UNIVERSITÄT BAYREUTH

Speicherallokation und Solveralgorithmen im Rahmen der Finite-Elemente-Analyse (FEA)

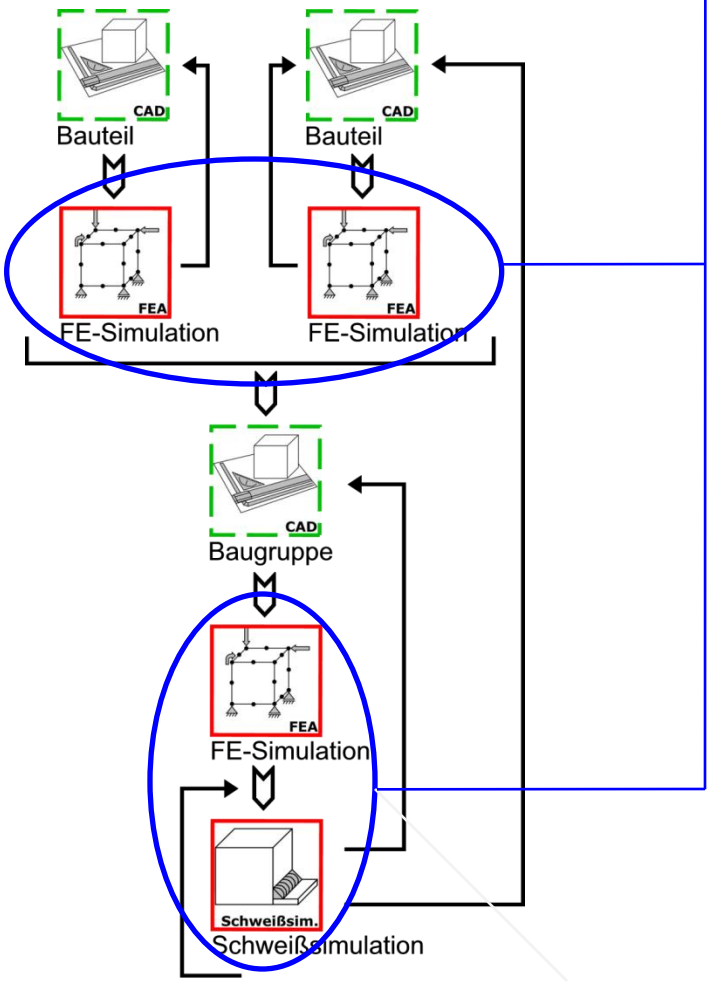
- mit Praxisvorführung
(Z88V13-Forschungsversion) -

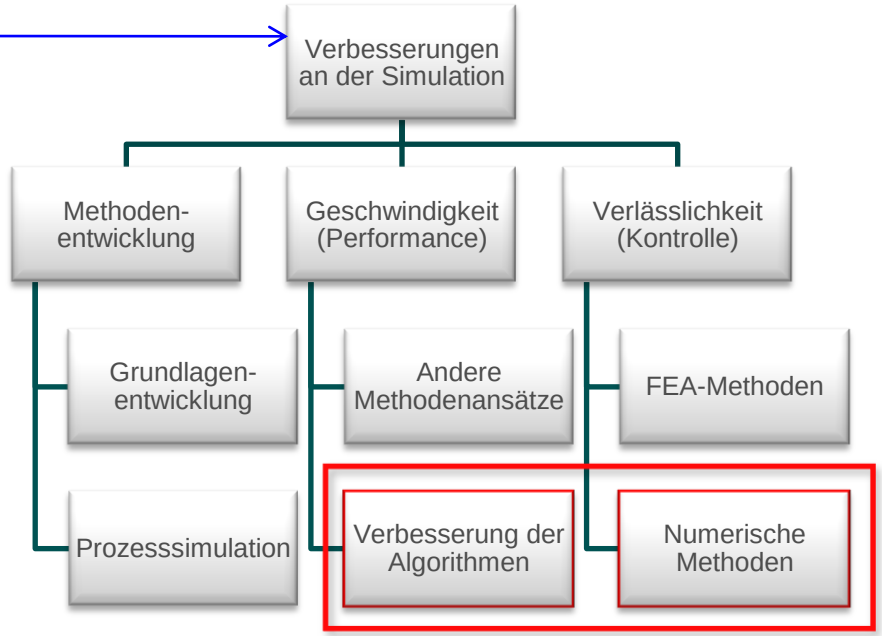
Dipl.-Ing. Bernd Roith
Dipl.-Ing. Martin Zimmermann
Prof. Dr.-Ing. Frank Rieg

