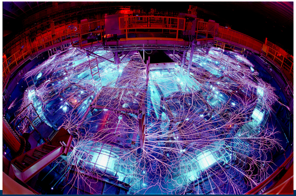
Exceptional service in the national interest

Sandia National Laboratories



Multigrid solvers for Maxwell's equations in Trilinos

Christian Glusa

Joint work with Jonathan Hu, Chris Siefert and Ray Tuminaro

Center for Computing Research, Sandia National Laboratories

March 19, 2026

U.S. DEPARTMENT OF ENERGY

NNSA National Nuclear Security Administration

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525. SAND NO. SAND2026-18372C

Curl-Curl problem

We want to solve

$$\nabla \times \nabla \times u + \mu u = f \qquad \text{+ boundary conditions}$$

Exact sequences (deRham complex; no boundary conditions)

$$0 \longrightarrow \mathbb{R} \longrightarrow H(\nabla) \xrightarrow{\nabla} H(\nabla\times) \xrightarrow{\nabla\times} H(\nabla\cdot) \xrightarrow{\nabla\cdot} L^2 \longrightarrow 0$$

$$0 \longrightarrow \mathbb{R} \longrightarrow \text{Nodal FE} \xrightarrow{D^{(n\to e)}} \text{Edge FE} \xrightarrow{D^{(e\to f)}} \text{Face FE} \xrightarrow{D^{(f\to v)}} \text{Volume FE} \longrightarrow 0$$

$n =$ nodes, $e =$ edges, $f =$ faces, $v =$ volumes

System matrix:

$$A^{(e)} := \underbrace{\left(D^{(e\to f)}\right)^T M^{(f)} D^{(e\to f)}}_{=:S^{(e)} \text{ stiffness matrix}} + M^{(e)} \qquad \text{mass matrices } M^{(f)}, M^{(e)}$$

Observation

Action of $A^{(e)}$ is quite differently on null $D^{(e\to f)} =$ range $D^{(n\to e)}$ and its complement.

$\Rightarrow$ Iterative solvers will need to act on both subspaces to be effective.

$\Rightarrow$ Construct preconditioners for $A^{(e)}$ and $A^{(n)} := \left(D^{(n\to e)}\right)^T A^{(e)} D^{(n\to e)}$.

Discrete differential operators

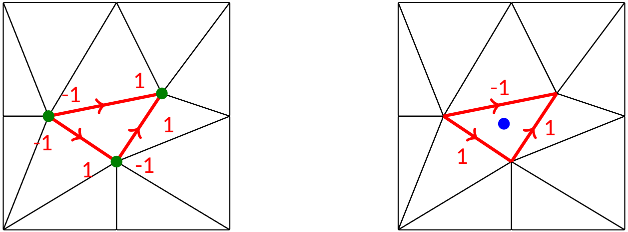


Figure: *Left:* discrete gradient $D^{(n\to e)}$, connecting nodes to edges *Right:* discrete curl $D^{(e\to f)}$, connecting edges to faces

- For lowest order finite elements, the discrete differential operators are signed incidence matrices.
- Finite element codes can be leveraged to assemble these operators via interpolation , e.g. $I_h\left(\nabla u_h\right)$
- In Trilinos: `Panzer` and `Intrepid2` can assemble discrete operators for arbitrary polynomial order into matrices or matrix-free operators.

Hybrid smoothers

"Hiptmair hybrid smoothing" combines the application of smoothers on $A^{(e)}$ and $A^{(n)}$ via

$$\mathbf{x} := \mathbf{x} + \left(\tilde{A}^{(e)}\right)^{-1}\left(\mathbf{b} - A^{(e)}\mathbf{x}\right)$$

