Advanced Numerical Tools

Reordering Techniques and Iterative Methods

B. Cerfontaine

ULg - ArGEnCo - Boursier FRIA

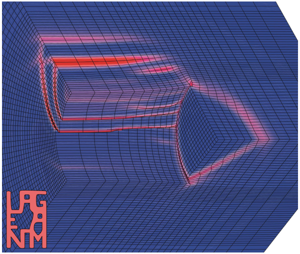
11/09/2013

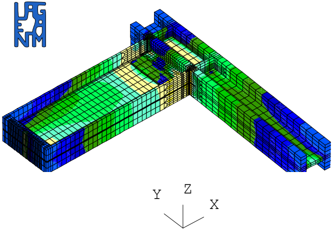
Table of contents

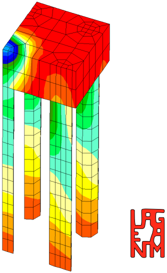
1. Introduction
2. Direct methods
3. Reordering
4. Iterative methods
5. Conclusion

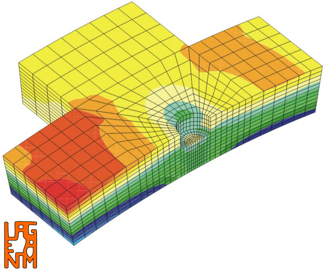
Table of contents

Increasing complexity of studied phenomena (localization, anisotropy, complex 3D geometry) ⇒ increasing number of equations



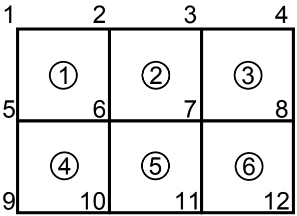






Assembling of global stiffness tangent matrix

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

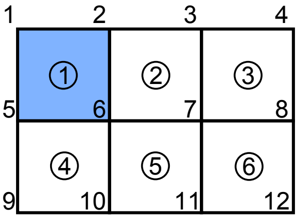


Assembling example :

- 6 elements
- 12 nodes
- 1dof/node

Assembling of global stiffness tangent matrix

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



Local stiffness matrix (element 1) :

$$\begin{array}{c|cccc|} & 1 & 2 & 6 & 5 \\ 1 & C_{11} & C_{12} & C_{13} & C_{14} \\ 2 & C_{21} & C_{22} & C_{23} & C_{24} \\ 6 & C_{31} & C_{32} & C_{33} & C_{34} \\ 5 & C_{41} & C_{42} & C_{43} & C_{44} \end{array}$$

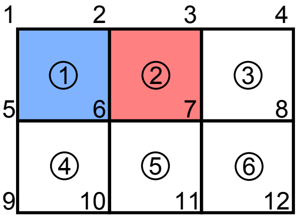
Global stiffness matrix (12 dofs x 12 dofs) :

	1	2	3	4	5	6	7	8	9	10	11	12
1	🟦	🟦			🟦	🟦						
2	🟦	🟦			🟦	🟦						
3												
4												
5	🟦	🟦			🟦	🟦						
6	🟦	🟦			🟦	🟦						
7												
8												
9												
10												
11												
12												

(coloured non-zero entries)

Assembling of global stiffness tangent matrix

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



Local stiffness matrix (element 2) :

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

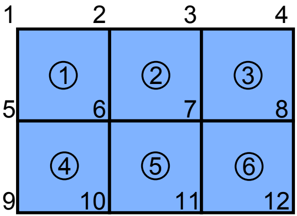
Global stiffness matrix (12 dofs x 12 dofs) :

	1	2	3	4	5	6	7	8	9	10	11	12
1	🟦	🟦			🟦	🟦						
2	🟦	🟪	🟥		🟦	🟪	🟥					
3		🟥	🟥			🟥	🟥					
4												
5	🟦	🟦			🟦	🟦						
6	🟦	🟪	🟥		🟦	🟪	🟥					
7		🟥	🟥			🟥	🟥					
8												
9												
10												
11												
12												

(coloured non-zero entries)

Assembling of global stiffness tangent matrix

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



Global stiffness matrix (12 dofs x 12 dofs) :

	1	2	3	4	5	6	7	8	9	10	11	12
1	■	■			■	■						
2	■	■	■		■	■	■					
3		■	■	■		■	■	■				
4			■	■			■	■				
5	■	■			■	■			■	■		
6	■	■	■		■	■	■		■	■	■	
7		■	■	■		■	■	■		■	■	■
8			■	■			■	■			■	■
9					■	■			■	■		
10					■	■	■		■	■	■	
11						■	■	■		■	■	■
12							■	■			■	■

