PETSc Interface for Elemental Package

Xuan Zhou

Illinois Institute of Technology

August 2, 2012

Joint work with Hong Zhang and Jed Brown

Introduction to Elemental

Elemental is a library implemented by Jack Poulson in C++ for distributed-memory dense linear algebra that strives to be both fast and user-friendly. Elemental is built on lessons learned from ScaLAPACK and PLAPACK.

- Unlike ScaLAPACK and like PLAPACK, the distribution block size is not linked to the algorithmic block size in Elemental.
- Elemental chooses the distribution block size to be 1 such that load balance is optimal and handling of submatrices is much more convenient.
- Elemental is a fix to the mild nonscalability of PLAPACK that results from simplified implementation at a time when the number of processors was small.

Matrix Distribution in Elemental - Grid

The Grid class converts MPI communicators into a two-dimensional processor grid meant for distributing matrices.

For example, an MPI communicator of 6 processors can be associated with a grid of size 1×6, 2×3, 3×2, or 6×1.

Matrix Distribution in Elemental - DistMatrix

The DistMatrix class is the basic building block of Elemental. An instance of the Grid class must be passed into the constructor of the DistMatrix class to determine how the data is distributed over different processors.

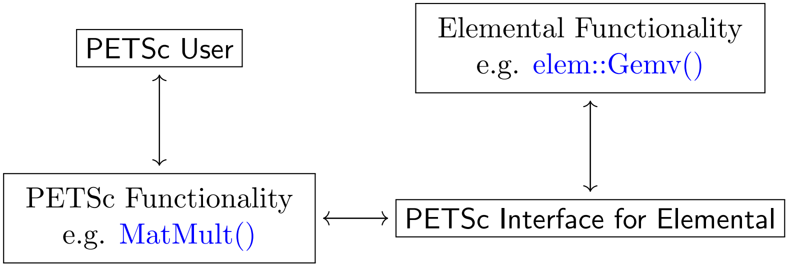
For a 7×7 matrix distributed over a 2×3 processor grid, individual entries would be owned by the following processors by default:

$$\begin{pmatrix} \boxed{\begin{matrix} 0 & 2 & 4 \\ 1 & 3 & 5 \end{matrix}} \begin{matrix} 0 & 2 & 4 & 0 \\ 1 & 3 & 5 & 1 \end{matrix} \\ \begin{matrix} 0 & 2 & 4 & 0 & 2 & 4 & 0 \\ 1 & 3 & 5 & 1 & 3 & 5 & 1 \\ 0 & 2 & 4 & 0 & 2 & 4 & 0 \\ 1 & 3 & 5 & 1 & 3 & 5 & 1 \\ 0 & 2 & 4 & 0 & 2 & 4 & 0 \end{matrix} \end{pmatrix}$$

Example Usage of Elemental Routines

- Matrix-vector multiplication $y = Ax$:
  `elem::Gemv(elem::NORMAL,1,A,x,0,y);`
- Cholesky Factorization $A = LL^*$:
  `elem::Cholesky(elem::UPPER,A);`
  The upper triangle of the Hermitian positive-definite matrix A is overwritten by its Cholesky factor L^*.
- Solving $Ax = b$ for x by (partially pivoted) LU factorization:
  `elem::GaussianElimination(A,b);`
  Upon completion, A will be overwritten with Gaussian elimination and b will be overwritten by x.

Reasons for Creating a PETSc Interface for Elemental



- Users are familiar with the PETSc environment.
- Users want dense linear algebra computation which Elemental promises to do a good job at.
- The interface of Elemental is quite different from that of PETSc.

Design of the Interface

- We define a new matrix type MATELEMENTAL which has a elem::DistMatrix as one of its members emat. This Elemental matrix is the logical matrix with its entries rearranged in Elemental's cyclical order.
- For vectors, we directly pass the local buffer to the constructor of elem::DistMatrix to create an Elemental vector.
- The Elemental matrices and Elemental vectors are passed to Elemental routines to perform desired calculations.
- The same processor grid is shared within one communicator to ensure compatibility.
- Bottom line: we use only index manipulation and no data copy is necessary.