Gliederung

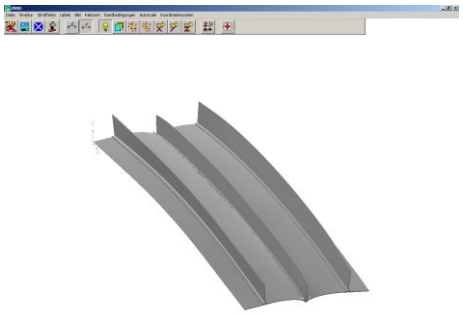
Aufgabenstellung

Moderner Entwicklungsprozess

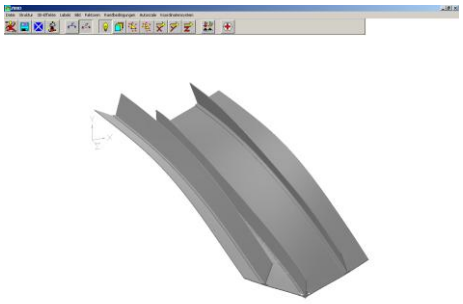




Numerische Methoden



Residuum 1E-9



Residuum 1E-10

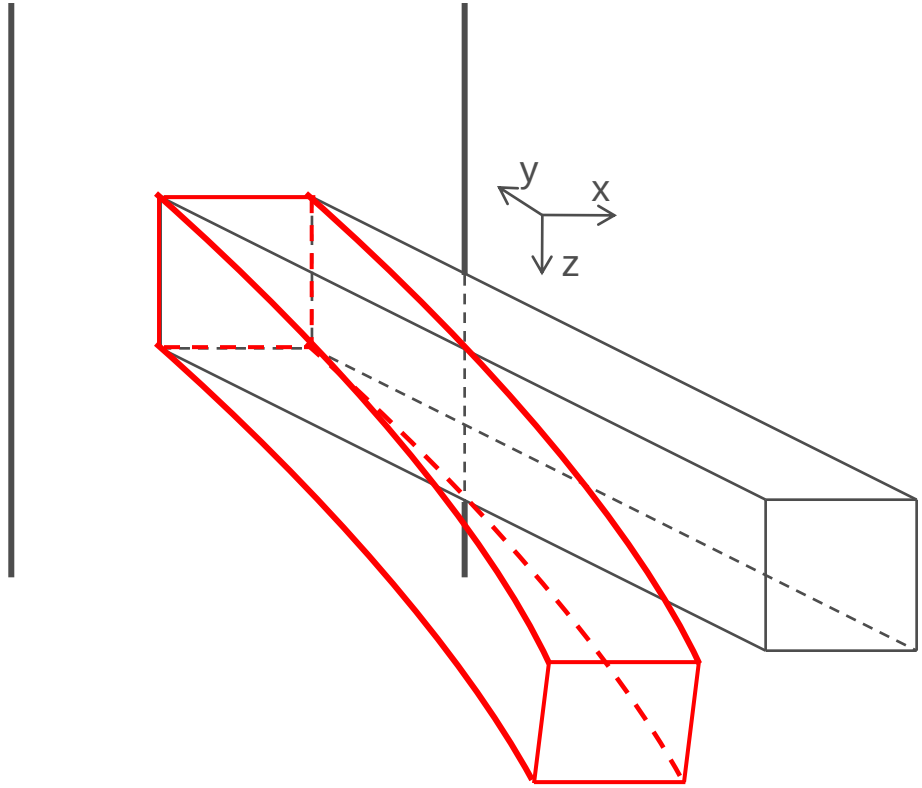
Aus BMBF-Projekt INSOFT

Modellierungs- und Synthesewerkzeuge

Analysewerkzeuge

Dipl.-Ing. Bernd Roith, Dipl.-Ing. Martin Zimmermann

Finite Elemente Theorie



deformiertes Model

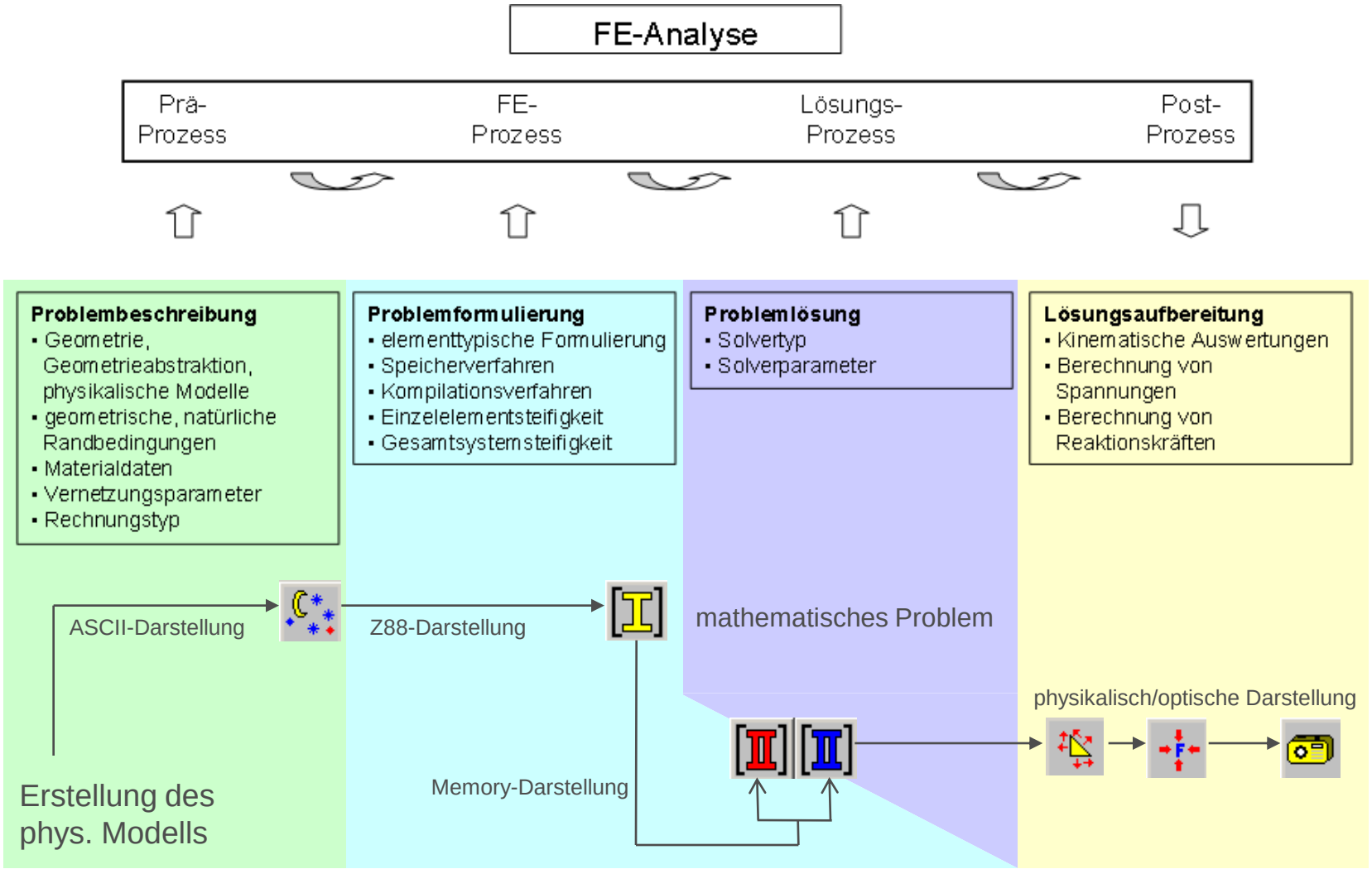
1. physikalisches Problem
2. Diskretisierung der Geometrie
3. Einzelelementbeschreibung
4. Gesamtsteifigkeitsmatrix
5. Randbedingungen
6. mathematischer Gleichungslöser
7. **Ergebnisse**

