Efficient high-order finite elements for Helmholtz equation and time-harmonic elastodynamics on hybrid meshes

M. Duruflé

IMB, Bacchus

13th December 2010
Bibliography

- S. Fauqueux, mixed spectral elements for wave and elastic equations (hexahedra)

- S. Pernet, Discontinuous Galerkin methods for Maxwell’s equations (hexahedra)

- G.E. Karniadakis, S. Sherwin, T. Warburton, continuous and discontinuous finite elements on tetrahedra/prisms/pyramids by considering “degenerated” cube

- Bedrosian, Early work on pyramids, nodal basis functions for order 1 and 2

- Nigam, Philips, Recent work on finite element spaces for pyramids, infinite pyramid is the reference element
Motivation of Morgane Bergot’s thesis

- Automatic generation of high-quality hexahedral meshes is difficult
- “Solution of split tetrahedra” is not interesting
- Some mesh tools are able to produce meshes with a high ratio of hexahedra and some remaining pyramids/tets/prisms.
- Pyramids elements not as well known as other elements.
Model equation

\[-\rho \omega^2 u - \text{Div}(\mu \nabla u) = f \in \Omega\]
Model equation

\[-\rho \omega^2 u - \text{Div}(\mu \nabla u) = f \in \Omega\]

Use of finite element method leads to the following linear system:

\[
( -\omega^2 D_h + K_h ) U_h = F_h
\]
Model equation

\[-\rho \omega^2 u - \text{Div}(\mu \nabla u) = f \in \Omega\]

Use of finite element method leads to the following linear system:

\[
(-\omega^2 D_h + K_h) U_h = F_h
\]

Our aim is to develop an efficient iterative solver for an high order of approximation \( r \). Therefore, we need a fast matrix-vector product \((-\omega^2 D_h + K_h) U_h\)
Model equation

\[-\rho \omega^2 u - \text{Div}(\mu \nabla u) = f \quad \in \Omega\]

Use of finite element method leads to the following linear system:

\[(-\omega^2 D_h + K_h) U_h = F_h\]

Our aim is to develop an efficient iterative solver for an high order of approximation \( r \). Therefore, we need a fast matrix-vector product

\[(-\omega^2 D_h + K_h) U_h\]

\(u\) scalar \(\Rightarrow\) Helmholtz equation

\(u\) vectorial \(\Rightarrow\) time-harmonic elastodynamics
Finite element on pyramids

Simplest expression of \( F_i \) (Bedrosian):

\[
F_i(\hat{x}, \hat{y}, \hat{z}) = A + B\hat{x} + C\hat{y} + D\hat{z} + \frac{\hat{x}\hat{y}}{4(1 - \hat{z})} (S_1 + S_3 - S_2 - S_4)
\]
Finite element on pyramids

- Use of rational fractions to define $F_i$
  - Early work of **Bedrosian** with explicit first and second order basis functions
  - Work of **Sherwin, Karniadakis, Warburton** : h-p Basis functions obtained by considering a degenerated cube (coincidence with Bedrosian functions for $r = 1$)
  - Recent work of **Nigam, Phillips** with a reference infinite pyramid (same basis functions as Bedrosian for $r = 1$)

- Use of piecewise polynomial to define $F_i$ (polynomial on each sub-tetrahedron)
  - Work of **Wieners**, with first and second order basis functions
  - Work of **Knabner and Summ**, with an analysis of this transformation
  - Work of **Bluck and Walker**, with a proposition of high order basis functions
Condition of optimality

We define the finite element space with real element $K_i$:

$$V_h = \{u \in H^1(\Omega) \text{ such that } u|_{K_i} \in V^r_F\}$$

$V^r_F$ : finite element space for the real element

We define the finite element space with reference element $\hat{K}$:

$$V_h = \{u \in H^1(\Omega) \text{ such that } u|_{K_i \circ F_i} \in \hat{V}^r\}$$

$\hat{V}^r$ : finite element space for the reference element

Condition of optimality :

$$V^r_F \supset P_r$$

For hexahedra, we can prove :

$$V^r_F \supset P_r \iff \hat{V}^r \supset Q_r$$
Optimal finite element space

Same approach than for hexahedra: We consider a monomial of $P_r$:

$$x^m, \quad m \leq r$$

$$(a + b\hat{x} + c\hat{y} + d\hat{z} + \alpha\left(\frac{\hat{x}\hat{y}}{1 - \hat{z}}\right))^m$$

$$\sum_k C_m^k (a + b\hat{x} + c\hat{y})^k (d\hat{z})^k \alpha^{m-k} \left(\frac{\hat{x}\hat{y}}{1 - \hat{z}}\right)^{m-k}$$

