Professur für Mathematik
Fakultät für Elektrische Energiesysteme
und Informationstechnik

der Bundeswehr Universität München

Praktikum Numerische Simulation in der Technik

Multigrid-Verfahren für Finite Elemente

Stichworte: Finite Elemente, Iterative Lösung linearer Gleichungssysteme, Mehrgittermethode

Betreuer: Richter

Termine: WT oder FT

Thema: Bei der Diskretisierung linearer partieller Differentialgleichungen mit der Methode der finiten Elemente (FEM) entstehen – je nach gewählter Diskretisierungsfeinheit – große bis sehr große lineare Gleichungssysteme (LGS). Die Lösung eines LGS für n Unbekannte mit dem Standardverfahren der Gauß-Elimination erfordert im ungünstigsten Fall einen arithmetischen Aufwand von $\mathcal{O}(n^3)$ Operationen, was bei großem n prohibitiv ist. Mit den sogenannten Mehrgitterverfahren (*multigrid methods*) stehen moderne Gleichungslöser zur Verfügung, mit denen es in gewissen Fällen gelingt, diesen Aufwand auf lediglich $\mathcal{O}(n)$ Operationen zu senken. In dieser Praktikumsaufgabe soll eine einfache partielle Differentialgleichung (PDGL) mit finiten Elementen diskretisiert und anschließend mit einem Mehrgitterverfahren (*full multigrid V cycle*) gelöst werden.

Anwendungsfall: Laplace-Gleichung auf dem Einheitsquadrat. Es sei $\Omega := (0,1) \times (0,1)$ das (offene) Einheitsquadrat mit Rand $\partial\Omega$. Es wird das sogenannte Dirichlet-Randwertproblem

$$\begin{aligned} Lu \equiv -u_{xx} - u_{yy} &= f \quad \text{auf } \Omega \\ u &= 0 \quad \text{auf } \partial\Omega \end{aligned} \tag{1}$$

betrachtet. Hier ist u eine auf $\Omega \cup \partial\Omega$ stetige und auf Ω zweimal stetig differenzierbare Funktion, f eine auf Ω stetige Funktion und L ein sogennanter (linearer) Differentialoperator.

Die konkret zu bearbeitenden Aufgaben sind im nachfolgenden Text in roter Farbe markiert.

1. Aufgabe: PDGL in schwacher Formulierung und als Minimierungsproblem. Auf dem Raum $L_2(\Omega)$ aller Funktionen $u : \Omega \to \mathbb{R}$, für die $\int_\Omega u^2(x,y)dxdy <$

∞, wird durch

$$(u, v) := \int_\Omega u(x, y)v(x, y)\, dx\, dy \tag{2}$$

ein Skalarprodukt definiert (genau genommen müssen je zwei Funktionen in $L_2(\Omega)$, die sich nur auf Nullmengen unterscheiden, miteinander identifiziert werden). Es sei ferner $\mathcal{H}$ die Menge aller Funktionen $u \in L_2(\Omega)$,

- deren erste beide partielle Ableitungen (fast überall) existieren und ebenfalls in $L_2(\Omega)$ liegen und
- die auf $\partial\Omega$ gleich null sind.

Ferner sei

$$F(u) := \frac{1}{2}(Lu, u) - (f, u) \quad \text{für alle} \quad u \in \mathcal{H}.$$

Verifizieren Sie unter Benutzung von [1], S. 177 f., die folgenden Aussagen:

- Die Lösung des Dirichlet-Problems (1) erfüllt die Gleichung

  $$(Lu, v) = (f, v) \quad \text{für alle} \quad v \in \mathcal{H} \tag{3}$$

  und umgekehrt.

- Die Lösung des Dirichlet-Problems (1) löst das Minimierungsproblem

  $$u = \text{argmin}_{v\in\mathcal{H}} F(v) \tag{4}$$

  und umgekehrt. Die Schreibweise (4) bedeutet, dass $u \in \mathcal{H}$ eine Funktion mit $F(u) \leq F(v)$ für alle Vergleichsfunktionen $v \in \mathcal{H}$ ist.

Man nennt (3) die **Galerkin-Formulierung** des Problems (1) und (4) die **Rayleigh-Ritz-Formulierung**. Leiten Sie ferner aus (3) unter Benutzung des Divergenzsatzes von Gauß ab, dass für eine Lösung $u \in \mathcal{H}$ von (1) die Identität

$$(\nabla u, \nabla v) = (f, v) \quad \text{für alle} \quad v \in \mathcal{H} \tag{5}$$

gilt, wobei ∇u den Gradienten von u bezeichnet. Die Gleichung (5) ist auch dann noch sinnvoll, wenn man die obige Definition von $\mathcal{H}$ revidiert und nur noch fordert, dass die *ersten* partiellen Ableitungen (fast überall) existieren und in $L_2(\Omega)$ liegen. Wenn man dies tut, dann ist eine Lösung von (5) eine „verallgemeinerte Lösung“, da sie nicht mehr unbedingt (1) lösen muss. Man nennt deswegen (5) die **schwache Formulierung** des Dirichletschen Randwertproblems. Ab sofort wird mit der schwachen Formulierung gearbeitet und die revidierte Definition von $\mathcal{H}$ benutzt. Entsprechend der schwachen Formulierung (5) wird ab sofort auch eine revidierte Definition der Funktion F benutzt, nämlich

$$F(u) := \frac{1}{2}(\nabla u, \nabla u) - (f, u), \quad u \in \mathcal{H}. \tag{6}$$

