Hybrid techniques in the solution of large scale problems

Iain S. Duff

iain.duff@stfc.ac.uk

STFC Rutherford Appleton Laboratory

Oxfordshire, UK.

and

CERFACS, Toulouse, France

Homepage: http://www.numerical.rl.ac.uk/people/isd/isd.html

Co-authors

In this talk, we discuss work mainly on projects at

CERFACS

The main people involved in this work are:

Mario Arioli, Luc Giraud, Serge Gratton, Azzam Haidar, Xavier Pinel, Jean-Christophe Rioual, Xavier Vasseur

Task is to solve

$$\mathbf{Ax} = \mathbf{b}$$

where the dimension of $\mathbf{A}$ may be 10^6 or greater.

In our case A is normally from the discretization of a three-dimensional PDE.

Outline

- Direct methods
- Hybrid methods
- Static pivoting
- Domain decomposition
- Helmholtz equation in geophysics

Direct methods

Grid dimensions	Matrix order	Work to factorize	Factor storage
$k \times k$	k^2	k^3	$k^2 \log k$
$k \times k \times k$	k^3	k^6	k^4

$\mathcal{O}$ complexity of direct method on 2D and 3D grids.

Hybrid methods

COMBINING DIRECT AND ITERATIVE METHODS

(can be thought of as sophisticated preconditioning)

Multigrid

Using direct method as coarse grid solver.

Domain Decomposition

Using direct method on local subdomains and “direct” preconditioner on interface.

Block Iterative Methods

Direct solver on sub-blocks.

Partial factorization as preconditioner

Factorization of nearby problem as a preconditioner

Direct methods ... static pivoting

A sparse direct method normally consists of three phases

- Analysis (determine ordering and data structures)
- Numerical factorization ($A \longrightarrow LDL^T$)
- Solution phase (obtain solution using sparse triangular solves)

When the matrix is positive definite this works well but in the indefinite case subsequent numerical pivoting may mean that the initial analysis is not respected.

Static Pivoting

The default action for general matrices is to use some form of threshold pivoting in the numerical factorization phase.

An alternative is to use **Static Pivoting**, by replacing potentially small pivots p_k by

$$p_k + \tau$$

and maintaining the same pivoting strategy as advocated in the analysis.

This is even more important in the case of parallel implementation where static data structures are often preferred

Static Pivoting

Several codes use (or have an option for) this device:

- SuperLU (Demmel and Li)
- PARDISO (Gärtner and Schenk)
- MA57 (Duff and Pralet)
- MUMPS (Amestoy, Duff, L'Excellent, and Koster)

`mumps.enseeiht.fr` mumps@cerfacs.fr

Static Pivoting

We thus have factorized

$$A + E = LDL^T$$

where $|E| \leq \tau I$

The four codes then have an **Iterative Refinement** option

The problem is that this sometimes does not converge

Static Pivoting

Choosing τ

Increase $\tau \Longrightarrow$ increase stability of decomposition

Decrease $\tau \Longrightarrow$ better approximation of the original matrix, reduces $||E||$

Static Pivoting

Choosing τ

Increase $\tau \Longrightarrow$ increase stability of decomposition

Decrease $\tau \Longrightarrow$ better approximation of the original matrix, reduces $||E||$

Trade-off

- $\approx 1 \Longrightarrow$ huge error $||E||$.
- $\approx \varepsilon \Longrightarrow$ big growth in preconditioning matrix

Conventional wisdom is to choose

$$\tau = \mathcal{O}(\sqrt{\varepsilon})$$

Static pivoting

Matrix	Order	Entries	Number delayed num	Number tiny static	Factorization time seconds num	Factorization time seconds static	Size of the factors num	Size of the factors static
BRAINPC2	27607	96601	14267	12932	0.18	0.11	656765	322971
BRATU3D	27792	88627	90052	8429	34.2	9.24	11484379	5569194
CONT-201	80595	239596	71296	27470	5.51	1.94	8820367	4304559
CONT-300	180895	562496	183306	67864	21.1	6.08	23838606	10714425
cvxqp3	17500	62481	30519	6277	9.73	3.08	4740141	2301836
DTOC	24993	34986	29478	9790	29.1	0.41	4714248	187639
mario001	38434	114643	15463	10305	0.28	0.23	817056	575373
NCVXQP1	12111	40537	12463	3619	2.69	1.29	2235743	1327920
NCVXQP5	62500	237483	16703	8402	25.7	23.0	13365963	11205204
NCVXQP7	87500	312481	195973	31043	195.	71.6	37683838	19367210
SIT100	10262	34094	2710	1388	0.13	0.11	483383	417147
stokes128	49666	295938	18056	12738	1.14	1.06	3437116	2753749
stokes64	12546	74242	4292	3106	0.33	0.29	736428	577581

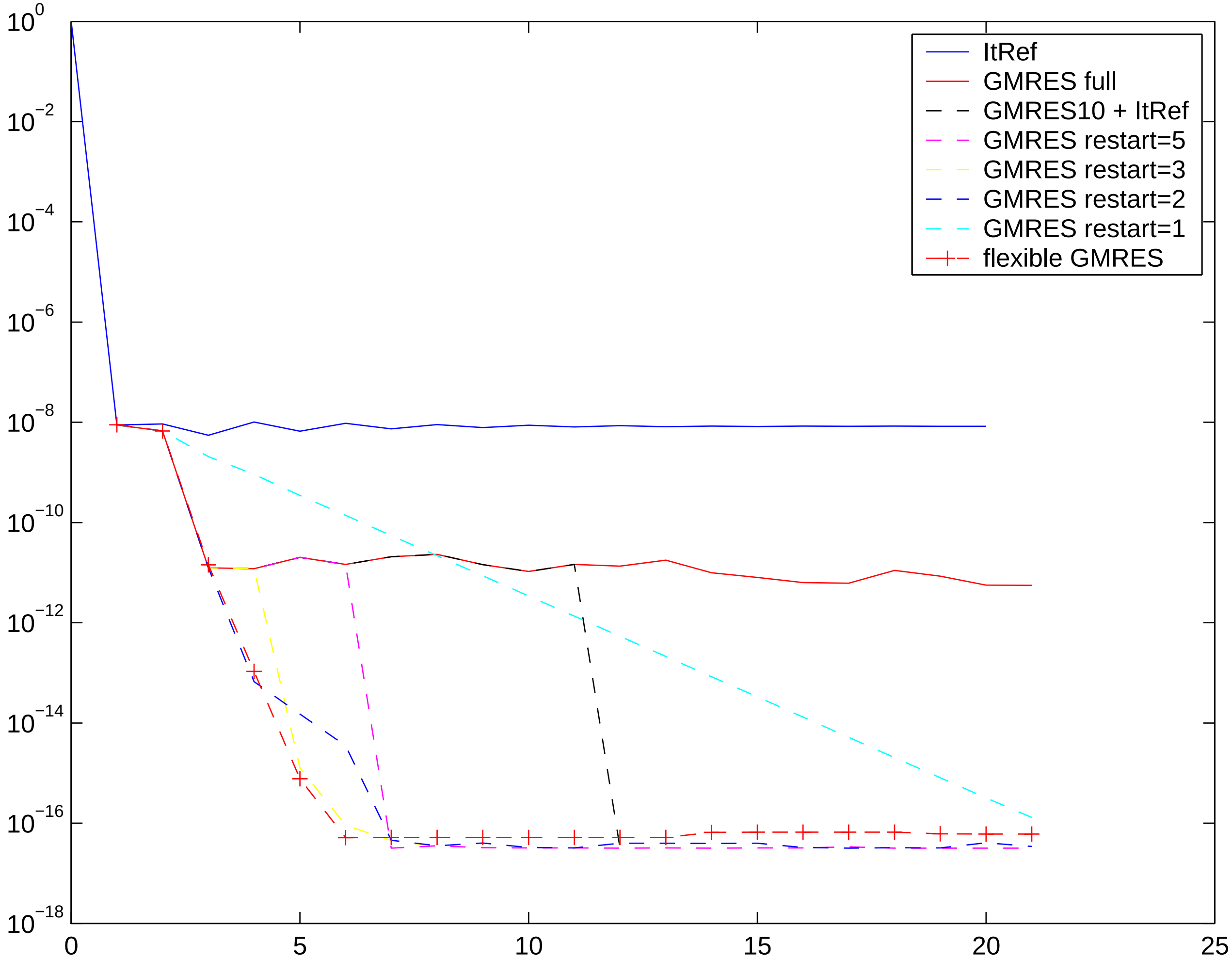
Component-wise backward error

Matrix	Num pivoting strategy		Static pivoting strategy		
	it. 0	it. 1	it. 0	it. 1	it. 2
BRAINPC2	1.6e-15	1.0e-15	2.1e-08	5.7e-15	9.8e-16
BRATU3D	2.0e-09	1.7e-16	9.2e-06	2.2e-11	1.7e-16
CONT-201	8.8e-11	1.6e-16	1.0e-05	9.4e-09	4.9e-09
CONT-300	7.6e-11	1.9e-16	2.1e-05	2.7e-09	2.5e-09
cvxqp3	5.2e-11	2.7e-16	8.5e-06	1.2e-12	3.4e-16
DTOC	2.1e-16	2.7e-20	8.3e-07	2.1e-13	1.9e-15
mario001	6.3e-15	1.3e-16	3.1e-08	2.5e-13	1.3e-16
NCVXQP1	4.6e-14	1.7e-17	4.9e-13	3.2e-15	2.6e-17
NCVXQP5	2.0e-11	2.0e-16	2.0e-08	6.7e-11	2.7e-14
NCVXQP7	9.6e-10	2.2e-16	4.9e-06	1.4e-12	2.2e-16
SIT100	4.4e-15	1.4e-16	2.0e-08	5.8e-15	1.5e-16
stokes128	1.1e-14	5.5e-16	4.2e-14	2.0e-15	1.7e-15
stokes64	4.3e-15	1.5e-15	1.6e-13	2.3e-14	2.2e-14

FGMRES

Arioli, Duff and Gratton have shown that using FGMRES rather than iterative refinement results in a backward stable method that converges for really quite poor factorizations of A.

