mathbf{Q}_t^*\mathbf{X}_t = \begin{pmatrix} \mathbf{V}_{t_1}^*\mathbf{X}|_{\hat{t}_1\times R_t} \\ \mathbf{V}_{t_2}^*\mathbf{X}|_{\hat{t}_2\times R_t} \end{pmatrix} \in \mathbb{K}^{(2k)\times R_t}.$$

Aus der Gleichung

$$\begin{aligned} \langle \mathbf{X}_t - \mathbf{Q}_t\mathbf{Q}_t^*\mathbf{X}_t, \mathbf{Q}_t(\mathbf{I} - \widehat{\mathbf{V}}_t\widehat{\mathbf{V}}_t^*)\widehat{\mathbf{X}}_t\rangle_F &= \langle \mathbf{Q}_t^*(\mathbf{X}_t - \mathbf{Q}_t\mathbf{Q}_t^*\mathbf{X}_t), \mathbf{I} - \widehat{\mathbf{V}}_t\widehat{\mathbf{V}}_t^*\widehat{\mathbf{X}}_t\rangle_F \\ &= \langle \mathbf{Q}_t^*\mathbf{X}_t - \mathbf{Q}_t^*\mathbf{X}_t, \mathbf{I} - \widehat{\mathbf{V}}_t\widehat{\mathbf{V}}_t^*\widehat{\mathbf{X}}_t\rangle_F = 0 \end{aligned}$$

und der Orthogonalität der Matrix $\mathbf{Q}_t$ folgt die Zerlegung

$$\|\mathbf{X}_t - \mathbf{V}_t\mathbf{V}_t^*\mathbf{X}_t\|_F^2 = \|\mathbf{X}_t - \mathbf{Q}_t\mathbf{Q}_t^*\mathbf{X}_t\|_F^2 + \|\widehat{\mathbf{X}}_t - \widehat{\mathbf{V}}_t\widehat{\mathbf{V}}_t^*\widehat{\mathbf{X}}_t\|_F^2. \tag{8.7}$$

In dieser Darstellung zerfällt der Fehler in zwei Anteile: Der erste Term der rechten Seite beschreibt den Fehler, den wir bereits bei der Wahl der Matrizen $\mathbf{V}_{t_1}$ und $\mathbf{V}_{t_2}$ in den Söhnen des Clusters t in Kauf genommen haben. Diesen Fehler können wir an diesem Punkt des Algorithmus nicht mehr beeinflussen.

Der zweite Term der Gleichung (8.7) dagegen beschreibt den Fehler, den wir durch die Wahl der Transfermatrizen im Cluster t hinzufügen, nämlich durch die Wahl der Matrix $\widehat{\mathbf{V}}_t$. Für unseren Algorithmus ist also nur dieser Teil der Fehlerdarstellung relevant, da wir nur ihn beeinflussen können. Offenbar geht es wieder darum, eine Matrix, in diesem Fall $\widehat{\mathbf{X}}_t$, durch eine Matrix niedrigen Rangs zu approximieren, denn da $\widehat{\mathbf{V}}_t$ lediglich k Spalten besitzt, kann die Matrix $\widehat{\mathbf{V}}_t\widehat{\mathbf{V}}_t^*\widehat{\mathbf{X}}_t$ höchstens von Rang k sein.

Bei der Wahl der Matrix $\widehat{\mathbf{V}}_t$ sind wir nicht völlig frei: Da $\mathbf{V}_t$ wieder orthogonal sein soll, folgt

$$\widehat{\mathbf{V}}_t^* \widehat{\mathbf{V}}_t = \widehat{\mathbf{V}}_t^* \mathbf{Q}_t^* \mathbf{Q}_t \widehat{\mathbf{V}}_t = (\mathbf{Q}_t \widehat{\mathbf{V}}_t)^* \mathbf{Q}_t \widehat{\mathbf{V}}_t = \mathbf{V}_t^* \mathbf{V}_t = \mathbf{I},$$

also muss $\widehat{\mathbf{V}}_t$ orthogonal sein, und die Matrix $\widehat{\mathbf{V}}_t \widehat{\mathbf{V}}_t^* \widehat{\mathbf{X}}_t$ ist die bestmögliche Näherung der Matrix $\widehat{\mathbf{X}}_t$.

Da unsere Aufgabe damit dieselbe Struktur wie im Fall des Blattclusters besitzt, können wir sie in derselben Weise lösen: Wir berechnen die Singulärwertzerlegung der Matrix $\widehat{\mathbf{X}}_t$ und konstruieren $\widehat{\mathbf{V}}_t$ aus den ersten k linken Singulärvektoren. Durch geeignete Wahl des Rangs k können wir wieder

$$\|\widehat{\mathbf{X}}_t - \widehat{\mathbf{V}}_t \widehat{\mathbf{V}}_t^* \widehat{\mathbf{X}}_t\|_F \leq \hat{\epsilon}_t \tag{8.8}$$

für jedes gegebene $\hat{\epsilon}_t \in \mathbb{R}_{\geq 0}$ sicherstellen.

