CG-SEM für die Poissongleichung mit Präkonditionierung

Doktorandenseminar WS 13/14

Serena Keller

Institut für Aerodynamik und Gasdynamik, Universität Stuttgart

5. November 2013

Gliederung

Poissongleichung

- Betrachte die Poissongleichung in $\Omega \subset \mathbb{R}^2$ mit Dirichlet-Randbedingungen:

$$\begin{aligned} \Delta u = u_{xx} + u_{yy} &= s && \text{in } \Omega \\ u &= g && \text{auf } \partial\Omega \end{aligned}$$

- Beispiele:
  - stationäre Wärmeleitung mit Wärmequelle s

$$\cancel{\partial_t u} - a\Delta u = s$$

  - inkompressible Strömungen
  - elektrostatisches Potential

CG-SEM Diskretisierung

- **Poissongleichung:** $\Delta u = s \qquad \text{in } \Omega.$
- **Schwache Formulierung:** (φ Testfunktion)

$$\int_\Omega \Delta u \varphi \, d\Omega = \int_\Omega s\varphi \, d\Omega.$$

- **Partielle Integration:** (Bei Dirichlet-Bedingungen setze $\varphi = 0$ auf $\partial\Omega$)

$$\underbrace{\int_{\partial\Omega} \varphi \nabla u \cdot \vec{n} \, dS}_{=0} - \int_\Omega \nabla u \cdot \nabla \varphi \, d\Omega \qquad = \int_\Omega s\varphi \, d\Omega$$

$$\Rightarrow \quad \sum_{k=1}^{K} \left\{ - \int_{e^k} \nabla u \cdot \nabla \varphi \, dxdy \right\} \qquad = \sum_{k=1}^{K} \left\{ \int_{e^k} s\varphi \, dxdy \right\}.$$

- Gauss-Lobatto Quadratur $\{(\xi_i, w_i)\}_{i=0}^N$.
- **Kollokation:** Interpolationspunkte = Quadraturpunkte
- Seien $\{l_i\}_{i=0}^N$ die Lagrangepolynome zu den Interpolationspunkten $\{\xi_i\}_{i=0}^N$
- **Tensorprodukt:** $\{l_i(\xi)l_j(\eta)\}_{i,j=0}^N$ Basisfunktionen in 2D
- **CG**: Stetige, stückweise polynomiale Aprroximation:

$$u(x,y)|_{e^k} \approx U^k(\xi,\eta) = \sum_{i,j=0}^{N} U_{i,j}^k \ell_i(\xi)\ell_j(\eta),$$

$$\varphi(x,y)|_{e^k} \approx \phi^k(\xi,\eta) = \sum_{i,j=0}^{N} \phi_{i,j}^k \ell_i(\xi)\ell_j(\eta),$$

$$s(x,y)|_{e^k} \approx S^k(\xi,\eta) = \sum_{i,j=0}^{N} S_{i,j}^k l_i(\xi)l_j(\eta),$$

wobei die Werte $U_{i,j}^k$ bzw. $\phi_{i,j}^k, S_{i,j}^k$ dieselben sind entlang den Kanten und in jedem Eckpunkt.

- **CG-SEM Diskretisierung:**

$$\sum_{k=1}^{K} \left\{ \sum_{i,j} \phi_{i,j}^{k} \left[(\Delta U, \ell_i \ell_j)_N - S_{i,j}^{k} J_{i,j}^{k} w_i w_j \right] \right\} = 0, \tag{1}$$

wobei J die Jacobideterminante der Transformation auf das Referenzelement ist und

$$(\Delta U, \ell_i \ell_j)_N = - \left\{ \sum_{n=0}^{N} D_{i,n}^{(\xi)T} F_{n,j} w_n w_j + \sum_{m=0}^{N} D_{j,m}^{(\eta)T} G_{i,m} w_i w_m \right\}$$

