Advanced Numerical Tools
Reordering Techniques and Iterative Methods

B. Cerfontaine

ULg - ArGENCo - Boursier FRIA

11/09/2013
Table of contents

1 Introduction
2 Direct methods
3 Reordering
4 Iterative methods
5 Conclusion
Increasing complexity of studied phenomena (localization, anisotropy, complex 3D geometry) \(\Rightarrow\) increasing number of equations
Solving $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$

Assembling example:
- 6 elements
- 12 nodes
- 1dof/node
Assembling of global stiffness tangent matrix

Solving $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$

Local stiffness matrix (element 1):

\[
\begin{bmatrix}
1 & 2 & 6 & 5 \\
1 & C_{11} & C_{12} & C_{13} & C_{14} \\
2 & C_{21} & C_{22} & C_{23} & C_{24} \\
6 & C_{31} & C_{32} & C_{33} & C_{34} \\
5 & C_{41} & C_{42} & C_{43} & C_{44}
\end{bmatrix}
\]

Global stiffness matrix (12 dofs x 12 dofs):

(coloured non-zero entries)
Assembling of global stiffness tangent matrix

Solving $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$

Global stiffness matrix (12 dofs x 12 dofs):

Local stiffness matrix (element 2):

$$
\begin{bmatrix}
2 & 3 & 7 & 6 \\
C_{11} & C_{12} & C_{13} & C_{14} \\
C_{21} & C_{22} & C_{23} & C_{24} \\
C_{31} & C_{32} & C_{33} & C_{34} \\
C_{41} & C_{42} & C_{43} & C_{44}
\end{bmatrix}
$$

(coloured non-zero entries)
Assembling of global stiffness tangent matrix

Solving $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$

Global stiffness matrix (12 dofs x 12 dofs):

Properties of the matrix:
- sparse
- potentially symmetric
- potentially band

(coloured non-zero entries)
Table of contents

1 Introduction

2 Direct methods

3 Reordering

4 Iterative methods

5 Conclusion
Triangular systems

Let’s consider the following system $L \cdot x = b$ ($N = 4$):

$$
\begin{bmatrix}
 l_{11} & 0 & 0 & 0 \\
 l_{21} & l_{22} & 0 & 0 \\
 l_{31} & l_{32} & l_{33} & 0 \\
 l_{41} & l_{42} & l_{43} & l_{44}
\end{bmatrix}
\begin{bmatrix}
 x_1 \\
 x_2 \\
 x_3 \\
 x_4
\end{bmatrix}
= 
\begin{bmatrix}
 b_1 \\
 b_2 \\
 b_3 \\
 b_4
\end{bmatrix}
$$

- exact solution
- $\sim O(N^2)$ operations
- easy to implement
- forward (L matrix) or backward (U matrix) substitution

\[
x_1 = \frac{b_1}{l_{11}}
\]

\[
x_2 = \frac{b_2 - l_{21} \cdot x_1}{l_{22}}
\]

\[
\ldots
\]

\[
x_i = \frac{b_i - \sum_{k=1}^{i-1} l_{ik}x_k}{l_{ii}}
\]

B. Cerfontaine
Gaussian elimination

Let’s consider the following system $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$ $(n = 4)$

$$
\begin{bmatrix}
a_{11} & a_{12} & a_{13} & a_{14} \\
a_{21} & a_{22} & a_{23} & a_{24} \\
a_{31} & a_{32} & a_{33} & a_{34} \\
a_{41} & a_{42} & a_{43} & a_{44}
\end{bmatrix}
\cdot
\begin{bmatrix}
x_1 \\
x_2 \\
x_3 \\
x_4
\end{bmatrix}
=
\begin{bmatrix}
b_1 \\
b_2 \\
b_3 \\
b_4
\end{bmatrix}
$$

