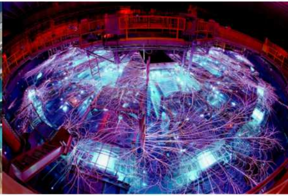
Laboratories

SAND2020-1762C



Polynomial Preconditioned GMRES in Trilinos:

Practical Considerations for High Performance Computing

Jennifer Loe, Heidi Thornquist, Erik Boman

2/9/20

Polynomial Preconditioning:

Solving Polynomial Preconditioned System: $Ax = b$ becomes

$$Ap(A)y = b,$$
$$x = p(A)y.$$

where $Ap(A)$ is a polynomial of degree d.

Choose $p(A)$ to be the minimum residual polynomial from GMRES.

Key fact: We are using the GMRES polynomial to precondition GMRES.

Trilinos

Sandia National Laboratories

Belos: Iterative Linear Solvers Package: CG, GMRES, Block Krylov methods, BiCGStab
Other Capabilities: Algebraic preconditioners (IFPACK), load partitioning (Zoltan), Direct Solvers (Amesos), Multigrid (MueLu), Eigensolvers (Anasazi), and more.
Application Areas: Circuit simulation, Ice sheet modeling, hydrodynamics, geophysics, etc.

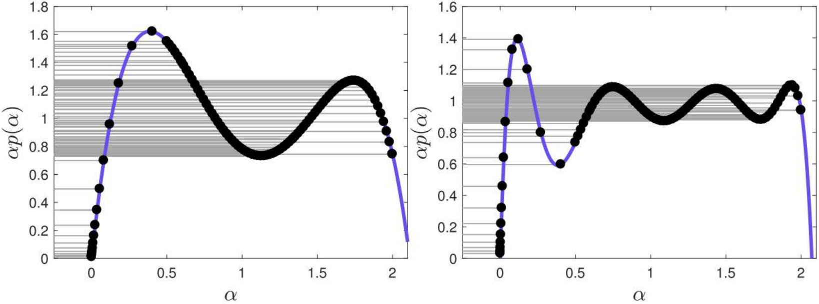
GMRES polynomial can precondition any solver in Belos!

Why precondition with the GMRES polynomial??

Sandia National Laboratories

- Reduces number of GMRES iterations (and often matrix-vector products).
- More work done between orthogonalization steps; avoid global synchronizations and communication.
- It's available in Trilinos! (Belos linear solvers package)
- General-purpose preconditioner.
- Matrix-free implementation.
- Can be composed with other preconditioners!
- Stability for high degrees with root-adding

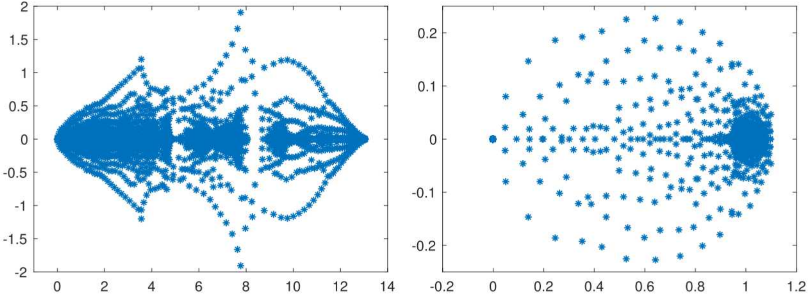
Re-Mapping Eigenvalues



(a) $deg(Ap(A)) = 4$ (b) $deg(Ap(A)) = 8$

x-axis: interval containing spectrum of *A*
y-axis: interval containing spectrum of $Ap(A)$
Black dots indicate eigenvalues of *A* being mapped by the polynomial to eigenvalues of $Ap(A)$.

Re-Mapping Eigenvalues



(a) Eigenvalues of A

(b) Eigenvalues of $Ap(A)$, with $deg(Ap(A)) = 6$

Obtaining the polynomial:

To find the polynomial $p(A)$ of degree $d-1$:

1. Run d steps of GMRES on the matrix A, using a random right-hand side.
   (To combine with another preconditioner M, run d steps of GMRES on AM.)
2. Use the resulting matrices to compute the harmonic Ritz values θ_i of A. (or AM.)
3. Order the θ_i's using a Modified Leja ordering. (Bai, Hu, Reichel)
4. Use the θ_i's to apply the polynomial as a preconditioner.