- Auf das Referenzelement transformierte Fluss ∇u

$$F_{i,j} = \left\{ \frac{Y_\eta^2 + X_\eta^2}{J} \right\}_{i,j} \sum_{k=0}^{N} D_{i,k}^{(\xi)} U_{k,j} - \left\{ \frac{Y_\eta Y_\xi + X_\eta X_\xi}{J} \right\}_{i,j} \sum_{k=0}^{N} D_{j,k}^{(\eta)} U_{i,k},$$

$$G_{i,j} = \left\{ \frac{Y_\xi^2 + X_x i^2}{J} \right\}_{i,j} \sum_{k=0}^{N} D_{j,k}^{(\eta)} U_{i,k} - \left\{ \frac{Y_\eta Y_\xi + X_\eta X_\xi}{J} \right\}_{i,j} \sum_{k=0}^{N} D_{i,k}^{(\xi)} U_{k,j}.$$

- Lineares Gleichungssystem mit symmetrischer Koeffizientenmatrix $\rightarrow$ Conjugate Gradient (CG) Verfahren.

- Betrachte die folgende Gleichung punktweise:

$$\sum_{k=1}^{K} \left\{ \sum_{i,j} \phi_{i,j}^k \underbrace{\left[(\Delta U, \ell_i \ell_j)_N - S_{i,j}^k J_{i,j}^k w_i w_j \right]}_{\text{lokale Approximation in } e^k} \right\} = 0, \tag{2}$$

- Im Inneren einer Gitterzelle sind $\phi_{i,j}^k$ unabhängig.
  $\Rightarrow$ Löse:

$$(\Delta U, \ell_i \ell_j)_N - S_{i,j}^k J_{i,j}^k w_i w_j = 0.$$

- An den **Kanten**: Summe der lokalen Approximationen in den 2 benachbarten Elementen
- An den **Eckpunkten**: Summe der lokalen Approximationen in den 4 Elementen, die dieselbe Ecke besitzen.

Implementierung des Continuous Galerkin Verfahrens

Globale Methoden:

- *Mask*-Methode:
  - Setzt die mehrfach vorkommenden Knotenwerte zu Null.
  - Damit beeinflussen sie nicht die globalen Operationen im iterativen Löser (z.Bsp. Skalarmultiplikationen)
  - Bei Dirichlet Randbedingungen wird auch der Rand zu Null gesetzt. (Damit das Residuum, das gemasked wird, keinen Einfluss auf die Randwerte hat)

  $\Rightarrow$ **Erlaubt globale Operationen.**
- *UnMask*-Methode:
  - Kopiert die Werte an die mehrfach vorkommenden Knotenwerte wieder zurück, die von der Mask-Funktion zu Null gesetzt wurden.
  - Für die Berechnung der lokalen Operationen (z.Bsp. transformiertes Volumenintegral, Residuum)

  $\Rightarrow$ **Erlaubt lokale Berechnungen.**
- *GlobalSum*-Methode:
  - Addiert die Werte an den mehrfach vorkommenden Knotenpunkten (führt die Operationen in der geschweiften Klammer aus) und gibt die Werte an all diese Knotenpunkte wieder zurück.

Conjugate Gradient (CG) Verfahren

Das CG Verfahren ist ein iterativer Löser eines LGS

$$Ax = b,$$

wobei $A \in \mathbb{R}^{n \times n}$ symmetrisch und positiv definit ist. Nach spätestens n Schritten liefert dieses Verfahren die exakte Lösung.

CG-Löser:
Wähle $x_0 \in \mathbb{R}^n$.

$$d_0 := r_0 := b - Ax_0,\ \alpha_0 := r_0 \cdot r_0$$

Für $k = 0, \ldots, n-1$

$$\left|\begin{array}{ll} v_k & := Ad_k \\ \lambda_k & := \frac{\alpha_k}{d_k \cdot v_k} \\ x_{k+1} & := x_k + \lambda_k d_k \\ r_{k+1} & := r_k - \lambda_k v_k \\ \alpha_{k+1} & := r_{k+1} \cdot r_{k+1} \\ (\alpha_{k+1})^{\frac{1}{2}} & < \text{tolerance} \quad \Rightarrow \text{exit loop} \\ d_{k+1} & := r_{k+1} + \frac{\alpha_{k+1}}{\alpha_k} d_k. \end{array}\right. \tag{3}$$

