

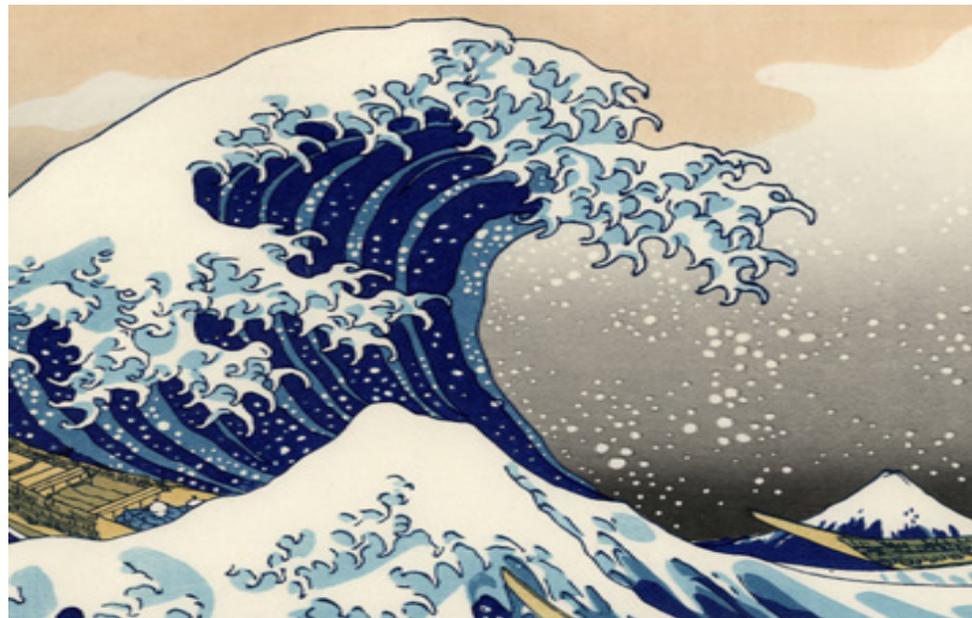


**Universität Bremen**  
**Vorlesung SoSe 2013**  
**Wissenschaftliches Rechnen**  
**und Anwendung in der Tsunami-Modellierung**

**Prof. Dr. Wolfgang Hiller, Prof. Dr. Alfred Schmidt**  
**Alfred-Wegener-Institut für Polar- und Meeresforschung,**  
**Bremerhaven/ Universität Bremen FB3**

**VAK: 03-252, ECTS:9**

**Termin: Mittwochs 10:15- 12:00, Ort: MZH 2490, Donnerstag 12:15-  
14:00 Uhr, Ort: MZH 2340 (je 2SWS), statt wöchentlicher Übungen  
ggfs. ein Blockpraktikum am Ende des Semesters**





## ***Vorlesung „ Wissenschaftliches Rechnen und Anwendung in der Tsunami Modellierung “***

---

In der Klima- und Umweltforschung am AWI werden häufig komplexe Systeme z.B. gekoppelte globale Ozean-Eis-Atmosphäre Modelle oder Risikowarnmodelle (Tsunami, Hangrutschungen,.. ) hoher algorithmischer Komplexität numerisch auf Höchstleistungsrechnern gerechnet. Dies setzt neben dem Verständnis der physikalischen Gesetze und Parametrisierungen (z.B. Flachwassergleichung, volle dynamische Bewegungsgleichung des Ozeans,..) eine intensive Beschäftigung mit den Methoden des wissenschaftlichen Rechnens und den zugrundeliegenden Algorithmen und deren Parallelisierung voraus, insbesondere im Bereich effizienter Löser für partielle Differentialgleichungen, aber auch den Grundoperationen der linearen Algebra.

Ziel der Vorlesung soll es sein, Studenten im Hauptstudium der Fachrichtung Mathematik, Technomathematik oder Informatik einen Überblick über die Methoden des Wissenschaftlichen Rechnens und speziell auch der Parallelisierung numerischer Verfahren zu geben. Dabei baut die Vorlesung auf dem Stoff der im Wechsel gehaltenen Vorlesung Grundlagen des Wissenschaftlichen Rechnens I "Parallele Algorithmen und Rechnerarchitektur" auf. Dort wurden die Kenntnisse zur Leistungsbewertung von parallelen Algorithmen sowie die Grundlagen zur Rechnerarchitektur, Leistungsmodellierung moderner Höchstleistungsrechner und zugehörige Programmiermodelle dargestellt. Für Hörer, welche diese Vorlesung nicht besucht haben, kann eine Zusammenfassung der wichtigsten Punkte zu Beginn dieser Vorlesung erfolgen.

In der Vorlesung werden neben einfachen Beispielen paralleler numerischer Verfahren aus dem Gebiet der Lösung linearer Gleichungssysteme auch ausgewählte parallele numerische Verfahren zur Lösung partieller Differentialgleichungen vorgestellt, welche Anwendung in einem Tsunami Simulationsmodell mit nichthydrostatischem Druckterm finden. Die parallele Effizienz dieser Verfahren sowie die Zeit/Speicherkomplexität im Vergleich mit der seriellen Version wird ebenfalls in der Vorlesung analysiert.