2. Aufgabe: Diskretisierung mit FEM. Man wählt

$$n \in \mathbb{N},\ n \geq 2 \quad \text{und} \quad h := \frac{1}{n}$$

sowie die Menge der Gitterpunkte

$$\Omega^h := \{(x_i, y_j) := (ih, jh), \quad i, j = 0, \ldots, n\}\,.$$

Jedes Rechteck $[ih, (i+1)h] \times [jh, (j+1)h]$, $i, j = 0, \ldots n-1$ bezeichnet man als ein **Element**, vergleiche Fig. 10.1 in [1]. Mit $\mathcal{H}^h$ wird der Unterraum aller Funktionen aus $\mathcal{H}$ bezeichnet, die auf jedem Element bilinear sind, also von der Form $(ax+b)(cy+d)$. Von Element zu Element können sich die Koeffizienten a, b, c und d ändern, die Funktionen in $\mathcal{H}^h$ sollen aber global stetig sein (keine Sprünge haben), vergleiche Fig. 10.2 in [1]. Funktionen aus $\mathcal{H}^h$ werden dann ebenfalls durch einen Hochindex h gekennzeichnet, man schreibt etwa $u^h \in \mathcal{H}^h$.

Spezielle Mitglieder aus $\mathcal{H}^h$ sind die Funktionen $e^h_{i,j} \in \mathcal{H}^h$, $i, j = 1, \ldots, n-1$, welche im (inneren) Gitterpunkt (x_i, y_j) den Wert 1 und an allen anderen Gitterpunkten den Wert 0 annehmen, vergleiche Fig 10.3 in [1]. Die Menge $\{e^h_{i,j};\ i, j = 1, \ldots, n-1\}$ heißt **nodale Basis** von $\mathcal{H}^h$. Jede Funktion $u^h \in \mathcal{H}^h$ besitzt eine eindeutige Darstellung

$$u^h(x, y) = \sum_{i,j=1}^{n-1} u^h_{i,j} e^h_{i,j}(x, y), \quad u^h_{i,j} = u^h(x_i, y_j) \tag{7}$$

bezüglich der nodalen Basis.

Eine diskretisierte Lösung $u^h \in \mathcal{H}^h$ von (5) berechnet sich nun aus dem LGS

$$(\nabla u^h, \nabla v^h) = (f, v^h) \quad \text{für alle} \quad v^h \in \mathcal{H}^h. \tag{8}$$

Da in (8) sowohl u^h als auch die Funktionen v^h in der Form (7) angesetzt werden können, ist (8) äquivalent zu den $(n-1)^2$ Gleichungen

$$\sum_{i,j=1}^{n-1} u^h_{i,j} (\nabla e^h_{i,j}, \nabla e^h_{k,\ell}) = (f, e^h_{k,\ell}), \quad k, \ell = 1, \ldots n-1 \tag{9}$$

für die $(n-1)^2$ Unbekannten $u^h_{i,j}$ aus (7).

Bestätigen Sie folgende Ergebnisse (vergleiche [1], S. 182): Es ist

$$\begin{aligned} A^h_{k,\ell} &:= \begin{pmatrix} (\nabla e^h_{k-1,\ell+1}, \nabla e^h_{k,\ell}) & (\nabla e^h_{k,\ell+1}, \nabla e^h_{k,\ell}) & (\nabla e^h_{k+1,\ell+1}, \nabla e^h_{k,\ell}) \\ (\nabla e^h_{k-1,\ell}, \nabla e^h_{k,\ell}) & (\nabla e^h_{k,\ell}, \nabla e^h_{k,\ell}) & (\nabla e^h_{k+1,\ell}, \nabla e^h_{k,\ell}) \\ (\nabla e^h_{k-1,\ell-1}, \nabla e^h_{k,\ell}) & (\nabla e^h_{k,\ell-1}, \nabla e^h_{k,\ell}) & (\nabla e^h_{k+1,\ell-1}, \nabla e^h_{k,\ell}) \end{pmatrix} \\ &= \frac{1}{3} \begin{pmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{pmatrix}, \end{aligned} \tag{10}$$

und alle Integrale $(\nabla e^h_{i,j}, \nabla e^h_{k,\ell})$ mit $|i-k|>1$ oder $|j-\ell|>1$ sind gleich null. Approximiert man zur Berechnung von

