Scalable Domain Decomposition Preconditioners in FreeFem++

F. Hecht, F. Nataf, P. Jolivet

Laboratoire Jacques-Louis Lions Université Pierre et Marie Curie

Projet Alpines, INRIA Rocquencourt.

http://www.freefem.org/ff++

Outline

1. Introduction
   - Main characteristics
2. Academic Examples
   - Poisson in a fish
   - Optimisation : Bose Einstein Condensate
3. A first way to break complexity
   - Build Matrix and vector of a problem
4. Schwarz method with overlap
   - Poisson equation with Schwarz method
   - Transfer Part
   - parallel GMRES
   - A simple Coarse grid solver
   - Numerical experiment
5. An other way to build a 2-level Schwarz with oscilation
   - Choice of the coarse space
   - First Numerical results
     - Implementation framework
   - Parallel implementation
     - Scalability tests
6. Future/Conclusion

Outline

1 Introduction
- Main characteristics

2 Academic Examples

3 A first way to break complexity

4 Schwarz method with overlap

5 An other way to build a 2-level Schwarz with oscilation

6 Future/Conclusion

Main characteristics I/II (2D)(3D)

- Wide range of finite elements : continuous P1,P2 elements, discontinuous P0, P1, RT0,RT1,BDM1, elements ,Edge element, vectorial element, mini-element, ...
- Automatic interpolation of data from a mesh to an other one (with matrix construction if need), so a finite element function is view as a function of (x, y, z) or as an array.
- LU, Cholesky, Crout, CG, GMRES, UMFPack, SuperLU, MUMPS, HIPS , SUPERLU_DIST, PASTIX. ... sparse linear solver ; eigenvalue and eigenvector computation with ARPACK.

Main characteristics II/II (2D)(3D)

- Automatic mesh generator, based on the Delaunay-Voronoï algorithm. (2d,3d (tetgen))
- Mesh adaptation based on metric, possibly anisotropic (only in 2d), with optional automatic computation of the metric from the Hessian of a solution. (2d,3d).
- Dynamic linking to add plugin.
- Full MPI interface
- Nonlinear Optimisation tools : CG, Ipopt, NLOpt, stochastic
- Wide range of examples : Navier-Stokes 3d, elasticity 3d, fluid structure, eigenvalue problem, Schwarz' domain decomposition algorithm, residual error indicator ...

Outline

1. Introduction
2. Academic Examples
   - Poisson in a fish
   - Optimisation : Bose Einstein Condensate
3. A first way to break complexity
4. Schwarz method with overlap
5. An other way to build a 2-level Schwarz with oscilation
6. Future/Conclusion

Poisson equation, weak form

Let a domain Ω with a partition of $\partial\Omega$ in Γ_2, Γ_e.
Find u a solution in such that :

$$-\Delta u = 1 \text{ in } \Omega, \quad u = 2 \text{ on } \Gamma_2, \quad \frac{\partial u}{\partial \vec{n}} = 0 \text{ on } \Gamma_e \qquad (1)$$

Denote $V_g = \{v \in H^1(\Omega) / v_{|\Gamma_2} = g\}$.
The Basic variational formulation with is : find $u \in V_2(\Omega)$, such that

$$\int_\Omega \nabla u . \nabla v = \int_\Omega 1 v + \int_\Gamma \frac{\partial u}{\partial n} v, \quad \forall v \in V_0(\Omega) \qquad (2)$$

The finite element method is just : replace V_g with a finite element space, and the `FreeFem++` code :

Poission equation in FreeFem++

The finite element method is just : replace V_g with a finite element space, and the FreeFem++ code :

```
mesh3 Th("fish3d.msh");                       //    read a mesh 3d
fespace Vh(Th,P1);                        //    define the P1 EF space

 Vh u,v;        //    set test and unknow FE function in Vh.
macro Grad(u) [dx(u),dy(u),dz(u)]          //   EOM Grad def
solve laplace(u,v,solver=CG) =
   int3d(Th)(  Grad(u)'*Grad(v)   )
   - int3d(Th) ( 1*v)
   + on(2,u=2);                                //     int on γ2
plot(u,fill=1,wait=1,value=0,wait=1);
```