Undeformiertes Ausgangsmodel

Dipl.-Ing. Bernd Roith, Dipl.-Ing. Martin Zimmermann

Ansatz für die Softwarelösung

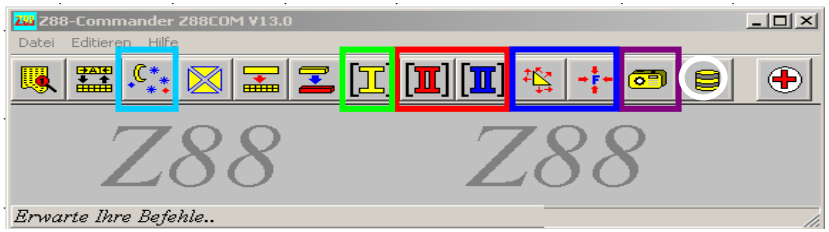
Prozessablauf der Softwarelösung



Quelle: Dissertation Martin Zimmermann, Universität Bayreuth, 2008

Finite Elemente Programm Z88

Aufbau und Module



Verfügbarkeit (32-64-Bit)

	Win32	Win64	LINUX 32	LINUX 64	SGI64, SUN64
Pointer	4	8	4	8	8
FR_XINT	4	4	4	4	8
FR_XLONG	4	4	4	8	8
FR_XLOLO	4	8	8	8	8
FR_XDOUB	8	8	8	8	8
FR_XQUAD	8	8 / 16	12 / 16	12 / 16	16

→ Schnittstellenkonverter:
- Cosmos
- Nastran (bdf)
- Abaqus (inp)

→ Sparsebuilder (für „Nicht-Null-Speicherung“ der Matrix K)

→ Finite Element Analyse Solver (rot: Iterativ, blau: direkt)

→ Spannungs- und Knotenkraftprozessor

→ Grafische Darstellung/Auswertung (opengl)

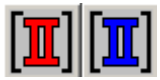
→ Steuerdatei (Speicherverwaltung für die einzelnen Module)

Dipl.-Ing. Bernd Roith, Dipl.-Ing. Martin Zimmermann

Funktionsumfang

Schnittstellenkonverter: MSC.Patran, MSC.Mentat, UGS NX 4.0, PTC Pro/ENGINEER Wildfire, Simufact, Abaqus CAE 6.3

Mapped Mesher mit Hilfe von Superelementen



Sparse-Solver sowohl ***Direkt*** als auch ***Iterative*** (CG mit SOR-Präkonditionierung und CG mit unvollständiger Choleskyzerlegung)

- Elementtypen: Stäbe, Balken, Kontinuumselemente (Tetraeder, Hexaeder (linear & quadratisch)), Platten, Schalen
- Randbedingungen: Kräfte, Verschiebungen, homogene Randbedingungen, Drücke, plastische und elastische Initaldehnungen

Spannungsprozessor (Gestaltänderungsenergiehypothese, Schubspannungshypothese, Normalspannungshypothese)

Knotenkraftprozessor

Opengl Darstellungsmodul: verformt, unverformt, FE-Netz, Verschiebungen, Spannungen (Gauß-Punkten), Randbedingungen

Dipl.-Ing. Bernd Roith, Dipl.-Ing. Martin Zimmermann

Praxisbeispiel

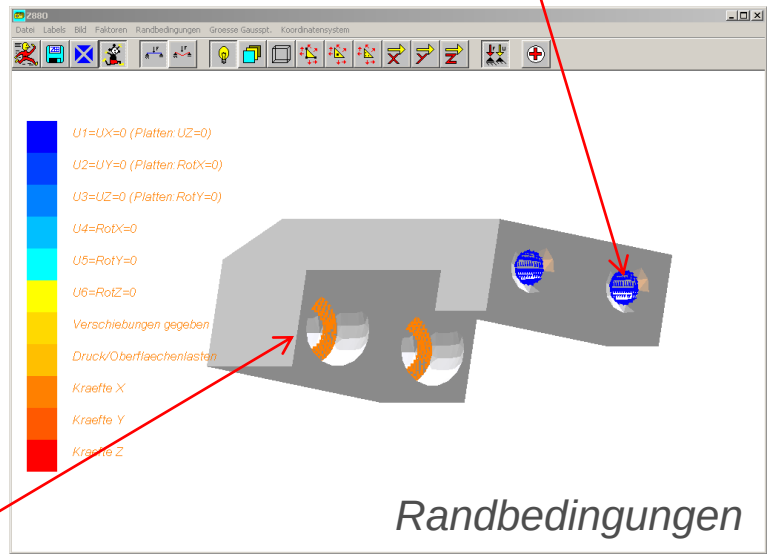
NASTRAN-Format

aus Pro/ENGINEER Wildfire 3.0

Einführung und Gesamtpräsentation

Z88V13

Lagerung (Fixierung)

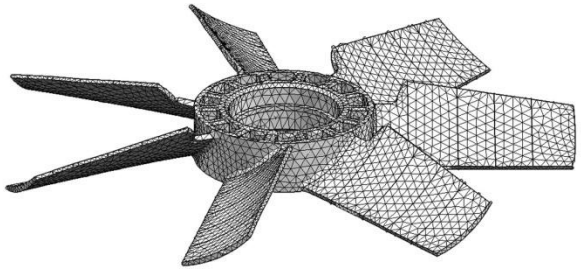


Randbedingungen

Kraft in Z-Richtung

Sparsebuilder 1

-Theorie 1-



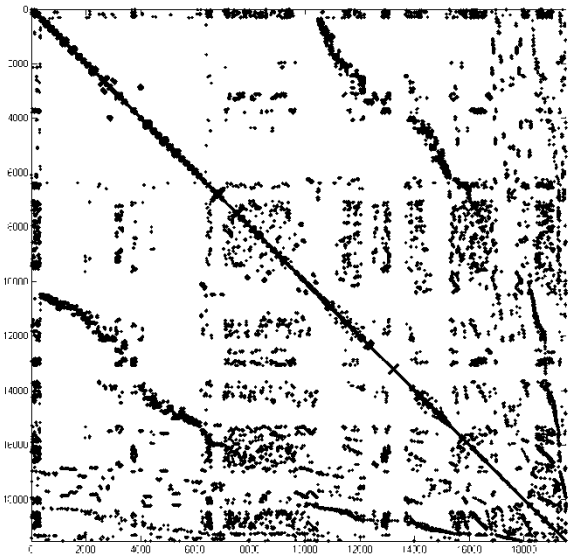
- **symmetrisch**
- **dünn besetzt**

- Problem:
  - Feststellung der Stellen der dünnen Besetzung,
  - Abspeicherungsverfahren der besetzten Stellen

$$\underline{\underline{K}} \cdot \underline{u} = \underline{F}$$

$$\sum_{E} \left(\iiint_{V^E} \underline{\underline{B}}^T \, \underline{\underline{C}} \, \underline{\underline{B}} dV \right) \cdot \underline{u} = \underline{F}$$