Gaussian elimination transforms $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$ into $\mathbf{L} \cdot \mathbf{x} = \mathbf{b}'$
Elimination of $a_{21}$:

\[
\begin{bmatrix}
  a_{11} & a_{12} & a_{13} & a_{14} \\
  0 & a^{(2)}_{22} & a^{(2)}_{23} & a^{(2)}_{24} \\
  a_{31} & a_{32} & a_{33} & a_{34} \\
  a_{41} & a_{42} & a_{43} & a_{44}
\end{bmatrix}
\begin{bmatrix}
  x_1 \\
  x_2 \\
  x_3 \\
  x_4
\end{bmatrix}
=
\begin{bmatrix}
  b^{(2)}_1 \\
  b^{(2)}_2 \\
  b_3 \\
  b_4
\end{bmatrix}
\]

\[
a^{(2)}_{21} \leftarrow a_{21} - \frac{a_{21}}{a_{11}} \cdot a_{11} = 0
\]

\[
a^{(2)}_{22} \leftarrow a_{22} - \frac{a_{21}}{a_{11}} \cdot a_{12}
\]

\[
a^{(2)}_{23} \leftarrow a_{23} - \frac{a_{21}}{a_{11}} \cdot a_{13}
\]

\[
a^{(2)}_{24} \leftarrow a_{24} - \frac{a_{21}}{a_{11}} \cdot a_{14}
\]

\[
b^{(2)}_2 \leftarrow b_2 - \frac{a_{21}}{a_{11}} \cdot b_1
\]
Elimination of $a_{31}$ and $a_{41}$

$$
\begin{bmatrix}
    a_{11} & a_{12} & a_{13} & a_{14} \\
    0 & a_{22}^{(2)} & a_{23}^{(2)} & a_{24}^{(2)} \\
    0 & a_{32}^{(2)} & a_{33}^{(2)} & a_{34}^{(2)} \\
    0 & a_{42}^{(2)} & a_{43}^{(2)} & a_{44}^{(2)}
\end{bmatrix}
\cdot
\begin{Bmatrix}
    x_1 \\
    x_2 \\
    x_3 \\
    x_4
\end{Bmatrix}
= 
\begin{Bmatrix}
    b_1^{(2)} \\
    b_2^{(2)} \\
    b_3^{(2)} \\
    b_4^{(2)}
\end{Bmatrix}
$$

$m_{i1} = \frac{a_{i1}}{a_{11}}$ for $i = 3, 4$ and $j = 2, \ldots, 4$

$a_{ij}^{(2)} \leftarrow a_{ij} - m_{i1} \cdot a_{1j}$

$b_i^{(2)} \leftarrow b_i - m_{i1} \cdot b_{1}$
Gaussian elimination

Elimination of $a_{32}$, $a_{42}$ and $a_{43}$

\[
\begin{bmatrix}
  a_{11} & a_{12} & a_{13} & a_{14} \\
  0 & a_{22}^{(2)} & a_{23}^{(2)} & a_{24}^{(2)} \\
  0 & 0 & a_{33}^{(3)} & a_{34}^{(3)} \\
  0 & 0 & 0 & a_{44}^{(4)} \\
\end{bmatrix}
\times
\begin{bmatrix}
  x_1 \\
  x_2 \\
  x_3 \\
  x_4 \\
\end{bmatrix}
=
\begin{bmatrix}
  b_1^{(1)} \\
  b_2^{(2)} \\
  b_3^{(3)} \\
  b_4^{(4)} \\
\end{bmatrix}
\]

then the remaining upper triangular matrix has to be solved.