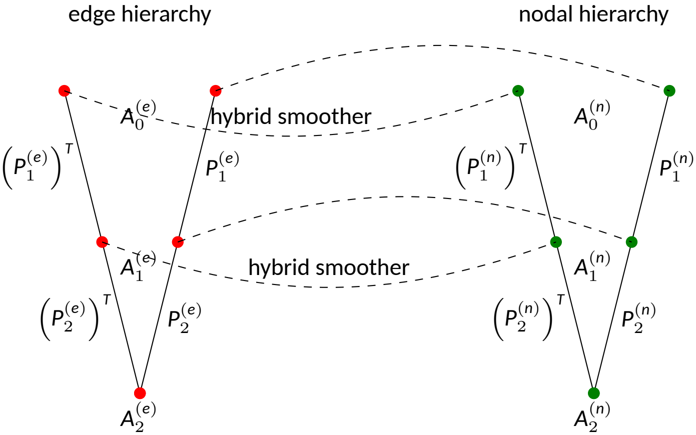
$$\mathbf{x} := \mathbf{x} + D^{(n\to e)}\left(\tilde{A}^{(n)}\right)^{-1}\left(D^{(n\to e)}\right)^{T}\left(\mathbf{b} - A^{(e)}\mathbf{x}\right)$$

where $\tilde{A}^{(e)}$ and $\tilde{A}^{(n)}$ are approximations of $A^{(e)}$ and $A^{(n)}$.

`MueLu` parameters:

```
"smoother: type": "hiptmair"
"smoother: params":
  "hiptmair: smoother type 1": "chebyshev"
  "hiptmair: smoother list 1": ...
  "hiptmair: smoother type 2": "chebyshev"
  "hiptmair: smoother list 2": ...
```

Geometric Multigrid



- Higher order discretizations can be solved using *p*-coarsening
  $\Rightarrow$ Need AMG approach for lowest order elements.

Commuting relationship

Nodal problem on the fine level:

$$A^{(n)} = \left(D^{(n\to e)}\right)^T A^{(e)} D^{(n\to e)}.$$

Coarse operators via Galerkin coarsening:

$$A_H^{(n)} := \left(P^{(n)}\right)^T A^{(n)} P^{(n)}, \qquad A_H^{(e)} := \left(P^{(e)}\right)^T A^{(e)} P^{(e)}.$$

We want that the same relationship between $A_H^{(n)}$ and $A_H^{(e)}$ holds as on the fine level, i.e.

$$A_H^{(n)} = \left(D_H^{(n\to e)}\right)^T A_H^{(e)} D_H^{(n\to e)}.$$

This is guaranteed if the commuting relationship

$$P^{(e)} D_H^{(n\to e)} = D^{(n\to e)} P^{(n)}$$

holds.

Commuting relationship

Algebraic multigrid preconditioners for Maxwell should aim to preserve

$$P^{(e)} D_H^{(n\to e)} = D^{(n\to e)} P^{(n)}$$

Reitzinger-Schöberl multigrid (`Maxwell1-RS`)

Idea

- Aggregate on the nodal problem $A^{(n)}$: $P^{(n,\text{piecewise})}$
- Construct coarse edges between aggregates using fine-level edges: $D_H^{(n\rightarrow e)}$
- Build edge prolongator that satisfies commuting relationship: $P^{(e,\text{piecewise})}$

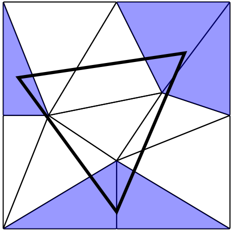


Figure: Nodal aggregates in blue, coarse edges in black

- The Reitzinger-Schöberl procedure relies heavily on the prolongators being piecewise constant.
- The resulting preconditioner is generally not scalable.

```
"maxwell1: 11list":
  "multigrid algorithm": "unsmoothed reitzinger"
"maxwell1: 22list":
  "multigrid algorithm": "unsmoothed"
```

Stefan Reitzinger and Joachim Schöberl. “An algebraic multigrid method for finite element discretizations with edge elements”. In: *Numerical linear algebra with applications* 9.3 (2002), pp. 223–238

Smoothed Maxwell multigrid (`Maxwell1-SA`)

Prolongator smoothing improves convergence. Using

$$P^{(n)} = \left\{I - \beta \operatorname{diag}\left[A^{(n)}\right]^{-1} A^{(n)}\right\} P^{(n,\text{piecewise})}$$

$$P^{(e)} = \left\{I - \alpha \operatorname{diag}\left[S^{(e)}\right]^{-1} S^{(e)}\right\} \left\{I - \beta D^{(n\to e)} \operatorname{diag}\left[A^{(n)}\right]^{-1} \left(D^{(n\to e)}\right)^T A^{(e)}\right\} P^{(e,\text{piecewise})}$$

also preserves the commuting relationship.