## ***Vorlesung „Wissenschaftliches Rechnen und Anwendung in der Tsunami-Modellierung“***

---

Prof Dr. Wolfgang Hiller, Prof. Dr. Alfred Schmidt  
Alfred-Wegener-Institut für Polar- und Meeresforschung, Bremerhaven/ FB3 Universität Bremen

### **Inhalt:**

#### ***Einführung in die Thematik und Motivation***

- Was ist wissenschaftliches Rechnen?
- Wissenschaftliches Rechnen und parallele numerische Verfahren

#### ***Rechnerarchitektur, Leistungsbewertung von Rechnern und Algorithmen und Konzepte der Parallelisierung***

- Zusammenfassung der wichtigsten Punkte aus der im Wechsel gehaltenen Vorlesung Wissenschaftliches Rechnen I "Parallele Algorithmen und Rechnerarchitektur"

#### ***Parallelisierung direkter Verfahren***

- Grundoperationen der linearen Algebra
- Vektor,- Matrixmultiplikation
- LU-Zerlegung (left-, right-looking, Crout)
- Optimierte numerische Bibliotheken
  - Blockung von Algorithmen
  - BLAS level 1,2,3
  - SCALAPACK
  - IMSL und NAG
- Gauß-Elimination
- Zyklische Reduktion
- Schur-Komplementmethode



## ***Vorlesung „Wissenschaftliches Rechnen und Anwendung in der Tsunami-Modellierung“***

---

### ***Parallelisierung iterativer Verfahren***

- Relaxationsverfahren (Gauß-Seidel-Verfahren, Jacobi -Verfahren)
- CG Verfahren
- Krylov Unterraum: GMRES-Verfahren
- parallele Vorkonditionierung
- geometrische/ algebraische Multigrid Methode
- Schwarzsche Gebietszerlegung
- Bei den in der Vorlesung behandelten parallelen numerischen Verfahren erfolgt stets ein Vergleich der seriellen Version mit der parallelen Version und ein Vergleich von inhärenter Parallelität, Speicher- und Zeitkomplexität und Datenorganisation.

# Tsunamimodellierung

## 2D - Flachwassergleichungen

Massenerhaltungsgleichung:

$$\eta_t + \nabla \cdot (\mathbf{u}H) = 0$$

Impulsgleichung (in horizontaler Richtung):

$$\mathbf{u}_t + (\mathbf{u} \cdot \nabla) \mathbf{u} + \mathbf{f} \times \mathbf{u} + g \nabla \eta + \mathbf{F} = 0$$

Die Oberflächenauslenkung  $\eta(x,y,t)$  und die horizontalen Geschwindigkeitskomponenten  $\mathbf{u}(x,y,t) = (u_x, u_y)$  sind unbekannt.

### Anfangsbedingungen

$$\eta(x,y,0) = \eta_0(x,y), \quad \mathbf{u}(x,y,0) = \mathbf{u}_0(x,y)$$

### Randbedingungen

$$\mathbf{u} \cdot \mathbf{n} = 0 \quad \text{an festen Rändern}$$

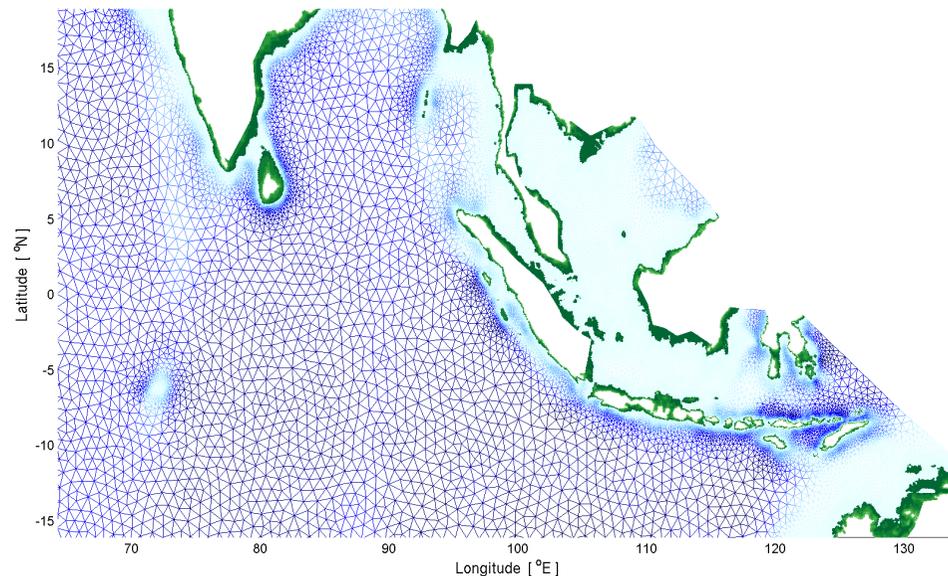
$$\mathbf{u} \cdot \mathbf{n} = \sqrt{(\eta g / H)} \quad \text{an offenen Rändern}$$

## Unstrukturierte Gitter

Mit unstrukturierten Gittern können die Küstenlinien vernünftig dargestellt werden.

## Finite Elemente Methode

TsunAWI basiert auf der  $P_1$ - $P_1^{nc}$  Finite Elemente Methode und dem Leap-Frog-Zeitschrittverfahren.

