Iterative methods: from theory to practice
(A tutorial)

Yousef Saad, Ruipeng Li, Yuanzhe Xi

(Minnesota) (LLNL) (Emory)

Copper Mountain Conference on Iterative Methods

March 31, 2022

LLNL-PRES-833179

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344. Lawrence Livermore National Security, LLC

SOFTWARE, APPLICATIONS AND DEMOS

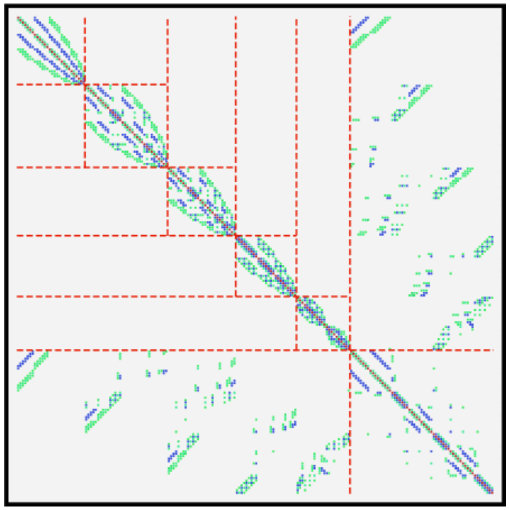
ITSOL and ZITSOL

➤ **Preconditioners for general sparse linear systems**

$$Ax = b, \; A \in \mathbb{R}^{n\times n}\text{(ITSOL)}, \; \mathbb{C}^{n\times n}\text{(ZITSOL)}$$

➤ **"Sequential" preconditioners (in v2.0)**

- **ILUK (ILU preconditioner with level of fill)**
- **ILUT (ILU preconditioner with threshold)**
- **ILUC (Crout version of ILUT)**
- **VBILU (Variable block ILU; Automatic block detection)**
- **ARMS (Algebraic Recursive Multilevel Solvers; Standard and ddPQ versions)**



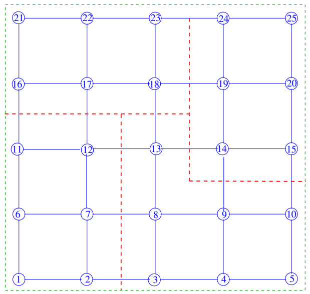
➤ **Demos**

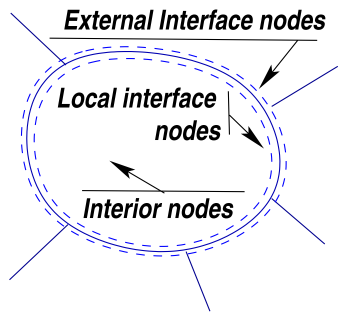
https://www-users.cse.umn.edu/~saad/software/ITSOL/index.html

pARMS: Parallel solvers for sparse linear systems

➤ **MPI-based with Domain Decomposition (DD)**

➤ **Also available from PETSc: `PCPARMS`. v2.2 and v3.2**



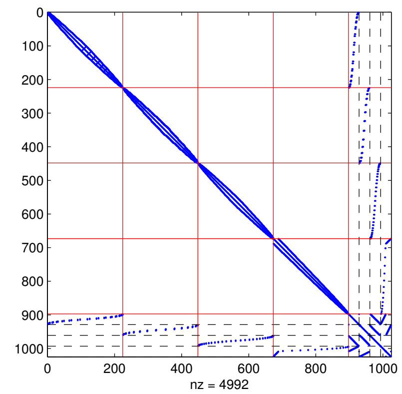


Local preconditioner

- **ILU0, ILUK, ILUT**
- **ARMS**

Global preconditioner

- **Restricted additive Schwarz (RAS)**
- **Block Jacobi (BJ)**
- **Schur complement**
- **Distributed ILU(0)/SSOR (v2.2)**



