Code_Aster HPC
Tips and Tricks for Nonlinear Analysis

EDF R&D - Saclay

Table of contents

Prerequisites

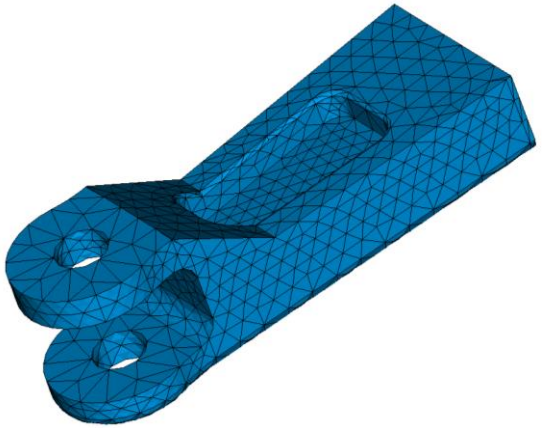
- The non-linear simulation must **robustly** converge!
  - You cannot optimize the performance of a fragile simulation
  - All tweaks will slightly disturb the solution procedure and may eventually lead to divergence if the simulation is not robust
- As a corollary, you must use the automatic time stepping **`DEFI_LIST_INST`**

The good – the linear solver

- Check the previous talk on the linear solvers in this training ;o)
- Use the parallel version of Code_Aster and try the default iterative solver
  - **SOLVEUR=_F(METHODE='PETSC')**
    - Krylov method : Flexible Generalized Minimum RESidual (FGMRES)
    - Preconditioner : **LDLT_SP**, single precision factorization provided by the MUMPS direct solver, kept constant for several resolutions and refactored only when needed
  - Expensive approach compared to optimal preconditioners…but bullet proof!
  - Warning : Can fail on contact problems!

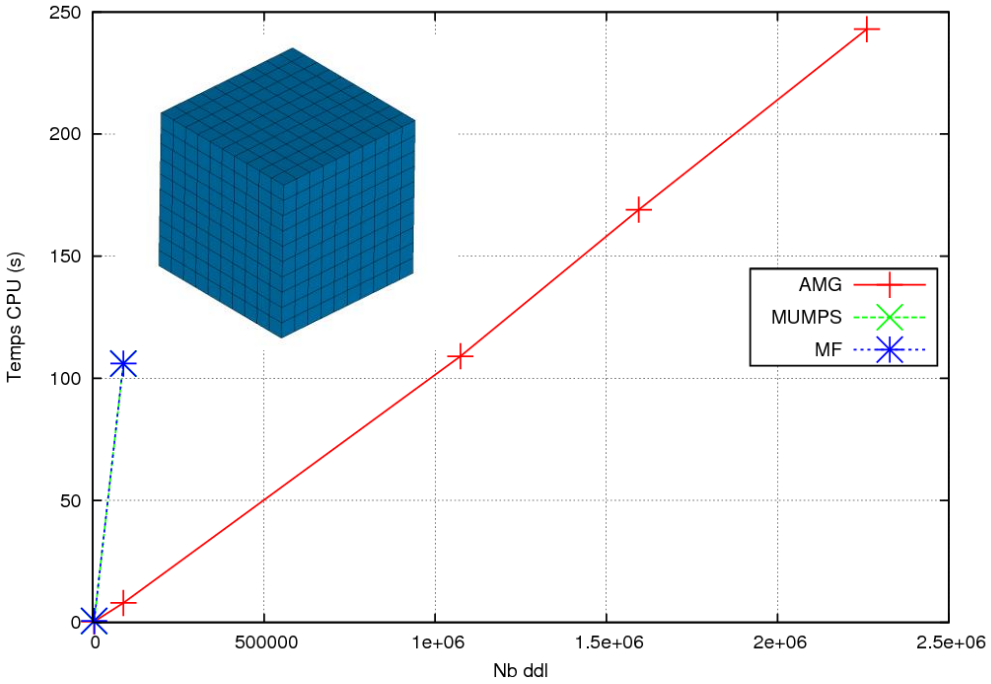
The good – the linear solver

- Elasto-plastic bearing submitted to 10 load-steps
  - MUMPS : 3800 s, 1.2 Go [tens of factorizations]
  - LDLT_SP : 850 s, 630 Mo [a single factorization]
- 46% memory gain
- 78% CPU time gain