(coloured non-zero entries)

Properties of the matrix :

- sparse
- potentially symmetric
- potentially band

Table of contents

1 Introduction

2 Direct methods

3 Reordering

4 Iterative methods

5 Conclusion

Triangular systems

Let's consider the following system $\underline{\mathbf{L}} \cdot \mathbf{x} = \mathbf{b} \quad (N = 4)$:

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

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

$$\begin{aligned} x_1 &= \frac{b_1}{l_{11}} \\ x_2 &= \frac{b_2 - l_{21} \cdot x_1}{l_{22}} \\ &\dots \\ x_i &= \frac{b_i - \sum\limits_{k=1}^{i-1} l_{ik} x_k}{l_{ii}} \end{aligned}$$

Gaussian elimination

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

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

Gaussian elimination transforms $\underline{\mathbf{A}} \cdot \mathbf{x} = \mathbf{b}$ into $\underline{\mathbf{L}} \cdot \mathbf{x} = \mathbf{b}'$

Gaussian elimination

Elimination of a_{21} :

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ 0 & a_{22}^{(2)} & a_{23}^{(2)} & a_{24}^{(2)} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \cdot \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{Bmatrix} = \begin{Bmatrix} b_1 \\ b_2^{(2)} \\ b_3 \\ b_4 \end{Bmatrix}$$

$$a_{21}^{(2)} \leftarrow a_{21} - \frac{a_{21}}{a_{11}} \cdot a_{11} = 0 \qquad a_{22}^{(2)} \leftarrow a_{22} - \frac{a_{21}}{a_{11}} \cdot a_{12}$$

$$a_{23}^{(2)} \leftarrow a_{23} - \frac{a_{21}}{a_{11}} \cdot a_{13} \qquad a_{24}^{(2)} \leftarrow a_{24} - \frac{a_{21}}{a_{11}} \cdot a_{14}$$

$$b_2^{(2)} \leftarrow b_2 - \frac{a_{21}}{a_{11}} \cdot b_1$$

Gaussian elimination

Elimination of a_{31} and a_{41}

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

$$m_{i1} = \frac{a_{i1}}{a_{11}} \qquad \text{for} \quad i = 3,4 \quad \text{and} \quad j = 2,.,4$$

$$a_{ij}^{(2)} \leftarrow a_{ij} - m_{i1} \cdot a_{1j} \quad b_i^{(2)} \leftarrow b_i - m_{i1} \cdot b_1$$

Gaussian elimination

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

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

then the remaining upper triangular matrix has to be solved.

- exact solution
- $\sim O(N^3)$ operations
- pivoting if $a_{ii} = 0$ (or $a_{ii} <<$ for numerical stability)
- best method for dense $\underline{\mathbf{A}}$ matrices
- implemented in LAGAMINE for sparse systems (KNSYM$=4$)

Table of contents

1. Introduction
2. Direct methods
3. Reordering
4. Iterative methods
5. Conclusion

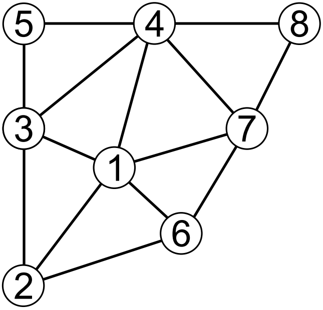
Definitions

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

	1	2	3	4	5	6	7	8
1	■	■	■	■		■	■	
2	■	■	■			■		
3	■	■	■	■	■			
4	■		■	■	■		■	■
5			■	■	■			
6	■	■				■	■	
7	■			■		■	■	■
8				■			■	■



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

	1	2	3	4	5	6	7	8
1	🟦	🟦	🟦	🟦		🟦	🟦	
2	🟥	🟪	🟪	🟩		🟪	🟩	
3	🟥	🟪	🟪	🟪	🟦	🟩	🟩	
4	🟥	🟩	🟪	🟪	🟦	🟩	🟪	🟦
5			🟦	🟦	🟦			
6	🟥	🟪	🟩	🟩		🟪	🟪	
7	🟥	🟩	🟩	🟪		🟪	🟪	🟦
8				🟦			🟦	🟦

Elimination of node 1 :

- red : entries to be cancelled
- green : previously zero entries
- purple : modified entries

$M^i = 5$ (in red)

