An Overview of the Trilinos Package Architecture

Michael A. Heroux
Sandia National Laboratories
Trilinos Workshop at Copper Mountain
March 30, 2004

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company,
for the United States Department of Energy under contract DE-AC04-94AL85000.

Sandia National Laboratories

The Team

Ross Bartlett
Lead Developer of TSFCore

Paul Boggs
Developer of TSF

Jason Cross
Developer of Jpetra

David Day
Developer of Komplex

Bob Heaphy
Lead developer of Trilinos SQA

Mike Heroux
Trilinos Project Leader
Lead Developer of Epetra, AztecOO, Kokkos, Komplex and IFPACK, TSF
Developer of Amesos, Belos

Robert Hoekstra
Developer of Epetra

Russell Hooper
Developer of NOX

Vicki Howle
Developer of Belos and TSF

Jonathan Hu
Developer of ML

Tammy Kolda
Lead Developer of NOX

Rich Lehoucq
Developer of Anasazi and Belos

Kevin Long
Lead Developer of TSF,
Developer of Belos

Roger Pawlowski
Lead Developer of NOX

Michael Phenow
Trilinos Webmaster

Eric Phipps
Developer of LOCA and NOX

Andrew Rothfuss
Developer of TSF

Andrew Salinger
Lead Developer of LOCA

Marzio Sala
Lead author of Trilinos Tutorial
Developer of ML and Amesos

Paul Sexton
Developer of Epetra and Tpetra

Ken Stanley
Lead Developer of Amesos

Heidi Thornquist
Lead Developer of Anasazi and Belos

Ray Tuminaro
Lead Developer of ML

Jim Willenbring
Developer of Epetra and Kokkos.
Trilinos library manager

Alan Williams
Developer of Epetra

Outline of Talk

- Motivation
- Trilinos Package Concepts.
- Overview of Major Packages.
- Examples Using Trilinos.
- SQA/SQE.
- Availability and support.
- Concluding remarks.

Sandia National Laboratories

Motivation For Trilinos

- Sandia does A LOT of solver work.
- Sandia uses MANY external solver packages.
- When I started at Sandia in May 1998:
  - Aztec was a mature package. Used in many codes.
  - FETI, PETSc, DSCPack, Spooles, ARPACK, DASPK, and many other codes were (and are) in use.
  - New projects were underway or planned in multi-level preconditioners, eigensolvers, non-linear solvers, etc…
  - New application capabilities were underway, demanding new solvers.
- The challenges:
  - Little or no coordination was in place to:
    - Efficiently reuse existing solver technology.
    - Leverage new development across various projects.
    - Support solver software processes.
    - Provide consistent solver APIs for applications.
  - ASCI was forming software quality assurance/engineering (SQA/SQE) requirements:
    - Daunting requirements for any single solver effort to address alone.

Sandia National Laboratories

Evolving Trilinos Solution

- Trilinos[1] is an evolving framework to address these challenges:
  - Includes common core set of vector, graph and matrix classes (Epetra).
  - Provides a common abstract solver API (TSF).
  - Provides a ready-made package infrastructure:
    - Source code management (cvs, bonsai).
    - Build tools (autotools).
    - Automated regression testing (queue directories within repository).
    - Communication tools (mailman mail lists).
  - Specifies requirements and suggested practices for SQA.
- In general allows us to categorize efforts:
  - Efforts best done at the Trilinos level (useful to most or all packages).
  - Efforts best done at a package level (peculiar or important to a package).
  - **Allows package developers to focus only on things that are unique to their package.**

1. Trilinos loose translation: “A string of pearls”

Trilinos Benefits: The Three "I"s

- **Infrastructure:**
  - Repository (CVS)
  - Issue Tracking (Bugzilla)
  - Communication (Mailman)
  - Debugging (Bonsai)
  - Jumpstart (new_package)
  - Portable build process (Autotools: configure, make)
  - SQA Tools and Policies.
  - Documentation: Developers Guides, Installation Guide, Tutorial, etc.
- **Interfaces:**
  - Interoperability: Between Trilinos Packages, with external SW.
  - Extensible.
  - Without Interdependence.
- **Implementations:**
  - Real, working code that implements all interfaces.
  - Solid default implementations, but *not required to use*.

Sandia National Laboratories

Trilinos Package Concepts

Trilinos Packages

- Trilinos is a collection of *Packages*.
- Each package is:
  - Focused on important, state-of-the-art algorithms in its problem regime.
  - Developed by a small team of domain experts.
  - Self-contained: No explicit dependencies on any other software packages (with some special exceptions).
  - Configurable/buildable/documented on its own.
- Sample packages: NOX, AztecOO, IFPACK, Meros.
- Special package collections: Petra, TSF, Teuchos.

Primary Trilinos Packages

8/4/2003 - v12

- **Basic Linear Algebra Libraries**
  - Epetra: Current Production C++ Library
    - Epetra Core
    - Epetra Extensions
  - Tpetra: Next Generation C++ Library
  - Jpetra: Java Library
- **Common Services**
  - Teuchos: Parameter Lists, BLAS Interfaces, etc
- **Abstract Interfaces and Adaptors**
  - TSFCore: Basic Abstract classes
  - TSF Extensions: Aggregate/composite, overloaded operators
  - TSF Utilities: Core utility classes
- **Linear Solvers**
  - Amesos: OO Interfaces to 3rd party direct solvers
    - SuperLU
    - KundertSparse
    - SuperLUDist
    - DSCPack
    - UMFPack
    - MUMPS
  - AztecOO: Preconditioned Krylov Package based on Aztec
  - Komplex: Complex solver via equivalent real formulations
  - Belos: Next generation Krylov and block Krylov solvers
- **Preconditioners**
  - ML: Multi-level preconditioners
  - Meros: Segregated/Block Preconditioners
  - IFPACK: Algebraic preconditioners
- **Nonlinear Solvers**
  - NOX: Collection of nonlinear solvers
  - LOCA: Library of Continuation Algorithms
- **Eigensolvers**
  - Anasazi: Collection of eigensolvers
- **Time Integration**
  - TOX: Planned development
- **"New Package"**
  - "Hello World": Package Template to aid integration of new packages
  - Web site with layout and instructions

Notes:

- ASCI Algorithms funds much of Trilinos development (LDRD, CSRF, MICS also).
- All packages available (except TOX, aka Rhythmos).
- All information available at Trilinos website: software.sandia.gov/trilinos

Sandia National Laboratories

Package	Description	Release 3.1 (9/2003)		4 (5/2004)	
		3.1 General	3.1 Limited	4 General	4 Limited
Amesos	3rd Party Direct Solver Suite		X	X	X
Anasazi	Eigensolver package				X
AztecOO	Linear Iterative Methods	X	X	X	X
Belos	Block Linear Solvers				X
Epetra	Basic Linear Algebra	X	X	X	X
EpetraExt	Extensions to Epetra		X	X	X
Ifpack	Algebraic Preconditioners	X	X	X	X
Jpetra	Java Petra Implementation				X
Kokkos	Sparse Kernels			X	X
Komplex	Complex Linear Methods	X	X	X	X
LOCA	Bifurcation Analysis Tools	X	X	X	X
Meros	Segregated Preconditioners		X		X
ML	Multi-level Preconditioners	X	X	X	X
NewPackage	Working Package Prototype	X	X	X	X
NOX	Nonlinear solvers	X	X	X	X
Pliris	Dense direct Solvers			X	X
Teuchos	Common Utilities			X	X
TSFCore	Abstract Solver API			X	X
TSFExt	Extensions to TSFCore			X	X
Tpetra	Templated Petra				X
Totals		**8**	**11**	**15**	**20**

Three Special Trilinos Package Collections

- **Petra**: Package of concrete linear algebra classes: Operators, matrices, vectors, graphs, etc.
  - Provides working, parallel code for basic linear algebra computations.
- **TSF**: Packages of abstract solver classes: Solvers, preconditioners, matrices, vectors, etc.
  - Provides an application programmer interface (API) to any other package that implements TSF interfaces.
  - Inspired by HCL.
- **Teuchos (pronounced Tef-hos)**: Package of basic tools:
  - Common Parameter list, smart pointer, error handler, timer.
  - Interface to BLAS, LAPACK, MPI, XML, …
  - Common traits mechanism.
  - Goal: Portable tools that enhance interoperability between packages.

Dependence vs. Interoperability

- Although most Trilinos packages have no explicit dependence, each package must interact with *some* other packages:
  - NOX needs operator, vector and solver objects.
  - AztecOO needs preconditioner, matrix, operator and vector objects.
  - Interoperability is enabled at configure time. For example, NOX:

    --enable-nox-lapack compile NOX lapack interface libraries

    --enable-nox-epetra compile NOX epetra interface libraries

    --enable-nox-petsc compile NOX petsc interface libraries
- Trilinos is a vehicle for:
  - Establishing interoperability of Trilinos components…
  - Without compromising individual package autonomy.
- Trilinos offers five basic interoperability mechanisms.

Sandia National Laboratories

Trilinos Interoperability Mechanisms

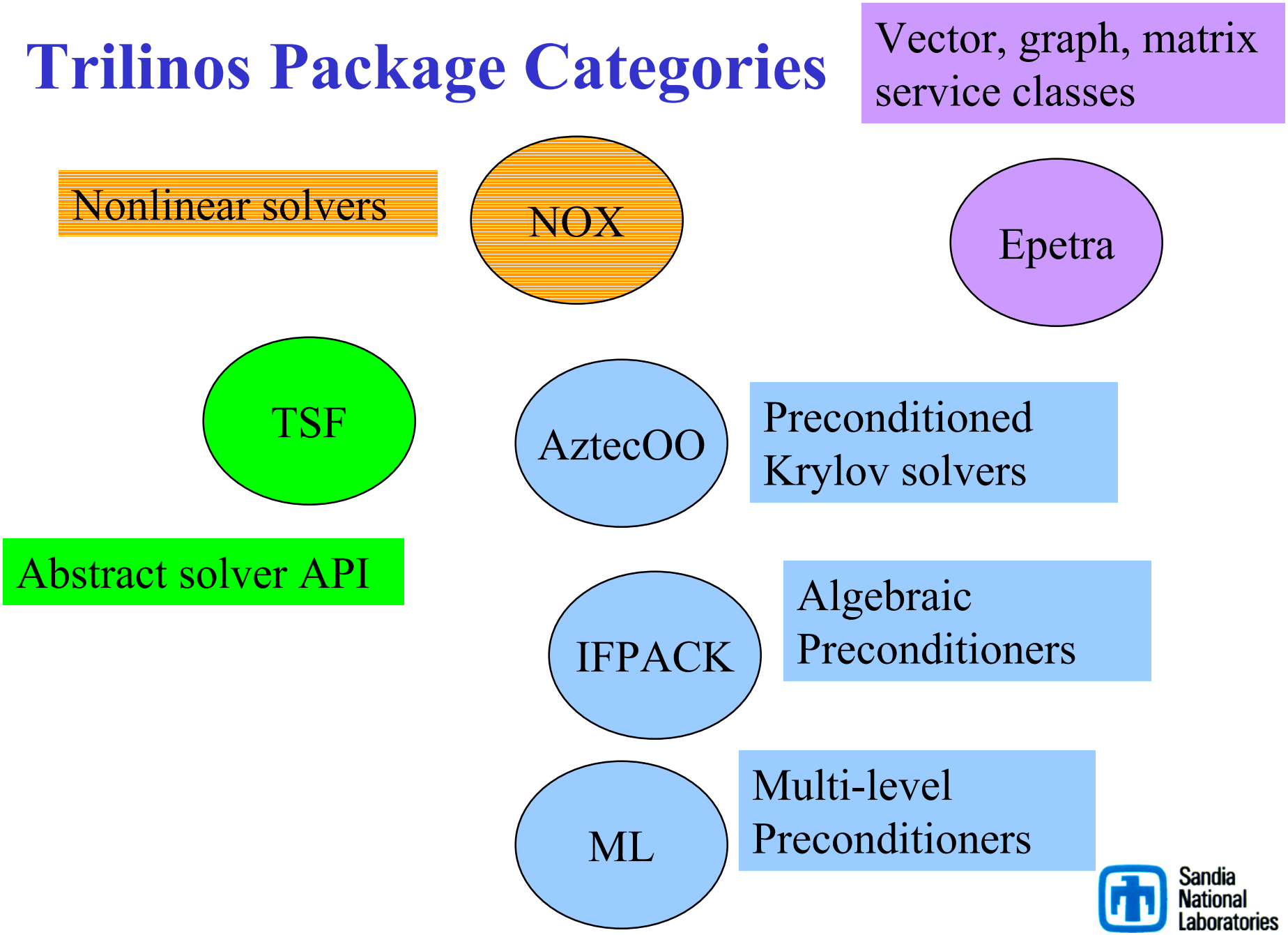
- M1: *Package* accepts user data as Epetra or TSF objects.
  =>Applications using Epetra/TSF can use *package*.
- M2: *Package* can be used via TSF abstract solver classes.
  => Applications or other packages using TSF can use *package*.
- M3: *Package* can use Epetra for private data.
  => *Package* can then use other packages that understand Epetra.
- M4: *Package* accesses solver services via TSF interfaces.
  => *Package* can then use other packages that implement TSF interfaces.
- M5: *Package* builds under Trilinos `configure` scripts.
  => *Package* can be built as part of a suite of packages.
  => Cross-package dependencies can be handled automatically.

