Aufgabe 11 Jacobi-Verfahren als additives Schwarz-Verfahren

In der Vorlesung haben Sie die gedämpfte additive Schwarz-Iteration

$$x^{(k+1)} = x^{(k)} + \omega \sum_{i=0}^{p} R_i^T A_i^{-1} R_i (b - Ax^{(k)}) \tag{1}$$

mit

$$A_i = R_i A R_i^T \tag{2}$$

kennengelernt.

a) Geben Sie an, wie der Dämpfungsfaktor ω, die Teilgebietsanzahl p und die Restriktionsmatrizen R_i zu wählen sind, damit (1) das Jacobi-Verfahren beschreibt.

b) Sei nun A die Steifigkeitsmatrix einer Finite-Elemente-Diskretisierung mit dem Finite-Elemente-Raum V^h und der Basis φ_i^h. Das Finite-Elemente-Gitter habe die Dimension d und die Gitterweite h. Für jedes $u^h \in V^h$ haben wir eine eindeutige Darstellung

$$u^h = \sum_{i=1}^{N} x_i \varphi_i^h, \quad x_i \in \mathbb{R}$$

und es gilt die Abschätzung

$$\|x\|_2 \leq C h^{-\frac{d}{2}} \left\|u^h\right\|_{L^2(\Omega),}$$

wobei $\|\cdot\|_2$ die euklidische Norm auf $\mathbb{R}^N$ bezeichnet und C eine von h unabhängige Konstante ist. Um die abstrakte Schwarz-Theorie auf das Jacobi-Verfahren anwenden zu können, müssen die folgenden beiden Voraussetzungen erfüllt sein:

Voraussetzung A1 (Stabile Zerlegung) *Es gibt eine solche Konstante C_0, dass zu jedem $x \in \mathbb{R}^N$ eine Zerlegung $x = \sum_{i=0}^{p} R_i^T x_i$ existiert, für die gilt*

$$\sum_{i=0}^{p} \langle R_i^T x_i, R_i^T x_i \rangle_A \leq C_0 \langle x, x \rangle_A.$$

Voraussetzung A2 (Verschärfte Cauchy-Schwarz-Ungleichung) *Es gibt Konstanten $0 \leq \varepsilon_{ij} \leq 1$, $1 \leq i, j \leq p$, so dass für alle x_i und x_j gilt*

$$\left|\langle R_i^T x_i, R_j^T x_j \rangle_A\right| \leq \varepsilon_{ij} \langle R_i^T x_i, R_i^T x_i \rangle_A^{\frac{1}{2}} \langle R_j^T x_j, R_j^T x_j \rangle_A^{\frac{1}{2}}.$$

Zeigen Sie, dass diese Voraussetzungen für das Jacobi-Verfahren erfüllt sind und geben Sie die bestmögliche Wahl der Konstanten C_0 und ε_{ij} an.

14 Punkte

Aufgabe 12 Reduktion der sequentiellen Komplexität überlappender Schwarz-Verfahren

Wir lösen eine partielle Differentialgleichung auf einem Gitter mit $N = n^d$ Gitterzellen. Dabei sei d die Dimension des Gitters. Der Aufwand für eine Iteration unsere Lösungsverfahrens betrage n^α mit einem $\alpha \geq d$. Nun möchten wir den sequentiellen Rechenaufwand verringern, indem wir ein überlappendes Schwarz-Verfahren verwenden. Dazu zerlegen wir das Gebiet in $p = n_H^d$ überlappende Teilgebiete (H bezeichnet die Grobgitterweite) mit der Überlappung β, in dem Sinne, dass die Seitenlänge der Teilgebiete $\frac{n}{n_H}(1+\beta)$ Zellen betrage. Als Teilgebietslöser und für die Grobgitterkorrektur verwenden wir wieder das oben erwähnte Iterationsverfahren.

Geben Sie (a) den Aufwand für eine Iteration des überlappenden Schwarz-Verfahrens und (b) die Wahl von n_H an, für die der Aufwand asymptotisch minimal wird. *6 Punkte*

Aufgabe 13 Paralleles überlappendes Schwarz-Verfahren

In dieser Aufgabe experimentieren wir erstmals praktisch mit einer Implementierung eines parallelen Lösers.