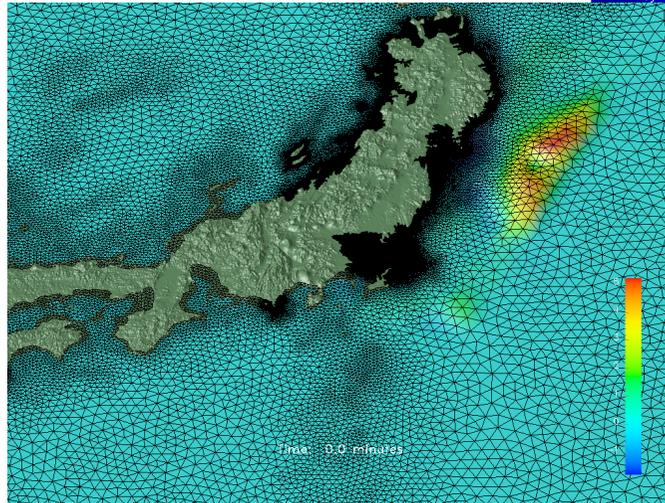


# Simulation des Japan-Tsunamis am 11. März 2011

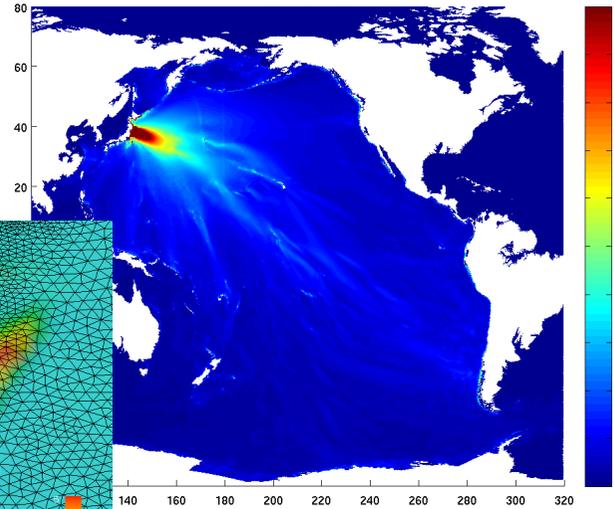
**Erdbeben** 38.31 N 142.52 E

Magnitude: Mw = 9.0

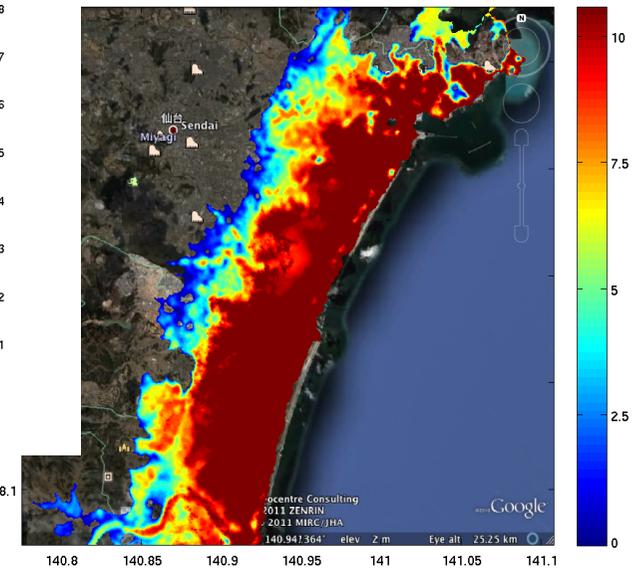
**Anfangsauslenkung  $\eta_0$**



Maximum wave height (largest value: 27.2 m)

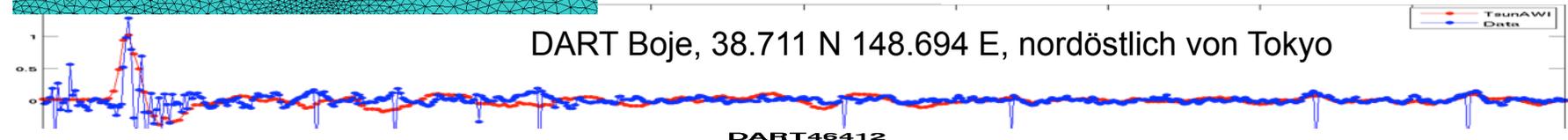


**Überflutung**



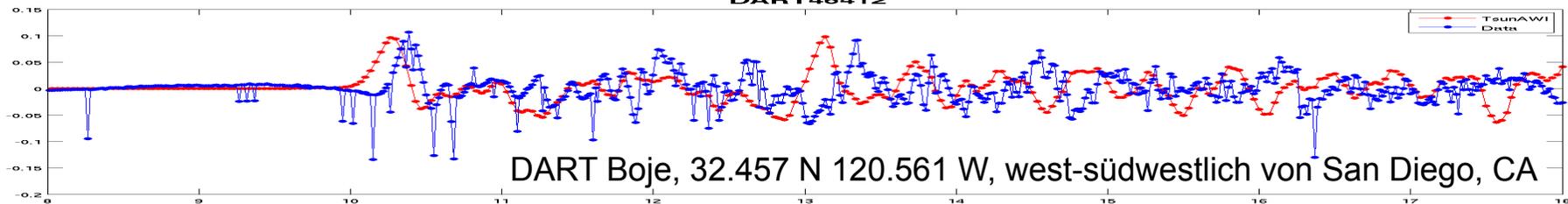
**Vergleich mit gemessenen Werten**

DART Boje, 38.711 N 148.694 E, nordöstlich von Tokyo



DART46412

DART Boje, 32.457 N 120.561 W, west-südwestlich von San Diego, CA



## Simulation des Japan-Tsunamis am 11. März 2011



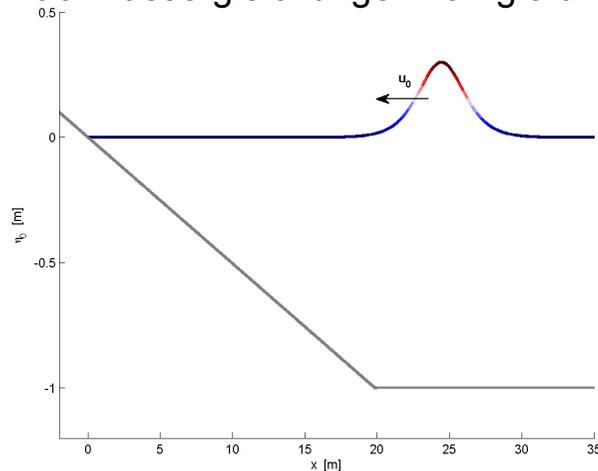
# Nichthydrostatischer Ansatz

## Der Druck, $p = p_h + q$

Aufspaltung in hydrostatischen Anteil  $p_h$  und nichthydrostatischen Anteil  $q$ . Im Flachwassermodell wird  $q$  vernachlässigt.