(Also options for root-adding or damping if needed for stability.)
[See Embree, Loe, Morgan 2018]

Polynomial Preconditioning: Implementation

Option 1: Use both formulas.

$$\begin{aligned} Ap(A)y &= b, \\ x &= p(A)y. \end{aligned}$$

$$Ap(A) = \prod_{i=1}^{d} \left(I - \frac{1}{\theta_i} A \right) \tag{1}$$

$$p(A) = \sum_{k=1}^{d} \frac{1}{\theta_k} \left(I - \frac{1}{\theta_1} A \right) \left(I - \frac{1}{\theta_2} A \right) \cdots \left(I - \frac{1}{\theta_{k-1}} A \right) \tag{2}$$

Advantage: Simpler formula. Less vector additions.
Disadvantage: Possible stability issues applying different operator.

Polynomial Preconditioning: Implementation

Option 2: Use one formula. (Implemented in Trilinos.)

$$Ap(A)y = b,$$
$$x = p(A)y.$$

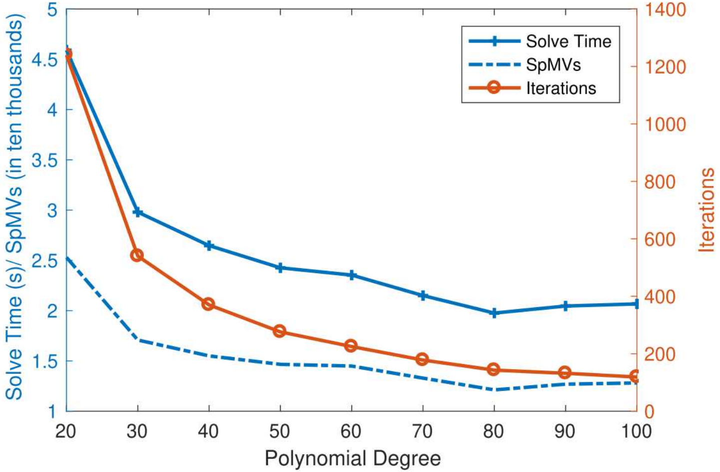
$$Ap(A) = \prod_{i=1}^{d} \left(I - \frac{1}{\theta_i} A \right) \quad (1)$$

$$p(A) = \sum_{k=1}^{d} \frac{1}{\theta_k} \left(I - \frac{1}{\theta_1} A \right) \left(I - \frac{1}{\theta_2} A \right) \cdots \left(I - \frac{1}{\theta_{k-1}} A \right) \quad (2)$$

Advantage: Applying a consistent operator.
Disadvantage: Up to 2x as many vector additions.

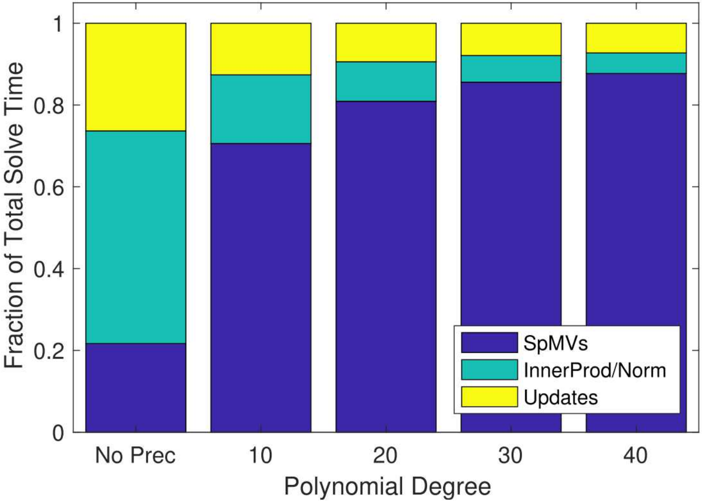
A Small CFD example:

Matrix **cfd2**, A is SPD, $n = 123440$. GMRES(50) with b random. (32 MPI processes over 1 node)



A Small CFD example:

Scaled solve time distribution per polynomial degree:



High Degrees can be Worth it!