Run:fish.edp   Run:fish3d.edp

Bose Einstein Condensate

Just a direct use of `Ipopt` interface (2day of works)
The problem is find a complex field u on domain $\mathcal{D}$ such that :

$$u = \underset{||u||=1}{\operatorname{argmin}} \int_{\mathcal{D}} \frac{1}{2}|\nabla u|^2 + V_{trap}|u|^2 + \frac{g}{2}|u|^4 - \Omega i \overline{u}\left(\left(\begin{smallmatrix} -y \\ x \end{smallmatrix}\right).\nabla\right)u$$

to code that in FreeFem++
use

- Ipopt interface (`https://projects.coin-or.org/Ipopt`)
- Adaptation de maillage
- Analyse of the result

Run:BEC.edp

Outline

1. Introduction
2. Academic Examples
3. A first way to break complexity
   - Build Matrix and vector of a problem
4. Schwarz method with overlap
5. An other way to build a 2-level Schwarz with oscilation
6. Future/Conclusion

A first way to break complexity

Idea :

$$a(u, v) = \int_{\Omega} \nabla u . \nabla v$$

take an equi-partition of Ω in Ω_i for $i = 0$ to $N_p - 1$ the number of processor.
then

$$a(u, v) = \sum_{i=0}^{N_p - 1} \int_{\Omega_i} \nabla u . \nabla v$$

A first way to break complexity

1. Build matrix in parallel by assembling par region, remark the change function you modify the region numbering, to defined Ω_i.

```
real c = mpisize/real(Th.nt) ;
Th=change(Th,fregion= min(mpisize-1,int(nuTriangle*c)));
```

2. Assemble the full matrix

```
varf vlaplace(uh,vh) =          //    definition de problem
     int3d(Th,mpirank)( uh*vh+ dt*Grad(uh)'*grad(vh) )
   + int3d(Th,mpirank)( dt*vh*f) + on(1,uh=g) ;
matrix A = vlaplace(Vh,Vh) ;
real[int] b =  vlaplace(0,Vh) ;
```

3. Solve the linear using a good parallel solver (MUMPS)

```
load "MUMPS"
set(A,solver=sparsesolver,master=-1);   //   Distributed A.
uh[] = A^-1*b ;                             //    resolution
```

Run:Heat3d.edp   Run:NSCaraCyl-100-mpi2.edp

Outline

1. Introduction
2. Academic Examples
3. A first way to break complexity
4. Schwarz method with overlap
   - Poisson equation with Schwarz method
   - Transfer Part
   - parallel GMRES
   - A simple Coarse grid solver
   - Numerical experiment
5. An other way to build a 2-level Schwarz with oscilation

To solve the following Poisson problem on domain Ω with boundary Γ in $L^2(\Omega)$:

$$-\Delta u = f, \text{ in } \Omega, \text{ and } u = g \text{ on } \Gamma,$$

where $f \in L^2(\Omega)$ and $g \in H^{\frac{1}{2}}(\Gamma)$ are two given functions.
Let introduce $(\pi_i)_{i=1,..,N_p}$ a positive regular partition of the unity of Ω, q-e-d :

$$\pi_i \in \mathcal{C}^0(\Omega) : \quad \pi_i \ge 0 \text{ and } \sum_{i=1}^{N_p} \pi_i = 1.$$

Denote Ω_i the sub domain which is the support of π_i function and also denote Γ_i the boundary of Ω_i, and $\Omega_i^\circ = \{x / 0 < \pi_i < 1\}$
The parallel Schwarz method is Let $\ell = 0$ the iterator and a initial guest u^0 respecting the boundary condition (i.e. $u^0_{|\Gamma} = g$).

$$\forall i = 1.., N_p : \quad -\Delta u_i^\ell = f, \text{ in } \Omega_i, \quad \text{and } u_i^\ell = u^\ell \text{ on } \Gamma_i \qquad (3)$$

$$u^{\ell+1} = \sum_{i=1}^{N_p} \pi_i u_i^\ell \qquad (4)$$

Some Remark

We never use finite element space associated to the full domain Ω because it to expensive.