Sandia National Laboratories

Interoperability Example: AztecOO

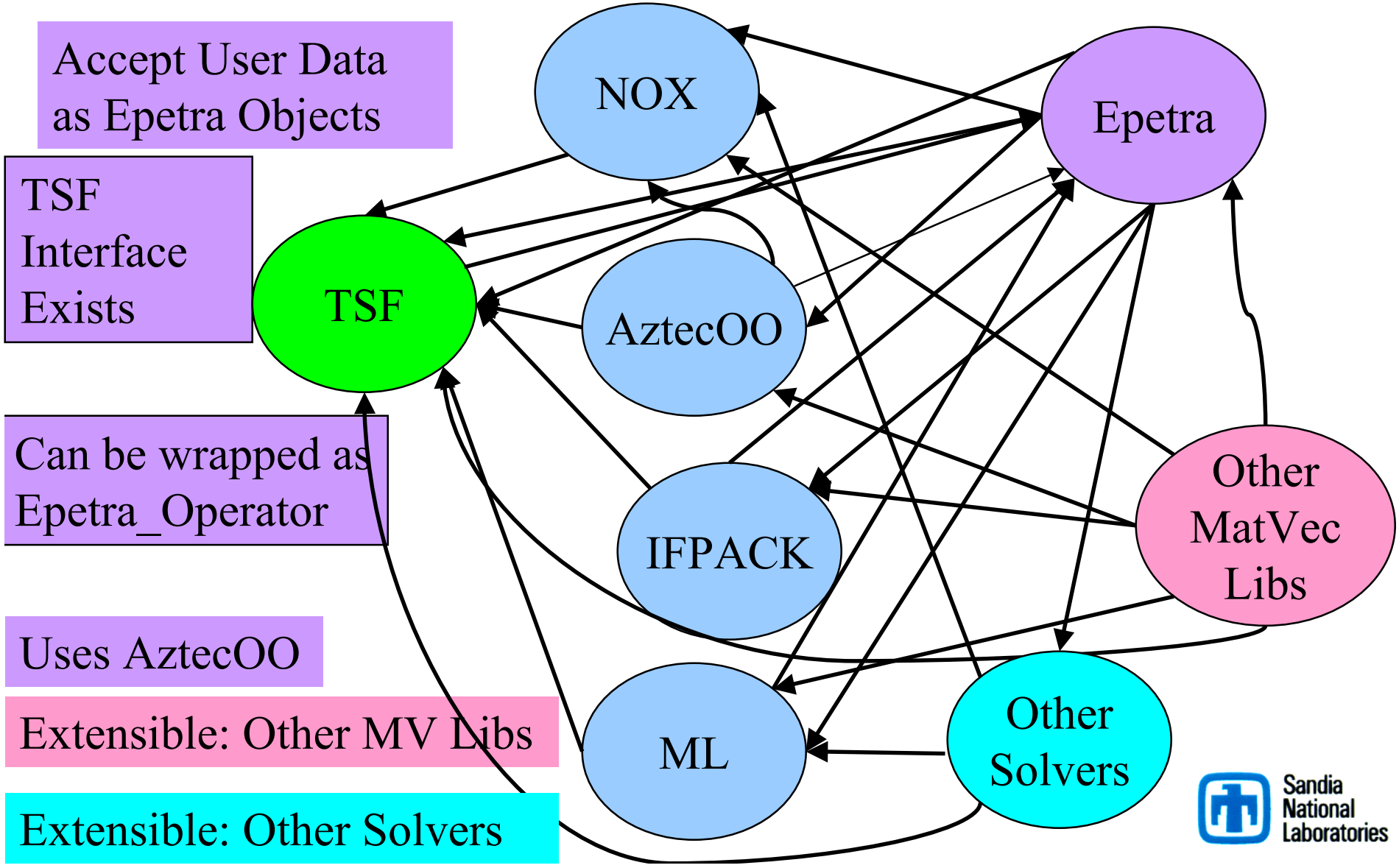
- AztecOO: Preconditioned Krylov Solver Package.
- Primary Developer: Mike Heroux.
- Minimal *explicit, essential* dependence on other Trilinos packages.
  - Uses abstract interfaces to matrix/operator objects.
  - Has independent configure/build process (but can be invoked at Trilinos level).
  - Sole dependence is on Epetra (but easy to work around).
- *Interoperable* with other Trilinos packages:
  - Accepts user data as Epetra matrices/vectors.
  - Can use Epetra for internal matrices/vectors.
  - Can be used via TSF abstract interfaces.
  - Can be built via Trilinos configure/build process.
  - Can provide solver services for NOX.
  - Can use IFPACK, ML or AztecOO objects as preconditioners.

