CEMRACS Project ElastoΦ

Boundary integral methods for elasticity around a crack network

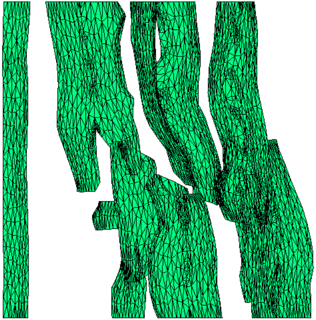
Marcella Bonazzoli[1], Pierre Marchand[2], Xavier Claeys[2], Ibtihel Ben Gharbia[3], Pierre-Henri Tournier[2], Frédéric Nataf[2]

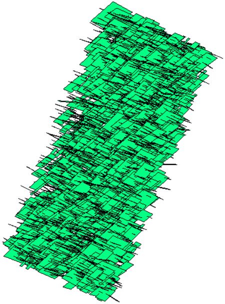
[1]Labo. J.A. Dieudonné, Univ. Nice Sophia Antipolis,
[2]INRIA Alpines / Labo. J.-L. Lions UPMC,
[3]IFP Energies Nouvelles.

August 25, 2016

The ElastoΦ project: the IFPEN problem

Elastostatic problem in crack networks of 2 types: geological *fault* network and discrete *fracture* network





Boundary *integral equation* posed at the surface of cracks
$\Rightarrow$ dense matrix $\mathrm{A} \in \mathbb{R}^{n \times n} \Rightarrow \mathcal{O}(n^2)$ for matrix-vector product.

IFPEN heuristic approach to sparsify A gives large error (16%–40%)

The ElastoΦ project

Many refined *complexity reduction* techniques in current literature on boundary integral equation.

Two ingredients in the approach we considered:

- Adaptative Cross Approximation (ACA),
- Hierarchical Matrices (HM).

Challenge: *strongly irregular geometry!*

[M. Bebendorf. Hierarchical matrices: A Means to Efficiently Solve Elliptic Boundary Value Problems, *Lecture Notes in Computational Science and Engineering*, 2008]

[S. Rjasanow, O. Steinbach. The fast solution of boundary integral equations. *Mathematical and Analytical Techniques with Applications to Engineering*, 2007]

[W. Hackbusch. Hierarchical Matrices: Algorithms and Analysis, *Springer Series in Computational Mathematics*, 2016]

Suppose that $\mathrm{A} \in \mathbb{R}^{n\times n}$ is of low rank, i.e.

$$\mathrm{A} = \sum_{j=1}^{k} \mathbf{u}_j \cdot \mathbf{v}_j^T \quad \text{with} \quad k \leq n/2.$$

$\Rightarrow$ $2kn$ operations for matrix-vector product.

SVD actually gives the following decomposition:

$$\mathrm{A} = \sum_{j=1}^{n} \sigma_j \, \mathbf{u}_j \cdot \mathbf{v}_j^T \quad \text{where } \{\sigma_j^2\}_{j=1\dots n} \text{ are the eigenvalues of } \mathrm{A}^T\mathrm{A}.$$

If the σ_j decrease fast, *truncated SVD* is a good approximation of A!

But it costs $\mathcal{O}(n^3)$ ...

Adaptative Cross approximation (ACA)

An approximation of the Singular Value Decomposition (SVD)

Algorithm 1 Partially Pivoted ACA

Initialize j_* $r = 0$

while(stopping criterion not satisfied){

$\mathbf{w} = \mathrm{A}(j_*, :)^T - \sum_{\ell=1}^{r} \mathbf{u}_\ell(j_*)\, \mathbf{v}_\ell$

$k_* = \mathrm{argmax}_{k=1\ldots n} |\mathbf{w}(k)|$

$w_* = \mathbf{w}(k_*)$

if$(w_* \neq 0)\{$

$\mathbf{v}_{r+1} = \mathbf{w}$

$\mathbf{w} = \mathrm{A}(:, k_*) - \sum_{\ell=1}^{r} \mathbf{v}_\ell(k_*)\, \mathbf{u}_\ell$

$\mathbf{u}_{r+1} = w_*^{-1} \mathbf{w}$

$j_* = \mathrm{argmax}_{j=1\ldots n} |\mathbf{w}(j)|$

$r = r + 1$

}

else{terminate algorithm}

}

Advantages:

- No need to assemble the entire matrix
- Cost and storage in $\mathcal{O}(n \log n)$