Eigentlich sind wir allerdings an Abschätzungen des Typs (8.5) für $\mathbf{X}_t$ interessiert. Glücklicherweise lassen sie sich relativ einfach mit Hilfe der Gleichung (8.7) gewinnen, indem Fehleranteile der Nachfahren des Clusters t aufsummiert werden. Zur Vermeidung von Fallunterscheidungen führen wir $\widehat{\mathbf{V}}_t$ und $\widehat{\mathbf{X}}_t$ auch für Blattcluster ein:

$$\widehat{\mathbf{V}}_t = \begin{cases} \mathbf{V}_t & \text{falls } \operatorname{sons}(t) = \emptyset, \\ \mathbf{Q}_t^* \mathbf{V}_t & \text{ansonsten} \end{cases} \qquad \text{für alle } t \in \mathcal{T}_\mathcal{I},$$

$$\widehat{\mathbf{X}}_t = \begin{cases} \mathbf{X}_t & \text{falls } \operatorname{sons}(t) = \emptyset, \\ \mathbf{Q}_t^* \mathbf{X}_t & \text{ansonsten} \end{cases} \qquad \text{für alle } t \in \mathcal{T}_\mathcal{I}.$$

Wir beschränken uns zunächst auf eine einfache Abschätzung des Fehlers: Für den ersten Term der Gleichung (8.7) erhalten wir wegen $R_{t_1}, R_{t_2} \supseteq R_t$ die Schranke

$$\begin{aligned} \|\mathbf{X}_t - \mathbf{Q}_t \mathbf{Q}_t^* \mathbf{X}_t\|_F^2 &= \|\mathbf{X}_t|_{\hat{t}_1 \times R_t} - \mathbf{V}_{t_1} \mathbf{V}_{t_1}^* \mathbf{X}_t|_{\hat{t}_1 \times R_t}\|_F^2 + \|\mathbf{X}_t|_{\hat{t}_2 \times R_t} - \mathbf{V}_{t_2} \mathbf{V}_{t_2}^* \mathbf{X}_t|_{\hat{t}_2 \times R_t}\|_F^2 \\ &\leq \|\mathbf{X}_{t_1} - \mathbf{V}_{t_1} \mathbf{V}_{t_1}^* \mathbf{X}_{t_1}\|_F^2 + \|\mathbf{X}_{t_2} - \mathbf{V}_{t_2} \mathbf{V}_{t_2}^* \mathbf{X}_{t_2}\|_F^2, \end{aligned}$$

so dass sich insgesamt

$$\|\mathbf{X}_t - \mathbf{V}_t \mathbf{V}_t^* \mathbf{X}_t\|_F^2 \leq \hat{\epsilon}_t^2 + \|\mathbf{X}_{t_1} - \mathbf{V}_{t_1} \mathbf{V}_{t_1}^* \mathbf{X}_{t_1}\|_F^2 + \|\mathbf{X}_{t_2} - \mathbf{V}_{t_2} \mathbf{V}_{t_2}^* \mathbf{X}_{t_2}\|_F^2$$

ergibt und wir mit einer einfachen Induktion

$$\|\mathbf{X}_t - \mathbf{V}_t \mathbf{V}_t^* \mathbf{X}_t\|_F^2 \leq \sum_{r \in \operatorname{sons}^*(t)} \hat{\epsilon}_r^2$$

erhalten. Indem wir also die „lokalen“ Fehlerschranken $\hat{\epsilon}_r$ für die Nachfahren $r \in \operatorname{sons}^*(t)$ des Clusters t geeignet wählen, können wir jede gewünschte Genauigkeit erreichen.

Fehlerabschätzungen für die Gesamtmatrix lassen sich elegant herleiten, indem wir sie für einzelne Blöcke untersuchen: Wir setzen

$$\mathbf{X}_{t,s} := \mathbf{X}|_{\hat{t} \times \hat{s}}, \qquad \widehat{\mathbf{X}}_{t,s} := \widehat{\mathbf{X}}_t|_{(2k) \times \hat{s}} \qquad \text{für alle } t \in \mathcal{T}_\mathcal{I},\ s \in \operatorname{row}^*(t)$$

und erhalten analog zu (8.7) die folgende Fehlerdarstellung:

Lemma 8.14 (Blockfehler) *Sei $t \in \mathcal{T}_\mathcal{I}$ und $s \in \operatorname{row}^*(t)$. Dann gilt*

$$\|\mathbf{X}_{t,s} - \mathbf{V}_t \mathbf{V}_t^* \mathbf{X}_{t,s}\|_F^2 = \sum_{r \in \operatorname{sons}^*(t)} \|\widehat{\mathbf{X}}_{r,s} - \widehat{\mathbf{V}}_r \widehat{\mathbf{V}}_r^* \widehat{\mathbf{X}}_{r,s}\|_F^2.$$

Beweis. Per Induktion über $\#\operatorname{sons}^*(t) \in \mathbb{N}$. Für $t \in \mathcal{T}_\mathcal{I}$ mit $\#\operatorname{sons}^*(t) = 1$ gilt $\operatorname{sons}(t) = \emptyset$, also $\widehat{\mathbf{X}}_{t,s} = \mathbf{X}_{t,s}$ und $\widehat{\mathbf{V}}_t = \mathbf{V}_t$, also ist die Gleichung trivial.

