Efficient resolution of time-harmonic Maxwell equations with high-order edge finite elements

M. Durufle, G Cohen

INRIA, project POEMS

23rd july 2007

Bibliography and motivation

- Cohen, Monk, mass lumping for Maxwell's equations (hexahedra)
- S. Fauqueux, mixed spectral elements for wave and elastic equations (hexahedra)
- S. Pernet, Discontinuous Galerkin methods for Maxwell's equations (hexahedra)
- M. Durufle, Numerical integration and high order finite element method applied to time-harmonic Maxwell equations
- Apply techniques of "mass lumping" and "mixed formulation", which are efficient in temporal domain
- Efficient preconditioning technique to solve linear system

Nedelec's first family on hexahedra

Space of approximation

$$V_h = \{ \vec{u} \in H(\mathrm{curl},\Omega) \text{ so that } DF_i^t \, \vec{u} \circ F_i \in Q_{r-1,r,r} \times Q_{r,r-1,r} \times Q_{r,r,r-1} \}$$

Basis functions

$$\hat{\vec{\varphi}}^1_{i,j,k}(\hat{x},\hat{y},\hat{z}) = \hat{\psi}^G_i(\hat{x})\,\hat{\psi}^{GL}_j(\hat{y})\,\hat{\psi}^{GL}_k(\hat{z})\,\vec{e_x} \quad 1 \le i \le r \quad 1 \le j,k \le r+1$$

$$\hat{\vec{\varphi}}^2_{j,i,k}(\hat{x},\hat{y},\hat{z}) = \hat{\psi}^{GL}_j(\hat{x})\,\hat{\psi}^{G}_i(\hat{y})\,\hat{\psi}^{GL}_k(\hat{z})\,\vec{e_y} \quad 1 \le i \le r \quad 1 \le j,k \le r+1$$

$$\hat{\vec{\varphi}}^3_{k,j,i}(\hat{x},\hat{y},\hat{z}) = \hat{\psi}^{GL}_k(\hat{x})\,\hat{\psi}^{GL}_j(\hat{y})\,\hat{\psi}^{G}_i(\hat{x})\,\vec{e_z} \quad 1 \le i \le r \quad 1 \le j,k \le r+1$$

ψ^G_i, ψ^{GL}_i lagragian functions linked respectively with Gauss points and Gauss-Lobatto points.
See. G. Cohen, P. Monk, Gauss points mass lumping

Nedelec's first family on hexahedra

Space of approximation

$$V_h = \{ \vec{u} \in \ \mathrm{H(curl,\Omega)} \ \text{so that } DF_i^t \, \vec{u} \circ F_i \in Q_{r-1,r,r} \times Q_{r,r-1,r} \times Q_{r,r,r-1} \}$$

Basis functions

$$\hat{\vec{\varphi}}^1_{i,j,k}(\hat{x},\hat{y},\hat{z}) = \hat{\psi}^G_i(\hat{x})\,\hat{\psi}^{GL}_j(\hat{y})\,\hat{\psi}^{GL}_k(\hat{z})\,\vec{e_x} \quad 1 \le i \le r \quad 1 \le j,k \le r+1$$

$$\hat{\vec{\varphi}}^2_{j,i,k}(\hat{x},\hat{y},\hat{z}) = \hat{\psi}^{GL}_j(\hat{x})\,\hat{\psi}^{G}_i(\hat{y})\,\hat{\psi}^{GL}_k(\hat{z})\,\vec{e_y} \quad 1 \le i \le r \quad 1 \le j,k \le r+1$$

$$\hat{\vec{\varphi}}^3_{k,j,i}(\hat{x},\hat{y},\hat{z}) = \hat{\psi}^{GL}_k(\hat{x})\,\hat{\psi}^{GL}_j(\hat{y})\,\hat{\psi}^{G}_i(\hat{x})\,\vec{e_z} \quad 1 \le i \le r \quad 1 \le j,k \le r+1$$

ψ_i^G, ψ_i^{GL} lagragian functions linked respectively with Gauss points and Gauss-Lobatto points.

See. G. Cohen, P. Monk, Gauss points mass lumping

Nedelec's first family on hexahedra

Space of approximation

$$V_h = \{ \vec{u} \in \mathrm{H(curl,\Omega)} \text{ so that } DF_i^t \, \vec{u} \circ F_i \in Q_{r-1,r,r} \times Q_{r,r-1,r} \times Q_{r,r,r-1} \}$$

Basis functions

$$\hat{\vec{\varphi}}^1_{i,j,k}(\hat{x},\hat{y},\hat{z}) = \hat{\psi}^G_i(\hat{x}) \, \hat{\psi}^{GL}_j(\hat{y}) \, \hat{\psi}^{GL}_k(\hat{z}) \, \vec{e_x} \quad 1 \le i \le r \quad 1 \le j,k \le r+1$$

$$\hat{\vec{\varphi}}^2_{j,i,k}(\hat{x},\hat{y},\hat{z}) = \hat{\psi}^{GL}_j(\hat{x}) \, \hat{\psi}^{G}_i(\hat{y}) \, \hat{\psi}^{GL}_k(\hat{z}) \, \vec{e_y} \quad 1 \le i \le r \quad 1 \le j,k \le r+1$$

$$\hat{\vec{\varphi}}^3_{k,j,i}(\hat{x},\hat{y},\hat{z}) = \hat{\psi}^{GL}_k(\hat{x}) \, \hat{\psi}^{GL}_j(\hat{y}) \, \hat{\psi}^{G}_i(\hat{x}) \, \vec{e_z} \quad 1 \le i \le r \quad 1 \le j,k \le r+1$$

