Block Locally Optimal Preconditioned Eigenvalue Xolvers

BLOPEX

Ilya Lashuk, Merico Argentati,

Evgenii Ovtchinnikov, Andrew Knyazev (speaker)

Department of Mathematical Sciences and

Center for Computational Mathematics

University of Colorado at Denver and Health Sciences Center

Center for Computational Mathematics, University of Colorado at Denver

University of Colorado at **Denver**

Mile High Math

Abstract

$$T(A - \lambda B)x = 0$$

Block Locally Optimal Preconditioned Eigenvalue Xolvers (BLOPEX) is a package, written in C, that at present includes only one eigenxolver, Locally Optimal Block Preconditioned Conjugate Gradient Method (LOBPCG). BLOPEX supports parallel computations through an abstract layer. BLOPEX is incorporated in the HYPRE package from LLNL and is availabe as an external block to the PETSc package from ANL as well as a stand-alone serial library.

Center for Computational Mathematics, University of Colorado at Denver

Acknowledgements

Center for Computational Mathematics, University of Colorado at Denver

University of Colorado at **Denver**

Mile High Math

CONTENTS

1. Background concerning the LOBPCG algorithm

2. Hypre and PETSc software libraries

3. LOBPCG implementation strategy in BLOPEX

4. Testing

5. Scalability Results on Beowulf and BlueGene/L

6. Conclusions

Center for Computational Mathematics, University of Colorado at Denver

LOBPCG Background

Locally Optimal Block Preconditioned Conjugate Gradient Method

The algorithm is described in:

- A. V. Knyazev, Toward the Optimal Preconditioned Eigensolver: Locally Optimal Block Preconditioned Conjugate Gradient Method. SIAM Journal on Scientific Computing 23 (2001), no. 2, pp. 517-541.

- A. V. Knyazev, I. Lashuk, M. E. Argentati, and E. Ovchinnikov, Block Locally Optimal Preconditioned Eigenvalue Xolvers (BLOPEX) in hypre and PETSc. SISC, submitted (2007). Published as a technical report http://arxiv.org/abs/0705.2626.

Center for Computational Mathematics, University of Colorado at Denver

LOBPCG Features

- LOBPCG solver finds the smallest eigenpairs of a symmetric generalized definite eigenvalue problem using preconditioning directly.

- For computing only the smallest eigenpair, the algorithm LOPCG (Block size $= 1$) implements a local optimization of a 3-term recurrence.

- For finding m smallest eigenpairs the Rayleigh-Ritz method on a $3m$–dimensional trial subspace is used during each iteration for the local optimization. Cluster robust, does not miss multiple eigenvalues!

- The algorithm is matrix free since the multiplication of a vector by the matrices of the eigenproblem and an application of the preconditioner to a vector are needed only as functions.

Center for Computational Mathematics, University of Colorado at Denver

What is **LOBPCG** for $Ax = \lambda Bx$?

The method combines *robustness and simplicity* of the steepest descent method with a *three-term recurrence* formula:

$$x^{(i+1)} = w^{(i)} + \tau^{(i)} x^{(i)} + \gamma^{(i)} x^{(i-1)},$$

$$w^{(i)} = T(Ax^{(i)} - \lambda^{(i)} Bx^{(i)}), \ \lambda^{(i)} = \lambda(x^{(i)}) = (x^{(i)}, Ax^{(i)})/(Bx^{(i)}, x^{(i)})$$

with properly chosen scalar iteration parameters $\tau^{(i)}$ and $\gamma^{(i)}$. The easiest and most efficient choice of parameters is based on an idea of *local optimality* Knyazev 1986, namely, select $\tau^{(i)}$ and $\gamma^{(i)}$ that minimize the Rayleigh quotient $\lambda(x^{(i+1)})$ by using the *Rayleigh–Ritz method*.

Three-term recurrence + Rayleigh–Ritz method =

Locally Optimal Conjugate Gradient Method

Center for Computational Mathematics, University of Colorado at Denver

Currently available **LOBPCG** software by others

- Earth Simulator CDIR/MPI (Yamada et al., Fermion-Habbard Model)

- SLEPc interface to Hypre LOBPCG (Jose Roman, SLEPc)

- C++ (Rich Lehoucq and Ulrich Hetmaniuk, Anasazi Trilinos)

- C (A. Stathopoulos, PRIMME, real and complex Hermitian)

- Fortran 77 (Randolph Bank, PLTMG)

- Python (Peter Arbenz and Roman Geus, PYFEMax)

- C++ (Sabine Zaglmayr and Joachim Schberl, NGSolve)

- Fortran 90 (Gilles Zèrah, ABINIT, complex Hermitian)

- Fortran 90 (S. Tomov and J. Langou, PESCAN, complex Hermitian)

- (A. Borzì and G. Borzì, AMG)

University of Colorado at **Denver**

Mile High Math

Portable, Extensible Toolkit for Scientific Computation (PETSc) and High Performance Preconditioners (Hypre)

- Software libraries for solving large systems on massively parallel computers

- The libraries are designed to provide robustnesss, ease of use, flexibility and interoperability.