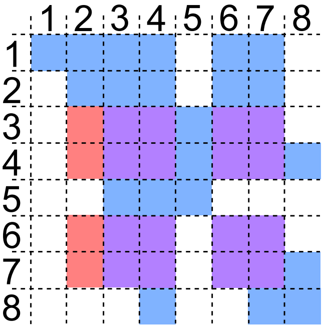
$\Rightarrow$ 25 operations (green + purple)

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 :

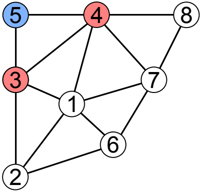
- red : entries to be cancelled
- green : previously zero entries
- purple : modified entries

$M^i = 4$ (in red)

$\Rightarrow$ 16 operations (green + purple)

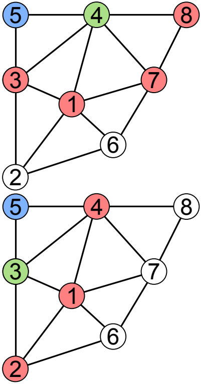
Basic Algorithm

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



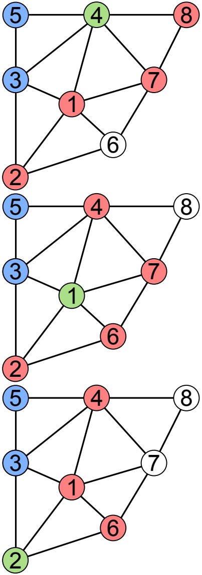
Basic Algorithm

1. Starting from one node to be eliminated (blue, to be adequately chosen)
2. 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



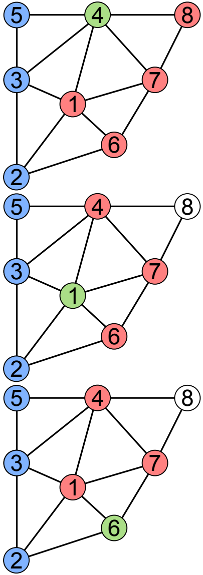
Basic Algorithm

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



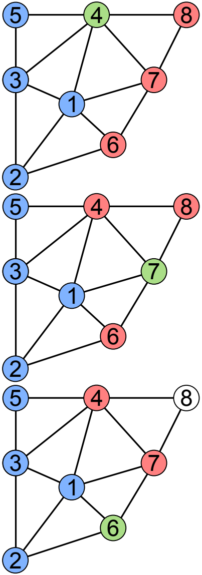
Basic Algorithm

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



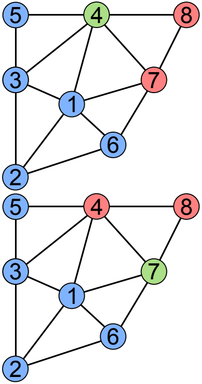
Basic Algorithm

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



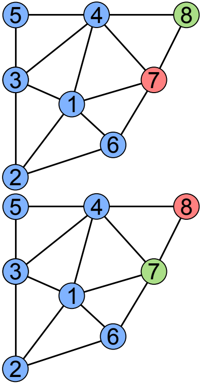
Basic Algorithm

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



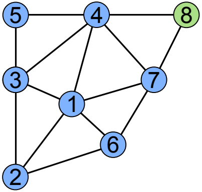
Basic Algorithm

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



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

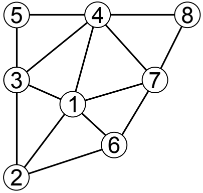


Basic Algorithm

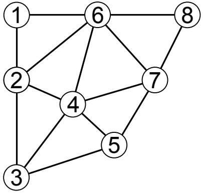
1. Starting from one node to be eliminated (blue, to be adequately chosen)
2. Chosing the next one
   - several possibilities (green)
   - for each possibility : how many new active nodes (red) ?
   - the next node is the one which leads to minimum new active nodes
   - if equality chose the one which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