ψ_i^G, ψ_i^{GL} lagragian functions linked respectively with Gauss points and Gauss-Lobatto points.
See. G. Cohen, P. Monk, Gauss points mass lumping

Elementary matrices

Mass matrix :

$$(M_h)_{i,j} \;=\; \int_{\hat{K}} J_i \, DF_i^{-1} \, \varepsilon \, DF_i^{*-1} \, \hat{\varphi}_i \cdot \hat{\varphi}_k \, d\hat{x}$$

Stiffness matrix :

$$(K_h)_{i,j} \;=\; \int_{\hat{K}} \frac{1}{J_i} \, DF_i^t \, \mu^{-1} \, DF_i \, \hat{\nabla} \times \hat{\varphi}_i \cdot \hat{\nabla} \times \hat{\varphi}_k \, d\hat{x}$$

- Use of Gauss-Lobatto quadrature $(\omega_k^{GL}, \xi_k^{GL})$
- Block-diagonal matrix

$$(A_h)_{k,k} = \left[J_i \, DF_i^{-1} \, \varepsilon \, DF_i^{*-1} \right] (\xi_k^{GL}) \omega_k^{GL}$$

- Block-diagonal matrix

$$(B_h)_{k,k} = \left[\frac{1}{J_i} DF_i^t \mu^{-1} \, DF_i \right] (\xi_k^{GL}) \omega_k^{GL}$$

Elementary matrices

Mass matrix :

$$(M_h)_{i,j} \; = \; \int_{\hat{K}} J_i \, DF_i^{-1} \, \varepsilon \, DF_i^{*-1} \, \hat{\varphi}_i \cdot \hat{\varphi}_k \, d\hat{x}$$

Stiffness matrix :

$$(K_h)_{i,j} \; = \; \int_{\hat{K}} \frac{1}{J_i} \, DF_i^t \, \mu^{-1} \, DF_i \, \hat{\nabla} \times \hat{\varphi}_i \cdot \hat{\nabla} \times \hat{\varphi}_k \, d\hat{x}$$

- Use of Gauss-Lobatto quadrature $(\omega_k^{GL}, \xi_k^{GL})$
- Block-diagonal matrix

$$(A_h)_{k,k} = \left[J_i \, DF_i^{-1} \, \varepsilon \, DF_i^{*-1} \right] (\xi_k^{GL}) \omega_k^{GL}$$

- Block-diagonal matrix

$$(B_h)_{k,k} = \left[\frac{1}{J_i} DF_i^t \mu^{-1} \, DF_i \right] (\xi_k^{GL}) \omega_k^{GL}$$

Elementary matrices

Mass matrix :

$$(M_h)_{i,j} \;=\; \int_{\hat{K}} J_i \, DF_i^{-1} \, \varepsilon \, DF_i^{*-1} \, \hat{\varphi}_i \cdot \hat{\varphi}_k \, d\hat{x}$$

Stiffness matrix :

$$(K_h)_{i,j} \;=\; \int_{\hat{K}} \frac{1}{J_i} \, DF_i^t \, \mu^{-1} \, DF_i \, \hat{\nabla} \times \hat{\varphi}_i \cdot \hat{\nabla} \times \hat{\varphi}_k \, d\hat{x}$$

- Use of Gauss-Lobatto quadrature (ω_k^{GL}, ξ_k^{GL})
- Block-diagonal matrix

$$(A_h)_{k,k} = \left[J_i \, DF_i^{-1} \, \varepsilon \, DF_i^{*-1} \right] (\xi_k^{GL}) \omega_k^{GL}$$

- Block-diagonal matrix

$$(B_h)_{k,k} = \left[\frac{1}{J_i} DF_i^t \mu^{-1} \, DF_i \right] (\xi_k^{GL}) \omega_k^{GL}$$

Fast matrix vector product

Let us introduce the two following matrices, independant of the geometry :

$$\hat{C}_{i,j} \; = \; \hat{\varphi}_i(\xi_j^{GL}) \qquad \hat{R}_{i,j} \; = \; \hat{\nabla} \times \hat{\varphi}_i^{GL}(\xi_j^{GL})$$

Then, we have : $\quad M_h \; = \; \hat{C}\, A_h \hat{C}^* \qquad K_h \; = \; \hat{C} \hat{R}\, B_h \hat{R}^* \hat{C}^*$

- Complexity of $\hat{C}\, U$: $6\,(r+1)^4$ operations in 3-D
- Complexity of $\hat{R}\, U$: $12\,(r+1)^4$ operations in 3-D
- Complexity of $A_h\, U + B_h\, U$: $30\,(r+1)^3$ operations

Complexity of standard matrix vector product $18 r^3\,(r+1)^3$

- Matrix-vector product 67% slower by using exact integration

Fast matrix vector product

Let us introduce the two following matrices, independant of the geometry :

$$\hat{C}_{i,j} = \hat{\varphi}_i(\xi_j^{GL}) \qquad \hat{R}_{i,j} = \hat{\nabla} \times \hat{\varphi}_i^{GL}(\xi_j^{GL})$$

