Some engineering issues in the MUMPS project

Guillaume Joslin, CERFACS-IRIT common laboratory
Clamart, 30 May 2013
Engineering work needed by both Users and Developers:

- **For Users (support or requests)**
  - Features: extension, stabilization or new ones
  - Installation, characterization/reproducibility of problems, bug tracking
  - Redesign specification sheets

- **For Developers/Researchers (and to support research transfer)**
  - Improve / automatize procedures to reproduce user error and to analyze performance
  - Support to transfer of new features from research to production
  - Extension of validation tests
  - Redesign website
  - Production of a new release (of interest also to users!!)
Outline

Illustration of users oriented features
   Restarting feature
   Null space feature
   Compatibility with external libraries

Developers oriented features
   Validation tool
   Experimentation tool
   Experimentations and visualizations scripts
   other engineering issues
Outline

Illustration of users oriented features
- Restarting feature
- Null space feature
- Compatibility with external libraries

Developers oriented features
- Validation tool
- Experimentation tool
- Experimentations and visualizations scripts
- Other engineering issues
Outline

Illustration of users oriented features
  Restarting feature
  Null space feature
  Compatibility with external libraries

Developers oriented features
  Validation tool
  Experimentation tool
  Experimentations and visualizations scripts
  other engineering issues
Restarting MUMPS

A New feature

- Wish list of the MUMPS users days 2010 and frequent request since the last users’ days.

How does it work?

- Main steps without restarting

![Diagram showing steps of analysis, factoring, and solving](image)
A New feature

- Wish list of the MUMPS users days 2010 and frequent request since the last users’ days.

How does it work?

- Main steps with restarting

  - Allow the user to stop MUMPS and to restart at the end of any step.
Restarting: main issues

Restarting more efficient in an Out Of Core context

Parallel makes restarting more complicated

Global file system
### Two new functions

- **CMUMPS_STOP**(mumps_structure, output_file_base)
- **CMUMPS_RESTART**(mumps_structure, input_file_base)

### Simple example (with no restarting)

#### C Initializing Code

```c
mumps_par%JOB = -1
CALL CMUMPS(mumps_par)
```

#### C Initializing user data (Matrix input, MUMPS options ...)

```c
mumps_par%JOB= 4 ! analysis + factorization
CALL CMUMPS(mumps_par)
```

#### C Generation of RHS using data from factorization

```c
mumps_par%JOB= 3 ! solve
CALL CMUMPS(mumps_par)
```

#### C Checking solution, analyzing results...

```c
mumps_par%JOB= -2
CALL CMUMPS(mumps_par)
```
Simple example: Stopping step

C Initializing Code
mumps_par%JOB = -1
CALL CMUMPS(mumps_par)

C Initializing user data (Matrix input, MUMPS options ...)
mumps_par%JOB= 4 ! analysis + factorization
CALL CMUMPS(mumps_par)
CALL CMUMPS_STOP(mumps_par,restarting_file)

Simple example: Restarting step

C Initializing Code
CALL CMUMPS_RESTART(mumps_par,restarting_file)

C Generation of RHS using data from previously stopped factorization
mumps_par%JOB= 3 ! solve
CALL CMUMPS(mumps_par)

C Checking solution, analyzing results...
mumps_par%JOB= -2
CALL CMUMPS(mumps_par)
Concluding remarks

Restarting will be available in MUMPS 5.0

Features

- Simple interface and separate module.
- Sequential and parallel (distributed MPI) feature.
- Compatible and more efficient with OOC.
- Works with parallel Out Of Core.

Current limitation

- Dumped files belong to a global file system.
Outline

Illustration of users oriented features
- Restarting feature
- Null space feature
- Compatibility with external libraries

Developers oriented features
- Validation tool
- Experimentation tool
- Experimentations and visualizations scripts
- Other engineering issues
Improvements of null space detection

- First prototype developed during ANR Solstice.
- Partnership between the MUMPS team via CERFACS and Total.
- Null pivot detection feature already available in the last version of MUMPS was improved.
- Postponing pseudo-singularities to the root node was redesigned.
- Can be applied to any matrix but vital for singular matrices.
Detection of Null Pivots (ICNTL(24)=1)

- If (row norm $r_i < T_{NPD}$ and column norm $c_i < T_{NPD}$) then replace pivot by:
  - a “large value“ (controlled by CNTL(5)),
  - or
  - 1, in which case rest of row/column is set to 0.

- Blocked updates as before but no impact of selected pivot on the rest of the factorization.

For $L D L^t$, we only check (and set) the column.
If best pivot smaller (absolute) than $T_{PS}$, then postpone pivot to root that is pre-split for efficiency.

Two rank-revealing factorizations to manage the root deficiency

- QR algorithm: $\text{Root} \cdot P = Q \cdot R$
- SVD algorithm: $\text{Root} = U \cdot S \cdot V^T$
Null pivot detection and Pseudo-Singularity postponing can be combined.