After some calculations, you can show that the optimal finite element space is

$$\hat{V}^r = P_r \oplus \sum_{k=0}^{r-1} \left(\frac{\hat{x}\hat{y}}{1 - \hat{z}}\right)^{r-k} P_k(\hat{x}, \hat{y})$$
Numerical comparison between different methods

We perform a dispersion analysis on the following hybrid mesh:
Numerical comparison between different methods

Optimal $r=2$
Optimal $r=3$
Sherwin $r=2$
Sherwin $r=3$
Nigam $r=2$
Nigam $r=3$
Walker $r=2$

Optimal finite element space constructed in Morgane Bergot's thesis for edge elements, different from Nigam/Phillips and Demkowicz/Zaglmayr.
Numerical comparison between different methods

- We obtained same finite element space as Demkowicz/Zaglmayr
- We obtained a smaller finite element space than Nigam/Phillips
- We proposed modifications of basis functions of Sherwin/Karniadakis/Warburton so that they span the optimal finite element space
- Alternative approach using piecewise polynomial (by splitting pyramid in two or four tets) is not consistent for non-affine pyramids and order greater than 2
Numerical comparison between different methods

- We obtained same finite element space as Demkowicz/Zaglmayr
- We obtained a smaller finite element space than Nigam/Phillips
- We proposed modifications of basis functions of Sherwin/Karniadakis/Warburton so that they span the optimal finite element space
- Alternative approach using piecewise polynomial (by splitting pyramid in two or four tets) is not consistent for non-affine pyramids and order greater than 2
- Optimal finite element space constructed in Morgane Bergot’s thesis for edge elements, different from Nigam/Phillips and Demkowicz/Zaglmayr
Nodal Basis functions

Orthogonal basis of pyramidal finite element space

\[ \psi_{i,j,k} = P_{i}^{0,0} \left( \frac{\hat{x}}{1 - \hat{z}} \right) P_{j}^{0,0} \left( \frac{\hat{y}}{1 - \hat{z}} \right) P_{k}^{2 \max(i,j)+2,0} (2\hat{z} - 1)(1 - z)^{\max(i,j)} \]

where \( P_{i}^{\alpha,\beta} \) are Jacobi polynomials orthogonal with respect to \((1 - x)^{\alpha}(1 + x)^{\beta}\).
Nodal Basis functions

Orthogonal basis of pyramidal finite element space

\[ \psi_{i,j,k} = P_{i}^{0,0}(\frac{\hat{x}}{1-\hat{z}})P_{j}^{0,0}(\frac{\hat{y}}{1-\hat{z}})P_{k}^{2\max(i,j)+2,0}(2\hat{z} - 1)(1 - z)^{\max(i,j)} \]

where \( P_{i}^{\alpha,\beta} \) are Jacobi polynomials orthogonal with respect to \((1 - x)^{\alpha}(1 + x)^{\beta}\)

\( M_{i} \): interpolation points on the reference pyramid

Vandermonde matrix:
\[ VDM_{i,j} = \psi_{i}(M_{j}) \]

Nodal basis functions:
\[ \varphi_{i} = \sum_{j} (VDM^{-1})_{i,j} \psi_{j} \]
Nodal Basis functions

M. Duruflé (IMB, Bacchus)  Efficient high-order finite elements for Helmholtz equation and time-harmonic elastodynamics on hybrid meshes  13th December 2010 10 / 21
Hierarchical Basis functions

- Same basis functions as Sherwin, Karniadakis, Warburton for hexahedra, prisms, tetrahedra, but different ones for pyramids
Hierarchical Basis functions

- Same basis functions as Sherwin, Karniadakis, Warburton for hexahedra, prisms, tetrahedra, but different ones for pyramids

- **Vertex:**
  \[
  N_1 = \frac{(1 - \hat{x} - \hat{z})(1 - \hat{y} - \hat{z})}{4 (1 - \hat{z})}
  \]

- **Apex:** \( \hat{z} \)

