SAND2007-7131P

Using Belos:

Next-generation Iterative Solvers

November 6th, 2007

Mike Heroux
Rich Lehoucq
Mike Parks
Heidi Thornquist (Lead)

Outline

- Introduction
  - Motivation
  - Belos design
- What’s in Trilinos 8.0
  - Framework implementations
  - Current linear solver iterations
  - Parameter list driven solver managers
  - Stratimikos interface
- Concluding Remarks
  - Milestone lessons
  - What’s next for Belos …

Introducing Belos

- Next-generation linear solver library, written in templated C++.

- Provide a generic framework for developing algorithms for solving large-scale linear problems.

- Algorithm implementation is accomplished through the use of traits classes and abstract base classes:
  - ex.: `MultiVecTraits`, `OperatorTraits`
  - ex.: `Iteration, SolverManager, StatusTest`

- Includes block linear solvers:
  - Higher operator performance.

Why Belos?

- AztecOO provides solvers for $Ax=b$
- What about solvers for:
  - Simultaneously solved systems w/ multiple-RHS: $AX = B$
  - Sequentially solved systems w/ multiple-RHS: $AX_i = B_i, i=1,\ldots,t$
  - Sequences of multiple-RHS systems: $A_iX_i = B_i, i=1,\ldots,t$
- Many advanced methods for these types of linear systems
  - Block methods: block GMRES [Vital], block CG/BICG [O'Leary]
  - "Seed" solvers: hybrid GMRES [Nachtigal, et al.]
  - Recycling solvers: recycled Krylov methods [Parks, et al.]
  - Restarting techniques, orthogonalization techiques, …
- Efficient R&D of advanced linear solution methods requires:
  - interoperability
  - extensibility
  - reusability

Sandia National Laboratories

Belos Design

Why not create a solver library that:

1. Provides an abstract interface to an operator-vector products, scaling, and preconditioning.
2. Allows the user to enlist any linear algebra package for the elementary vector space operations essential to the algorithm. (Epetra, PETSc, etc.)
3. Allows the user to define convergence of any algorithm (a.k.a. status testing).
4. Allows the user to determine the verbosity level, formatting, and processor for the output.
5. Allows for easier creation of new solvers through "managers" using "iterations" as the basic kernels.

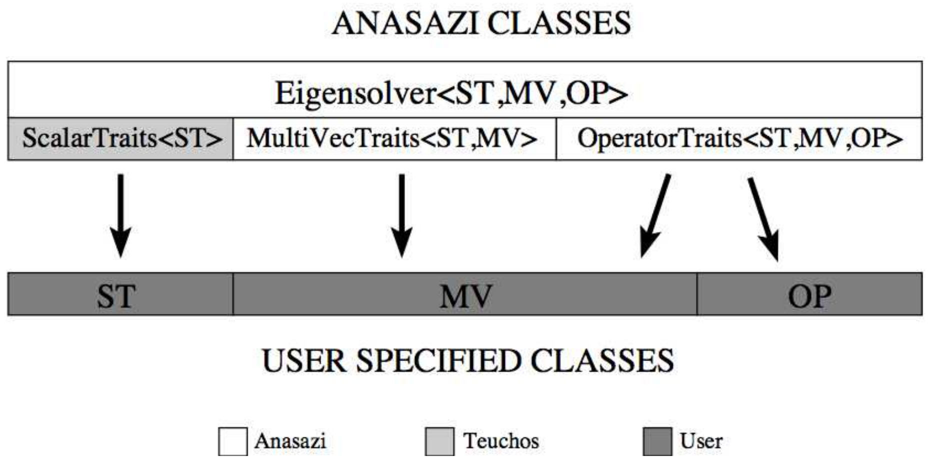
Sandia National Laboratories

Belos Classes

- LinearProblem Class
  - Describes the problem and stores the answer
- SolverManager Class
  - Parameter list driven strategy object describing behavior of solver
- Iteration Class
  - Provide basic linear solver iteration interface.
- MultiVecTraits and OperatorTraits
  - Traits classes for interfacing linear algebra
- OrthoManager Class
  - Provide basic interface for orthogonalization
- StatusTest Class
  - Control testing of convergence, etc.
- OutputManager Class
  - Control verbosity and printing in a MP scenario

Linear Algebra Interface

- **Belos::MultiVecTraits**
  - Abstract interface to define the linear algebra required by most iterative linear solvers:
    - creational methods
    - dot products, norms
    - update methods
    - initialize / randomize



- **Belos::OperatorTraits**
  - Abstract interface to enable the application of an operator to a multivector.