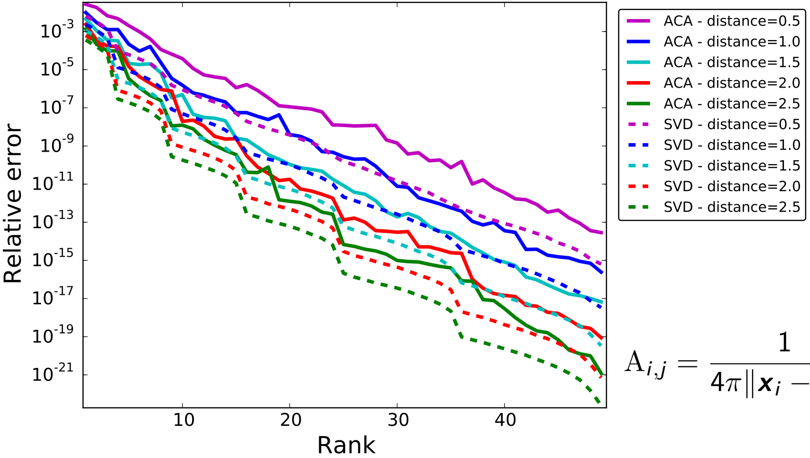
Remark:

- Function computing the coefficients on the fly necessary to be more efficient in practice

Adaptative Cross Approximation (ACA)

Comparaison between SVD and ACA

Compression of the interaction matrix between *two clusters* $\{\boldsymbol{x}_i\}$ and $\{\boldsymbol{y}_j\}$



$$\mathrm{A}_{i,j} = \frac{1}{4\pi\|\boldsymbol{x}_i - \boldsymbol{y}_j\|}$$

The matrix comes from the discretization of a boundary integral equation

$$\mathrm{A}_{j,k} := \int_{\tau\times\tau'} \mathscr{G}(\boldsymbol{x}-\boldsymbol{y})\psi_j(\boldsymbol{x})\psi_k(\boldsymbol{y})d\sigma(\boldsymbol{x})d\sigma(\boldsymbol{y}), \quad j,k = 1\ldots n.$$

$\mathscr{G}$ is an *integral kernel* with these properties:

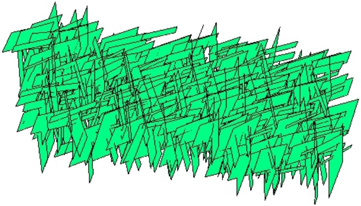
- it is *singular* for $\boldsymbol{x} = \boldsymbol{y}$, i.e. if $\tau \cap \tau' \neq \emptyset$,
- it is *regularizing* if τ and τ' are distant from each other.

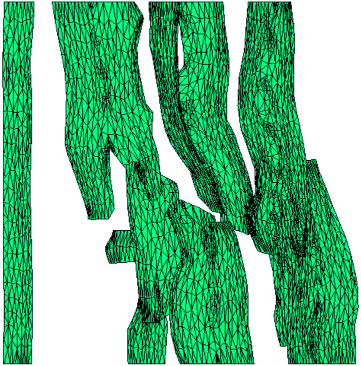
$\Rightarrow$ ACA is applicable to admissible blocks of A

$\Rightarrow$ Hierarchical Matrices (HM)

1. First step to build the blocks of the hierarchical matrix: building a *tree* of clusters of points.

The unknowns correspond to mesh element barycenters.



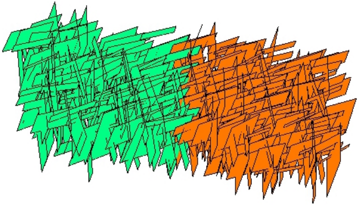


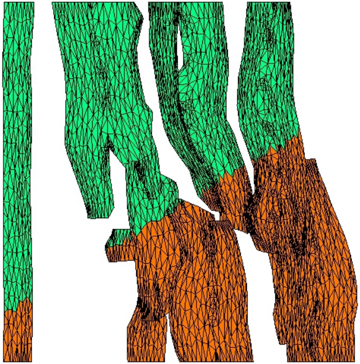
Hierarchical matrices (HM)

An example of cluster tree

1. First step to build the blocks of the hierarchical matrix: building a *tree* of clusters of points.

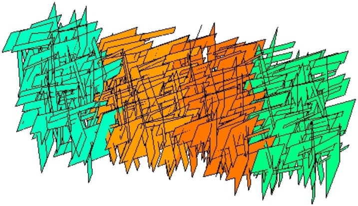
The unknowns correspond to mesh element barycenters.





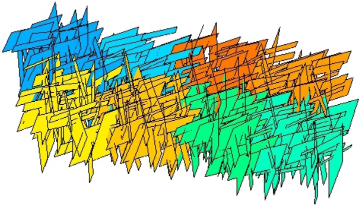
1. First step to build the blocks of the hierarchical matrix: building a *tree* of clusters of points.

The unknowns correspond to mesh element barycenters.