Numerical experiments



Restarted GMRES vs. FGMRES on CONT-201 test example: $\tau = 10^{-8}$

Domain decomposition

Two PhD theses at CERFACS

Jean-Christophe Rioual
Solving linear systems for semiconductor device simulations on parallel distributed computers

CERFACS report: TH/PA/02/49

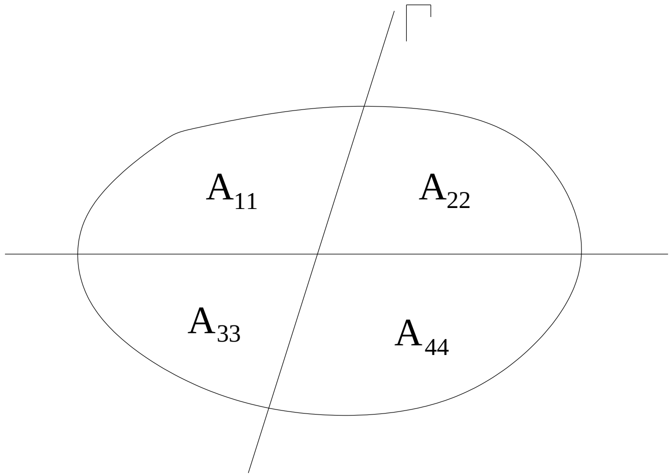
and

Azzam Haidar
On the parallel scalability of hybrid linear solvers for large 3D problems

CERFACS report: TH/PA/08/93

`www.cerfacs.fr/algor/`

Domain decomposition



Matrix representation is:

$$\begin{pmatrix} A_{11} & & & & A_{1\Gamma} \\ & A_{22} & & & A_{2\Gamma} \\ & & A_{33} & & A_{3\Gamma} \\ & & & A_{44} & A_{4\Gamma} \\ A_{\Gamma 1} & A_{\Gamma 2} & A_{\Gamma 3} & A_{\Gamma 4} & A_{\Gamma\Gamma} \end{pmatrix}$$

Schur complement is:

$$A_{\Gamma\Gamma} - \sum_{i=1}^{4} A_{\Gamma i} A_{ii}^{-1} A_{i\Gamma}$$

Domain Decomposition

$$\begin{pmatrix} A_{ii} & A_{i\Gamma} \\ A_{\Gamma i} & A_{\Gamma\Gamma}^{(i)} \end{pmatrix}$$

where

- A_{ii} : is the local subproblem,
- $A_{i\Gamma}$: is the boundary of the local problem, and
- $A_{\Gamma\Gamma}^{(i)}$: is the contribution to the stiffness matrix entries from variables on the artificial interface (Γ_i) around the ith subregion.

resulting in a contribution to the Schur complement of

$$S^{(i)} = A_{\Gamma\Gamma}^{(i)} - A_{\Gamma i} A_{ii}^{-1} A_{i\Gamma},$$

called a local Schur (complement).

Hybrid approach

We will use a direct method on the subproblems $\mathbf{A_{ii}}$

and

an iterative one (perhaps) on the Schur complement

MUMPS is used as the direct code

Science & Technology Facilities Council
Rutherford Appleton Laboratory

Non-overlapping Domain Decomposition

Algebraic Additive Schwarz preconditioner [L.Carvalho, L.Giraud, G.Meurant - 01]

$$\mathcal{S} = \sum_{i=1}^{N} \mathcal{R}_{\Gamma_i}^T \mathcal{S}^{(i)} \mathcal{R}_{\Gamma_i}$$

$$\mathcal{S} = \begin{pmatrix} \ddots & & & & \\ & \mathcal{S}_{kk} & \mathcal{S}_{k\ell} & & \\ & \mathcal{S}_{\ell k} & \mathcal{S}_{\ell\ell} & \mathcal{S}_{\ell m} & \\ & & \mathcal{S}_{m\ell} & \mathcal{S}_{mm} & \mathcal{S}_{mn} \\ & & & \mathcal{S}_{nm} & \mathcal{S}_{nn} \end{pmatrix} \Longrightarrow \mathcal{M} = \begin{pmatrix} \ddots & & & & \\ & \boxed{\begin{matrix}\mathcal{S}_{kk} & \mathcal{S}_{k\ell} \\ \mathcal{S}_{\ell k} & \mathcal{S}_{\ell\ell}\end{matrix}}^{-1} & & & \\ & & \boxed{\begin{matrix}\mathcal{S}_{\ell\ell} & \mathcal{S}_{\ell m} \\ \mathcal{S}_{m\ell} & \mathcal{S}_{mm}\end{matrix}}^{-1} & & \\ & & & \boxed{\begin{matrix}\mathcal{S}_{mm} & \mathcal{S}_{mn} \\ \mathcal{S}_{nm} & \mathcal{S}_{nn}\end{matrix}} & \end{pmatrix}$$

$$\mathcal{M} = \sum_{i=1}^{N} \mathcal{R}_{\Gamma_i}^T (\bar{\mathcal{S}}^{(i)})^{-1} \mathcal{R}_{\Gamma_i}$$

where $\bar{\mathcal{S}}^{(i)}$ is obtained from $\mathcal{S}^{(i)}$

$$\underbrace{\mathcal{S}^{(i)} = \begin{pmatrix} \mathcal{S}_{kk}^{(\iota)} & \mathcal{S}_{k\ell} \\ \mathcal{S}_{\ell k} & \mathcal{S}_{\ell\ell}^{(\iota)} \end{pmatrix}}_{\text{local Schur}} \Longrightarrow \underbrace{\bar{\mathcal{S}}^{(i)} = \begin{pmatrix} \mathcal{S}_{kk} & \mathcal{S}_{k\ell} \\ \mathcal{S}_{\ell k} & \mathcal{S}_{\ell\ell} \end{pmatrix}}_{\text{local assembled Schur}}$$

$$\searrow \quad \sum_{\iota \in adj} \mathcal{S}_{\ell\ell}^{(\iota)} \quad \nearrow$$

From Two to Three Dimensions

The main difference lies in the interface problem (Schur complement). In 2D the interface/interior ratio is small while in 3D there are severe problems in computing and storing the preconditioner.

Therefore, we must seek a cheaper alternative.

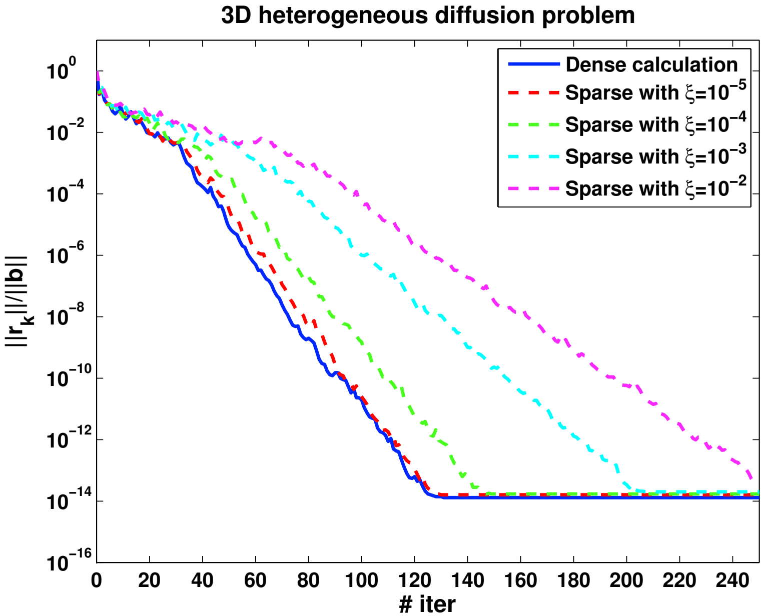
Two main ideas (used by Giraud and Haidar)

