Iterative Solvers for Large Linear Systems

Part VI: Preconditioning

Andreas Meister

University of Kassel, Department of Analysis and Applied Mathematics

Outline

- Basics of Iterative Methods
- Splitting-schemes
  - Jacobi- u. Gauß-Seidel-scheme
  - Relaxation methods
- Methods for symmetric, positive definite Matrices
  - Method of steepest descent
  - Method of conjugate directions
  - CG-scheme

Outline

- Multigrid Method
  - Smoother, Prolongation, Restriction
  - Twogrid Method and Extension
- Methods for non-singular Matrices
  - GMRES
  - BiCG, CGS and BiCGSTAB
- Preconditioning
  - ILU, IC, GS, SGS, ...

Preconditioning

Goal: Convergence acceleration and stabilization

Condition number

Let A $\in \mathbb{R}^{nxn}$ be non-singular, then

$$cond_a(A) = ||A||_a \, ||A^{-1}||_a$$

is called the condition number of A w.r.t. $||.||_a$

Alternatives:

Let $P_L, P_R \in \mathbb{R}^{nxn}$ be non-singular, then

$$\begin{aligned} P_L \, A \, P_R \, y &= P_L \, b \\ x &= P_R \, y \end{aligned}$$

is called a preconditioned system associated with $Ax = b$.

Left preconditioning: $P_L \neq I$

Right preconditioning: $P_R \neq I$

Two-sided preconditioning: $P_L \neq I \neq P_R$

Preconditioning

Goal: Convergence acceleration and stabilization

Condition number

Let A $\in \mathbb{R}^{nxn}$ be non-singular, then

$$cond_a(A) = ||A||_a \, ||A^{-1}||_a$$

is called the condition number of A w.r.t. $||.||_a$

Alternatives:

Let $P_L, P_R \in \mathbb{R}^{nxn}$ be non-singular, then

$$\begin{aligned} P_L \, A \, P_R \, y &= P_L \, b \\ x &= P_R \, y \end{aligned}$$

is called a preconditioned system associated with $Ax = b$.

Left preconditioning: $P_L \neq I$

Right preconditioning: $P_R \neq I$

Two-sided preconditioning: $P_L \neq I \neq P_R$

Preconditioning

Properties of the condition number:

- $cond(I) \; = \; ||I|| \; ||I^{-1}|| \; = \; 1 \cdot 1 \; = \; 1$
- $cond(A) \; = \; ||A|| \; ||A^{-1}|| \geq ||A \cdot A^{-1}|| \; = \; ||I|| \; = \; 1$
- Let A be normal (d.h. $A^T A \; = \; AA^T$), then

$$cond_2(A) \; = \; \frac{|\lambda_n|}{|\lambda_1|}$$

where λ_1, λ_n are both eigenvalues with the smallest and largest absolut value, respectively.
(A symmetric $\Longrightarrow$ A normal)

Preconditioning

Crucial points:

- P_L, P_R easy to calculated (or more precisely matrix-vector-products with P_L, P_R easy to calculated).
- A sparse $\Longrightarrow$ P_L, P_R sparse.
- $P_L\ A\ P_R \approx I$, such that

$$cond(P_L\ A\ P_R) \approx cond(I) \ll cond(A)$$

as good as possible.

Scaling

Choose: $P_L = D$ or $P_R = D$ with $D = diag\{d_{11}, \cdots, d_{nn}\}$

Possibilities:

- Scaling using the diagonal element

$$d_{ii} = a_{ii}^{-1}, \; i = 1, \cdots, n \qquad (a_{ii} \neq 0 \; \forall i)$$

- Scaling row- or columnwise w.r.t. the 1-Norm

$$d_{ii} = \Big(\sum_{j=1}^{n} |a_{ij}| \Big)^{-1}, \; d_{jj} = \Big(\sum_{i=1}^{n} |a_{ij}| \Big)^{-1}$$

- Scaling row- or columnwise w.r.t. the 2-Norm

$$d_{ii} = \Big(\sum_{j=1}^{n} a_{ij}^2 \Big)^{-\frac{1}{2}}, \; d_{jj} = \Big(\sum_{i=1}^{n} a_{ij}^2 \Big)^{-\frac{1}{2}}$$

- Scaling row- or columnwise w.r.t. the ∞-Norm

$$d_{ii} = \Big(\max_{j=1,\ldots,n} |a_{ij}| \Big)^{-1}, \; d_{jj} = \Big(\max_{i=1,\ldots,n} |a_{ij}| \Big)^{-1}$$

Advantage: $\longrightarrow$ Easy to calculate, low storage requirements

Disadvantage: $\longrightarrow$ Usually very low acceleration of the convergence

Model problem: Convection-Diffusion-Equation

Given: $\beta = (\, cos(\alpha)\,,\, sin(\alpha)\,)\,, \alpha = \frac{\pi}{4}, \; \epsilon = 0.1\,,\; \Omega = (0,1)\; x\; (0,1)$

Sought after: $u \in C^2(\Omega) \cap C(\overline{\Omega})$ with

$$\beta\,\nabla u - \epsilon\,\triangle u = 0 \text{ in } \Omega \quad \text{and} \quad u(x,y) = x^2 + y^2 \text{ on } \partial\,\Omega$$

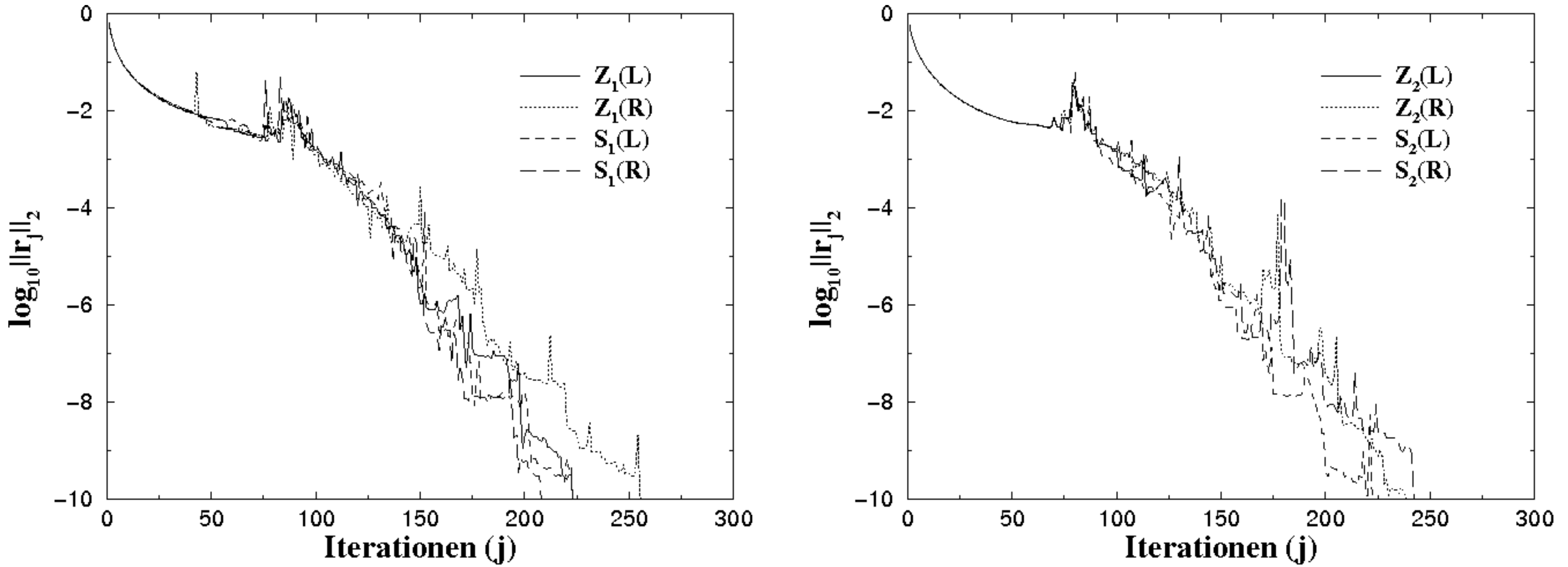
Discretization:

- $N = 100,\; h = x_{i+1} - x_i = y_{i+1} - y_i = \frac{1}{N+1}$
- Central differences for $\triangle u = \partial_x^2 u + \partial_y^2 u$
- One-sided differences for $\nabla u = (\partial_x u, \partial_y u)^T$
- Yields $\boxed{\;A\,u = b\;}$ with

$$\begin{aligned} A &= tridiag\{\, D\,,\; B\,,\; -\epsilon\, I\,\} \in \mathbb{R}^{N^2 x N^2} \\ B &= tridiag\{\, -\epsilon - h\, cos\alpha\,,\; 4\epsilon + h\,(cos\alpha + sin\alpha)\,,\; -\epsilon\} \in \mathbb{R}^{NxN} \\ D &= diag\,\{\, -\epsilon - h\, sin\alpha\} \in \mathbb{R}^{NxN} \end{aligned}$$

Results of different Preconditioners based on scaling

- **L/R:** Left/Right Preconditioning
- **Z/S:** Scaling row- (Z) or columnwise (S)
- **1/2:** Scaling w.r.t. the 1-/2-Norm



- Iterative Solution Method: BiCGSTAB

Splitting-associated preconditioners

Splitting method: $x_{m+1} = B^{-1}(B-A)\,x_m + B^{-1}b$

with $B \approx A$ and $B^{-1}x$ simple to calculate.

Idea: Choose $P_{L/R} = B^{-1}$

Types: $A = D + L + R$

- Jacobi - method $\longrightarrow P = D^{-1}$
- Gauß - Seidel - method $\longrightarrow P = (D + L)^{-1}$
- SOR - method $\longrightarrow P = \omega(D + \omega L)^{-1}$
- Symm. Gauß - Seidel - method $\longrightarrow$
  $P = (D + R)^{-1} D (D + L)^{-1}$
- SSOR - method $\longrightarrow$
  $P = \omega(2 - \omega)(D + \omega R)^{-1} D (D + \omega L)^{-1}$

Advantages:

- No additional staroge requirements
- No calculations
- often good acceleration of the convergence

Splitting-associated preconditioners

Splitting method: $x_{m+1} = B^{-1}(B-A)\,x_m + B^{-1}b$

with $B \approx A$ and $B^{-1}x$ simple to calculate.

