FREEFEM++ DISTRIBUTED SOLVERS:
STATUS AND FUTURE

Pierre Jolivet, IRIT–CNRS

Contributions from:
- Julien Garaud, KTH
- Olivier Marquet, Johann Moulin, Jean-Lou Pfister, Onera Meudon

9th FreeFem++ days
December 14

INTRODUCTION

- matrix assembly using domain decomposition
- linear solvers using direct methods, iterative solvers, preconditioners...

- matrix assembly using domain decomposition
- linear solvers using direct methods, iterative solvers, preconditioners...

In FreeFem++, meshes are decomposed either via:

- `load "metis"`
- `load "scotch"`
- `load "parmetis"` (new!)

There is no parallelism inside the FreeFem++ kernel:

- no ghost elements
- no distributed meshes
- no distributed matrices

There is no parallelism inside the FreeFem++ kernel:

- no ghost elements
- no distributed meshes
- no distributed matrices

Users can access MPI via extra keywords in FreeFem++-mpi

Cons: parallelism is explicit (and must be hand coded)
Pros: users know what they are doing

DISTRIBUTED EXAMPLES

In the folder examples++-hpddm:

- scalable matrix assembly
- scalable linear solvers

DISTRIBUTED EXAMPLES

In the folder examples++-hpddm:

- scalable matrix assembly
- scalable linear solvers

Three linear algebra backends:

- HPDDM
- PETSc
- SLEPc (new!)

SOME OF CURRENT (AS OF 12/12/2017) EXAMPLES

- diffusion-2d.edp
- diffusion-2d-substructuring-PETSc.edp
- elasticity-2d.edp
- elasticity-2d-substructuring.edp
- elasticity-3d.edp
- elasticity-3d-PETSc.edp
- heat-2d.edp
- heat-2d-PETSc.edp
- helmholtz-2d-PETSc.edp
- maxwell-3d.edp
- stokes-2d-PETSc.edp
- stokes-3d.edp
- stokes-3d-PETSc-fieldsplit.edp
- diffusion-periodic-2d.edp
- diffusion-periodic-2d-PETSc.edp
- laplace-2d-SLEPc.edp
- schrodinger-2d-axial-well-SLEPc.edp
- ...

SOME OF CURRENT (AS OF 12/12/2017) EXAMPLES

- diffusion-2d.edp
- diffusion-2d-substructuring-PETSc.edp
- elasticity-2d.edp
- elasticity-2d-substructuring.edp
- elasticity-3d.edp
- elasticity-3d-PETSc.edp
- heat-2d.edp
- heat-2d-PETSc.edp
- helmholtz-2d-PETSc.edp
- maxwell-3d.edp
- stokes-2d-PETSc.edp
- stokes-3d.edp
- stokes-3d-PETSc-fieldsplit.edp
- diffusion-periodic-2d.edp
- diffusion-periodic-2d-PETSc.edp
- laplace-2d-SLEPc.edp
- schrodinger-2d-axial-well-SLEPc.edp
- ...

SOME OF CURRENT (AS OF 12/12/2017) EXAMPLES

- diffusion-2d.edp
- diffusion-2d-substructuring-PETSc.edp
- elasticity-2d.edp
- elasticity-2d-substructuring.edp
- elasticity-3d.edp
- elasticity-3d-PETSc.edp
- heat-2d.edp
- heat-2d-PETSc.edp
- helmholtz-2d-PETSc.edp
- maxwell-3d.edp
- stokes-2d-PETSc.edp
- stokes-3d.edp
- stokes-3d-PETSc-fieldsplit.edp
- diffusion-periodic-2d.edp
- diffusion-periodic-2d-PETSc.edp
- laplace-2d-SLEPc.edp
- schrodinger-2d-axial-well-SLEPc.edp
- ...

SOME OF CURRENT (AS OF 12/12/2017) EXAMPLES

- diffusion-2d.edp
- diffusion-2d-substructuring-PETSc.edp
- elasticity-2d.edp
- elasticity-2d-substructuring.edp
- elasticity-3d.edp
- elasticity-3d-PETSc.edp
- heat-2d.edp
- heat-2d-PETSc.edp
- helmholtz-2d-PETSc.edp
- maxwell-3d.edp
- stokes-2d-PETSc.edp
- stokes-3d.edp
- stokes-3d-PETSc-fieldsplit.edp
- diffusion-periodic-2d.edp
- diffusion-periodic-2d-PETSc.edp
- laplace-2d-SLEPc.edp
- schrodinger-2d-axial-well-SLEPc.edp
- ...

SOME OF CURRENT (AS OF 12/12/2017) EXAMPLES

- diffusion-2d.edp
- diffusion-2d-substructuring-PETSc.edp
- elasticity-2d.edp
- elasticity-2d-substructuring.edp
- elasticity-3d.edp
- elasticity-3d-PETSc.edp
- heat-2d.edp
- heat-2d-PETSc.edp
- helmholtz-2d-PETSc.edp
- maxwell-3d.edp
- stokes-2d-PETSc.edp
- stokes-3d.edp
- stokes-3d-PETSc-fieldsplit.edp
- diffusion-periodic-2d.edp
- diffusion-periodic-2d-PETSc.edp
- laplace-2d-SLEPc.edp
- schrodinger-2d-axial-well-SLEPc.edp
- ...