- exact solution
- ~ $O(N^3)$ operations
- pivoting if $a_{ii} = 0$ (or $a_{ii} <<$ for numerical stability)
- best method for dense $A$ matrices
- implemented in LAGAMINE for sparse systems (KNSYM= 4)
**Principle**: rearranging the equations in order to minimize the number of operations to carry

Ordering efficiency parameter: \( \sigma = \sum_{i=1}^{N} M_i^2 \)

where \( M_i \) = number of off-diagonal term during the elimination of node \( i \)
**Principle**: rearranging the equations in order to minimize the number of operations to carry

Ordering efficiency parameter: \( \sigma = \sum_{i=1}^{N} M_i^2 \)

where \( M_i \) = number of off-diagonal term during the elimination of node \( i \)

Elimination of node 1:
- red: entries to be cancelled
- green: previously zero entries
- purple: modified entries

\( M^1 = 5 \) (in red)

\( \Rightarrow 25 \) operations (green + purple)
**Definitions**

**Principle**: rearranging the equations in order to minimize the number of operations to carry

Ordering efficiency parameter: \( \sigma = \sum_{i=1}^{N} M_i^2 \)

where \( M_i \) = number of off-diagonal term during the elimination of node \( i \)

Elimination of node 2:
- red: entries to be cancelled
- green: previously zero entries
- purple: modified entries

\( M^2 = 4 \) (in red)

\( \Rightarrow 16 \) operations (green + purple)
Basic Algorithm

1. Starting from one node to be eliminated (blue, to be adequately chosen)
Basic Algorithm

1. Starting from one node to be eliminated (blue, to be adequately chosen)
2. Choosing the next one
   - several possibilities (green)
   - for each possibility: how many new active nodes (red)?
   - the next node is the one which leads to minimum new active nodes
Basic Algorithm

1. Starting from one node to be eliminated (blue, to be adequately chosen)
2. Choosing the next one
   - several possibilities (green)
   - for each possibility: how many new active nodes (red)?
   - the next node is the one which leads to minimum new active nodes
Basic Algorithm

1. Starting from one node to be eliminated (blue, to be adequately chosen)
2. Chosing the next one
   - several possibilities (green)
   - for each possibility: how many new active nodes (red)?
   - the next node is the one which leads to minimum new active nodes
   - if equality chose the one which is active since the greater number of steps

```
    5
   / \
  3   4
 /     |
1     7
   \
    2
```

B. Cerfontaine

LAGASHOP 2013
Basic Algorithm

1. Starting from one node to be eliminated (blue, to be adequately chosen)
2. Chosing the next one
   - several possibilities (green)
   - for each possibility: how many new active nodes (red)?
   - the next node is the one which leads to minimum new active nodes
   - if equality chose the one which is active since the greater number of steps
Reordering

Basic Algorithm

1. Starting from one node to be eliminated (blue, to be adequately chosen)
2. Chosing the next one
   - several possibilities (green)
   - for each possibility: how many new active nodes (red)?
   - the next node is the one which leads to minimum new active nodes
   - if equality chose the one which is active since the greater number of steps

```plaintext
1 6
2 7
3 8
4
5
```

B. Cerfontaine

LAGASHOP 2013 11/09/2013 11 / 34
**Basic Algorithm**

1. **Starting from one node to be eliminated** (blue, to be adequately chosen)

2. **Choosing the next one**
   - several possibilities (green)
   - for each possibility: how many new active nodes (red)?
   - the next node is the one which leads to minimum new active nodes
   - if equality chose the one which is active since the greater number of steps

![Diagram of the network with nodes and connections labeled 1 to 8.](image-url)
Basic Algorithm

1. Starting from one node to be eliminated (blue, to be adequately chosen)

2. Choosing the next one
   - several possibilities (green)
   - for each possibility: how many new active nodes (red)?
   - the next node is the one which leads to minimum new active nodes
   - if equality chose the one which is active since the greater number of steps

Reorder following the sequence of eliminated nodes
Basic Algorithm

1. Starting from one node to be eliminated (blue, to be adequately chosen)
2. Choosing the next one
   - several possibilities (green)
   - for each possibility: how many new active nodes (red)?
   - the next node is the one which leads to minimum new active nodes
   - if equality chose the one which is active since the greater number of steps
3. Reorder following the sequence of eliminated nodes