https://www-users.cse.umn.edu/~saad/software/pARMS/index.html

CUDA-ITSOL: ITSOL for single CUDA GPU

➤ **Sparse matrix kernels**

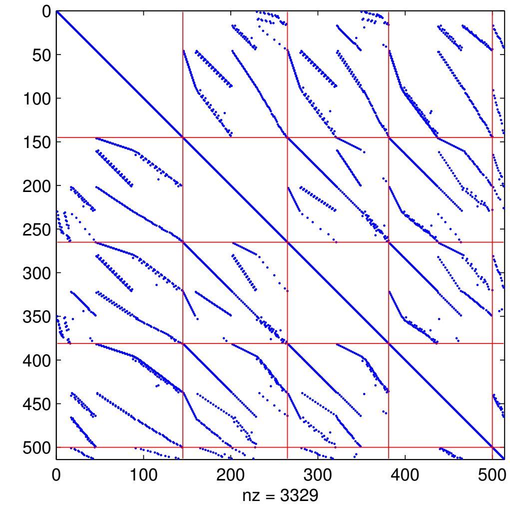
- **SpMV in DIA, ELL, JAD, CSR**
- **Level-scheduling SpTrSV**

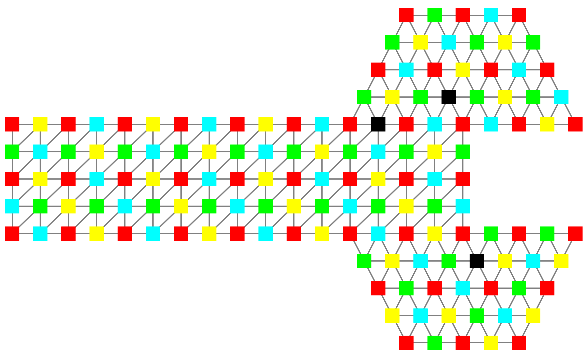
➤ **GPU-friendly preconditioners**

- **Block Jacobi ILU**
- **Multicolor SSOR/ILU(0)**
- **Least-squares polynomial**

➤ **Krylov subspace methods**

- **CG**
- **GMRES**





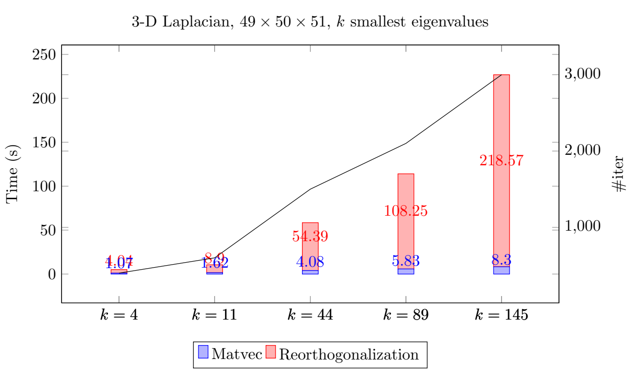
https://www-users.cse.umn.edu/~saad/software/

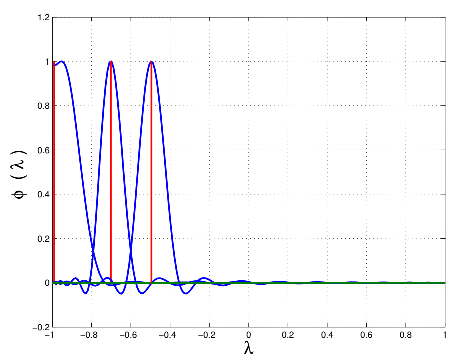
➤ EigenValues Slicing Library (v1.1.1)

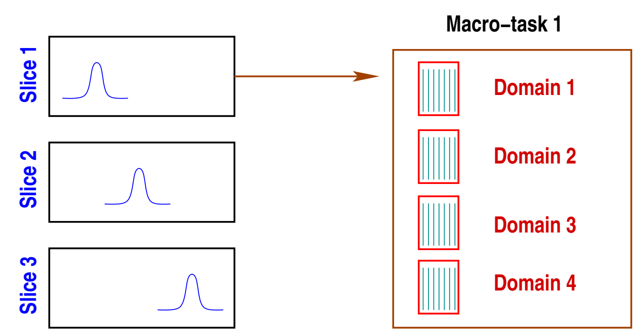
- Compute (interior) eigenvalues of (A, B), A is symmetric, B is SPD
- Spectral slicing by Kernel Polynomial Method or Stochastic Lanczos
  - Reduces memory cost for storing basis
  - Reduces orthogonalization cost
  - Enables parallelism

✍ 3-D Laplacian $n = 49^3$. Compute all the $1,971$ eigenvalues in $[0, 1]$

`eigs(A,1971,'sa')`: 4 hours; EVSL with 5 slices, less than 300 sec in total!