- β-smoothing needs to use the same damping parameter β.
- α-smoothing requires the curl-curl part $S^{(e)}$ of the edge matrix $A^{(e)}$.
  $\rightarrow$ Need to project separately. The triple matrix product is the most expensive part of AMG setup.
- $P^{(e)}$ is quite dense.

```
"maxwell1: 11list":
  "multigrid algorithm": "smoothed reitzinger"
  "sa: damping factor": alpha
"maxwell1: 22list":
  "multigrid algorithm": "sa"
  "sa: damping factor": beta
```

Jonathan J Hu, Raymond S Tuminaro, Pavel B Bochev, Christopher J Garasi, and Allen C Robinson. "Toward an h-independent algebraic multigrid method for Maxwell's equations". In: *SIAM Journal on Scientific Computing* 27.5 (2006), pp. 1669–1688

NEW Energy minimization multigrid (`Maxwell1-Emin`)

Fix a sparsity pattern $\mathcal{N}$ for the edge prolongator.

$$P^{(e)} := \operatorname{argmin}_P \frac{1}{2} \sum_j \|P_{:j}\|_A^2 \qquad \text{subject to } P \in \mathcal{N} \text{ and } PD_H^{(n\to e)} = D^{(n\to e)} P^{(n)}.$$

- Use Reitzinger-Schöberl approach to construct $D^{(n\to e)}$, but pick any $P^{(n)}$ we like.
  $\Rightarrow$ fewer constraints on $P^{(n)}$ (as long as we can construct an initial guess for the optimization)
- Better control of density of edge prolongators
- "Solve" optimization problem using a few steps of projected CG.
- Optimization step involves pseudo-inverse of large block-diagonal system.
  Solved using `KokkosKernels` batched LU. Might want to switch to batched QR or Krylov.

```
"maxwell1: 11list":
  "multigrid algorithm": "emin reitzinger"
  "emin: num iterations": 2
"maxwell1: 22list":
  "multigrid algorithm": "sa" # or "emin" or "unsmoothed"
```

Raymond Tuminaro and Christian Glusa. "A structure preserving H-curl algebraic multigrid method for the eddy current equations". In: *arXiv preprint arXiv:2506.08284* (2025)

Reformulated Maxwell multigrid (`RefMaxwell`)

Idea

Handle the nullspace of curl by augmenting to a Hodge Laplacian

$$\operatorname{curl}\operatorname{curl} u + \color{red}{\operatorname{grad}\operatorname{div} u} + \mu u$$

Augmented system

$$\begin{pmatrix} A^{(e)} + \text{addon} & M^{(e)} D^{(n\to e)} \\ \left(D^{(n\to e)}\right)^T M^{(e)} & A^{(n)} \end{pmatrix}$$

- Solve using block Jacobi preconditioner with AMG sub-solvers
- Project edge problem from Nedelec edge elements onto vectorial nodal basis.
- Edge and nodal problem are handled additively $\Rightarrow$ more efficient coarse levels
- No preservation of commuting relationship

Pavel B Bochev, Jonathan J Hu, Christopher M Siefert, and Raymond S Tuminaro. "An algebraic multigrid approach based on a compatible gauge reformulation of Maxwell's equations". In: *SIAM Journal on Scientific Computing* 31.1 (2008), pp. 557–583

Numerical experiments

Solve

$$\nabla \times \nabla \times u + \mu u = f \qquad \text{in } \Omega = [0,1]^3, \qquad \text{CFL} := \frac{1}{\mu h^2}$$

with

- $\mu = 1$, $\text{CFL} = h^{-2}$
- $\mu = \frac{1}{10h^2}$, $\beta = 1$, $\text{CFL} = 10$

Conjugate gradients with preconditioners:

- `Maxwell1-RS`
- `Maxwell1` with α- and β-smoothing
- `Maxwell1` with β-smoothing
- `Maxwell1` with α-smoothing
- Energy minimization `Maxwell1`, 2 iterations
- Energy minimization `Maxwell1`, 0 iterations
- `RefMaxwell-PA` ("plain aggregation")
- `RefMaxwell-SA`

Using `Panzer`'s `MiniEM` executable on Sandia's Solo system (Broadwell CPUs)

Caveat

I am comparing lots of solvers. Potential for missed parameter tuning opportunities.
All solvers use the same dropping strategy and the same rebalancing (except for `Maxwell1-SA` with α+β).