SOME OF CURRENT (AS OF 12/12/2017) EXAMPLES

- diffusion-2d.edp
- diffusion-2d-substructuring-PETSc.edp
- elasticity-2d.edp
- elasticity-2d-substructuring.edp
- elasticity-3d.edp
- elasticity-3d-PETSc.edp
- heat-2d.edp
- heat-2d-PETSc.edp
- helmholtz-2d-PETSc.edp
- maxwell-3d.edp
- stokes-2d-PETSc.edp
- stokes-3d.edp
- stokes-3d-PETSc-fieldsplit.edp
- diffusion-periodic-2d.edp
- diffusion-periodic-2d-PETSc.edp
- laplace-2d-SLEPc.edp
- schrodinger-2d-axial-well-SLEPc.edp
- ...

SOME OF CURRENT (AS OF 12/12/2017) EXAMPLES

- diffusion-2d.edp
- diffusion-2d-substructuring-PETSc.edp
- elasticity-2d.edp
- elasticity-2d-substructuring.edp
- elasticity-3d.edp
- elasticity-3d-PETSc.edp
- heat-2d.edp
- heat-2d-PETSc.edp
- helmholtz-2d-PETSc.edp
- maxwell-3d.edp
- stokes-2d-PETSc.edp
- stokes-3d.edp
- stokes-3d-PETSc-fieldsplit.edp
- diffusion-periodic-2d.edp
- diffusion-periodic-2d-PETSc.edp
- laplace-2d-SLEPc.edp
- schrodinger-2d-axial-well-SLEPc.edp
- ...

SOME OF CURRENT (AS OF 12/12/2017) EXAMPLES

- diffusion-2d.edp
- diffusion-2d-substructuring-PETSc.edp
- elasticity-2d.edp
- elasticity-2d-substructuring.edp
- elasticity-3d.edp
- elasticity-3d-PETSc.edp
- heat-2d.edp
- heat-2d-PETSc.edp
- helmholtz-2d-PETSc.edp
- maxwell-3d.edp
- stokes-2d-PETSc.edp
- stokes-3d.edp
- stokes-3d-PETSc-fieldsplit.edp
- diffusion-periodic-2d.edp
- diffusion-periodic-2d-PETSc.edp
- laplace-2d-SLEPc.edp
- schrodinger-2d-axial-well-SLEPc.edp
- ...

OPTIONS AND TYPES

Command line options parsed by PETSc/SLEPc or HPDDM

Additional types:

- `[z|d]matrix` (PETSc backend, new!)
- `[z|d]schwarz` (overlapping Schwarz backend)
- `[z|d]bdd` (BDD backend)
- `[z|d]feti` (FETI backend)
- `[z|d]eigensolver` (SLEPc backend, new!)

Command line options parsed by PETSc/SLEPc or HPDDM

Additional types:

- `[z|d]matrix` (PETSc backend, new!)
- `[z|d]schwarz` (overlapping Schwarz backend)
- `[z|d]bdd` (BDD backend)
- `[z|d]feti` (FETI backend)
- `[z|d]eigensolver` (SLEPc backend, new!)

All types support the operations:

- `set(A, sparams = "...");`
- `y = A * x;` // be careful with TGVs!
- `y =` $\mathtt{A}^{-1}$ `* x;`
- `changeOperator(A, Mat);`

Most important options to keep in mind:

- `-help`
- `-ksp_type`
- `-pc_type`
- `-eps_type` (new!)

Only the linear solvers are interfaced (no nonlinear solver or time integrator)

Most important options to keep in mind:

- `-help`
- `-ksp_type`
- `-pc_type`
- `-eps_type` (new!)

Only the linear solvers are interfaced (no nonlinear solver or time integrator)

Mixing complex and real libraries is tricky!

DD preconditioners, with a recent focus on:

- block iterative methods
- recycled iterative methods

Most important options to keep in mind:

- `-hpddm_help`
- `-hpddm_krylov_method`
- `-hpddm_geneo_nu`

MESHES

Meshes generated offline are not always “nice”

`Th = trunc(Th, true, renum = true);` // (in 3D, new!)

Nice improvements, especially in parallel

PARALLEL MESH PARTITIONER

- examples++-mpi/parmetis.edp
- examples++-mpi/parmetis-3d.edp

- examples++-mpi/parmetis.edp
- examples++-mpi/parmetis-3d.edp

Fake parallel meshes

Distributed evenly among MPI processes

- examples++-load/splitmesh3.edp
- examples++-load/splitmesh6.edp
- examples++-load/splitmesh4.edp (new!)
- examples++-load/splitmesh12.edp (new!)

$\Longrightarrow$ Scott–Vogelius FE as an alternative to Taylor–Hood FE

DISTRIBUTED SOLVERS

- `LinearCG`
- `LinearGMRES`

$\Longrightarrow$ only left preconditioning out of the box

- `LinearCG`
- `LinearGMRES`

$\Longrightarrow$ only left preconditioning out of the box

- `IterativeMethod`, examples++-hpddm/iterative.edp (new!)