Output Manager Interface

- Concrete class that enables control of the linear solver output.
- `Belos::OutputManager`
  - Get/Set Methods:
    - `void setVerbosity( int vbLevel );`
    - `int getVerbosity( );`
    - `ostream& stream( MsgType type );`
  - Query Methods:
    - `bool isVerbosity( MsgType type );`
  - Print Methods:
    - `void print( MsgType type, const string output );`
- Default is lowest verbosity (errors), output on one processor.

StatusTest Interface

- Informs linear solver iteration of change in state, as defined by user.
- Similar to NOX / AztecOO.
- Multiple criterion can be logically connected.
- Abstract base class `Belos::StatusTest`
  - `TestStatus checkStatus( Iteration<…>* iterate );`
  - `TestStatus getStatus() const;`
  - `void clearStatus();`
  - `void reset();`
  - `ostream& print( ostream& os, int indent = 0 ) const;`
- Concrete classes:
  - `Belos::StatusTestMaxIters`
  - `Belos::StatusTestGenResNorm`
  - `Belos::StatusTestImpResNorm`
  - `Anasazi::StatusTestOutput`
  - `Anasazi::StatusTestCombo`

Orthogonalization Manager

- Abstract interface to orthogonalization / orthonormalization routines for multivectors.
- Abstract base class `Belos::[Mat]OrthoManager`
    - Inner product
    - Multivector norm
    - Project multivector
    - Normalize multivector
    - Project and normalize multivector
    - Orthogonalization error
    - Orthonormalization error
- Concrete classes:
    - `Anasazi::DGKSOrthoManager`
    - `Anasazi::ICGSOrthoManager`
    - `Anasazi::IMGSOrthoManager`

Linear Problem Interface

- Provides an interface between the basic iterations and the linear problem to be solved.
- Concrete class `Belos::LinearProblem`
  - Allows preconditioning / scaling to be removed from the algorithm.
  - Stores operator, LHS, RHS, left-preconditioner, right-preconditioner

Iteration Interface

- Provides an abstract interface to Belos basic iteration kernels.
- Abstract base class `Belos::Iteration`
  - get / reset number of iterations
  - native residuals
  - current / maximum subspace size
  - linear problem
  - initialize / iterate
- Concrete iterations:
  - `Belos::BlockGmres`
  - `Belos::PseudoBlockGmres`
  - `Belos::GCRODR`
  - `Belos::CG, Belos::BlockCG`
- Requires these classes:
  - `Belos::LinearProblem`
  - `Belos::OutputManager`
  - `Belos::OrthoManager`
  - `Belos::StatusTest`

Trilinos 6.0

Solver Manager

- Provides an abstract interface to Belos solver managers
  - solution strategies
- Abstract base class **`Belos::SolverManager`**
  - Set / get the linear problem and solver parameters
  - Solve the linear problem
- Solvers are parameter list driven, take two arguments:
  - **`Belos::LinearProblem`**
  - **`Teuchos::ParameterList`**
- Concrete solver managers:
  - **`Belos::BlockGmresSolMgr`**
  - **`Belos::PseudoBlockGmresSolMgr`**
  - **`Belos::BlockFGmresSolMgr`**
  - **`Belos::GCRODRSolMgr`**
  - **`Belos::BlockCGSolMgr`**

What’s next for Belos?

- Belos provides a powerful framework for developing robust linear solvers
- Future Development:
  - More performance improvements
  - More advanced solution methods
- Check out the Trilinos Tutorial:

  http://trilinos.sandia.gov/Trilinos8.0Tutorial.pdf
- See Belos website for more info:

  http://trilinos.sandia.gov/packages/belos