$$(f, e^h_{k,\ell}) = \int_{x_{k-1}}^{x_{k+1}} \int_{y_{\ell-1}}^{y_{\ell+1}} f(x,y) e^h_{k,\ell}(x,y)\, dy\, dx$$

die Funktion f durch den konstanten Wert $f(x_k, y_\ell)$ so erhält man

$$(f, e^h_{k,\ell}) \approx h^2 f(x_k, y_\ell) =: f^h_{k,\ell}. \tag{11}$$

Mit den Abkürzungen

$$\boldsymbol{u}^h = (u^h_{i,j})_{\substack{i=1,\dots,n-1\\ j=1,\dots,n-1}} \in \mathbb{R}^{n-1,n-1}, \quad \boldsymbol{f}^h = (f^h_{i,j})_{\substack{i=1,\dots,n-1\\ j=1,\dots,n-1}} \in \mathbb{R}^{n-1,n-1}.$$

lässt sich das LGS (9) symbolisch in der Kurzform

$$A^h \boldsymbol{u}^h = \boldsymbol{f}^h \tag{12}$$

schreiben. A^h ist ein Operator, dessen Wirkung auf $\boldsymbol{u}^h$ im Gitterpunkt (x_k, y_ℓ) in Pseudo-MATLAB-Notation mit dem „punktweisen Operator“ `.*` in der Form

$$\texttt{sum}(A^h_{k,\ell} \texttt{ .* } \boldsymbol{u}^h(k-1:k+1, \ell-1:\ell+1), \texttt{'all'}) \tag{13}$$

beschrieben werden kann, wobei die Randwerte 0 von $\boldsymbol{u}^h$ genutzt werden.[1] Für die folgenden Teilaufgaben ist es *nicht nötig*, $\boldsymbol{u}^h$ und $\boldsymbol{f}^h$ als Spaltenvektoren und A^h als dazu passende Matrix explizit anzugeben (was eine einfach indizierte Reihung der Doppelindizes (i,j) erforderlich machen würde; würde man dies tun, dann erhielte man eine symmetrische, positiv definite Matrix A^h).

Äquivalent zu (12) ist das Minimierungsproblem

$$\boldsymbol{u}^h = \text{argmin}_{\boldsymbol{v}^h} F^h(\boldsymbol{v}^h) \tag{14}$$

mit der Funktion

$$F^h(\boldsymbol{v}^h) = \frac{1}{2}(A^h \boldsymbol{v}^h, \boldsymbol{v}^h) - (\boldsymbol{f}^h, \boldsymbol{v}^h), \tag{15}$$

wobei hier mit $(\cdot,\cdot)$ das Euklidische Skalarprodukt gemeint ist (wenn man die Matrizen $A^h\boldsymbol{v}^h$, $\boldsymbol{v}^h$ und $\boldsymbol{f}^h$ als Spaltenvektoren angeordnet). Die Funktion F^h aus (14) ist auf dem Raum der nodalen Werte $\boldsymbol{u}^h = (u^h_{i,j})$ definiert, während F in (4) auf dem Raum von Funktionen $\mathcal{H}$ definiert ist.

3. Aufgabe: Lösung des LGS mit dem Gauß-Seidel-Verfahren (GS). Das GS-Verfahren zur Lösung eines allgemeinen LGS $Ax = b$ mit $A \in \mathbb{R}^{n,n}$ und $b \in \mathbb{R}^n$ lautet:

[1] Man beachte die MATLAB-Konvention der Indizierung ab 1, nicht ab 0.

für m:=$1, 2, \ldots$

$$\text{für i:=1 bis } n\text{: } x_i := x_i + \left(b_i - \sum_j a_{i,j} x_j \right) / a_{i,i}$$

Das Verfahren bricht zusammen, wenn die Matrix A ein Nullelement auf der Diagonale hat, was im Fall der positiv definiten Matrix $A = A^h$ nicht passieren kann. Jeden Durchlauf durch die äußere Schleife bezeichnet man als einen **sweep**. In einem sweep werden sukzessive die Komponenten von x geändert, wobei im i-ten Schritt die Komponente x_i so geändert wird, dass gerade die i-te Gleichung erfüllt ist. Im nächsten Schritt wird x_{i+1} geändert und dann ist die i-te Gleichung bereits nicht mehr erfüllt. Deswegen wiederholt man die sweeps ($m = 1, 2, \ldots$) und hofft so, der Lösung immer näher zu kommen. Eine allgemeine Konvergenzanalyse des GS-Verfahrens findet man in vielen Lehrbüchern, zum Beispiel in *Matrix Iterative Analysis* von Richard Vargas. In der Praxis muss die Schleife über m mit einem geeigneten Kriterium abgebrochen werden.