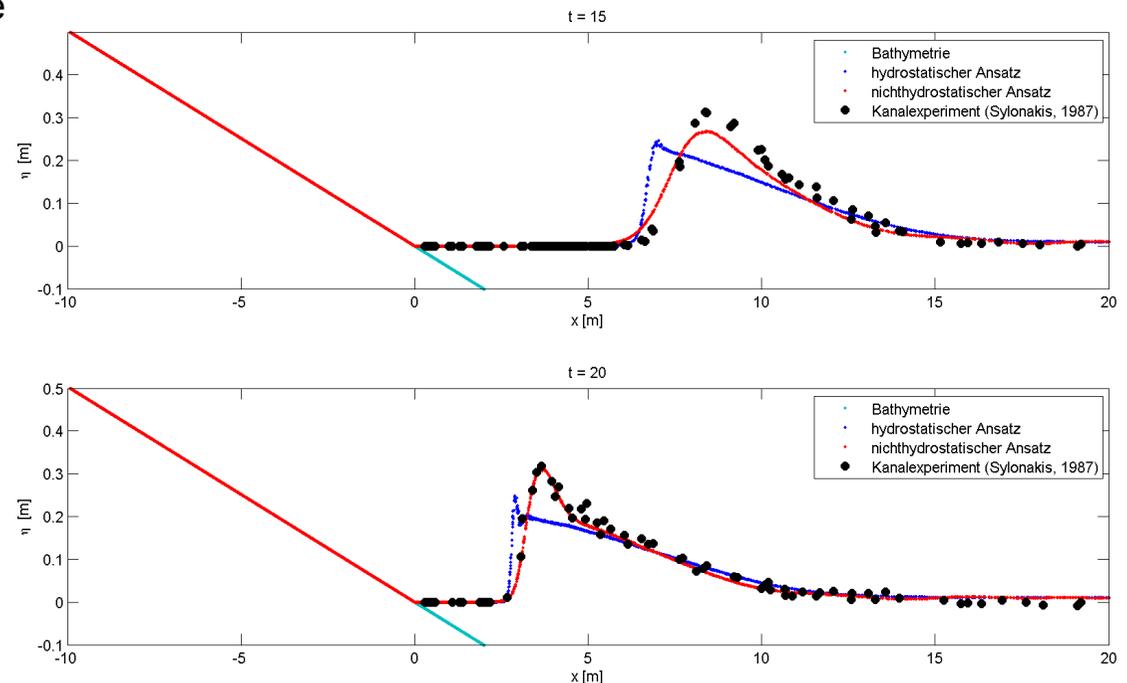
## Nichthydrostatischer Ansatz

Zur Steigerung der Genauigkeit werden die Ergebnisse der hydrostatischen Flachwassergleichungen korrigiert.



## Genauigkeit hat seinen Preis

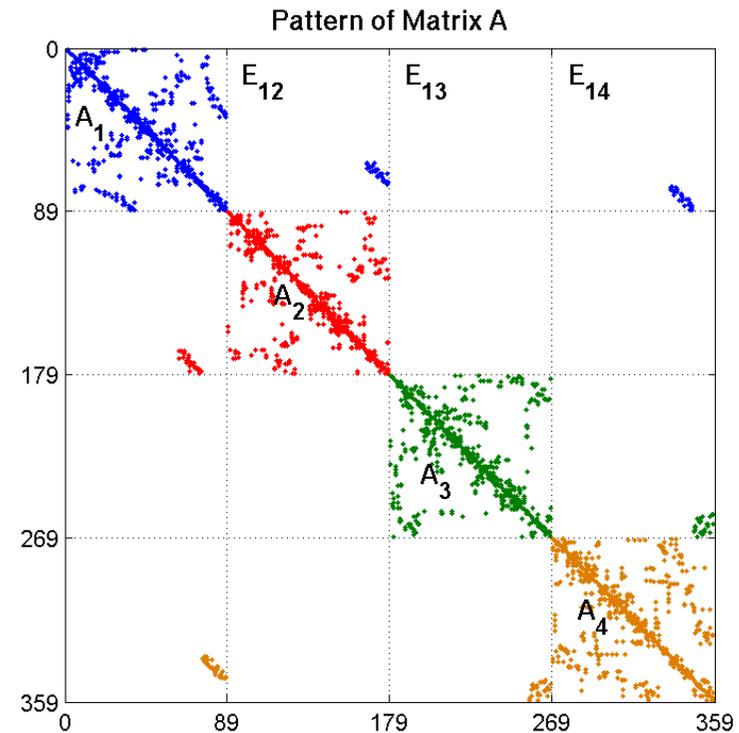
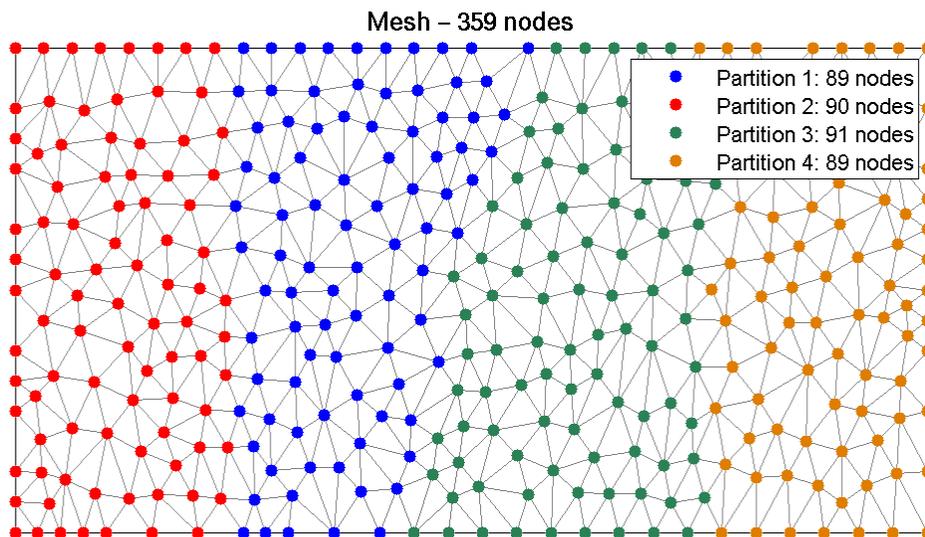
Zur Bestimmung der zusätzlichen Unbekannten  $q$  muss in jedem Zeitschritt ein großes, dünnbesetztes Gleichungssystem  $Aq = b$  gelöst werden. Dies spiegelt sich in erhöhtem Speicheraufwand und verlängerter Rechendauer wider.