- The primary goal of Hypre is to provide users with advanced high-quality parallel preconditioners for linear systems.

- The primary goal of PETSc is to facilitate the integration of independently developed application modules with strict attention to component interoperability.

Center for Computational Mathematics, University of Colorado at Denver

BLOPEX serial/Hypre/PETSc Implementation

- Abstract matrix- and vector-free implementation in C-language

- Hypre/PETSc and LAPACK libraries

- User-provided functions for matrix-vector multiply and preconditioner

- LOBPCG implementation utilizes Hypre/PETSc parallel vector manipulation routines

Advantages of native Hypre/PETSc implementations of LOBPCG:

- A native Hypre LOBPCG version efficiently takes advantage of powerfull Hypre algebraic and geometric multigrid preconditioners

- A native PETSc LOBPCG version gives the PETSc users community an easy access to a customizable code of the high quality modern preconditioned eigensolver and an opportunity to easily call Hypre preconditioners from PETSc

BLOPEX Implementation Using Hypre/PETSc

Abstract LOBPCG in C

Interface PETSc-BLOPEX

Interface Hypre-BLOPEX

PETSc driver for LOBPCG

Hypre driver for LOBPCG

PETSc libraries

Hypre libraries

Center for Computational Mathematics, University of Colorado at Denver

# Domain Decomposition and Multilevel Preconditioners Tested with LOBPCG

Hypre Implementation:

- PFMG-PCG: geometric multigrid called directly or through PCG

- AMG-PCG: algebraic multigrid called directly or through PCG

- Schwarz-PCG: additive Schwarz called directly or through PCG

PETSc Implementation:

- Additive Schwarz called directly or through PCG

- Algebraic preconditioners from the Hypre package

Center for Computational Mathematics, University of Colorado at Denver

LOBPCG Performance vs Preconditioner Iterations
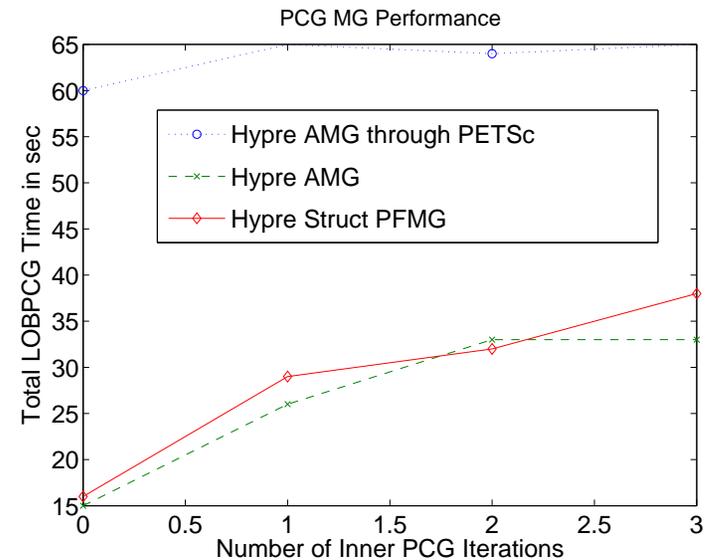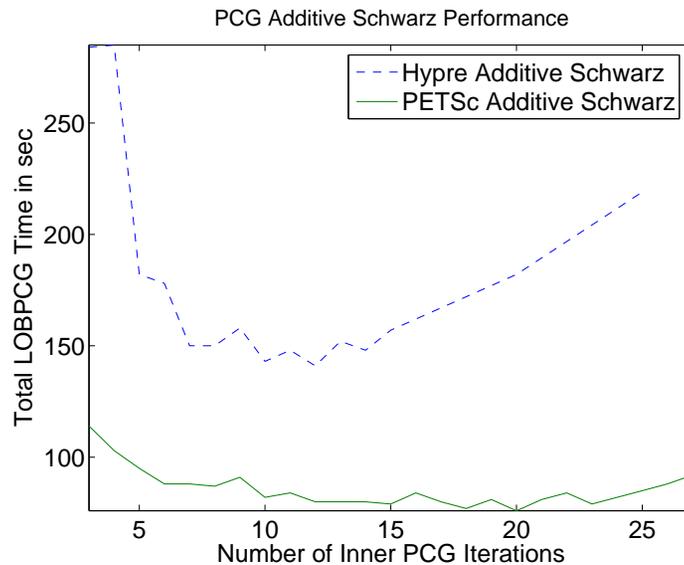
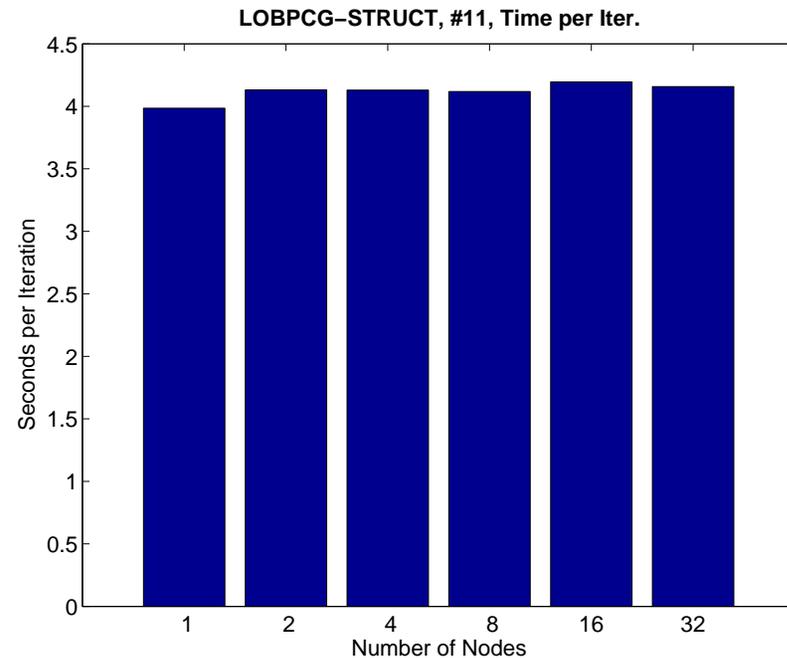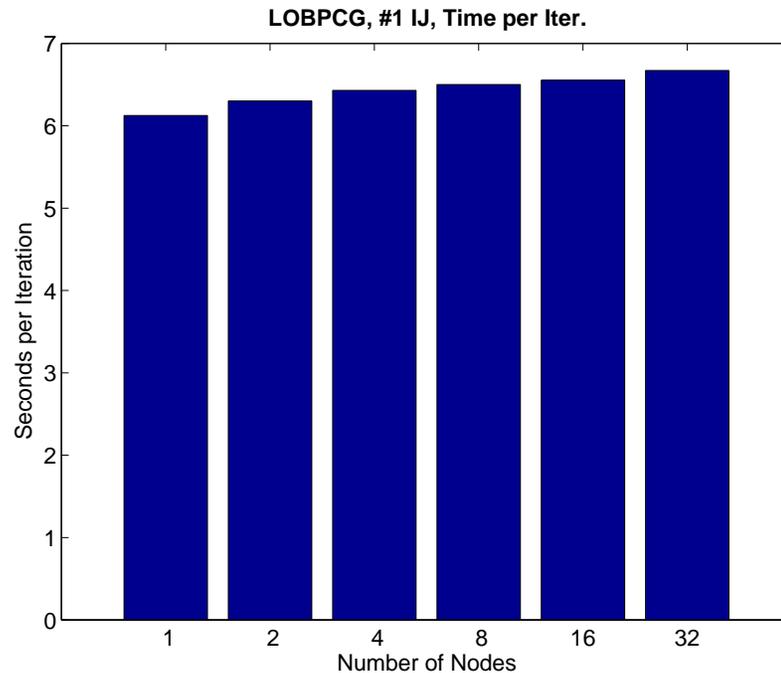

7–Point 3-D Laplacian, 1,000,000 unknowns. 1 MCR node (two 2.4-GHz Pentium 4 Xeon processors and 4 GB of memory).

