SIAM SIAG on Supercomputing track

- MS7/16: New Algorithms for Scientific Computing at Exascale
  Part 2: Mon 4:00 PM - 6:00 PM
- MS36/24 HPC and Data Science in Molecular Engineering (Tue AM, Tue PM)
- MS40/51 Resilient Computation in Large Scale Scientific Computing (Tue PM, Wed AM)
- MS58/69 High Performance Tensor Computations (Wed PM, Th AM)
- MS76/93 Communication-Avoiding Algorithms (Th PM, Fri PM)
- MS87/98 Parallel-in-time integration of differential equations (Th AM, Fr AM)

Topical speaker: Jacqueline Chen, SNL, Friday AM

NLAFET: Parallel Numerical Linear Algebra for Future Extreme Scale Systems

L. Grigori and the NLAFET Consortium Team
Umeå University, Inria, STFC, and University of Manchester

July 2017

Towards Exascale High Performance Computing

Aim of the Horizon 2020 FETHPC call:

Attract projects that can achieve world-class extreme scale computing capabilities in platforms, technologies and applications

19 Research and Innovation Actions (RIA); 2 Coordination and Support Actions (CSA) Among those:

- Mathematics: NLAFET, ExaHYPE
- Algorithms: ExaFLOW, ExCAPE, ComPat, NLAFET, ExaHYPE

Members of the NLAFET Consortium

- Umeå University, Sweden (UMU; *Coordinator* Bo Kågström; Lennart Edblom)

- The University of Manchester, UK (UNIMAN; Jack Dongarra; Nick Higham)

- Institute National de Recherche en Informatique et en Automatique, France (INRIA; Laura Grigori)

- Science and Technology Facilities Council, UK (STFC; Iain Duff)

NLAFET
Parallel Numerical Linear Algebra
for Future Extreme Scale Systems

Collaborating partners:

- Innovative Computing Laboratory (ICL), UT Knoxville
- James Demmel, UC Berkeley
- Individual researchers (academia and vendors)

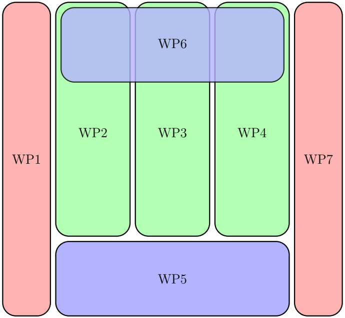
Key European players with recognized leadership, proven expertise, experience, and skills across the scientific areas of NLAFET!

NLAFET—Aim and Main Research Objectives

Aim: *Enable a radical improvement in the performance and scalability of a wide range of real-world applications relying on linear algebra software for future extreme-scale systems.*

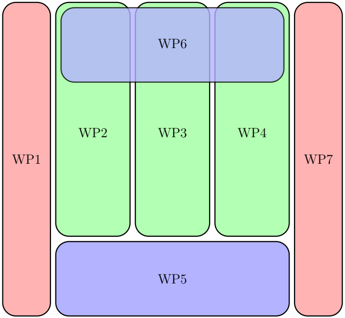
- Development of novel *architecture-aware algorithms* that expose as much parallelism as possible, exploit heterogeneity, avoid communication bottlenecks, respond to escalating fault rates, and help meet emerging power constraints
- Exploration of *advanced scheduling strategies and runtime systems* focusing on the extreme scale and strong scalability in multi/many-core and hybrid environments
- Design and evaluation of novel strategies and software support for both *offline and online auto-tuning*
- Results will appear in the open source *NLAFET software library*

NLAFET Work Package Overview



- WP1: *Management and coordination* (start **M1**)
- WP5: *Challenging applications—a selection* (start M13)
  Materials science, power systems, study of energy solutions, and data analysis in astrophysics
- WP7: *Dissemination and community outreach* (start **M1**)
  Research and validation results; stakeholder communities

Research focus—Critical set of NLA operations



- WP2: *Dense linear systems and eigenvalue problem solvers*
- WP3: *Direct solution of sparse linear systems*
- WP4: *Communication-optimal algorithms for iterative methods*
- WP6: *Cross-cutting issues* (All four start **M1**)

WP2, WP3 and WP4: research into extreme-scale parallel algorithms
WP6: research into methods for solving common cross-cutting issues

WP2, WP3 and WP4 at a glance!