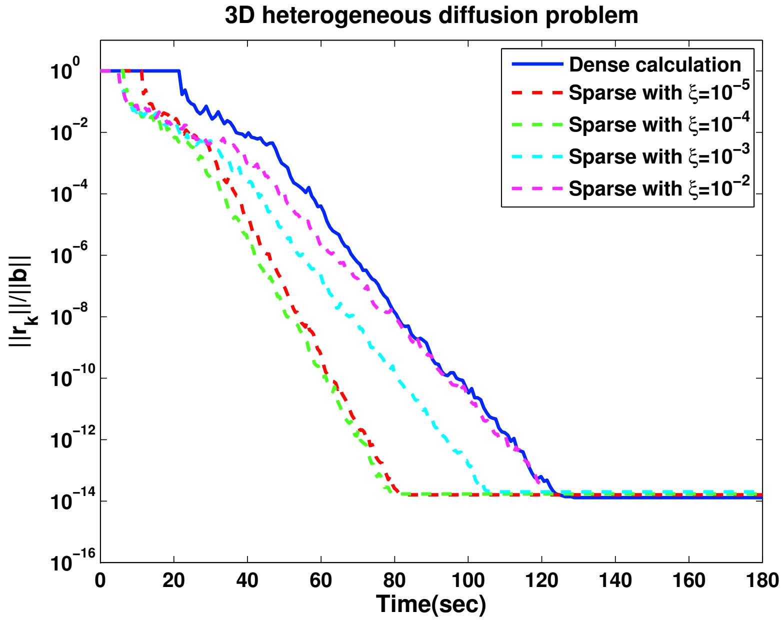
Sparsify the preconditioner

Set $s_{kl} = 0$ if $s_{kl} < \xi(|s_{kk}| + |s_{ll}|)$

Use 32-bit arithmetic

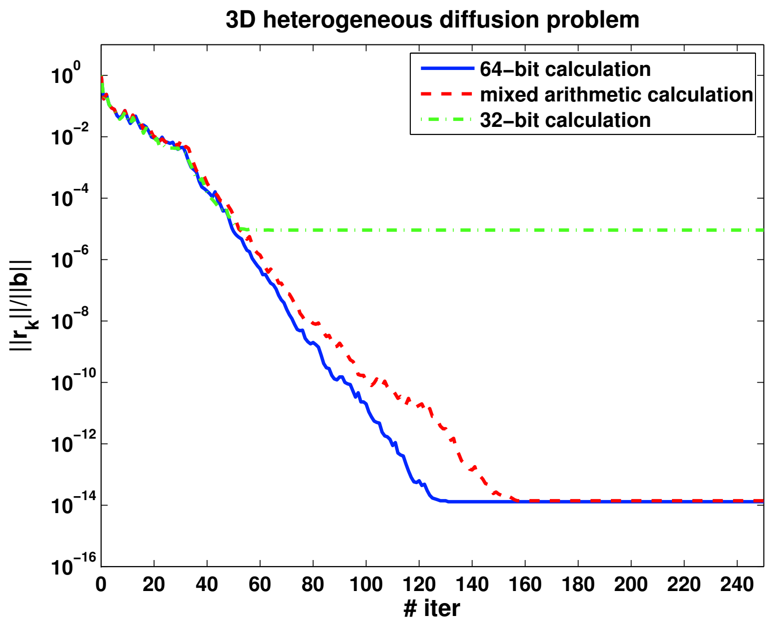
Diffusion Problem

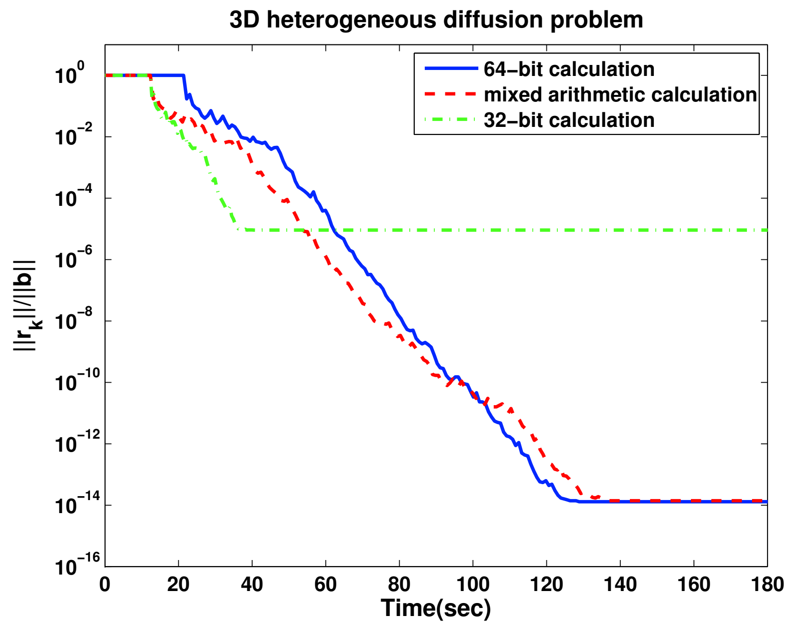




- Runs on System X
- 3D heterogeneous diffusion problem with $43 * 10^6$ on 1000 processors
- Graphs show effect of sparsification
- Even though more iterations are required, the sparsified versions are faster as the time per iteration and preconditioner setup require less time

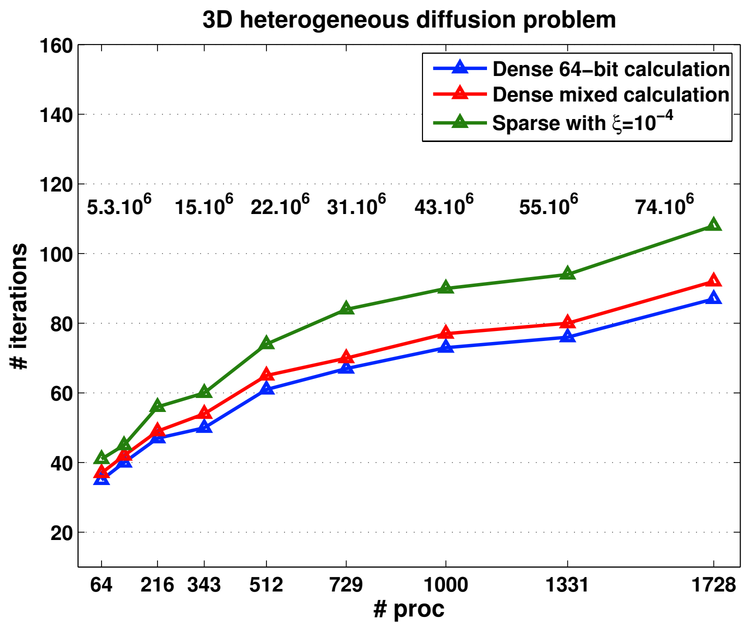
Diffusion Problem

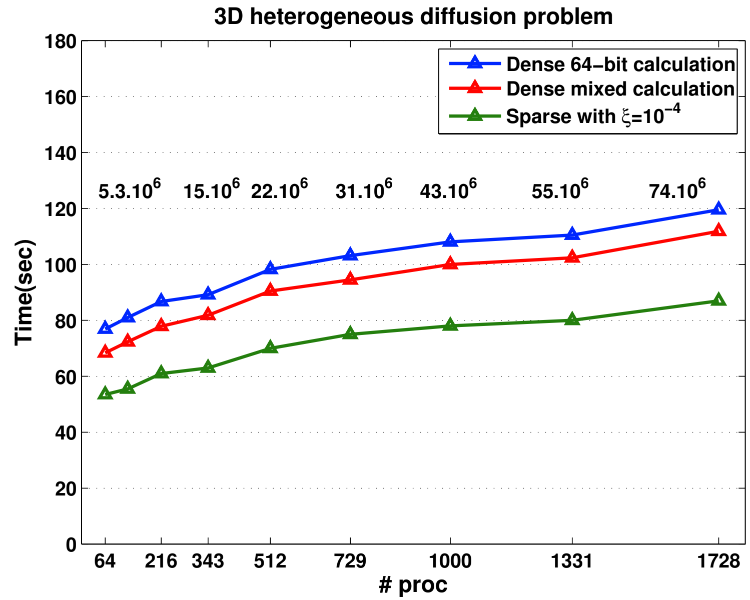




- 3D heterogeneous diffusion problem with $43 * 10^6$ on 1000 processors
- Graphs show effect of using mixed precision
- Although the number of iterations slightly increases, the mixed approach is fastest down to a level commensurate with the problem

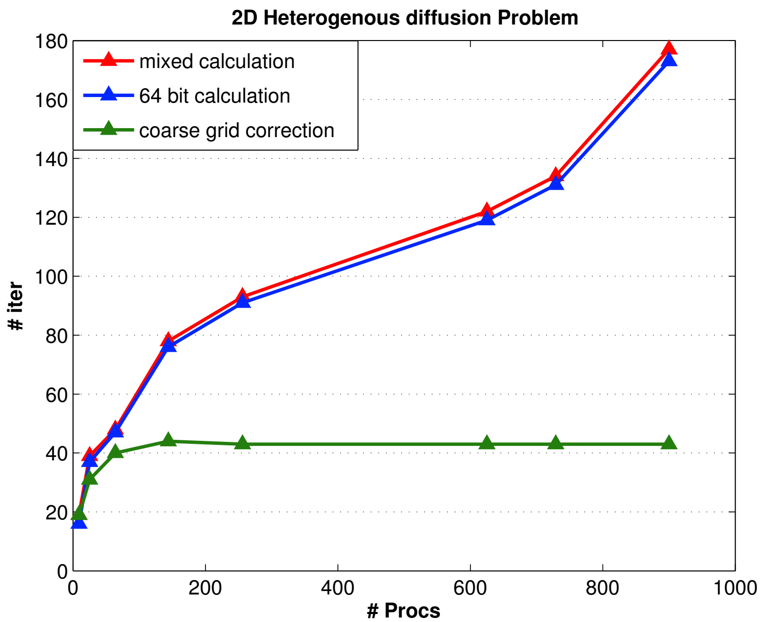
Diffusion Problem

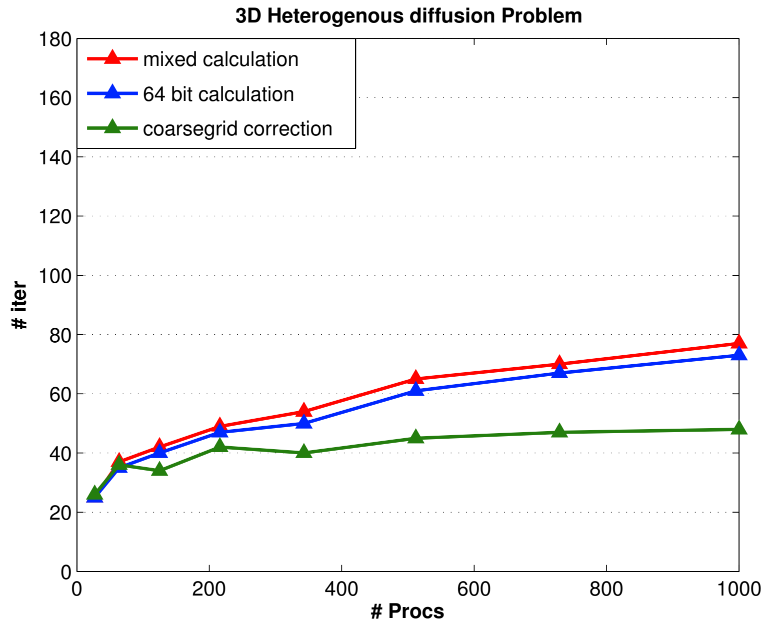




- 3D heterogeneous diffusion problem with size varying from 5.3 to $74 * 10^6$ degrees of freedom
- There is good scalability although the number of iterations grows with the number of subdomains
- Two ways to overcome this problem are:
  - Coarse grid correction
  - Two-level parallelism