Idea: Choose $P_{L/R} = B^{-1}$

Types: $A = D + L + R$

- Jacobi - method $\longrightarrow P = D^{-1}$
- Gauß - Seidel - method $\longrightarrow P = (D + L)^{-1}$
- SOR - method $\longrightarrow P = \omega(D + \omega L)^{-1}$
- Symm. Gauß - Seidel - method $\longrightarrow$
  $P = (D + R)^{-1} D (D + L)^{-1}$
- SSOR - method $\longrightarrow$
  $P = \omega(2 - \omega)(D + \omega R)^{-1} D (D + \omega L)^{-1}$

Advantages:

- No additional staroge requirements
- No calculations
- often good acceleration of the convergence

Splitting-associated preconditioners

Splitting method: $x_{m+1} = B^{-1}(B-A)\,x_m + B^{-1}b$

with $B \approx A$ and $B^{-1}x$ simple to calculate.

Idea: Choose $P_{L/R} = B^{-1}$

Types: $A = D + L + R$

- Jacobi - method $\longrightarrow P = D^{-1}$
- Gauß - Seidel - method $\longrightarrow P = (D + L)^{-1}$
- SOR - method $\longrightarrow P = \omega (D + \omega L)^{-1}$
- Symm. Gauß - Seidel - method $\longrightarrow$
  $P = (D + R)^{-1} D (D + L)^{-1}$
- SSOR - method $\longrightarrow$
  $P = \omega (2 - \omega)(D + \omega R)^{-1} D (D + \omega L)^{-1}$

Advantages:

- No additional staroge requirements
- No calculations
- often good acceleration of the convergence

Splitting-associated preconditioners

Splitting method: $x_{m+1} = B^{-1}(B - A)\,x_m + B^{-1}b$
with $B \approx A$ and $B^{-1}x$ simple to calculate.

Idea: Choose $P_{L/R} = B^{-1}$

Types: $A = D + L + R$

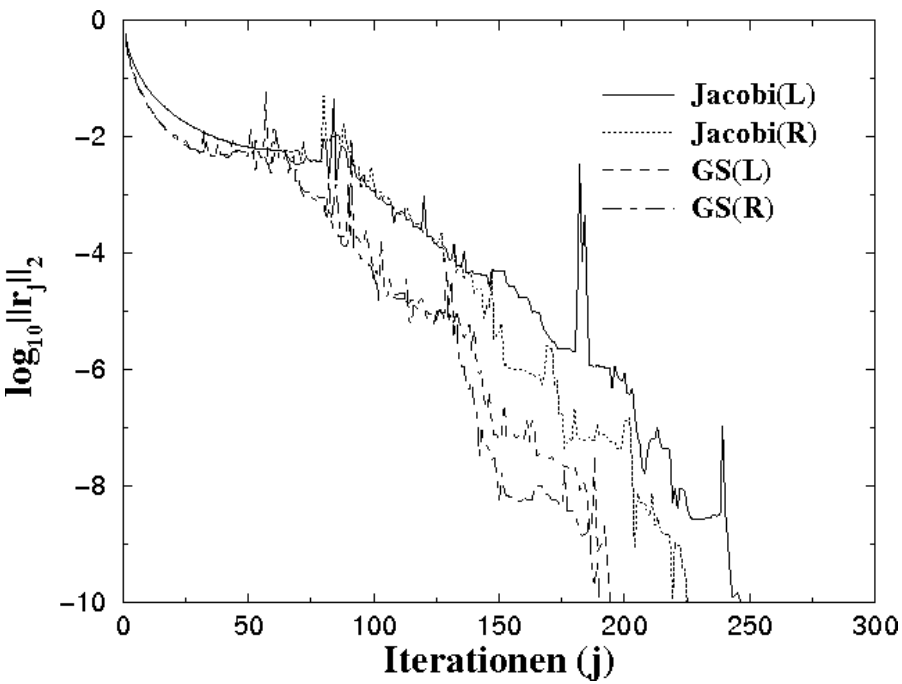
- Jacobi - method $\longrightarrow P = D^{-1}$
- Gauß - Seidel - method $\longrightarrow P = (D + L)^{-1}$
- SOR - method $\longrightarrow P = \omega (D + \omega L)^{-1}$
- Symm. Gauß - Seidel - method $\longrightarrow$
  $P = (D + R)^{-1} D (D + L)^{-1}$
- SSOR - method $\longrightarrow$
  $P = \omega (2 - \omega)(D + \omega R)^{-1} D (D + \omega L)^{-1}$

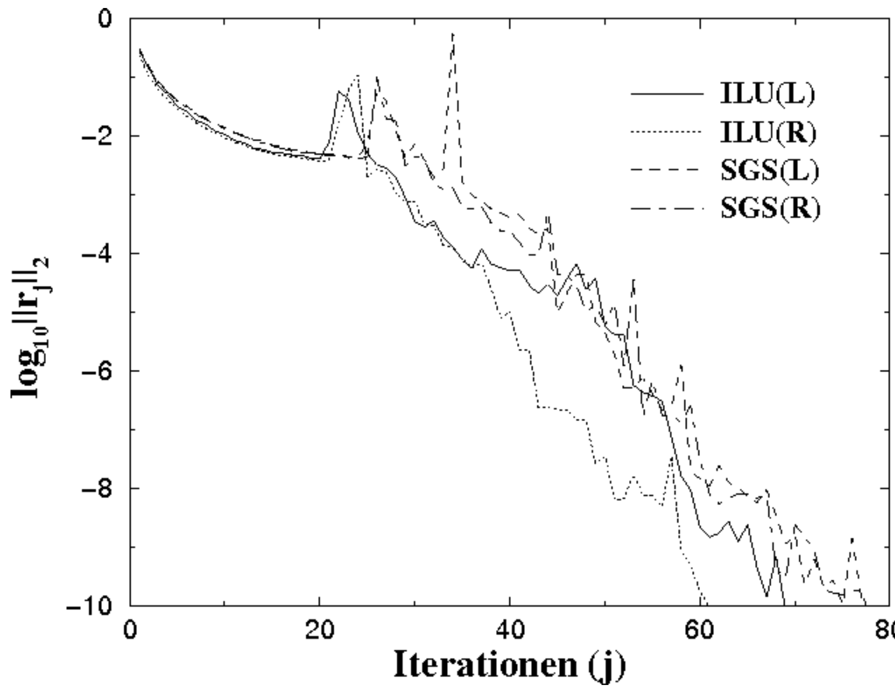
Advantages:

- No additional staroge requirements
- No calculations
- often good acceleration of the convergence

Results of Splitting-associated preconditioner

- **L/R:** Left/Right Preconditioning





- Iterative Solution Method: BiCGSTAB

Incomplete LU-Factorization

Standard Gaussian elimination yields $A = L \cdot R$

Problems w.r.t. large, sparse matrices:

- Huge computational effort and storage requirements
- Rounding errors

Idea:

- Caclculate $A = L \cdot R + F$, with
  - $r_{ii} = 1, i = 1, ..., n$
  - $l_{ij} = r_{ij} = 0$, if $a_{ij} = 0$
  - $l_{ij} = r_{ji} = 0$, if $i < j$
  - $(L \cdot R)_{ij} = a_{ij}$, if $a_{ij} \neq 0$

Properties:

- $f_{ij} = 0$, if $a_{ij} \neq 0$
- Storage of A = Complete storage of L and R
- Low computational effort compared to standard Gaussian elimination

Incomplete LU-Factorization

Standard Gaussian elimination yields $A = L \cdot R$

Problems w.r.t. large, sparse matrices:

- Huge computational effort and storage requirements
- Rounding errors

Idea:

- Caclculate $A = L \cdot R + F$, with
  - $r_{ii} = 1, i = 1, ..., n$
  - $l_{ij} = r_{ij} = 0$, if $a_{ij} = 0$
  - $l_{ij} = r_{ji} = 0$, if $i < j$
  - $(L \cdot R)_{ij} = a_{ij}$, if $a_{ij} \neq 0$

Properties:

- $f_{ij} = 0$, if $a_{ij} \neq 0$
- Storage of A $=$ Complete storage of L and R
- Low computational effort compared to standard Gaussian elimination

Incomplete LU-Factorization

Standard Gaussian elimination yields $A = L \cdot R$

Problems w.r.t. large, sparse matrices:

- Huge computational effort and storage requirements
- Rounding errors

Idea:

- Caclculate $A = L \cdot R + F$, with
  - $r_{ii} = 1, i = 1, ..., n$
  - $l_{ij} = r_{ij} = 0$, if $a_{ij} = 0$
  - $l_{ij} = r_{ji} = 0$, if $i < j$
  - $(L \cdot R)_{ij} = a_{ij}$, if $a_{ij} \neq 0$

Properties:

- $f_{ij} = 0$, if $a_{ij} \neq 0$
- Storage of A = Complete storage of L and R
- Low computational effort compared to standard Gaussian elimination

Incomplete LU-Factorization

Standard Gaussian elimination yields $A = L \cdot R$

Problems w.r.t. large, sparse matrices:

- Huge computational effort and storage requirements
- Rounding errors

Idea:

- Caclculate $A = L \cdot R + F$, with
  - $r_{ii} = 1, i = 1, ..., n$
  - $l_{ij} = r_{ij} = 0$, if $a_{ij} = 0$
  - $l_{ij} = r_{ji} = 0$, if $i < j$
  - $(L \cdot R)_{ij} = a_{ij}$, if $a_{ij} \neq 0$

Properties:

- $f_{ij} = 0$, if $a_{ij} \neq 0$
- Storage of A $=$ Complete storage of L and R
- Low computational effort compared to standard Gaussian elimination