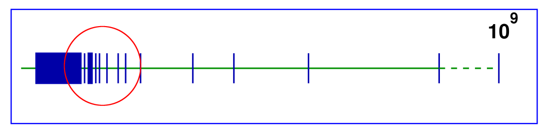


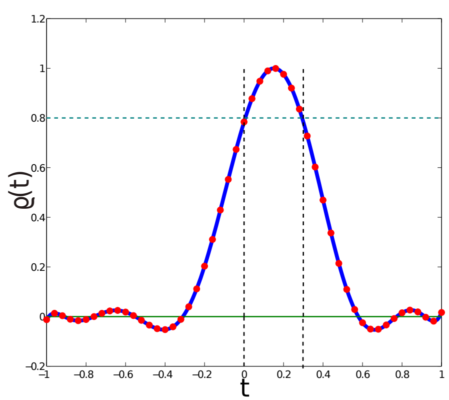




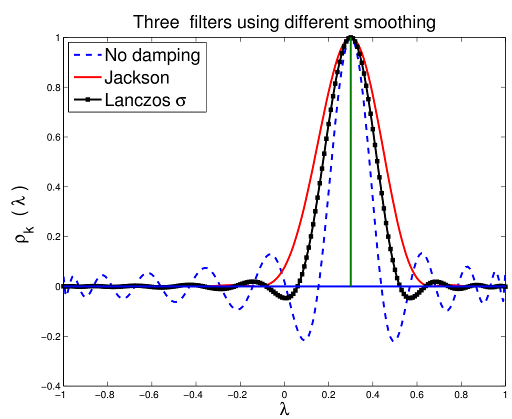
EVSL: Solvers for symmetric eigenvalue problems

- **Compute (interior) eigenvalues of (A, B), A is symmetric, B is SPD**
- **Polynomial and rational filtering**
  - **Extract eigenvalues at any location of the spectrum**
  - **Least-squares Chebyshev polynomial filtering**
    - **does not require expensive matrix factorizations**
  - **Least-squares rational filtering**
    - **handles stretched spectra better**

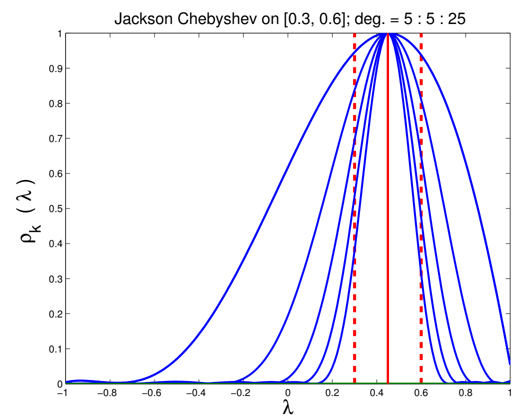




Expansion to $\delta(t - \gamma)$



Removing oscillations



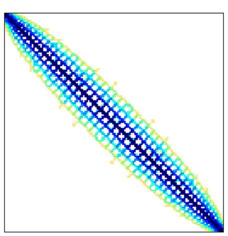
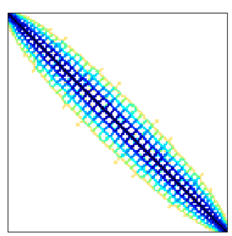
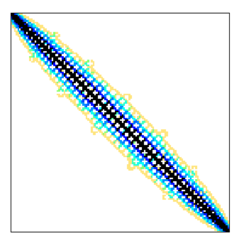
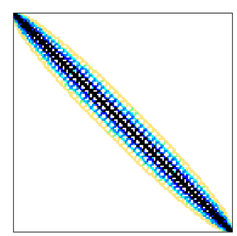
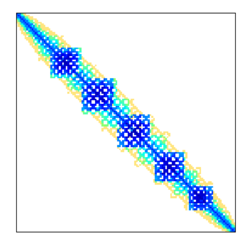
Automatically determine degrees

Application of EVSL: Kohn-Sham equation

- $\hat{H}\Psi = E\Psi$ n_0 is corresponding to the Fermi level
- SEVP: compute all the eigenpairs in $[0.5n_0, 1.5n_0]$