We have to defined to operator to build the previous algorithm :
We denote $u^\ell_{h|i}$ the restriction of u^ℓ_h on V_{hi}, so the discrete problem on Ω_i of problem (3) is find $u^\ell_{hi} \in V_{hi}$ such that :

$$\forall v_{hi} \in V_{0i} : \int_{\Omega_i} \nabla v_{hi}.\nabla u^\ell_{hi} = \int_{\Omega_i} f v_{hi},$$

$$\forall k \in \mathcal{N}^{\Gamma_i}_{hi} \; : \; \sigma^k_i(u^\ell_{hi}) = \sigma^k_i(u^\ell_{h|i})$$

where g^k_i is the value of g associated to the degree of freedom $k \in \mathcal{N}^{\Gamma_i}_{hi}$.

To compute $v_i = \pi_i u_i + \sum_{j\in J_i} \pi_j u_j$ and can be write the freefem++ function Update with asynchronous send/recv.

```
func bool Update(real[int] &ui, real[int] &vi)
{ int n= jpart.n;
  for(int j=0;j<njpart;++j)  Usend[j][]=sMj[j]*ui;
  mpiRequest[int] rq(n*2);
  for (int j=0;j<n;++j)
        Irecv(processor(jpart[j],comm,rq[j  ]), Ri[j][]);
  for (int j=0;j<n;++j)
        Isend(processor(jpart[j],comm,rq[j+n]), Si[j][]);
  for (int j=0;j<n*2;++j)   int k= mpiWaitAny(rq);
  vi = Pii*ui;                                  //    set to πi ui
  //    apply the unity local partition .
   for(int j=0;j<njpart;++j)
     vi += rMj[j]*Vrecv[j][];                     //    add πj uj
 return true; }
```

parallel GMRES

Finally you can easily accelerate the fixe point algorithm by using a parallel GMRES algorithm after the introduction the following affine $\mathcal{S}_i$ operator sub domain Ω_i.

```
func real[int] Si(real[int]& U) {
 real[int] V(U.n)   ;   b= onG .* U;
  b  = onG ? b : Bi ;
  V = Ai^-1*b;                                        //     (3)
  Update(V,U);                                        //     (4)
  V -= U;   return V; }
```

Where the parallel MPIGMRES or MPICG algorithm is to solve $A_i x_i = b_i, i = 1, .., N_p$ by just changing the dot product by reduce the local dot product of all process with the following MPI code :

```
template<class R> R ReduceSum1(R s,MPI_Comm * comm)
{   R r=0;
    MPI_Allreduce( &s, &r, 1 ,MPI_TYPE<R>::TYPE(),
                            MPI_SUM,  *comm );
    return r; }
```

A simple coarse grid is we solve the problem on the coarse grid :

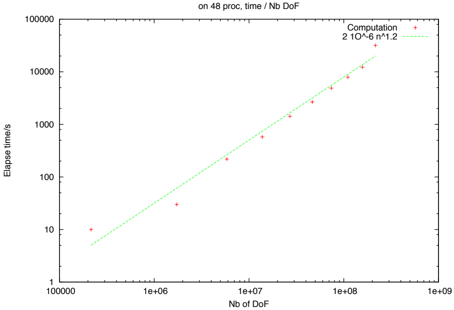
```
func bool  CoarseSolve(real[int]& V,real[int]& U,
               mpiComm& comm)
{
    if(AC.n==0 && mpiRank(comm)==0) //    first time build
      AC = vPbC(VhC,VhC,solver=sparsesolver);
   real[int] Uc(Rci.n),Bc(Uc.n);
   Uc= Rci*U;                             //    Fine to Coarse
   mpiReduce(Uc,Bc,processor(0,comm),mpiSUM);
   if(mpiRank(comm)==0)
      Uc = AC^-1*Bc;                     //    solve of proc 0
    broadcast(processor(0,comm),Uc);
   V = Pci*Uc;                            //    Coarse to Fine
}
```

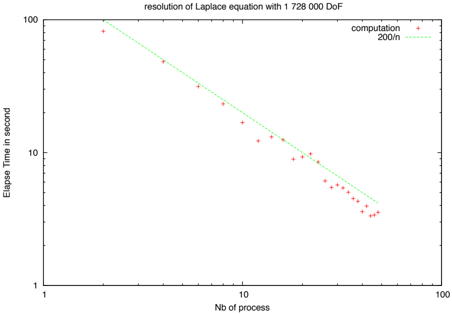
Limitation : if the initial problem, data have oscillation, you must an other on coarse problem : The GENO algorithm for example form the Nataf and co., See section 5.

