Elemental

Distributed direct linear algebra and optimization

Jack Poulson

Dept. of Mathematics, Inst. of Comput. & Math. Eng.

Stanford University

Contact Information:
450 Serra Mall, Building 380
Stanford University
Stanford, CA 94306

Email: poulson@stanford.edu

Abstract

Elemental is a continually-evolving high-performance C++11 library built on top of BLAS/LAPACK/MPI/METIS in order to provide distributed dense and sparse-direct linear algebra and optimization functionality. The fundamental data structure for Elemental's dense linear algebra is the `DistMatrix` class, which is templated over both the underlying scalar and the data distribution, in order to provide a simple and efficient means of both moving between, and abstracting over, a variety of 4D decompositions of the available processes.

In addition to supporting classical matrix decompositions (with distributed Schur decompositions employing ScaLAPACK's Hessenberg QR algorithms) and novel high-performance routines for computing one and two-norm pseudospectra, Elemental heavily exploits quasi-semidefinite LDL^H factorizations in order to extend its sparse-direct Cholesky functionality to nonsymmetric systems, as well as Least-Squares and Minimum-Length regression problems. Furthermore, the same solvers have been extended to support distributed primal-dual predictor-corrector Interior Point Methods for Linear and Quadratic Programs.

Core data structures

DistMatrix

Elemental [10] employs process grid decompositions closely related to those employed by "2.5D" and "3D" algorithms [12] for the purposes of abstracting over the wide variety of distributions used by the library. In particular, the `AbstractDistMatrix<T>` class, where `T` is the scalar datatype, is manipulated in terms of the 4D decomposition

$$\text{dist} \times \text{redundant} \times \text{cross} = (\text{column} \times \text{row}) \times \text{redundant} \times \text{cross},$$

where the *column* communicator distributes each column of the matrix, the *row* communicator distributes each row of the matrix, the *redundant* communicator redundantly stores the same data, and the *cross* communicator only assigns data to a single process. Each instance of the `DistMatrix<T,U,V>` template class chooses a different 4D decomposition, with `U` and `V` roughly respectively specifying the column and row communicators (at some point, Elemental may switch to four distribution template parameters).

DistSparseMatrix

Elemental currently employs 1D sparse matrix distributions (with the exception of multifrontal trees) for its distributed sparse matrix interfaces. In particular, a simplified, local version of PETSc's interface [1] is currently used, but it is known that, for matrices with imbalanced numbers of nonzero entries per row, 2D matrix distributions reduce the $O(n_{\text{nonzero}})$ worst-case parallel runtime to $O(n_{\text{nonzero}}/\sqrt{p})$ [5] (consider $O(n)$ nonzeros and one dense row).

High-perf. computation of pseudospectra

Defining the p-norm ϵ-pseudospectrum of a matrix A as

$$\mathcal{L}_\epsilon^p(A) \equiv \{\xi \in \mathbb{C} : \|(A-\xi I)^{-1}\|_p > \frac{1}{\epsilon}\},$$

a Schur decomposition $A = QTQ^H$ implies

$$\mathcal{L}_\epsilon^p(A) = \{\xi \in \mathbb{C} : \|Q(T-\xi I)^{-1}Q^H\|_p > \frac{1}{\epsilon}\},$$

and, in the case of $p = 2$, that

$$\mathcal{L}_\epsilon^2(A) = \{\xi \in \mathbb{C} : \sigma_{\min}(T-\xi I) < \epsilon\}.$$

Van Loan suggested [8] using the Schur decomposition as a preprocessing step for probing the minimum singular values for shifts along a line in the complex plane, but the generalization is obvious.

Since determining the pseudospectra over a large number of points in the complex plane (for various ϵ) requires evaluating $\|Q(T-\xi I)^{-1}Q^H\|_p$ for various different choices of $\xi \in \mathbb{C}$, recognizing that

$$TX - X\,\text{diag}((\xi_i)_i) = Y$$

can be solved for X using a trivial modification of TRSM implies that it is worthwhile to interleave many iterative norm estimation routines (Lanczos in the case where $p = 2$, and block 1-norm estimation [7] when $p = 1$). Elemental implements distributed-memory variants of these high-performance schemes (see, for example, `SpectralPortrait`).

Regularized quasi-semidefinite factorizations

Elemental supports a variety of schemes (e.g., quad-precision iterative refinement and FGMRES(k)) for preconditioning the *quasi-semidefinite* [13] matrix

$$K = \begin{pmatrix} G & A^H \\ A & -H \end{pmatrix}, \quad G, H \succeq 0$$

using an iteratively-refined solution from a Cholesky-like factorization of the *a priori* regularized quasi-*definite* matrix

$$K_{\delta,\delta} \equiv \begin{pmatrix} G+\delta I & A^H \\ A & -H-\delta I \end{pmatrix},$$

where requiring $\text{Econd}(A) \approx \left(\frac{\|A\|_2}{\delta}\right)^2 < \frac{1}{\epsilon}$ implies $\delta \gtrsim \sqrt{\epsilon}\|A\|_2$ [11, 6].

Consider the matrix

$$K_n = \begin{pmatrix} 4J_{1/2}(n)^T J_{1/2}(n) & I \\ I & -I \end{pmatrix},$$

which, making use of dynamic regularization (based upon the pivot magnitude) would have an *effective* condition number similar to 2^{n-1} despite its true condition number being quite small (e.g., for $n = 100$, it is less than 15).