Sei $n \in \mathbb{N}$ nun so gewählt, dass die Gleichung für alle $t \in \mathcal{T}_\mathcal{I}$ mit $\#\operatorname{sons}(t) \leq n$ gilt. Sei ein $t \in \mathcal{T}_\mathcal{I}$ mit $\#\operatorname{sons}(t) = n+1$ fixiert. Entsprechend (8.7) erhalten wir

$$\|\mathbf{X}_{t,s} - \mathbf{V}_t \mathbf{V}_t^* \mathbf{X}_{t,s}\|_F^2 = \|\mathbf{X}_{t,s} - \mathbf{Q}_t \mathbf{Q}_t^* \mathbf{X}_{t,s}\|_F^2 + \|\widehat{\mathbf{X}}_{t,s} - \widehat{\mathbf{V}}_t \widehat{\mathbf{V}}_t^* \widehat{\mathbf{X}}_{t,s}\|_F^2.$$

Da $R_{t'} \supseteq R_t$ für alle $t' \in \operatorname{sons}(t)$ gilt, können wir im ersten Term $\mathbf{X}_{t,s}$ aus den Teilmatrizen $\mathbf{X}_{t',s}$ zusammensetzen und erhalten infolge der Blockdiagonalgestalt der Matrix $\mathbf{Q}_t \mathbf{Q}_t^*$ die Darstellung

$$\|\mathbf{X}_{t,s} - \mathbf{V}_t \mathbf{V}_t^* \mathbf{X}_{t,s}\|_F^2 = \|\widehat{\mathbf{X}}_{t,s} - \widehat{\mathbf{V}}_t \widehat{\mathbf{V}}_t^* \widehat{\mathbf{X}}_{t,s}\|_F^2 + \sum_{t' \in \operatorname{sons}(t)} \|\mathbf{X}_{t',s} - \mathbf{V}_{t'} \mathbf{V}_{t'}^* \mathbf{X}_{t',s}\|_F^2.$$

Da für jeden Sohn $t' \in \operatorname{sons}(t)$ insbesondere $\#\operatorname{sons}^*(t') < \#\operatorname{sons}^*(t) = n+1$ gilt, können wir die Induktionsvoraussetzung anwenden, um

$$\begin{aligned}
\|\mathbf{X}_{t,s} - \mathbf{V}_t \mathbf{V}_t^* \mathbf{X}_{t,s}\|_F^2 &= \|\widehat{\mathbf{X}}_{t,s} - \widehat{\mathbf{V}}_t \widehat{\mathbf{V}}_t^* \widehat{\mathbf{X}}_{t,s}\|_F^2 + \sum_{t' \in \operatorname{sons}(t)} \|\mathbf{X}_{t',s} - \mathbf{V}_{t'} \mathbf{V}_{t'}^* \mathbf{X}_{t',s}\|_F^2 \\
&= \|\widehat{\mathbf{X}}_{t,s} - \widehat{\mathbf{V}}_t \widehat{\mathbf{V}}_t^* \widehat{\mathbf{X}}_{t,s}\|_F^2 + \sum_{t' \in \operatorname{sons}(t)} \sum_{r \in \operatorname{sons}^*(t')} \|\widehat{\mathbf{X}}_{r,s} - \widehat{\mathbf{V}}_r \widehat{\mathbf{V}}_r^* \widehat{\mathbf{X}}_{r,s}\|_F^2 \\
&= \sum_{r \in \operatorname{sons}^*(t)} \|\widehat{\mathbf{X}}_{r,s} - \widehat{\mathbf{V}}_r \widehat{\mathbf{V}}_r^* \widehat{\mathbf{X}}_{r,s}\|_F^2
\end{aligned}$$

zu erhalten. Damit ist die Induktion vollständig. ∎

Mit Hilfe dieser Aussage können wir immerhin eine „halbe $\mathcal{H}^2$-Matrix“ definieren, indem wir die zulässigen Blöcke mit Hilfe der Zeilenbasis $\mathbf{V}$ approximieren, aber auf eine Spaltenbasis verzichten: Wir definieren $\widetilde{\mathbf{X}} \in \mathbb{K}^{\mathcal{I} \times \mathcal{J}}$ durch

$$\widetilde{\mathbf{X}}|_{\hat{t} \times \hat{s}} := \begin{cases} \mathbf{V}_t \mathbf{V}_t^* \mathbf{X}|_{\hat{t} \times \hat{s}} & \text{falls } b = (t,s) \in \mathcal{L}^+_{\mathcal{I} \times \mathcal{J}}, \\ \mathbf{X}|_{\hat{t} \times \hat{s}} & \text{ansonsten} \end{cases} \qquad \text{für alle } b = (t,s) \in \mathcal{L}_{\mathcal{I} \times \mathcal{J}}.$$

Für diesen Zwischenschritt erhalten wir eine exakte Fehlerdarstellung:

Satz 8.15 (Gesamtfehler) *Es gilt*

$$\|\mathbf{X} - \widetilde{\mathbf{X}}\|_F^2 = \sum_{t \in \mathcal{T}_\mathcal{I}} \|\widehat{\mathbf{X}}_t - \widehat{\mathbf{V}}_t \widehat{\mathbf{V}}_t^* \widehat{\mathbf{X}}_t\|_F^2.$$

Beweis. Wir zerlegen den Fehler mit Hilfe der Folgerung 2.22 wie üblich in Beiträge der einzelnen Blöcke und wenden Lemma 8.14 an:

$$\begin{aligned}
\|\mathbf{X} - \widetilde{\mathbf{X}}\|_F^2 &= \sum_{b=(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}^+} \|\mathbf{X}_{t,s} - \mathbf{V}_t\mathbf{V}_t^*\mathbf{X}_{t,s}\|_F^2 \\
&= \sum_{b=(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}^+} \sum_{r\in\operatorname{sons}^*(t)} \|\widehat{\mathbf{X}}_{r,s} - \widehat{\mathbf{V}}_r\widehat{\mathbf{V}}_r^*\widehat{\mathbf{X}}_{r,s}\|_F^2 \\
&= \sum_{t\in\mathcal{T}_\mathcal{I}} \sum_{r\in\operatorname{sons}^*(t)} \sum_{\substack{s\in\operatorname{row}(t)\\(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}^+}} \|\widehat{\mathbf{X}}_{r,s} - \widehat{\mathbf{V}}_r\widehat{\mathbf{V}}_r^*\widehat{\mathbf{X}}_{r,s}\|_F^2 \\
&= \sum_{r\in\mathcal{T}_\mathcal{I}} \sum_{t\in\operatorname{pred}(r)} \sum_{\substack{s\in\operatorname{row}(t)\\(t,s)\in\mathcal{L}_{\mathcal{I}\times\mathcal{J}}^+}} \|\widehat{\mathbf{X}}_{r,s} - \widehat{\mathbf{V}}_r\widehat{\mathbf{V}}_r^*\widehat{\mathbf{X}}_{r,s}\|_F^2 \\
&= \sum_{r\in\mathcal{T}_\mathcal{I}} \sum_{s\in\operatorname{row}^*(r)} \|\widehat{\mathbf{X}}_{r,s} - \widehat{\mathbf{V}}_r\widehat{\mathbf{V}}_r^*\widehat{\mathbf{X}}_{r,s}\|_F^2 \\
&= \sum_{r\in\mathcal{T}_\mathcal{I}} \|\widehat{\mathbf{X}}_r - \widehat{\mathbf{V}}_r\widehat{\mathbf{V}}_r^*\widehat{\mathbf{X}}_r\|_F^2.
\end{aligned}$$

■

Um die Matrix $\mathbf{X}$ durch eine $\mathcal{H}^2$-Matrix zu approximieren wenden wir die Konstruktion auf Zeilen und Spalten an: Für jedes zulässige Blatt $b = (t, s) \in \mathcal{L}_{\mathcal{I}\times\mathcal{J}}^+$ gilt

$$\begin{aligned}
&\|\mathbf{X}|_{\hat{t}\times\hat{s}} - \mathbf{V}_t\mathbf{V}_t^*\mathbf{X}|_{\hat{t}\times\hat{s}}\mathbf{W}_s\mathbf{W}_s^*\|_F^2 \\
&\quad= \|\mathbf{X}|_{\hat{t}\times\hat{s}} - \mathbf{V}_t\mathbf{V}_t^*\mathbf{X}|_{\hat{t}\times\hat{s}} + \mathbf{V}_t\mathbf{V}_t^*\mathbf{X}|_{\hat{t}\times\hat{s}} - \mathbf{V}_t\mathbf{V}_t^*\mathbf{X}|_{\hat{t}\times\hat{s}}\mathbf{W}_s\mathbf{W}_s^*\|_F^2 \\
&\quad= \|\mathbf{X}|_{\hat{t}\times\hat{s}} - \mathbf{V}_t\mathbf{V}_t^*\mathbf{X}|_{\hat{t}\times\hat{s}}\|_F^2 + \|\mathbf{V}_t\mathbf{V}_t^*(\mathbf{X}|_{\hat{t}\times\hat{s}} - \mathbf{X}|_{\hat{t}\times\hat{s}}\mathbf{W}_s\mathbf{W}_s^*)\|_F^2 \\
&\qquad+ 2\langle\mathbf{X}|_{\hat{t}\times\hat{s}} - \mathbf{V}_t\mathbf{V}_t^*\mathbf{X}|_{\hat{t}\times\hat{s}}, \mathbf{V}_t\mathbf{V}_t^*(\mathbf{X}|_{\hat{t}\times\hat{s}} - \mathbf{X}|_{\hat{t}\times\hat{s}}\mathbf{W}_s\mathbf{W}_s^*)\rangle_F \\
&\quad= \|\mathbf{X}|_{\hat{t}\times\hat{s}} - \mathbf{V}_t\mathbf{V}_t^*\mathbf{X}|_{\hat{t}\times\hat{s}}\|_F^2 + \|\mathbf{V}_t\mathbf{V}_t^*(\mathbf{X}|_{\hat{t}\times\hat{s}} - \mathbf{X}|_{\hat{t}\times\hat{s}}\mathbf{W}_s\mathbf{W}_s^*)\|_F^2 \\
&\quad\leq \|\mathbf{X}|_{\hat{t}\times\hat{s}} - \mathbf{V}_t\mathbf{V}_t^*\mathbf{X}|_{\hat{t}\times\hat{s}}\|_F^2 + \|\mathbf{X}|_{\hat{t}\times\hat{s}} - \mathbf{X}|_{\hat{t}\times\hat{s}}\mathbf{W}_s\mathbf{W}_s^*\|_F^2 \\
&\quad\leq \|\mathbf{X}|_{\hat{t}\times\hat{s}} - \mathbf{V}_t\mathbf{V}_t^*\mathbf{X}|_{\hat{t}\times\hat{s}}\|_F^2 + \|\mathbf{X}|_{\hat{t}\times\hat{s}}^* - \mathbf{W}_s\mathbf{W}_s^*\mathbf{X}|_{\hat{t}\times\hat{s}}^*\|_F^2,
\end{aligned}$$