So we finally we get 4 algorithms

1. The basic schwarz algorithm $u^{\ell+1} = \mathcal{S}(u^\ell)$, where $\mathcal{S}$ is one iteration of schwarz process.
2. Use the GMRES to find u solution of the linear system $\mathcal{S}u - u = 0$.
3. Use the GMRES to solve parallel problem $\mathcal{A}_i u_i = b_i$, $i = 1, \ldots, N_p$, with RAS precondicionneur
4. Use the method with two level precondicionneur RAS and Coarse.

On the SGI UV 100 of the lab :





A Parallel Numerical experiment on laptop

We consider first example in an academic situation to solve Poisson Problem on the cube $\Omega =]0,1[^3$

$$-\Delta u = 1, \text{in } \Omega; \qquad u = 0, \text{ on } \partial\Omega. \tag{5}$$

With a cartesian meshes $\mathcal{T}_{hn}$ of Ω with $6n^3$ tetrahedron, the coarse mesh is $\mathcal{T}^*_{hm}$, and m is a divisor of n.
We do the validation of the algorithm on a Laptop Intel Core i7 with 4 core at 1.8 Ghz with $4Go$ of RAM DDR3 at 1067 Mhz,

Run:DDM-Schwarz-Lap-2dd.edp Run:DDM-Schwarz-Lame-2d.edp
Run:DDM-Schwarz-Lame-3d.edp

Outline

1 Introduction

2 Academic Examples

3 A first way to break complexity

4 Schwarz method with overlap

5 An other way to build a 2-level Schwarz with oscilation
- Choice of the coarse space
- First Numerical results
- Parallel implementation

6 Future/Conclusion

Motivation

Large discretized system of PDEs
strongly heterogeneous coefficients
(high contrast, nonlinear, multiscale)

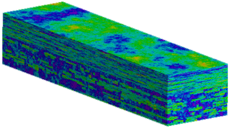
E.g. Darcy pressure equation,
P_1-finite elements :

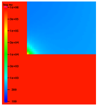
$$A\mathbf{U} = \mathbf{F}$$

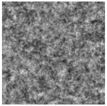
$$\text{cond}(A) \sim \frac{\alpha_{\max}}{\alpha_{\min}} h^{-2}$$

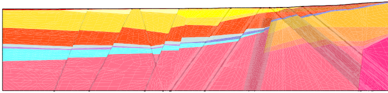
Goal :
iterative solvers
robust in size and heterogeneities

Applications :
flow in heterogeneous / stochastic / layered media
structural mechanics
electromagnetics
etc.









Adding a coarse space

We add a coarse space correction (*aka* second level)
Let V_H be the coarse space and Z be a basis, $V_H = \mathrm{span}\, Z$, writing $R_0 = Z^T$ we define the two level preconditioner as :

$$M^{-1}_{ASM,2} := R_0^T \left(R_0 A R_0^T\right)^{-1} R_0 + \sum_{i=1}^{N} R_i^T A_i^{-1} R_i.$$

The Nicolaides approach is to use the kernel of the operator as a coarse space, this is the constant vectors, in local form this writes :

$$Z := (R_i^T D_i R_i \mathbf{1})_{1 \le i \le N}$$

where D_i are chosen so that we have a partition of unity :

$$\sum_{i=1}^{N} R_i^T D_i R_i = Id.$$

Theoretical convergence result

Theorem (Widlund, Dryija)

Let $M_{ASM,2}^{-1}$ *be the two-level additive Schwarz method :*

$$\kappa(M_{ASM,2}^{-1}A) \leq C\left(1 + \frac{H}{\delta}\right)$$

where δ *is the size of the overlap between the subdomains and* H *the subdomain size.*

This does indeed work very well

Number of subdomains	8	16	32	64
ASM	18	35	66	128
ASM + Nicolaides	20	27	28	27

Failure for Darcy equation with heterogeneities

$$
\begin{aligned}
-\nabla \cdot (\alpha(x,y)\nabla u) &= 0 \quad \text{in} \quad \Omega \subset \mathbb{R}^2, \\
u &= 0 \quad \text{on} \quad \partial\Omega_D, \\
\frac{\partial u}{\partial n} &= 0 \quad \text{on} \quad \partial\Omega_N.
\end{aligned}
$$

Decomposition

$\alpha(x,y)$

Jump	1	10	10^2	10^3	10^4
ASM	39	45	60	72	73
ASM + Nicolaides	30	36	50	61	65

Our approach

Fix the problem by an optimal and proven choice of a coarse space Z.