$\mu = 1$, CFL $= h^{-2}$

Data format: iterations / solve time / setup time

Hexahedral elements

size	ranks	Maxwell1-RS	Maxwell1-α+β	Maxwell1-α	Maxwell1-β	Maxwell1-Emin2	Maxwell1-Emin0	RefMaxwell-PA	RefMaxwell-SA
50k	1	21 / 0.51 / 0.40	10 / 0.29 / 0.98	16 / 0.39 / 0.30	19 / 0.46 / 0.30	7 / 0.17 / 0.72	8 / 0.20 / 0.49	15 / 0.20 / 0.53	14 / 0.19 / 0.53
390k	8	37 / 1.88 / 0.51	15 / 1.01 / 3.59	27 / 1.54 / 1.86	35 / 1.92 / 0.78	10 / 0.54 / 1.57	12 / 0.65 / 1.06	30 / 0.79 / 0.59	17 / 0.47 / 0.65
3M	64	72 / 6.88 / 0.67	19 / 2.64 / 9.07	43 / 5.04 / 5.40	63 / 6.94 / 1.62	11 / 1.20 / 2.75	14 / 1.56 / 1.89	49 / 2.67 / 0.84	20 / 1.36 / 3.27
24M	512	138 / 13.77 / 0.78	20 / 3.13 / 28.1	67 / 14.49 / 32.04	101 / 13.10 / 11.8	11 / 1.34 / 5.09	14 / 1.71 / 4.15	66 / 4.21 / 2.22	27 / 1.97 / 2.30
192M	4,096	272 / 28.92 / 1.20		94 / 56.70 / 239.7	203 / 34.04 / 9.49	14 / 1.86 / 5.11	17 / 2.30 / 3.95	114 / 7.79 / 1.47	42 / 3.45 / 2.15

- Unsmoothed variants are not scalable: `Maxwell1-RS` and `RefMaxwell-PA`

$\mu = 1$, CFL $= h^{-2}$

Data format: iterations / solve time / setup time

Hexahedral elements

size	ranks	Maxwell1-RS	Maxwell1-α+β	Maxwell1-α	Maxwell1-β	Maxwell1-Emin2	Maxwell1-Emin0	RefMaxwell-PA	RefMaxwell-SA
50k	1	21 / 0.51 / 0.40	10 / 0.29 / 0.98	16 / 0.39 / 0.30	19 / 0.46 / 0.30	7 / 0.17 / 0.72	8 / 0.20 / 0.49	15 / 0.20 / 0.53	14 / 0.19 / 0.53
390k	8	37 / 1.88 / 0.51	15 / 1.01 / 3.59	27 / 1.54 / 1.86	35 / 1.92 / 0.78	10 / 0.54 / 1.57	12 / 0.65 / 1.06	30 / 0.79 / 0.59	17 / 0.47 / 0.65
3M	64	72 / 6.88 / 0.67	19 / 2.64 / 9.07	43 / 5.04 / 5.40	63 / 6.94 / 1.62	11 / 1.20 / 2.75	14 / 1.56 / 1.89	49 / 2.67 / 0.84	20 / 1.36 / 3.27
24M	512	138 / 13.77 / 0.78	20 / 3.13 / 28.1	67 / 14.49 / 32.04	101 / 13.10 / 11.8	11 / 1.34 / 5.09	14 / 1.71 / 4.15	66 / 4.21 / 2.22	27 / 1.97 / 2.30
192M	4,096	272 / 28.92 / 1.20		94 / 56.70 / 239.7	203 / 34.04 / 9.49	14 / 1.86 / 5.11	17 / 2.30 / 3.95	114 / 7.79 / 1.47	42 / 3.45 / 2.15

- Unsmoothed variants are not scalable: `Maxwell1-RS` and `RefMaxwell-PA`
- Maxwell1 with α+β prolongator smoothing builds pretty dense matrices and runs out of memory