1. First step to build the blocks of the hierarchical matrix: building a *tree* of clusters of points.

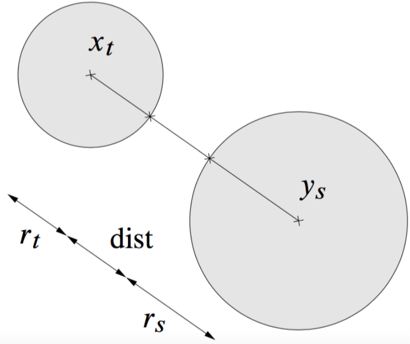
The unknowns correspond to mesh element barycenters.



Interactions between nodes of the cluster tree $\Leftrightarrow$ sub-blocks of A.

2. Second step: checking *recursively* the admissibility of the blocks with the following admissibility condition on the associated clusters:

$$\min(\mathrm{diam}(\mathrm{B}_t), \mathrm{diam}(\mathrm{B}_s)) < \eta \,\mathrm{dist}(\mathrm{B}_t, \mathrm{B}_s)$$



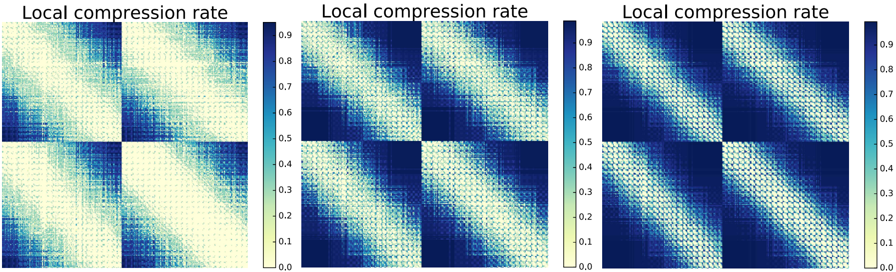
$\Rightarrow$ apply ACA to the admissible blocks.

Results

C++ implementation, our source code is available on GitHub at:

`https://github.com/xclaeys/ElastoPhi`

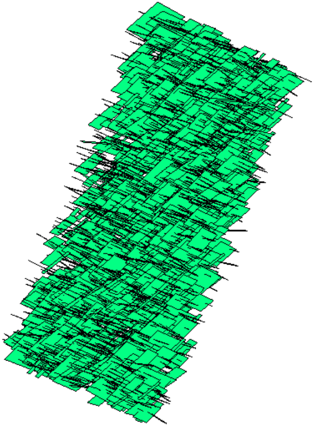
Varying the parameters of the algorithm:

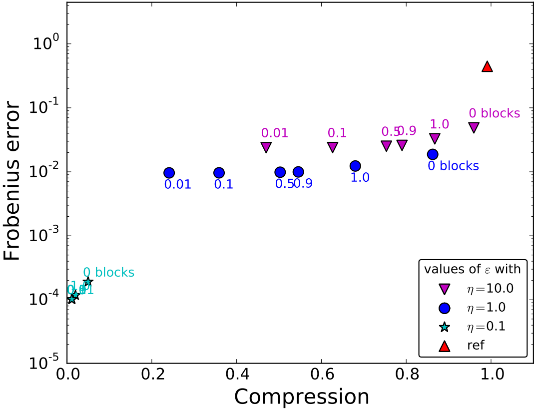


- Remark: a block is not necessarily a connected part of the matrix.

Results

Network of $N = 1994$ fractures: size $\mathrm{A} = 3N \times 3N$





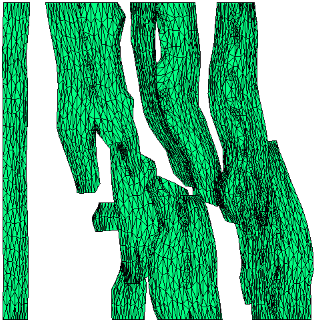
IFPEN sparse matrix: compression rate $= 0.99$, error $= 45\%$

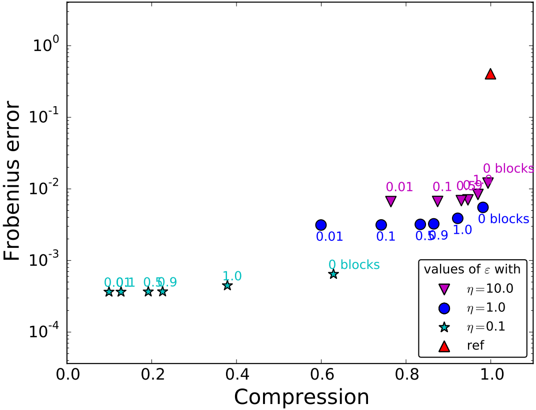
H matrix, zero blocks and $\eta = 1$: compression rate $= 0.86$, error $= 2\%$