Sequence: 5, 3, 2, 1, 6, 4, 7, 8
Number of adjacent nodes $= M^i$ then minimizing it minimizes $\sigma$ and cpu time.
Comparison between old and new matrices

Without renum $\sigma = 87$

With renum $\sigma = 40$
Non zero entries of matrices (before and after renum)

Types of reordering widely used in LAGAMINE:

1. Oil stain (ITYREN = 1)
   - Efficient for every mesh;
   - Parameters needed;

2. Directional reordering (ITYREN = 2)
   - Efficient for rectangular meshes;
   - Need a direction in which the structure has the greatest number of nodes.

3. Based on Sloan method: improved oil stain (ITYREN = 3)
   - Includes algorithm for best start node;
   - Efficient for every mesh;
   - No parameter needed;
   - Fastest than oil stain (?).
Reordering

Non zero entries of matrices (before and after renum)

Size 619x619

Size : 10505x10505
Oedometer test: with/without reordering (and comparison directional/Sloan)


## Table of contents

1. Introduction
2. Direct methods
3. Reordering
4. Iterative methods
   - Introduction and principles
   - GMRES method
   - Practical use
5. Conclusion
**Example: Jacobi iterations**

**Principle:** finding the new value of $\xi_i^{(k)}$ that annihilates the $i$-th component of the residual vector $r_i = (b - A \cdot x_k)_i$

$$r_i \equiv a_{ii} \cdot \xi_i^{(k+1)} + \sum_{j=1}^{n} a_{ij} \cdot \xi_j^{(k)} - \beta_i = 0 \quad \text{for} \quad i = 1, \ldots, n$$
Example: Jacobi iterations

**Principle**: finding the new value of $\xi_i^{(k)}$ that annihilates the i-th component of the residual vector $r_i = (b - A \cdot x_k)_i$

\[
\begin{align*}
 r_i &\equiv 0 = a_{ii} \cdot \xi_i^{(k+1)} + \sum_{j=1 \atop j \neq i}^{n} a_{ij} \cdot \xi_j^{(k)} - \beta_i \\
\Leftrightarrow \quad \xi_i^{(k+1)} & = \frac{1}{a_{ii}} \left( \beta_i - \sum_{j=1 \atop j \neq i}^{n} a_{ij} \cdot \xi_j^{(k)} \right) \quad \text{for} \quad i = 1, \ldots, n
\end{align*}
\]
**Example: Jacobi iterations**

**Principle:** finding the new value of $\xi_i^{(k)}$ that annihilates the $i$-th component of the residual vector $r_i = (b - A \cdot x_k)_i$

$$
\xi_i^{(k+1)} = \frac{1}{a_{ii}} \left( \beta_i - \sum_{j=1}^{n} a_{ij} \cdot \xi_j^{(k)} \right) \quad \text{for} \quad i = 1, \ldots, n
$$

Rewritten in vector form

$$
x_{k+1} = D^{-1} \cdot (E + F) \cdot x_k + D^{-1} \cdot b
$$

where

- $D$: diagonal of $A$
- $-E$: lower triangular part of $A$
- $-F$: upper triangular part of $A$
**Preconditioning**

**Principle**: "Preconditioning is a means of transforming the original linear system into one which has the same solution, but which is likely to be easier to solve with an iterative solver" (Saad, 2003)

Preconditioning matrix $\mathbf{M}$:
- close to $\mathbf{A}$;
- nonsingular;
- inexpensive to solve linear system $\mathbf{M} \cdot \mathbf{x} = \mathbf{b}$.