Objectives

Strategy

Define an appropriate coarse space $V_{H2} = \text{span}(Z_2)$ and use the framework previously introduced, writing $R_0 = Z_2^T$ the two level preconditioner is :

$$P_{ASM2}^{-1} := R_0^T (R_0 A R_0^T)^{-1} R_0 + \sum_{i=1}^{N} R_i^T A_i^{-1} R_i.$$

The coarse space must be

- Local (calculated on each subdomain) $\rightarrow$ parallel
- Adaptive (calculated automatically)
- Easy and cheap to compute
- Robust (must lead to an algorithm whose convergence is proven not to depend on the partition nor the jumps in coefficients)

Abstract eigenvalue problem

Gen.EVP per subdomain :

Find $p_{j,k} \in V_{h|\Omega_j}$ and $\lambda_{j,k} \geq 0$:

$$a_{\Omega_j}(p_{j,k},\, v) \;=\; \lambda_{j,k}\, a_{\Omega_j^\circ}(\Xi_j p_{j,k},\, \Xi_j v) \qquad \forall v \in V_{h|\Omega_j}$$

$$A_j \mathbf{p}_{j,k} \;=\; \lambda_{j,k}\, \mathbf{X}_j A_j^\circ \mathbf{X}_j\, \mathbf{p}_{j,k} \qquad (\mathbf{X}_j \text{ ...diagonal})$$

Ξ_j is the partition unity

$a_D \ldots$ restriction of a to D

In the two-level ASM :

Choose first m_j eigenvectors per subdomain :

$$V_0 \;=\; \text{span}\{\Xi_j p_{j,k}\}_{k=1,\ldots,m_j}^{j=1,\ldots,N}$$

This automatically includes Zero Energy Modes.

Galvis & Efendiev (SIAM 2010) :

$$\int_{\Omega_j} \boldsymbol{\kappa}\, \nabla p_{j,k} \cdot \nabla v \, dx \;=\; \lambda_{j,k} \int_{\Omega_j} \boldsymbol{\kappa}\, p_{j,k}\, v\, dx \qquad \forall v \in V_{h|\Omega_j}$$

Efendiev, Galvis, Lazarov & Willems (submitted) :

$$a_{\Omega_j}(p_{j,k},\, v) \;=\; \lambda_{j,k} \sum_{i \in \text{neighb}(j)} a_{\Omega_j}(\xi_j\, \xi_i\, p_{j,k},\, \xi_j\, \xi_i\, v) \qquad \forall v \in V_{|\Omega_j}$$

ξ_j ... partition of unity, calculated adaptively (MS)

Our gen.EVP :

$$a_{\Omega_j}(p_{j,k},\, v) \;=\; \lambda_{j,k}\, a_{\Omega_j^\circ}(\Xi_j p_{j,k},\, \Xi_j v) \qquad \forall v \in V_{h|\Omega_j}$$

both matrices typically singular $\Longrightarrow$ $\lambda_{j,k} \in [0,\, \infty]$

Theory

Two technical assumptions.

Theorem (Spillane, Dolean, Hauret, N., Pechstein, Scheichl)

If for all j : $\quad 0 < \lambda_{j,m_{j+1}} < \infty$ *:*

$$\kappa(M^{-1}_{ASM,2}A) \;\leq\; (1+k_0)\Big[2 + k_0\,(2k_0+1)\,\max_{j=1}^{N}\Big(1 + \frac{1}{\lambda_{j,m_j+1}}\Big)\Big]$$

Possible criterion for picking m_j : (used in our Numerics)

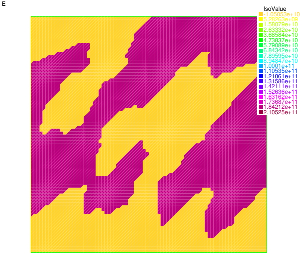
$$\lambda_{j,m_j+1} \;<\; \frac{\delta_j}{H_j}$$

$H_j \ldots$ subdomain diameter, $\delta_j \ldots$ overlap

Numerical results via a Domain Specific Language

`FreeFem++` (`http://www.freefem.org/ff++`), with :