- **Horizontal edge:**
  \[
  N_1 \frac{(1 + \hat{x} - \hat{z})}{2} (1 - \hat{z})^{i-1} P_{i-1}^{1,1}(\frac{\hat{x}}{1 - \hat{z}})
  \]

- **Vertical edge:**
  \[
  N_1 \hat{z} P_{i-1}^{1,1}(\hat{z} + \frac{\hat{x} + \hat{y}}{2})
  \]

- **Triangular face:**
  \[
  N_1 \frac{(1 + \hat{x} - \hat{z})}{2} \hat{z} (1 - \hat{z})^{i-1} P_{i-1}^{1,1}(\frac{\hat{x}}{1 - \hat{z}}) P_{j-1}^{2i+1,1}(2\hat{z} - 1)
  \]
Hierarchical Basis functions

(Differences with Sherwin, Karniadakis, Warburton denoted in red)

• Base :

\[ N_1 N_3 (1 - \hat{z})^\text{max}(i,j)-1 P_{i-1}^{1,1}(\frac{\hat{x}}{1-\hat{z}}) P_{j-1}^{1,1}(\frac{\hat{y}}{1-\hat{z}}) \]

• Interior :

\[ N_1 N_3 \hat{z} (1 - \hat{z})^\text{max}(i,j)-1 P_{i-1}^{1,1}(\frac{\hat{x}}{1-\hat{z}}) P_{j-1}^{1,1}(\frac{\hat{y}}{1-\hat{z}}) P_{k-1}^{2\text{max}(i,j)+2,1}(2\hat{z} - 1) \]
Fast matrix-vector product

Semi-tensorization of basis functions $\Rightarrow$ fast matrix-vector product

$$\varphi_j = \varphi_{j_1}(\hat{x}) \varphi_{j_2}(\hat{y}) \varphi_{j_3}(\hat{z})$$
Fast matrix-vector product

\[(D_h)_{i,j} = \int_K \rho J_i \hat{\varphi}_i \hat{\varphi}_j \, d\hat{x}\]

Use of quadrature formulas \((\omega_m, \xi_m)\) on the reference element
Fast matrix-vector product

\[(D_h)_{i,j} = \int_{\hat{K}} \rho J_i \hat{\varphi}_i \hat{\varphi}_j d\hat{x}\]

Use of quadrature formulas \((\omega_m, \xi_m)\) on the reference element

\[(D_h)_{i,j} = \sum_m \omega_m \rho J_i \hat{\varphi}_i(\xi_m) \hat{\varphi}_j(\xi_m)\]

Matrix-vector product \(D_h U\) can be split into three steps:

\[v_m = \sum_j \hat{\varphi}_j(\xi_m) u_j\]

\[w_m = \omega_m \rho J_i(\xi_m) v_m\]

\[y_i = \sum_m \hat{\varphi}_i(\xi_m) w_m\]
Fast matrix-vector product

\[(D_h)_{i,j} = \int_{\hat{K}} \rho J_i \hat{\phi}_i \hat{\phi}_j \, d\hat{x}\]

Use of quadrature formulas \((\omega_m, \xi_m)\) on the reference element

\[(D_h)_{i,j} = \sum_m \omega_m \rho J_i \hat{\phi}_i(\xi_m) \hat{\phi}_j(\xi_m)\]

Underlying factorization

\[\hat{C}_{i,j} = \hat{\phi}_i(\xi_j)\]

\[(A_h)_m = \omega_m \rho J_i(\xi_m)\]

\[D_h = \hat{C} A_h \hat{C}^*\]

⇒ only storage of \(\omega_m \rho J_i(\xi_m)\)
Fast matrix-vector product

\[(D_h)_{i,j} = \int_K \rho J_i \hat{\phi}_i \hat{\phi}_j \, d\hat{x}\]

Use of quadrature formulas \((\omega_m, \xi_m)\) on the reference element

Product \(Y = \hat{C}U\) is split into three steps:

\[v_{j_1,j_2,j_3} = \sum_{j_3} \hat{\phi}_{j_3}^{j_1,j_2}(\xi_{i_3}) u_{j_1,j_2,j_3}\]

\[w_{j_1,i_2,i_3} = \sum_{j_2} \hat{\phi}_{j_2}^{j_1}(\xi_{i_2}) v_{j_1,j_2,i_3}\]

\[y_{i_1,i_2,i_3} = \sum_{j_1} \hat{\phi}_{j_1}(\xi_{i_1}) w_{j_1,i_2,i_3}\]
Stiffness terms

\[(K_h)_{i,j} = \int_{\mathcal{K}} J_i \, D F_i^{-1} \mu D F_i^{*,-1}(\xi_m) \, \nabla \hat{\phi}_j \cdot \nabla \hat{\phi}_i \, d\hat{x}\]
Stiffness terms

\[(K_h)_{i,j} = \int_{\hat{K}} J_i \, DF_i^{-1} \mu DF_i^{*-1}(\xi_m) \nabla \hat{\phi}_j \cdot \nabla \hat{\phi}_i \, d\hat{x}\]