One of the simplest ways of defining a preconditioner is to perform an *incomplete factorization* of the matrix $\mathbf{A}$

$$\mathbf{A} = \mathbf{L} \cdot \mathbf{U} - \mathbf{R}$$
Principle: "preconditioning is a means of transforming the original linear system into one which has the same solution, but which is likely to be easier to solve with an iterative solver" (Saad, 2003)

Applied to the left
\[ M^{-1} \cdot A \cdot x = M^{-1} \cdot b \]
or applied to the right
\[ A \cdot M^{-1} \cdot u = b \quad x \equiv M^{-1} \cdot u \]
# Table of contents

1. **Introduction**

2. **Direct methods**

3. **Reordering**

4. **Iterative methods**
   - Introduction and principles
   - GMRES method
   - Practical use

5. **Conclusion**
Building \textbf{iteratively} a solution ($\tilde{x}$)

- to the problem $A\cdot x = b$
- from an initial guess ($x_0$) of the solution
- and a basis of \textit{linearly independent vectors} $\mathcal{K}_m$
- through the following \textit{linear combination}:

$$
\tilde{x} = x_0 + \sum_{i=1}^{m} y_i \cdot v_i \quad v_i \in \mathcal{K}_m
$$

The size of the basis, $m$

- is unknown, \textit{a priori} (iterative concept)
- is much less than $N$, the number of equations.
There is two ways for solving the problem of a beam submitted to an arbitrary time-dependent loading:

1. Nodal base method: solving a $N \times N$ system.


\[ \tilde{y}(t) = \sum_{i=1}^{m} \eta_i(t) \cdot y_i \]
There is two ways for solving the problem of a beam submitted to an arbitrary time-dependent loading:

1. **Nodal base method**: solving a $N \times N$ system

2. **Modal base method**: projection onto eigenmodes (increasing eigenfrequencies)

$$\tilde{y}(t) = \sum_{i=1}^{m} \eta_i(t) \cdot y_i$$

$m$ decoupled equations to be solved $\ll N$
Which basis?

In the following iterative method, subspace $\mathcal{K}_m$ is a Krylov subspace, i.e.

$$\mathcal{K}_m(A, v) \equiv \text{span} \{ v, Av, A^2v, \ldots, A^{m-1}v \}$$

and specifically, GMRES uses

$$\mathcal{K}_m(A, r_0) \equiv \text{span} \{ r_0, Ar_0, A^2r_0, \ldots, A^{m-1}r_0 \}$$

where

$$r_0 = b - A \cdot x_0$$

$$A \in \mathbb{R}^{n\times n} \quad \text{and} \quad v, r_0 \in \mathbb{R}^n$$
Iterative methods

GMRES method

Which basis?

This basis can be computed iteratively since there is a relation between two of its consecutive vectors, \( v_j = A \cdot v_{j-1} \)

Then following a Gram-Schmidt procedure for orthogonal bases

1. \( v_1 = \frac{r_0}{\|r_0\|_2} \)
Which basis?

This basis can be computed iteratively since there is a relation between two of its consecutive vectors, \( i.e. \mathbf{v}_j = \mathbf{A} \cdot \mathbf{v}_{j-1} \)

Then following a Gram-Schmidt procedure for orthogonal bases

1. \( \mathbf{v}_1 = \frac{\mathbf{r}_0}{\|\mathbf{r}_0\|_2} \)
2. actual basis of degree \( m \), compute \( \mathbf{v}_{m+1} = \mathbf{A} \cdot \mathbf{v}_m \)
Which basis?

This basis can be computed iteratively since there is a relation between two of its consecutive vectors, \( \mathbf{v}_j = \mathbf{A} \cdot \mathbf{v}_{j-1} \).

Then following a Gram-Schmidt procedure for orthogonal bases:

1. \( \mathbf{v}_1 = \frac{\mathbf{r}_0}{\|\mathbf{r}_0\|_2} \)
2. actual basis of degree \( m \), compute \( \mathbf{v}_{m+1} = \mathbf{A} \cdot \mathbf{v}_m \)
3. to be orthogonalized against \( \mathbf{v}_i \) for \( i = 1, \ldots, m \)

\[
\mathbf{v}_{m+1} := \mathbf{v}_{m+1} - \sum_{i=1}^{m} (\mathbf{v}_{m+1}, \mathbf{v}_i) \cdot \mathbf{v}_i
\]

\( \sim O(m \times n) \) operations!
Which basis?