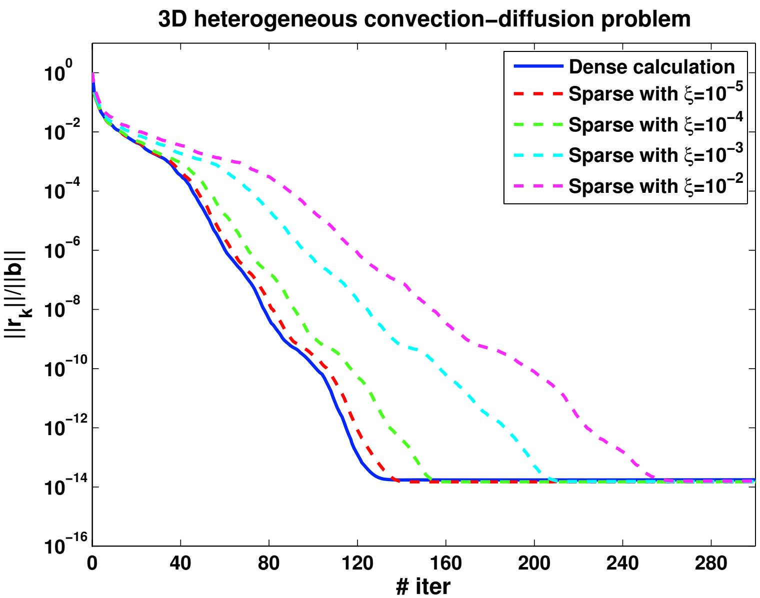
Effect of coarse grid correction

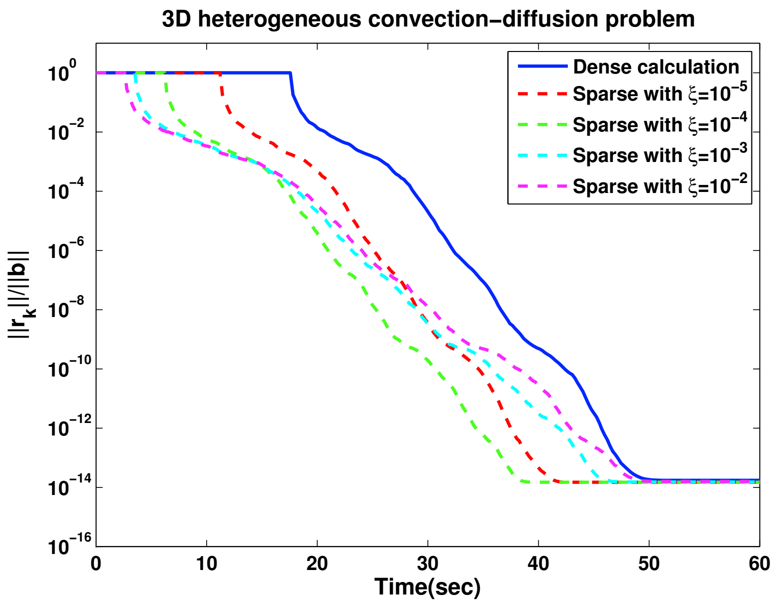




- Use as many degrees of freedom in the coarse space as subdomains
- Work of Carvalho, Giraud, Le Tallec (2001)

Convection-diffusion problem



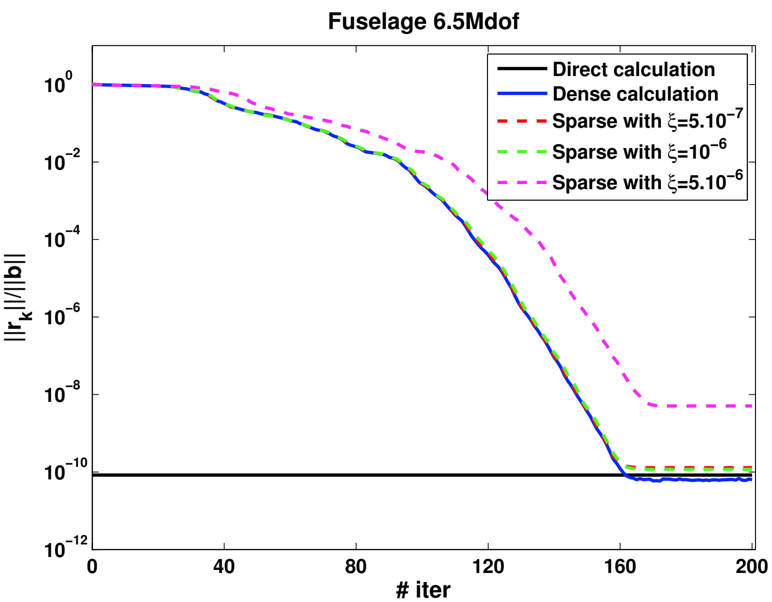


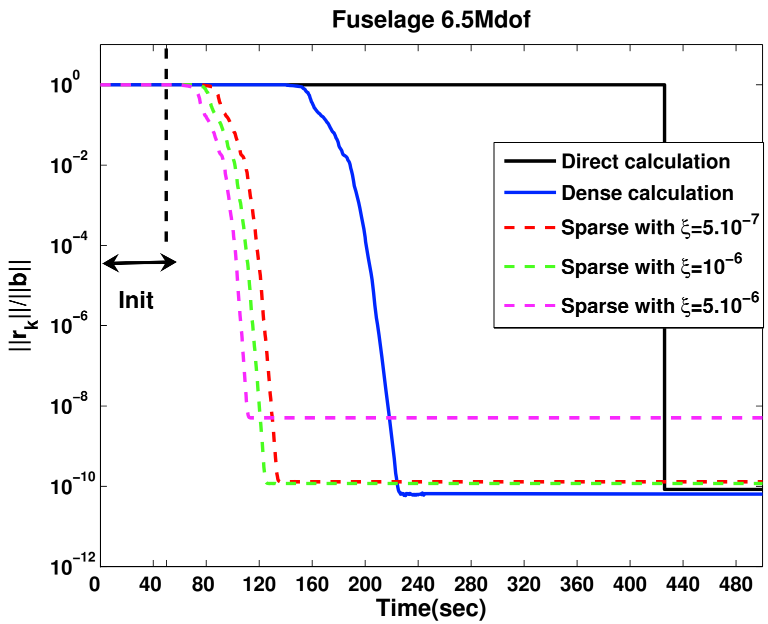
- 3D heterogeneous convection-diffusion problem with $27 * 10^6$ on 1000 processors
- Graphs show effect of sparsification
- Even though more iterations are required, the sparsified versions are faster as the time per iteration and preconditioner setup require less time.
- Roughly the same as for the pure diffusion problem

Industrial Problem

- Structural mechanics problem from Samtech (Pralet)
- Aircraft fuselage
- 6.5 million degrees of freedom

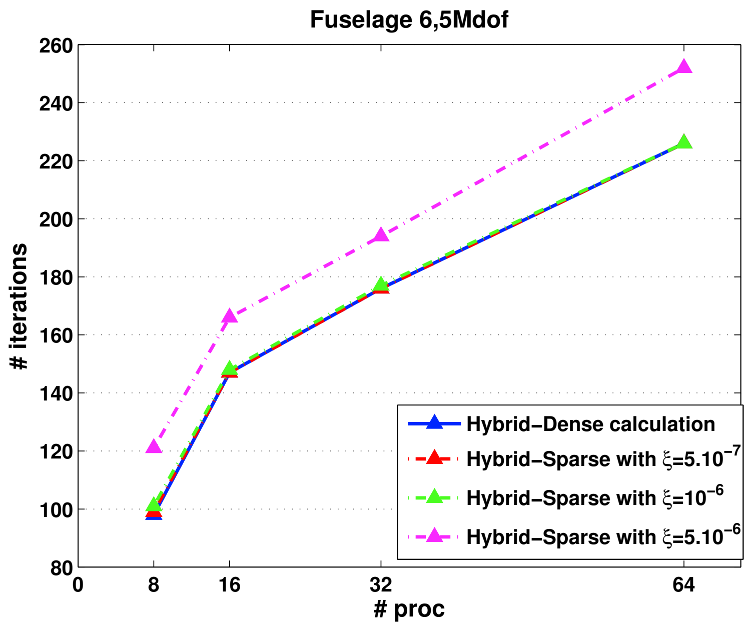
Fuselage Problem

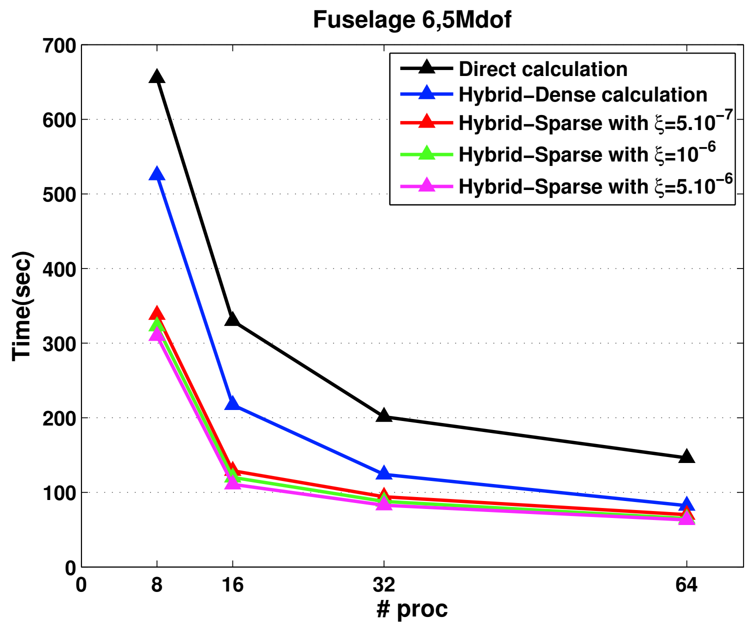




- Fuselage problem of 6.5 million dof mapped on 16 processors
- Runs on IBM JS21 at CERFACS
- The sparse preconditioner setup is four times faster than the dense one (19.5 v.s. 89 seconds)
- In term of global computing time, the sparse algorithm is about twice as fast
- The accuracy of the hybrid solver is comparable to that of the direct solver

Scalability of Fuselage Problem





- Fixed problem size: increasing the # of subdomains $\Longrightarrow$ an increase in the # of iterations
- Attractive speedups can be observed
- The sparsified variant is the most efficient