Die Anwendung des GS-Verfahrens erfordert keine explizite Abspeicherung der Matrix A. Implementieren Sie eine MATLAB-Funktion

```
u = function GS(n, f, u, it)
```

welche einen Startwert `u` für die gesuchte Lösung $\boldsymbol{u}^h$ von (12) mit rechter Seite `f` $= \boldsymbol{f}^h$ in `it` sweeps des GS-Verfahrens zu einer neuen Näherung `u` (Ausgabewert der Funktion) verbessert. Die Implementierung sollte auf (13) beruhen, also *nicht* auf einer expliziten Abspeicherung der vollständigen Matrix A^h. Testen Sie das Verfahren an einem Beispiel mit analytisch berechenbarer Lösung, etwa mit

$$f(x, y) = 10\pi^2 \sin(3\pi x) \sin(\pi y) \quad \Longrightarrow \quad u(x, y) = \sin(3\pi x) \sin(\pi y).$$

Modifizieren Sie die rechte Seite `f`$:= A^h \boldsymbol{u}^h$, so dass der Fehler der berechneten Lösung genau dem Fehler des GS-Verfahrens entspricht und nicht zusätzlich den Diskretisierungsfehler beinhaltet. Sie werden feststellen, dass für befriedigend genaue Näherungslösungen sehr viele Iterationsschritte (sweeps) ausgeführt werden müssen. Machen Sie nun noch folgendes Experiment: Starten Sie das GS-Verfahren für die rechte Seite $f(x, y) = 0$ (dann ist die exakte Lösung des Dirichlet-Randwertproblems $u(x, y) = 0$) mit dem Startwert

```
u=randn(n+1)
```

Nach m sweeps erhält man eine diskrete Näherungslösung $\boldsymbol{u}^m$ für die exakte Lösung $\boldsymbol{u}^h = 0$ und $-\boldsymbol{u}^m$ ist der Fehler dieser Näherungslösung. Plotten Sie den Startfehler und den Fehler nach jeder der ersten 5 sweeps (MATLAB-Befehl `surf`) und bestätigen Sie (durch Inspektion der Plots): Die Größe des Fehlers verringert sich kaum, der Fehler ändert jedoch sehr schnell seine Struktur, da die hochfrequenten Fehleranteil verschwinden. Dies ist die sogenannte **Glättungseigenschaft** des GS-Verfahrens (in

der Anwendung auf die diskretisierte PDGL).

Die Grundidee des Mehrgitterverfahrens. Man betrachte allgemein ein LGS (mit invertierbarer quadratischer Matrix A)

$$Ax = b$$

und eine (mit ein paar sweeps des GS-Verfahrens gewonnene) Näherungslösung

$$\tilde{x} \quad \text{mit Fehler} \quad e = x - \tilde{x}.$$

Formal erhält man die exakte Lösung aus

$$x = \tilde{x} + e \quad \text{mit} \quad Ae = r := b - A\tilde{x},$$

der Vektor r heißt das **Residuum** der Näherung $\tilde{x}$. Der Fehler e, der bei Unkenntnis der exakten Lösung x nicht direkt berechenbar ist, lässt sich also als Lösung eines LGS mit gleicher Matrix A wie das originale LGS und mit einer bekannten rechten Seite r berechnen. Die Lösung des LGS $Ae = r$ erfordert im Allgemeinen allerdings denselben Aufwand wie die Lösung von $Ax = b$. Wenn nun aber, wie im oben betrachteten Fall, das LGS aus der Diskretisierung einer PDGL stammt und wenn bekannt ist, dass e keine hochfrequenten Anteile hat, sondern glatt ist, so kann man Folgendes versuchen: Man löse nicht $Ae = r$, sondern $\hat{A}\hat{e} = \hat{r}$, wobei $\hat{A}$ einer Diskretisierung der PDGL auf einem gröberen Gitter entspricht (und damit von kleinerer Dimension als A ist) und $\hat{e}$ einer Restriktion von e auf dieses gröbere Gitter entspricht. Anschließend interpoliere man die Grobgitternäherung $\hat{e}$ auf das feinere Gitter und erhält so eine (gute) Näherung für (das glatte) e. Die folgenden Aufgaben beleuchten, wie Restriktion vom feinen auf das grobe Gitter) und Interpolation (vom groben auf das feine Gitter) und wie ein Übergang vom Operator A (auf dem Feingitter) zum Operator $\hat{A}$ (auf dem Grobgitter) aussehen. Ab jetzt sei

$$n = 2^N. \tag{16}$$

Das Gitter Ω^{2h} (grob) entsteht aus Ω^h (fein), indem man in jeder Dimension nur die Punkte mit geradzahligem Index auswählt:

$$\Omega^{2h} = \{(2ih, 2jh);\ i = 0, \ldots, n/2,\ j = 0, \ldots, n/2\}\,.$$

Genauso entsteht Ω^{4h} (noch gröber) aus Ω^{2h}, indem man *dessen* geradzahlig indizierte Gitterpunkte auswählt:

$$\Omega^{4h} = \{(4ih, 4jh);\ i = 0, \ldots, n/4,\ j = 0, \ldots, n/4\}$$

und so weiter.