- Null pivot detection (ICNTL(24)) detects obvious null pivots.
- Rank-revealing factorization at the root node.

\[
T_{NPD} = 10^{-10} \\
T_{PS} = 10^{-8}
\]
### Null pivot detection + Pseudo-singularity postponing

- **Activation**: \( \text{ICNTL}(24) = 1 \) & \( \text{ICNTL}(16) = 1 \)

- **Threshold management**:
  - **Pseudo-singularity postponing**:
    - \( \text{CNTL}(3) < 0 \): \( \text{Postponing Threshold} = |\text{CNTL}(3)| \)
    - \( \text{CNTL}(3) > 0 \): \( \text{Postponing Threshold} = \text{CNTL}(3) \cdot ||A|| \)
    - \( \text{CNTL}(3) = 0 \): \( \text{Postponing Threshold} = N \cdot \epsilon \cdot ||A|| \)

- **Detection of null pivots**:
  - \( \text{CNTL}(6) \in [0,1) \):
    - \( \text{Null Pivot Threshold} = \text{CNTL}(6) \cdot \text{Postponing Threshold} \)
  - \( \text{CNTL}(6) \notin [0,1) \):
    - \( \text{Null Pivot Threshold} = 0.01 \cdot \text{Postponing Threshold} \)

- **Deficiency is returned in INFOG(28)**
Null Space computations

Computing a Null Space Basis

- Find \( x \) such as \( A \cdot x = 0 \)
- based on the list of null pivots obtained by rank revealing on the root, null pivots detection or both.

API description

- \( ICNTL(25) = i, 1 \leq i \leq \text{Deficiency} \)
  - Computes the \( i \)th vector from the null space basis
  - \( i \) can either correspond to a null pivot, or to a null singular value detected at the root
- \( ICNTL(25) = -1 \)
  - Computes full null space basis in \( \text{RHS}(1 : N, 1 : \text{Deficiency}) \)
  - Each vector either corresponds to a null pivot in the sense of \( ICNTL(24) \), or is an expansion of an eigenvector of the ROOT (\( ICNTL(16) \)).
Outline

Illustration of users oriented features
- Restarting feature
- Null space feature
- Compatibility with external libraries

Developers oriented features
- Validation tool
- Experimentation tool
- Experimentations and visualizations scripts
- other engineering issues
**Compatibly with external libraries**

### Metis
- MUMPS 4.10.0 is not compatible with metis 5.x
- Compatibility with metis 5.x and parmetis 4.x assured in MUMPS 5.0
- Compatibility with metis 4.x and parmetis 3.x is kept.
- User friendly solution adopted to limit Makefile manipulation by the users.

### Scotch 6.0
- Work in progress for MUMPS 5.0

### 64 bits integers
- Works in the sequential version of MPI provided in MUMPS. Using -DINTSIZE64 in the Makefile.
- Difficulties with the MPI distributions.
Outline

Illustration of users oriented features
  Restarting feature
  Null space feature
  Compatibility with external libraries

Developers oriented features
  Validation tool
  Experimentation tool
  Experimentations and visualizations scripts
  other engineering issues
Outline

Illustration of users oriented features
  Restarting feature
  Null space feature
  Compatibility with external libraries

Developers oriented features
  Validation tool
  Experimentation tool
  Experimentations and visualizations scripts
  Other engineering issues
Validation tool: night tests

Description

- Nightly usage to check the stability of the code.

Achievements

- Main modules:
  - TM_01_RANGE
  - TM_02_PRINTING
  - TM_03_TRANS
  - TM_04_MEMORIES
  - TM_05_ORDERING
  - TM_06_SCALING
  - TM_07_SOLVE
  - TM_08_SCHUR
  - TM_09_OOC
  - TM_10_NUMERICAL
  - TM_11_PHASES
  - TM_12_KEEP
  - TM_13_AM1
  - TM_14_DET
  - TM_15_SAMTECH
  - TM_16_NULL

- 2 auxiliary modules for error management and miscellaneous routines.

- Template (only 200 lines of Fortran) for creating new series of tests corresponding to new major features.
Outline

Illustration of users oriented features
  Restarting feature
  Null space feature
  Compatibility with external libraries

Developers oriented features
  Validation tool
  **Experimentation tool**
  Experimentations and visualizations scripts
  other engineering issues
## Experimentation tool

### Context

- Already existing heavy-to-use driver.
- Daily use by the MUMPS team (development, debugging, experimentation).