Trilinos Package Categories



Sandia National Laboratories

Trilinos Package Interoperability



Sandia National Laboratories

What Trilinos is not

- Trilinos is not a single monolithic piece of software. Each package:
  - Can be built independent of Trilinos.
  - Has its own self-contained CVS structure.
  - Has its own Bugzilla product and mail lists.
  - Development team is free to make its own decisions about algorithms, coding style, release contents, testing process, etc.
- Trilinos top layer is not a large amount of source code:
  - Trilinos repository contains 452,187 source lines of code (SLOC).
  - Sum of the packages SLOC counts : 445,937.
  - Trilinos top layer SLOC count: 6, 250 (1.4%).
- Trilinos is not “indivisible”:
  - You don’t need all of Trilinos to get things done.
  - Any collection of packages can be combined and distributed.
  - Current public release contains only 6 of the 20+ Trilinos packages.

Overview of Trilinos Packages

Trilinos Concrete Support Component: Petra

- Petra[1] provides distributed matrix and vector services.
- Exists in basic form as an object model:
  - Describes basic user and support classes in UML, independent of language/implementation.
  - Describes objects and relationships to build and use matrices, vectors and graphs.
  - Has 3 implementations under development.

[1] Petra is Greek for "foundation".

Sandia National Laboratories

Petra Implementations

Three version under development:

- Epetra (*Essential Petra*):
  - Current production version.
  - Restricted to real, double precision arithmetic.
  - Uses stable core subset of C++ (circa 2000).
  - Interfaces accessible to C and Fortran users.
- Tpetra (*Templated Petra*):
  - Next generation C++ version.
  - Templated scalar and ordinal fields.
  - Uses namespaces, and STL: Improved usability/efficiency.
- Jpetra (*Java Petra*):
  - Pure Java. Portable to any JVM.
  - Interfaces to Java versions of MPI, LAPACK and BLAS via interfaces.

Epetra

- Package of concrete linear algebra classes:

  Operators, matrices, vectors, graphs, etc.
- Working, parallel code for basic linear algebra computations.
- Uses stable core subset of C++
- C/Fortran wrappers
- Restricted to real, double precision arithmetic
- Concrete implementation of the Petra object model

Sandia National Laboratories

Epetra User Class Categories

- Sparse Matrices: ***RowMatrix***, (CrsMatrix, VbrMatrix, FECrsMatrix, FEVbrMatrix)
- Linear Operator: ***Operator*****:** (AztecOO, ML, Ifpack)
- Dense Matrices: DenseMatrix, DenseVector, BLAS, LAPACK, SerialDenseSolver
- Vectors: Vector, MultiVector
- Graphs: CrsGraph
- Data Layout: Map, BlockMap, LocalMap
- Redistribution: Import, Export, LbGraph, LbMatrix
- Aggregates: LinearProblem
- Parallel Machine: ***Comm***, (SerialComm, MpiComm, MpiSmpComm)
- Utilities: Time, Flops

Sandia National Laboratories

Summary of Epetra Features

- Basic Stuff: What you would expect.
- Variable block matrix data structures.
- Multivectors.
- Arbitrary index labeling.
- Flexible, versatile parallel data redistribution.
- Language support for inheritance, polymorphism and extensions.
- View vs. Copy.

Trilinos Solver Framework (TSF)

- Epetra, AztecOO, Ifpack, ML, etc. PETSc, SuperLU, Hypre, HSL, ScaLapack

  } Lots of good solvers available

- TSF is an abstract class hierarchy:
  - Inspired by HCL, especially VectorSpace concept.
  - Provides uniform API to solvers, vectors, matrices.
  - Allows integration of many solvers via implementation of abstract classes.
  - Supports “generic” programming.
  - Provides compositional classes.
- Composed of TSFExtended, TSFCore, TSFCoreUtils.

Generic Programming using TSF

- Generic Programming:
  Implementation of algorithms using abstract interfaces.
- Example: CG using TSF interfaces.
- Allows use of CG with *any* vector/matrix classes that implement TSF interfaces.
- Very powerful for complex algorithms: Robust Block GMRES, etc.

Sandia National Laboratories

Aggregate Operator Construction

- TSF facilitates implicit (and explicit) construction of operators:
  - Partitioned (block): $A = \begin{bmatrix} F & B \\ C & 0 \end{bmatrix}$
  - Composite: $A = B \circ C$
  - Sum: $A = B + C$
  - Inverse: $A^{-1} = A \,\&\, solver(A)$
  - Others: Zero, Identity, Transpose, …
  - Recursively.

Sandia National Laboratories

AztecOO

- Aztec is the workhorse solver at Sandia:
  - Extracted from the MPSalsa reacting flow code.
  - Installed in dozens of Sandia apps.
  - 1700+ external licenses.
- AztecOO leverages the investment in Aztec:
  - Uses Aztec iterative methods and preconditioners.
- AztecOO improves on Aztec by:
  - Using Epetra objects for defining matrix and RHS.
  - Providing more preconditioners/scalings.
  - Using C++ class design to enable more sophisticated use.
- AztecOO interfaces allows:
  - Continued use of Aztec for functionality.
  - Introduction of new solver capabilities outside of Aztec.

ML: Multi-level Preconditioners

- ML package developed by Ray Tuminaro, Jonathan Hu, Marzio Sala.
- Critical technology for scalable performance of some key apps.
- ML compatible with other Trilinos packages:
  - Accepts user data as Epetra_RowMatrix object (abstract interface).
    - Any implementation of Epetra_RowMatrix works.
  - Implements the Epetra_Operator interface.
    - Allows ML preconditioners to be used with AztecOO and TSF.
- Can also be used completely independent of other Trilinos packages.

IFPACK: Algebraic Preconditioners

- Overlapping Schwarz preconditioners.
- Accept user matrix via abstract matrix interface (Epetra versions).
- Uses Epetra for basic matrix/vector calculations.
- Supports simple perturbation stabilizations and condition estimation.
- Separates graph construction from factorization, improves performance substantially.
- Compatible with AztecOO and TSF. Can be used with NOX.

Komplex: Complex linear solver

$$(A+iB)(x+iy) = (b+ic)$$

$$\begin{pmatrix} A & -B \\ B & A \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} b \\ c \end{pmatrix}$$

- Most algorithms work for complex numbers (with real numbers as a special case).
- Majority of our applications produce real-valued data.
- Solver development has been focused on real-valued problems.
- Writing complex versions of all software is not appealing.

- Alternative: Consider equivalent real formulations (ERFs).