Incomplete LU-Factorization

Procedure: $a_{ki} = (L \cdot R)_{ki}$ for $a_{ki} \neq 0$ directly yields

$$a_{ki} = \sum_{m=1}^{n} l_{km}\, r_{mi} \overset{r_{mi}=0,\ m>i}{=} \sum_{m=1}^{i} l_{km}\, r_{mi} \overset{r_{ii}=1}{=} \sum_{m=1}^{i-1} l_{km}\, r_{mi} + l_{ki}$$

Concerning the i-th coloumn of L one gets

$$l_{ki} = a_{ki} - \sum_{m=1}^{i-1} l_{km}\, r_{mi},\ k = i, \ldots, n,\ \text{mit } a_{ki} \neq 0$$

Analogously, the i-th row of R is given by

$$r_{ik} = \frac{1}{l_{ii}}\Big(a_{ik} - \sum_{m=1}^{i-1} l_{im}\, r_{mk}\Big), k = i+1, \ldots, n,\ \text{mit } a_{ik} \neq 0$$

Preconditioner: $P = R^{-1}\, L^{-1}$

Advantage: Good improvement of the covergence

Disadvantage: Low additional computational effort and storage requirements

Incomplete LU-Factorization

Procedure: $a_{ki} = (L \cdot R)_{ki}$ for $a_{ki} \neq 0$ directly yields

$$a_{ki} = \sum_{m=1}^{n} l_{km}\, r_{mi} \overset{r_{mi}=0,\ m>i}{=} \sum_{m=1}^{i} l_{km}\, r_{mi} \overset{r_{ii}=1}{=} \sum_{m=1}^{i-1} l_{km}\, r_{mi} + l_{ki}$$

Concerning the i-th coloumn of L one gets

$$l_{ki} = a_{ki} - \sum_{m=1}^{i-1} l_{km}\, r_{mi},\ k = i, \ldots, n,\ \text{mit } a_{ki} \neq 0$$

Analogously, the i-th row of R is given by

$$r_{ik} = \frac{1}{l_{ii}}\left(a_{ik} - \sum_{m=1}^{i-1} l_{im}\, r_{mk}\right), k = i+1, \ldots, n,\ \text{mit } a_{ik} \neq 0$$

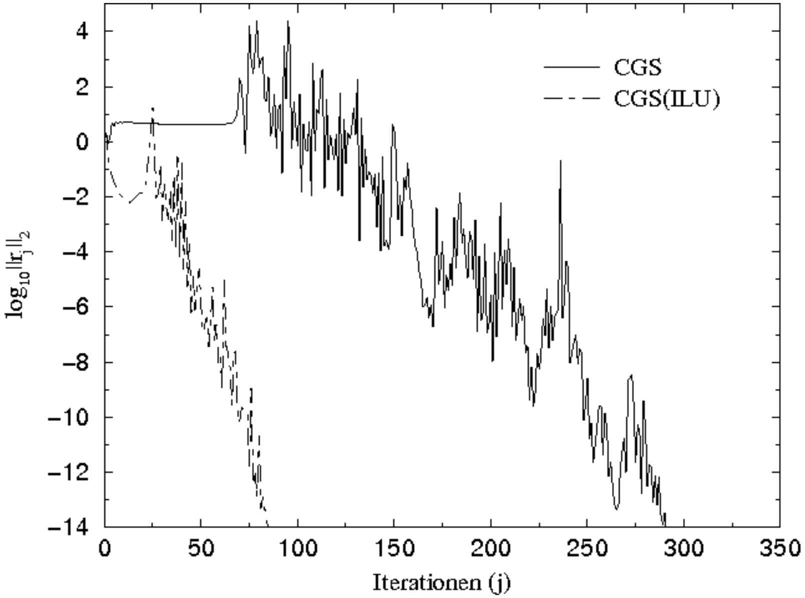
Preconditioner: $P = R^{-1}\, L^{-1}$

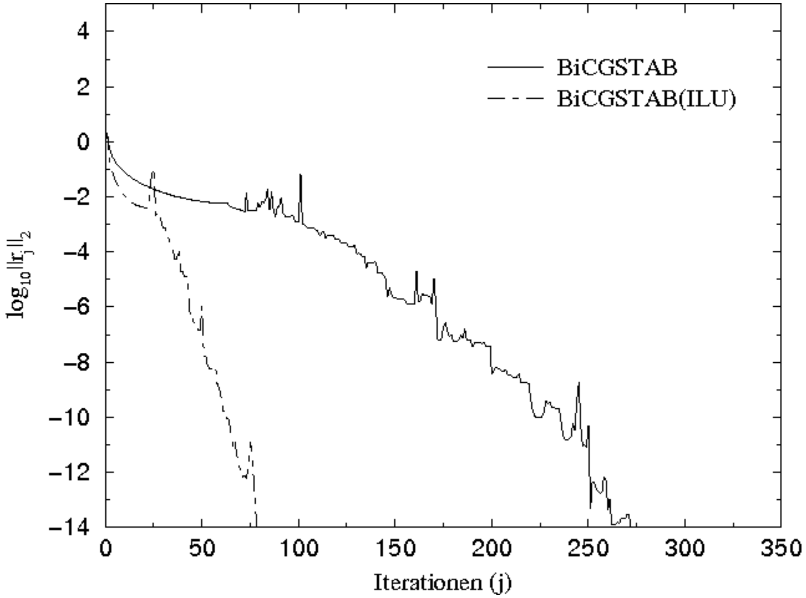
Advantage: Good improvement of the covergence

Disadvantage: Low additional computational effort and storage requirements

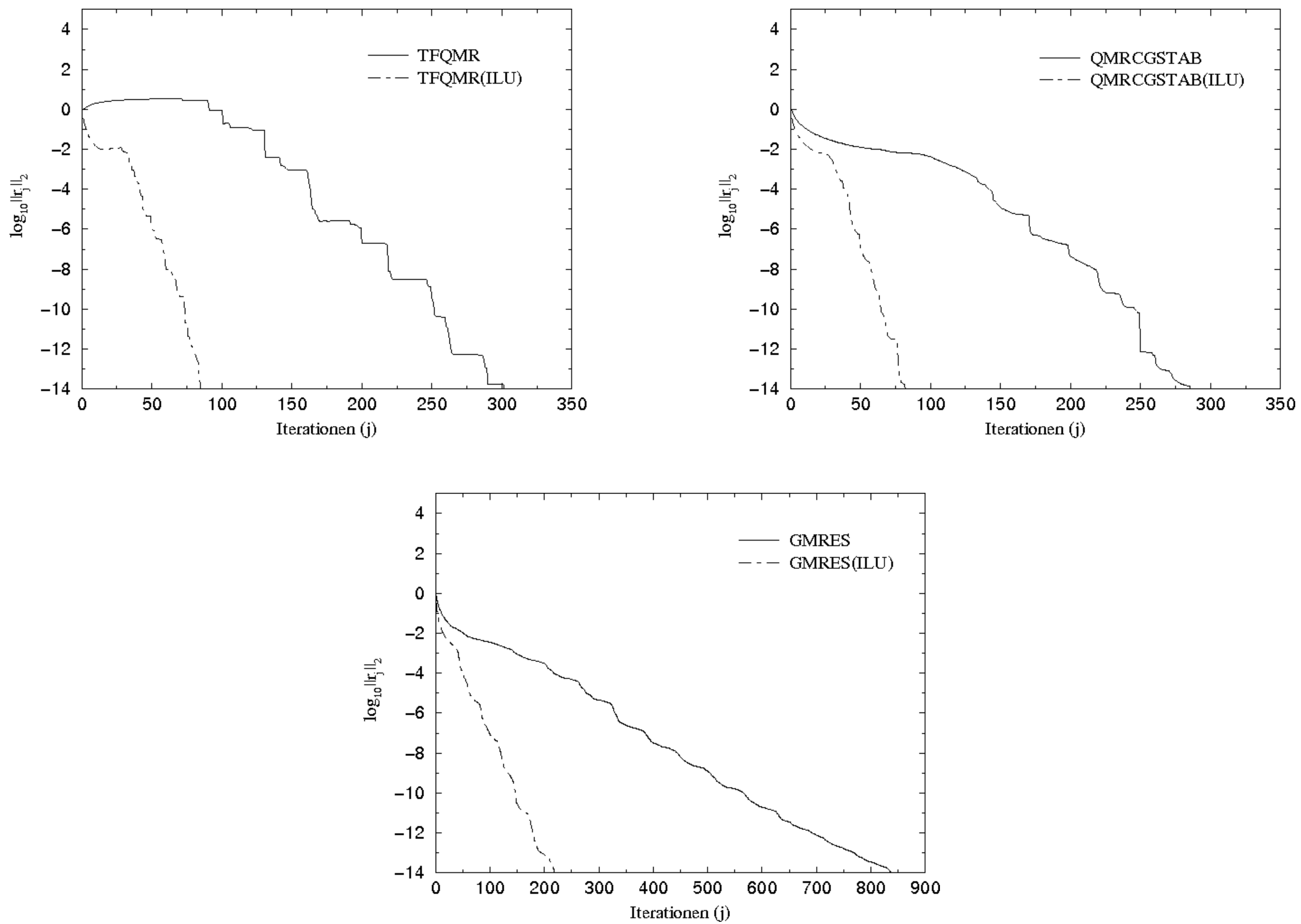
Results of the ILU-Factorization

- **L/R:** Convection-Diffusion-Equation: $\epsilon = 0.01$





Results of the ILU-Factorization



Preconditioned CG-scheme (PCG)

Given: $\quad A\,x = b, \quad$ where A symmetric, positive definite

Form of the PCG-scheme:

$$\underbrace{P_L\,A\,P_R}_{=A^p}\,x^p = P_L\,b$$

$$x = P_R\,x^p$$

Assumption concerning the applicability of CG:

A^p symmetric, positive definite

Preconditioned CG-scheme (PCG)

Proceeding:

Employ $P_R = P_L^T$ to obtain

(a)
$$\begin{aligned}(A^p)^T &= (P_L A P_L^T)^T &&= (P_L^T)^T A P_L^T \\ &= P_L A P_L^T &&= A^p\end{aligned}$$

$\longrightarrow A^p$ symmetric

(b) Since $y = P_L^T x \neq 0$ for all $x \neq 0$ one gets

$$\begin{aligned}(x, A^p x) &= x^T A^p x &&= x^T P_L A P_L^T x \\ &= (P_L^T x)^T A P_L^T x &&= y^T A y &&> 0\end{aligned}$$

$\longrightarrow A^p$ positive definite

Incomplete Cholesky-Factorization:

- Procedure is equivalent to the ILU approach
- Benefit from the symmetry of A
- Form

$$A = L\,L^T + F$$

- Symmetric Preconditioning

$$\begin{aligned} P_L\, A\, P_R\, x^{p} &= P_L\, b \\ x &= P_R\, x^{p} \end{aligned}$$

with

$$P_L = L^{-1} \text{ and } P_R = L^{-T}$$

Preconditioners for PCG

Symmetric Gauß-Seidel method:

- Classical splitting in terms of $A = L + D + R$
- Basic formulation of the preconditioner

$$P_{SGS} = (\ D\ +\ R\ )^{-1}\ D\ (\ D\ +\ L\ )^{-1}$$

- Principles of symmetric preconditioners

$$A \text{ symmetric} \Longrightarrow D + R = D + L^T = (D + L)^T$$

$$A \text{ positive definite} \Longrightarrow a_{ii} = (e_i, Ae_i) > 0,$$

$$e_i = \text{i-th canonical basis vector}$$

$$\Longrightarrow D = diag\{a_{11}, ..., a_{nn}\} = D^{1/2}\ D^{1/2}$$

$$\text{with } D^{1/2} = diag\{a_{11}^{1/2}, ..., a_{nn}^{1/2}\}$$

- Determination of the symmetric preconditioner

$$\begin{aligned} P_{SGS} &= (\ D\ +\ L\ )^{-T}\ D^{1/2}\ D^{1/2}\ (\ D\ +\ L\ )^{-1} \\ &= \underbrace{(D^{1/2}\ (\ D\ +\ L\ )^{-1})^T}_{P_L :=}\ \underbrace{D^{1/2}\ (\ D\ +\ L\ )^{-1}}_{P_R :=} = P_L\ P_R \end{aligned}$$

Preconditioners for PCG

Symmetric Gauß-Seidel method:

- Classical splitting in terms of $A = L + D + R$
- Basic formulation of the preconditioner

$$P_{SGS} = (\, D \; + \; R \,)^{-1} \, D \, (\, D \; + \; L \,)^{-1}$$

- Principles of symmetric preconditioners

$$A \text{ symmetric} \Longrightarrow D + R = D + L^T = (D + L)^T$$

$$A \text{ positive definite} \Longrightarrow a_{ii} = (e_i, Ae_i) > 0,$$

$$e_i = \text{i-th canonical basis vector}$$

$$\Longrightarrow D = diag\{a_{11}, ..., a_{nn}\} = D^{1/2} \, D^{1/2}$$

$$\text{with } D^{1/2} = diag\{a_{11}^{1/2}, ..., a_{nn}^{1/2}\}$$

- Determination of the symmetric preconditioner

$$P_{SGS} = (\, D \; + \; L \,)^{-T} \, D^{1/2} \, D^{1/2} \, (\, D \; + \; L \,)^{-1}$$

$$= \underbrace{(D^{1/2} \, (\, D \; + \; L \,)^{-1})^T}_{P_L :=} \, \underbrace{D^{1/2} \, (\, D \; + \; L \,)^{-1}}_{P_R :=} = P_L \, P_R$$

Preconditioners for PCG

Symmetric Gauß-Seidel method:

- Classical splitting in terms of $A = L + D + R$
- Basic formulation of the preconditioner

$$P_{SGS} = (\ D\ +\ R\ )^{-1}\ D\ (\ D\ +\ L\ )^{-1}$$

- Principles of symmetric preconditioners

$$A \text{ symmetric} \Longrightarrow D + R = D + L^T = (D + L)^T$$

$$A \text{ positive definite} \Longrightarrow a_{ii} = (e_i, Ae_i) > 0,$$

$$e_i = \text{i-th canonical basis vector}$$

$$\Longrightarrow D = diag\{a_{11}, ..., a_{nn}\} = D^{1/2}\ D^{1/2}$$

$$\text{with } D^{1/2} = diag\{a_{11}^{1/2}, ..., a_{nn}^{1/2}\}$$

- Determination of the symmetric preconditioner

$$P_{SGS} = (\ D\ +\ L\ )^{-T}\ D^{1/2}\ D^{1/2}\ (\ D\ +\ L\ )^{-1}$$

$$= \underbrace{(D^{1/2}\ (\ D\ +\ L\ )^{-1})^T}_{P_L :=}\ \underbrace{D^{1/2}\ (\ D\ +\ L\ )^{-1}}_{P_R :=} = P_L\ P_R$$

Preconditioners for PCG

Symmetric Gauß-Seidel method:

- Classical splitting in terms of $A = L + D + R$
- Basic formulation of the preconditioner

$$P_{SGS} = (\,D \;+\; R\,)^{-1}\,D\,(\,D \;+\; L\,)^{-1}$$

- Principles of symmetric preconditioners

$$A \text{ symmetric} \Longrightarrow D + R = D + L^T = (D + L)^T$$

$$A \text{ positive definite} \Longrightarrow a_{ii} = (e_i, Ae_i) > 0,$$
$$e_i = \text{i-th canonical basis vector}$$

$$\Longrightarrow D = diag\{a_{11}, ..., a_{nn}\} = D^{1/2}\,D^{1/2}$$
$$\text{with } D^{1/2} = diag\{a_{11}^{1/2}, ..., a_{nn}^{1/2}\}$$

- Determination of the symmetric preconditioner

$$P_{SGS} = (\,D \;+\; L\,)^{-T}\,D^{1/2}\,D^{1/2}\,(\,D \;+\; L\,)^{-1}$$
$$= \underbrace{(D^{1/2}\,(\,D \;+\; L\,)^{-1})^T}_{P_L :=}\;\underbrace{D^{1/2}\,(\,D \;+\; L\,)^{-1}}_{P_R :=} = P_L\,P_R$$

Preconditioners for PCG

Symmetric Gauß-Seidel method:

- Classical splitting in terms of $A = L + D + R$
- Basic formulation of the preconditioner
$$P_{SGS} = (\, D \;+\; R\,)^{-1}\, D\, (\, D \;+\; L\,)^{-1}$$
- Principles of symmetric preconditioners

$$A \text{ symmetric} \Longrightarrow D + R = D + L^T = (D + L)^T$$

$$A \text{ positive definite} \Longrightarrow a_{ii} = (e_i, Ae_i) > 0,$$
$$e_i = \text{i-th canonical basis vector}$$

$$\Longrightarrow D = diag\{a_{11}, ..., a_{nn}\} = D^{1/2}\, D^{1/2}$$
$$\text{with } D^{1/2} = diag\{a_{11}^{1/2}, ..., a_{nn}^{1/2}\}$$

- Determination of the symmetric preconditioner

$$P_{SGS} = (\, D \;+\; L\,)^{-T}\, D^{1/2}\, D^{1/2}\, (\, D \;+\; L\,)^{-1}$$
$$= \underbrace{(D^{1/2}\, (\, D \;+\; L\,)^{-1})^T}_{P_L :=}\; \underbrace{D^{1/2}\, (\, D \;+\; L\,)^{-1}}_{P_R :=} = P_L\, P_R$$

Preconditioners in practical applications

Applications:

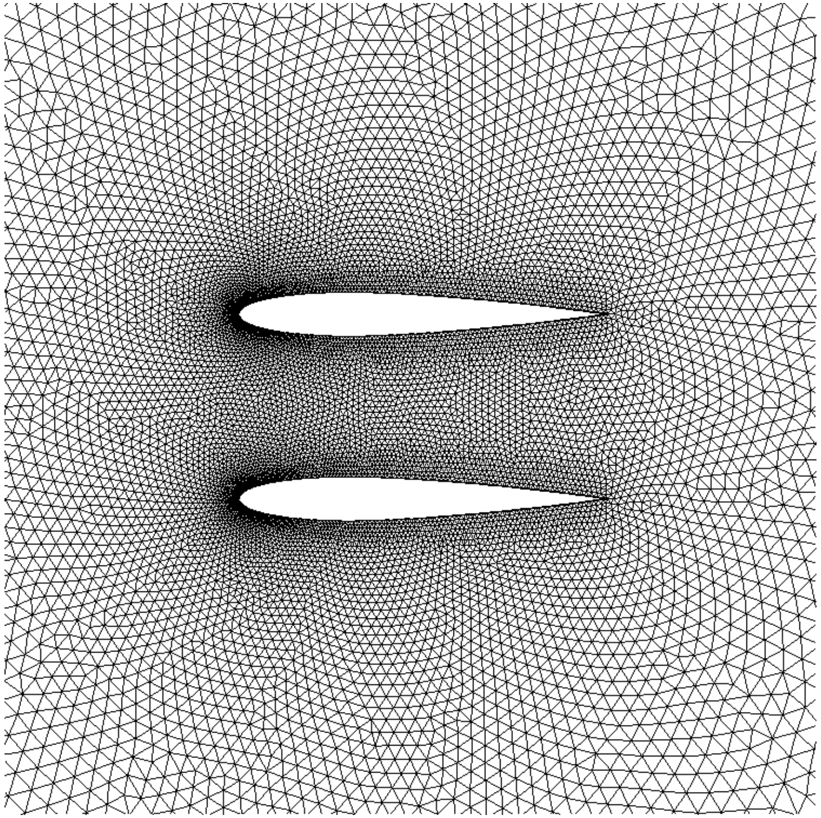
- Simulation of inviscid fluid flow
  Euler equations
- Simulation of viscous fluid flow
  Navier-Stokes equations