$\mu = 1$, CFL $= h^{-2}$

Data format: iterations / solve time / setup time

Hexahedral elements

size	ranks	Maxwell1-RS	Maxwell1-α+β	Maxwell1-α	Maxwell1-β	Maxwell1-Emin2	Maxwell1-Emin0	RefMaxwell-PA	RefMaxwell-SA
50k	1	21 / 0.51 / 0.40	10 / 0.29 / 0.98	16 / 0.39 / 0.30	19 / 0.46 / 0.30	7 / 0.17 / 0.72	8 / 0.20 / 0.49	15 / 0.20 / 0.53	14 / 0.19 / 0.53
390k	8	37 / 1.88 / 0.51	15 / 1.01 / 3.59	27 / 1.54 / 1.86	35 / 1.92 / 0.78	10 / 0.54 / 1.57	12 / 0.65 / 1.06	30 / 0.79 / 0.59	17 / 0.47 / 0.65
3M	64	72 / 6.88 / 0.67	19 / 2.64 / 9.07	43 / 5.04 / 5.40	63 / 6.94 / 1.62	11 / 1.20 / 2.75	14 / 1.56 / 1.89	49 / 2.67 / 0.84	20 / 1.36 / 3.27
24M	512	138 / 13.77 / 0.78	20 / 3.13 / 28.1	67 / 14.49 / 32.04	101 / 13.10 / 11.8	11 / 1.34 / 5.09	14 / 1.71 / 4.15	66 / 4.21 / 2.22	27 / 1.97 / 2.30
192M	4,096	272 / 28.92 / 1.20		94 / 56.70 / 239.7	203 / 34.04 / 9.49	14 / 1.86 / 5.11	17 / 2.30 / 3.95	114 / 7.79 / 1.47	42 / 3.45 / 2.15

- Unsmoothed variants are not scalable: `Maxwell1-RS` and `RefMaxwell-PA`
- Maxwell1 with α+β prolongator smoothing builds pretty dense matrices and runs out of memory
- `Maxwell1` needs α+β smoothing for good iteration counts

$\mu = 1$, CFL $= h^{-2}$

Data format: iterations / solve time / setup time

Hexahedral elements

size	ranks	Maxwell1-RS	Maxwell1-α+β	Maxwell1-α	Maxwell1-β	Maxwell1-Emin2	Maxwell1-Emin0	RefMaxwell-PA	RefMaxwell-SA
50k	1	21 / 0.51 / 0.40	10 / 0.29 / 0.98	16 / 0.39 / 0.30	19 / 0.46 / 0.30	7 / 0.17 / 0.72	8 / 0.20 / 0.49	15 / 0.20 / 0.53	14 / 0.19 / 0.53
390k	8	37 / 1.88 / 0.51	15 / 1.01 / 3.59	27 / 1.54 / 1.86	35 / 1.92 / 0.78	10 / 0.54 / 1.57	12 / 0.65 / 1.06	30 / 0.79 / 0.59	17 / 0.47 / 0.65
3M	64	72 / 6.88 / 0.67	19 / 2.64 / 9.07	43 / 5.04 / 5.40	63 / 6.94 / 1.62	11 / 1.20 / 2.75	14 / 1.56 / 1.89	49 / 2.67 / 0.84	20 / 1.36 / 3.27
24M	512	138 / 13.77 / 0.78	20 / 3.13 / 28.1	67 / 14.49 / 32.04	101 / 13.10 / 11.8	11 / 1.34 / 5.09	14 / 1.71 / 4.15	66 / 4.21 / 2.22	27 / 1.97 / 2.30
192M	4,096	272 / 28.92 / 1.20		94 / 56.70 / 239.7	203 / 34.04 / 9.49	14 / 1.86 / 5.11	17 / 2.30 / 3.95	114 / 7.79 / 1.47	42 / 3.45 / 2.15

- Unsmoothed variants are not scalable: `Maxwell1-RS` and `RefMaxwell-PA`
- Maxwell1 with α+β prolongator smoothing builds pretty dense matrices and runs out of memory
- `Maxwell1` needs α+β smoothing for good iteration counts
- Smoothed variants give iterations $\sim$ log CFL