- Linear Systems Solvers
- Hybrid (Batched) BLAS
- Eigenvalue Problem Solvers
- Singular Value Decomposition Algorithms

- Lower Bounds on Communication for Sparse Matrices
- Direct Methods for (Near–)Symmetric Systems
- Direct Methods for Highly Unsymmetric Systems
- Hybrid Direct–Iterative Methods

- Computational Kernels for Preconditioned Iterative Methods
- Iterative Methods: use t vectors per it, nearest-neighbor comm
- Preconditioners: multi-level, communication reducing

Krylov subspace methods

Solve $Ax = b$ by finding a sequence $x_1, x_2, ..., x_k$ that minimizes some measure of error over the corresponding spaces

$$x_0 + \mathcal{K}_i(A, r_0), \quad i = 1, ..., k$$

.

They are defined by two conditions:

1. Subspace condition: $x_k \in x_0 + \mathcal{K}_k(A, r_0)$
2. Petrov-Galerkin condition: $r_k \perp \mathscr{L}_k$

$$\iff (r_k)^t y = 0, \ \forall \ y \in \mathscr{L}_k$$

where

- x_0 is the initial iterate, r_0 is the initial residual,
- $\mathcal{K}_k(A, r_0) = span\{r_0, Ar_0, A^2 r_0, ..., A^{k-1} r_0\}$ is the Krylov subspace of dimension k,
- $\mathscr{L}_k$ is a well-defined subspace of dimension k.

Challenge in getting efficient and scalable solvers

- Solve $Ax = b$ by using a Krylov subspace method:
  find x_k from $x_0 + \mathcal{K}_k(A, r_0)$ where

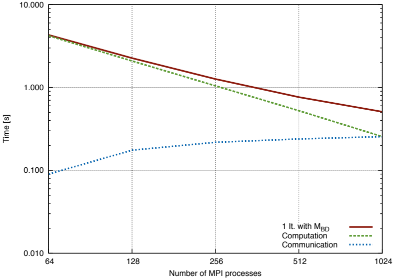
$$\mathcal{K}_k(A, r_0) = span\{r_0, Ar_0, A^2 r_0, ..., A^{k-1} r_0\},$$

such that the Petrov-Galerkin condition $b - Ax_k \perp \mathscr{L}_k$ is satisfied.

Typically, each iteration requires

- Sparse matrix vector product
  → point-to-point communication
- Dot products for orthogonalization
  → global communication

Map making on Cray XE6



Challenge in getting efficient and scalable solvers

- Solve $Ax = b$ by using a Krylov subspace method:
  find x_k from $x_0 + \mathcal{K}_k(A, r_0)$ where

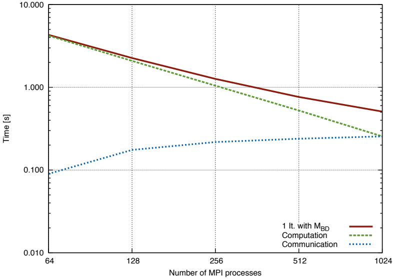
  $$\mathcal{K}_k(A, r_0) = span\{r_0, Ar_0, A^2 r_0, ..., A^{k-1} r_0\},$$

  such that the Petrov-Galerkin condition $b - Ax_k \perp \mathscr{L}_k$ is satisfied.