Then, we have : $M_h = \hat{C}\, A_h \hat{C}^*$ $\quad K_h = \hat{C} \hat{R}\, B_h \hat{R}^* \hat{C}^*$

- Complexity of $\hat{C}\, U$: $6\,(r+1)^4$ operations in 3-D
- Complexity of $\hat{R}\, U$: $12\,(r+1)^4$ operations in 3-D
- Complexity of $A_h\, U + B_h\, U$: $30\,(r+1)^3$ operations

Complexity of standard matrix vector product $18 r^3\,(r+1)^3$

- Matrix-vector product 67% slower by using exact integration

Fast matrix vector product

Let us introduce the two following matrices, independant of the geometry :

$$\hat{C}_{i,j} = \hat{\varphi}_i(\xi_j^{GL}) \qquad \hat{R}_{i,j} = \hat{\nabla} \times \hat{\varphi}_i^{GL}(\xi_j^{GL})$$

Then, we have : $M_h = \hat{C}\, A_h \hat{C}^* \qquad K_h = \hat{C}\hat{R}\, B_h \hat{R}^* \hat{C}^*$

- Complexity of $\hat{C}\, U$: $6\,(r+1)^4$ operations in 3-D
- Complexity of $\hat{R}\, U$: $12\,(r+1)^4$ operations in 3-D
- Complexity of $A_h\, U + B_h\, U$: $30\,(r+1)^3$ operations

Complexity of standard matrix vector product $18 r^3\,(r+1)^3$

- Matrix-vector product 67% slower by using exact integration

Fast matrix vector product

Let us introduce the two following matrices, independant of the geometry :

$$\hat{C}_{i,j} \;=\; \hat{\varphi}_i(\xi_j^{GL}) \qquad \hat{R}_{i,j} \;=\; \hat{\nabla} \times \hat{\varphi}_i^{GL}(\xi_j^{GL})$$

Then, we have : $\quad M_h \;=\; \hat{C}\, A_h \hat{C}^* \qquad K_h \;=\; \hat{C}\hat{R}\, B_h \hat{R}^* \hat{C}^*$

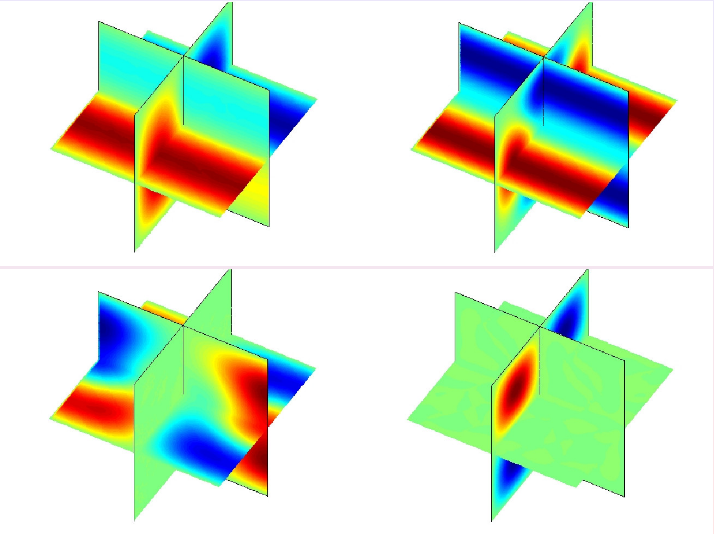
- Complexity of $\hat{C}\, U$: $6\,(r+1)^4$ operations in 3-D
- Complexity of $\hat{R}\, U$: $12\,(r+1)^4$ operations in 3-D
- Complexity of $A_h\, U + B_h\, U$: $30\,(r+1)^3$ operations