Symbolisches Abbild einer Steifigkeitsmatrix
(Nicht-Null-Elemente gekennzeichnet)

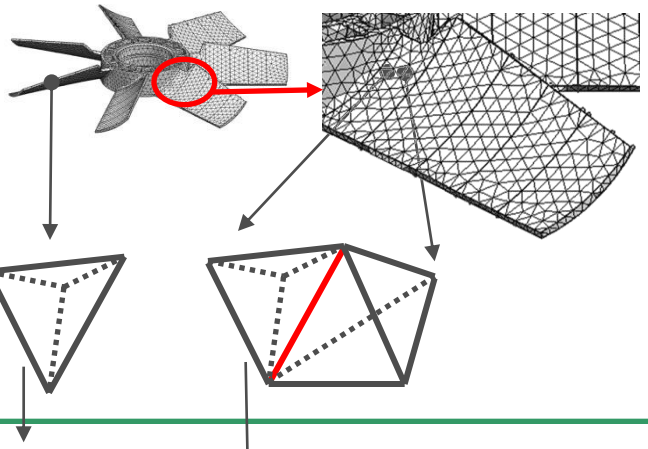


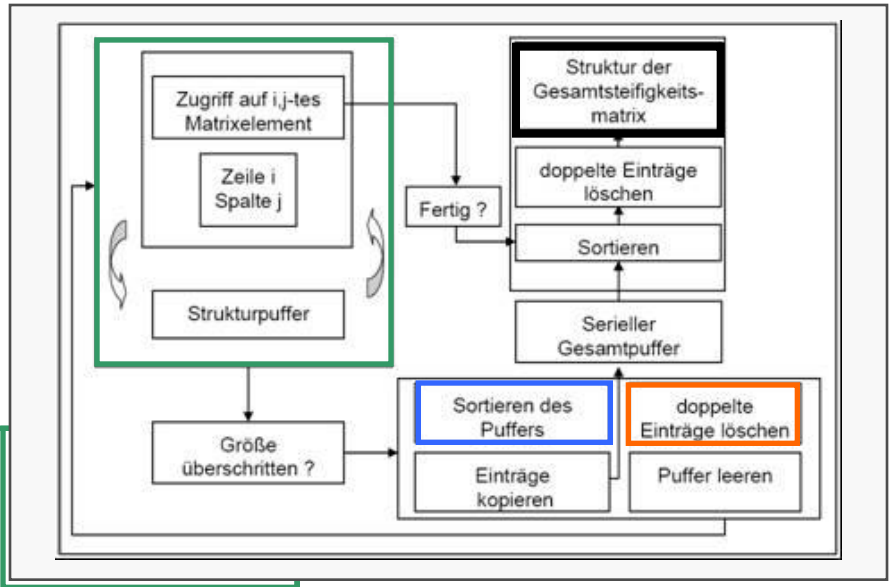
Dipl.-Ing. Bernd Roith, Dipl.-Ing. Martin Zimmermann

Sparsebuilder 1

-Theorie 2-

- **1. Variante: symbolische Kompilation (elementweises Vorgehen)**
- 2. Variante: NCP-Verfahren (Node-Cross-Prinzip)





((4,4) (4,5) (4,6) (1234,128) …)

.....

(…(125,126) (125,125) (123,258) (123,258) (125,125) …)

((4,4) (4,5) (4,6) … (123,258) (123,258) (125,125) (125,125) (125,126) … (1234,128) …)

((4,4) (4,5) (4,6) … (123,258) (125,125) (125,126) (1234,128) …)

Quelle: Dissertation Martin Zimmermann, Universität Bayreuth, 2008

Dipl.-Ing. Bernd Roith, Dipl.-Ing. Martin Zimmermann

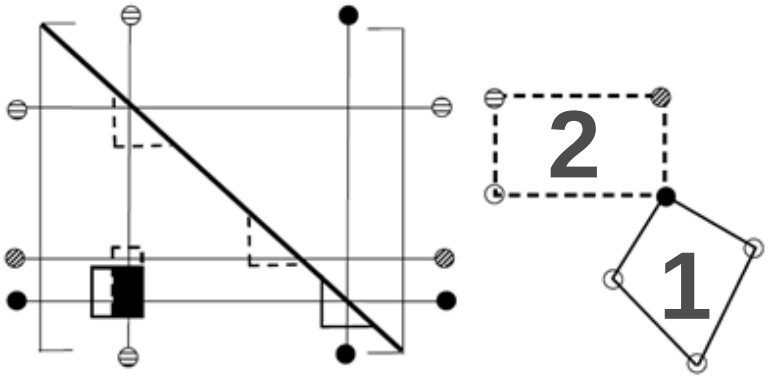
Sparsebuilder 1

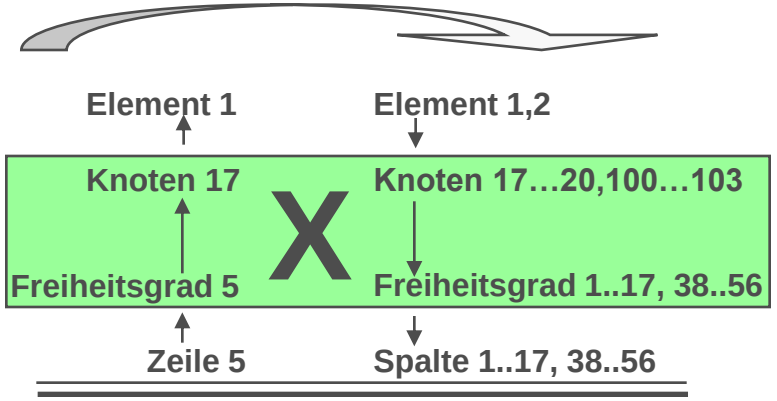
-Theorie 3-

- 1. Variante: symbolische Kompilation **(elementweises Vorgehen)**
- **2. Variante: NCP-Verfahren (Node-Cross-Prinzip) (zeilenweises Vorgehen)**