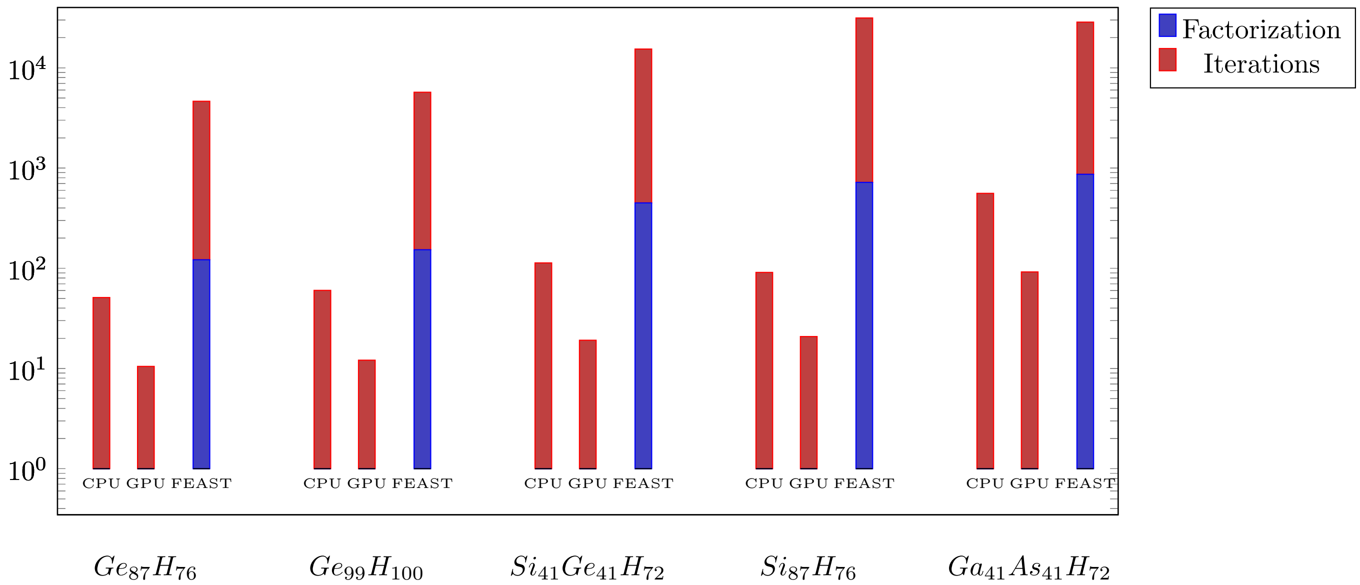
➤ SuiteSparse Matrix Collection: PARSEC

Hamiltonian	n	nnz	$[a, b]$	$[\xi, \eta]$	$\nu_{[\xi,\eta]}$
$Ge_{87}H_{76}$	$112,985$	$7,892,195$	$[-1.214, 32.764]$	$[-0.645, -0.0053]$	212
$Ge_{99}H_{100}$	$112,985$	$8,451,295$	$[-1.226, 32.703]$	$[-0.650, -0.0096]$	250
$Si_{41}Ge_{41}H_{72}$	$185,639$	$15,011,265$	$[-1.121, 49.818]$	$[-0.640, -0.0028]$	218
$Si_{87}H_{76}$	$240,369$	$10,661,631$	$[-1.196, 43.074]$	$[-0.660, -0.0300]$	213
$Ga_{41}As_{41}H_{72}$	$268,096$	$18,488,476$	$[-1.250, 1300.9]$	$[-0.640, -0.0000]$	201

Application of EVSL: Kohn-Sham equation

- **Shift-and-invert: extremely expensive LU factorization**
- **Polynomial filtering is much more efficient**
- **And can be accelerated by GPUs: 7x speedup**