Complexity of standard matrix vector product $18 r^3\,(r+1)^3$

- Matrix-vector product 67% slower by using exact integration

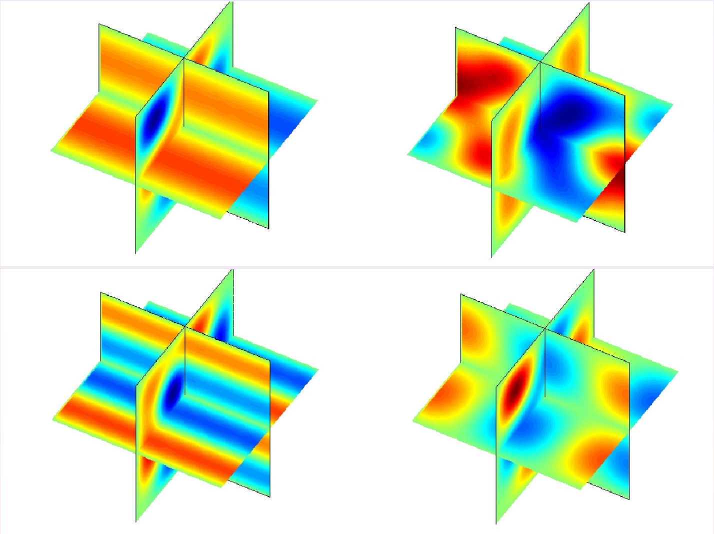
Spurious free method

Computation of eigenvalues in a cubic cavity, with tetrahedra split in hexahedra



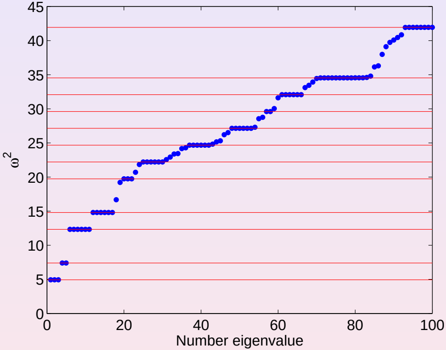
Spurious free method

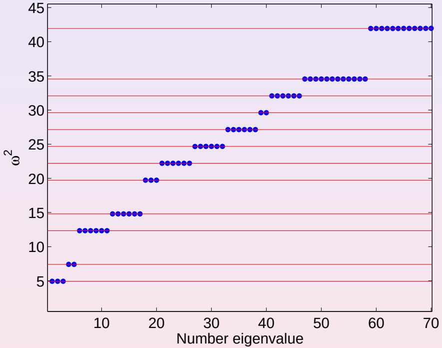
Computation of eigenvalues in a cubic cavity, with tetrahedra split in hexahedra



Spurious free method

Computation of eigenvalues in a cubic cavity, with tetrahedra split in hexahedra



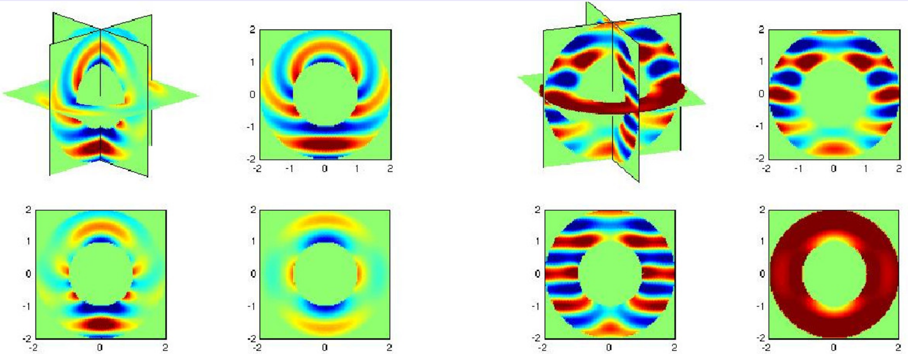


Numerical eigenvalues if we use Gauss-Lobatto points at right, or Gauss points for the stiffness matrix at left.

- Gauss-Lobatto integration leads to a spurious-free method

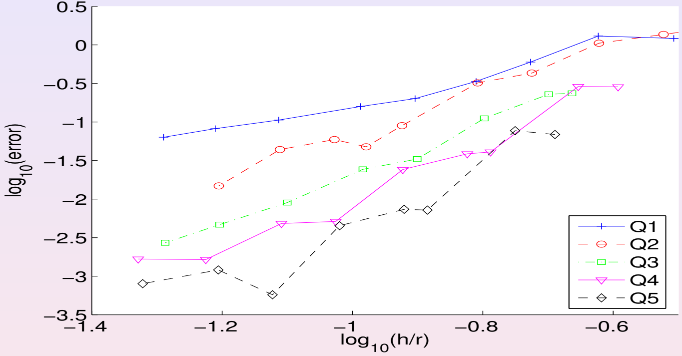
Convergence of the method

Scattering by a perfectly conductor sphere $E \times n = 0$



Convergence of the method

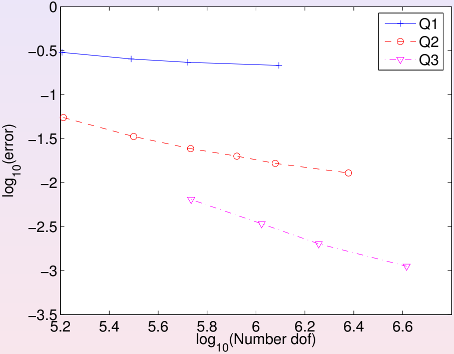
Convergence of Nedelec's first family on regular meshes



- Optimal convergence $O(h^r)$ in H(curl,Ω) norm

Convergence of the method

Convergence on tetrahedral meshes split in hexahedra



- Loss of one order, convergence $O(h^{r-1})$ in H(curl,Ω) norm

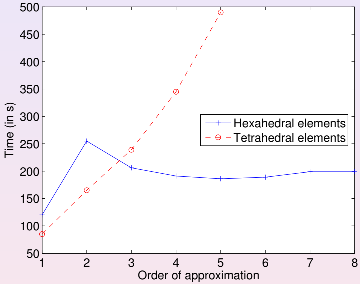
Is the matrix-vector product fast ?

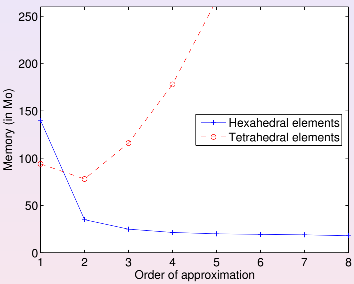
Comparison between standard formulation and discrete factorization

Order	1	2	3	4	5
Time, standard formulation	55s	127s	224s	380s	631
Time, discrete factorization	244s	128s	106s	97s	96s
Storage, standard formulation	18 Mo	50 Mo	105 Mo	187 Mo	308 Mo
Storage, discrete factorization	23 Mo	9.9 Mo	6.9 Mo	5.7 Mo	5.0 Mo

Is the matrix-vector product fast ?