Sequence : 5,3,2,1,6,4,7,8

Initial

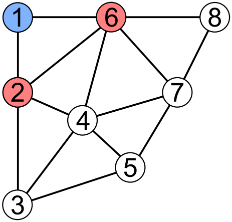


Reordered

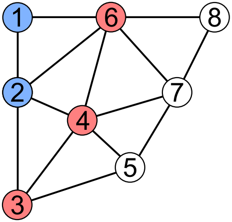


Relation between adjacent nodes and number of operations

Number of adjacent nodes $= M^i$ then minimizing it minimizes σ and cpu time



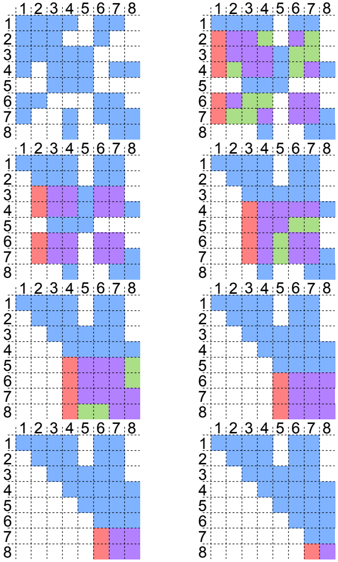
	1	2	3	4	5	6	7	8
1	🟦	🟦				🟦		
2	🟥	🟪	🟦	🟦		🟪		
3		🟦	🟦	🟦	🟦			
4		🟦	🟦	🟦	🟦	🟦	🟦	
5			🟦	🟦	🟦		🟦	
6	🟥	🟪		🟦		🟪	🟦	🟦
7				🟦	🟦	🟦	🟦	🟦
8						🟦	🟦	🟦



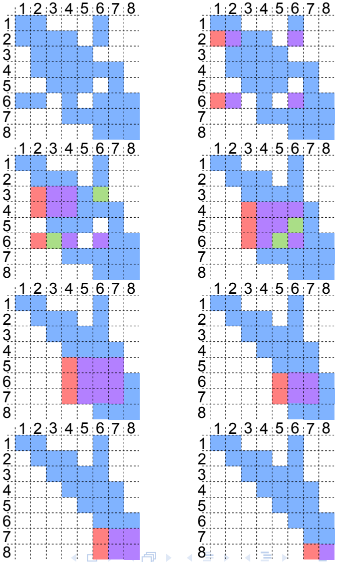
	1	2	3	4	5	6	7	8
1	🟦	🟦				🟦		
2		🟦	🟦	🟦		🟦		
3		🟥	🟪	🟪	🟦	🟩		
4		🟥	🟪	🟪	🟦	🟦	🟦	
5			🟦	🟦	🟦		🟦	
6		🟥	🟩	🟪		🟪	🟦	🟦
7				🟦	🟦	🟦	🟦	🟦
8						🟦	🟦	🟦

Comparison between old and new matrices

Without renum $\sigma = 87$



With renum $\sigma = 40$

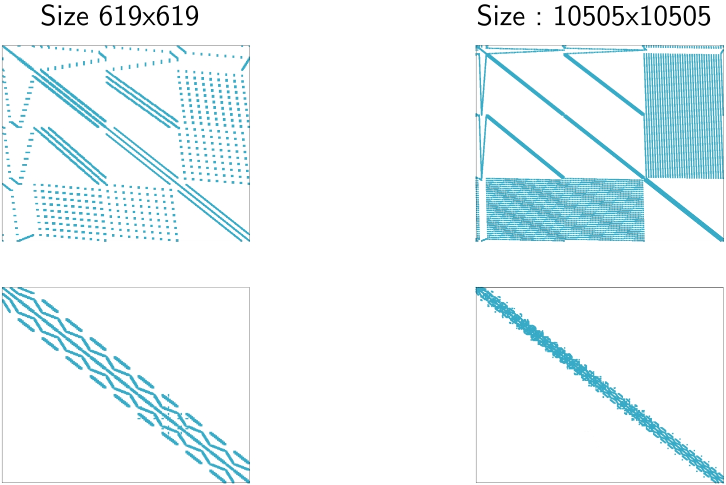


Non zero entries of matrices (before and after renum)

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 (?).

Non zero entries of matrices (before and after renum)

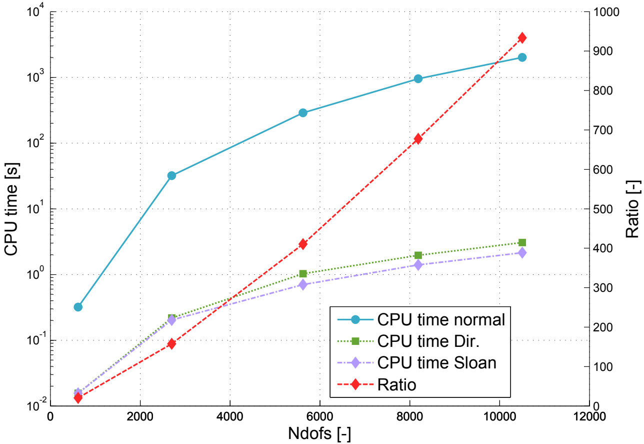


Size 619x619

Size : 10505x10505

Example

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



Bibliography

E. Cuthill and J. McKee, 'Reducting the bandwidth of sparse symmetric matrices', *Proc. ACM Nat. Conf.*, Association of Computing Machinery, New-York (1969)