$\mu = 1$, CFL $= h^{-2}$

Data format: iterations / solve time / setup time

Hexahedral elements

size	ranks	Maxwell1-RS	Maxwell1-α+β	Maxwell1-α	Maxwell1-β	Maxwell1-Emin2	Maxwell1-Emin0	RefMaxwell-PA	RefMaxwell-SA
50k	1	21 / 0.51 / 0.40	10 / 0.29 / 0.98	16 / 0.39 / 0.30	19 / 0.46 / 0.30	7 / 0.17 / 0.72	8 / 0.20 / 0.49	15 / 0.20 / 0.53	14 / 0.19 / 0.53
390k	8	37 / 1.88 / 0.51	15 / 1.01 / 3.59	27 / 1.54 / 1.86	35 / 1.92 / 0.78	10 / 0.54 / 1.57	12 / 0.65 / 1.06	30 / 0.79 / 0.59	17 / 0.47 / 0.65
3M	64	72 / 6.88 / 0.67	19 / 2.64 / 9.07	43 / 5.04 / 5.40	63 / 6.94 / 1.62	11 / 1.20 / 2.75	14 / 1.56 / 1.89	49 / 2.67 / 0.84	20 / 1.36 / 3.27
24M	512	138 / 13.77 / 0.78	20 / 3.13 / 28.1	67 / 14.49 / 32.04	101 / 13.10 / 11.8	11 / 1.34 / 5.09	14 / 1.71 / 4.15	66 / 4.21 / 2.22	27 / 1.97 / 2.30
192M	4,096	272 / 28.92 / 1.20		94 / 56.70 / 239.7	203 / 34.04 / 9.49	14 / 1.86 / 5.11	17 / 2.30 / 3.95	114 / 7.79 / 1.47	42 / 3.45 / 2.15

- Unsmoothed variants are not scalable: `Maxwell1-RS` and `RefMaxwell-PA`
- Maxwell1 with α+β prolongator smoothing builds pretty dense matrices and runs out of memory
- `Maxwell1` needs α+β smoothing for good iteration counts
- Smoothed variants give iterations $\sim$ log CFL
- Setup time differences between `Emin2` and `Emin0` show cost of optimization

$\mu = 1$, CFL $= h^{-2}$

Data format: iterations / solve time / setup time

Hexahedral elements

size	ranks	Maxwell1-RS	Maxwell1-α+β	Maxwell1-α	Maxwell1-β	Maxwell1-Emin2	Maxwell1-Emin0	RefMaxwell-PA	RefMaxwell-SA
50k	1	21 / 0.51 / 0.40	10 / 0.29 / 0.98	16 / 0.39 / 0.30	19 / 0.46 / 0.30	7 / 0.17 / 0.72	8 / 0.20 / 0.49	15 / 0.20 / 0.53	14 / 0.19 / 0.53
390k	8	37 / 1.88 / 0.51	15 / 1.01 / 3.59	27 / 1.54 / 1.86	35 / 1.92 / 0.78	10 / 0.54 / 1.57	12 / 0.65 / 1.06	30 / 0.79 / 0.59	17 / 0.47 / 0.65
3M	64	72 / 6.88 / 0.67	19 / 2.64 / 9.07	43 / 5.04 / 5.40	63 / 6.94 / 1.62	11 / 1.20 / 2.75	14 / 1.56 / 1.89	49 / 2.67 / 0.84	20 / 1.36 / 3.27
24M	512	138 / 13.77 / 0.78	20 / 3.13 / 28.1	67 / 14.49 / 32.04	101 / 13.10 / 11.8	11 / 1.34 / 5.09	14 / 1.71 / 4.15	66 / 4.21 / 2.22	27 / 1.97 / 2.30
192M	4,096	272 / 28.92 / 1.20		94 / 56.70 / 239.7	203 / 34.04 / 9.49	14 / 1.86 / 5.11	17 / 2.30 / 3.95	114 / 7.79 / 1.47	42 / 3.45 / 2.15

- Unsmoothed variants are not scalable: `Maxwell1-RS` and `RefMaxwell-PA`
- Maxwell1 with α+β prolongator smoothing builds pretty dense matrices and runs out of memory
- `Maxwell1` needs α+β smoothing for good iteration counts
- Smoothed variants give iterations $\sim$ log CFL
- Setup time differences between `Emin2` and `Emin0` show cost of optimization
- `RefMaxwell` iterations are generally faster than `Maxwell1`, but `Maxwell1-Emin` requires fewer iterations.