Matrix-vector product $K_h U$ can be split into three steps

\[v_m = \sum_j \nabla \hat{\phi}_j(\xi_m) u_j\]

\[w_m = \omega_m J_i \, DF_i^{-1} \mu DF_i^{*-1} v_m\]

\[y_i = \sum_q \nabla \hat{\phi}_i(\xi_m) w_m\]
Stiffness terms

\[(K_h)_{i,j} = \int_{\hat{K}} J_i \, DF_i^{-1} \mu DF_i^{*-1} (\xi_m) \nabla \hat{\phi}_j \cdot \nabla \hat{\phi}_i \, d\hat{x}\]

Underlying factorization

\[\hat{S}_{i,j} = \nabla \hat{\phi}_i(\xi_j)\]

\[(B_h)_m = \omega_m J_i \, DF_i^{-1} \mu DF_i^{*-1}\]

\[K_h = \hat{S} B_h \hat{S}^*\]

⇒ only storage of \(J_i \, DF_i^{-1} \mu DF_i^{*-1}\) for Helmholtz equation, and only \(J_i\) and \(DF_i^{-1}\) for elastodynamics
By using the matrices

\[ \hat{C}_{i,j} = \hat{\phi}_i(\xi_j) \]
\[ \hat{S}_{i,j} = \hat{\nabla} \hat{\phi}_i(\xi_j) \]
\[ \hat{R}_{i,j} = \hat{\nabla} \hat{\psi}_i(\xi_j) \]

where \( \psi \) are basis functions associated with quadrature points, we have \( \hat{S} = \hat{R} \hat{C} \)

final matrix : \( \hat{C}(\omega^2 A_h + \hat{RB}_h \hat{R}^*) \hat{C}^* \)
Comparison Nodal/Hp

Computational time for 100 iterations of COCG on a mesh containing one million dofs. Pyramids, Helmholtz equation

<table>
<thead>
<tr>
<th>Order</th>
<th>r = 2</th>
<th>r = 4</th>
<th>r = 6</th>
<th>r = 8</th>
<th>r = 10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nodal</td>
<td>327s</td>
<td>499s</td>
<td>1021s</td>
<td>1918s</td>
<td>4345s</td>
</tr>
<tr>
<td>Hierarchic</td>
<td>285.6s</td>
<td>183s</td>
<td>183.7s</td>
<td>194s</td>
<td>238s</td>
</tr>
<tr>
<td>Stored matrix</td>
<td>26s</td>
<td>55s</td>
<td>113s</td>
<td>234s</td>
<td>359s</td>
</tr>
<tr>
<td></td>
<td>0.27 Go</td>
<td>0.78 Go</td>
<td>1.68 Go</td>
<td>3.09 Go</td>
<td>5.13 Go</td>
</tr>
</tbody>
</table>

Hexahedra, Helmholtz equation

<table>
<thead>
<tr>
<th>Order</th>
<th>r = 2</th>
<th>r = 4</th>
<th>r = 6</th>
<th>r = 8</th>
<th>r = 10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nodal</td>
<td>77s</td>
<td>49s</td>
<td>45s</td>
<td>42s</td>
<td>46s</td>
</tr>
<tr>
<td>Hierarchic</td>
<td>99s</td>
<td>64s</td>
<td>62s</td>
<td>77s</td>
<td>68s</td>
</tr>
<tr>
<td>Stored matrix</td>
<td>22s</td>
<td>45s</td>
<td>79s</td>
<td>120s</td>
<td>171s</td>
</tr>
<tr>
<td></td>
<td>0.27 Go</td>
<td>0.64 Go</td>
<td>1.170 Go</td>
<td>1.85 Go</td>
<td>2.72 Go</td>
</tr>
</tbody>
</table>
Comparison Nodal/Hp