- Nachteil der symbolischen Kompilation ist das Ausführen der **Sortieroperationen** über **große Felder** und das Ausführen des **gesamten Kompilationsalgorithmus**
- **Speicher- und zeitintensiv**

NCP:
-zeilenweise (Freiheitsgradweise) Erzeugung der Besetzungspositionen der Matrix
-> 1-dimensionaler Charakter des Algorithmus
- Sortieroperationen auf kleine Felder



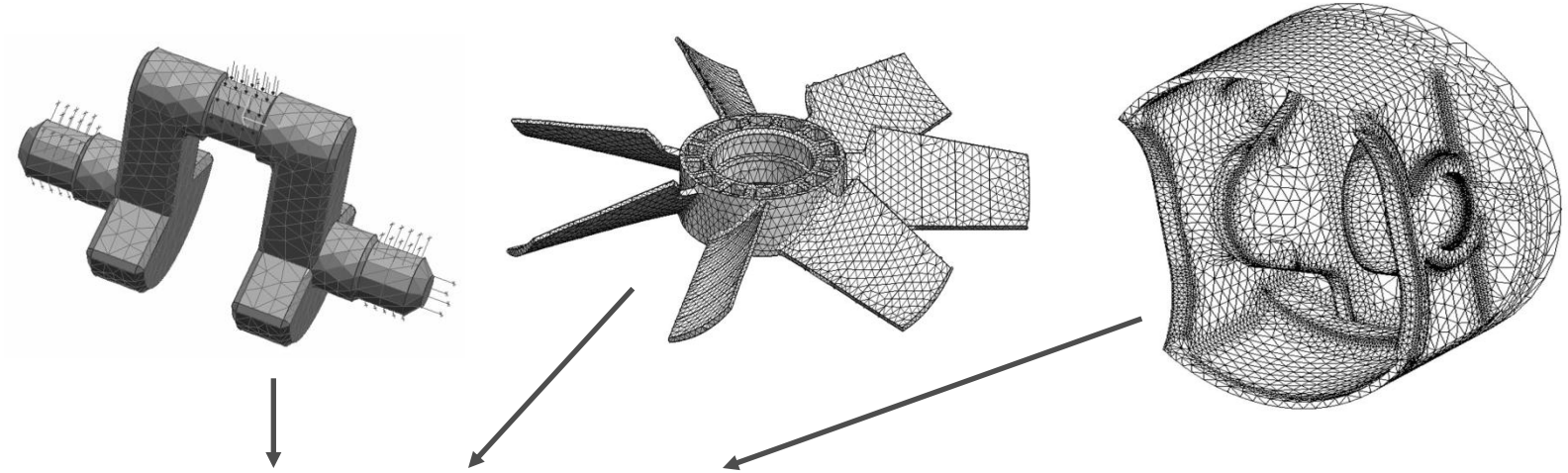


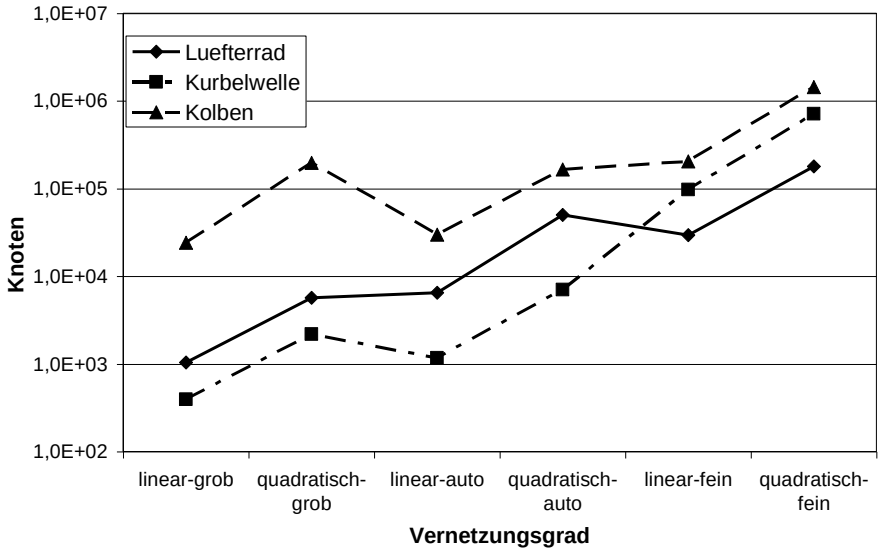
Quelle: Dissertation Martin Zimmermann, Universität Bayreuth, 2008

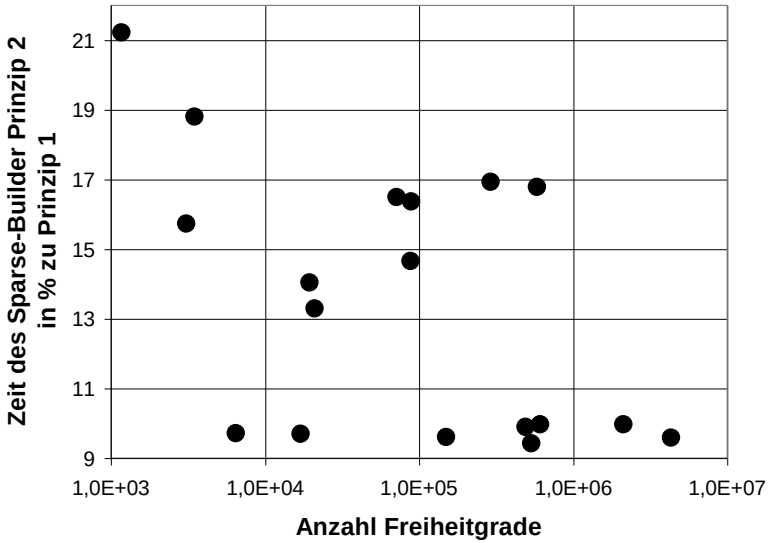
Dipl.-Ing. Bernd Roith, Dipl.-Ing. Martin Zimmermann