Matrix **ML_Geer** (Janna collection), poroelastic structure problem, $n = 1{,}504{,}002$, nonsymmetric, GMRES(100), rtol = 1×10^{-8}

	SPMVs	**Time**	**Iters**	**Add roots**
Deg 20	260500	3214	12897	0
Deg 40	61580	731.5	1487	1
Deg 60	29570	346.7	472	2
Deg 80	16970	197	200	4

(Using 32 MPI processes over 1 node.)

Composing with other Preconditioners

Poly preconditioning alone:

$$\begin{aligned} Ap(A)y &= b, \\ x &= p(A)y. \end{aligned}$$

With other preconditioners (e.g. ILU, Block Jacobi, ...) :

$$\begin{aligned} AMp(AM)y &= b, \\ x &= p(AM)y. \end{aligned}$$

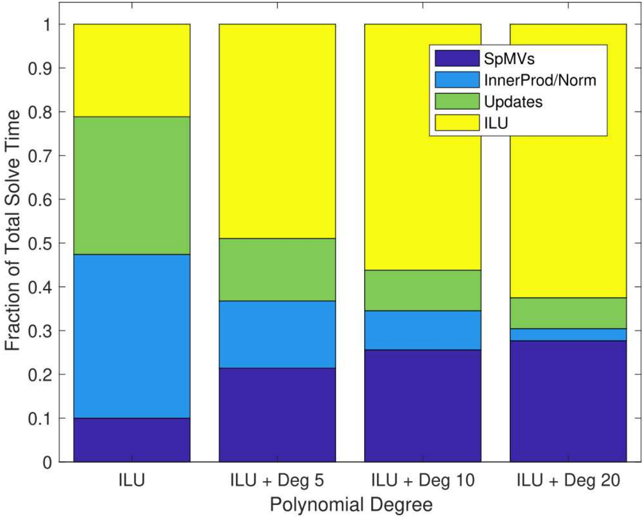
No extra work to code this in your Trilinos solver!
Just pass your preconditioner *M* to the linear problem like usual.

Example with ILU:

- Matrix: **Transport** (From SuiteSparse Janna collection)
- Problem: 3D finite element flow and transport
- Size: $n = 1{,}602{,}111$, nonsymmetric, NNZ: 23,487,281

	SPMVs	**Time**	**Iters**	**Add roots**
No Prec	40670	1042	40268	
Deg 20	6048	26.16	285	1
Deg 40	3948	15.19	93	2
Deg 60	4032	14.51	63	3
ILU Only	1898	55.9	1879	
ILU + Deg 5	1595	22.59	315	0
ILU + Deg 10	920	10.81	91	0
ILU + Deg 20	960	10.22	47	0

Solve time Distribution with ILU



(32 MPI processes over 1 node)

Preconditioner Generation Time

TODO update ILU timings!!
Over 32 MPI processes.
(Solve time does not include preconditioner setup time.)

	Prec Setup Time	Solve Time
ILU	0.2157	36.05
ILU + Deg 10	0.3334	6.66
ILU + Deg 20	0.4922	6.58
ILU + Deg 40	0.8659	6.27
Deg 20	0.1754	26.16
Deg 40	0.4926	15.19
Deg 80	1.655	14.62

What about Multigrid?

Can polynomial preconditioning help algebraic multigrid?

If it works well (e.g. GMRES converges in 4 iterations), probably not.

If multigrid struggles (convection-diffusion or Helmholtz?), then possibly.

Potential for Combining with Multigrid

Sandia National Laboratories

Matrix: 3D Laplacian from Galeri, $n = 15{,}625{,}000$

	Iters	Solve Time	Poly Create	Solve + Poly Create
AMG only	42	13.95		13.95
AMG + Deg 2	26	9.71	0.29	10.00
AMG + Deg 3	19	8.55	0.44	8.99
AMG + Deg 5	12	7.78	0.75	8.53
AMG + Deg 7	9	7.90	1.10	9.00
AMG + Deg 10	6	7.62	1.69	9.31
AMG + Deg 12	4	7.75	2.14	9.89