Implementierung des Conjugate Gradient Verfahrens:

- Initialisierung der Lösung:
  - Dirichlet-Randbedingungen: Exakte Lösung am Rand.
  - Neumann-Randbedingungen: Initialer Wert am Rand.
- *Residual*
  - *Unmask*(U)
  - Berechne lokales, transformiertes Volumenintegral: $(\Delta U, \ell_i \ell_j)_N$
  - Berechne lokales Residuum
  - *GlobalSum(Residuum)*
  - *Mask(Residuum)*
- *MatrixAction*
  - *Unmask*(d)
  - Berechne lokales, transformiertes Volumenintegral: $(\Delta d, \ell_i \ell_j)_N$
  - *GlobalSum*$(\Delta d, \ell_i \ell_j)_N)$
  - *Mask*$((\Delta d, \ell_i \ell_j)_N)$
  - *Mask*(d)

Präkonditionierung

- **Ziel**: Bessere Kondition, schnellere Konvergenz.
- Äquivalente Umformung des Gleichungssystems

$$Ax = b$$

in

$$P_l A P_r z = P_l b$$
$$x = P_r z.$$

- Optimale Wahl von P_l und P_r im Sinne der Konvergenz:

$$P_l A P_r = I.$$

Diese Forderung ist aber äquivalent zur Invertierbarkeit von A.

- **Ziel:** Wähle P_l und P_r (leicht invertierbar + geringer Speicherplatzbedarf), so dass

$$P_l A P_r \approx I \quad \text{und somit} \quad \kappa(P_l A P_r) << \kappa(A)$$

Preconditioned CG-Verfahren (PCG)

- Wähle $P_r = P_l^T$, um Symmetrie des Problems zu behalten und das CG-Verfahren noch anwenden zu können
- Definiere $C^{-1} := P_l^T P_l$. Es soll $C \approx A$ gelten.

PCG-Löser:
Wähle $x_0 \in \mathbb{R}^n$.

$$
\begin{aligned}
& r_0 := b - Ax_0 \\
& \text{Löse } Cd_0 = r_0,\ \alpha_0 := r_0 \cdot d_0 \\
& \text{Für } k = 0, \ldots, n-1 \\
& \left|\begin{array}{lll}
v_k & := Ad_k & \\
\lambda_k & := \frac{\alpha_k}{v_k \cdot d_k} & \\
x_{k+1} & := x_k + \lambda_k d_k & \\
r_{k+1} & := r_k - \lambda_k v_k & \\
\|r_{k+1}\| & < \text{tolerance} & \Rightarrow \text{exit loop} \\
\text{Löse} & Cz_{k+1} = r_{k+1},\ \alpha_{k+1} := r_{k+1} \cdot z_{k+1} & \\
d_{k+1} & := z_{k+1} + \frac{\alpha_{k+1}}{\alpha_k} d_k. &
\end{array}\right.
\end{aligned}
\tag{4}
$$

Wahl des Präkonditionierers

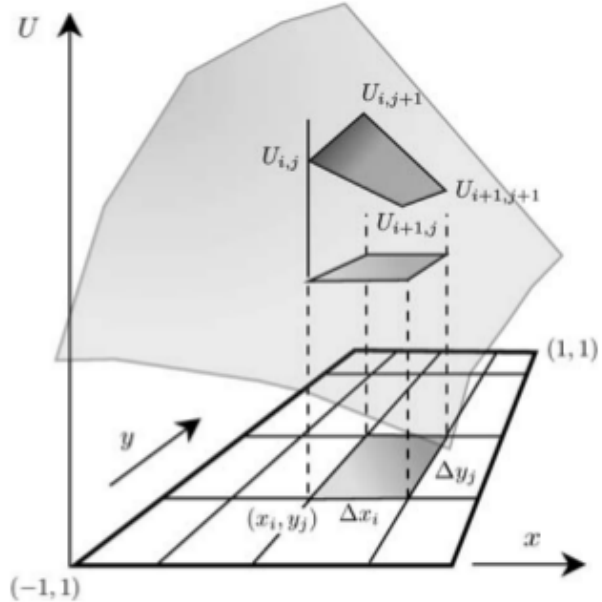
- $C = D$, wobei D die Hauptdiagonale von A ist (Jacobi-Präkonditionierer)
- FE Präkonditionierer: Lineare Finite Elemente Approximation auf dem Gitter, das von den Gauss-Lobatto Punkten erzeugt wird