Typically, each iteration requires

- Sparse matrix vector product
  → point-to-point communication
- Dot products for orthogonalization
  → global communication

Map making on Cray XE6



Ways to improve performance

Typical approaches

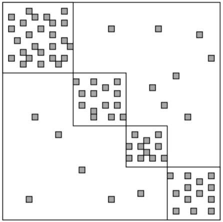
- Improve the performance of sparse matrix-vector product
- Improve the performance of collective communication

NLAFET approach

- Change numerics - reformulate or introduce Krylov subspace algorithms to:
  - reduce communication
  - increase arithmetic intensity - compute sparse matrix-set of vectors product
- Use preconditioners to decrease the number of iterations till convergence

Enlarged Krylov methods

- Partition the matrix into t domains
- Split the residual r_0 into t vectors corresponding to the t domains,



$$r_0 \rightarrow T(r_0) = \begin{bmatrix} * & 0 & & 0 \\ \vdots & \vdots & & \vdots \\ * & 0 & & 0 \\ 0 & * & & 0 \\ \vdots & \vdots & & \vdots \\ 0 & * & & 0 \\ & & \ddots & \\ 0 & 0 & & * \\ \vdots & \vdots & & \vdots \\ 0 & 0 & & * \end{bmatrix}$$

- Generate t new basis vectors, obtain an enlarged Krylov subspace

$$\mathcal{K}_{t,k}(A, r_0) = \text{span}\{T(r_0), AT(r_0), A^2T(r0), ..., A^{k-1}T(r_0)\}$$

- Search for the solution of $Ax = b$ in $\mathcal{K}_{t,k}(A, r_0)$
- [Grigori and Moufawad, 2014, Grigori and Tissot, 2017, Daas et al., 2017]

Properties of enlarged Krylov subspaces

- The Krylov subspace $\mathcal{K}_k(A, r_0)$ is a subset of the enlarged one

$$\mathcal{K}_k(A, r_0) \subset \mathcal{K}_{t,k}(A, r_0)$$

- For all $k < k_{max}$ the dimensions of $\mathcal{K}_{t,k}$ and $\mathcal{K}_{t,k+1}$ are stricltly increasing by some number i_k and i_{k+1} respectively, where

$$t \geq i_k \geq i_{k+1} \geq 1.$$

- The enlarged subspaces are increasing subspaces, yet bounded.

$$\mathcal{K}_{t,1}(A, r_0) \subsetneq ... \subsetneq \mathcal{K}_{t,k_{max}-1}(A, r_0) \subsetneq \mathcal{K}_{t,k_{max}}(A, r_0) = \mathcal{K}_{t,k_{max}+q}(A, r_0), \forall q > 0$$

- The solution of the system $Ax = b$ belongs to the subspace $x_0 + \mathcal{K}_{t,k_{max}}$.

Enlarged Krylov subspace methods based on CG

Defined by the subspace $\mathcal{K}_{t,k}$ and the following two conditions:

1. Subspace condition: $x_k \in x_0 + \mathcal{K}_{t,k}$
2. Orthogonality condition: $r_k \perp \mathcal{K}_{t,k}$

- At each iteration, the new approximate solution x_k is found by minimizing $\phi(x) = \frac{1}{2}(x)^t Ax - b^t x$ over $x_0 + \mathcal{K}_{t,k}$:

$$\phi(x_k) = min\{\phi(x), \forall x \in x_0 + \mathcal{K}_{t,k}(A, r_0)\}$$

- Can be seen as a particular case of a block Krylov method:
  - $AX = S(b)$ such that $S(b) \cdot ones(t, 1) = b$, $R_0 = AX_0 - S(b)$
  - Orthogonality condition involves the block residual $R_k \perp \mathcal{K}_{t,k}$

Convergence analysis

Given

- A is an SPD matrix, x^* is the solution of $Ax = b$
- $||\overline{e}_k||_A = ||x^* - \overline{x}_k||_A$ is the k^{th} error of CG
- $||e_k||_A = ||x^* - x_k||_A$ is the k^{th} error of enlarged methods
- CG converges in $\overline{K}$ iterations

Result

Enlarged Krylov methods converge in K iterations, where $K \leq \overline{K} \leq n$.