### Objectives and achievements

- Easy-to-use driver for MUMPS developers.
- Access to all MUMPS features.
- Easy evolution following new MUMPS features.
- Reproduce users’ test cases for a better support.
Figure: Null pivot detection analysis
Experimentation tool: example

Pseudo-language illustration

```
1 # Example of input file
2 PAR 1
3 SYM 2
4 JOB -1
5 ICNTL 24 1 #NULL PIVOT DETECTION ON
6 CNTL 3 1E-10 #NULL PIVOT THRESHOLD
7 Matrix_file AUTO mat.mtx
8 job 4
9 ICNTL 25 -1 #NULL SPACE COMPUTATION ON
10 alloc RHS "%INFOG 28"
11 JOB 3
12 check_NULLSPACE
13 CSV_GENERIC
14 JOB -2
15 END
```
Figure: Time for the computation of a null space basis.
Outline

Illustration of users oriented features
  Restarting feature
  Null space feature
  Compatibility with external libraries

Developers oriented features
  Validation tool
  Experimentation tool
  Experimentations and visualizations scripts
  other engineering issues
# A Script for the management of experimentations

## Two main goals

1. validation of real scenarios interactively or on a regular basis
2. performance monitoring over MUMPS revisions

## Features

1. management of the whole chain: MUMPS Makefile configuration, compilation and driver execution
2. experimentations on local host or some remote hosts (with rsync and ssh)
3. sets of scenarios from the product of list and range parameters
4. a relational database for the experimentation tool outputs and MUMPS factorizations traces
Management of experimentations: an example

list and range parameters

```bash
metatest HOSTNAME=localhost,apowerfullmachine \
    OPTF="-O -DSOME_FLAG","-O -DMY_EXPERIMENTAL_FLAG" \
    NBPROCS=1:8 \
    OMP_NUM_THREADS=4,5,6 \
    'ICNTL 7'=1,3,4,5,6,7 \
    'KEEP 1'=1:10 \
    'CNTL 1'=0.01:0.01:0.1 \
    input_file
```
Traces are events on master process or slave process

- events represent different factorization steps: AssemblyBegin, AssemblyEnd, FactoBegin, FactoEnd, StackBegin, StackEnd, ...
- events have common parameters: process id, tree node, current time, current memory consumption
- specific parameters may be attached to particular events

How to visualize the traces?

- events on different processes represent a Gantt chart
- events workflow is from leaves to root in the elimination tree
### Visualizations: some possibilities

**On a tree**

- display of some metrics by process, by node and time
- display of factorization states by process, by node and time
- display of some metrics by process or by node and time
- display of some static metrics by node
- for different factorizations, display of some comparisons or statistics

### Example of application

A numerical issue on a matrix from ESI Group leads to a bad solution

- identified problem with numerical pivoting strategy
- design of a strategy to improve the numerical behavior
- study of the compared behavior of the standard and new strategy
Static metrics: example

**Figure:** delayed pivots (solution with numerical issue)

**Figure:** delayed pivots (solution with new pivoting strategy)
Static metrics: difference

Clamart, 30 May 2013
traces structure suggests the use of a relational database

- database management systems are used widely on huge databases
- lots of web resources on SQL queries
- SQL queries can be the direct sources of VTK visualization workflows

Figure: An imperfect schema for the factorization traces
a simple case: time spent on a node

@query
def time():
    return ""
    select events.factorization, inode,
        max(time) - min(time) as {1}
    from events
    where events.factorization = {0}
group by inode
""
def delayed_pivots():
    return """
    select nfront_ncb_tbl.factorization,
    nfront_ncb_tbl.inode,
    nfront-ncb-npivots as {1} from
    (select factorization, inode, nfront, ncb from
    events inner join MasterAssemblyBegin_params as MAB
    on events.factorization = {0} and
    MAB.event = events.event) as nfront_ncb_tbl,
    (select inode, npivots from
    events inner join MasterFactoEnd_params as MAE
    on events.factorization = {0} and
    MAE.event = events.event) as npivots_tbl
    where nfront_ncb_tbl.inode = npivots_tbl.inode
    """
**performance monitoring over MUMPS revisions**

1. “metatest” nightly collects some MUMPS measures (more than one year of data now)
2. the database needs some analysis, what measures are critical?
3. scenarios remain to be set in order to control the performance evolution

**factorization analysis : a draft tool at the moment**

1. features : selection of important points like memory peak, …
2. scalability : improve “pipeline” in order to process bigger trees
3. maintainability : improve traces in mumps code
4. more realistic visualization for training and lectures
5. performance monitoring at the node level?
Outline

Illustration of users oriented features
   Restarting feature
   Null space feature
   Compatibility with external libraries

Developers oriented features
   Validation tool
   Experimentation tool
   Experimentations and visualizations scripts
   other engineering issues
other engineering issues

- software engineering
  - code coverage
  - code quality, code documentation
  - code re-factoring
- integration and combination of research codes in main stream sources
- design of a new web site
- new design of user documentation
- interfaces to other languages
- ...
Engineering work is less visible than research work and is often shared by all developers (researchers, PhDs and engineers). It is however a vital investment for a software platform since it is critical to enable quality support and technological transfer.
Thank you for your attention.

Any questions?