Comparison between tetrahedral and hexahedral elements





At left, time computation for a thousand iterations of COCG
At right, storage for mesh and matrices

Preconditioning used

- Incomplete factorization with threshold on the damped Maxwell equation :

$$-k^2(\alpha + i\beta)\varepsilon E - \nabla \times (\frac{1}{\mu}\nabla \times E) = 0$$

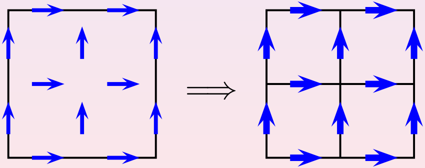
  - ILUT threshold ≥ 0.05 in order to have a low storage

Preconditioning used

- Incomplete factorization with threshold on the damped Maxwell equation :

$$-k^2(\alpha + i\beta)\varepsilon E - \nabla \times (\frac{1}{\mu}\nabla \times E) = 0$$

  - ILUT threshold ≥ 0.05 in order to have a low storage
  - Use of a Q_1 subdivided mesh to compute matrix



Preconditioning used

- Incomplete factorization with threshold on the damped Maxwell equation :

$$-k^2(\alpha + i\beta)\varepsilon E - \nabla \times (\frac{1}{\mu}\nabla \times E) = 0$$

- Multigrid iteration on the damped Maxwell equation
  - Use of the Hiptmair smoother
  - Low-storage algorithm even with high order

- Without damping, both preconditioners **does not lead** to convergence.

- A good choice of parameter is $\alpha = 1,\ \beta = 1$

Caracteristics of the incomplete factorization

Let us count the number of iterations and the memory used by the preconditioner, for different values of (α, β)

Threshold	$1e-4$	$1e-3$	0.01	0.05	0.1
$\alpha = 1$ $\beta = 0$	$30/370$ *Mo*	$\infty/350$ *Mo*	$\infty/340$ *Mo*	$\infty/326$ *Mo*	$\infty/314$ *Mo*
$\alpha = 1$ $\beta = 0.5$	$55/299$ *Mo*	$55/242$ *Mo*	$55/149$ *Mo*	$82/74$ *Mo*	$145/47$ *Mo*
$\alpha = 1$ $\beta = 1$	$97/244$ *Mo*	$97/197$ *Mo*	$99/108$ *Mo*	$110/53$ *Mo*	$155/34$ *Mo*

Caracteristics of the incomplete factorization

The use of a Q1 subdivided mesh is very accurate for the scalar Helmholtz equation, but has some difficulties for Maxwell equations. Let us count the number of iterations depending the frequency. The frequency 1 corresponds to the "normal" frequency.

Order	$F = 0.125$	$F = 0.25$	$F = 0.5$	$F = 1.0$	$F = 1.5$
Q_2(110 000ddl)	*NC*	49	19	16	49
Q_4(92 000ddl)	*NC*	*NC*	42	30	123
Q_6(72 000ddl)	*NC*	*NC*	71	47	159

Caracteristics of the incomplete factorization

The use of a Q1 subdivided mesh is very accurate for the scalar Helmholtz equation, but has some difficulties for Maxwell equations. Let us count the number of iterations depending the frequency. The frequency 1 corresponds to the “normal” frequency.

Order	$F = 0.125$	$F = 0.25$	$F = 0.5$	$F = 1.0$	$F = 1.5$
Q_2(110 000ddl)	*NC*	49	19	16	49
Q_4(92 000ddl)	*NC*	*NC*	42	30	123
Q_6(72 000ddl)	*NC*	*NC*	71	47	159

Problems in low-frequency case, because of the difference between the discrete kernels (of Q1 and high order).

Caracteristics of the multigrid iteration

The smoother of Hiptmair is based on the Helmholtz decomposition :

$$E = \nabla \varphi + u$$

Caracteristics of the multigrid iteration

The smoother of Hiptmair is based on the Helmholtz decomposition :

$$E = \nabla \varphi + u$$

The potential φ is solution of the "Laplacian" variationnal formulation :

$$\int_\Omega \nabla \varphi \nabla \psi - ik \int_\Sigma \nabla \varphi \times n \cdot \nabla \psi \times n = \int_\Omega f \cdot \nabla \psi$$

A_ϕ finite element matrix associated to this formulation

Caracteristics of the multigrid iteration

The smoother of Hiptmair is based on the Helmholtz decomposition :

$$E = \nabla\varphi + u$$

Let us introduce the operator P :

$$\begin{array}{lcl} P : H_0^1(\Omega) & \leftrightarrow & H(\text{curl}, \Omega) \\ \varphi & \leftrightarrow & \nabla\varphi \end{array}$$

then $A_\phi = P^* A_e P$ and the smoother can be written as :

- Relaxation on edge finite element operator $A_e x = b$
- Projection on nodal finite element $b_\phi = P^*(b - A_e x)$
- Relaxation on nodal finite element operator $A_\phi x_\phi = b_\phi$
- Projection on edge finite element $x = x + P x_\phi$