$$||e_k||_A = ||x^* - x_k||_A \leq ||\overline{e}_k||_A$$

Classical CG *vs.* Enlarged CG

Algorithm 1 Classic CG

```
$r_0 = b - Ax_0$
$p_1 = \frac{r_0}{\sqrt{r_0^t A r_0}}$
while $||r_{k-1}||_2 > \varepsilon ||b||_2$ do
    $\alpha_k = p_k^t r_{k-1}$
    $x_k = x_{k-1} + p_k \alpha_k$
    $r_k = r_{k-1} - A p_k \alpha_k$
    $p_{k+1} = r_k - p_k (p_k^t A r_k)$
    $p_{k+1} = \frac{p_{k+1}}{\sqrt{p_{k+1}^t A p_{k+1}}}$
end while
```

Algorithm 2 ECG(Odir)

```
$R_0 = T(b - Ax_0)$
$P_1 = \text{A-orthonormalize}(R_0)$
while $|| \sum_{i=1}^{t} R_k^{(i)} ||_2 < \varepsilon ||b||_2$ do
    $\alpha_k = P_k^t R_{k-1}$                                ▷ $t \times t$
    $X_k = X_{k-1} + P_k \alpha_k$                            ▷ $n \times t$
    $R_k = R_{k-1} - A P_k \alpha_k$                          ▷ $n \times t$
    $P_{k+1} = A P_k - P_k (P_k^t A A P_k) -$
        $P_{k-1} (P_{k-1}^t A A P_k)$                         ▷ $n \times t$
    $P_{k+1} = \text{A-orthonormalize}(P_{k+1})$
end while
$x = \sum_{i=1}^{t} X_k^{(i)}$                                ▷ $n \times 1$
```

- Enlarged CG based on Orthodir (Lanczos formula) [Ashby et al., 1990]
- More stable than Orthomin [OLeary., 1980], $P_{k+1} = R_k - P_k (P_k^t A R_k)$.

Classical CG *vs.* Enlarged CG

Algorithm 3 Classic CG

```
$r_0 = b - Ax_0$
$p_1 = \frac{r_0}{\sqrt{r_0^t A r_0}}$
while $||r_{k-1}||_2 > \varepsilon ||b||_2$ do
    $\alpha_k = p_k^t r_{k-1}$
    $x_k = x_{k-1} + p_k \alpha_k$
    $r_k = r_{k-1} - A p_k \alpha_k$
    $p_{k+1} = r_k - p_k (p_k^t A r_k)$
    $p_{k+1} = \frac{p_{k+1}}{\sqrt{p_{k+1}^t A p_{k+1}}}$
end while
```

Algorithm 4 ECG(Odir)

```
$R_0 = T(b - Ax_0)$
$P_1 =$ A-orthonormalize($R_0$)
while $|| \sum_{i=1}^{t} R_k^{(i)} ||_2 < \varepsilon ||b||_2$ do
    $\alpha_k = P_k^t R_{k-1}$                                  ▷ $t \times t$
    $X_k = X_{k-1} + P_k \alpha_k$                              ▷ $n \times t$
    $R_k = R_{k-1} - A P_k \alpha_k$                            ▷ $n \times t$
    $P_{k+1} = AP_k - P_k (P_k^t AAP_k) -$
        $P_{k-1} (P_{k-1}^t AAP_k)$                             ▷ $n \times t$
    $P_{k+1} =$ A-orthonormalize($P_{k+1}$)
end while
$x = \sum_{i=1}^{t} X_k^{(i)}$                                  ▷ $n \times 1$
```

BLAS 1&2 operations

# messages per iteration

O(log P) from dot prod + norm +

O(1) from SpMV

BLAS 3 operations

# messages per iteration

O(log P) from BCGS + A-ortho +

O(1) from SpMsetV

Dynamic reduction of search directions

- If R_{k-1} becomes rank deficient,

$$\begin{aligned} R_{k-1}v = 0 \quad &\Longrightarrow \quad R_i v = 0, \forall i \geq k \\ &\Longrightarrow \quad X_{i-1}v = X^* v \end{aligned}$$