Herleitung des Finite Elemente Präkonditionierers:

- Wir starten von derselben schwachen Galerkin Formulierung, die wir benutzt haben um die Spektralapproximation herzuleiten.

$$-\int_{\Omega} \nabla u \cdot \nabla \phi \, d\Omega.$$

- Betrachte das von den Interpolationspunkten erzeugte Gitter.



- Transformiere einzelne rechteckige Elemente auf das Einheitsquadrat in Koordinaten (ξ, η).
- Approximiere u auf einer Gauss-Lobatto Gitterzelle durch eine bilineare Interpolation.

$$U(\xi, \eta) = \sum_{k,l=0}^{1} U_{i+k,j+l} \phi_{k,l}$$

mit den bilinearen Basisfunktionen:

$$\begin{aligned}
\phi_{0,0} &= (1-\xi)(1-\eta), \\
\phi_{1,0} &= \xi(1-\eta), \\
\phi_{1,1} &= \xi\eta, \\
\phi_{0,1} &= (1-\xi)\eta.
\end{aligned}$$

- Der FE Präkonditionierer approximiert den spektralen Operator, da er denselben Ausdruck

$$-\int_{\Omega} \nabla u \cdot \nabla \phi \, d\Omega$$

approximiert.

- Wir bekommen auf den einzelnen Elementen lokale Steifigkeitsmatrizen, die wir dann zur globalen Steifigkeitsmatrix aufsummieren.

Lösen des präkonditionierten LGS

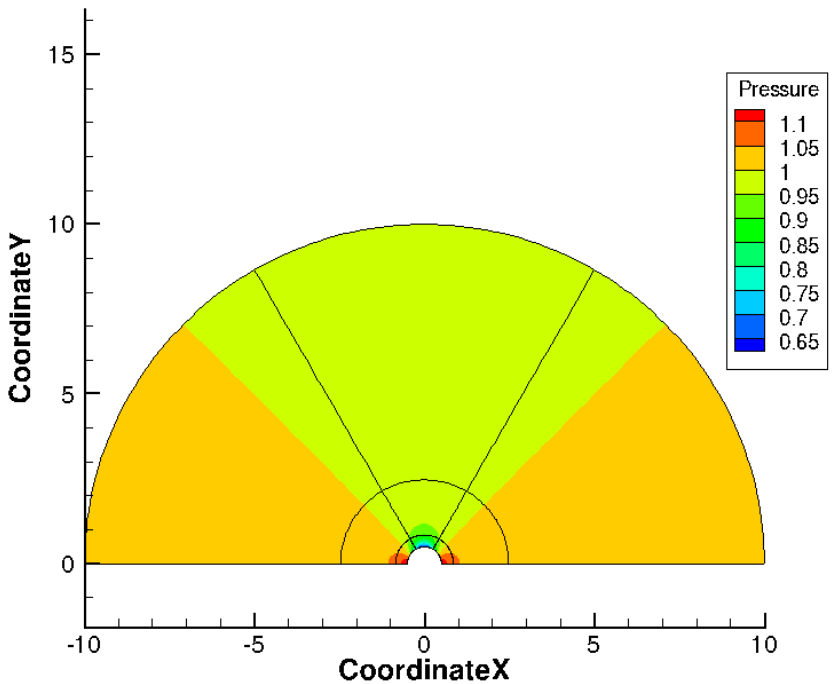
- Zum Lösen von
$$C_{\mathrm{FE}} z_{k+1} = r_{k+1},$$
verwenden wir ein Sweep des SSOR Verfahrens (Symmetrisches Gauß-Seidel-Relaxationsverfahren)
- Gauß-Seidel-Verfahren:
$$x_{m+1} = -(D+L)^{-1} R x_m + (D+L)^{-1} b.$$
- Rückwärtsgerichtetes Gauß-Seidel-Verfahren:
$$x_{m+1} = -(D+R)^{-1} L x_m + (D+R)^{-1} b.$$
- Kompositionen beider:
$$x_{m+\frac{1}{2}} = M_{GS} x_m + N_{GS} b, \qquad x_{m+1} = M_{RGS} x_{m+\frac{1}{2}} + N_{RGS} b$$
- Komposition der relaxierten Verfahren (SSOR):
$$x_{m+1} = M_{SGS}(\omega) x_m + N_{SGS}(\omega) b$$
- Nachteil: Wahl des Relaxationsparameters $\omega \in [1, 2)$.