- Komplex is an add-on module to AztecOO that:
  - Intelligently builds an ERF for a complex valued problem.
  - Computes the real-valued solution using AztecOO.
  - Returns the complex result to user.

- This is an effective approach in important practical settings.

Komplex Formulation

Consider a complex-valued matrix C:

$$C = \begin{bmatrix} c_{11} & c_{12} & 0 & 0 \\ c_{21} & c_{22} & 0 & 0 \\ 0 & c_{32} & c_{33} & 0 \\ 0 & 0 & c_{43} & c_{44} \end{bmatrix}$$

With each entry:

$$c_{ij} = a_{ij} + ib_{ij}$$

Rewrite as real-valued of twice the dimension:

$$K = \left[\begin{array}{cc|cc|cc|cc} a_{11} & -b_{11} & a_{12} & -b_{12} & 0 & 0 & 0 & 0 \\ b_{11} & a_{11} & b_{12} & a_{12} & 0 & 0 & 0 & 0 \\ \hline a_{21} & -b_{21} & a_{22} & -b_{22} & 0 & 0 & 0 & 0 \\ b_{21} & a_{21} & b_{22} & a_{22} & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & a_{32} & -b_{32} & a_{33} & -b_{33} & 0 & 0 \\ 0 & 0 & b_{32} & a_{32} & b_{33} & a_{33} & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & a_{43} & -b_{43} & a_{44} & -b_{44} \\ 0 & 0 & 0 & 0 & b_{43} & a_{43} & b_{44} & a_{44} \end{array}\right]$$

Sandia National Laboratories

NOX: Nonlinear Solvers

- Suite of nonlinear solution methods:
  - Uses abstract vector and “group” interfaces.
  - Allows flexible selection and tuning of various strategies:
    - Directions.
    - Line searches.
  - Epetra/AztecOO/ML, LAPACK, PETSc implementations of abstract vector/group interfaces.
- Designed to be easily integrated into existing applications.

Sandia National Laboratories

LOCA

- Library of continuation algorithms.
- Provides
  - Zero order continuation
  - First order continuation
  - Arc length continuation
  - Multi-parameter continuation (via Henderson's MF Library)
  - Turning point continuation
  - Pitchfork bifurcation continuation
  - Hopf bifurcation continuation
  - Phase transition continuation
  - Eigenvalue approximation (via ARPACK or Anasazi)

Amesos: Direct Solver Wrappers

- Direct sparse solver use at Sandia:
  - Salinas (Structures): DSCPACK (Raghavan), SPOOLES (Ashcraft), others.
  - Xyce (Circuits): Kundert, SuperLU serial.
  - PCx (LP): DSCPACK, WSMP.
  - Numerous other uses.
- Amesos contains wrapper classes to important third party direct sparse solvers:
  - Use Epetra objects.
  - Provide data redistribution capabilities (e.g., replication).
  - Provide common look-and-feel across variety of solvers.
  - Provide common resource for direct solvers at Sandia.
  - Presently wraps: DscPack, KLU, Mumps, SuperLU, UMFPack, Kundert

Goal: "Make solving $Ax=b$ as easy as it sounds."

Sandia National Laboratories

Epetraext: Extensions to Epetra

- Library of useful classes not needed by everyone.
- Most classes are types of “transforms”.
- Examples:
  - Graph/matrix view extraction.
  - Epetra/Zoltan interface.
  - Explicit sparse transpose.
  - Singleton removal filter.
  - Static condensation filter.
  - Overlapped graph constructor.
  - Graph colorings.
  - Permutations.
  - Sparse matrix-matrix multiply.
  - Matlab, MatrixMarket I/O functions.
  - …
- Most classes are small, useful, but non-trivial to write.

Meros

- Meros: Preconditioner package for incompressible NS problems.
  - Addresses problems: $Ax = b$.
  - where: $$A = \begin{bmatrix} F & B \\ C & 0 \end{bmatrix}$$
  - Makes use of TSF to orchestrate use of:
    - ML, Epetra, AztecOO, Ifpack.
  - Provides rapidly-developed, scalable implementation of state-of-the-art preconditioner.
  - Results shown below.

Sandia National Laboratories

Belos and Anasazi

- Next generation linear solvers (Belos) and eigensolvers (Anasazi) libraries, written in templated C++.
- Provide a generic interface to a collection of algorithms for solving large-scale linear problems and eigenproblems.
- Algorithm implementation is accomplished through the use of abstract base classes. Interfaces are derived from these base classes to matrix-vector products, status tests, and any arbitrary linear algebra library.

New Package: Kokkos

- Goal:
  - Isolate key non-BLAS kernels for the purposes of optimization.
- Kernels:
  - Dense vector/multivector updates and collective ops (not in BLAS).
  - Sparse MV, MM, SV, SM.
- Serial-only for now.
- Reference implementation provided.
- Mechanism for improving performance:
  - Default is aggressive compilation of reference source (Fortran loops).
  - BeBOP: Jim Demmel, Kathy Yelick, Rich Vuduc, UC Berkeley.
  - Vector version: Cray.

Sandia National Laboratories

Teuchos

- Utility package of commonly useful tools:
  - ParameterList class: key/value pair database, recursive capabilities.
  - LAPACK, BLAS wrappers (templated on ordinal and scalar type).
  - Dense matrix and vector classes (compatible with BLAS/LAPACK).
  - FLOP counters, Timers.
  - Ordinal, Scalar Traits support: Definition of ‘zero’, ‘one’, etc.
  - Reference counted pointers, and more…
- Takes advantage of advanced features of C++:
  - Templates
  - Standard Template Library (STL)
- ParameterList:
  - Allows easy control of solver parameters.

Teuchos BLAS/LAPACK

- Teuchos provides C++ wrappers for BLAS/LAPACK Fortran routines
- Insulates users of Teuchos from details of Fortran function calls
- Instead of using simple wrappers, Teuchos provides templated wrappers
- Two main benefits of using a templated interface
  - Template Specialization
    - Fortran BLAS supports 4 datatypes:
      - Single- and double- precision real, single- and double-precision complex
    - We can use template specialization to support these datatypes using existing Fortran BLAS code
  - General Implementation
    - We can write a generic implementation in C++ that can handle many arbitrary datatypes

NewPackage Package

- NewPackage provides jump start to develop/integrate a new package.
- NewPackage is a “Hello World” program and website:
  - Simple but it does work with autotools.
  - Compiles and builds.
- NewPackage directory contains:
  - Commonly used directory structure: src, test, doc, example, config.
  - Working autotools files.
  - Documentation templates (doxygen).
  - Working regression test setup.
- Substantially cuts down on:
  - Time to integrate new package.
  - Variation in package integration details.
  - Development of website.