Sparsebuilder 2

-Praxistests-





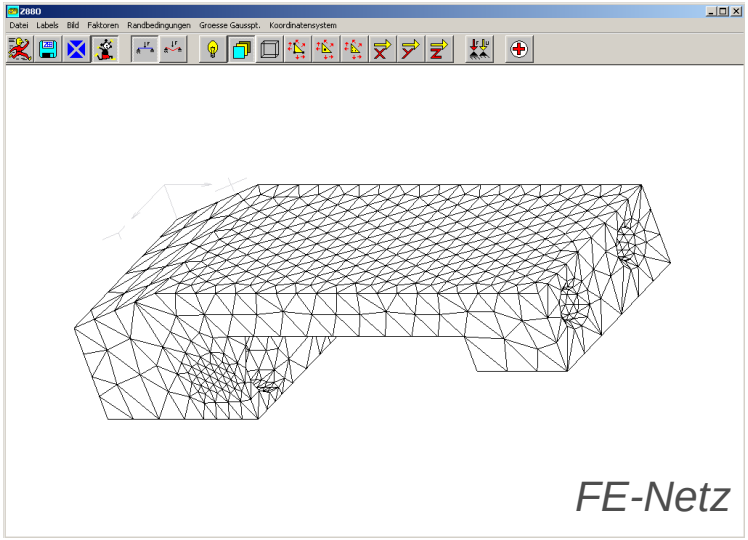


Dipl.-Ing. Bernd Roith, Dipl.-Ing. Martin Zimmermann

Praxisbeispiel

Vergleich Sparsebuilder Z88V12 zu Z88V13

- Z88V12
- Z88V13



FE-Netz

Dipl.-Ing. Bernd Roith, Dipl.-Ing. Martin Zimmermann

$$\underline{\underline{K}} \cdot \underline{u} = \underline{F}$$

Direkte Lösung
z.B. Solver PARDISO

Iterative Lösung
z.B. SOLVER CG

$$\underline{\underline{P}}\,\underline{\underline{K}}\,\underline{\underline{P}}^T$$

SORCG

$$\underline{\underline{K}} = \underline{\underline{D}} - \underline{\underline{E}} - \underline{\underline{F}}$$

$$\underline{u}_{k+1} = \left(\underline{\underline{D}} - \omega \cdot \underline{\underline{E}}\right)^{-1} \cdot \left[\left(\omega \cdot \underline{\underline{F}} + (1-\omega) \cdot \underline{\underline{D}}\right) \cdot \underline{u}_k + \omega \cdot \underline{F}\right]$$

Permutationsmatrix:

- speicher- und algorithmen-optimiertes Zeilen-/ Spaltentauschen

Relaxationsfaktor zur Konvergenzbeschleunigung

$$\underline{\underline{P}}\,\underline{\underline{K}}\,\underline{\underline{P}}^T = \underline{\underline{Q}}\,\underline{\underline{L}}\,\underline{\underline{U}}\,\underline{\underline{R}}$$