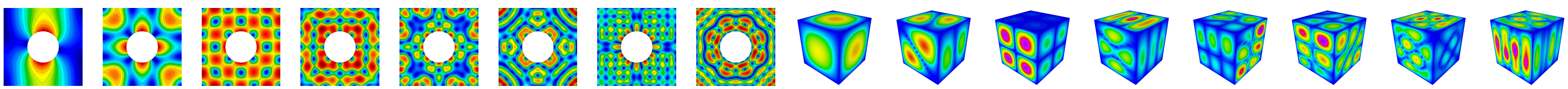


Intel Xeon E5-2695 CPU (24 cores) and NVIDIA P100 GPU

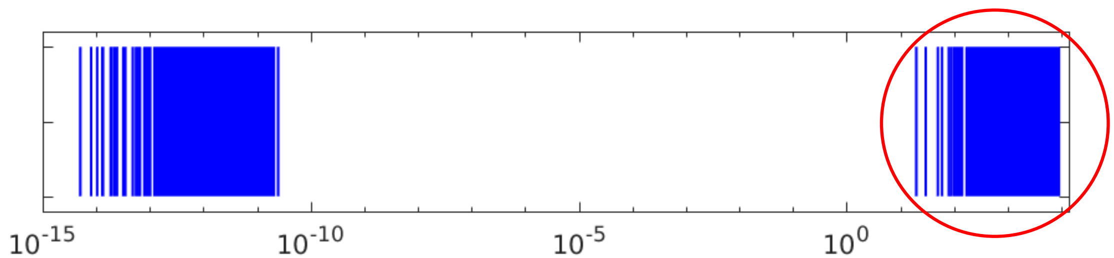
Application of EVSL: Maxwell eigenproblem

$$\nabla \times \nabla \times \vec{E} = \lambda \vec{E}, \ \nabla \cdot \vec{E} = 0 \text{ in } \Omega, \ \vec{E} \times \vec{n} = 0 \text{ on } \partial\Omega$$

- $\vec{E}$: electric field intensity. Discretized by 2nd order Nédélec FEM



- B^{-1} with simple iterative methods
- Interior GEVP: interested in nonzero eigenvalues



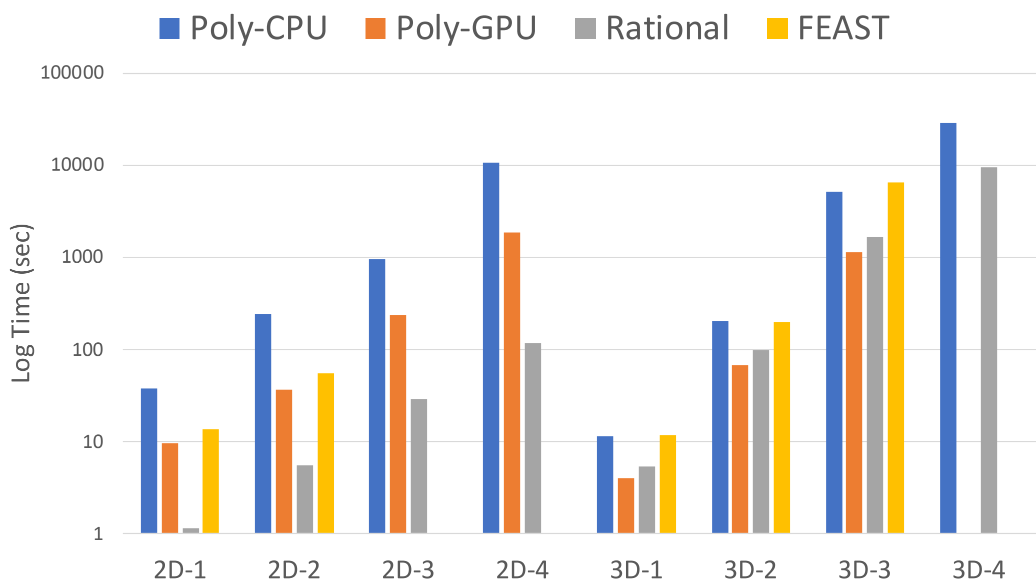
- Algebraic method: without knowing the discretized gradient

Application of EVSL: Maxwell eigenproblem

- **Rational filtering is more efficient for 2-D. Polynomial filtering is more efficient for 3-D with GPUs**

$$(\xi, \eta) = (19.5, 250)$$

Prob	n	(a, b)	$\nu_{[\xi,\eta]}$
3D-1	10,800	$(0, 9e3)$	115
3D-2	92,256	$(0, 3.7e4)$	121
3D-3	762,048	$(0, 1.5e5)$	121
3D-4	2,599,200	$(0, 3.3e5)$	121