Center for Computational Mathematics, University of Colorado at Denver

LOBPCG Performance vs. Block Size
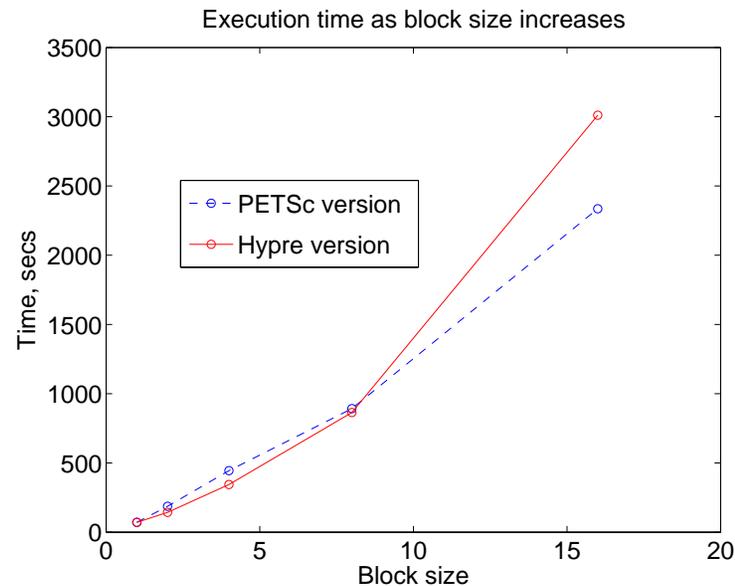
7–Point 3-D Laplacian, 2,000,000 unknowns. Preconditioner: AMG.

System: Sun Fire 880, 6 CPU.

Center for Computational Mathematics, University of Colorado at Denver

University of Colorado at **Denver**
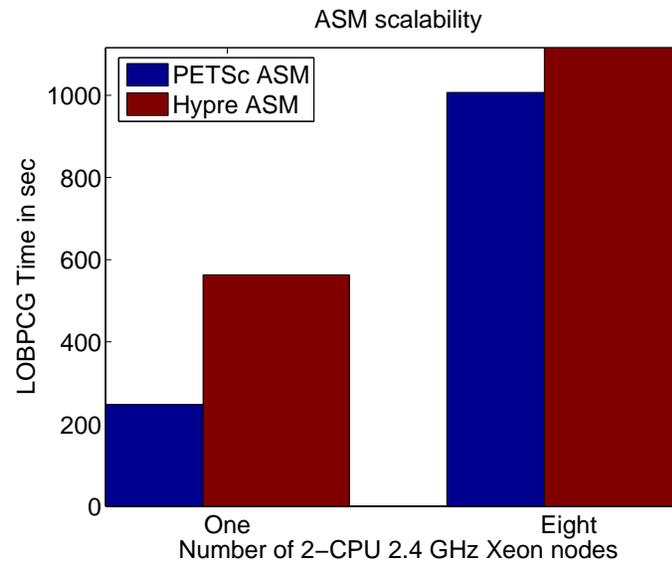**Mile High Math**

LOBPCG Scalability

Hypre, 7–Point Laplacian, 1,000,000 unknowns per node. Preconditioner: AMG. System: Beowulf (36 dual P3 1GHz 2GB nodes)