Sparse Min Length, Least Squares, and Linear Solve

Until the mid-1990's, it was standard practice [9] to transform

$$\min_x \|Ax-b\|_2, \text{ and}$$
$$\min_x \|x\|_2 \text{ s.t. } Ax = b,$$

respectively into

$$\begin{pmatrix} \alpha I & A \\ A^H & 0 \end{pmatrix}\begin{pmatrix} r/\alpha \\ x \end{pmatrix} = \begin{pmatrix} b \\ 0 \end{pmatrix}, \text{ and}$$
$$\begin{pmatrix} \alpha I & A^H \\ A & 0 \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 0 \\ b \end{pmatrix},$$

with $\alpha = \sigma_{\min}(A)$ being known to result in quasi-optimal conditioning w.r.t. both the augmented system [3] and x [4], with conditioning within a small factor of $\text{cond}(A)$.

When A is square and nonsingular, it is of course possible to use either of these methods and to also set $\alpha = 0$ [6].

Convex optimization

IPMs for LPs and QPs

The workhorses for convex optimization in Elemental are currently distributed sparse-direct primal-dual predictor-corrector IPMs for QPs of the form

$$\min_x \frac{1}{2}x^T Q x + c^T x$$
$$\text{s.t. } Ax = b,\ Gx + s = h,\ s \geq 0,$$

though customized interfaces for $Q = 0$ and $G = -I$ are also available.

Schemes implemented on top of IPMs

- Basis Pursuit
- Basis Pursuit Denoising / LASSO
- Chebyshev Points
- Dantzig Selectors
- Elastic Nets
- Least Absolute Value regression
- Non-negative Least Squares
- Non-negative Matrix Factorization (via alternating NNLS)
- Support Vector Machines (soft-margin)
- Total Variation Denoising (1D)

Models implemented using ADMM

- Robust PCA
- Sparse Inverse Covariance Selection

External interfaces

In addition to the native `C++11` API, Elemental exposes nearly all of its functionality to both C and Python, with the C interface serving as the bridge to Python via `ctypes`. A similar interface to Julia [2] is actively planned.

Future work

- Adding sparse condition estimation to provide robust default regularization parameters and convergence tolerances for QSD systems
- SOCP, GP, and SDP solvers
- Julia [2] and IJulia/Jupyter interfaces
- Switching from Hager-Higham to Higham-Tisseur one-norm estimation for pseudospectra [7]
- Switch from 1D to 2D sparse matrix distributions [5]
- High-performance quad-precision sparse-direct factorizations
- Accelerator support

Availability

Elemental is available under The New BSD License from `libelemental.org` and `github.com/elemental/Elemental`. A release candidate for version 0.86 is now available.

Acknowledgements

- Michael Saunders for extended discussions on QSD systems
- Stephen Boyd and AJ Friend for detailed advice on Interior Point Methods
- Haesun Park for extended discussions about NMF

References

[1] Satish Balay, Shrirang Abhyankar, Mark F. Adams, Jed Brown, Peter Brune, Kris Buschelman, Victor Eijkhout, William D. Gropp, Dinesh Kaushik, Matthew G. Knepley, Lois Curfman McInnes, Karl Rupp, Barry F. Smith, and Hong Zhang. PETSc Web page. `http://www.mcs.anl.gov/petsc`, 2014.

[2] Jeff Bezanson, Alan Edelman, Stefan Karpinski, and Viral B. Shah. Julia: A fresh approach to numerical computing. 2014.

[3] Ake Björck. Iterative refinement of least squares solutions I. *BIT*, 7:257–278, 1967.

[4] Ake Björck. Pivoting and stability in the augmented system method. In D.F. Griffiths and G.A. Watson, editors, *Proc. 14th Dundee Conf.*, pages 1–16. Pitman Research Notes in Math., 1992.

[5] Aydın Buluç and John R. Gilbert. The Combinatorial BLAS: Design, implementation, and applications. *Internat. J. High Perf. Comput. Appl.*, 25(4):496 – 509, 2011.

[6] Alan George and Michael A. Saunders. Solution of sparse linear equations using cholesky factors of augmented systems. Technical report, SOL 99-1, Dept of EESOR, Stanford University, 1999.

[7] Nicholas J. Higham and Francoise Tisseur. A block algorithm for matrix 1-norm estimation, with an application to 1-norm pseudospectra. *SIAM J. Matrix Anal. Appl.*, 21(4):1185–1201, 2000.

[8] Charles Van Loan. How near is a stable matrix to an unstable matrix? Technical report, Cornell University, 1984.

[9] Pontus Matstoms. Sparse QR factorization in MATLAB. *ACM Trans. Math. Softw.*, 20(1):136–159, 1994.

[10] Jack Poulson, Bryan Marker, Robert A. van de Geijn, Jeff R. Hammond, and Nichols A. Romero. Elemental: A new framework for distributed memory dense matrix computations. *ACM Trans. Math. Softw.*, 39(2):13:1–13:24, February 2013.

[11] Michael A. Saunders. *Cholesky-based Methods for Sparse Least Squares: The Benefits of Regularization*, pages 92–100. SIAM, 1996.

[12] Edgar Solomonik and James Demmel. Communication-optimal parallel 2.5D matrix multiplication and LU factorization algorithms. In *Proc. 17th Internat. Conf. Parallel Proc. - Vol. II*, Euro-Par'11, pages 90–109. Springer-Verlag, 2011.

[13] R. J. Vanderbei. Symmetric quasi-definite matrices. *SIAM J. Optim.*, 5(1):100–113, 1995.