I.P. King, 'An automatic reordering scheme for simultaneous equations derived from network systems', *International journal of numerical methods in engineering*, **2**, 523-533 (1970)

S.W. Sloan and M.F. Randolph, 'Automatic element reordering for finite element analysis with frontal solution schemes', *International journal of numerical methods in engineering*, **19**, 1153-1181 (1983)

S.W. Sloan, 'An algorithm for profile and wavefront reduction of sparse matrices', *International journal of numerical methods in engineering*, **23**, 239-251 (1986)

S.W. Sloan and W.S. Ng, 'A direct comparison of three algorithms for reducing profile and wavefront', *Computers & Structures*, **33**, 411-4191 (1989)

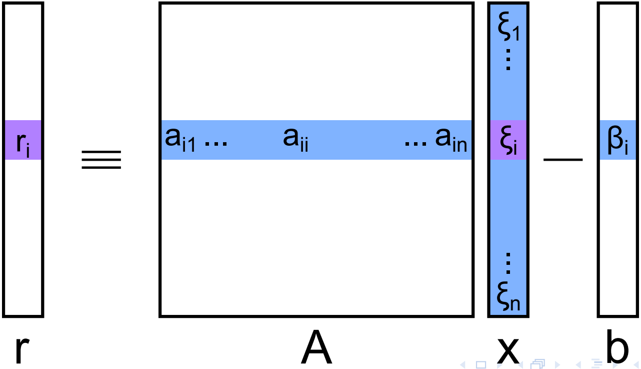
J.A. Scott, 'On ordering elements for a frontal solver', Communications in Numerical methods in engineering', *Communications in numerical methods in engineering*, **15**, 309-323, (1999)

Table of contents

Example : Jacobi iterations

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

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



Example : Jacobi iterations

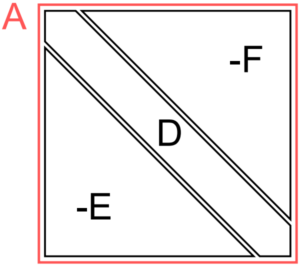
Principle : finding the new value of $\xi_i^{(k)}$ that annihilates the i-th component of the residual vector $r_i = (\mathbf{b} - \underline{\mathbf{A}} \cdot \mathbf{x}_k)_i$

$$
\begin{aligned}
r_i \quad &\equiv \quad 0 \quad = \quad a_{ii} \cdot \boxed{\xi_i^{(k+1)}} + \sum_{\substack{j=1 \\ j \neq i}}^{n} a_{ij} \cdot \xi_j^{(k)} - \beta_i \\
\Leftrightarrow \quad \xi_i^{(k+1)} \quad &= \quad \frac{1}{a_{ii}} \left(\beta_i - \sum_{\substack{j=1 \\ j \neq i}}^{n} a_{ij} \cdot \xi_j^{(k)} \right) \qquad \text{for} \quad i = 1, \ldots n
\end{aligned}
$$

Example : Jacobi iterations

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

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



$$\underline{\mathbf{A}} = \underline{\mathbf{D}} - \underline{\mathbf{E}} - \underline{\mathbf{F}}$$

Rewritten in vector form

$$\mathbf{x}_{k+1} = \underline{\mathbf{D}}^{-1} \cdot (\underline{\mathbf{E}} + \underline{\mathbf{F}}) \cdot \mathbf{x}_k + \underline{\mathbf{D}}^{-1} \cdot \mathbf{b}$$

where

- $\underline{\mathbf{D}}$: diagonal of $\underline{\mathbf{A}}$
- $-\underline{\mathbf{E}}$: lower triangular part of $\underline{\mathbf{A}}$
- $-\underline{\mathbf{F}}$: upper triangular part of $\underline{\mathbf{A}}$

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 $\underline{\mathbf{M}}$:

- close to $\underline{\mathbf{A}}$;
- 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 $\underline{\mathbf{A}}$

$$\underline{\mathbf{A}} = \underline{\mathbf{L}} \cdot \underline{\mathbf{U}} - \underline{\mathbf{R}}$$

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)