4. Aufgabe: GS als Optimierer. Die nodalen Werte $\boldsymbol{u}^h = (u^h_{i,j})$ definieren eine Funktion $u^h \in \mathcal{H}^h$ gemäß (7). Für einen festen Index (i,j) sei

$$s = \text{argmin}_{t\in\mathbb{R}} F(u^h - te^h_{i,j}). \tag{17}$$

Bestätigen Sie: Der Übergang $u^h \longrightarrow u^h - se^h_{i,j}$ entspricht genau dem Update des nodalen Werts $u^h_{i,j}$ von u^h beim GS-Verfahren (wenn man den Term $(f, e^h_{i,j,k})$ in der Definition (6) von F durch $f^h_{i,j,k}$ ersetzt).

5. Aufgabe: Gitterwechsel. Es sei $v^{2h} \in \mathcal{H}^{2h}$. Diese Funktion ist stückweise bilinear zu den Elementen des groben Gitters Ω^{2h}, es existiert also eine Darstellung

$$v^{2h}(x,y) = \sum_{i,j=1}^{\frac{n}{2}-1} v^{2h}_{i,j} e^{2h}_{i,j}(x,y).$$

Wegen $\mathcal{H}^{2h} \subseteq \mathcal{H}^h$ muss auch eine Darstellung

$$v^{2h}(x,y) = \sum_{i,j=1}^{n-1} v^h_{i,j} e^h_{i,j}(x,y)$$

existieren. Verifizieren Sie folgende Berechnungsformel für die nodalen Werte der Feingitterdarstellung:

$$\begin{aligned}
v^h_{2i,2j} &= v^{2h}_{i,j} \\
v^h_{2i+1,2j} &= \frac{1}{2}\left(v^{2h}_{i,j} + v^{2h}_{i+1,j}\right) \\
v^h_{2i,2j+1} &= \frac{1}{2}\left(v^{2h}_{i,j} + v^{2h}_{i,j+1}\right) \\
v^h_{2i+1,2j+1} &= \frac{1}{4}\left(v^{2h}_{i,j} + v^{2h}_{i+1,j} + v^{2h}_{i,j+1} + v^{2h}_{i+1,j+1}\right),
\end{aligned}$$

wobei $i,j = 0,\ldots,\frac{n}{2}-1$ und die verschwindenden Randwerte sowohl von $\boldsymbol{v}^h$ als auch von $\boldsymbol{v}^{2h}$ zu berücksichtigen sind. Gemäß diesen Formeln lässt sich ein **Interpolationsoperator**

$$I_{2h\to h} : \mathbb{R}^{\frac{n}{2}-1,\frac{n}{2}-1} \to \mathbb{R}^{n-1,n-1}, \quad \boldsymbol{v}^{2h} \mapsto I_{2h\to h}\boldsymbol{v}^{2h} = \boldsymbol{v}^h \tag{18}$$

für die nodalen Werte von v^h (an den inneren Gitterpunkten) definieren. Der Operator $I_{2h\to h}$ wird manchmal auch als **Prolongationsoperator** bezeichnet. Würde man $\boldsymbol{v}^{2h}$ und $\boldsymbol{v}^h$ als Spaltenvektoren schreiben, dann ließe sich $I_{2h\to h}$ als eine Matrix mit $(n-1)^2$ Zeilen und $(n/2-1)^2$ Spalten explizit angeben. Das Transponieren dieser Matrix führt auf einen Operator

$$(I_{2h\to h})^\top =: I_{h\to 2h} : \mathbb{R}^{n-1,n-1} \to \mathbb{R}^{\frac{n}{2}-1,\frac{n}{2}-1}, \quad \boldsymbol{v}^h \mapsto I_{h\to 2h}\boldsymbol{v}^h = \boldsymbol{v}^{2h}, \tag{19}$$

welcher nodale Werte zum Feingitter Ω^h auf solche des Grobgitters Ω^{2h} überführt. Dieser Operator wird als **Restriktionsoperator** bezeichnet. Er ist explizit durch

$$\begin{aligned} v_{i,j}^{2h} &= \frac{1}{4}\left(v_{2i-1,2j-1}^h + v_{2i-1,2j+1}^h + v_{2i+1,2j-1}^h + v_{2i+1,2j+1}^h\right) \\ &\quad + \frac{1}{2}\left(v_{2i,2j-1}^h + v_{2i,2j+1}^h + v_{2i-1,2j}^h + v_{2i+1,2j}^h\right) + v_{2i,2j}^h \end{aligned}$$

gegeben (bei Interesse nachrechnen). Bemerkung: Die (nodalen Werte der) Funktion $v^h \equiv 1$ des feinen Gitters werden durch $I_{h\to 2h}$ auf die (nodalen Werte der) Funktion $v^{2h} \equiv 4$ des groben Gitters überführt. Um dies zu vermeiden, müsste man $I_{h\to 2h} = \frac{1}{4}(I_{2h\to h})^\top$ wählen. Wegen der nachfolgenden Formel (20) wird hier aber an $I_{h\to 2h} := (I_{2h\to h})^\top$ festgehalten.