Two levels of parallelism on Fuselage

# total processors	Algo	# subdomains	# processors/ subdomain	# iter	iterative loop time
16 processors	*1-level parallel*	16	1	147	77.9
	2-level parallel	8	2	98	51.4
32 processors	*1-level parallel*	32	1	176	58.1
	2-level parallel	16	2	147	44.8
	2-level parallel	8	4	98	32.5
64 processors	*1-level parallel*	64	1	226	54.2
	2-level parallel	32	2	176	40.1
	2-level parallel	16	4	147	31.3
	2-level parallel	8	8	98	27.4

- Reduce the number of subdomains $\Longrightarrow$ reduce the number of iterations
- Though the subdomain size increases, the time for the iterative loop decreases because:
  - The number of iterations decreases
  - Each subdomain is handled in parallel

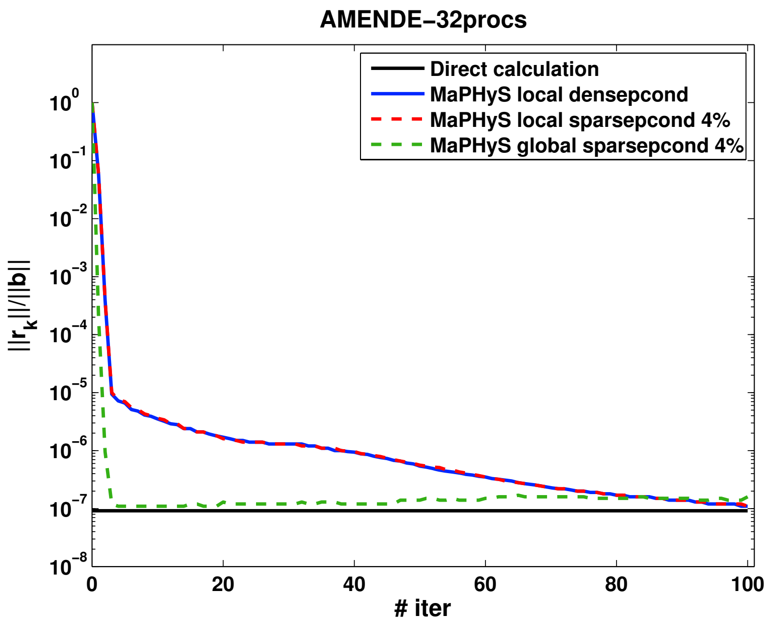
Domain decomposition without the mesh

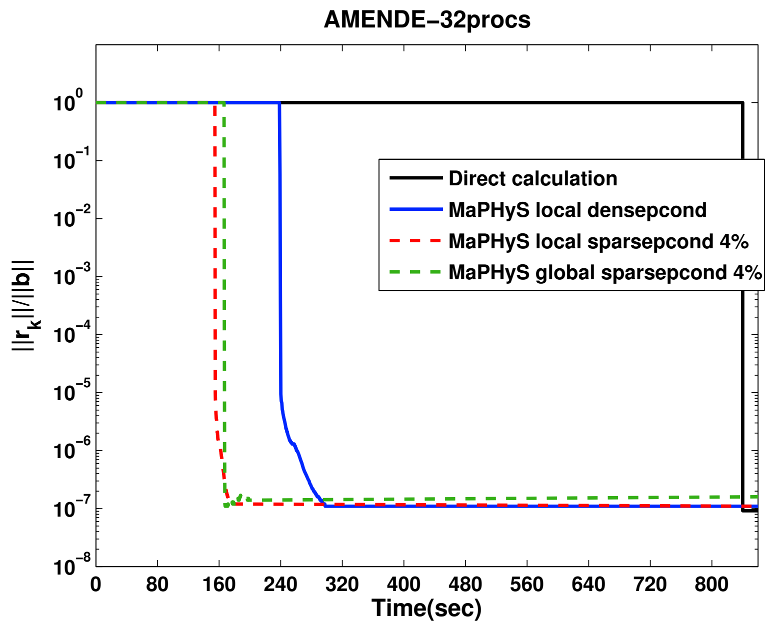
Quite recently, Azzam Haidar has been experimenting with matrices which are given as a sparse algebraic data structure without any information about the original problem or the grid. We now show his results from two industrial problems: `AMENDE` and `AUDI`, the first from CEA-CESTA and the second from the PARASOL project.

Their characteristics are:

Problem	Application	order	number entries
Amende	Electromagnetics	6,994,683	58,477,383
Audi	Structures	943,695	39,297,771

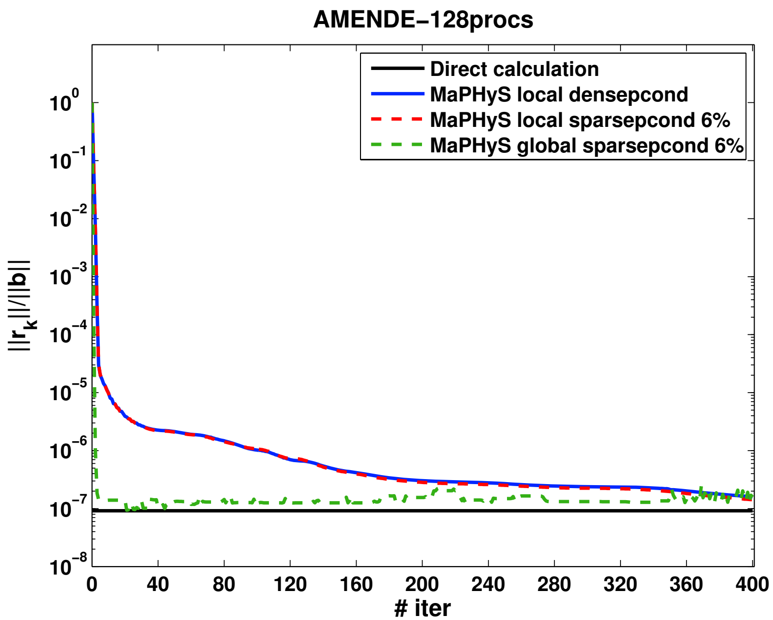
Amende Problem .. 32 processors

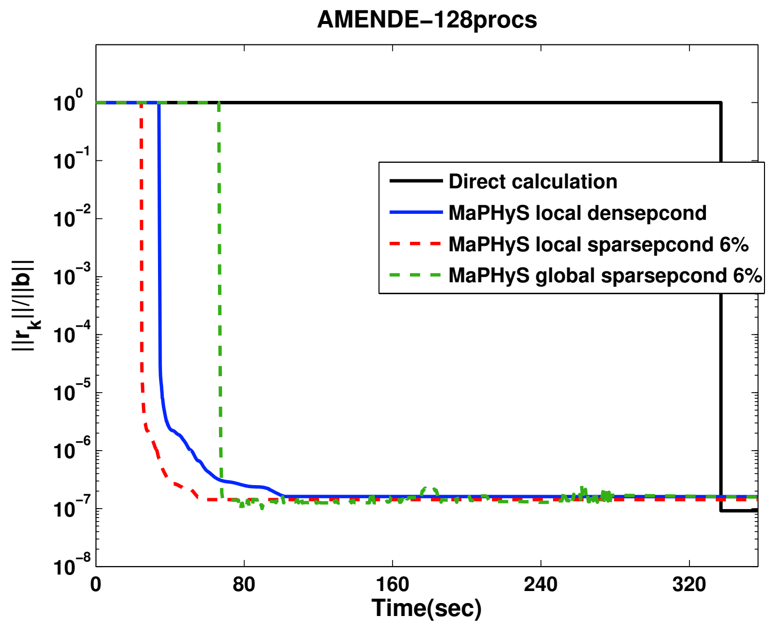




- Amende problem of 6.99M dof mapped on 32 processors
- Sparse algorithm is twice as fast
- Global sparse conditioner performs well
- Accuracy of hybrid solver is comparable with direct solver

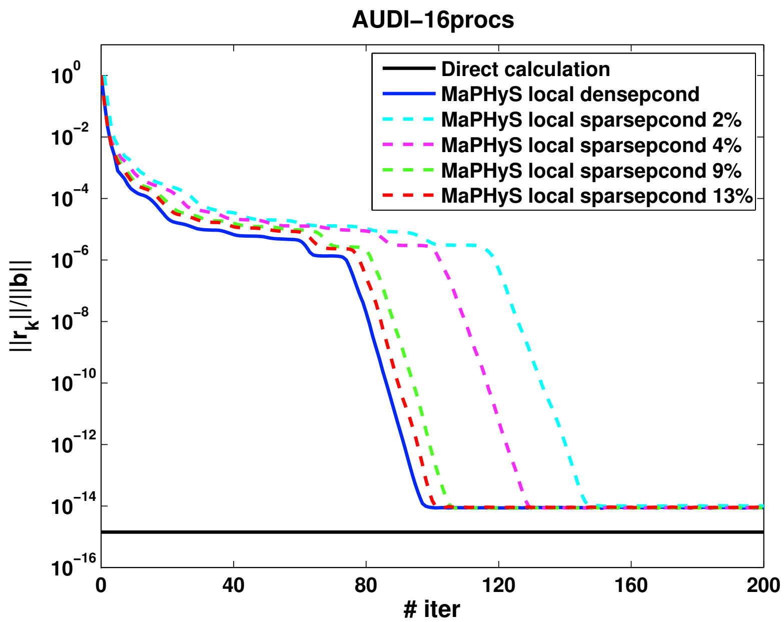
Amende Problem .. 128 processors

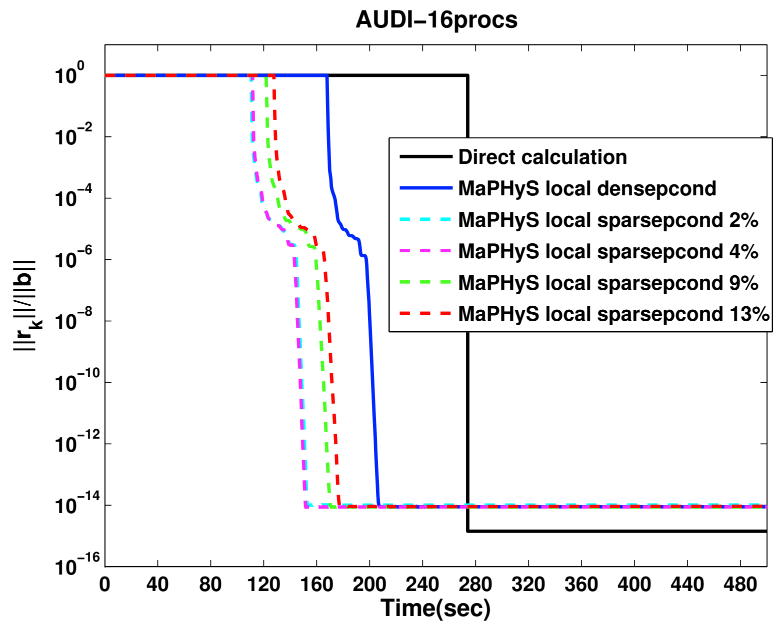




- Amende problem of 6.99M dof mapped on 128 processors
- Sparse algorithm is similar to dense
- Dense preconditioner works well because local Schurs are small
- Global sparse conditioner is good numerically but slower