Numerical method:

- Finite-Volumen method using unstructered grids
- Implicit time integration scheme
  - Solution of a (non-)linear system of equations $Ax = b$ each timestep
  - Properties of the matrix $A \in \mathbb{R}^{nxn}$
    ◦ large: $n \approx 10^4 - 10^6$
    ◦ sparse ($\approx 0.1\%$)
    ◦ unsymmetric
    ◦ badly conditioned

Ma$=$ 0.55, Angle of attack 6°, inviscid,

Triangulation: 13577 points



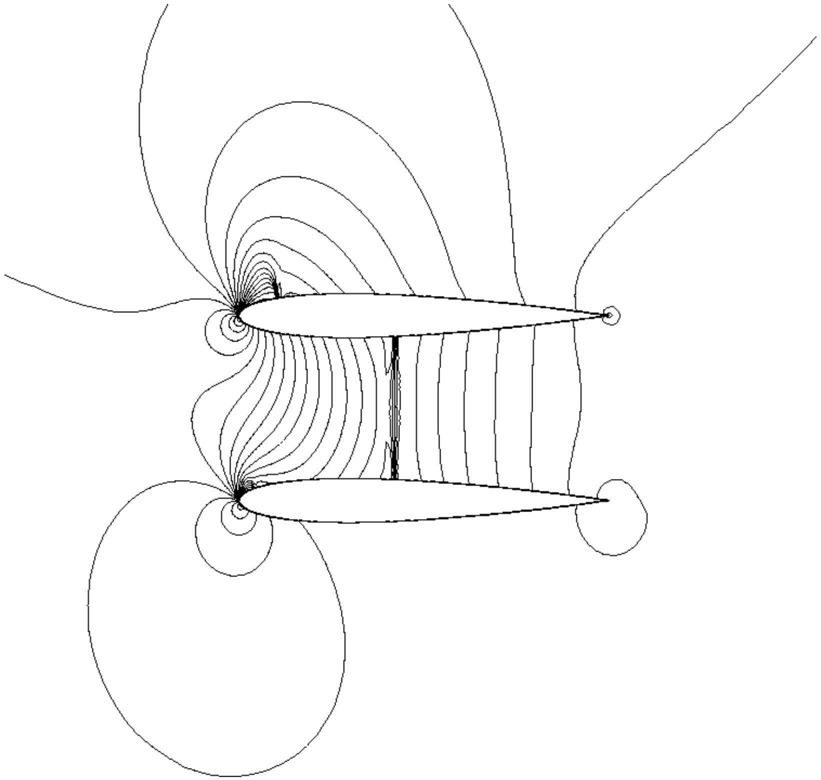
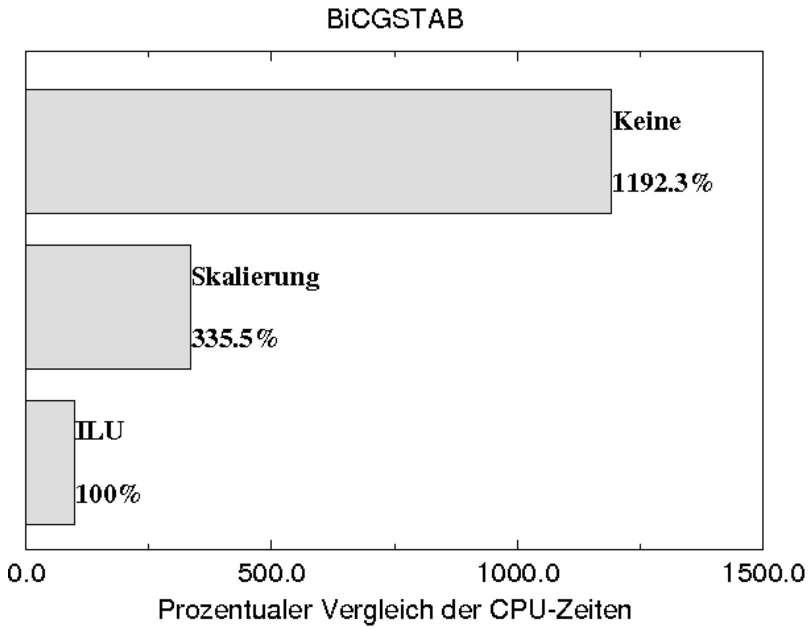
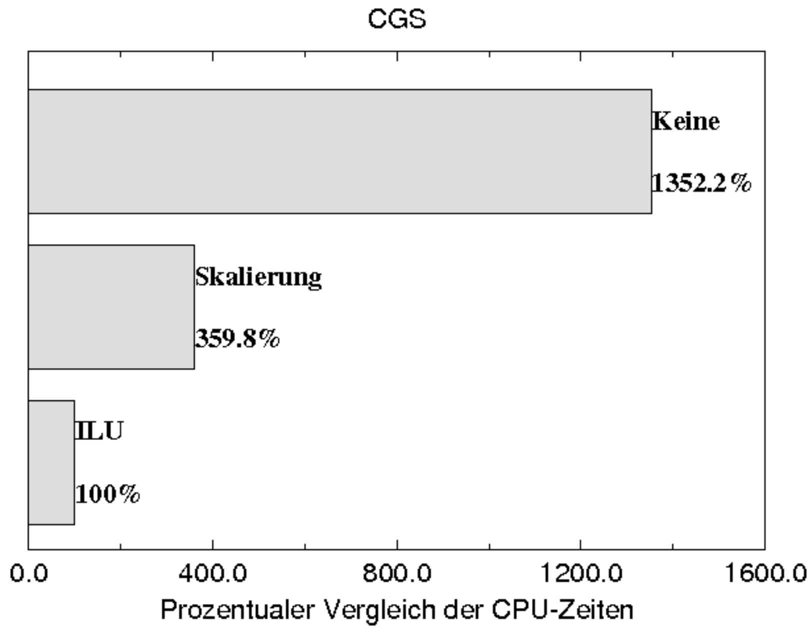
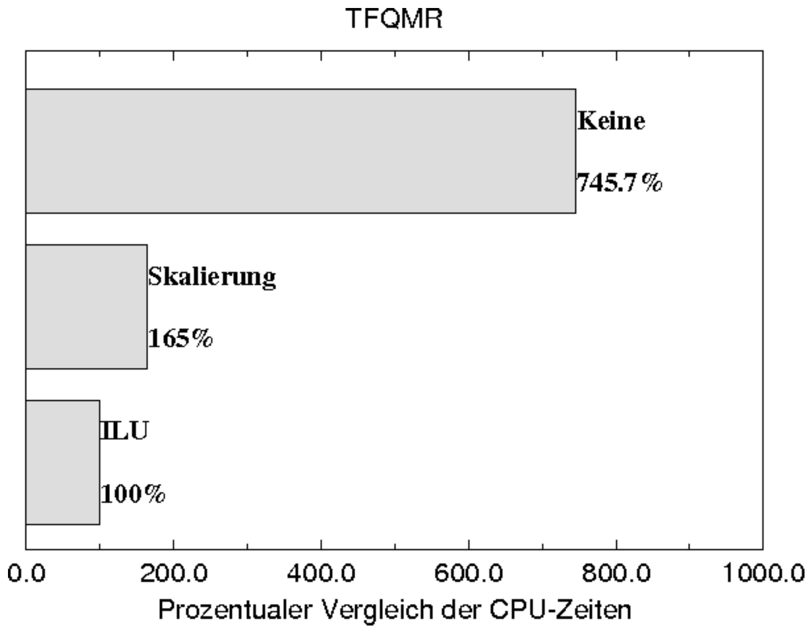


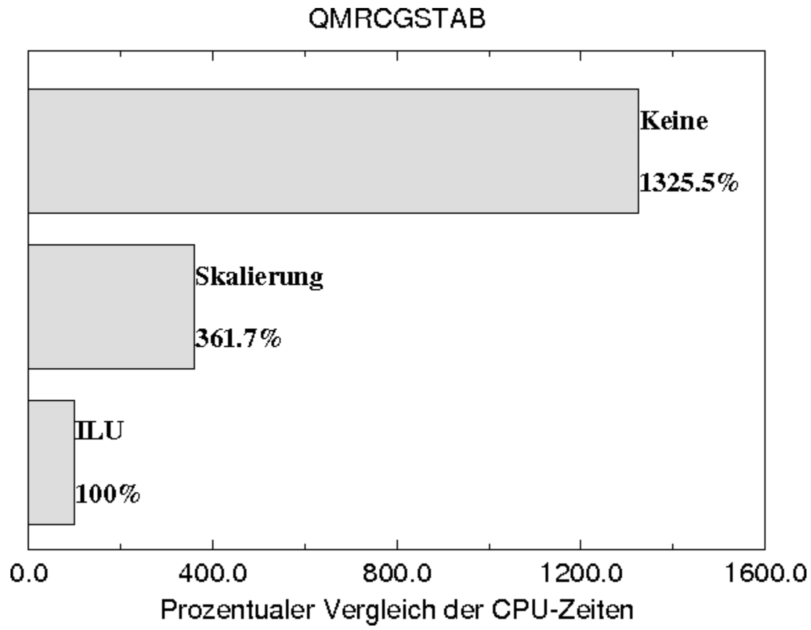
Fig.: Triangulation and isolines of the density distribution

