Exploring generalized Jacobi preconditioners and smoothers in MFEM

Gabriel Pinochet-Soto [1] Tzanio Kolev [2] Chak Shing Lee [2]

[1] Fariborz Maseeh Department of Mathematics and Statistics, Portland State University

[2] Center for Applied Scientific Computing, Lawrence Livermore National Laboratory

October 21, 2024

Portland State UNIVERSITY

Lawrence Livermore National Laboratory

Preliminaries

Consider $A = (a_{ij})_{ij} \in \mathbb{R}^{n \times n}$ an SPD operator.

A-convergent smoother

We say M is an A-convergent smoother if $M + M^T - A$ is SPD, i.e., $(Ax, x) < (Mx, x) + (M^T x, x) = 2(Mx, x)$, for all $x \in \mathbb{R}^n$.

Boundedness of off-diagonal entries

A C.B.S. inequality implies $|a_{ij}| \leq \sqrt{a_{ii}}\sqrt{a_{jj}}$.

Two-level preconditioner

The two-level preconditioner will be SPD provided M being a A-convergent smoother.

$$B_{\mathrm{TL}}^{-1} = M(M^T + M - A)^{-1} M^T + \text{ SPSD term } .$$

$\ell_{p,q}$-Jacobi preconditioners

We define the family of (diagonal) $\ell_{p,q}$-Jacobi preconditioners $\{\mathrm{D}_{p,q}\}_{p,q}$, for $p \geq 0, q \in \mathbb{R}$, by

$$(\mathrm{D}_{p,q})_{i,i} := \sum_j \left(\frac{|a_{ij}|}{a_{ii}^{1-\frac{q}{p}} a_{jj}^{\frac{q}{p}}} \right)^p a_{ii}, \qquad (1)$$

which can conveniently written as $\mathrm{D}_{p,q} = \mathrm{diag}(\mathrm{D}^{1+q-p}|\mathrm{A}|^p \mathrm{D}^{-q}1)$, where D is the diagonal matrix of A, i.e., $(\mathrm{D})_{ii} := a_{ii}$, and we understand the operations as *entry-wise* operations.

Specific examples of Jacobi-type preconditioners

Under the assumption of a diagonally dominant matrices ($a_{ii} = \max_j |a_{ij}|$), we have some examples.

1. $p = 0, q = 0 \mapsto$ Row-wise re-scaled Jacobi smoother
   $(\mathrm{D}_{0,0})_{ii} = \mathrm{nnz}_i \, a_{ii}$.
2. $p = 1, q = 0 \mapsto \ell_1$-Jacobi smoother
   $(\mathrm{D}_{1,0})_{ii} = a_{ii} + \sum_{j \neq i} |a_{ij}|$.
3. $p = 2, q = 0 \mapsto \mathrm{D}_{2,0} = \arg\min_{\mathrm{D}} \|\mathrm{Id} - \mathrm{D}^{-1}\mathrm{A}\|_{\mathrm{Fro}}$
   $(\mathrm{D}_{2,0})_{ii} = a_{ii} + \sum_{j \neq i} \frac{|a_{ij}|^2}{a_{ii}}$.
4. $p = \infty, q = 0 \mapsto$ Jacobi smoother
   $(\mathrm{D}_{\infty,0})_{ii} = a_{ii}$.

Properties of the $\ell_{p,q}$-Jacobi family

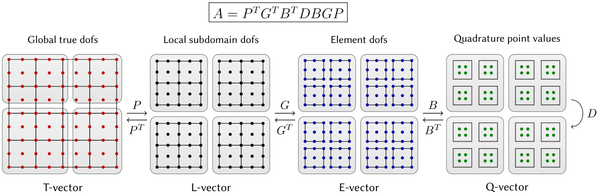
1. $\mathrm{D}_{p_1,q_1} \leq \mathrm{D}_{p_2,q_2}$, if $q_1 - q_2 = \frac{p_1 - p_2}{2}$ and $p_1 \geq p_2$.
2. A weighted Young's inequality is the key step to prove $\mathrm{A} \leq \mathrm{D}_{p,q}$ for $0 \leq p \leq 1$ and all q.
3. All $\mathrm{D}_{p,q}$, for $0 \leq p \leq 1$ and all q, are A-convergent smoothers.

Question

Can we get a better smoother (or preconditioner) when increasing p?