# Gebietszerlegung

## MPI-Parallelisierung

Für die Parallelisierung wird das Rechengebiet zerlegt und die einzelnen Partitionen den Prozessoren zugewiesen. Die Daten an Partitions Grenzen werden zwischen den Prozessoren mit Hilfe des Message Passing Interface (MPI) übermittelt.



## Matrixstruktur

In der zugehörigen Matrixstruktur stehen in der Offdiagonalen die Daten, welche mehr als einen Prozessor betreffen, also über die Partitions Grenzen hinaus gehen.

## Krylov-Unterraumverfahren und Präkonditionierung

### Krylov-Unterraumverfahren GMRES

Zum Lösen des linearen Gleichungssystems wird in einem iterativen Verfahren das Residuum  $r_i = b_i - Ax_i$  minimiert. Dabei gilt  $r_i \in \text{span}\{r_0, Ar_0, A^2r_0, \dots, A^{i-1}r_0\}$ .

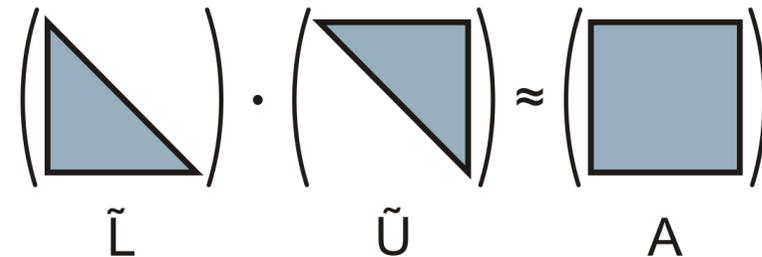
Die Konvergenzrate hängt von der Kondition der Matrix ab.

### Lokale und globale Präkonditionierung

Bei der Präkonditionierung wird  $A^{-1}$  durch eine Matrix mit besserer Kondition approximiert.

Nun führt jeder Prozessor lokal eine unvollständige LU-Zerlegung (ILU) auf einen Teil der Matrix aus.

Wie dieser Teil definiert ist, wird durch die globale Präkonditionierung festgelegt.



$$\begin{pmatrix} \triangle \\ \square \end{pmatrix} \cdot \begin{pmatrix} \square \\ \triangle \end{pmatrix} \approx \begin{pmatrix} \square \\ \square \end{pmatrix}$$

$\tilde{L} \quad \tilde{U} \quad A$

### Unvollständige LU-Zerlegung

Da die Dreiecksmatrizen der vollständigen LU-Zerlegung einer dünnbesetzten Matrix im Allgemeinen nicht dünnbesetzt sind, wird diese nur approximiert. Es gibt verschiedene Ansätze, wie Einträge ungleich Null unterdrückt werden. Zum Beispiel:

ILU(2):  $l_{ij}/u_{ij} = 0$ , falls  $a_{ij} = 0$  und  $\bar{a}_{ij} = 0$ ,

ILUT:  $l_{ij}/u_{ij} = 0$ , falls  $a_{ij} < \varepsilon$ ,

mit  $L = (l_{ij})_{ij}$ ,  $U = (u_{ij})_{ij}$ ,  $A = (a_{ij})_{ij}$ ,  $A^2 = (\bar{a}_{ij}^2)_{ij}$