$\mu = 1$, CFL $= h^{-2}$

Data format: iterations / solve time / setup time

Hexahedral elements

size	ranks	Maxwell1-RS	Maxwell1-α+β	Maxwell1-α	Maxwell1-β	Maxwell1-Emin2	Maxwell1-Emin0	RefMaxwell-PA	RefMaxwell-SA
50k	1	21 / 0.51 / 0.40	10 / 0.29 / 0.98	16 / 0.39 / 0.30	19 / 0.46 / 0.30	7 / 0.17 / 0.72	8 / 0.20 / 0.49	15 / 0.20 / 0.53	14 / 0.19 / 0.53
390k	8	37 / 1.88 / 0.51	15 / 1.01 / 3.59	27 / 1.54 / 1.86	35 / 1.92 / 0.78	10 / 0.54 / 1.57	12 / 0.65 / 1.06	30 / 0.79 / 0.59	17 / 0.47 / 0.65
3M	64	72 / 6.88 / 0.67	19 / 2.64 / 9.07	43 / 5.04 / 5.40	63 / 6.94 / 1.62	11 / 1.20 / 2.75	14 / 1.56 / 1.89	49 / 2.67 / 0.84	20 / 1.36 / 3.27
24M	512	138 / 13.77 / 0.78	20 / 3.13 / 28.1	67 / 14.49 / 32.04	101 / 13.10 / 11.8	11 / 1.34 / 5.09	14 / 1.71 / 4.15	66 / 4.21 / 2.22	27 / 1.97 / 2.30
192M	4,096	272 / 28.92 / 1.20		94 / 56.70 / 239.7	203 / 34.04 / 9.49	14 / 1.86 / 5.11	17 / 2.30 / 3.95	114 / 7.79 / 1.47	42 / 3.45 / 2.15

Tetrahedral elements

size	ranks	Maxwell1-RS	Maxwell1-α+β	Maxwell1-α	Maxwell1-β	Maxwell1-Emin2	Maxwell1-Emin0	RefMaxwell-PA	RefMaxwell-SA
115k	1	55 / 2.19 / 0.53	26 / 3.31 / 10.26	32 / 1.60 / 0.84	52 / 3.56 / 1.44	13 / 0.77 / 2.82	14 / 0.82 / 1.75	33 / 0.78 / 3.01	30 / 0.67 / 3.03
897k	8	95 / 7.57 / 0.78	28 / 8.16 / 83.20	48 / 5.38 / 2.50	63 / 9.98 / 13.79	14 / 1.84 / 10.14	15 / 1.95 / 8.37	42 / 1.59 / 1.05	34 / 1.28 / 1.08
7M	64	142 / 21.33 / 1.25		59 / 12.93 / 22.94	144 / 44.64 / 11.66	15 / 3.65 / 9.31	17 / 4.12 / 6.32	66 / 4.69 / 1.05	36 / 2.84 / 1.38
56M	512	288 / 45.16 / 1.39		94 / 27.36 / 38.90	298 / 122.5 / 24.13	16 / 4.45 / 11.76	20 / 5.61 / 8.27	111 / 8.55 / 1.20	44 / 4.37 / 2.85
449M	4,096	582 / 197.45 / 1.7		153 / 76.78 / 129.4	499 / 250.3 / 63.19	24 / 8.06 / 26.32	25 / 8.44 / 22.80	162 / 15.20 / 3.47	68 / 7.70 / 2.79

- Unsmoothed variants are not scalable: `Maxwell1-RS` and `RefMaxwell-PA`
- Maxwell1 with α+β prolongator smoothing builds pretty dense matrices and runs out of memory
- `Maxwell1` needs α+β smoothing for good iteration counts
- Smoothed variants give iterations $\sim$ log CFL
- Setup time differences between `Emin2` and `Emin0` show cost of optimization
- `RefMaxwell` iterations are generally faster than `Maxwell1`, but `Maxwell1-Emin` requires fewer iterations.
- Tets give worse results than hexes.