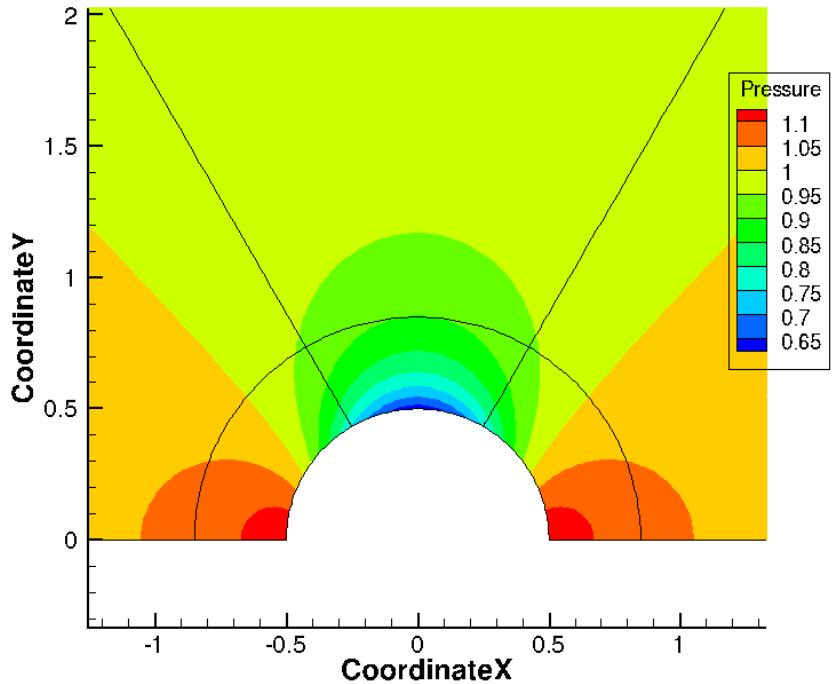
Implementierung des PCG Verfahrens:

- Unmask(Residuum)
- SSORSolver(Residuum,z) (elementlokal)
- GlobalSum(z)
- Mask(z)
- Mask(Residuum)

Potentialströmung um einen Halbzylinder

N=10, 3x3 Zellen. 1089 DOFs.
Für $V_\infty = 0.5$, $\rho_\infty = 1$ und $p_\infty = 1$.