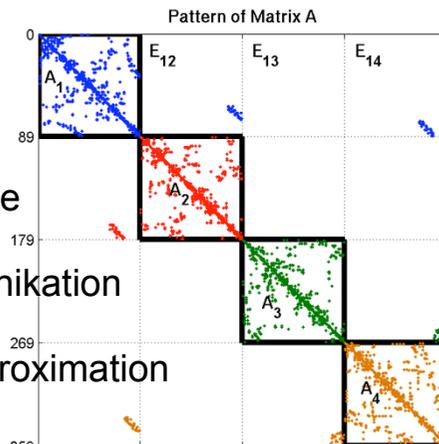
# Globale Präkonditionierung

## Block Jacobi

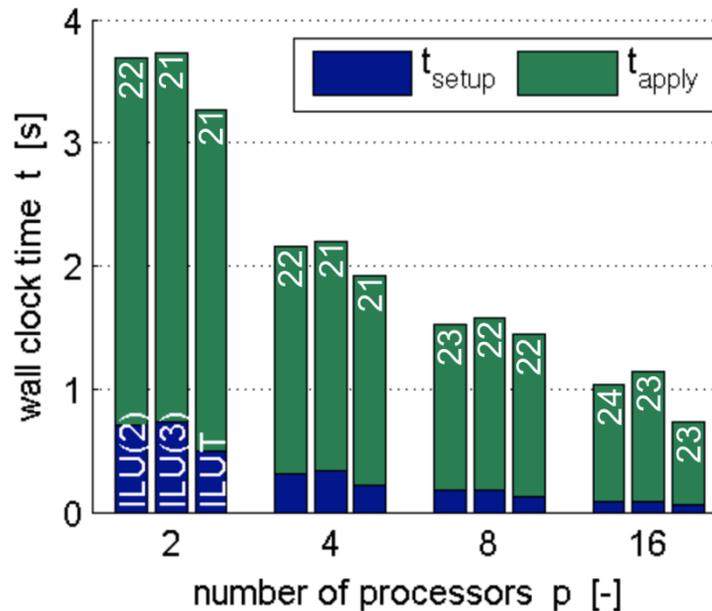
Diagonalblöcke

keine Kommunikation

schlechte Approximation



Block Jacobi



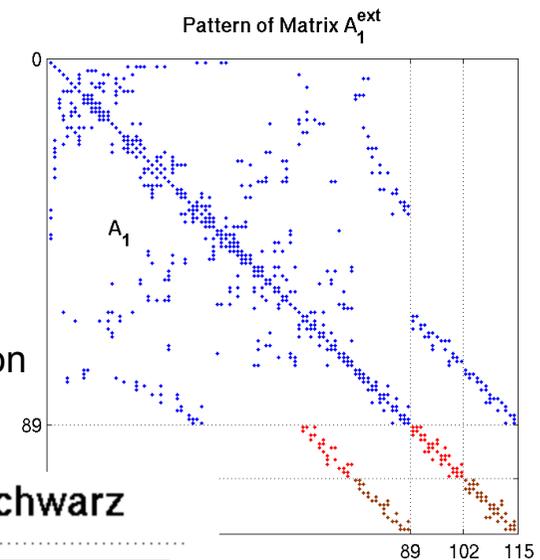
## restricted Additive Schwarz

erweiterte Diagonalblöcke

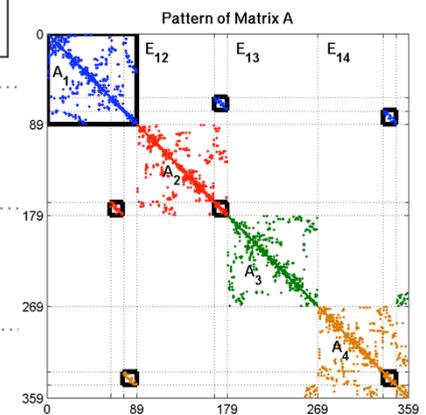
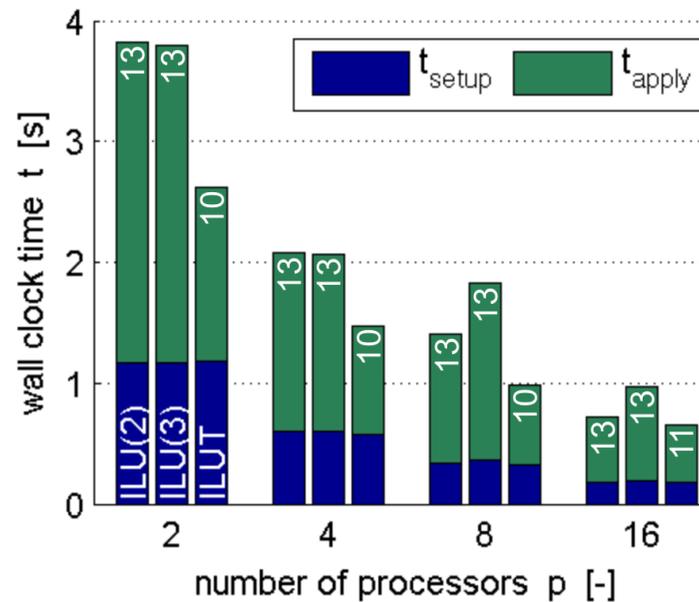
bessere Approximation

Setup teuer, da Kommunikation

nötig



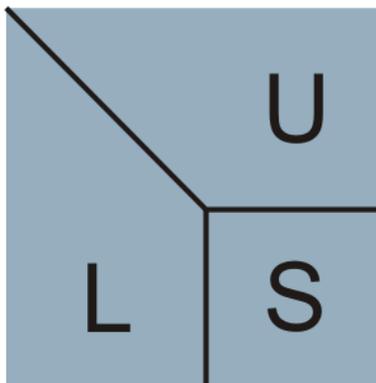
restricted Additive Schwarz



# Globale Präkonditionierung

## Schurkomplement

Mit Hilfe des Schurkomplements kann lokal ein, auf die Interface-Knoten reduziertes Gleichungssystem gelöst werden. Die Werte an den inneren Knoten werden anschließend explizit berechnet. Das Schurkomplement erhält man durch Abbruch der Gaußelimination.