- Metis Karypis and Kumar 1998
- SCOTCH Chevalier and Pellegrini 2008
- UMFPACK Davis 2004
- ARPACK Lehoucq et al. 1998
- MPI Snir et al. 1995
- Intel MKL
- PARDISO Schenk et al. 2004
- MUMPS Amestoy et al. 1998
- PaStiX Hénon et al. 2005



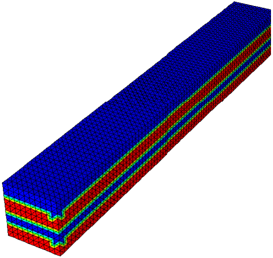
$$E_1 = 2 \cdot 10^{11}$$
$$\nu_1 = 0.3$$

$$E_2 = 2 \cdot 10^{7}$$
$$\nu_2 = 0.45$$

METIS partitions with 2 layers added

subd.	dofs	AS-1	AS-ZEM	(V_H)	GENEO	(V_H)
4	13122	93	134	*(12)*	42	*(42)*
16	13122	164	165	*(48)*	45	*(159)*
25	13122	211	229	*(75)*	47	*(238)*
64	13122	279	167	*(192)*	45	*(519)*

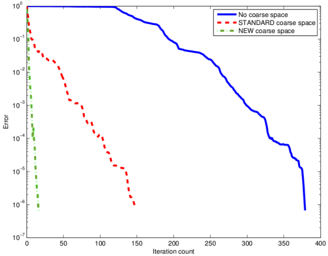
Iterations (CG) vs. number of subdomains



$$E_1 = 2 \cdot 10^{11}$$
$$\nu_1 = 0.3$$

$$E_2 = 2 \cdot 10^{7}$$
$$\nu_2 = 0.45$$

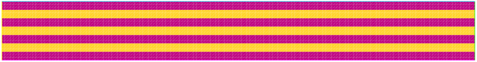
Relative error vs. iterations
16 regular subdomains



subd.	dofs	AS-1	AS-ZEM	(V_H)	GENEO	(V_H)
4	1452	79	54	*(24)*	16	*(46)*
8	29040	177	87	*(48)*	16	*(102)*
16	58080	378	145	*(96)*	16	*(214)*

AS-ZEM (Rigid body motions) : $m_j = 6$

Numerical results – Optimality



Layers of hard and soft elastic materials

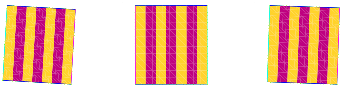
m_i is given automatically by the strategy.

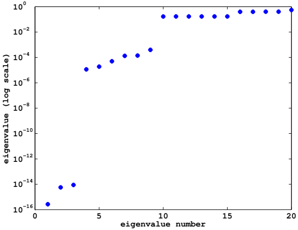
$\#Z$ per subd.	one level	ZEM	GenEO
$\max(m_i - 1, 3)$			2600 (93)
m_i	$5.1\,e5$ (184)	$1.4\,e4$ (208)	53 (35)
$m_i + 1$			45 (25)

condition number (iteration count) for one and two level ASMs

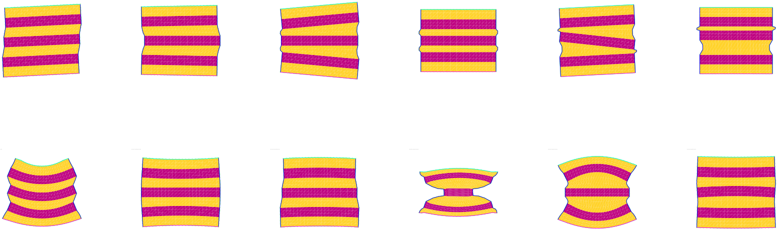
- Taking one fewer eigenvalue has a huge influence on the iteration count
- Taking one more has only a small influence