Center for Computational Mathematics, University of Colorado at Denver

LOBPCG Scalability

7–Point Laplacian, 2,000,000 unknowns per node. Preconditioner: ASM. System: LLNL MCR, cluster of dual Pentium 4 Xeon (2.4-GHz, 4 GB) nodes.
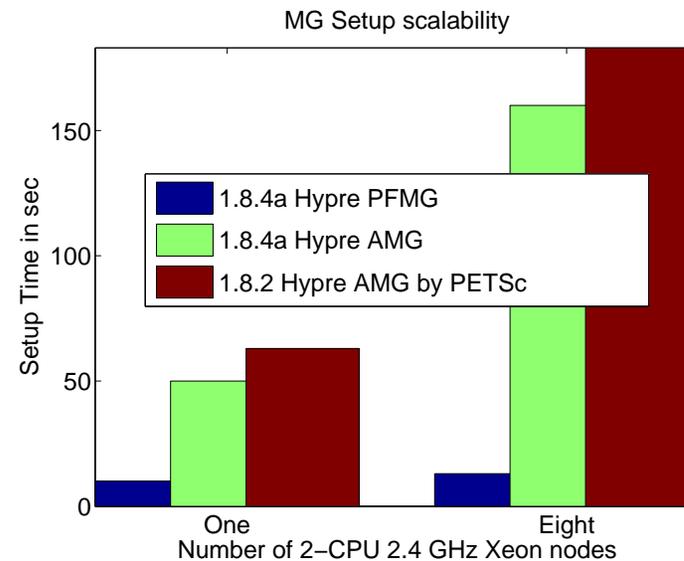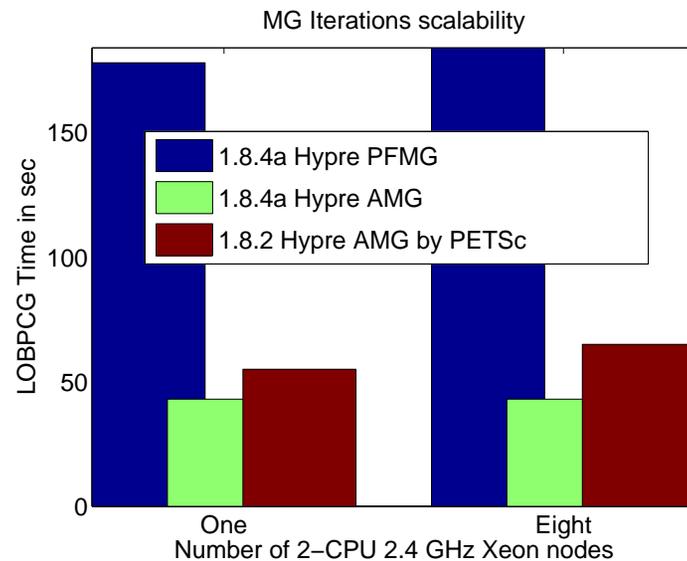
Center for Computational Mathematics, University of Colorado at Denver

LOBPCG Scalability

7–Point Laplacian, 2,000,000 unknowns per node. Preconditioners: AMG, PFMG. System: LLNL MCR, cluster of dual Pentium 4 Xeon (2.4-GHz, 4 GB) nodes.

Center for Computational Mathematics, University of Colorado at Denver

University of Colorado at Denver — Mile High Math

Scalability of BLOPEX-AMG on IBM BlueGene/L

N Proc	N Iter.	Prec. setup (sec)	Apply Prec. (sec)	Lin. Alg. (sec)
32	12	66	30	6
64	14	32	18	3
128	12	18	8	1
256	12	10	4	0.5
512	21	5.4	4	0.5
1024	13	4	2	0.2

TABLE 1: **Scalability Data for 24 megapixel image segmentation**

BLOPEX in PETSc using Hypre AMG, Block size: 1, LOBPCG tolerance: 10^{-6}.

NCAR's single-rack Blue Gene/L with 1024 compute nodes, organized in 32 I/O nodes with 32 compute nodes each. One node is a dual-core chip, containing two 700MHz PowerPC-440 CPUs and 512MB of memory. We run 1 CPU per node.

Center for Computational Mathematics, University of Colorado at Denver

Scalability of BLOPEX-SMG on IBM BlueGene/L

N Proc	Matrix Size	N Iter.	Prec. Setup (sec)	Solve (sec)
8	4.096 M	10	7	74
64	32.768 M	8	11	67
512	0.262144 B	7	19	61

TABLE 2: *Scalability for 3D Laplacian $80 \times 80 \times 80 = 512,000$ mesh per CPU*

BLOPEX in Hypre struct with SMG, Block size: 1, LOBPCG tolerance: 10^{-8}.
Uniform cube partitioning. 1 CPU per node.

Center for Computational Mathematics, University of Colorado at Denver

Scalability of BLOPEX-SMG on IBM BlueGene/L

N Proc	Matrix Size	N Iter.	Prec. Setup (sec)	Solve (sec)
8	1 M	21	2	2168
64	8 M	19	5	2360
512	64 M	18	13	2205

TABLE 3: *Scalability for 3D Laplacian* $50 \times 50 \times 50 = 125,000$ *mesh per CPU*

BLOPEX in Hypre struct with SMG, Block size: 50, LOBPCG tolerance: 10^{-4}.
Uniform cube partitioning. 1 CPU per node.

Conclusions

- BLOPEX is the only currently available package that solves eigenvalue problems using Hypre and PETSc preconditioners

- Our abstract C implementation of the LOBPCG in BLOPEX allows easy deployment with different software packages

- User interface routines

  - are easy to use

  - are based on Hypre/PETSc standard interfaces

  - give user an opportunity to provide matrix-vector multiply and preconditioned solver functions

- Initial scalability results look promising

Center for Computational Mathematics, University of Colorado at Denver