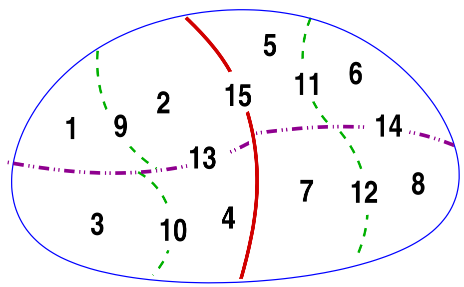
- **Rational filtering requires much more memory to store the factors; 68 GB for the 3D-4 problem**

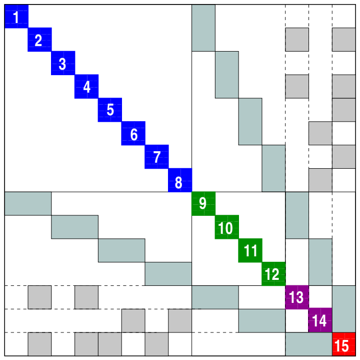
➤ **Demos**

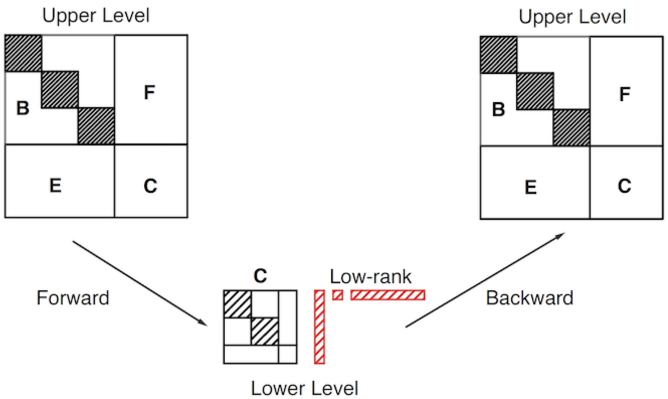
https://github.com/eigs/EVSL

ParGeMSLR: Generalized Multilevel Schur Low-Rank

- Parallel preconditioner for distributed linear systems
- Recursive multilevel DD with low-rank corrections
- $S^{-1} \approx C^{-1} +$ Low-Rank Correction
- Nonsymmetric systems and complex systems
- Fully parallel: MPI-based library with OpenMP and CUDA



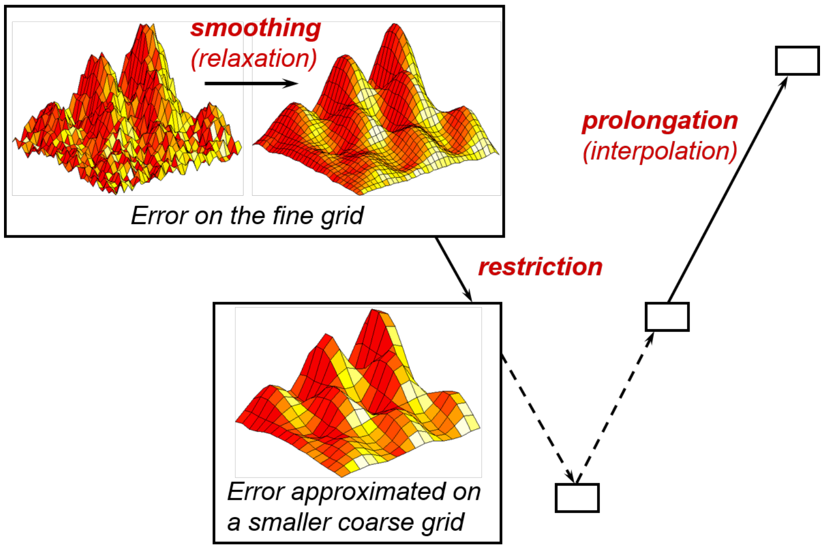




➤ **Demos** https://github.com/hitenze/pargemslr.git

BoomerAMG: Parallel Algebraic Multigrid Method

- Various parallel coarsening techniques, interpolation and relaxation schemes
- More advanced approaches to increase efficiency and scalability: aggressive coarsening, Non-Galerkin coarse-grid operator
- AMG for systems of PDEs
- Special additive V-cycles
- Full GPU-support
- Available in hypre



➤ Demos

https://github.com/hypre-space/hypre