Applied to the left

$$\underline{\mathbf{M}}^{-1} \cdot \underline{\mathbf{A}} \cdot \mathbf{x} = \underline{\mathbf{M}}^{-1} \cdot \mathbf{b}$$

or applied to the right

$$\underline{\mathbf{A}} \cdot \underline{\mathbf{M}}^{-1} \cdot \mathbf{u} = \mathbf{b} \qquad \mathbf{x} \equiv \underline{\mathbf{M}}^{-1} \cdot \mathbf{u}$$

Table of contents

1. Introduction
2. Direct methods
3. Reordering
4. Iterative methods
   - Introduction and principles
   - GMRES method
   - Practical use
5. Conclusion

Principle

Building **iteratively** a solution ($\tilde{\mathbf{x}}$)

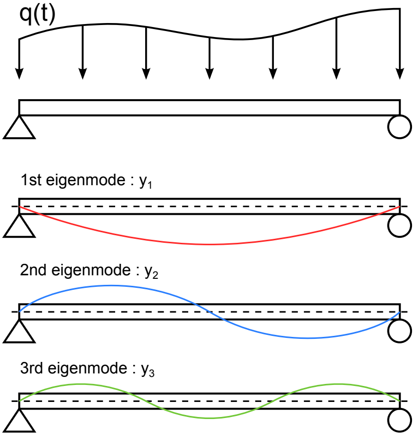
- to the problem $\underline{\mathbf{A}} \cdot \mathbf{x} = \mathbf{b}$
- from an initial guess ($\mathbf{x}_0$) of the solution
- and a basis of **linearly independent vectors** $\mathcal{K}_m$
- through the following **linear combination** :

$$\tilde{\mathbf{x}} = \mathbf{x}_0 + \sum_{i=1}^{m} y_i \cdot \mathbf{v}_i \qquad \mathbf{v}_i \in \mathcal{K}_m$$

The size of the basis, m

- is unknown, *a priori* (iterative concept)
- is much less than N, the number of equations.

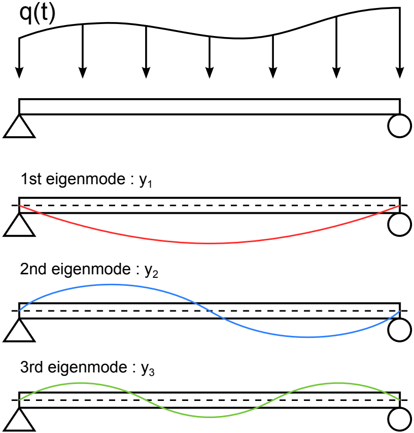
Analogy with structural analysis



There is two ways for solving the problem of a beam submitted to an arbitrary time-dependent loading

1. Nodal base method : solving a NxN system

Analogy with structural analysis



There is two ways for solving the problem of a beam submitted to an arbitrary time-dependent loading

1. Nodal base method : solving a NxN system
2. Modal base method : projection onto eigenmodes (increasing eigenfrequencies)

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

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.*

$$\mathcal{K}_m(\underline{\mathbf{A}}, \mathbf{v}) \equiv \mathrm{span}\left\{\mathbf{v}, \underline{\mathbf{A}}\mathbf{v}, \underline{\mathbf{A}}^2\mathbf{v}, \ldots, \underline{\mathbf{A}}^{m-1}\mathbf{v}\right\}$$

and specifically, GMRES uses

$$\mathcal{K}_m(\underline{\mathbf{A}}, \mathbf{r_0}) \equiv \mathrm{span}\left\{\mathbf{r_0}, \underline{\mathbf{A}}\mathbf{r_0}, \underline{\mathbf{A}}^2\mathbf{r_0}, \ldots, \underline{\mathbf{A}}^{m-1}\mathbf{r_0}\right\}$$

where

$$\mathbf{r}_0 = \mathbf{b} - \underline{\mathbf{A}} \cdot \mathbf{x}_0$$

$$\underline{\mathbf{A}} \in \mathbb{R}^{n \times n} \qquad \text{and} \qquad \mathbf{v}, \mathbf{r_0} \in \mathbb{R}^n$$

Which basis ?

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

Then following a Gram-Schmidt procedure for orthogonal bases

1. $\mathbf{v}_1 = \dfrac{\mathbf{r}_0}{\|\mathbf{r}_0\|_2}$

Which basis ?

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

Then following a Gram-Schmidt procedure for orthogonal bases

1. $\mathbf{v}_1 = \dfrac{\mathbf{r}_0}{\|\mathbf{r}_0\|_2}$
2. actual basis of degree m, compute $\mathbf{v}_{m+1} = \underline{\mathbf{A}} \cdot \mathbf{v}_m$

Which basis ?