also können wir Zeilen- und Spaltenbasen unabhängig voneinander konstruieren. Für die Spaltenbasen wenden wir die bereits vorgestellte Konstruktion auf die adjungierte Matrix $\mathbf{X}^*$ an und können so auch die bereits gewonnenen Aussagen über den Fehler übertragen.

Um aus dem theoretischen Algorithmus eine praktisch durchführbare Rechenvorschrift zu machen, ist es hilfreich, die Hilfsgrößen

$$\mathbf{Z}_{t,s} := \mathbf{V}_t^*\mathbf{X}|_{\hat{t}\times\hat{s}} \qquad \text{für alle } t \in \mathcal{T}_\mathcal{I},\ s \in \operatorname{row}^*(t)$$

einzuführen, denn aus ihnen lassen sich die für den Algorithmus entscheidenden Matrizen

$$\widehat{\mathbf{X}}_t = \begin{pmatrix} \mathbf{V}_{t_1}^* & \\ & \mathbf{V}_{t_2}^* \end{pmatrix} \mathbf{X}|_{\hat{t}\times\hat{s}} = \begin{pmatrix} \mathbf{V}_{t_1}^*\mathbf{X}|_{\hat{t}_1\times\hat{s}} \\ \mathbf{V}_{t_2}^*\mathbf{X}|_{\hat{t}_2\times\hat{s}} \end{pmatrix} = \begin{pmatrix} \mathbf{Z}_{t_1,s} \\ \mathbf{Z}_{t_2,s} \end{pmatrix}$$

für $\operatorname{sons}(t) = \{t_1, t_2\}$ einfach zusammensetzen. Wichtiger ist, dass sich diese Matrizen auch rekursiv berechnen lassen: Es gilt nämlich

$$\begin{aligned}\mathbf{Z}_{t,s} = \mathbf{V}_t^* \mathbf{X}|_{\hat{t}\times\hat{s}} &= \begin{pmatrix}\mathbf{V}_{t_1}\mathbf{E}_{t_1}\\ \mathbf{V}_{t_2}\mathbf{E}_{t_2}\end{pmatrix}\begin{pmatrix}\mathbf{X}|_{\hat{t}_1\times\hat{s}}\\ \mathbf{X}|_{\hat{t}_2\times\hat{s}}\end{pmatrix}\\ &= \begin{pmatrix}\mathbf{E}_{t_1}^* & \mathbf{E}_{t_2}^*\end{pmatrix}\begin{pmatrix}\mathbf{V}_{t_1}^*\mathbf{X}|_{\hat{t}_1\times\hat{s}}\\ \mathbf{V}_{t_2}^*\mathbf{X}|_{\hat{t}_2\times\hat{s}}\end{pmatrix} = \widehat{\mathbf{V}}_t^*\widehat{\mathbf{X}}_{t,s}\end{aligned}$$

Unter Ausnutzung dieser Gleichung erhalten wir den in Abbildung 8.2 dargestellten Algorithmus zur Bestimmung der Zeilenbasis $\mathbf{V}$.