$\Longrightarrow$ inherits most of HPDDM options

`https://github.com/hpddm/hpddm/blob/master/doc/cheatsheet.pdf`

- define `macro partitioner()metis// EOM`
- include `macro_ddm.idp`

MESH PARTITIONING

- define `macro partitioner()metis// EOM`
- include `macro_ddm.idp`

Three options:

1. standard mesh partitioning
2. partitioning with periodic BC (new!)
3. user-supplied partitioning (new!)

ParaView used to post-process distributed solutions:

- save all local solutions with `iovtk`
- convert .vtk (legacy format) to .vtu
- create a .pvd master file to load all .vtu

PETSC

New Krylov subspace method (KSP)

- `-ksp_type hpddm` (new!)
- `-hpddm_krylov_method ...` (used from within PETSc)

Lots of PDE involve coupled problems:

- velocity and pressure (Stokes)
- velocity and displacement (FSI)
- velocity and magnetic force (MHD)
- ...

Lots of PDE involve coupled problems:

- velocity and pressure (Stokes)

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} x_v \\ x_p \end{bmatrix} = \begin{bmatrix} f_v \\ f_p \end{bmatrix}$$

Lots of PDE involve coupled problems:

- velocity and pressure (Stokes)

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} x_v \\ x_p \end{bmatrix} = \begin{bmatrix} f_v \\ f_p \end{bmatrix}$$

```
fespace Vh(Th, [P2, P2, P1]); Vh [vx, vy, p];
fespace Wh(Th, [P2, P2, P1, P2]); Vh [vx, vy, p, vz];
```

Monolithic preconditioners

- costly to consider the complete problem
- not always easy to define a preconditioner

PCFIELDSPLIT

- block preconditioner
- aggregate same physics together
- preconditioner inside each block
- coupling using $+$, $\times$, or Schur complement techniques

`http://www.caam.rice.edu/~mk51/presentations/SIAMCSE13.pdf`

```
fespace Vh(Th, [P2, P2, P1, P2]);
Vh             [vx, vy, p,  vz];
```

INTERFACE WITH PCFIELDSPLIT

```
fespace Vh(Th, [P2, P2, P1, P2]);
Vh             [vx, vy, p,  vz];

Vh [a, b, c, d] = [1, 1, 2, 1];
```

```
fespace Vh(Th, [P2, P2, P1, P2]);
Vh             [vx, vy, p,  vz];

Vh [a, b, c, d] = [1, 1, 2, 1];

set(A, sparams = "-pc_type fieldsplit", list = a[]);
```

```
fespace Vh(Th, [P2, P2, P1, P2]);
Vh             [vx, vy, p,  vz];

Vh [a, b, c, d] = [1, 1, 2, 1];

set(A, sparams = "-pc_type fieldsplit", list = a[]);

string[int] names(2);
names[0] = "velocity";
names[1] = "pressure";
```

APPROXIMATING THE SCHUR COMPLEMENT

-pc_fieldsplit_type schur

- $Sx = y$ with $S = A - CD^{-1}B$

APPROXIMATING THE SCHUR COMPLEMENT

```
-pc_fieldsplit_type schur
```

- $Sx = y$ with $S = A - CD^{-1}B$
- define a KSP for D^{-1}

Approximating the Schur complement

-pc_fieldsplit_type schur

- $Sx = y$ with $S = A - CD^{-1}B$
- define a KSP for D^{-1}
- use A as a preconditioner

-pc_fieldsplit_type schur

- $Sx = y$ with $S = A - CD^{-1}B$
- define a KSP for D^{-1}
- use A as a preconditioner

Preconditionner based on physics, e.g., for Navier–Stokes:

- augmented Lagrangian
- modified Grad-Div

```
fespace Ph(Th, P1); // pressure space only!
varf Precond(p, q) = int2d(Th)(...);
matrix[int] S(1); // combination of matrices
S[0] = Precond(Ph, Ph); // preconditioner "S^{-1}"
set(A, schurPreconditioner = S);
```

SLEPC

DISTRIBUTED EIGENSOLVER

- extension of PETSc for eigenproblems
- may use ARPACK from SLEPC
- access to all of PETSc features (PCFIELDSPLIT...)
- HPDDM iterative methods also registered
- distributed I/O

CONCLUSION

DOCUMENTATIONS

- http://www.freefem.org/ff++/ftp/freefem++doc.pdf (section 11.5)
- https://github.com/hpddm/hpddm/blob/master/doc/cheatsheet.pdf
- http://www.mcs.anl.gov/petsc/petsc-current/docs/manual.pdf
- http://slepc.upv.es/documentation/slepc.pdf

FINAL WORDS

Summary:

- you should use FreeFem++-mpi
- many examples to start from
- new developments are user-driven

Future work:

- other fancy preconditioners
- different applications

Summary:

- you should use FreeFem++-mpi
- many examples to start from
- new developments are user-driven

Future work:

- other fancy preconditioners
- different applications

Thank you!