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

Then following a Gram-Schmidt procedure for orthogonal bases

1. $\mathbf{v}_1 = \dfrac{\mathbf{r}_0}{\|\mathbf{r}_0\|_2}$
2. actual basis of degree m, compute $\mathbf{v}_{m+1} = \underline{\mathbf{A}} \cdot \mathbf{v}_m$
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.* $\mathbf{v}_j = \underline{\mathbf{A}} \cdot \mathbf{v}_{j-1}$

Then following a Gram-Schmidt procedure for orthogonal bases

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

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

4. normalize $\mathbf{v}_{m+1} = \dfrac{\mathbf{v}_{m+1}}{\|\mathbf{v}_{m+1}\|_2}$

How to compute $\mathbf{y}$? How to choose the size of m ?

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

$$\tilde{\mathbf{x}} = \mathbf{x}_0 + \underline{\mathbf{V}}_m \cdot \mathbf{y}$$

where $\mathbf{y}$ minimizes

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

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

$$\|\mathbf{r}_i\|_2 < \epsilon \cdot \|\mathbf{r}_0\|_2$$

Principle

GMRES method implemented in LAGAMINE (KNSYM $= \pm 6$)

- approached solution (depends on the convergence norm !)
- no need to inverse the global matrix $\underline{\mathbf{A}}$
- 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 $\underline{\mathbf{A}}$
- requires input parameters

Table of contents

1 Introduction

2 Direct methods

3 Reordering

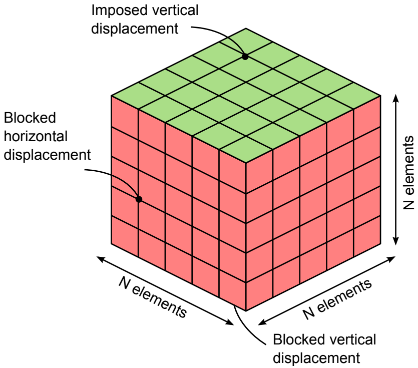
4 Iterative methods
- Introduction and principles
- GMRES method
- Practical use

5 Conclusion

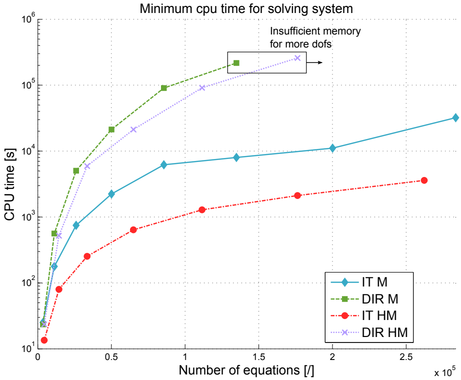
Practical use : oedometer cube

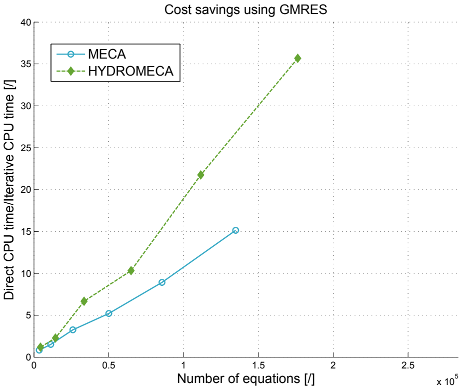
Systematic comparison using a cubic example

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

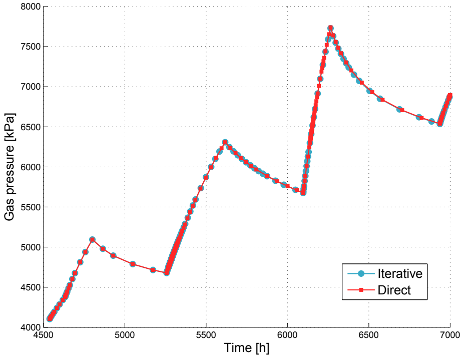

Imposed vertical displacement
Blocked horizontal displacement
N elements
N elements
N elements
Blocked vertical displacement

Practical use : oedometer cube





Practical use : gas injection



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

Parameters

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}

$$m_{kl} = \frac{a_{kl}}{a_{kk}}$$
$$a_{kj} \leftarrow a_{kj} - m_{kl} \cdot a_{kj} \quad b_k \leftarrow b_k - m_{kl} \cdot b_k \qquad \text{for} \qquad j = 1, \dots, n$$