HM-ACA matrix, $\varepsilon = 1$ and $\eta = 1$: compression rate $= 0.68$, error $= 1\%$

Results

Network of faults with $N = 5364$ mesh triangles: size $\mathrm{A} = 3N \times 3N$





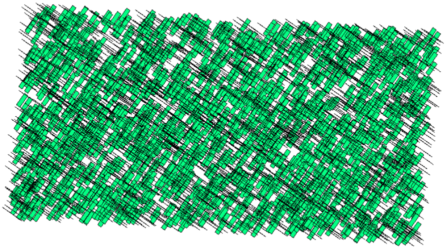
IFPEN sparse matrix: compression rate $= 0.999$, error $= 41\%$

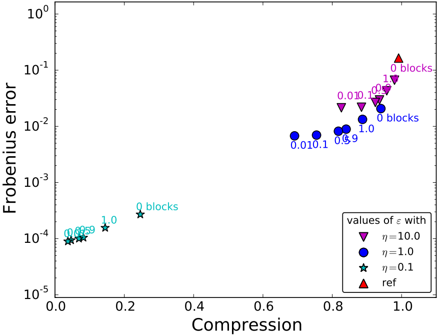
H matrix, zero blocks and $\eta = 1$: compression rate $= 0.98$, error $= 0.55\%$

HM-ACA matrix, $\varepsilon = 1$ and $\eta = 1$: compression rate $= 0.92$, error $= 0.38\%$

Results

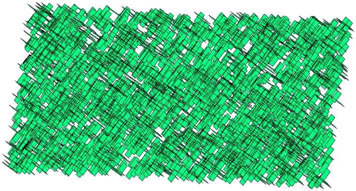
Network of $N = 3600$ fractures: size $\mathrm{A} = 3N \times 3N$

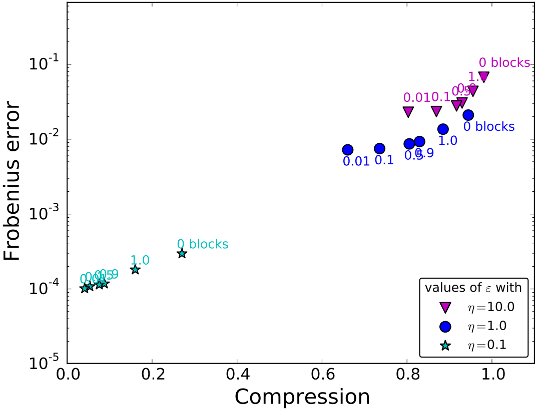




Results

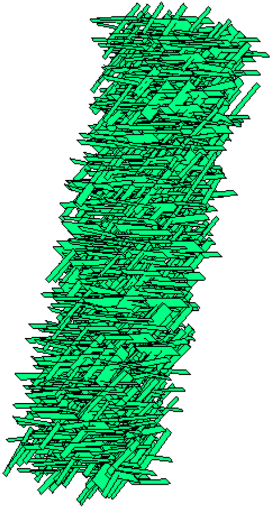
Network of $N = 2700$ fractures: size $\mathrm{A} = 3N \times 3N$

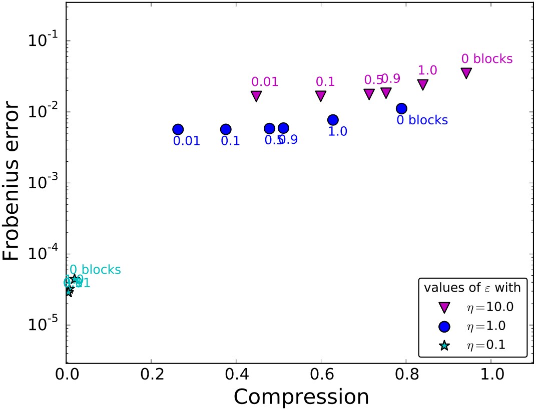




Results

Network of $N = 1363$ fractures: size $\mathrm{A} = 3N \times 3N$





Conclusion and outlook

- We obtained approximation errors suited to IFPEN applications with high compression rates,
- to treat large industrial cases, we have to compute *only the needed coefficients* of the matrix *on the fly*, in order to really exploit the low complexity of ACA
  (not possible during CEMRACS for confidentiality reasons),
- design a *new admissibility condition* to take in account the *direction* of the fractures,
- *parallelization* of the code.

Conclusion and outlook

- We obtained approximation errors suited to IFPEN applications with high compression rates,
- to treat large industrial cases, we have to compute *only the needed coefficients* of the matrix *on the fly*, in order to really exploit the low complexity of ACA
(not possible during CEMRACS for confidentiality reasons),
- design a *new admissibility condition* to take in account the *direction* of the fractures,
- *parallelization* of the code.

Thank you for your attention!