Zunächst sollten Sie `dune-parsolve` und `dune-pdelab` mittels `svn update` auf den neuesten Stand bringen. Falls Sie auf Ihrem eigenen Rechner arbeiten, stellen Sie als nächstes sicher, dass Sie über eine Installation der Bibliothek SuperLU zum direkten Lösen linearer Gleichungssysteme verfügen. In den meisten Linux-Distribution gibt es dafür ein Paket, ansonsten werden Sie leicht im WWW fündig. Nach der Installation von SuperLU sollten Sie in Ihrer DUNE-Optionendatei zu den `CONFIGURE_OPTS` den Parameter `--with-superlu=<Installationspfad>` hinzufügen. (Wenn Sie SuperLU als Paket oder unter `/usr` installiert haben, ist dieser Parameter normalerweise nicht erforderlich.) Danach kompilieren Sie am besten DUNE komplett neu.

Führen Sie nun einige Testrechnungen mit dem Programm `parallel_cg_with_overlap` aus `dune-parsolve` durch. Wie in der Übung gezeigt, können Sie im Code zwischen den Teilgebietslösern `exactsubdomainprec` und `inexactsubdomainprec` wählen. Ersterer basiert dabei auf SuperLU, letzterer auf dem CG-Verfahren. Wenn Sie den `inexactsubdomainprec` verwenden, sollte die Deklaration von `exactsubdomainprec` aus dem Code auskommentiert werden, da bereits im Konstruktoraufruf die LU-Zerlegung der Matrix berechnet wird. Im Hauptprogramm können Sie die Gittergröße und damit die Anzahl der Unbekannten variieren.

Geben Sie für folgende Kombinationen jeweils die Rechenzeiten für den Matrixaufbau, die Jacobi-Matrix-Berechnung und das Lösen sowie die Anzahl der benötigten Iterationen an:

Nr.	Löser	Gittergröße	Anzahl der Prozesse
1	SuperLU	512×512	1
2	CG	512×512	1
3	SuperLU	512×512	4
4	CG	512×512	4
5	SuperLU	512×512	16
6	CG	512×512	16
7	SuperLU	1024×1024	4
8	CG	1024×1024	4
9	SuperLU	1024×1024	64
10	CG	1024×1024	64

Um `parallel_cg_with_overlap` auf mehreren Rechnern im CIP-Pool zu starten, sollten Sie als erstes eine Datei mit den Namen der Rechner erzeugen, die an der Rechnung beteiligt sein sollen. Dazu können Sie im Verzeichnis `dune-parsolve/src` das Skript

```
./create_mpihosts.sh
```

aufrufen, dass in die Datei `mpihosts` die Namen derjenigen Rechner schreibt, die im Augenblick nicht ausgelastet sind. Danach können Sie die parallele Rechnung mittels

```
mpirun -np <p> -machinefile mpihosts ./parallel_cg_with_overlap
```

starten. Für `<p>` setzen Sie die Anzahl der gewünschten Prozesse ein. Achten Sie dabei bitte auf folgende Punkte:

- Rechnen Sie nach Möglichkeit nicht tagsüber, weil dann im Pool Übungen stattfinden. Am Wochenende oder am Abend stören die Rechnungen niemanden.
- Rufen Sie vor jedem parallelen Prozessstart das Skript `create_mpihosts.sh` auf, damit die aktuell nicht ausgelasteten Rechner neu ermittelt werden.
- Achten Sie auf die Anzahl der Rechner, die in Ihrer `mpihosts`-Datei eingetragen wurden. Jeder Rechner verfügt über zwei Prozessorkerne, so dass Sie mit zwei Prozessen pro Rechner optimale Leistung erreichen sollten. Insbesondere die Rechnung mit 64 Prozessen sollten Sie nicht ausführen, wenn nicht mindestens 32 Rechner zur Verfügung stehen.
- Überschlagen Sie vor der Rechnung den Speicherverbrauch. Auf jedem Rechner sollten Sie höchstens 1.5 GB Speicher verbrauchen, damit die Rechner stabil weiterlaufen. Bei Verwendung von CG beträgt der Speicherverbrauch von 400 Byte pro Unbekannter, bei der Verwendung des direkten Lösers SuperLU ungefähr 2.200 Byte. Berücksichtigen Sie auch, wieviele Prozesse auf einem Rechner entstehen.

Sollten Sie einmal eine parallele Rechnung abbrechen wollen, drücken Sie einfach Strg-C. Dabei kann es passieren, dass auf den Knoten einzelne Prozesse hängebleiben. Diese können Sie mit der Kommandozeile

```
for i in ‘seq -w 50‘; do ssh cip$i killall -9 parallel_cg_with_overlap; done
```

entfernen. *10 Punkte*