- Monitor the rank of α_k instead of R_{k-1} to select search directions

$$\alpha_k = P_k^T R_{k-1} \approx U_k^{(1)} \sigma_k^{(1)} W_k^{(1)}$$

- The new search directions are given by the relation:

$$\begin{aligned} X_k &= X_{k-1} + P_k \alpha_k = X_{k-1} + (P_k U_k^{(1)})(\Sigma_k^{(1)} V_k^{(1)^t}) \\ &= X_{k-1} + P_k^{(1)} \alpha_k^{(1)}, \\ & \quad X_k, R_k \in \mathbb{R}^{n \times t}, \alpha_k^{(1)} \in \mathbb{R}^{\mathrm{rank}(\alpha_k) \times t}, P_k^{(1)} \in \mathbb{R}^{n \times \mathrm{rank}(\alpha_k)} \end{aligned}$$

- Idea adapted from Robbé and Sadkane (2006)

Related work

- Block Krylov methods (O'Leary 1980): solve systems with multiple rhs

  $$AX = B,$$

  by searching for an approximate solution $X_k \in X_0 + \mathcal{K}_k(A, R_0)$,

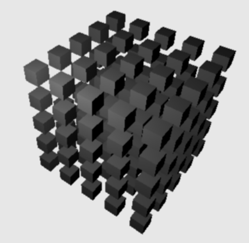
  $$\mathcal{K}_k(A, R_0) = block - span\{R_0, AR_0, A^2R_0, ..., A^{k-1}R_0\}$$

- BRRHS-CG (Nikishin and Yeremin, 1995) use a block method with $t - 1$ random right hand sides
- coopCG (Bhaya et al, 2012): solve one system by starting with t different initial guesses

- GMRES with multiple preconditioners (Greif, Rees, Szyld, 2011)
- AMPFETI (D. Rixen 97, Gosselet et al, 2015)

Test cases: boundary value problem

3D Skyscraper Problem - SKY3D



$$
\begin{aligned}
-div(\kappa(x)\nabla u) &= f \ in \ \Omega \\
u &= 0 \ on \ \partial\Omega_D \\
\frac{\partial u}{\partial n} &= 0 \ on \ \partial\Omega_N
\end{aligned}
$$

discretized on a 3D grid , where

$$
\kappa(x) = \begin{cases} 10^3 * ([10 * x_2] + 1), if \ [10 * x_i] = 0 mod(2), \ i = 1, 2, 3, \\ 1, \qquad\qquad\qquad\qquad\qquad\qquad otherwise. \end{cases}
$$

3D Anisotropic layers - ANI3D

- Ω divided into 10 layers parallel to $z = 0$, of size 0.1
- in each layer, the coefficients are constants (κ_x equal to 1, 10^2 or 10^4, $\kappa_y = 10\kappa_x$, $\kappa_z = 1000\kappa_x$).

Test cases (contd)

Linear elasticity 3D problem

$$\begin{aligned} \operatorname{div}(\sigma(u)) + f &= 0 && \text{on } \Omega, \\ u &= u_D && \text{on } \partial\Omega_D, \\ \sigma(u) \cdot n &= g && \text{on } \partial\Omega_N, \end{aligned}$$

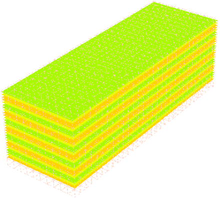


Figure : The distribution of Young's modulus

- $u \in \mathbb{R}^d$ is the unknown displacement field, f is some body force.
- Young's modulus E and Poisson's ratio ν take two values, $(E_1, \nu_1) = (2 \cdot 10^{11}, 0.25)$, and $(E_2, \nu_2) = (10^7, 0.45)$.
- Cauchy stress tensor $\sigma(u)$ is given by Hooke's law, defined by E and ν.