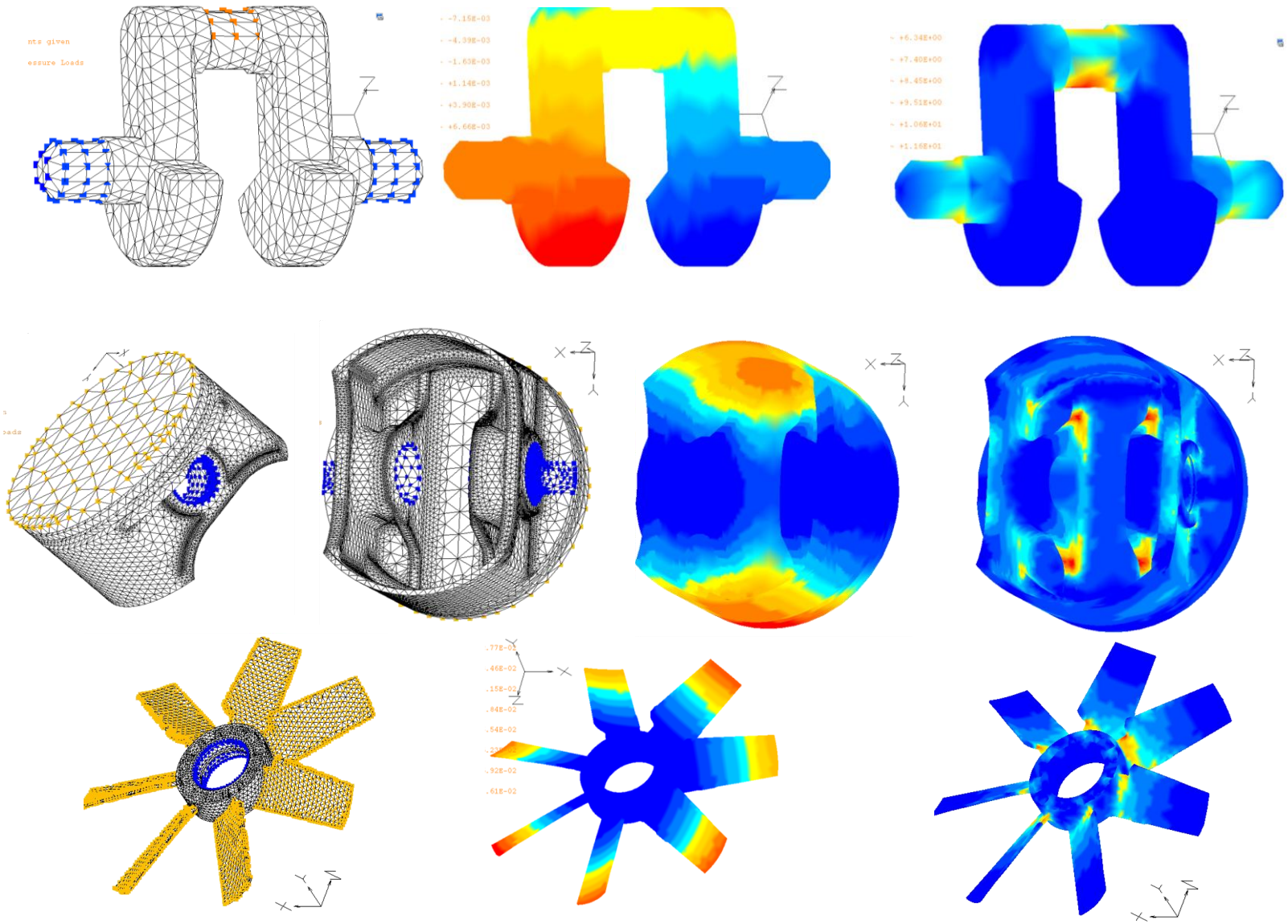
linke und rechte Dreiecksmatrix

pivotisierende Matrizen

Dipl.-Ing. Bernd Roith, Dipl.-Ing. Martin Zimmermann

Finite Elemente Solver 2

- Praxistest Beispiele -



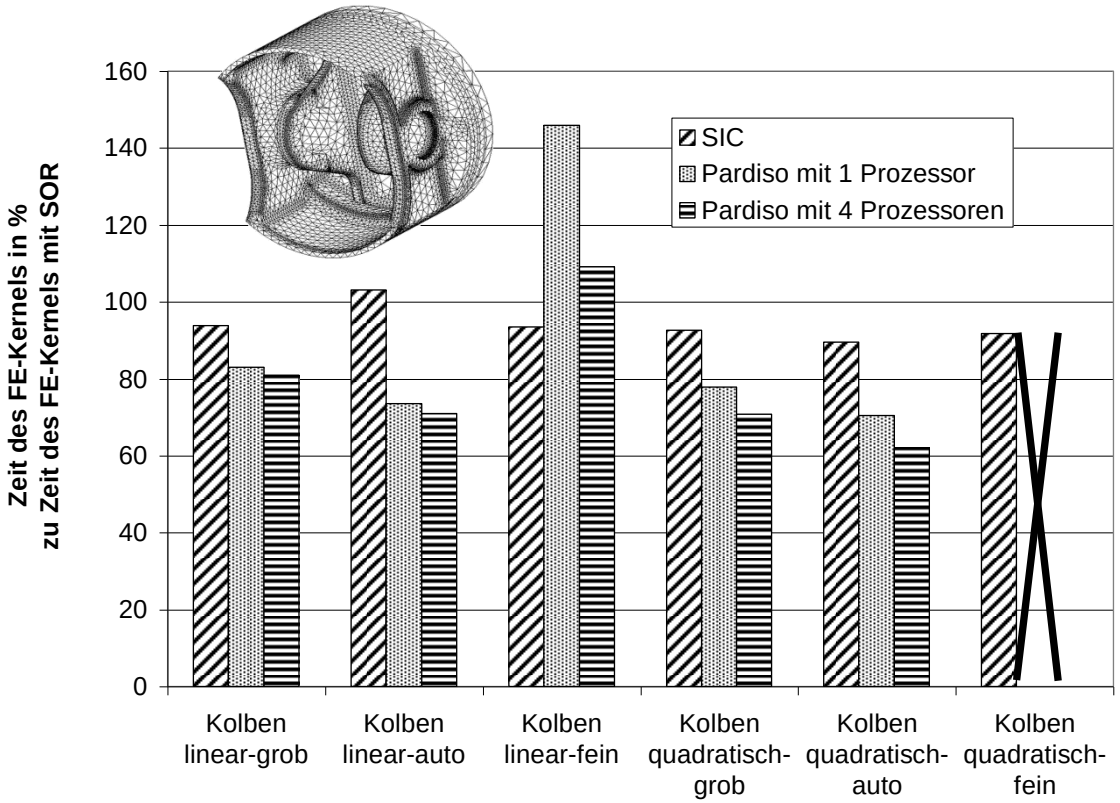
Dipl.-Ing. Bernd Roith, Dipl.-Ing. Martin Zimmermann

Finite Elemente Solver 3

- Praxistest Auswertung -

- iterative Solver: SOR-, SIC-Verfahren und direkte Solver (Single- und Multiprozessor im Vergleich)

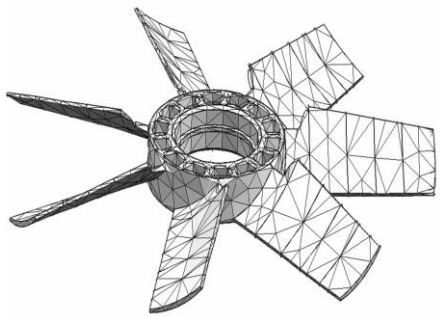
- Mit Parallelisierung lässt sich auf dem FE-Sektor ein Rechenvorgang beschleunigen
- das Verhältnis des Zeitbedarfs von Iterationssolvern und direkten Solvern ist Problemabhängig

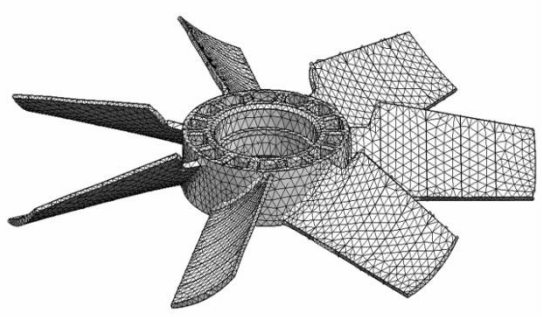


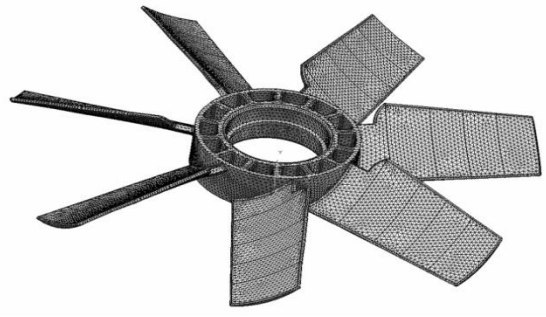
Dipl.-Ing. Bernd Roith, Dipl.-Ing. Martin Zimmermann