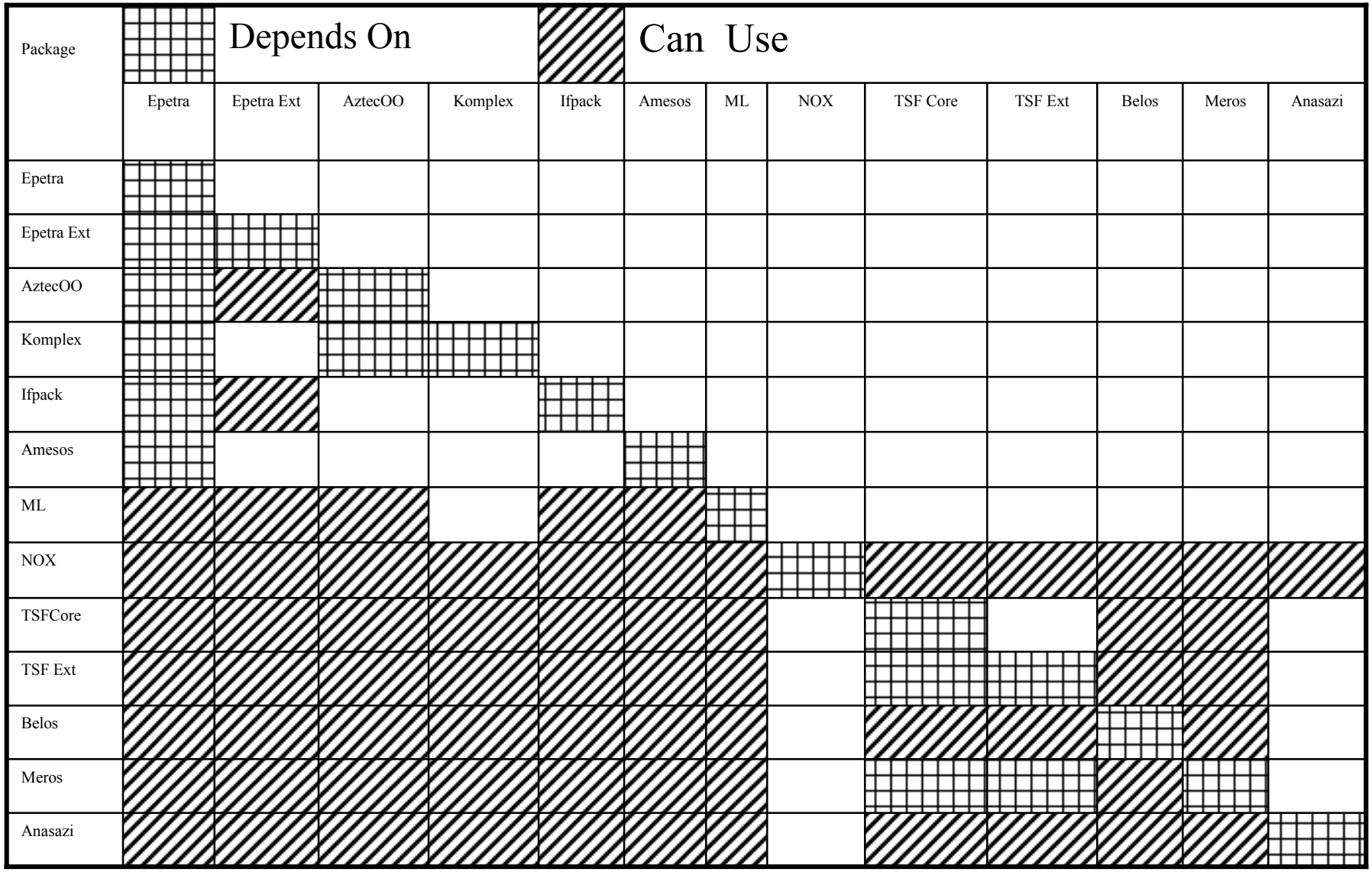
What do algorithms developers get from the Trilinos package architecture?

Pliris

- Migration of Linpack benchmark code to supported environment.
- Used NewPackage as starting point.
- Put under source control for the first time.
- Portable build process.
- Can accept linear problem as Epetra data.
- Has its own Bugzilla product, mail lists.
- Source code browsable via Bonsai.
- Portable interface to BLAS/LAPACK.
- And more…
- But Pliris is still an independent piece of code…and better off after the process.

Sandia National Laboratories

Trilinos Package Interoperability (Release 4)



Based on this chart:

- AztecOO depends on Epetra, but Epetra is independent of AztecOO
- NOX can use Epetra, but is independent of Epetra.

Sandia National Laboratories

Applications

Sandia National Laboratories

Meros

- **Block preconditioners for incompressible flow**

$$\begin{bmatrix} F & \nabla \\ \nabla\cdot & C \end{bmatrix} = \begin{bmatrix} I & 0 \\ \nabla\cdot F^{-1} & I \end{bmatrix} \begin{bmatrix} F & 0 \\ 0 & S \end{bmatrix} \begin{bmatrix} I & F^{-1}\nabla \\ 0 & I \end{bmatrix}$$

- where $S = C - \nabla \cdot F^{-1} \nabla$

- Easier to apply MG on blocks than whole system
  ⇒ **scalability!**

- Close to traditional pressure-Poisson type solvers
- Approximation choices ⇒ different methods