If $|m_{kl}| <$ DROPTOL nothing is done and $a_{kl} \leftarrow 0$

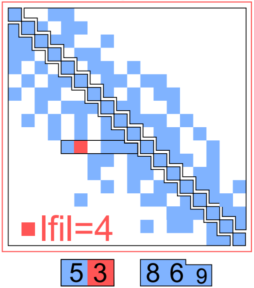
Parameters

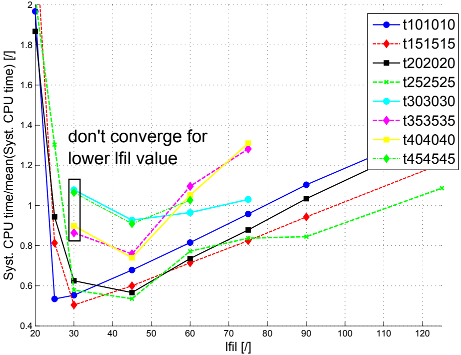
Preconditioning parameters : **LFIL**, DROPTOL

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





don't converge for lower lfil value

$$\frac{CPU\ \ time(lfil_i)}{\frac{1}{n}\cdot\sum_{j=lfil_1}^{lfil_n} CPU\ \ time_j}$$

Parameters

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

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 $\mathbf{x}_0 = \mathbf{x}_m$,

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 $\mathbf{x}_0 = \mathbf{x}_m$,

The total number of iterations (n_{its}) is equal to the number of orthogonalization processes already done which is equal

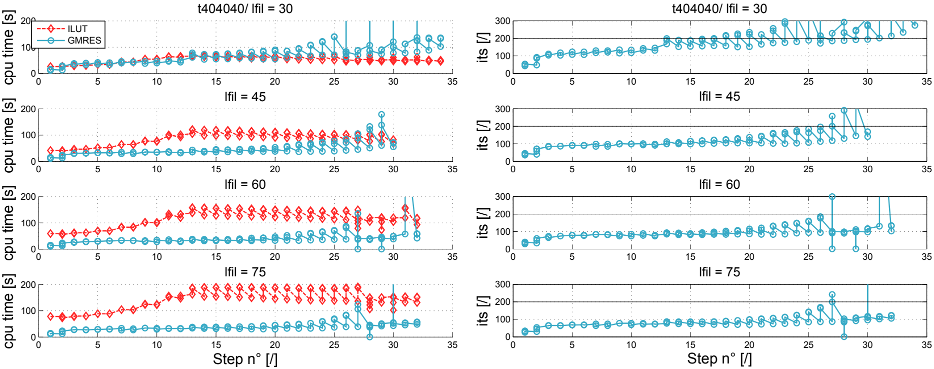
- to the actual size k of the Krylov process if $n_{its} < \mathrm{IM}$;
- to $n_{restart} \times \mathrm{IM} + k$ if $n_{its} > \mathrm{IM}$.

If no convergence occurs after $n_{its} = \mathrm{MAXITS}$

- 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



Parameters

Convergence parameters : EPS, RESTOL
The internal iterative process ends if (**relative** convergence)

$$\|\mathbf{r}_i\|_2 < \mathrm{EPS} \cdot \|\mathbf{r}_0\|_2$$

or if (**absolute** convergence)

$$\|\mathbf{r}_i\|_2 < \mathrm{RESTOL}$$

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

Example of configurations

RESTOL is imposed to 10^{-40}

NDOFS	TYPE	LFIL	DROPTOL	IM	MAXITS	EPS
283544	cube M	45	10^{-6}	200	500	10^{-5}
220940	gallery M	130	10^{-6}	500	4000	10^{-5}
262236	cube HM	20	10^{-6}	200	500	10^{-5}
752226	loca M	**200**	10^{-8}	500	1000	10^{-5}

High LFIL with high number of dofs to avoid **instabilities** !

Bibliography

Mainly for GMRES and iterative methods (available online)

Y. Saad, 'Iterative methods for sparse linear systems', Society for industrial and applied mathematics, (2003)

Or a summary
Y Saad and M.H. Schultz, 'GMRES : a generalized minimal residual algorithm for solving nonsymmetric systems', *Journal on Scientific Computing*, **7**, 856-869 (1986)

Table of contents

1 Introduction

2 Direct methods

3 Reordering

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

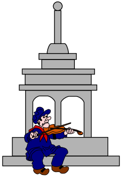
5 Conclusion

- 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

Université de Liège

GMO MECA

FSA, créatrice d'avenirs



fnrs

LA LIBERTÉ DE CHERCHER