Caracteristics of the multigrid iteration

The smoother of Hiptmair is based on the Helmholtz decomposition :

$$E = \nabla \varphi + u$$

- Jacobi relaxation used, because it avoids us storing the matrices (compared to a SSOR relaxation).
- Prolongation operator is an interpolation from the coarse order to the fine order. A matrix-free implementation of the prolongation operator is used.
- Incomplete factorization for the resolution of the coarsest order
- Use of a W-cycle, and one step of pre and post-smoothing (in order to get the symmetry of the preconditioning)

Caracteristics of the multigrid iteration

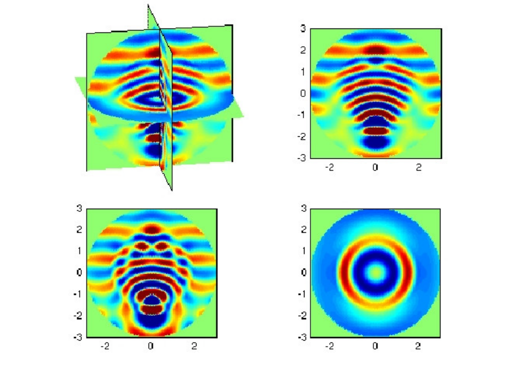
The smoother of Hiptmair is based on the Helmholtz decomposition :

$$E = \nabla \varphi + u$$

The use of a multigrid iteration on the Q1 subdivided mesh is not optimal

- Fail of a good preconditioning in low-frequency case, because the "Q1" discrete kernel is different from the high order discrete kernel
- Overcost in storage, because we store all the needed Q1 matrices

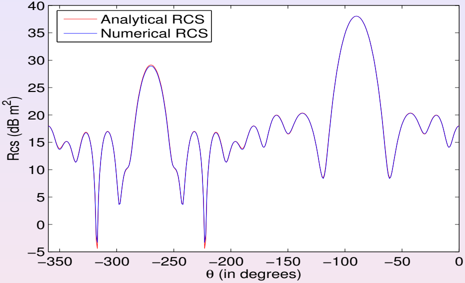
Scattering by a dielectric sphere



- Sphere of radius 2 with $\varepsilon = 3.5$ $\mu = 1$
- Outside boundary on a sphere of radius 3.

Scattering by a dielectric sphere

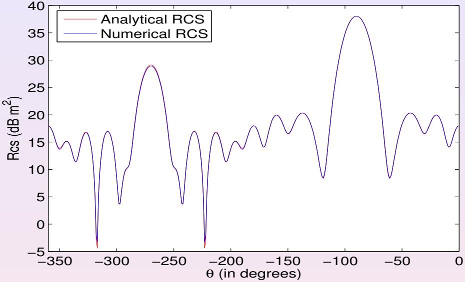
How many dofs/time to reach an error less than 0.5 dB



Finite Element	Q_2	Q_4	Q_6
Nb dofs	940 000	88 000	230 000
No preconditioning	19 486 s	894 s	4 401 s
ILUT(0.05)	-	189 s	1 035 s
Two-grid	5 814 s	280 s	1 379 s
Multi-grid	5 814 s	499 s	2 515 s
Q1 Two-grid	4 4344 s	488 s	1 095 s

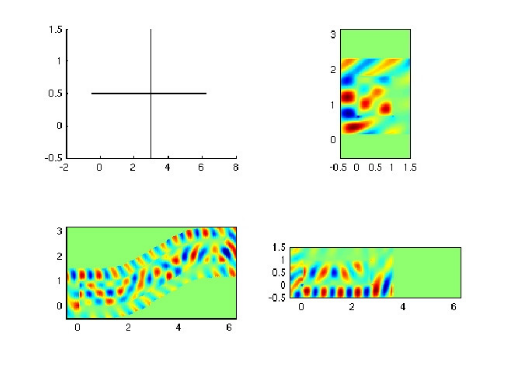
Scattering by a dielectric sphere

How many dofs/time to reach an error less than 0.5 dB



Finite Element	$\mathbf{Q_2}$	$\mathbf{Q_4}$	$\mathbf{Q_6}$
No preconditioning	171 Mo	10 Mo	24 Mo
ILUT(0.05)	-	99 Mo	271 Mo
Two-grid	402 Mo	34 Mo	132 Mo
Multi-grid	402 Mo	12 Mo	28 Mo
Q1 Two-grid	947 Mo	67 Mo	180 Mo