This basis can be computed iteratively since there is a relation between two of its consecutive vectors, i.e. $v_j = A \cdot v_{j-1}$

Then following a Gram-Schmidt procedure for orthogonal bases

1. $v_1 = \frac{r_0}{\|r_0\|_2}$
2. actual basis of degree $m$, compute $v_{m+1} = A \cdot v_m$
3. to be orthogonalized against $v_i$ for $i = 1, \ldots, m$

$$v_{m+1} := v_{m+1} - \sum_{i=1}^{m} (v_{m+1}, v_i) \cdot v_i$$

4. normalize $v_{m+1} = \frac{v_{m+1}}{\|v_{m+1}\|_2}$
How to compute $y$? How to choose the size of $m$?

If $\mathbf{V}_m$ is the $n \times m$ matrix describing the basis $\mathcal{K}_m$ then solution is computed as

$$\tilde{x} = x_0 + \mathbf{V}_m \cdot y$$

where $y$ minimizes

$$\|\mathbf{b} - \mathbf{A} \cdot \tilde{x}\|_2 = \|\mathbf{b} - \mathbf{A} \cdot (x_0 + \mathbf{V}_m \cdot y)\|_2 = \|\mathbf{r}_0 - \mathbf{A} \cdot \mathbf{V}_m \cdot y\|_2$$

which is a least square problem of size $m \ll N$

Practically the size $m$ of $\mathcal{K}_m$ increases **iteratively** up to achieve a residual

$$\|\mathbf{r}_i\|_2 < \epsilon \cdot \|\mathbf{r}_0\|_2$$
GMRES method implemented in LAGAMINE (KNSYM = ±6)

- approached solution (depends on the convergence norm!)
- no need to inverse the global matrix $A$
- best method for very big number of equations
- $\sim O((m + 3 + 1/m) \cdot N + NZ)$ where
  - $m$ is the number of iterations
  - $N$ is the number of unknowns
  - $NZ$ is the number of nonzero terms in $A$
- requires input parameters
Table of contents

1. Introduction

2. Direct methods

3. Reordering

4. Iterative methods
   - Introduction and principles
   - GMRES method
   - Practical use

5. Conclusion
Practical use: oedometer cube

Systematic comparison using a cubic example

- oedometric boundary conditions
- purely mechanical/hydro-mechanical behaviour
- 8-nodes 3D blocs
- varying mesh size $N^3$
  $N \in [10, 15, 20, 25, 30, 35, 40, 45]$
Practical use: oedometer cube

Minimum CPU time for solving system

Cost savings using GMRES

Direct CPU time/Iterative CPU time

Iterative methods

Practical use
Iterative and direct methods for gas injection: gas pressure at a given point

- global convergence depends on LAGAMINE criteria: PRECU and PRECF identical for both methods
- difference of convergence rate: different time steps
**Preconditioning** parameters : LFIL, DROPTOL

The incomplete factorization of LU follows the same steps as the gaussian elimination.

For a given off-diagonal term to be eliminated \( a_{kk} \)

\[
m_{kl} = \frac{a_{kl}}{a_{kk}}
\]

\[
a_{kj} \leftarrow a_{kj} - m_{kl} \cdot a_{kj} \quad b_k \leftarrow b_k - m_{kl} \cdot b_k \quad \text{for} \quad j = 1, \ldots, n
\]

If \(|m_{kl}| < \text{DROPTOL}\) nothing is done and \(a_{kl} \leftarrow 0\)
**Preconditioning** parameters: LFIL, DROPTOL

During the LU factorization process:
- sort terms of row i of L and U;
- keep the lfil greatest in L and U;
- next row.