The good – the linear solver

- If you are a lucky guy…
  - No Lagrange multipliers
  - Only 3D elements or only displacement degrees of freedom
  - Try a multigrid preconditioner!
    - **`PRE_COND='GAMG'`**

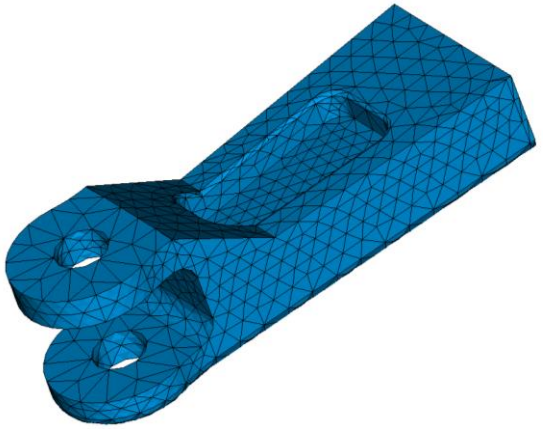


Optimal complexity in time!
- O(n) for multigrid
- $O(n^{7/2})$ for direct solvers

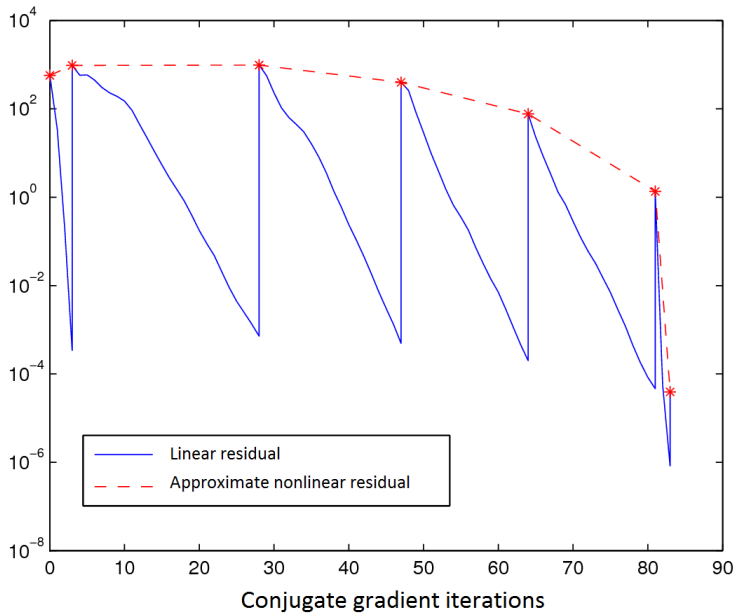
The bad – the non-linear solver

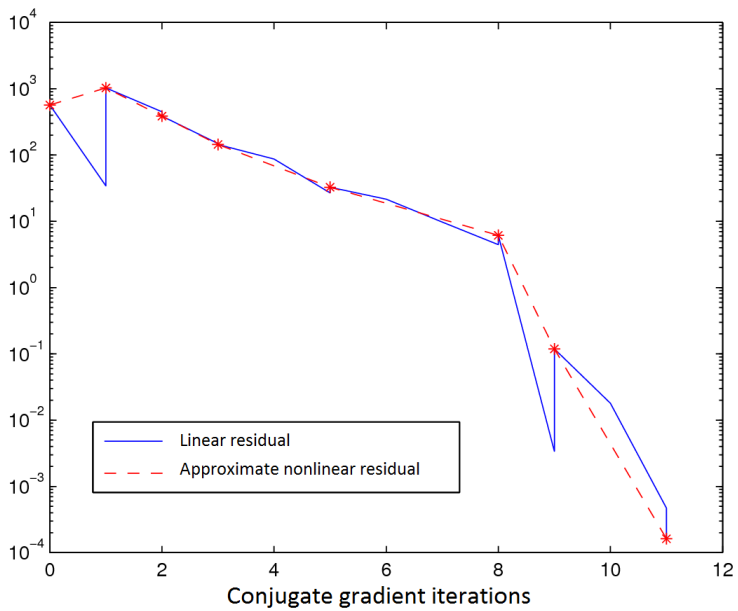
- In conjunction with iterative solvers, the Newton-Krylov approach is an alternative to the Newton's method
  - Rather than solving $K^{n-1}\delta u^n = -R^{n-1}$, solve a weaker condition $\|K^{n-1}\delta u^n + R^{n-1}\| \leq \alpha\|R^{n-1}\|$, α being the 'forcing term', evolving at each iteration
    - $\alpha \ll 1$: returns a solution equal to the classical Newton's step
    - $\alpha \gg 1$: returns an approximate solution of the linearized system close to the previous one u^{n-1}
  - The strategy : reduce α during each non-linear step
    - Solve approximately the linearized system where far from the solution (where the quadratic approximation might be false)