Absolute-value ℓ_1-Jacobi preconditioner

In the context of finite element methods, we usually have an operator of the form



We know that the ℓ_1-Jacobi preconditioner is convergent: $A \le D_1$. A rough approximation of the ℓ_1-Jacobi smoother can be made by just employing triangle inequality.

$$D_{\mathrm{Abs}-\ell_1} = \mathrm{diag}(|P|^T |G|^T |B|^T D |B||G||P| 1), \qquad (2)$$

so we have $A \le D_1 \le D_{\mathrm{Abs}-\ell_1}$.

Numerical Results: Diffusion Problem on ICF Mesh

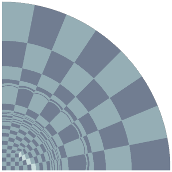
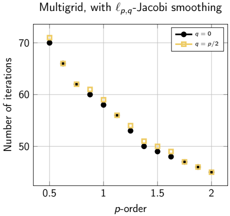


Figure: ICF mesh



Ord.	N. Iter.	N. DoF.
1	235	3,721
2	587	14,609
3	941	32,665
4	1,370	57,889
5	1,824	90,281
6	2,323	129,841
7	2,855	176,569
8	3,399	230,465
9	3,949	291,529
10	4,531	359,761

(a) Without preconditioner

Ord.	N. Iter.	N. DoF.
1	155	3,721
2	351	14,609
3	569	32,665
4	766	57,889
5	895	90,281
6	1,125	129,841
7	1,314	176,569
8	1,596	230,465
9	1,812	291,529
10	2,068	359,761

(b) Abs. ℓ_1-Jacobi precond.

Ord.	N. Iter.	N. DoF.
1	49	3,721
2	98	14,609
3	143	32,665
4	148	57,889
5	192	90,281
6	190	129,841
7	227	176,569
8	247	230,465
9	271	291,529
10	272	359,761

(c) MG, abs. ℓ_1-Jacobi smooth.

Table: Diffusion problem on `icf.mesh`, with partial assembly, utilizing PCG.

Current status

Main modifications

1. Current PR #4498 Jacobi-type of preconditions/smoothers
2. Implementation of `AbsMult` on matrices.
3. `AddAbsMult` on domain integrators.
4. Miniapp with Multigrid wrappers (cf. Example 26p).
5. Comparable with previous examples (cf. Example 1p, 2p, 3p).

Examine the current status!

(a) MFEM website

(b) PR #4498

Implementation of absolute-value multiplication: CurlCurl kernel

Let us consider u_h a FE discretization for the definite Maxwell problem:

$$\operatorname{curl} u_h = \sum_i u_i \operatorname{curl} \phi_{h,i}.$$

The curl of a function of the form (e.g.) $v = \phi^{3\mathrm{D}}(x) e_1$ is

$$\operatorname{curl} v = (0, \partial_2 \phi^{3\mathrm{D}}(x), -\partial_1 \phi^{3\mathrm{D}}(x)).$$

We get the absolute-value application of B by taking the absolute value of the basis function on the quadrature points and making sure the curl does not introduce a negative sign.

Implementation of absolute-value multiplication: CurlCurl kernel

```cpp
template<int T_D1D = 0, int T_Q1D = 0>
inline void PACurlCurlApply3D(const int d1d,
                              const int q1d,
                              const bool symmetric,
                              const int NE,
                              const Array<real_t> &bo,
                              const Array<real_t> &bc,
                              const Array<real_t> &bot,
                              const Array<real_t> &bct,
                              const Array<real_t> &gc,
                              const Array<real_t> &gct,
                              const Vector &pa_data,
                              const Vector &x,
                              Vector &y,
                              bool useAbs = false)
{
   // ...
         // x component
                  for (int qx = 0; qx < Q1D; ++qx)
                  {
                     // \hat{\nabla}\times\hat{u} is [0, (u_0)_{x_2}, -(u_0)_{x_1}]
                     curl[qz][qy][qx][1] += gradXY[qy][qx][1] * wDz; // (u_0)_{x_2}
                     if (!useAbs) { curl[qz][qy][qx][2] -= gradXY[qy][qx][0] * wz; } // -(u_0)_{x_1}
                     else { curl[qz][qy][qx][2] += gradXY[qy][qx][0] * wz; }    // +(u_0)_{x_1}
                  }
   // ...
}
```