Enlarged CG: numerical results

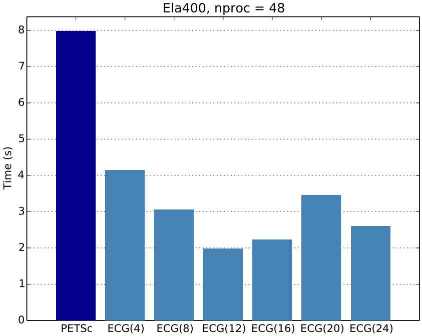
- Block Jacobi preconditioner (1024 blocks)
- Stopping criterion 10^{-6}
- Initial block size 32
- BRRHS-CG (Nikishin and Yeremin, 1995): block method with $t - 1$ random rhs

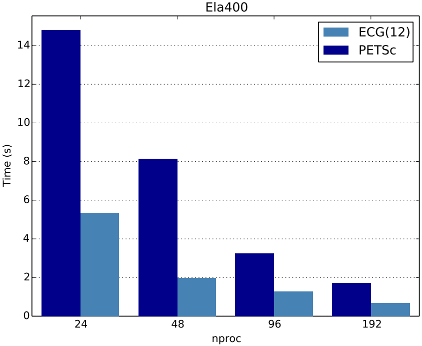
		PCG	BRRHS-CG		ECG	
	red.	iter	iter	$\dim(\mathcal{K}_{t,k})$	iter	$\dim(\mathcal{K}_{t,k})$
SKY2D	×	655	61	1952	57	1824
	✓	655	61	1739	59	1546
Ela3D100	×	955	102	3264	109	3488
	✓	955	102	3093	116	2384
Ela2D200	×	4551	255	8160	253	8096
	✓	4551	258	7331	266	6553

Enlarged CG: parallel performance

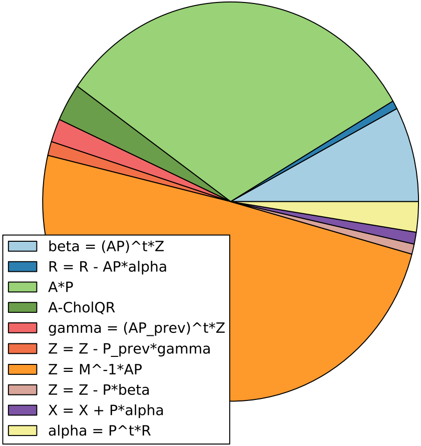
- MeSU (UPMC cluster)
  Intel Xeon E5-2670v3 (12 cores),
  24 cores per node
- Comparison with PETSc 3.7.4

Method	iter	time (s)	time/iter
ECG(12)	318	1.3	4.1×10^{-3}
PETSc	5198	3.3	6.3×10^{-4}





Detailed profiling (source slide O. Tissot)



- Ela400 on 96 cores
- Orthodir ECG(12)
- Around 50% of the time spent in applying the preconditioner
- Around 30% of the time spent in Sparse Matrix-Matrix

Method	iter	time (s)	time/iter
ECG(12)	318	1.3	4.1×10^{-3}
PETSc	5198	3.3	6.3×10^{-4}

Table : Comparison with PETSc PCG. PETSc iteration is 6.5 times faster than ECG(12) one. MKL-Pardiso has a strange behaviour with multiple rhs n our experiments: 1 rhs solve is 3 times faster than 2 rhs solve.

Conclusions

- Enlarged CG converges faster than classic CG on our test matrices
  - □ number of global reductions is reduced significantly
  - □ arithmetic intensity is increased
- Prototype code in C and MPI available

- Implement the block size reduction, tuning, and optimize the code
- Code will be available in november 2017

References (1)

Ashby, S. F., Manteuffel, T. A., and Saylor, P. E. (1990).
A taxonomy for conjugate gradient methods.
SIAM Journal on Numerical Analysis, 27(6):1542–1568.

Daas, H. A., Grigori, L., Hénon, P., and Ricoux, P. (2017).
Enlarged GMRES for reducing communication.
Technical Report 8910, Inria.

Grigori, L. and Moufawad, S. (2014).
Communication avoiding incomplete LU0 factorization.
SIAM Journal on Scientific Computing, in press.
Also as INRIA TR 8266.

Grigori, L. and Tissot, O. (2017).
Reducing the communication and computational costs of enlarged krylov subspaces conjugate gradient.
Research Report RR-9023.

OLeary., D. P. (1980).
The block conjugate gradient algorithm and related methods.
Linear Algebra and Its Applications, 29:293–322.