3D Driven Cavity Results

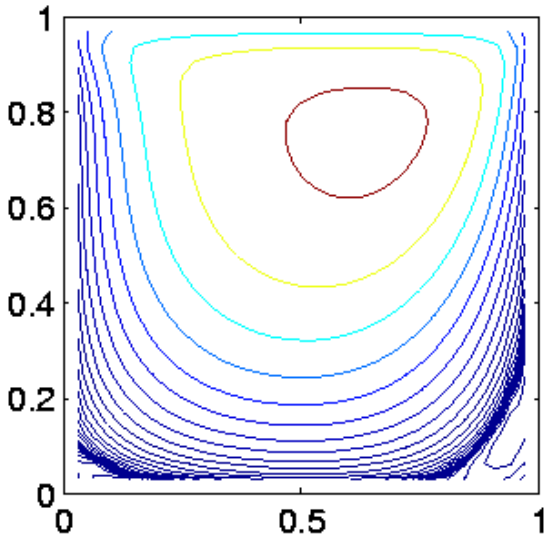
CFL insensitivity

Re	CFL	Oseen (K&L)	Newton (Oseen K&L)	Newton (BJB)
500	5000	5	24	4
	10000	5	36	6

3D Driven Cavity

h-independence

Grids	16^3	32^3	64^3
its @ Re=100	15	17	18

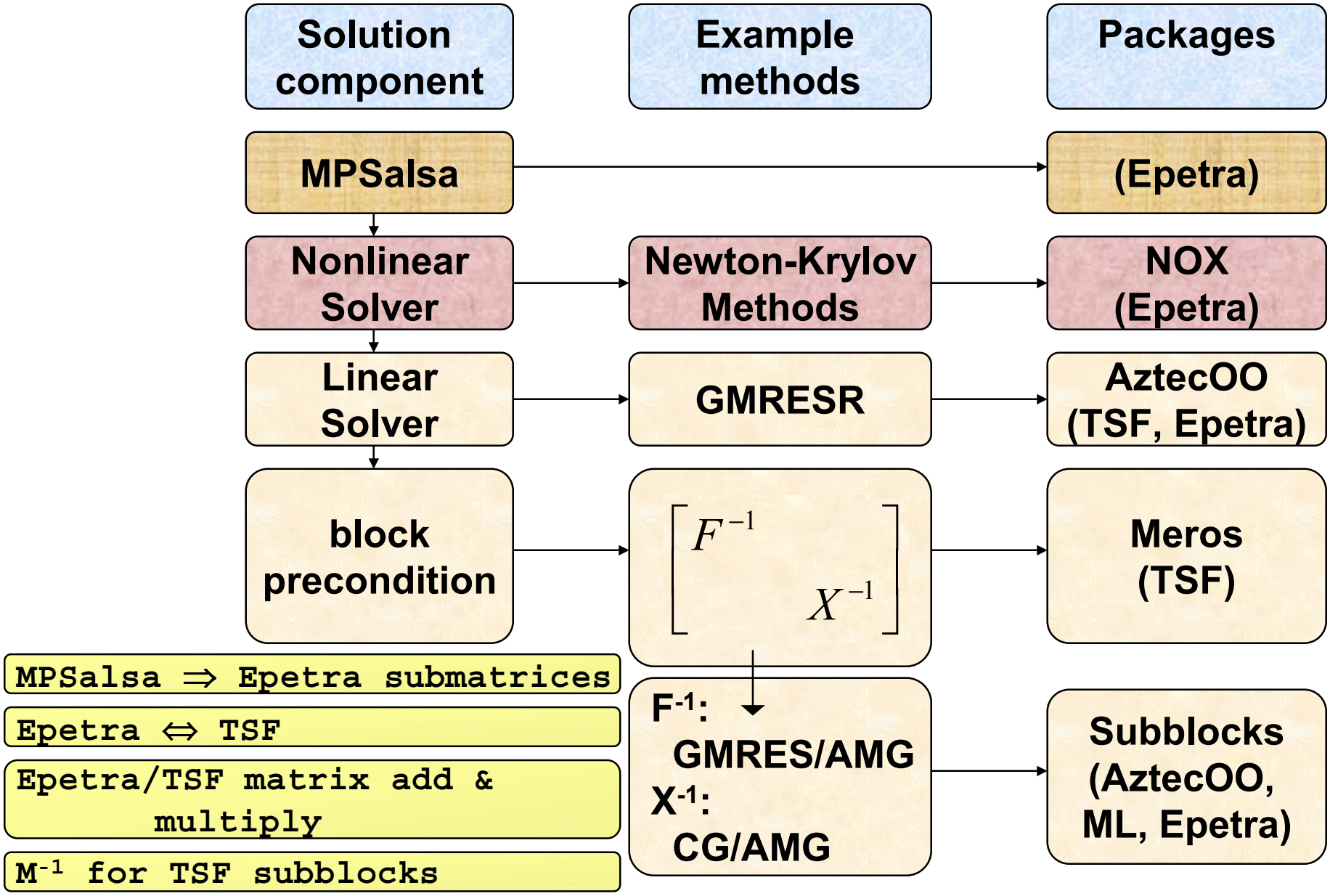


Re sensitivity?

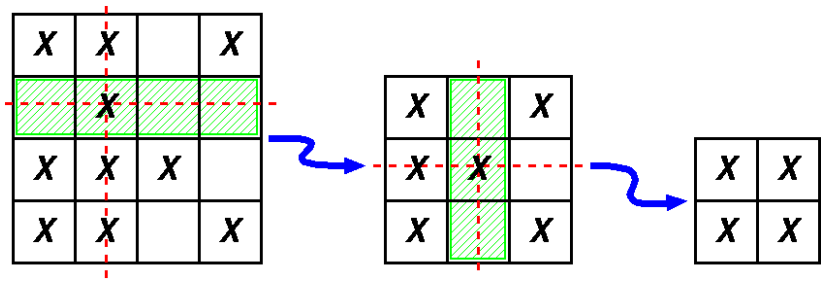
IMPACT Block preconditioning is a practical & efficient way to achieve scalable linear solvers.