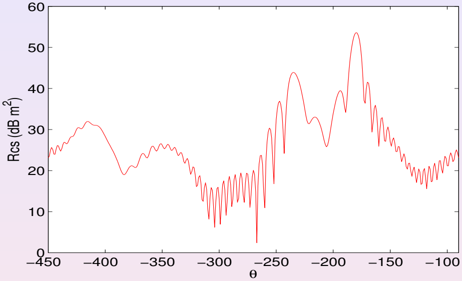
Scattering by a cobra cavity



- Cobra cavity of length 10, and depth 2
- Outside boundary at a distance of 1

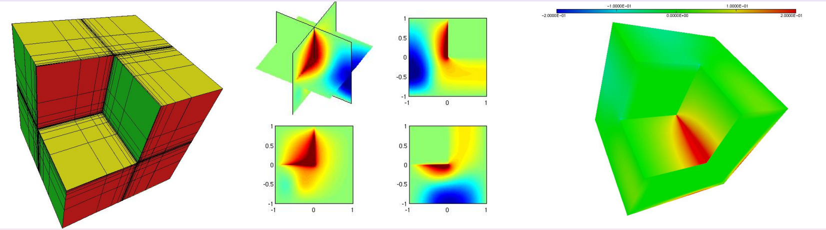
Scattering by a cobra cavity

How many dofs/time to reach an error less than 0.5 dB

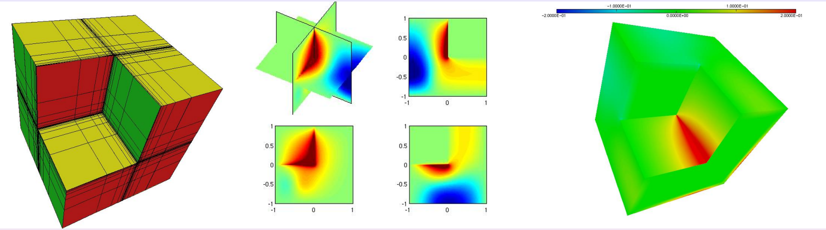


Finite Element	Q_4	Q_6
Nb dofs	412 000	187 000
No preconditioning	14 039 s (47 Mo)	12 096 s (22 Mo)
ILUT(0.05)	2 247 s (391 Mo)	846 s (161 Mo)
Two-grid	2 355 s (91 Mo)	2 319 s (65 Mo)
Multigrid	4 519 s (59 Mo)	-
Q1 Two-grid	9 294 s (260 Mo)	10 500 s (130 Mo)

Local refinement for the Fichera corner

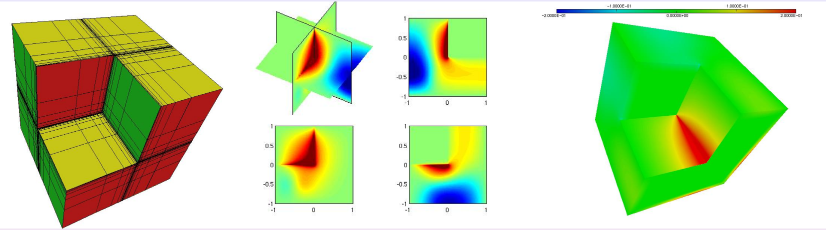


Local refinement for the Fichera corner



- Q4 approximation on a local refined mesh
- Incomplete factorization fails on this case
- Multigrid preconditioning needs SSOR smoother to be efficient
- 480 000 dofs and cells 256 times smaller than other ones

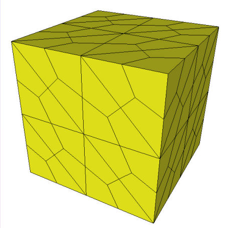
Local refinement for the Fichera corner



Algorithm	Iterations	Time	Memory
No preconditioning	$> 1\ 000\ 000$	∞	33 Mo
Multigrid with Jacobi smoother	30 560	35h	63 Mo
Multigrid with SSOR smoother	579	1h	790 Mo

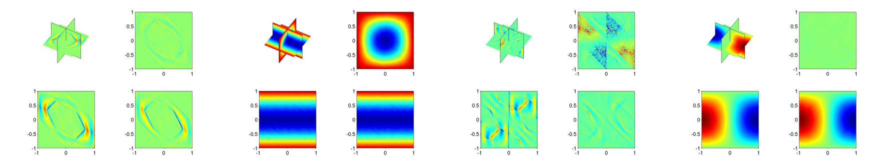
Eigenmodes with the second family

The second family uses Q_r^3 instead of $Q_{r-1,r,r} \times Q_{r,r-1,r} \times Q_{r,r,r-1}$
Mesh used for the simulations ($\mathbf{Q_3}$)



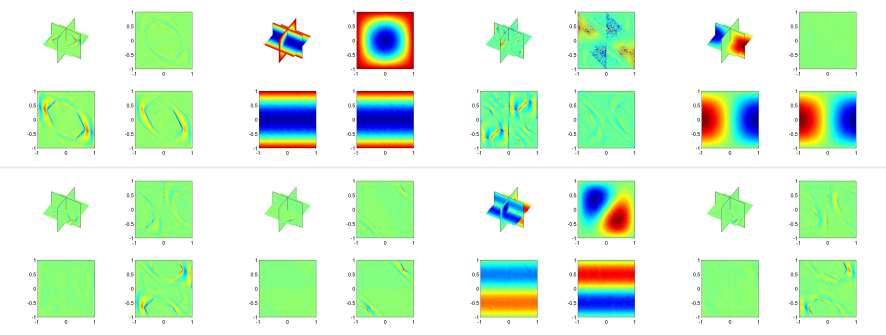
Eigenmodes with the second family

The second family uses Q_r^3 instead of $Q_{r-1,r,r} \times Q_{r,r-1,r} \times Q_{r,r,r-1}$



Eigenmodes with the second family

The second family uses Q_r^3 instead of $Q_{r-1,r,r} \times Q_{r,r-1,r} \times Q_{r,r,r-1}$