---

procedure approx_h2(t, $\mathbf{X}$, ϵ, **var** $\mathbf{V}$, $\mathbf{E}$, $\mathbf{Z}$);
if $\operatorname{sons}(t) = \emptyset$ **then**
  for $s \in \operatorname{row}^*(t)$ **do**
    $\mathbf{X}_{t,s} \leftarrow \mathbf{X}|_{\hat{t}\times\hat{s}}$; $\mathbf{X}_t|_{\hat{t}\times\hat{s}} \leftarrow \mathbf{X}_{t,s}$
  end for;
  Berechne $\mathbf{V}_t$ aus der Singulärwertzerlegung von $\mathbf{X}_t$;
  for $s \in \operatorname{row}^*(t)$ **do**
    $\mathbf{Z}_{t,s} \leftarrow \mathbf{V}_t^*\mathbf{X}_{t,s}$
  end for
else
  $\tau \leftarrow \#\operatorname{sons}(t)$; $\{t_1, \ldots, t_\tau\} \leftarrow \operatorname{sons}(t)$;
  for $t' \in \operatorname{sons}(t)$ **do**
    approx_h2(t', $\mathbf{X}$, ϵ, $\mathbf{V}$, $\mathbf{E}$, $\mathbf{Z}$);
  end for;
  for $s \in \operatorname{row}^*(t)$ **do**
    $\widehat{\mathbf{X}}_{t,s} \leftarrow \begin{pmatrix}\mathbf{Z}_{t_1,s}\\ \vdots\\ \mathbf{Z}_{t_\tau,s}\end{pmatrix} \in \mathbb{K}^{(\tau k)\times\hat{s}}$; $\widehat{\mathbf{X}}_t|_{(\tau k)\times\hat{s}} \leftarrow \widehat{\mathbf{X}}_{t,s}$
  end for;
  Berechne $\widehat{\mathbf{V}}_t$ (und damit $\mathbf{E}_{t_1}, \ldots, \mathbf{E}_{t_\tau}$) aus der Singulärwertzerlegung von $\widehat{\mathbf{X}}_t$;
  for $s \in \operatorname{row}^*(t)$ **do**
    $\mathbf{Z}_{t,s} \leftarrow \widehat{\mathbf{V}}_t^*\widehat{\mathbf{X}}_{t,s}$
  end for
end if

Abbildung 8.2: Konstruktion einer Zeilenbasis $\mathbf{V}$ für eine Matrix $\mathbf{X}$

Da der Algorithmus ohnehin jeden Eintrag der Matrix $\mathbf{X}$ mindestens einmal untersuchen muss, kann er nicht mit weniger als $(\#\mathcal{I})(\#\mathcal{J})$ Operationen auskommen, besitzt also mindestens quadratische Komplexität.

Lemma 8.16 (Aufwand $\mathcal{H}^2$-Basisbestimmung) *Sei $\mathbf{X} \in \mathbb{K}^{\mathcal{I}\times\mathcal{J}}$. Gelte $\#\hat{t} \leq r_{\mathcal{I}}$ für jeden Blattcluster $t \in \mathcal{L}_{\mathcal{I}}$ und gelte $\#\operatorname{sons}(t) \leq C_{sn}$ für jeden Nicht-Blattcluster $t \in$*

$\mathcal{T}_\mathcal{I} \setminus \mathcal{L}_\mathcal{I}$. Sei C_{svd} die Konstante aus Bemerkung 6.3. Dann benötigt der Algorithmus aus Abbildung 8.2 nicht mehr als

$$(C_{svd}+2)\max\{k,r_\mathcal{I}\}(\#\mathcal{I}+C_{sn}^2 k\#\mathcal{T}_\mathcal{I})(\#\mathcal{J}) \quad \textit{Operationen.}$$

Beweis. Sei $t \in \mathcal{T}_\mathcal{I}$. Falls t ein Blatt ist, erfordert die Berechnung der Singulärwertzerlegung der Matrix $\mathbf{X}_t \in \mathbb{K}^{\hat{t}\times R_t}$ wegen $R_t \subseteq \mathcal{J}$ und $\#\hat{t} \leq r_\mathcal{I}$ laut Bemerkung 6.3 nicht mehr als

$$C_{\mathrm{svd}} r_\mathcal{I}(\#\hat{t})(\#\mathcal{J}) \quad \text{Operationen.}$$

Die Berechnung der Matrizen $\mathbf{Z}_{t,s}$ erfordert laut Lemma 8.12 nicht mehr als

$$\sum_{s\in\mathrm{row}^*(t)} 2k(\#\hat{t})(\#\hat{s}) \leq 2k(\#\hat{t})(\#\mathcal{J}) \quad \text{Operationen,}$$

so dass insgesamt nicht mehr als

$$(C_{\mathrm{svd}}+2)\max\{k,r_\mathcal{I}\}(\#\hat{t})(\#\mathcal{J}) \quad \text{Operationen}$$

anfallen. Nach Folgerung 2.21 genügen somit für *alle* Blattcluster

$$\sum_{t\in\mathcal{L}_\mathcal{I}} (C_{\mathrm{svd}}+2)\max\{k,r_\mathcal{I}\}(\#\hat{t})(\#\mathcal{J}) = (C_{\mathrm{svd}}+2)\max\{k,r_\mathcal{I}\}(\#\mathcal{I})(\#\mathcal{J}) \quad \text{Operationen.}$$

Falls dagegen t kein Blatt ist, erfordert die Berechnung der Singulärwertzerlegung der Matrix $\widehat{\mathbf{X}}_t \in \mathbb{K}^{(\tau k)\times R_t}$ wegen $\tau \leq C_{\mathrm{sn}}$ und $R_t \subseteq \mathcal{J}$ nicht mehr als

$$C_{\mathrm{svd}} C_{\mathrm{sn}}^2 k^2(\#\mathcal{J}) \quad \text{Operationen.}$$