    - Solve precisely the linearized system where close to the solution in order to recover the quadratic convergence
  - Save iterations on the linear stage
  - Gain robustness on the nonlinear stage

The bad – the non-linear solver



- Illustration for the elasto-plastic bearing
  - Reduction of the iterative solver iterations by a factor of 8

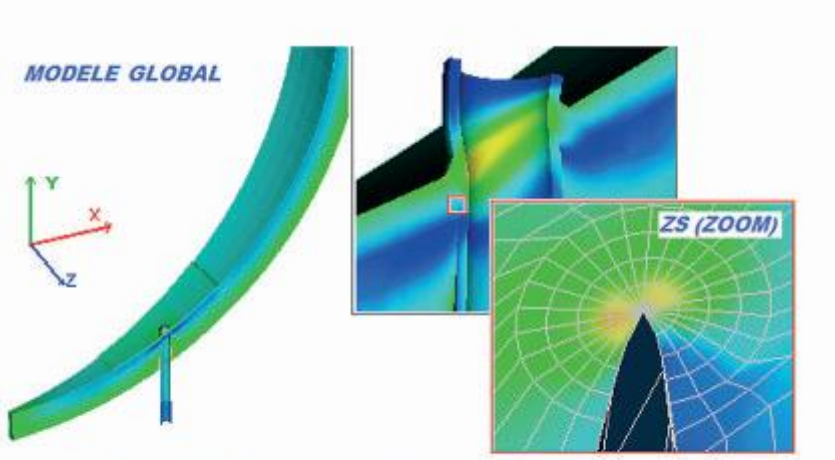




The bad – the non-linear solver

- Code_Aster usage
  - **METHODE='NEWTON_KRYLOV'** in **STAT_NON_LINE**
- For contact problems, can gain robustness in conjunction with **LDLT_SP**

- Study of the heating cans of a pressurizer
  - Contact + elasto-plasticity + 750 000 DOF
  - More than 8h with **NEWTON+MUMPS**
  - 3h30 with **NEWTON_KRYLOV+LDLT_SP**



The ugly - the parallelism

- The all-road way : use the MUMPS direct solver
  - Robust
  - Guaranteed CPU gain up to some tens of procs

- The more tricky way : all previous strategies are parallel
  - Back to the heating cans of a pressurizer solved with **`NEWTON_KRYLOV+LDLT_SP`**
    - 3h30 in sequential
    - 35min for 64 procs

Thank you for your attention

- Questions?