Finite Elemente Solver 3

- Einfluss der Diskretisierung bei FE-Modellen -

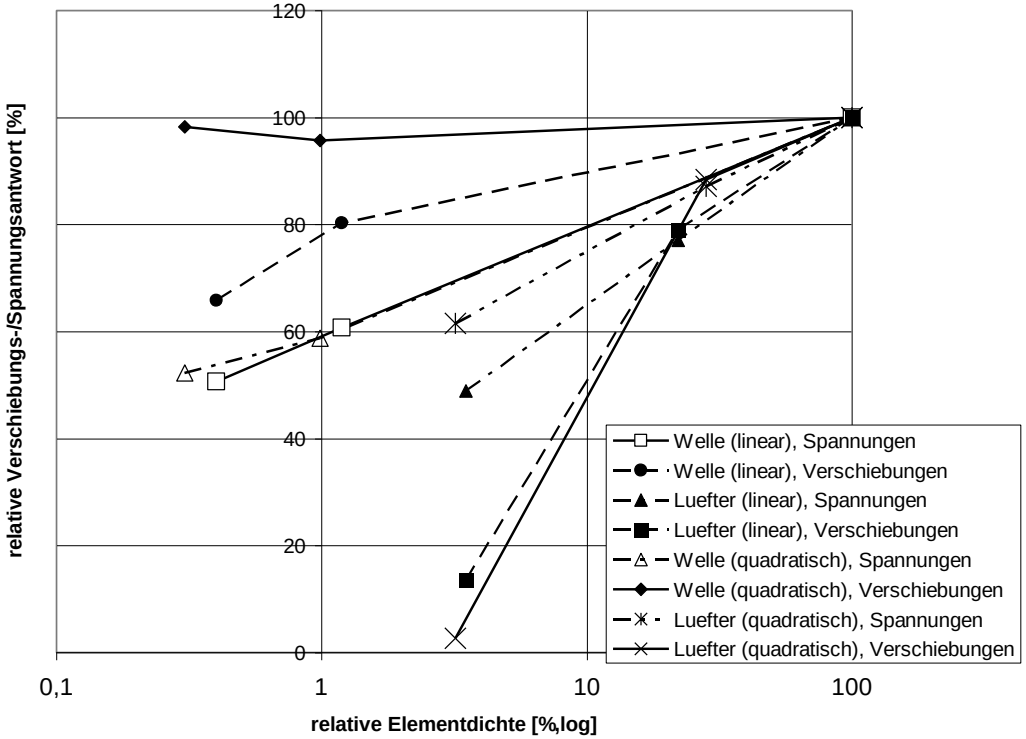






- Diskretisierung beeinflusst maßgeblich den Raum der zur Verfügung stehenden Lösungen
- **zentrales Problem** in der Praxis:

1. **Meist** wird **nur** mit **einer Diskretisierung** gerechnet.
2. Man weiß erst mit **mehreren Lösungen** unterschiedlicher Diskretisierungen, wie **vertrauenswürdig** ein Ergebnis ist.



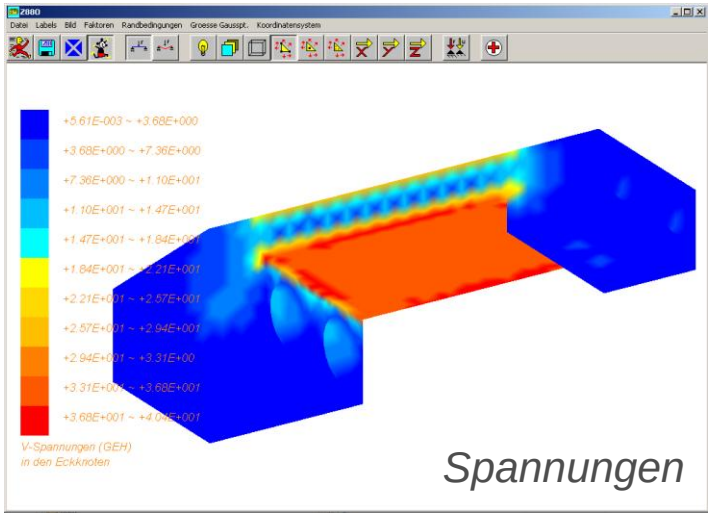
Dipl.-Ing. Bernd Roith, Dipl.-Ing. Martin Zimmermann

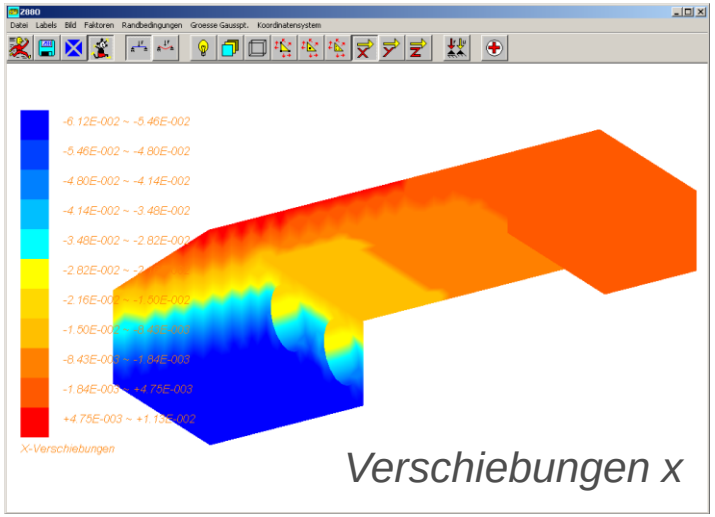
UNIVERSITÄT BAYREUTH

Praxisbeispiel

Finite Elemente Solver Z88V13 - Direkt (Pardiso)

Z88V13





Dipl.-Ing. Bernd Roith, Dipl.-Ing. Martin Zimmermann

Zusammenfassung

- Speicherverwaltung und Lösung großer linearer Gl.-Systeme -

- Eine effiziente **Speicherverwaltung** spielt **für die Berechnung** großer Strukturen eine **entscheidend**e Rolle. Durch die Untersuchung und **Verbesserung der** damit verbundenen **Algorithmen** kann Zeit und Rechenspeicher gespart werden. **Simulationen** als Teil des Produktentwicklungsprozesses werden damit **effizienter durchführbar**.

- Für **große Strukturen** ist die **Sparse-Speichertechnik** zur Realisierung entscheidend.

- **Sparse-Solver** ermöglichen die Berechnung der **großen Gleichungssysteme**. Es ist vorab nicht abzuschätzen mit welcher Solvervariante man **schneller und effektiver zum Ziel** kommt.

- Sowohl **iterative**, als auch **direkte Sparse-Solver** besitzen daher eine **Daseinsberechtigung**.