Die Berechnung der Matrizen $\mathbf{Z}_{t,s}$ erfordert nicht mehr als

$$\sum_{s\in\mathrm{row}^*(t)} 2k(C_{\mathrm{sn}}k)(\#\hat{s}) \leq 2C_{\mathrm{sn}}k^2(\#\mathcal{J}) \quad \text{Operationen,}$$

so dass insgesamt nicht mehr als

$$(C_{\mathrm{svd}}C_{\mathrm{sn}}^2 + 2C_{\mathrm{sn}})k^2(\#\mathcal{J}) \leq (C_{\mathrm{svd}}+2)\max\{k,r_\mathcal{I}\}C_{\mathrm{sn}}^2 k(\#\mathcal{J}) \quad \text{Operationen}$$

anfallen. Mit Hilfe der groben Abschätzung $\#(\mathcal{T}_\mathcal{I} \setminus \mathcal{L}_\mathcal{I}) \leq \#\mathcal{T}_\mathcal{I}$ folgt die Behauptung. ∎

Mit $n = \#\mathcal{I}$ und $m = \#\mathcal{J}$ und mit den üblichen Annahmen $r_\mathcal{I} \leq k$ und $\#\mathcal{T}_\mathcal{I} \sim n/k$ erhalten wir aus dieser Aufwandsabschätzung, dass die Anzahl der Rechenoperationen höchstens wie $\sim knm$ wachsen kann. Wie wir bereits gesehen haben sind mindestens nm Operationen für jeden sinnvollen Algorithmus erforderlich, insofern ist die Komplexität des Algorithmus praktisch optimal.

Wir dürfen also festhalten, dass wir mit einem zu $k(\#\mathcal{I})(\#\mathcal{J})$ proportionalen Aufwand jede beliebige Matrix $\mathbf{X} \in \mathbb{K}^{\mathcal{I}\times\mathcal{J}}$ durch eine $\mathcal{H}^2$-Matrix approximieren können und dass sich die Genauigkeit der Approximation mit Hilfe des Satzes 8.15 steuern lässt.

Bemerkung 8.17 (Datenschwache Ausgangsmatrix) *Eine quadratische Komplexität ist nur dann zufriedenstellend, wenn auch die Anzahl der Eingabedaten quadratisch wächst. Falls* $\mathbf{X}$ *dagegen in einer effizienteren Darstellung vorliegt, wäre es wünschenswert, auch den Approximationsalgorihmus effizienter durchführen zu können.*

Das ist tatsächlich möglich [10, 5, 7]: Falls $\mathbf{X}$ *eine hierarchische Matrix ist, können wir die Matrizen* $\mathbf{X}_{t,s}$ *und* $\widehat{\mathbf{X}}_{t,s}$ *unter Ausnutzung ihrer Niedrigrangstruktur kompakter darstellen und so den Rechenaufwand für die Kompression auf* $\sim nk^2 \log_2 n$ *reduzieren.*

Als „Nebenprodukt“ unseres Approximationsalgorithmus erhalten wir eine vollständig implizite Charakterisierung der Menge aller durch $\mathcal{H}^2$-Matrizen exakt darstellbaren Matrizen:

Bemerkung 8.18 (Exakte Darstellung) *Falls die Teilmatrizen* $\mathbf{X}|_{\hat{t}\times R_t}$ *und* $\mathbf{X}|_{C_s\times\hat{s}}$ *aus Folgerung 8.13 jeweils höchstens Rang* k *besitzen, gilt dasselbe auch für die im Approximationsalgorithmus auftretenden Matrizen* $\widehat{\mathbf{X}}_t$*. Damit berechnet die Singulärwertzerlegung nicht nur eine Approximation, sondern eine* exakte *Darstellung, so dass auch die Matrix* $\mathbf{X}$ *sich in den mit Hilfe unseres Algorithmus gewonnenen Zeilen- und Spaltenbasen exakt darstellen lässt.*

Also ist bewiesen, dass die Matrix $\mathbf{X}$ *genau dann eine* $\mathcal{H}^2$*-Matrix ist, wenn die Matrizen aus Folgerung 8.13 höchstens von Rang* k *sind.*

Index

Literaturverzeichnis

[1] M. Bebendorf. Approximation of boundary element matrices. *Numer. Math.*, 86(4):565–589, 2000.

[2] M. Bebendorf. *Effiziente numerische Lösung von Randintegralgleichungen unter Verwendung von Niedrigrang-Matrizen.* Doctoral thesis, Universität Saarbrücken, 2000.

[3] M. Bebendorf and R. Grzhibovskis. Accelerating Galerkin BEM for linear elasticity using adaptive cross approximation. *Math. Meth. Appl. Sci.*, 29:1721–1747, 2006.

[4] M. Bebendorf and S. Rjasanow. Adaptive Low-Rank Approximation of Collocation Matrices. *Computing*, 70(1):1–24, 2003.

[5] S. Börm. Approximation of integral operators by $\mathcal{H}^2$-matrices with adaptive bases. *Computing*, 74(3):249–271, 2005.