Eigenvalues and eigenvectors





Logarithmic scale



Darcy pressure equation

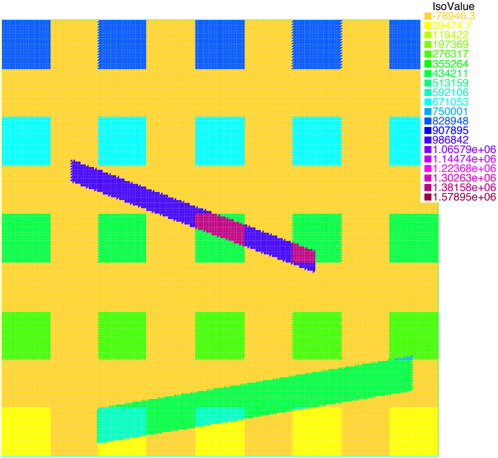
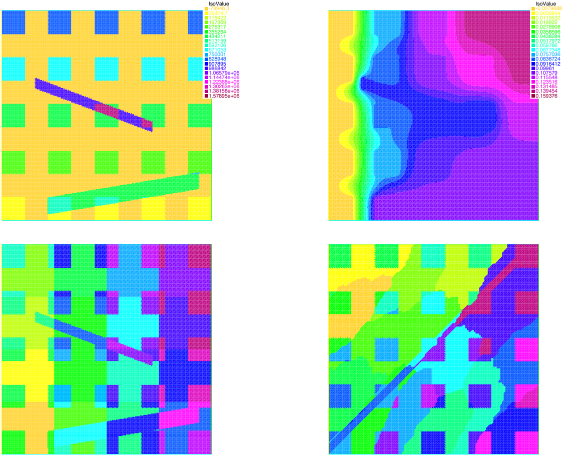


Figure : Two dimensional diffusivity κ



Channels and inclusions : $1 \leq \alpha \leq 1.5 \times 10^6$, the solution and partitionings (Metis or not)

Parallel implementation

PhD of Pierre Jolivet.

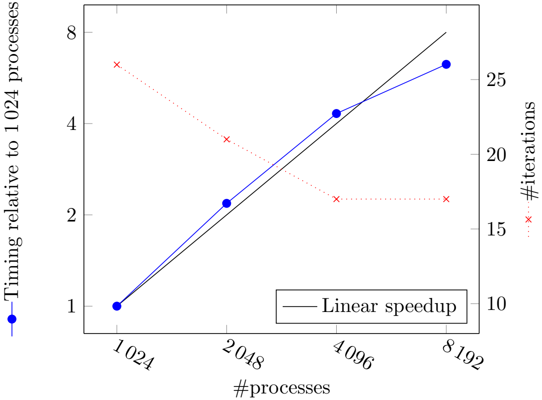
Since version 1.16, bundled with the `Message Passing Interface`. `FreeFem++` is working on the following parallel architectures (among others) :

	N° of cores	Memory	Peak perf
hpc1@LJLL	160@2.00 Ghz	640 Go	$\sim$ 10 TFLOP/s
titane@CEA	12192@2.93 Ghz	37 To	140 TFLOP/s
babel@IDRIS	40960@850 Mhz	20 To	139 TFLOP/s
curie@CEA	92000@2.93 Ghz	315 To	1.6 PFLOP/s

`http://www-hpc.cea.fr`, Bruyères-le-Châtel, France.
`http://www.idris.fr`, Orsay, France.

`http://www.prace-project.eu`.

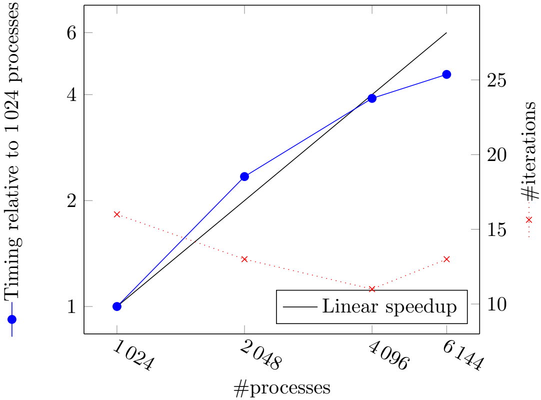
Elasticity problem with heterogeneous coefficients



Speed-up for a 1.2 billion unknowns 2D problem. Direct solvers in the subdomains. Peak performance wall-clock time : 26s.