$\mu = 0.1h^{-2}$, CFL $= 10$

Data format: iterations / solve time / setup time

Hexahedral elements

size	ranks	Maxwell1-RS	Maxwell1-α+β	Maxwell1-α	Maxwell1-β	Maxwell1-Emin2	Maxwell1-Emin0	RefMaxwell-PA	RefMaxwell-SA
50k	1	13 / 0.34 / 0.39	7 / 0.19 / 0.96	12 / 0.29 / 0.30	11 / 0.27 / 0.30	7 / 0.17 / 0.74	7 / 0.17 / 0.50	12 / 0.16 / 0.53	14 / 0.19 / 0.52
390k	8	15 / 0.76 / 0.48	8 / 0.55 / 3.64	13 / 0.75 / 1.85	13 / 0.70 / 0.78	7 / 0.38 / 1.56	7 / 0.38 / 1.06	12 / 0.32 / 0.59	15 / 0.41 / 0.67
3M	64	15 / 1.47 / 0.65	8 / 1.11 / 9.13	13 / 1.53 / 5.42	13 / 1.47 / 1.62	7 / 0.77 / 2.77	7 / 0.81 / 1.91	12 / 0.68 / 0.84	16 / 1.07 / 3.27
24M	512	15 / 1.54 / 0.79	8 / 1.26 / 28.07	13 / 2.81 / 31.96	13 / 1.68 / 11.81	7 / 0.85 / 5.01	7 / 0.86 / 4.18	12 / 0.76 / 2.21	16 / 1.15 / 1.61
192M	4,096	15 / 1.67 / 1.75		13 / 8.05 / 240.97	13 / 2.20 / 9.16	7 / 0.95 / 5.33	7 / 0.95 / 3.81	12 / 0.77 / 1.43	16 / 1.36 / 2.38

Tetrahedral elements

size	ranks	Maxwell1-RS	Maxwell1-α+β	Maxwell1-α	Maxwell1-β	Maxwell1-Emin2	Maxwell1-Emin0	RefMaxwell-PA	RefMaxwell-SA
115k	1	34 / 1.37 / 0.55	18 / 2.31 / 10.28	25 / 1.23 / 0.85	29 / 1.98 / 1.46	12 / 0.70 / 2.79	13 / 0.76 / 1.72	29 / 0.64 / 3.02	32 / 0.71 / 3.00
897k	8	39 / 3.13 / 0.78	18 / 5.29 / 82.01	25 / 2.80 / 2.49	31 / 4.91 / 12.60	12 / 1.57 / 10.16	14 / 1.83 / 8.26	28 / 1.07 / 1.06	34 / 1.27 / 1.09
7M	64	41 / 6.17 / 1.21		26 / 5.72 / 20.97	34 / 10.84 / 12.10	12 / 2.92 / 9.27	14 / 3.42 / 6.34	28 / 2.00 / 1.05	34 / 2.67 / 1.38
56M	512	42 / 6.58 / 1.37		26 / 7.59 / 38.42	35 / 14.21 / 23.59	12 / 3.33 / 11.76	14 / 3.95 / 8.24	28 / 2.14 / 1.21	34 / 3.39 / 2.84
449M	4,096	42 / 6.86 / 1.92		26 / 12.92 / 130.63	35 / 17.13 / 67.63	12 / 3.88 / 27.45	13 / 4.11 / 23.27	28 / 2.53 / 3.45	34 / 3.69 / 2.57

- Unsmoothed variants are not scalable: `Maxwell1-RS` and `RefMaxwell-PA`
- Maxwell1 with α+β prolongator smoothing builds pretty dense matrices and runs out of memory
- `Maxwell1` needs α+β smoothing for good iteration counts
- Smoothed variants give iterations $\sim$ log CFL
- Setup time differences between `Emin2` and `Emin0` show cost of optimization
- `RefMaxwell` iterations are generally faster than `Maxwell1`, but `Maxwell1-Emin` requires fewer iterations.
- Tets give worse results than hexes.
- At fixed CFL, all options scale well.

Conclusions and Outlook

Sandia National Laboratories

Choice of appropriate Maxwell solver depends primarily on

- CFL regime
- ratio of # setups vs # solves
- problem size

- Promising new energy minimization preconditioner of `Maxwell1` type
- Still some room for algorithmic improvements and parameter tuning
- Comparisons on a more interesting problems (coefficient variations, boundary conditions) and on GPU systems
- Experiment with replacing hybrid smoothing in `Maxwell1` variants with `RefMaxwell` style additive subspace solves
- `RefMaxwell` idea has been extended to other spaces in the deRham complex. Do the same with `Maxwell1`?