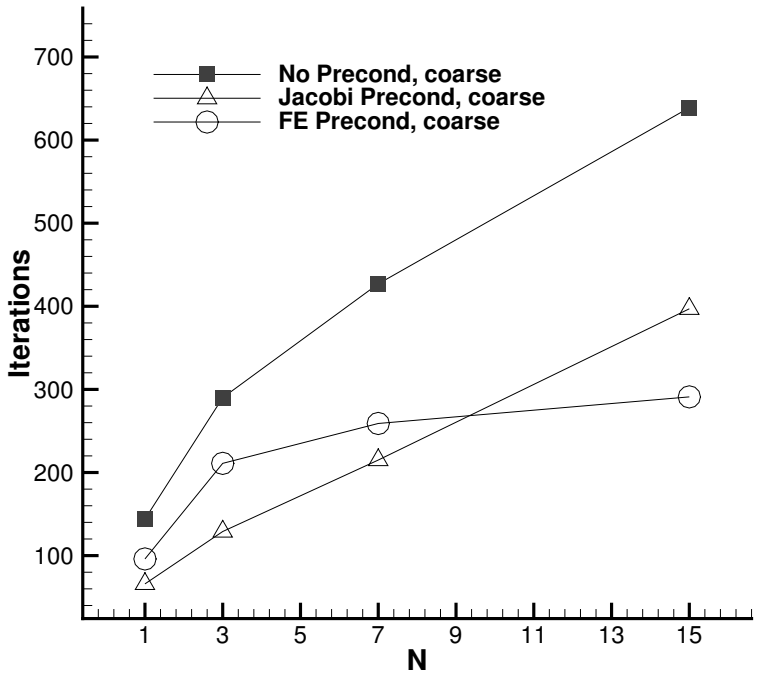
Konvergenztest

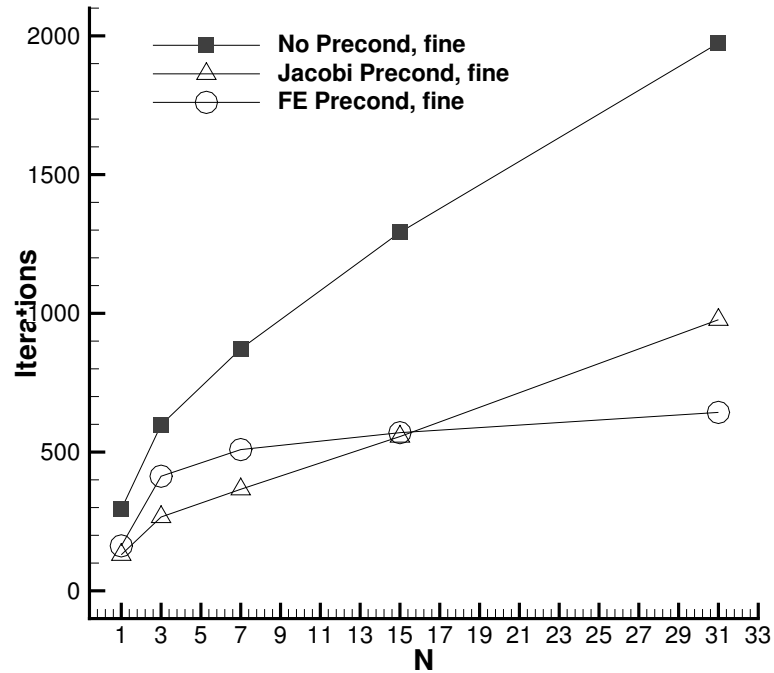
Tabelle: Konvergenztest (N=3)

Elemente	L2-Fehler	Konvergenzordnung
1x1	2.76E-002	
2x2	1.01E-003	4.7675229066
4x4	3.15E-005	5.0097026423
8x8	1.78E-006	4.1463647881
16x16	1.14E-007	3.9684714738
32x32	7.17E-009	3.9867647712
64x64	4.49E-010	3.9962087581
128x128	2.81E-011	3.9995377525