Die Operatoren $I_{2h\to h}$ und $I_{h\to 2h}$ transferieren nodale Werte und damit stückweise bilineare Funktionen vom groben auf das feine Gitter und umgekehrt. Es bleibt noch zu klären, wie die Grobgitterversion des Operators A^h aussehen soll. Orientiert am Minimierungsproblem (14) soll $\boldsymbol{v}^{2h}$ so bestimmt werden, dass

$$\boldsymbol{v}^{2h} = \text{argmin}_{\boldsymbol{w}^{2h}} F^h(\boldsymbol{v}^h + I_{2h\to h}\boldsymbol{w}^{2h}).$$

Ist $\boldsymbol{v}^h$ eine Näherungslösung des LGS $A^h\boldsymbol{u}^h = \boldsymbol{f}^h$, so bedeutet dies, dass $\boldsymbol{v}^{2h}$ als optimale Grobgitterkorrektur gewählt werden soll, um $\boldsymbol{v}^h$ zu verbessern. Dies entspricht gerade der Grundidee der Mehrgitteridee, eine optimale Grobgitterkorrektur $\hat{e}$ der Näherungslösung $\tilde{x}$ von $Ax = b$ zu finden.

Verifizieren Sie das in [1], S. 185, berechnete Resultat, dass für alle $\boldsymbol{v}^h \in \mathcal{H}^h$ und alle $\boldsymbol{w}^{2h} \in \mathcal{H}^{2h}$ und die in (15) definierte Funktion F^h gilt:

$$F^h(\boldsymbol{v}^h + I_{2h\to h}\boldsymbol{w}^{2h}) = \tag{20}$$

$$F^h(\boldsymbol{v}^h) + \frac{1}{2}\left(\underbrace{(I_{2h\to h})^\top A^h I_{2h\to h}}_{=:\,A^{2h}}\boldsymbol{w}^{2h}, \boldsymbol{w}^{2h}\right) - \left(\underbrace{(I_{2h\to h})^\top}_{=:\,I_{h\to 2h}}\underbrace{\left(\boldsymbol{f}^h - A^h\boldsymbol{v}^h\right)}_{=:\,\boldsymbol{r}^h}, \boldsymbol{w}^h\right)$$

Hier ist $\boldsymbol{r}^h$ das Residuum der Näherungslösung $\boldsymbol{v}^h$ des LGS $A^h\boldsymbol{u}^h = \boldsymbol{f}^h$. Der Operator A^{2h} ist die Entsprechung von A^h auf dem groben Gitter. Man kann explizit nachrechnen, dass

$$A_{k,\ell}^{2h} = \frac{1}{3}\begin{pmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{pmatrix},$$

das heißt der Operator A^{2h} wirkt exakt wie A^h, wird jedoch auf die Punkte Ω^{2h} des groben Gitters angewendet. Mit

$$F^{2h}(\boldsymbol{w}^{2h}) = \frac{1}{2}\left(A^{2h}\boldsymbol{w}^{2h}, \boldsymbol{w}^{2h}\right) - \left(I_{h\to 2h}\boldsymbol{r}^h, \boldsymbol{w}^{2h}\right)$$

zeigt (20), dass

$$\boldsymbol{v}^{2h} = \text{argmin}_{\boldsymbol{w}^{2h}} F^h(\boldsymbol{v}^h + I_{2h\to h}\boldsymbol{w}^{2h}) = \text{argmin}_{\boldsymbol{w}^{2h}} F^{2h}(\boldsymbol{w}^{2h}).$$

Die Folgerung hiervon ist, dass $\boldsymbol{v}^{2h}$ in Analogie zu (12) als Lösung des LGS

$$A^{2h}\boldsymbol{w}^{2h} = I_{h\to 2h}\boldsymbol{r}^h, \quad \boldsymbol{r}^h = \boldsymbol{f}^h - A^h\boldsymbol{v}^h, \tag{21}$$

zu berechnen ist.

Grundsätzlich kann die Grundidee der Mehrgittermethode nun folgendermaßen umgesetzt werden:

1. Man wähle eine Näherungslösung $\boldsymbol{v}^h$ von (12).

2. Mit dem Startwert $\boldsymbol{v}^h$ führe man einige Iterationsschritte des GS-Verfahrens zum LGS (12) aus. Das so verbesserte Resultat heiße erneut $\boldsymbol{v}^h$.