Computational time for 100 iterations of COCG on a mesh containing one million dofs. Pyramids, Helmholtz equation

<table>
<thead>
<tr>
<th>Order</th>
<th>Nodal</th>
<th>Hierarchic</th>
<th>Stored matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>r = 2</td>
<td>327s</td>
<td>285.6s</td>
<td>26s</td>
</tr>
<tr>
<td>r = 4</td>
<td>499s</td>
<td>183s</td>
<td>55s</td>
</tr>
<tr>
<td>r = 6</td>
<td>1021s</td>
<td>183.7s</td>
<td>113s</td>
</tr>
<tr>
<td>r = 8</td>
<td>1918s</td>
<td>194s</td>
<td>234s</td>
</tr>
<tr>
<td>r = 10</td>
<td>4345s</td>
<td>238s</td>
<td>359s</td>
</tr>
<tr>
<td></td>
<td>0.27 Go</td>
<td>0.78 Go</td>
<td>1.68 Go</td>
</tr>
</tbody>
</table>

Hexahedra, Helmholtz equation

<table>
<thead>
<tr>
<th>Order</th>
<th>Nodal</th>
<th>Hierarchic</th>
<th>Stored matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>r = 2</td>
<td>77s</td>
<td>99s</td>
<td>22s</td>
</tr>
<tr>
<td>r = 4</td>
<td>49s</td>
<td>64s</td>
<td>45s</td>
</tr>
<tr>
<td>r = 6</td>
<td>45s</td>
<td>62s</td>
<td>79s</td>
</tr>
<tr>
<td>r = 8</td>
<td>42s</td>
<td>77s</td>
<td>120s</td>
</tr>
<tr>
<td>r = 10</td>
<td>46s</td>
<td>68s</td>
<td>171s</td>
</tr>
<tr>
<td></td>
<td>0.27 Go</td>
<td>0.64 Go</td>
<td>1.170 Go</td>
</tr>
</tbody>
</table>
### Pyramids, Elastodynamics

<table>
<thead>
<tr>
<th>Order</th>
<th>r = 2</th>
<th>r = 4</th>
<th>r = 6</th>
<th>r = 8</th>
<th>r = 10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nodal</td>
<td>675s</td>
<td>630s</td>
<td>999s</td>
<td>1 553s</td>
<td>3 418s</td>
</tr>
<tr>
<td>Hierarchic</td>
<td>723s</td>
<td>468s</td>
<td>482s</td>
<td>517s</td>
<td>670 s</td>
</tr>
<tr>
<td>Stored matrix</td>
<td>205s</td>
<td>498s</td>
<td>1 935s</td>
<td>4 163s</td>
<td>5 351s</td>
</tr>
<tr>
<td></td>
<td>2.56 Go</td>
<td>7.36 Go</td>
<td>16.5 Go</td>
<td>30.3 Go</td>
<td>50.8 Go</td>
</tr>
</tbody>
</table>

### Hexahedra, Elastodynamics

<table>
<thead>
<tr>
<th>Order</th>
<th>r = 2</th>
<th>r = 4</th>
<th>r = 6</th>
<th>r = 8</th>
<th>r = 10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nodal</td>
<td>197s</td>
<td>120s</td>
<td>114s</td>
<td>107s</td>
<td>123s</td>
</tr>
<tr>
<td>Hierarchic</td>
<td>259s</td>
<td>179s</td>
<td>165s</td>
<td>178s</td>
<td>184s</td>
</tr>
<tr>
<td>Stored matrix</td>
<td>216s</td>
<td>410s</td>
<td>814s</td>
<td>3 029s</td>
<td>3 105s</td>
</tr>
<tr>
<td></td>
<td>2.52 Go</td>
<td>5.69 Go</td>
<td>11.4 Go</td>
<td>18.3 Go</td>
<td>24.3 Go</td>
</tr>
</tbody>
</table>
### Comparison Nodal/Hp, Elastodynamics