BiNACA0012-profil









NACA0012-profil

Re$=$ 500, Ma$=$ 0.85, Angle of attack 0°, Triangulation: 8742 points

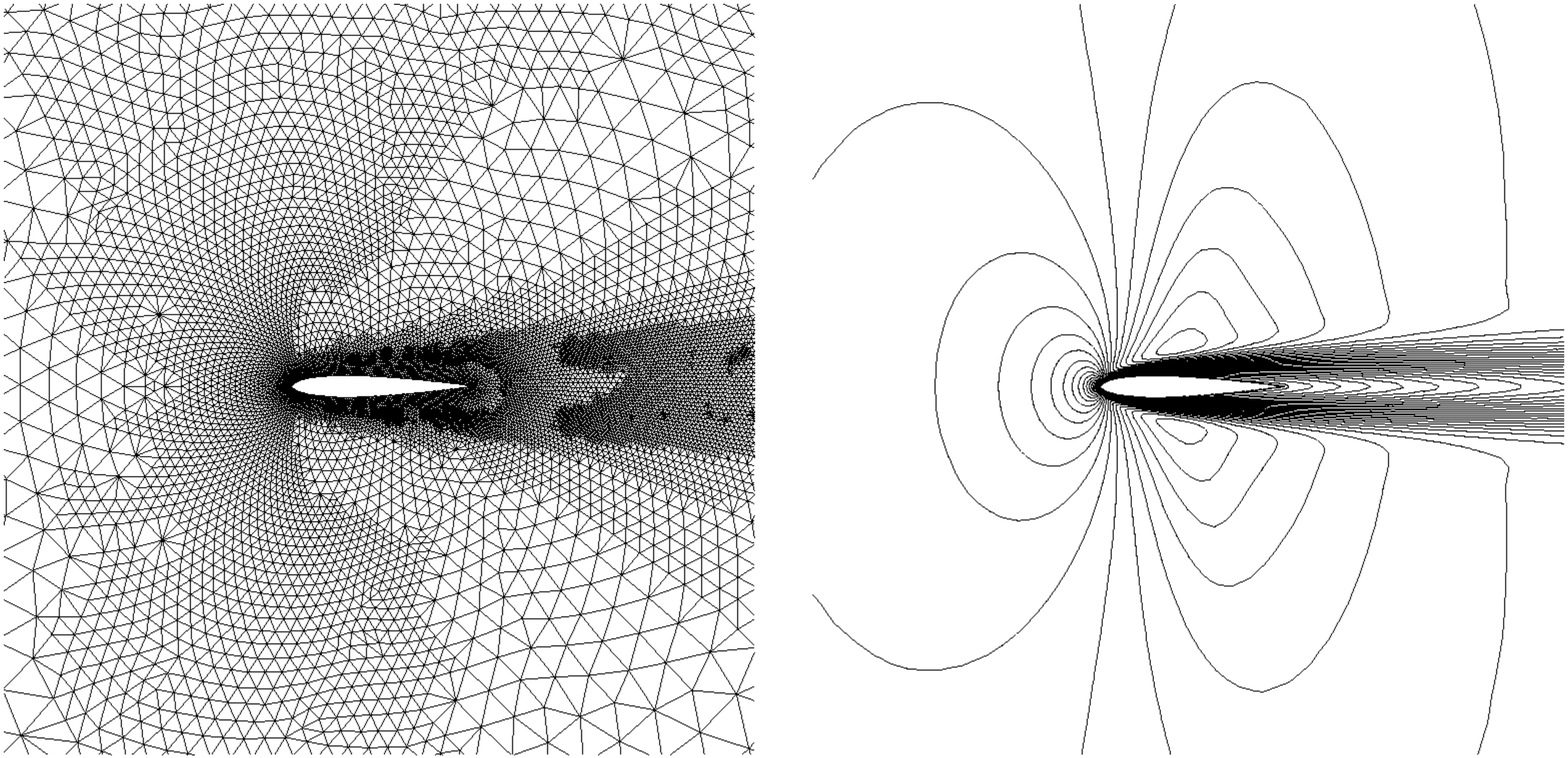
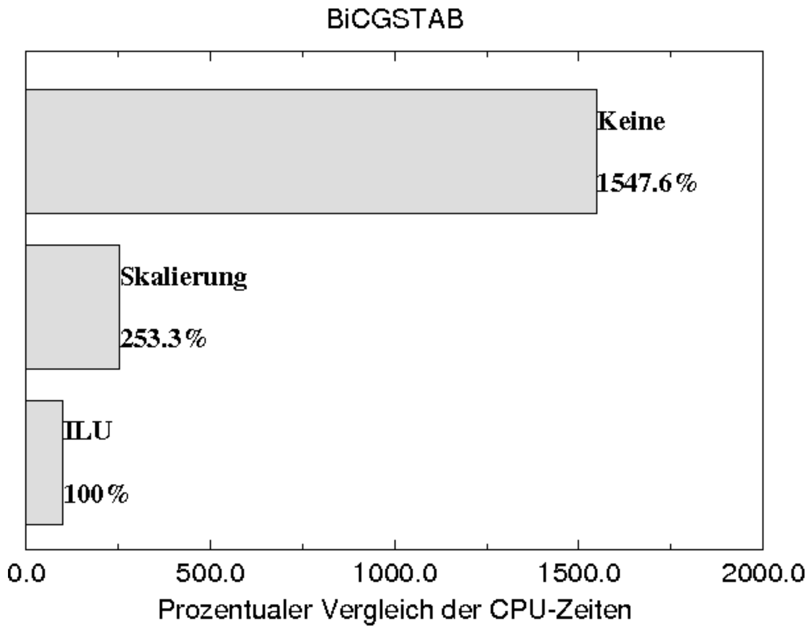
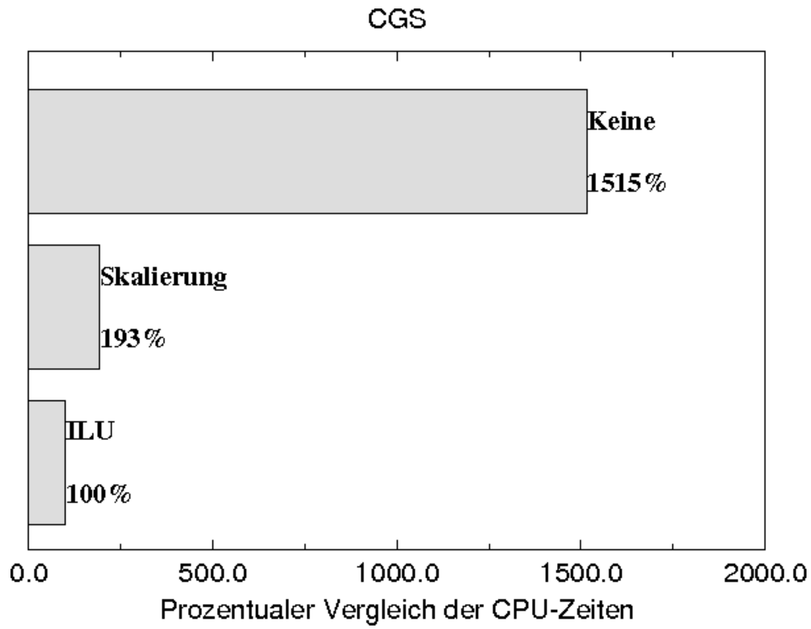
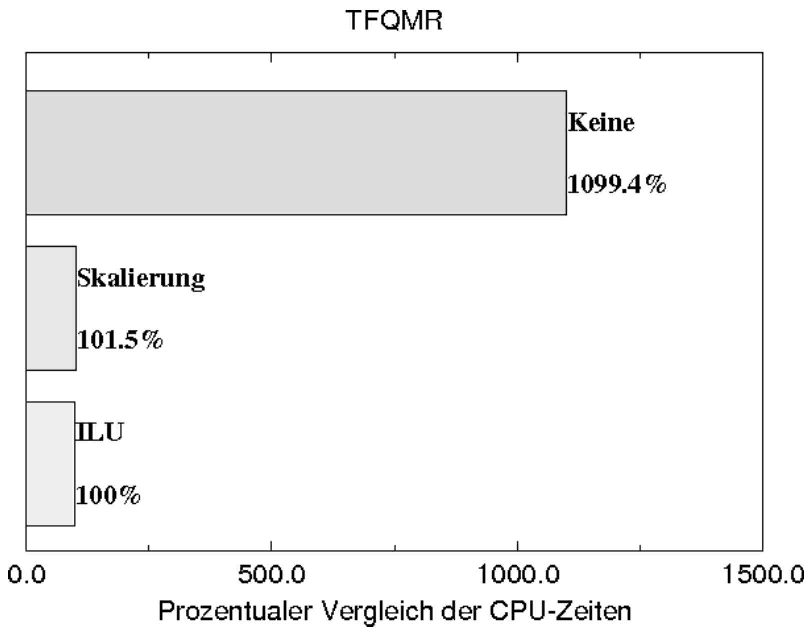


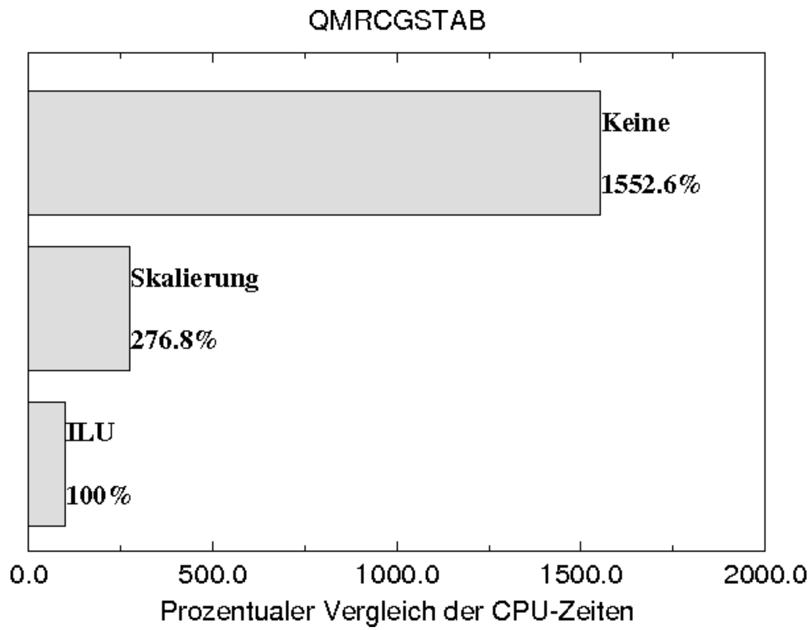
Fig.: Triangulation and isolines of the Mach number distribution