AUDI Problem .. 16 processors





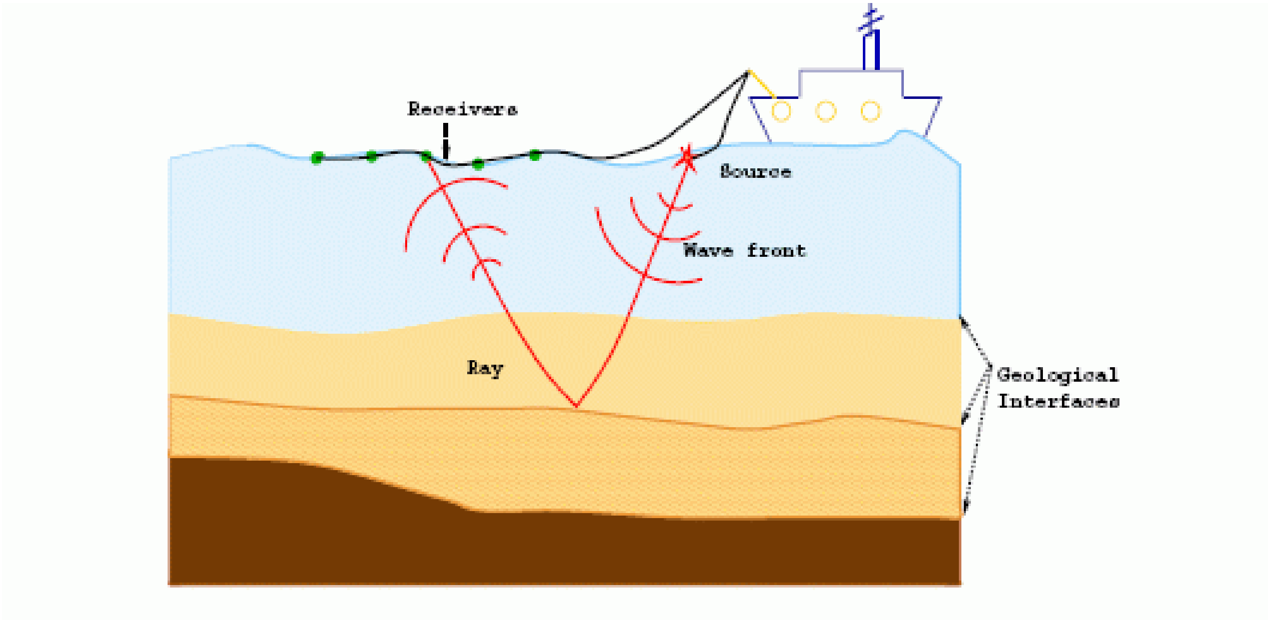
- Audi problem of 0.9M dof mapped on 16 processors
- For very small ξ convergence only marginally affected but memory savings are substantial
- For larger ξ memory is reduced but convergence is poor
- Sparsified versions require more iterations but are faster
- Accuracy of hybrid solver is comparable with direct solver

Helmholtz Equation in Geophysics

Work with

Serge Gratton
Xavier Vasseur
and
Xavier Pinel

at CERFACS



Technical Report: CERFACS:TR/PA/07/03 and RAL-TR-2007-002

Helmholtz problem

- Helmholtz equation in the frequency domain:

$$-\Delta u - \frac{\omega^2}{v^2} u = g \quad \text{in} \quad \Omega$$

- with radiation boundary conditions [$k = \frac{\omega}{v}$: wavenumber]:

$$\frac{\partial u}{\partial n} - i\,k\,u = 0 \quad or \quad \frac{\partial u}{\partial n} - i\,k\,u - \frac{i}{2\,k}\frac{\partial^2 u}{\partial^2 \tau} = 0 \quad \text{on} \quad \delta\Omega$$

- or with Perfectly Matched Layer (PML) [Berenger, 1994]

Notation:

$\omega = 2\pi\, f$ is the angular frequency, v the velocity of the wave, u the pressure of the wave, g represents the source term

Helmholtz problem with PML formulation

- Ω is divided into two sets: Ω_I and Ω_{PML}
- PDE with variable coefficients must now be solved:

$$\begin{cases} \frac{-\omega^2\, u}{v^2(x,y,z)} - \frac{1}{\xi_x(x)}\frac{\partial}{\partial x}\left(\frac{1}{\xi_x(x)}\frac{\partial u}{\partial x}\right) - \frac{1}{\xi_y(y)}\frac{\partial}{\partial y}\left(\frac{1}{\xi_y(y)}\frac{\partial u}{\partial y}\right) - \frac{1}{\xi_z(z)}\frac{\partial}{\partial z}\left(\frac{1}{\xi_z(z)}\frac{\partial u}{\partial z}\right) = g \\ \\ u = 0 \ \text{ on } \delta\Omega = \delta\Omega_{PML} \end{cases}$$

- Variable complex-valued coefficients only in Ω_{PML}:

$$\xi_d(\delta) = 1 \ \text{ in } \Omega_I \quad \text{and} \quad \xi_d(\delta) = 1 + i\,\frac{\eta_d(\delta)}{\omega} \quad \text{in } \Omega_{PML}$$

for $d = x, y, z$ and where η_d is called a PML function.

- PML function [Operto et al., 2004]

$$\eta_d(\delta) = c_{PML}\, cos\left(\frac{\pi}{2L_{PML}}\delta\right) \quad \text{in } \Omega_{PML}$$

where L_{PML} is the width of the PML and c_{PML} is a real positive number.

Discretized problem

- Ω is always box shaped
- Second-order finite difference discretization methods on non-uniform grids
- Seven-point discretization in three dimensions

- Accuracy requirement for second order discretization: $k\,h \leq \dfrac{\pi}{6}$ for 12 points per wavelength
- This leads to a large complex sparse linear system (symmetric in case of radiation boundary conditions)

State of the art solution schemes

- **Sparse multifrontal direct methods:**
  - Very robust but requires too much storage for large-scale problems
- **Multigrid methods:**
  - Multigrid as a solver on the original Helmholtz problem [Elman et al, 2001].
  - Geometric multigrid preconditioner on a complex shifted Helmholtz operator [Erlangga, Oosterlee, Vuik, 2006].

Hybrid preconditioner

- We use a two-level grid to avoid both smoothing and coarse grid correction difficulties and simultaneously to benefit from the robustness and computational efficiency of modern sparse direct solvers.
- We thus use a direct method on the nearby problem from a not too coarse grid from multigrid applied to the original Helmholtz equation.
- Multigrid is not a convergent method but acts as a preconditioner for the original (unshifted) Helmholtz operator
- Eigenspectrum of AC^{-1} is clustered around 1 with the isolated eigenvalues captured using Krylov subspace methods

Numerical results

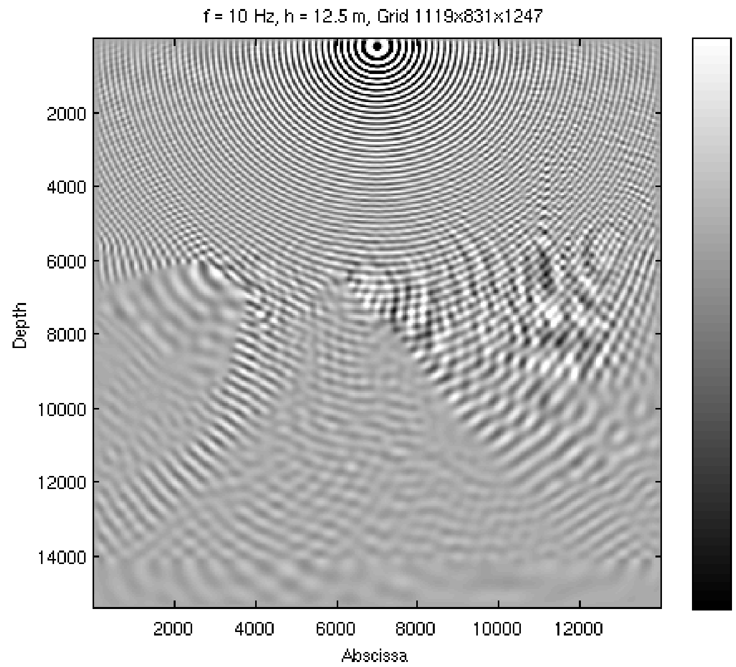
Constant wavenumber: Runs on the CERFACS IBM JS21

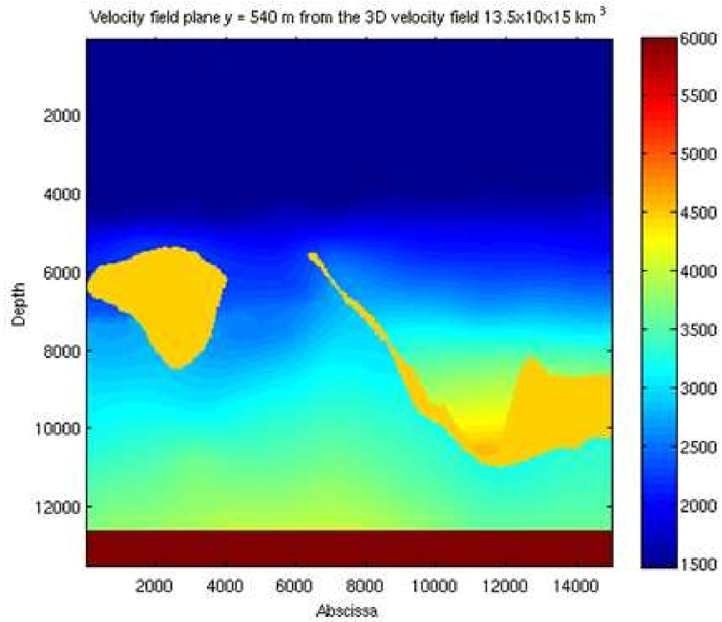
Two-grid preconditioned FGMRES(5)					
k	Grid	It	Time (s) Fac.	Mem. (Mb) Fac.	Proc
30	64^3	10	3.94	529	2
45	96^3	11	33.24	3323	3
60	128^3	12	73.38	11359	16
90	192^3	13	696.21	62970	32

- Smoother: Gauss-Seidel
- Direct method: MUMPS
- Robustness of the two-grid approach with respect to the wavenumber k

Where are the challenges ?