Vergleich der Iterationszahlen

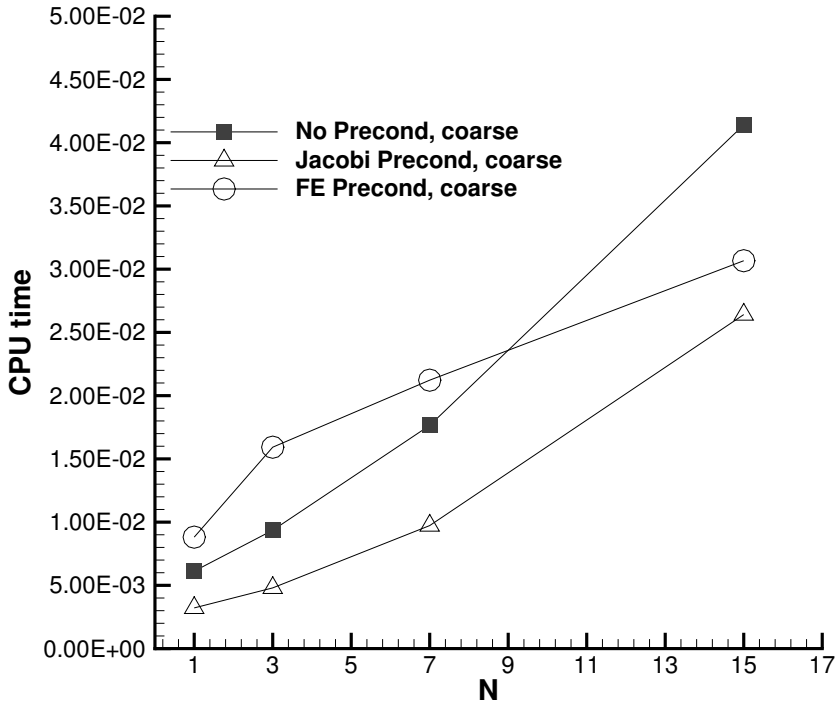
Coarse: 1024 DOFs
Fine: 4096 DOFs

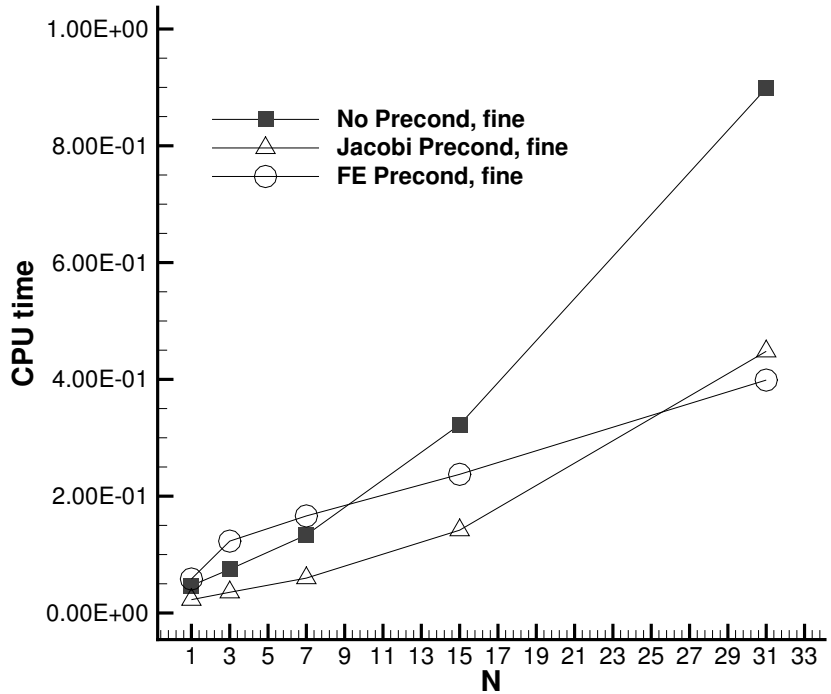




Vergleich der CPU Zeiten

Coarse: 1024 DOFs
Fine: 4096 DOFs.[1]