Multigrid: Smoothed aggregation (with Chebyshev smoothing) over 5 levels on 32 MPI processes

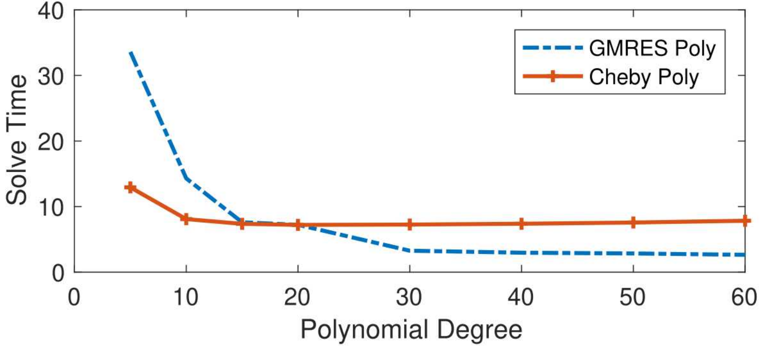
(Thanks to Christian Glusa for help running MueLu code!)

Polynomial Degree Starting Suggestions:

- **Poly Prec alone:** Degree ≥ 40
- **With ILU, Block Jacobi, Factorization-based preconditioning:** Degree between 5 and 30
- **With Multigrid:** Degree ≤ 15

What about Chebyshev Polynomials?

cfd2, GMRES(50), rtol = 1×10^{-8}, 32 MPI processes



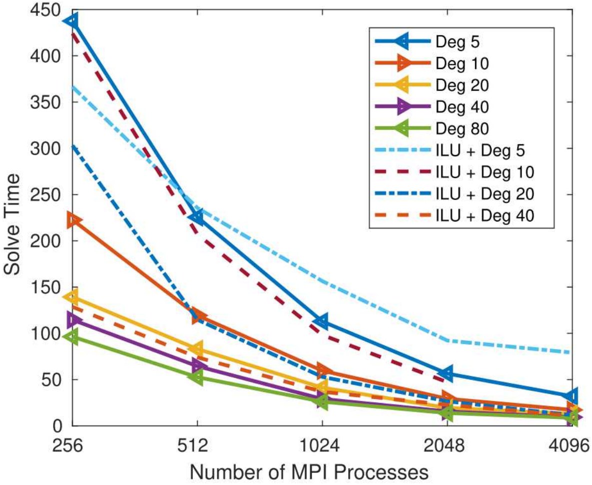
Caveat: Ifpack's Chebyshev preconditioner includes diagonal scaling. The GMRES polynomial does not.

Chebyshev: Min solve time: 7.21s, 1758 iterations, (deg=20)

GMRES: Min solve time: 2.65s, 225 iterations (deg=60)

Scaling it up

3D Laplacian, $n = 166$ million, 1.1 billion nonzero elements



Communication Avoiding S-Step GMRES

Sandia National Laboratories

Delayed Orthogonalization:
Can avoid dot products in GMRES by orthogonalizing every s steps:
E.g. $s = 3$:

$$\mathcal{K} = span\{b, Ab, A^2b, A^3b, A^4b, A^5b, A^6b, \ldots, A^{m-1}b\}$$

Use TSQR to orthogonalize the blocks.

Matrix Powers Kernel:

- Used for performing repeated matvecs with A.
- Minimizes the number of reads from slow memory and cache.

[Demmel, Hoemmen, et al.]

Combining with Communication-Avoiding Methods

Sandia National Laboratories

1. **Polynomial preconditioned standard GMRES.**
   - Use Matrix Powers Kernel (MPK) to evaluate the polynomial.
2. **Polynomial preconditioning within CA-GMRES.**
   - More SpMVs per orthogonalization.
3. **Polynomial Preconditioning for Pipelined Methods**
   - Use polynomial to create longer pipeline length for better stability.

Future Work:

- More applications
- Comparison/ combination with S-step GMRES?
- Larger-scale experiments: Lots of GPUs
- Use the polynomial as a smoother for multigrid?

Thank you!

Sandia National Laboratories

This research was funded in part by Sandia National Laboratories. 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-NA-0003525. This work was in part supported by the Department of Energy's Exascale Computing Project (ECP).