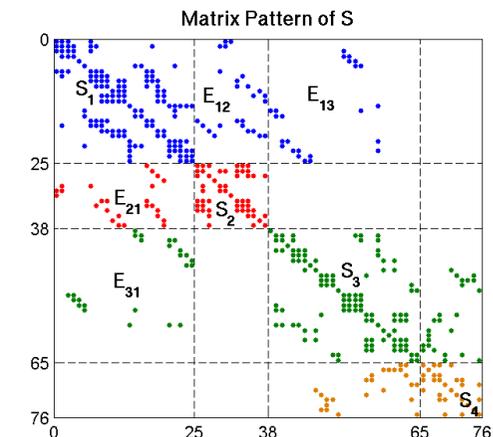
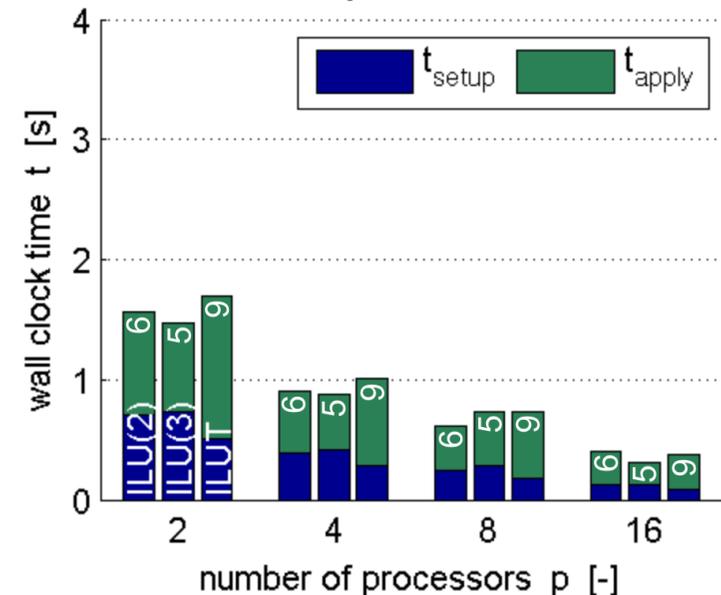
Als Präkonditionierer wird das Schurkomplement nur approximiert und das reduzierte Gleichungssystem mit einem inneren GMRES-Verfahren gelöst.

### Vorteile

Das Schurkomplement ist besser konditioniert als die Matrix selbst und die Dimension des reduzierten LGS ist viel geringer.

Gilt  $LU \approx A$ , dann gilt automatisch  $L_s U_s \approx S$ .

## Schur Complement + GMRES

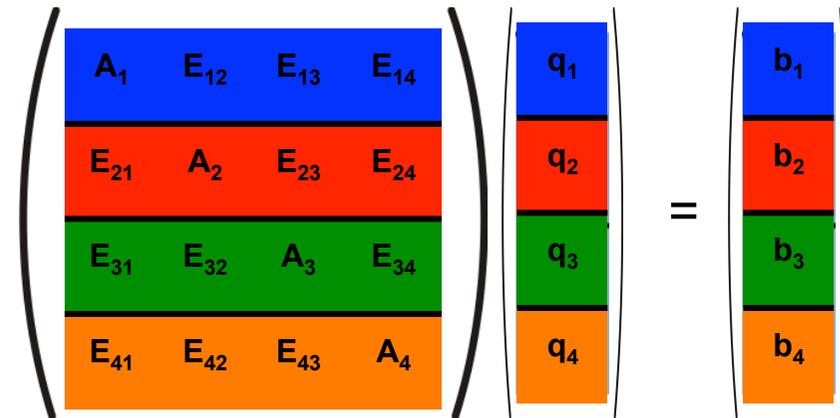


# Lösen des Gleichungssystems $Aq = b$

## Verteilte Daten

Die Werte des Gleichungssystems, der rechten Seite und des Lösungsvektors sind auf die verschiedenen Prozessoren verteilt. Somit berechnet der Prozessor  $P_i$

$$A_i q_i + \sum E_{ij} q_j = b_i, \quad i = 1, \dots, 4.$$



## Ablauf (Pseudocode)

```
CALL PCSetType( pc, BJ )
```

```
CALL PCSetILUType( pc, ILUT )
```

```
CALL PCSetup( pc, A )
```

```
CALL SolverApply( A, b, q )
```

### Löser: GMRES

```
 $r_0 = b - Aq$ 
```

```
DO WHILE( norm( $r_k$ ) >  $\epsilon$  )
```

```
...
```

```
!!$  $z = K^{-1}v$ , mit  $K^{-1} \approx A^{-1}$ 
```

```
CALL PCApply( pc, v, z )
```

```
...
```

```
 $r_k = \dots$ 
```

```
END DO
```

$$z_i = U_i^{-1} L_i^{-1} v_i$$

### Globaler Prädiktionierer: Block Jacobi

Der lokale Prädiktionierer wird auf den Diagonalblock  $A_i$  angewendet.

### Lokaler Prädiktionierer: ILU Threshold

ILU-Regel:  $a_{ij} < \epsilon \Rightarrow l_{ij} / u_{ij} = 0$

LU-Zerlegung per Gauß-Elimination. Werte ungleich Null werden gemäß der ILU-Regel erlaubt oder unterdrückt.

```
CALL PCILU( pc,  $A_i$  )  $L_i U_i \approx A_i$ 
```

## Lösen des Gleichungssystems $Aq = b$

### Ablauf II (Pseudocode)

CALL PCSetType( pc, RAS )

CALL PCSetILUType( pc, ILU2 )

CALL PCSetup( pc, A ) 

CALL SolverApply( A, b, q )

### Löser: GMRES

$r_0 = b - Aq$

DO WHILE( norm( $r_k$ ) >  $\epsilon$  )

...