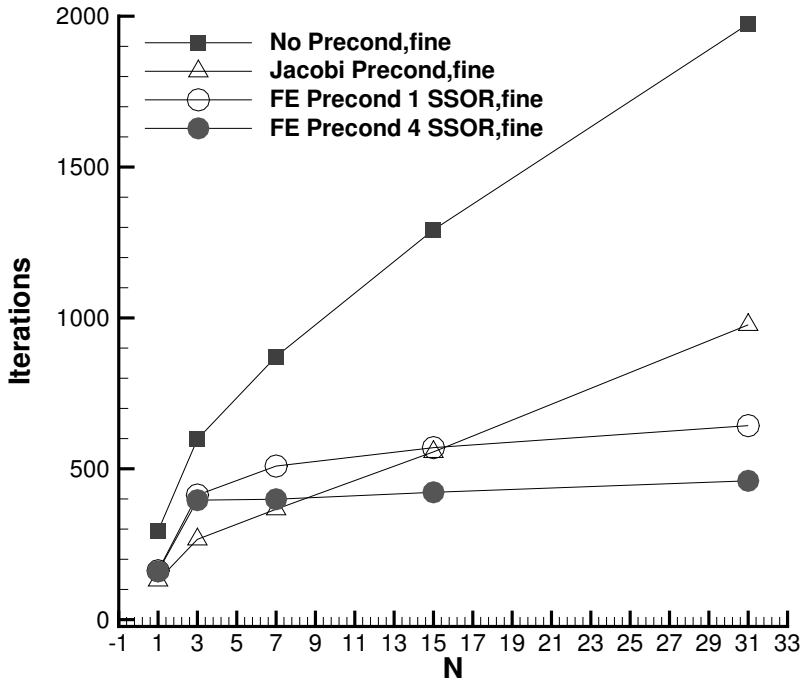


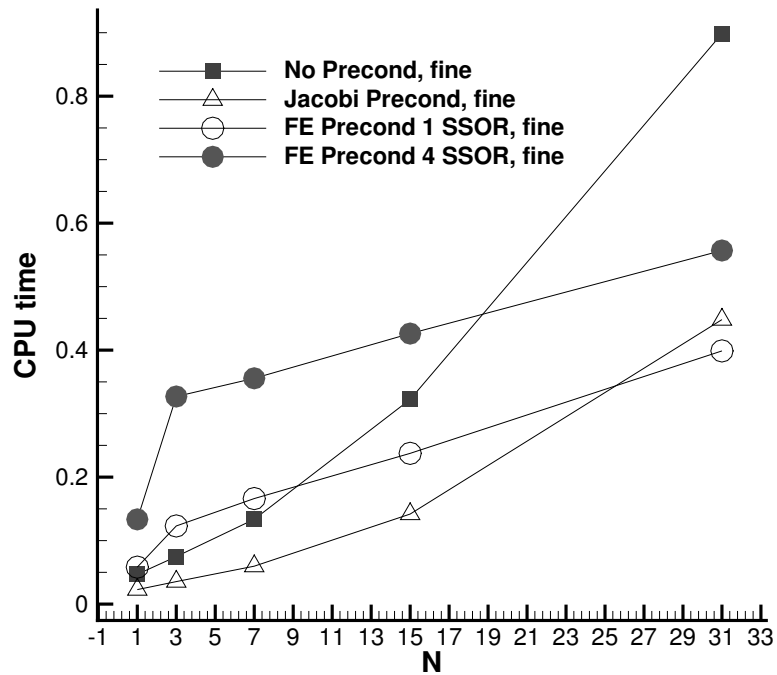


[1] Improving the Accuracy of Discontinuous Galerkin Schemes at Boundary Layers, Florian J. Hindenlang, Gregor J. Gassner, Claus-Dieter Munz, submitted to IJNMF, 2013

FE Präkonditionierer mit 4 SSOR Sweeps

Fine: 4096 DOFs.





Zusammenfassung

- CG-SEM Diskretisierung für Poissongleichung
- Für die lokale Berechnungen des Matrix-Vektorprodukts und den globalen Operationen im iterativen Löser, brauchen wir Mask-Methoden
- LGS symmetrisch $\rightarrow$ verwende das Conjugate Gradient Verfahren
- Benötigen Präkonditionierung, um die Iterationszahlen zu verringern
- FE besser als Jacobi Präkonditionierer für große Ordnung und kleine Anzahl an Elementen

Ausblick

- Einbau des Poissonlösers für die Implementierung der inkompressiblen Navier-Stokes Gleichungen

Vielen Dank für Ihre Aufmerksamkeit!