3. Man berechne das Residuum $\boldsymbol{r}^h = \boldsymbol{f}^h - A^h\boldsymbol{v}^h$ und löse das LGS (21) mit Ergebnis $\boldsymbol{w}^{2h}$.

4. Man korrigiere die Näherungslösung $\boldsymbol{v}^h$ zu $\boldsymbol{u}^h = \boldsymbol{v}^h + I_{2h\to h}\boldsymbol{w}^{2h}$.

5. Abschließend werden noch einmal einige Iterationsschritte des GS-Verfahrens zum LGS (12) ausgeführt, diesmal mit Startwert $\boldsymbol{u}^h$.

Unklar ist hier noch der erste Schritt: Wie kommt man zu einer ersten Näherungslösung ? Naheliegend ist es, das LGS (12) auf Ω^{2h} zu lösen und die so erhaltene Lösung auf das Gitter Ω^h zu interpolieren.

Da das Gitter Ω^{2h} immer noch sehr fein sein kann, ist es weiterhin naheliegend, sowohl die Beschaffung der Anfangslösung als auch die Lösung des LGS (21) rekursiv auf immer noch gröbere Gitter zurückzuspielen. Dazu wird von der Voraussetzung (16) ausgegangen, es werden also insgesamt N Gitter der Stufen $i = 1, 2, \ldots, N$ mit Diskretisierungsfeinheiten $2^{-i} = 2^{-1}, 2^{-2}, \ldots, 2^{-N}$ betrachtet. Die Schritte 2 – 5 von oben lassen sich als eine rekursive Funktion MGV in einer Pseudo-Programmiersprache beschreiben – nähere Erkläuterungen dazu im Anschluss.

$u^{(i)} =$ **function MGV** $(i, u^{(i)}, f^{(i)}, gs1, gs2)$

falls $i = 1$:

löse $A^{(1)}u^{(1)} = f^{(1)}$ exakt

return $u^{(1)}$

sonst:

$$\tilde{u}^{(i)} = GS(2^i, f^{(i)}, u^{(i)}, gs1) \tag{22}$$

$$r^{(i)} = f^{(i)} - A^{(i)}\tilde{u}^{(i)} \tag{23}$$

$$d^{(i)} = In(MGV(i-1, 0, R(r^{(i)}), gs1, gs2)) \tag{24}$$

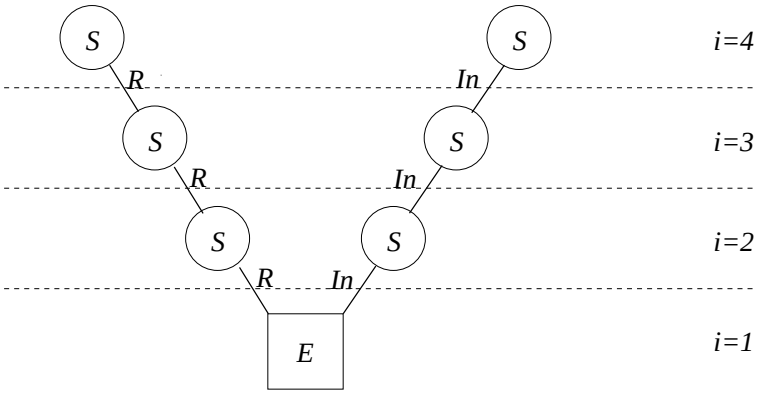
$$\hat{u}^{(i)} = \tilde{u}^{(i)} + d^{(i)} \tag{25}$$

$$u^{(i)} = GS(2^i, f^{(i)}, \hat{u}^{(i)}, gs2) \tag{26}$$

return $u^{(i)}$

Bemerkungen. MGV ist eine rekursive Funktion, i bezeichnet das aktuelle Gitterlevel. Der Operator A^h zum Gitter Ω^h wird mit $A^{(i)}$ bezeichnet, wenn $h = 2^{-i}$. Ebenso werden dann $\boldsymbol{u}^h$ mit $u^{(i)}$ und $\boldsymbol{f}^h$ mit $f^{(i)}$ bezeichnet. GS ist die Funktion `GS` aus der dritten Aufgabe. Mit In wird der Interpolationsoperator $I_{2h\to h}$ für die aktuelle Gitterstufe gemäß (18) bezeichnet und mit R wird der entsprechende Restriktionsoperator $I_{h\to 2h}$ bezeichnet. Die Zahlen $gs1$ und $gs2$ bezeichnen die Anzahl von sweeps des GS-Verfahrens in (22) bzw. (26).

Die folgende Graphik illustriert den Gang der Berechnung von MGV bei insgesamt 4 Gitterstufen. „S“ entspricht dem GS-Verfahren in (22) bzw. (26), „E “bedeutet exaktes Lösen des LGS $A^{(1)}x = f^{(1)}$, die Übergänge „R“ bedeuten Restriktionen, die Übergänge „In“ bedeuten Interpolation.