Potential Applications: Aria, Charon, Fuego, Goma, MPSalsa, Sundance

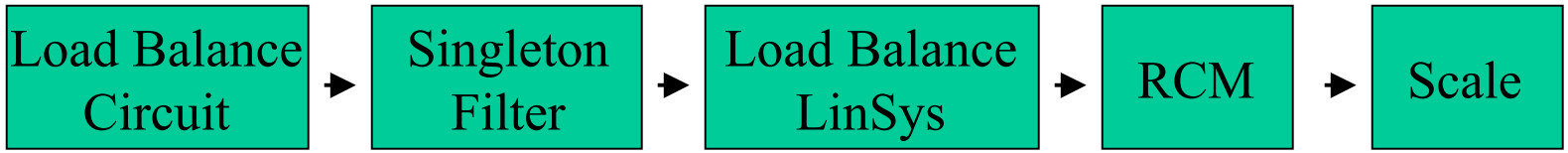
Inter-operation of Trilinos packages



Circuit Solvers
Bringing Multiple Tools Together



- Reduction to "Essential" System
  - Constraints ('Singleton' Rows)
  - Auxiliary Equations ('Singleton' Columns)
  - Based on Achim Basermann's Results (NEC Europe)
- Available in EpetraExt.



	N	Total Cuts	CondEst	GMRES Iters	LinSolve Time	Newton Steps
	1220	~1000	3.00E+05	500	4.7	72
LBC	1220	54	2.00E+05	200	1.1	45
LBC+SF+LBL+RCM+SCALE	1054	68	1.00E+04	127	0.36(0.43)	53

Sandia National Laboratories

ARPREC

- The ARPREC library uses arrays of 64-bit floating-point numbers to represent high-precision floating-point numbers.
- ARPREC values behave just like any other floating-point datatype, except the maximum working precision (in decimal digits) must be specified before any calculations are done
  - `mp::mp_init(200);`
- Illustrate the use of ARPREC with an example using Hilbert matrices.

Hilbert Matrices

- A Hilbert matrix $\boldsymbol{H_N}$ is a square $\boldsymbol{N}$-by-$\boldsymbol{N}$ matrix such that:

- For Example: $$H_{N_{ij}} = \frac{1}{i + j - 1}$$

$$H_3 = \begin{bmatrix} 1 & 1/2 & 1/3 \\ 1/2 & 1/3 & 1/4 \\ 1/3 & 1/4 & 1/5 \end{bmatrix}$$

Sandia National Laboratories

Hilbert Matrices

- Notoriously ill-conditioned
  - $\kappa(H_3) \approx 524$
  - $\kappa(H_5) \approx 476610$
  - $\kappa(H_{10}) \approx 1.6025 \times 10^{13}$
  - $\kappa(H_{20}) \approx 7.8413 \times 10^{17}$
  - $\kappa(H_{100}) \approx 1.7232 \times 10^{20}$
- Hilbert matrices introduce large amounts of error

Hilbert Matrices and Cholesky Factorization

- With double-precision arithmetic, Cholesky factorization will fail for $\boldsymbol{H}_N$ for all $N > 13$.
- Can we improve on this using arbitrary-precision floating-point numbers?

Precision	Largest N for which Cholesky Factorization is successful
Single Precision	**8**
Double Precision	**13**
Arbitrary Precision (20)	**29**
Arbitrary Precision (40)	**40**
Arbitrary Precision (200)	**145**
Arbitrary Precision (400)	**233+ (out of memory to go higher)**

Sandia National Laboratories

Preliminary Performance Analysis

Time (sec) of GEMV

N	double	mp_real(40)	Ratio mp_real/double
100	0.0001257 sec	0.0437751sec	349
500	0.0045598 sec	1.0673 sec	234
1000	0.093067 sec	4.31996 sec	46.5

Sandia National Laboratories

Potential Benefits of Templated Types

Templated scalar types have great potential:

- Efficient: Allow expression of algorithms over any abstract field.
- Can facilitate variety of algorithmic studies.
- Allow application developers to study asymptotic behavior of discretizations.
- Can facilitate debugging: Reduces FP error as source error.
- Use your imagination…

SQA/SQE

- Software Quality Assurance/Engineering is important.
- Not sufficient to convince ourselves that "we do a good job".
  - At start of development: Must state the practices we will follow.
  - At end of development: Must provide evidence that we followed practices.
- Trilinos facilitates SQA/SQE development/processes for packages:
  - 32 of 47 ASCI SQE practices are directly handled by Trilinos (no requirements on packages).
  - Trilinos provides significant support for the remaining 15.
  - Trilinos Dev Guide Part II: Specific to ASCI requirements.
  - Trilinos software engineering policies provide a ready-made infrastructure for new packages.
  - Trilinos philosophy:
    Few *requirements*. Instead mostly *suggested practices*. Provides package with option to provide alternate process.

Trilinos Availability/Support

- Trilinos and related packages are available via LGPL.
- Current release (3.1) is “click release”. Unlimited availability.
- Release 4 scheduled for May 2004.
- New platform facilitates development and support:
  - http://software.sandia.gov
  - Location of cvs repository, bugzilla, bonzai and mailman servers.
  - Accessible from anywhere via ssh/scp.
  - Documentation (generated via doxygen) is all available online.

Mailman Mail Lists

- Each Trilinos package, including Trilinos itself, has four mail lists:
  - package-checkins@software.sandia.gov
    - CVS commit emails.
  - package-developers@software.sandia.gov
    - Mailing list for developers.
  - package-users@software.sandia.gov
    - Issues for package users.
  - package-announce@software.sandia.gov
    - Releases and other announcements specific to the package.
- Additional list: Trilinos-Leaders@software.sandia.gov
- http://software.sandia.gov/mailman/listinfo/

Conclusions

- Trilinos services to developers and users:
  - The 3 I's: Infrastructure, Interfaces, Implementations.
  - Simplifies installation, support for users of total collection.
  - Epetra & TSF promote common APIs across all other Trilinos packages.
  - Each package can be built, used independently, and exists as independent project.
- **Primary goals:**
  - **Rapid development and installation of robust numerical solvers.**
  - **High-quality production software for the critical path.**
- Can SW infrastructure make our jobs easier? Yes!
  - NewPackage, mail lists, bugzilla, CVS, Bonsai
    - Make you an instant SW engineering professional.
  - Package architecture always respects your independence.
    - Compatability from augmentation not conformance.
    - You are free to "walk away" at any time.

Sandia National Laboratories

More information

- http://software.sandia.gov
- http://software.sandia.gov/trilinos
- Additional documentation at my website: http://www.cs.sandia.gov/~mheroux.

Sandia National Laboratories