NACA0012-profil









Laminar flow about a flat plate

Re$= 6 \cdot 10^6$, Ma$= 5.0$, Angle of attack 0°

Triangulation: 7350 points

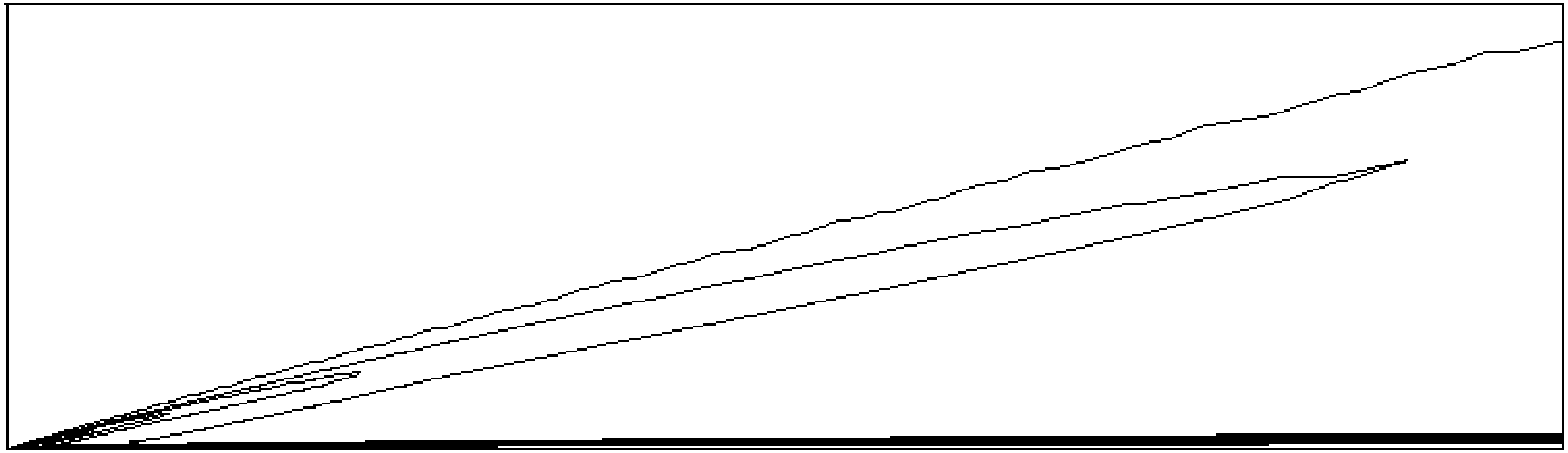
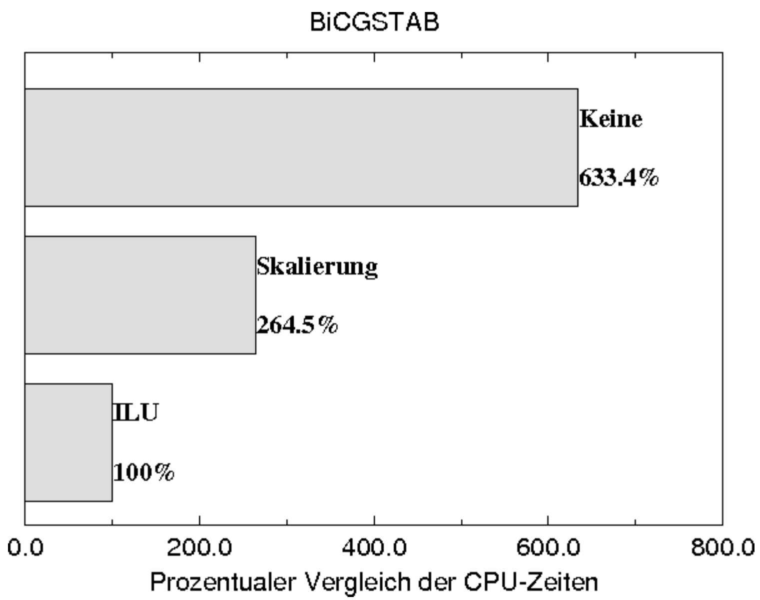
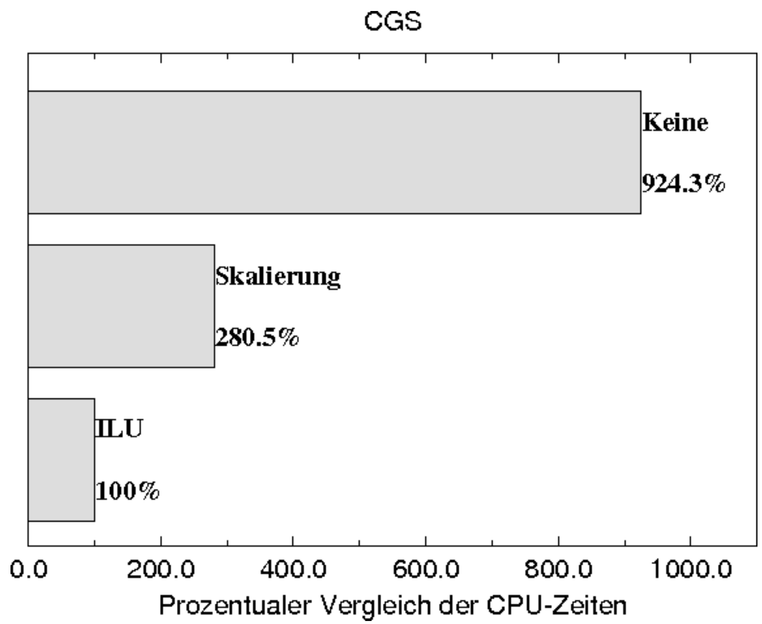
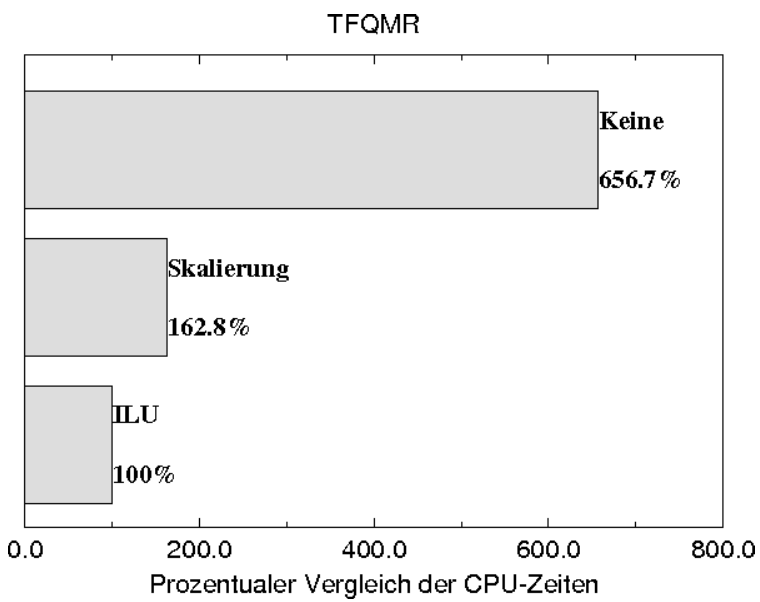


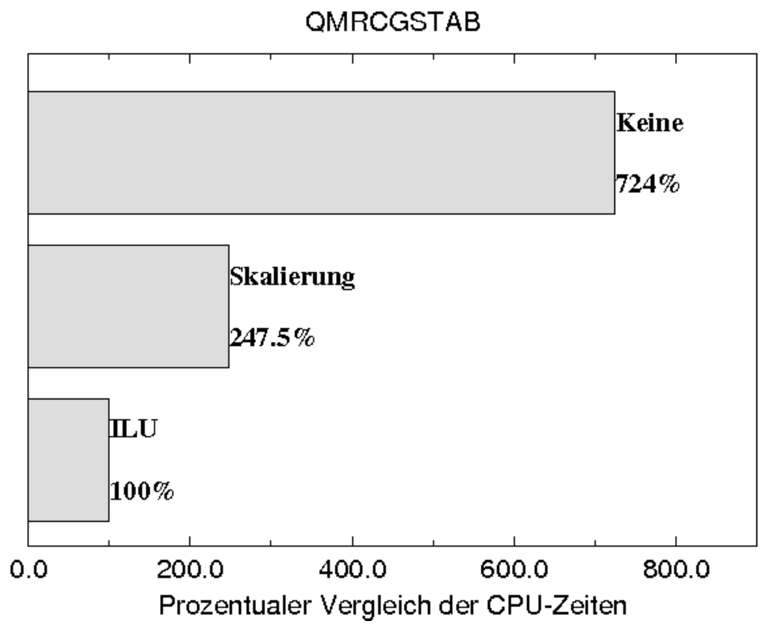
Fig.: Isolines of the Mach number distribution