Zurück zur Frage, wie man zu einem guten Startwert $u^{(i)}$ für den Aufruf von MGV kommt. Wie schon angedeutet, lässt sich eine solche wie folgt beschaffen. Man interpoliere zunächst eine auf dem gröbsten Gitter berechnete Lösung auf das zweitgröbste Gitter. Dann starte man MGV mit $i = 2$ zur Verbesserung der Lösung auf dem zweitgröbsten Gitter. Diese Lösung wird auf das drittgröbste Gitter interpoliert und ein Dreigitter-Verfahren (also MGV mit $i = 3$) zur Verbesserung der Lösung gestartet und so weiter. Diese Vorgehensweise heißt **Full Multigrid Algorithm.** Der Algorithmus kann wieder als Funktion in einer Pseudo-Programmiersprache beschrieben werden.

$z =$ **function FMG** $(i, u^{(i)}, f^{(i)}, gs1, gs2)$

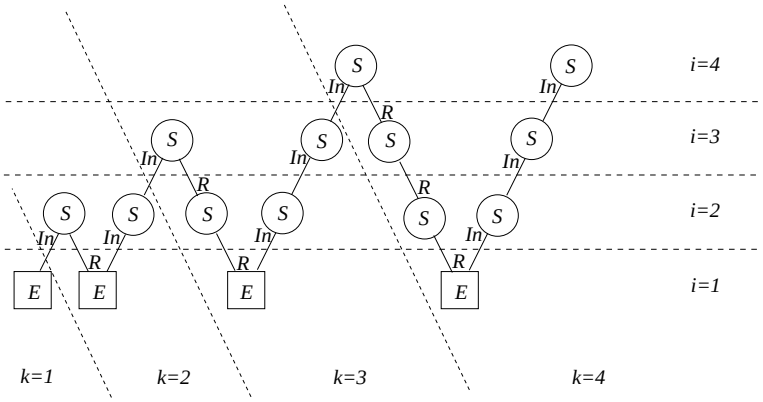
löse $A^{(1)}u^{(1)} = f^{(1)}$ exakt

für $k = 2, \ldots, i$

$u^{(k)} = MGV(k, In(u^{(k-1)}), f^{(k)}, gs1, gs2)$

FMG wird mit dem Parameter $i = N$ aufgerufen. Beim ersten Aufruf von FMG setzt man $u^{(N)} = 0$ (Nullvektor als Startnäherung, da man nichts Besseres weiß) und $f^{(N)} = \boldsymbol{f}^h$, die originale rechte Seite des feinsten Gitters. Dann sind $f^{(1)}, \ldots, f^{(N-1)}$ deren Restriktionen auf die entsprechenden Gitter $\Omega^{2^{-i}}$. Man schreibt dazu eine Funktion $\tilde{r} = mgvrhs(n, m, r)$, welche die Restriktion von r (gegeben auf dem Gitter der Stufe i mit $2^i = n$) auf $\tilde{r}$ (auf dem Gitter der Stufe j mit $2^j = m$) berechnet, $n > m$.

Die folgende Skizze zeigt den Gang der Berechnung, wenn FMG mit $i = 4$ aufgerufen wird.



Nach dem ersten Aufruf von FMG ist eine Näherungslösung $\tilde{z} := z$ von (12) berechnet. Diese kann verbessert werden, indem man ihr Residuum $r^{(N)} = f^{(N)} - A^{(N)}\tilde{z}$ berechnet, dann FMG mit Startwert $u^{(N)} = \tilde{z}$ und rechter Seite $f^{(N)} = r^{(N)}$ aufruft und mit dem Rückgabewert z von FMG die Korrektur $\hat{z} = \tilde{z} + z$ ausführt.

6. Aufgabe. Programmieren Sie die Funktionen MGV, FMG und $mgvrhs$. Testen Sie Ihre Implementierung mit dem zur Verfügung gestellten Testprogramm `testfmg`.

Templates für die Funktionen MGV, FMG und $mgvrhs$ werden zur Verfügung gestellt.

7. Aufgabe. Im Programm `testfmg` wird eine Ausgabegröße `costpi` berechnet. Überlegen Sie, warum diese Größe tatsächlich, wie behauptet, ein Maß für die „Anzahl Rechenoperationen pro Unbekannter und Iteration“ ist. Beobachten Sie den Wert von `costpi` beim Start von `testfmgv` mit verschiedenen Diskretisierungsfeinheiten. Was folgt daraus für die Behauptung, dass das Mehrgitterverfahren das lineare Gleichungssystem (12) mit einem arithmetischen Aufwand löst, der proportional zur Anzahl der Unbekannten ist?

Literatur:

[1] William L. Briggs, Van Emden Hanson, Steve F. McCormick, *A Multigrid Tutorial*, SIAM, 2nd ed., 2000.