Design of the Interface - Supported Functions

MatSetValues	MatMult	MatMultAdd
MatMultTranspose	MatMultTransposeAdd	MatSolve
MatSolveAdd	MatLUFactor	MatCholeskyFactor
MatTranspose	MatGetInfo	MatGetDiagonal
MatDiagonalScale	MatNorm	MatZeroEntries
MatSetUp	MatDuplicate	MatAXPY
MatCopy	MatScale	MatView
MatConvert	MatMatMult	MatMatTransposeMult
MatConjugate	MatMatSolve	MatHermitianTranspose

Design of the Interface - A Simple Example

For example, if we wish to compute $y = Ax$ using the PETSc interface for Elemental, where A is of size $M \times N$ and x is of size $N \times 1$, then the following would happen mathematically (logically):

- Step 1: Elemental inputs A^E and x^E are created by $A^E = P_M^* A P_N$ and $x^E = P_N^* x$.
- Step 2: The Elemental routine elem::Gemv then computes $y^E = A^E x^E$.
- Step 3: PETSc output y is generated by $y = P_M y^E$.

Design of the Interface - A Simple Example (Continued)

If we use 4 processors with a 2×2 grid, the logical vector $(0,0,1,1,2,2,3,3)^T$ will be mapped to the Elemental vector $(0,1,2,3,0,1,2,3)^T$.

A mapping example from a logical matrix to an Elemental matrix is

$$\begin{pmatrix} 0 & 0 & 2 & 2 & 0 & 0 & 2 & 2 \\ 0 & 0 & 2 & 2 & 0 & 0 & 2 & 2 \\ 1 & 1 & 3 & 3 & 1 & 1 & 3 & 3 \\ 1 & 1 & 3 & 3 & 1 & 1 & 3 & 3 \\ 0 & 0 & 2 & 2 & 0 & 0 & 2 & 2 \\ 0 & 0 & 2 & 2 & 0 & 0 & 2 & 2 \\ 1 & 1 & 3 & 3 & 1 & 1 & 3 & 3 \\ 1 & 1 & 3 & 3 & 1 & 1 & 3 & 3 \end{pmatrix} \rightarrow \begin{pmatrix} 0 & 2 & 0 & 2 & 0 & 2 & 0 & 2 \\ 1 & 3 & 1 & 3 & 1 & 3 & 1 & 3 \\ 0 & 2 & 0 & 2 & 0 & 2 & 0 & 2 \\ 1 & 3 & 1 & 3 & 1 & 3 & 1 & 3 \\ 0 & 2 & 0 & 2 & 0 & 2 & 0 & 2 \\ 1 & 3 & 1 & 3 & 1 & 3 & 1 & 3 \\ 0 & 2 & 0 & 2 & 0 & 2 & 0 & 2 \\ 1 & 3 & 1 & 3 & 1 & 3 & 1 & 3 \end{pmatrix}$$

Design of the Interface - A Simple Example (Continued)

Actually the interface works in the following manner.

```
PetscErrorCode MatMult_Elemental(Mat A,Vec X,Vec Y) {
 Mat_Elemental *a = (Mat_Elemental*)A->data;
 VecGetArrayRead(X,x); VecGetArray(Y,y);
 elem::DistMatrix<PetscScalar,VC,*> xe(N,1,0,x,n,*a->grid);
 elem::DistMatrix<PetscScalar,VC,*> ye(M,1,0,y,m,*a->grid);
 elem::Gemv(elem::NORMAL,one,*a->emat,xe,zero,ye);
 VecRestoreArrayRead(X,x); VecRestoreArray(Y,y);}
```

Example Usage of PETSc Interface for Elemental

- Configure PETSc with the option –download-elemental –with-clanguage=cxx. If you are dealing with complex numbers, add the option –with-scalar-type=complex
- Matrix-vector multiplication $y = Ax$:
  MatMult(A,x,y);
- Cholesky Factorization $A = LL^*$:
  MatCholesky(A,0,0);
  The upper triangle of the Hermitian positive-definite matrix A is overwritten by its Cholesky factor L^*.
- Solving $Ax = b$ for x by (partially pivoted) Gaussian elimination:
  Use PETSc's KSP interface with the option
  -pc_type lu -pc_factor_mat_solver_package elemental

Example Usage of PETSc Interface for Elemental (Continued)

- petsc-dev/src/mat/examples/tests/ex38.c
- petsc-dev/src/mat/examples/tests/ex39.c
- petsc-dev/src/mat/examples/tests/ex145.c
- petsc-dev/src/ksp/ksp/examples/tests/ex40.c

Thank you!