!!\$  $z = K^{-1}v$ , mit  $K^{-1} \approx A^{-1}$

CALL PCApply( pc, v, z )

...

$r_k = \dots$  

END DO

### Globaler Prädiktionierer: **restricted Additive Schwarz**

Der lokale Prädiktionierer wird auf die Blockmatrix  $A_{i,\text{ext}}$  angewendet. Dabei besteht  $A_{i,\text{ext}}$  aus dem Diagonalblock  $A_i$  und den Werten der Interface-Knoten angrenzender Partitionen.

### Lokaler Prädiktionierer: **ILU(2)**

ILU-Regel:  $a_{ij} = 0 \wedge \bar{a}_{ij} = 0 \Rightarrow l_{ij} / u_{ij} = 0$ , mit  $(\bar{a}_{ij})_{ij} = A^2$

LU-Zerlegung per Gauß-Elimination. Werte ungleich Null werden gemäß der ILU-Regel erlaubt oder unterdrückt.

CALL MatExtend(  $A_i$ ,  $A_{i,\text{ext}}$  )

CALL PCILU( pc,  $A_{i,\text{ext}}$  )

$$L_{i,\text{ext}} U_{i,\text{ext}} \approx A_{i,\text{ext}}$$

1) Erweiterung von  $v_i$  und  $z_i$ , entsprechend  $A_{i,\text{ext}}$ .

2) Berechnung von  $z_{i,\text{ext}}$ :

$$z_{i,\text{ext}} = U_{i,\text{ext}}^{-1} L_{i,\text{ext}}^{-1} v_{i,\text{ext}}$$

3) Beschränkung des Vektors  $z_{i,\text{ext}}$  auf die Werte der eigenen Partition:  $z_i = (z_{i,\text{ext}})_i$

# Lösen des Gleichungssystems $Aq = b$

## Ablauf III (Pseudocode)

CALL PCSetType( pc, SCHUR)

CALL PCSetILUType( pc, ILUT )

CALL PCSetup( pc, A) →

CALL SolverApply( A, b, q)

### Löser: GMRES

$r_0 = b - Aq$

DO WHILE( norm( $r_k$ ) >  $\epsilon$  )

...

!!\$  $z = K^{-1}v$ , mit  $K^{-1} \approx A^{-1}$

CALL PCApply( pc, v, z) →

...

$r_k = \dots$

END DO

### Globaler Präkonditionierer: Schurkomplement

Der lokale Präkonditionierer wird auf den Diagonalblock  $A_i$  angewendet. Die ILU-Zerlegung des lokalen Schurkomplements  $S_i$  ist Teil der Zerlegung von  $A_i$ .

### Lokaler Präkonditionierer: ILUT

LU-Zerlegung per Gauß-Elimination. Werte ungleich Null werden gemäß der ILU-Regel erlaubt oder unterdrückt.

CALL PCILU( pc,  $A_i$ )

CALL PCRestrict( pc,  $pc^s$ )

$$L_i U_i \approx A_i$$

$$L_i^s U_i^s = (L_i U_i)|_{\text{interface}} \approx S_i$$

1) Lösen des Gleichungssystems auf dem Interface mit Hilfe des lokalen Präkonditionierers  $L_i^s U_i^s$ .

CALL VecRestrict( z,  $z^s$ )

CALL VecSchurRHS( v,  $v^s$ )

CALL SolverApply(  $pc^s$ ,  $v^s$ ,  $z^s$ )

$$z_i^s = (z_i)|_{\text{interface}}$$

$$v_i^s = \dots$$

2) Explizite Berechnung der übrigen Werte von  $z_i$ , ausgehend von den Werten  $z_i^s$  auf dem Interface.



## **Literatur**

---

Begleitend zur Vorlesung empfiehlt sich das Buch von Dongarra et al. zusammen mit dem Buch von Gundolf Haase, eine tiefergehende Einführung in parallele Multilevel Verfahren findet man in dem sehr empfehlenswerten Buch von Smith, B., Bjorstad, P. und Gropp, W. .

Dongarra, Duff, Sorensen, van der Vorst, 1998: Numerical linear Algebra for High-Performance Computers, SIAM, ISBN 0-89871-428-1

Frommer, A, 1990: Lösung linearer Gleichungssysteme auf Parallelrechnern. Institut für Angewandte Mathematik, Universität Karlsruhe.

Golub, G., Ortega, J. M., 1996: Scientific Computing. Eine Einführung in das wissenschaftliche Rechnen und Parallele Numerik. Teubner Verlag, Stuttgart, ISBN 3-519-02969-3

Haase, G., 1999: Parallelisierung numerischer Algorithmen für partielle Differentialgleichungen. Teubner Verlag, Stuttgart, Leipzig, ISBN 3-519-02970-7

Hockney, R. W., C. R. Jesshope: 1988, Parallel Computers 2, Adam Hilger, ISBN 0-85274-811-6.

Smith, B., Bjorstad, P. and Gropp, W., : Parallel Multilevel Methods for Elliptic Partial Differential Equations. Cambridge University Press, ISBN 0-521-49589-X

Van der Vorst, H.A., van Dooren, P. (Eds), 1989: Parallel Algorithms For Numerical Linear Algebra. North-Holland, Amsterdam, New York, Oxford, Tokyo, ISBN 0 444 88621 4

Hwang, K. 1993: Advanced Computer Architecture: Parallelism, Scalability, Programmability. McGraw-Hill.