Heterogeneous velocity field: $13.5 \times 10 \times 15\ km^3$, f = 10 Hz, h = 12.5m.





- Problem size of 1.16×10^9 unknowns to be solved for multiple sources (around 500 to 1000 in practice) !
- Indefinite complex-valued problem known as difficult for iterative methods !

Geometric two-grid preconditioner

Two-grid preconditioner

- One cycle of a two-grid method is used as a preconditioner
- Krylov "smoother" as in [Elman, 2001] and [Adams, 2007]: preconditioned **GMRES(2)**
- Trilinear interpolation and adjoint as restriction
- GMRES(m) as coarse grid solver to solve only approximately the coarse grid systems: preconditioned **GMRES(10)**

Outer Krylov subspace method

- Flexible GMRES [Saad, 1993]: **FGMRES(5)**

Geometric two-grid preconditioner

- Stopping criterion: $\frac{\|\bar{r}^{(it)}\|_2}{\|\bar{r}^{(0)}\|_2} \leq 10^{-1}$ with a maximum of 100 iterations of **GMRES(10)** for the coarse grid
- Stopping criterion: $\frac{\|r^{(it)}\|_2}{\|r^{(0)}\|_2} \leq 10^{-6}$ with zero initial guess

Three-dimensional benchmark problems

- Both homogeneous and heterogeneous velocity fields
- PML formulation with 15 points on each side of the domain
- Experiments performed on BG/L and BG/P

Homogeneous velocity field on BG/P

Weak scalability experiments [fixed local problem size per core]

[**PRACE Summer School, Stockholm, 2008**]						
$1/h$	Grid	# Cores	Time (s)	It	Time/It	Mem (GB)
1024	1024^3	1024	1687	58	29.08	170
2048	2048^3	8192	3718	127	29.28	1362
4096	4096^3	65536	9634	327	29.46	10892

- Computations performed in single precision arithmetic
- Velocity is homogeneous and equal to $1500\ m\ s^{-1}$
- The wavenumber k is variable ($k\,h = \pi/6$)
- Number of iterations (It) increases linearly with k
- The time per iteration is nearly constant
- Memory required (Mem) is increased by a factor of 8 as expected
- A sparse indefinite linear system of more than 68 billion unknowns has been solved

Homogeneous velocity field on BG/P

Strong scalability experiments [fixed global problem size]

[PRACE Summer School, Stockholm, 2008]						
$1/h$	Grid	# Cores	Time (s)	It	Time/It	Mem (GB)
2048	2048^3	4096	7706	128	60.20	1341
2048	2048^3	8192	3719	127	29.28	1361
2048	2048^3	16384	1773	128	13.85	1382
2048	2048^3	32768	798	129	6.19	1404

- Computations performed in single precision arithmetic
- Velocity is homogeneous and equal to $1500\ m\ s^{-1}$
- The wavenumber k is now fixed: $k\,h = \pi/6$
- Number of iterations (It) is almost independent of the number of cores
- The time per iteration is divided by a factor of 2 as expected [factor greater than 2 due to cache effects]

Heterogeneous velocity field on BG/L

Experiments on BG/L ($13.5 \times 10 \times 15\ km^3$ domain).

Grid	h (m)	f (Hz)	Processors	It	T (min)
$295 \times 227 \times 327$	50	2.5	16	39	25
$567 \times 431 \times 639$	25	5.0	128	83	47
$1119 \times 831 \times 1247$	12.5	10.0	1024	205	107

- Computations performed in double precision arithmetic
- Minimum and maximum velocity are $1500\ m\ s^{-1}$ and $6000\ m\ s^{-1}$
- Number of iterations increases still linearly with the frequency

Heterogeneous velocity on BG/P IDRIS

Experiments on BG/P (SEG/EAGE Overthrust domain $20 \times 20 \times 5\ km^3$).

Grid	h (m)	f (Hz)	Processors	It	T (min)
$863 \times 863 \times 231$	24.21	7.5	64	37	2678
$1690 \times 1690 \times 426$	12.11	15.0	512	102	6362
$3356 \times 3356 \times 829$	6.05	30.0	4096	490	28601

- Computations performed in double precision arithmetic
- Minimum and maximum velocity are $2200\ m\ s^{-1}$ and $6000\ m\ s^{-1}$
- Number of iterations no longer increases linearly with the frequency

Heterogeneous velocity on BG/P IDRIS

Experiments on BG/P (SEG/EAGE salt domain $8 \times 8 \times 4\ km^3$ domain).

Grid	h (m)	f (Hz)	Processors	It	T (min)
$671 \times 671 \times 351$	12.500	10	64	43	2797
$1311 \times 1311 \times 671$	6.250	20	512	101	6117
$2597 \times 2597 \times 1317$	3.125	40	4096	283	16492

- Computations performed in double precision arithmetic
- Minimum and maximum velocity are $1500\ m\ s^{-1}$ and $4400\ m\ s^{-1}$
- Number of iterations no longer increases linearly with the frequency

Conclusions

We can solve really large, realistic and computationally challenging problems in important application areas.

A range of techniques involving both sparse direct and a range of sparse iterative solvers is required including hybrid methods.