**LU incomplete**

\[
\text{CPU time}(\text{lfil}_i) = \frac{1}{n} \sum_{j=\text{lfil}_1}^{\text{lfil}_n} \text{CPU time}_j
\]

B. Cerfontaine
LAGASHOP 2013
11/09/2013
Preconditioning parameters: LFIL, DROPTOL

- Increasing LFIL
  - Increasing memory storage required
  - Increasing computational cost for computing the preconditioner
  - Increasing computational cost for solving the system

- Decreasing LFIL
  - Risk of no convergence
  - Risk of instability during the process (observed for large hydro-mechanical systems)

- Increasing DROPTOL
  - Conditional decreasing of computational cost

- Decreasing DROPTOL
  - Conditional increasing of computational cost
Resolution parameter: IM, MAXITS

If the size of the Krylov basis is greater than IM,

- initial guess of the solution is deemed too far from the actual one,
- algorithm is restarted with $m = 1$ and $x_0 = x_m$. 
**Parameters**

**Resolution parameter**: IM, MAXITS

If the size of the Krylov basis is greater than IM,
- initial guess of the solution is deemed too far from the actual one,
- algorithm is restarted with $m = 1$ and $x_0 = x_m$,

The total number of iterations ($n_{its}$) is equal to the number of orthogonalization processes already done which is equal
- to the actual size $k$ of the Krylov process if $n_{its} < IM$;
- to $n_{restart} \times IM + k$ if $n_{its} > IM$.

If no convergence occurs after $n_{its} = MAXITS$
- algorithm crashes and the time step is reduced
Parameters

There is a relationship between the LFIL parameter and the number of iterations necessary to reach convergence.
Convergence parameters: EPS, RESTOL

The internal iterative process ends if (relative convergence)

$$\| r_i \|_2 < \text{EPS} \cdot \| r_0 \|_2$$

or if (absolute convergence)

$$\| r_i \|_2 < \text{RESTOL}$$

The smaller the EPS the higher the number of iterations to reach convergence!
Iterative methods

Practical use

Example of configurations

RESTOL is imposed to $10^{-40}$

<table>
<thead>
<tr>
<th>NDOFS</th>
<th>TYPE</th>
<th>LFIL</th>
<th>DROPTOL</th>
<th>IM</th>
<th>MAXITS</th>
<th>EPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>283544</td>
<td>cube M</td>
<td>45</td>
<td>$10^{-6}$</td>
<td>200</td>
<td>500</td>
<td>$10^{-5}$</td>
</tr>
<tr>
<td>220940</td>
<td>gallery M</td>
<td>130</td>
<td>$10^{-6}$</td>
<td>500</td>
<td>4000</td>
<td>$10^{-5}$</td>
</tr>
<tr>
<td>262236</td>
<td>cube HM</td>
<td>20</td>
<td>$10^{-6}$</td>
<td>200</td>
<td>500</td>
<td>$10^{-5}$</td>
</tr>
<tr>
<td>752226</td>
<td>loca M</td>
<td>200</td>
<td>$10^{-8}$</td>
<td>500</td>
<td>1000</td>
<td>$10^{-5}$</td>
</tr>
</tbody>
</table>

High LFIL with high number of dofs to avoid instabilities!
Mainly for GMRES and iterative methods (available online)


Or a summary
Table of contents

1 Introduction
2 Direct methods
3 Reordering
4 Iterative methods
   - Introduction and principles
   - GMRES method
   - Practical use
5 Conclusion
Increasing size of systems to be solved as a corollary effect of increasing complexity of modelling

High degree of sparsity of the systems to be solved leading to specific methods

Reordering techniques:
- very efficient
- coupled with direct solvers
- available in LAGAMINE

Iterative methods:
- approximate solution
- efficient for very large number of equations
- much complex algorithms
- parameters: mix of art and science
- available in LAGAMINE
FSA, créatrice d'avenirs