**Pyramids, Elastodynamics**

<table>
<thead>
<tr>
<th>Order</th>
<th>r = 2</th>
<th>r = 4</th>
<th>r = 6</th>
<th>r = 8</th>
<th>r = 10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nodal</td>
<td>675s</td>
<td>630s</td>
<td>999s</td>
<td>1 553s</td>
<td>3 418s</td>
</tr>
<tr>
<td></td>
<td>723s</td>
<td>468s</td>
<td>482s</td>
<td>517s</td>
<td>670 s</td>
</tr>
<tr>
<td>Hierarchic</td>
<td>205s</td>
<td>498s</td>
<td>1 935s</td>
<td>4 163s</td>
<td>5 351s</td>
</tr>
<tr>
<td>Stored matrix</td>
<td>2.56 Go</td>
<td>7.36 Go</td>
<td>16.5 Go</td>
<td>30.3 Go</td>
<td>50.8 Go</td>
</tr>
</tbody>
</table>

**Hexahedra, Elastodynamics**

<table>
<thead>
<tr>
<th>Order</th>
<th>r = 2</th>
<th>r = 4</th>
<th>r = 6</th>
<th>r = 8</th>
<th>r = 10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nodal</td>
<td>197s</td>
<td>120s</td>
<td>114s</td>
<td>107s</td>
<td>123s</td>
</tr>
<tr>
<td></td>
<td>259s</td>
<td>179s</td>
<td>165s</td>
<td>178s</td>
<td>184s</td>
</tr>
<tr>
<td>Hierarchic</td>
<td>216s</td>
<td>410s</td>
<td>814s</td>
<td>3 029s</td>
<td>3 105s</td>
</tr>
<tr>
<td>Stored matrix</td>
<td>2.52 Go</td>
<td>5.69 Go</td>
<td>11.4 Go</td>
<td>18.3 Go</td>
<td>24.3 Go</td>
</tr>
</tbody>
</table>
Comparison Nodal/Hp, Condition number

[Graph showing the condition number of the mass matrix for different elements: Nodal Tetrahedron, Hierarchic Tetrahedron, Nodal Pyramid, Hierarchic Pyramid, Nodal Prism, Hierarchic Prism, Nodal Hexahedron, Hierarchic Hexahedron. The graph plots log₁₀(condition number of mass matrix) against r.]
Preconditioning techniques

- p-multigrid iteration on damped equation
  
  \[-\omega^2(\alpha + i\beta)u - \text{Div}(\mu \nabla u) = 0\]

- Jacobi smoother for hexahedral meshes
- Gauss-Seidel smoother for hybrid meshes

- subdomain-preconditioning (additive Schwarz-like):

  \[M = \sum P_i A_i^{-1} P_i\]

  where \(A_i\) is the finite element matrix on subdomain \(\Omega_i\) with absorbing boundary conditions
  
  one processor = one domain
Scattering of an airplane
Scattering of an airplane

Hybrid mesh used:
Scattering of an airplane
### Statistics for airplane

#### Without preconditioning:

<table>
<thead>
<tr>
<th>Mesh</th>
<th>Hybrid Dofs</th>
<th>Split tetrahedra Dofs</th>
<th>Tetrahedra Dofs</th>
<th>$L^2$ error</th>
<th>Iterations</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>6.08 millions</td>
<td>13.2 millions</td>
<td>5.39 millions</td>
<td>1.05 %</td>
<td>13 113</td>
<td>24 253s</td>
</tr>
<tr>
<td>$L^2$ error</td>
<td>1.05 %</td>
<td>0.89 %</td>
<td>1.14 %</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Iterations</td>
<td>13 113</td>
<td>94 500</td>
<td>24 325</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time</td>
<td>24 253s</td>
<td>981 139s</td>
<td>80 274s</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Multigrid preconditioning:

| Iterations | 193 | 781 | 268 |
| Time       | 2 870s | 68 354s | 9 177s |

#### Subdomain preconditioning (128 domains):

| Iterations | 545 | 579 | 481 |
| Time       | 26 121s | 39 500s | 10 684s |
Two-layer problem

Eqn: \(-1 \times 0\)
Two-layer problem
Statistics of two-layer problem

Without preconditioning:

<table>
<thead>
<tr>
<th></th>
<th>Hexahedra</th>
<th>Hybrid</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dofs</td>
<td>274 625</td>
<td>189 669</td>
</tr>
<tr>
<td>Iterations</td>
<td>2808</td>
<td>10 530</td>
</tr>
<tr>
<td>Time</td>
<td>2 285s</td>
<td>7 788s</td>
</tr>
</tbody>
</table>

Subdomain preconditioning (32 subdomains):

| Iterations | 263 505 |
| Time       | 3 838s  | 19 644s |

Two-grid preconditioning:

| Iterations | 59 117 |
| Time       | 307s  | 346s   |