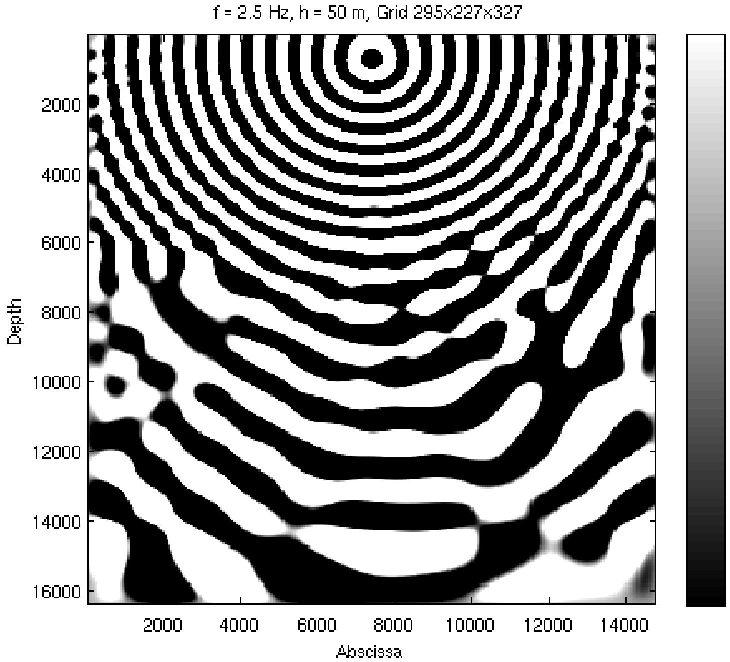
Conclusions

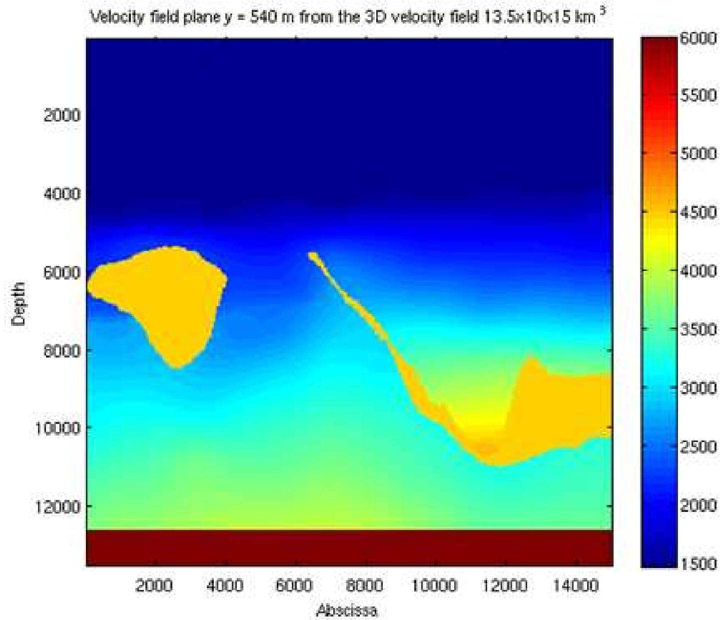
THANK YOU

for your attention

Heterogeneous velocity field on BG/L

$13.5 \times 10 \times 15\ km^3$, f = 2.5 Hz

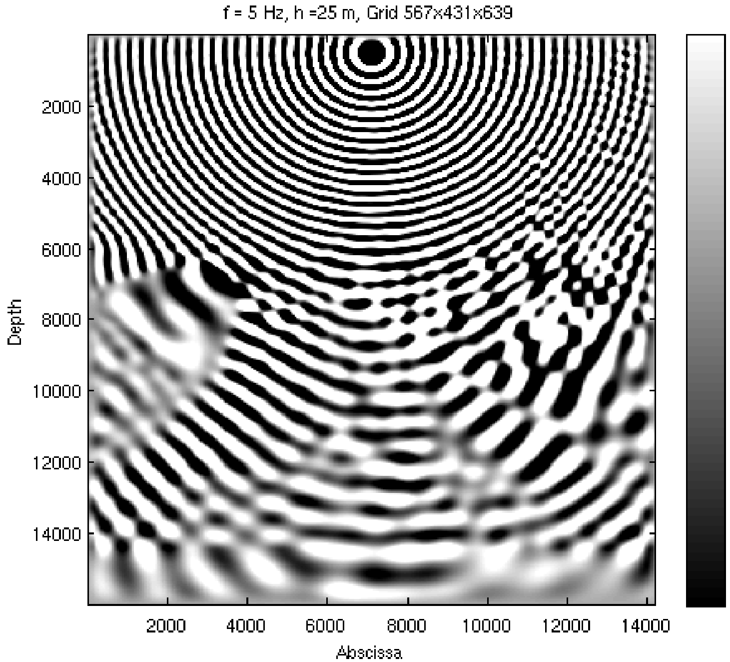


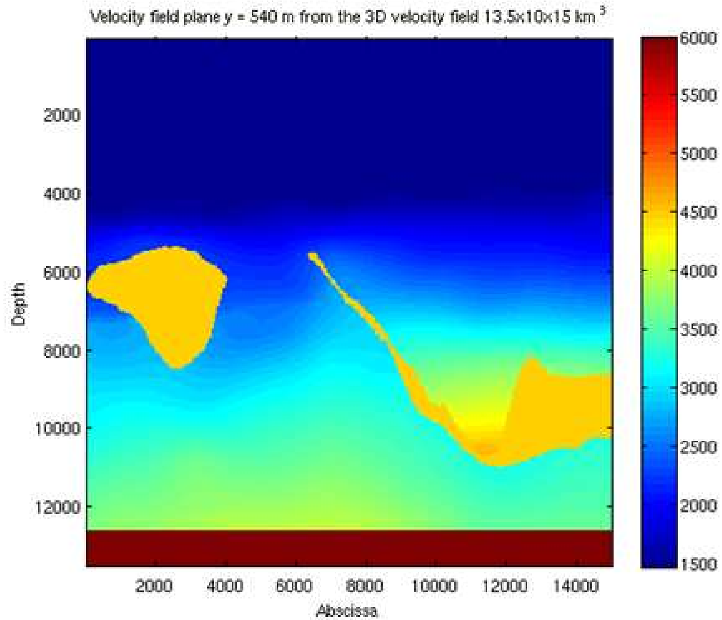


- Problem size of 2.19×10^7 unknowns

Heterogeneous velocity field on BG/L

$13.5 \times 10 \times 15\ km^3$, f = 5 Hz

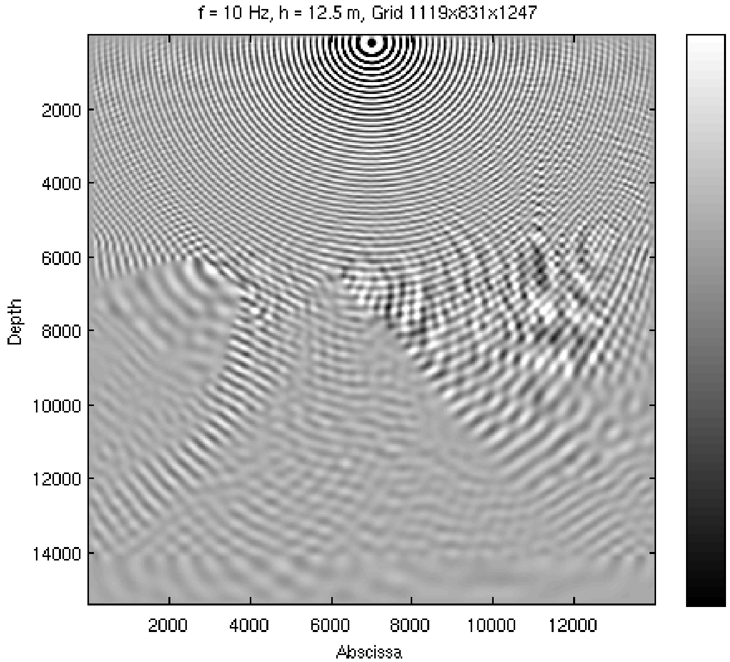


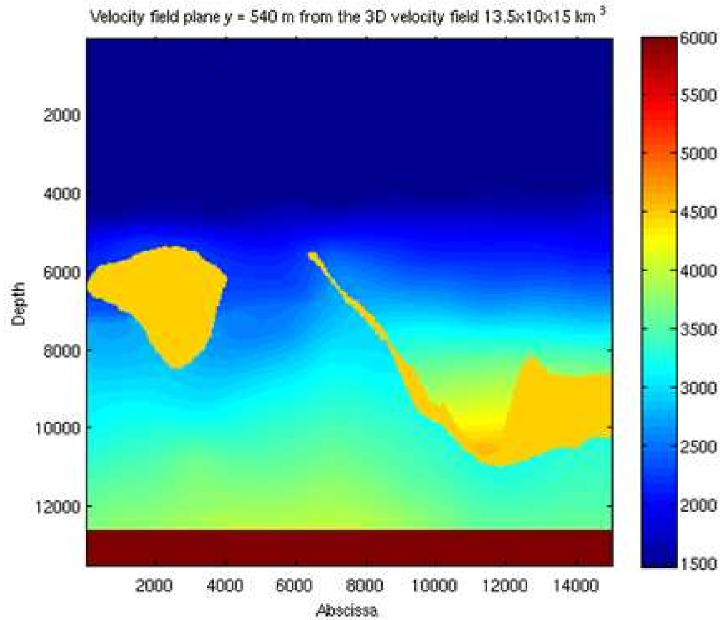


- Problem size of 1.56×10^8 unknowns

Heterogeneous velocity field on BG/L

$13.5 \times 10 \times 15\ km^3$, f = 10 Hz

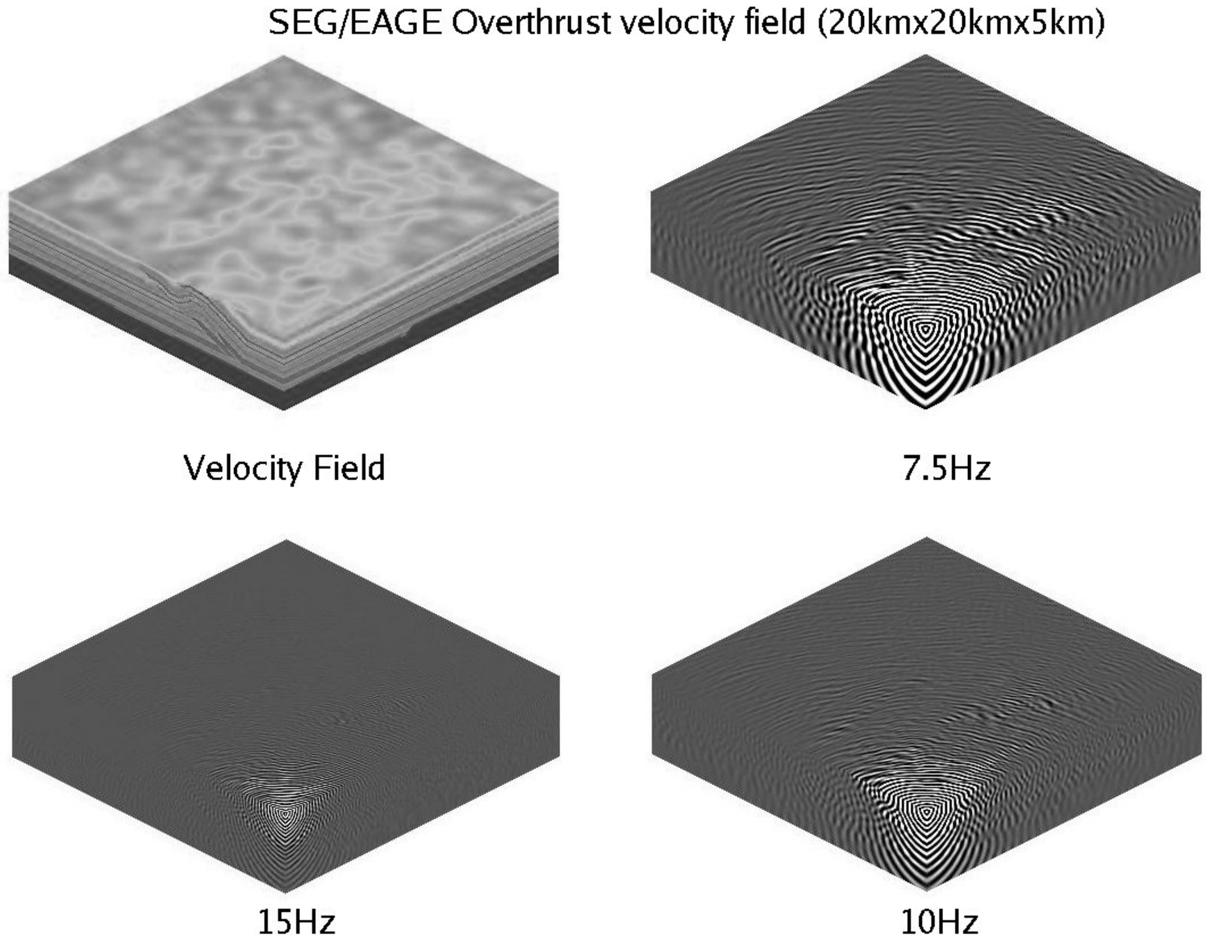




- Problem size of 1.16×10^9 unknowns

SEG/EAGE Overthrust velocity field on BG/P

$20 \times 20 \times 5 \ km^3$



SEG/EAGE Salt velocity field on BG/P

$8 \times 8 \times 4\, km^3$