Two types of penalization

Mixed formulation of Maxwell equations

$$
\begin{aligned}
-\omega \int_\Omega E \cdot \varphi + \int_\Omega H \cdot \mathrm{rot}(\varphi) - i\alpha \sum_{e \ \text{face}} \int_{\Gamma_e} [E \cdot n][\varphi \cdot n] &= \int_\Omega f \cdot \varphi \\
-\omega \int_\Omega H \cdot \varphi + \int_\Omega \mathrm{rot}(E) \cdot \varphi - i\delta \sum_{e \ \text{face}} \int_{\Gamma_e} [H \times n] \cdot [\varphi \times n] &= 0
\end{aligned}
$$

Approximation space for H

$$
W_h = \{ \vec{u} \in L^2(\Omega) \text{ so that } DF_i^* \, \vec{u} \circ F_i \in (Q_r)^3 \}
$$

- Equivalence with second-order formulation ($\alpha = \delta = 0$)
- Dissipative terms of penalization
- Penalization in α does not need of a mixed formulation

Two types of penalization

Mixed formulation of Maxwell equations

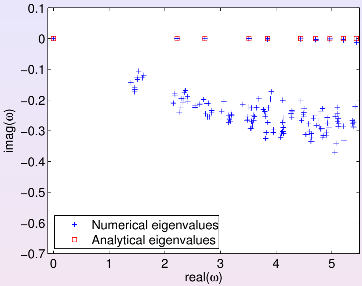
$$
\begin{aligned}
-\omega \int_\Omega E \cdot \varphi + \int_\Omega H \cdot \mathrm{rot}(\varphi) - i\alpha \sum_e \sum_{\text{face}} \int_{\Gamma_e} [E \cdot n][\varphi \cdot n] &= \int_\Omega f \cdot \varphi \\
-\omega \int_\Omega H \cdot \varphi + \int_\Omega \mathrm{rot}(E) \cdot \varphi - i\delta \sum_e \sum_{\text{face}} \int_{\Gamma_e} [H \times n] \cdot [\varphi \times n] &= 0
\end{aligned}
$$

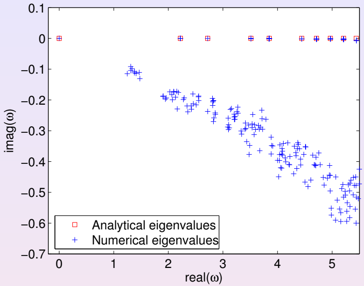
Approximation space for H

$$
W_h = \{ \vec{u} \in L^2(\Omega) \text{ so that } DF_i^* \, \vec{u} \circ F_i \in (Q_r)^3 \}
$$

- Equivalence with second-order formulation ($\alpha = \delta = 0$)
- Dissipative terms of penalization
- Penalization in α does not need of a mixed formulation

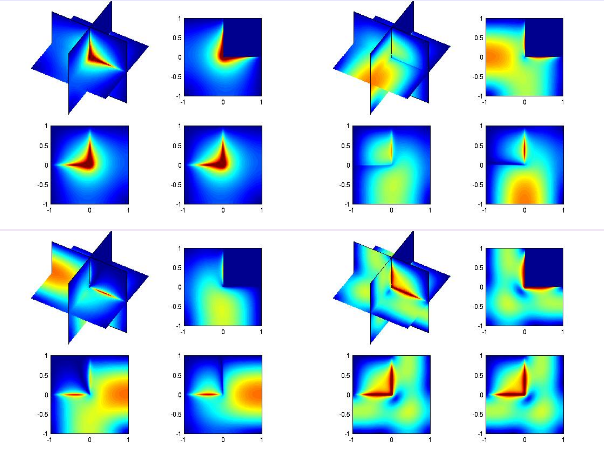
Effects of penalization





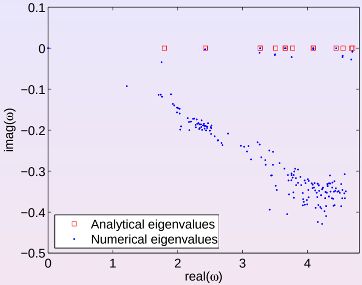
- Case of the cubic cavity meshed with split tetrahedrals
- At left $\alpha = 0.1$, at right $\alpha = 0.5$

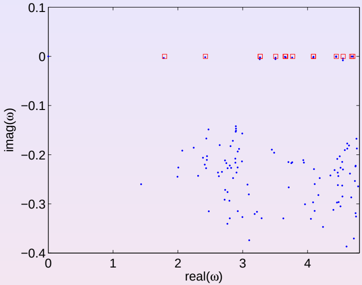
Effects of penalization



Four modes of the Fichera corner

Effects of penalization





- Case of the Fichera corner
- At left $\alpha = 0.5$, at right $\delta = 0.5$

- Both penalizations efficient for regular domains
- Delta-penalization more robust for singular domains

Why choosing first family compared to second family or DG method ?

- All the methods are spectrally correct
- All the methods have a fast MV product
- DG and second family need more dof
- Helmholtz decomposition more natural for the first family
- Because of spurious modes, DG and second family need specific preconditioning

Why choosing first family compared to second family or DG method ?

- All the methods are spectrally correct
- All the methods have a fast MV product

- DG and second family need more dof
- Helmholtz decomposition more natural for the first family
- Because of spurious modes, DG and second family need specific preconditioning