[6] S. Börm. *$\mathcal{H}^2$-Matrices — An efficient tool for the treatment of dense matrices.* Habilitationsschrift, Christian-Albrechts-Universität zu Kiel, 2006.

[7] S. Börm. Adaptive variable-rank approximation of dense matrices. *SIAM J. Sci. Comp.*, 30(1):148–168, 2007.

[8] S. Börm. Data-sparse approximation of non-local operators by $\mathcal{H}^2$-matrices. *Lin. Alg. Appl.*, 422:380–403, 2007.

[9] S. Börm and L. Grasedyck. Hybrid cross approximation of integral operators. *Numer. Math.*, 101:221–249, 2005.

[10] S. Börm and W. Hackbusch. Data-sparse approximation by adaptive $\mathcal{H}^2$-matrices. *Computing*, 69:1–35, 2002.

[11] S. Börm and W. Hackbusch. $\mathcal{H}^2$-matrix approximation of integral operators by interpolation. *Appl. Numer. Math.*, 43:129–143, 2002.

[12] S. Börm, M. Löhndorf, and J. M. Melenk. Approximation of integral operators by variable-order interpolation. *Numer. Math.*, 99(4):605–643, 2005.

[13] S. Börm and S. A. Sauter. BEM with linear complexity for the classical boundary integral operators. *Math. Comp.*, 74:1139–1177, 2005.

[14] R. A. DeVore and G. G. Lorentz. *Constructive Approximation.* Springer-Verlag, 1993.

[15] Z. Drmač and K. Veselić. New fast and accurate Jacobi SVD algorithm I. *SIAM J. Matrix Anal. Appl.*, 29:1322–1342, 2008.

[16] Z. Drmač and K. Veselić. New fast and accurate Jacobi SVD algorithm II. *SIAM J. Matrix Anal. Appl.*, 29:1343–1362, 2008.

[17] G. H. Golub and W. Kahan. Calculating the singular values and pseudo-inverse of a matrix. *Journal of the Society for Industrial and Applied Mathematics: Series B, Numerical Analysis*, 2(2):205–224, 1965.

[18] S. A. Goreinov, E. E. Tyrtyshnikov, and N. L. Zamarashkin. A theory of pseudoskeleton approximations. *Lin. Alg. Appl.*, 261:1–22, 1997.

[19] L. Grasedyck. *Theorie und Anwendungen Hierarchischer Matrizen.* Doctoral thesis, Universität Kiel, 2001.

[20] L. Grasedyck. Adaptive recompression of $\mathcal{H}$-matrices for BEM. *Computing*, 74(3):205–223, 2004.

[21] L. Grasedyck and W. Hackbusch. Construction and arithmetics of $\mathcal{H}$-matrices. *Computing*, 70:295–334, 2003.

[22] L. Grasedyck, R. Kriemann, and S. LeBorne. Domain decomposition based $\mathcal{H}$-LU preconditioning. *Numer. Math.*, 112(4):565–600, 2009.

[23] W. Hackbusch. *Iterative Solution of Large Sparse Systems.* Springer-Verlag New York, 1994.

[24] W. Hackbusch. *Integralgleichungen. Theorie und Numerik.* Vieweg+Teubner, 1997.

[25] W. Hackbusch. A sparse matrix arithmetic based on $\mathcal{H}$-matrices. Part I: Introduction to $\mathcal{H}$-matrices. *Computing*, 62(2):89–108, 1999.

[26] W. Hackbusch. *Hierarchische Matrizen — Algorithmen und Analysis.* Springer, 2009.

[27] W. Hackbusch and B. N. Khoromskij. A sparse matrix arithmetic based on $\mathcal{H}$-matrices. Part II: Application to multi-dimensional problems. *Computing*, 64:21–47, 2000.

[28] W. Hackbusch, B. N. Khoromskij, and S. A. Sauter. On $\mathcal{H}^2$-matrices. In H. Bungartz, R. Hoppe, and C. Zenger, editors, *Lectures on Applied Mathematics*, pages 9–29. Springer-Verlag, Berlin, 2000.

[29] M. Lintner. The eigenvalue problem for the 2d Laplacian in $\mathcal{H}$-matrix arithmetic and application to the heat and wave equation. *Computing*, 72:293–323, 2004.

[30] T. J. Rivlin. *The Chebyshev Polynomials.* Wiley-Interscience, New York, 1990.

[31] Y. Saad. *Iterative Methods for Sparse Linear Systems.* Society for Industrial Mathematics, 2nd edition, 2003.

[32] S. A. Sauter. Variable order panel clustering. *Computing*, 64:223–261, 2000.

[33] S. A. Sauter and C. Schwab. *Randelementmethoden.* Teubner, 2004.

[34] O. Steinbach. *Numerische Näherungsverfahren für elliptische Randwertprobleme.* Teubner, 2003.

[35] J. Stoer. *Einführung in die Numerische Mathematik I.* Springer, 5th edition edition, 1989.

[36] E. E. Tyrtyshnikov. Incomplete cross approximation in the mosaic-skeleton method. *Computing*, 64:367–380, 2000.

[37] J. Wloka. *Partial differential equations.* Cambridge University Press, 1987.