Laminar flow about a flat plate





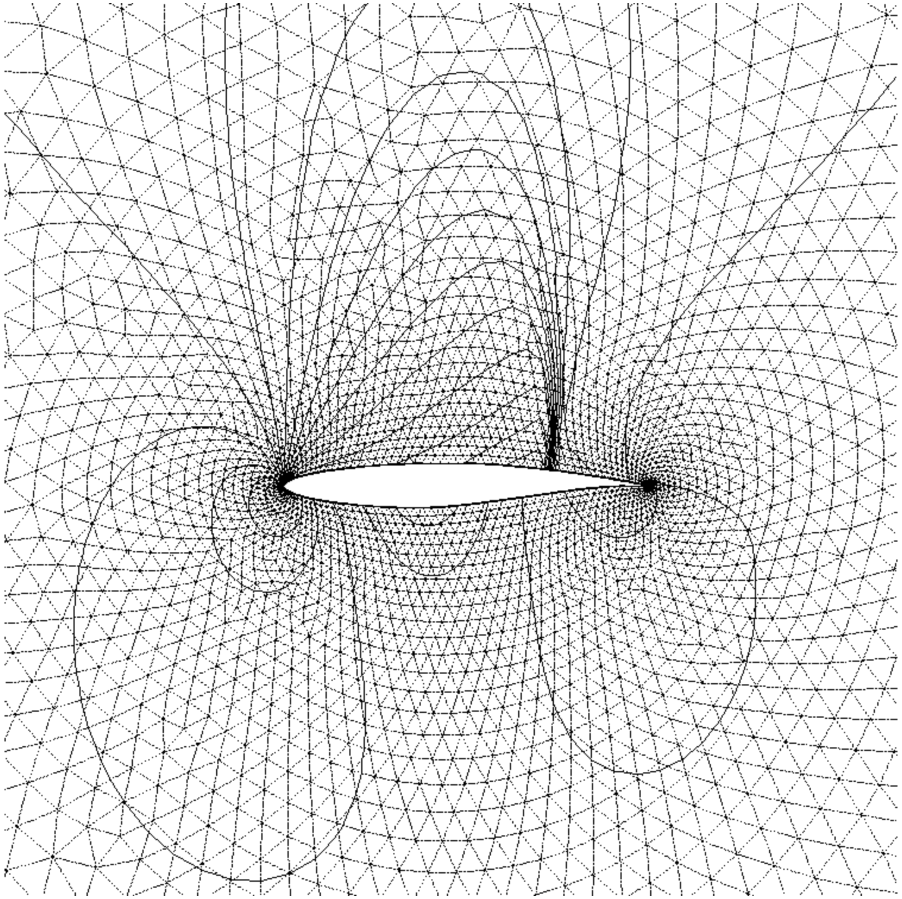




RAE 2822-profil

Ma$=$ 0.75, Angle of attack 3°, inviscid

Triangulation: 9974 triangles, 5071 points



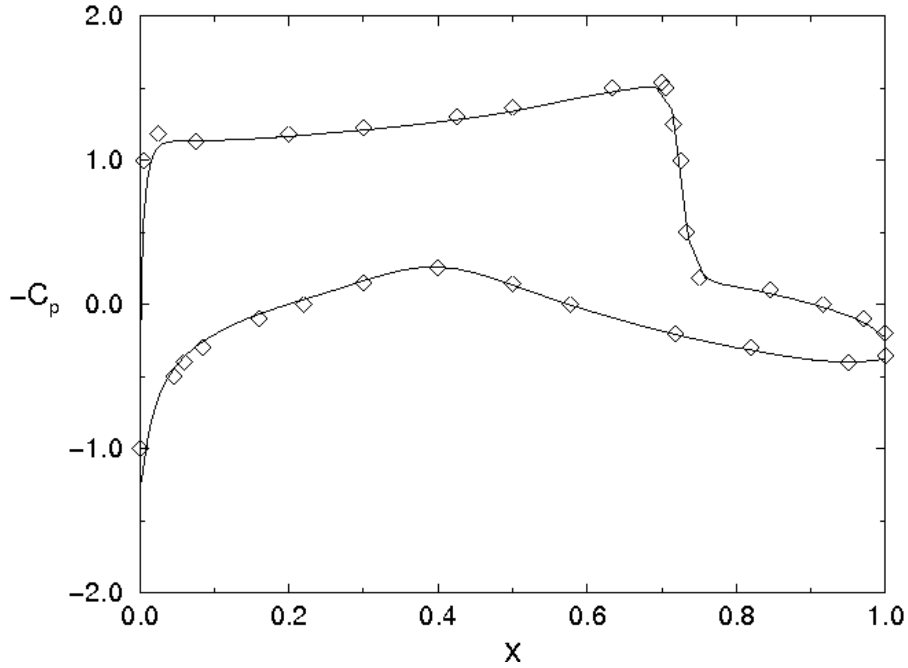


Fig.: Density and C_p-distribution

RAE 2822-profil

Explicit scheme	Implicit scheme	
	Scaling	Incomplete LU(5)
3497%	852%	100%

Tab.: Percentage comparison of the CPU-time

Ma$=$ 0.65, Angle of attack 3°, inviscid

Triangulation: 46914 triangles, 23751 points

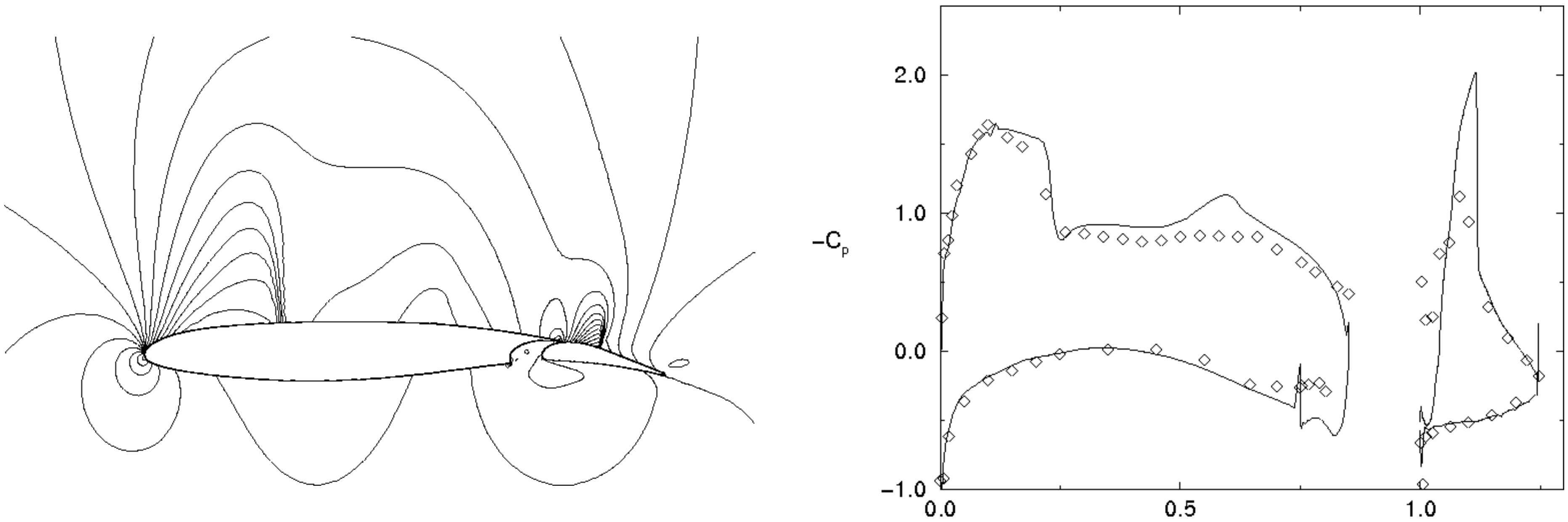


Fig.: Density and C_p-distribution

RAE 2822-profil

Explicit scheme	Implicit scheme	
	Scaling	Incomplete LU(5)
1003%	688%	100%

Tab.: Percentage comparison of the CPU-time

Summary

Preconditioning: General procedure

$$A\,x = b \iff \begin{cases} P_L\,A\,P_R\,y = P_L\,b \\ x = P_R y \end{cases}$$

Goal: Convergence acceleration and stabilization

Alternatives:

- Scaling
- Splitting-associated preconditioners (Gauß-Seidel, SOR, ...)
- Incomplete Factorization (ILU, IC)

PCG-scheme:

- $P_R = P_L^T \Longrightarrow P_L A P_R$ symm. pos. def., if A symm. pos. def.
- Symmetric Gauß-Seidel-scheme, IC

Results for FVM: Acceleration up to a factor of 10 by using ILU

Summary

Preconditioning: General procedure

$$A\,x = b \iff \begin{cases} P_L\,A\,P_R\,y = P_L\,b \\ x = P_R y \end{cases}$$

Goal: Convergence acceleration and stabilization

Alternatives:

- Scaling
- Splitting-associated preconditioners (Gauß-Seidel, SOR, ...)
- Incomplete Factorization (ILU, IC)

PCG-scheme:

- $P_R = P_L^T \Longrightarrow P_L A P_R$ symm. pos. def., if A symm. pos. def.
- Symmetric Gauß-Seidel-scheme, IC

Results for FVM: Acceleration up to a factor of 10 by using ILU

Summary

Preconditioning: General procedure

$$A\,x = b \iff \begin{cases} P_L\,A\,P_R\,y = P_L\,b \\ x = P_R y \end{cases}$$

Goal: Convergence acceleration and stabilization

Alternatives:

- Scaling
- Splitting-associated preconditioners (Gauß-Seidel, SOR, ...)
- Incomplete Factorization (ILU, IC)

PCG-scheme:

- $P_R = P_L^T \Longrightarrow P_L A P_R$ symm. pos. def., if A symm. pos. def.
- Symmetric Gauß-Seidel-scheme, IC

Results for FVM: Acceleration up to a factor of 10 by using ILU

Summary

Preconditioning: General procedure

$$A\,x = b \iff \begin{cases} P_L\,A\,P_R\,y = P_L\,b \\ x = P_R y \end{cases}$$

Goal: Convergence acceleration and stabilization

Alternatives:

- Scaling
- Splitting-associated preconditioners (Gauß-Seidel, SOR, ...)
- Incomplete Factorization (ILU, IC)

PCG-scheme:

- $P_R = P_L^T \Longrightarrow P_L A P_R$ symm. pos. def., if A symm. pos. def.
- Symmetric Gauß-Seidel-scheme, IC

Results for FVM: Acceleration up to a factor of 10 by using ILU

Summary

Preconditioning: General procedure

$$A\,x = b \iff \begin{cases} P_L\,A\,P_R\,y = P_L\,b \\ x = P_R y \end{cases}$$

Goal: Convergence acceleration and stabilization

Alternatives:

- Scaling
- Splitting-associated preconditioners (Gauß-Seidel, SOR, ...)
- Incomplete Factorization (ILU, IC)

PCG-scheme:

- $P_R = P_L^T \Longrightarrow P_L A P_R$ symm. pos. def., if A symm. pos. def.
- Symmetric Gauß-Seidel-scheme, IC

Results for FVM: Acceleration up to a factor of 10 by using ILU