Strong scalability in three dimensions heterogeneous elasticity

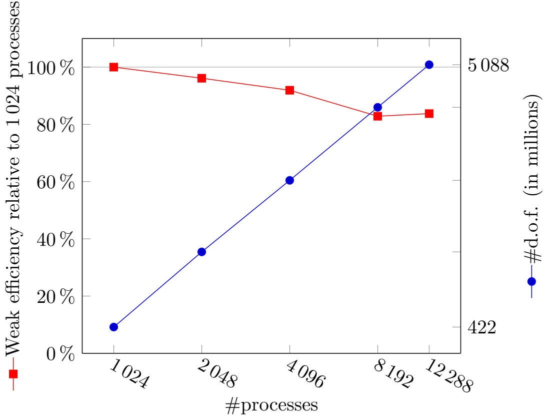
Elasticity problem with heterogeneous coefficients



Speed-up for a 160 million unknowns 3D problem. Direct solvers in subdomains. Peak performance wall-clock time : 36s.

Weak scalability in two dimensions

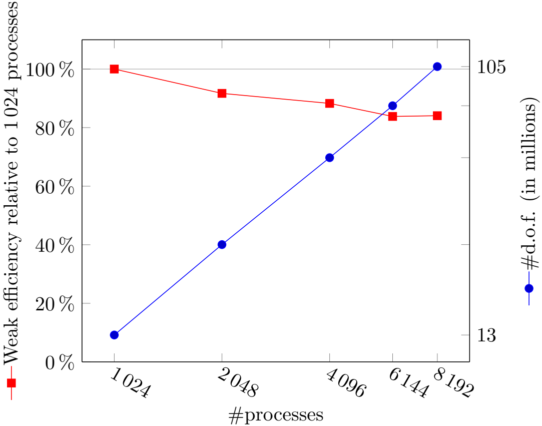
Darcy problems with heterogeneous coefficients



Efficiency for a 2D problem. Direct solvers in the subdomains. Final size : 22 billion unknowns. Wall-clock time : $\simeq$ 200s.

Weak scalability in three dimensions

Darcy problems with heterogeneous coefficients



Efficiency for a 3D problem. Direct solvers in the subdomains. Final size : 2 billion unknowns. Wall-clock time : $\simeq$ 200s.

Outline

1. Introduction
2. Academic Examples
3. A first way to break complexity
4. Schwarz method with overlap
5. An other way to build a 2-level Schwarz with oscilation
6. Future/Conclusion

Conclusion/Future

Freefem++ v3 is

- very good tool to solve non standard PDE in 2D/3D
- to try new domain decomposition domain algorithm

The the future we try to do :

- Build more graphic with VTK, paraview , ... (in progress)
- Add Finite volume facility for hyperbolic PDE (just begin C.F. FreeVol Projet)
- 3d anisotrope mesh adaptation
- automate the parallel tool

Thank for you attention.

Bibliography Schwarz : DtN and GenEO

Preprints available on HAL :
`http://hal.archives-ouvertes.fr/`

N. Spillane, V. Dolean, P. Hauret, F. Nataf, C. Pechstein, R. Scheichl, "An Algebraic Local Generalized Eigenvalue in the Overlapping Zone Based Coarse Space : A first introduction", *C. R. Mathématique*, Vol. 349, No. 23-24, pp. 1255-1259, 2011.

F. Nataf, H. Xiang, V. Dolean, N. Spillane, "A coarse space construction based on local Dirichlet to Neumann maps", *SIAM J. Sci Comput.*, Vol. 33, No.4, pp. 1623-1642, 2011.

F. Nataf, H. Xiang, V. Dolean, "A two level domain decomposition preconditioner based on local Dirichlet-to-Neumann maps", *C. R. Mathématique*, Vol. 348, No. 21-22, pp. 1163-1167, 2010.

V. Dolean, F. Nataf, R. Scheichl, N. Spillane, "Analysis of a two-level Schwarz method with coarse spaces based on local Dirichlet–to–Neumann maps", CMAM, 2012 vol. 12, `http://hal.archives-ouvertes.fr/hal-00586246/fr/`.

P. Jolivet, V. Dolean, F. Hecht, F. Nataf, C. Prud'homme, N. Spillane, "High Performance domain decomposition methods on massively parallel architectures with FreeFem++", J. of Numerical Mathematics, 2012 vol. 20.

N. Spillane, V. Dolean, P. Hauret, F. Nataf, C. Pechstein, R. Scheichl, "Abstract Robust Coarse Spaces for Systems of PDEs via Generalized Eigenproblems in the Overlaps", submitted to *Numerische Mathematik*, 2011.

V. Dolean, F. Nataf, P. Jolivet : *"Domain decomposition methods. Theory and parallel implementation with FreeFEM++"*, Lecture notes (coming soon).