Linear Algebra Libraries

Claire Mouton

Linear Algebra Libraries

Claire Mouton
claire.mouton@inria.fr

March 2009

Contents

I Requirements 3

II CPPLapack 4

III Eigen 5

IV Flens 6

V Gmm++ 7

VI GNU Scientific Library (GSL) 8

VII IT++ 9

VIII Lapack++ 10

IX Matrix Template Library (MTL) 11

X PETSc 13

XI Seldon 14

XII SparseLib++ 15

XIII Template Numerical Toolkit (TNT) 16

XIV Trilinos 17

XV uBlas 19

XVI Other Libraries 20

XVII Links and Benchmarks 22

1 Links 22

2 Benchmarks 22
2.1 Benchmarks for Linear Algebra Libraries 22
2.2 Benchmarks including Seldon 23
2.2.1 Benchmarks for Dense Matrix 23
2.2.2 Benchmarks for Sparse Matrix 26

XVIII Appendix 27

3 Flens Overloaded Operator Performance Compared to Seldon 27

4 Flens, Seldon and Trilinos Content Comparisons 29
4.1 Available Matrix Types from Blas (Flens and Seldon) 29
4.2 Available Interfaces to Blas and Lapack Routines (Flens and Seldon) . . . 30
4.3 Available Interfaces to Blas and Lapack Routines (Trilinos) 35

5 Flens and Seldon Synoptic Comparison 36

Part I
Requirements

This document has been written to help in the choice of a linear algebra library to be included in Verdandi, a scientific library for data assimilation. The main requirements are

1. Portability: Verdandi should compile on BSD systems, Linux, MacOS, Unix and Windows. Beyond the portability itself, this often ensures that most compilers will accept Verdandi. An obvious consequence is that all dependencies of Verdandi must be portable, especially the linear algebra library.
2. High-level interface: the dependencies should be compatible with the building of the high-level interface (e. g. with SWIG, this implies that headers (.hxx) have to be separated from sources (.cxx)).
3. License: any dependency must have a license compatible with Verdandi licenses (GPL and LGPL).
4. C++ templates, sparse matrices and sparse vectors have to be supported.

Information reported here was collected from December 2008 to March 2009.

Part II
CPPLapack

CPPLapack is a C++ class wrapper for Blas and Lapack.

CPPLapack	Distribution beta – Mar. 2005	Doc. updated in Mar. 2005
http://cpplapack.sourceforge.net/		
in development	4 developers	First release: Apr. 2004
License: GPL (GPL compatible)		

Types of matrices and vectors available
– Real double-precision and complex double-precision vectors and matrices
– Real double-precision and complex double-precision band, symmetric and sparse matrices

Linear algebra operations available
– Eigenvalues computation
– Linear systems solving
– SVD decomposition

Interface with other packages
– Blas
– Lapack

Performance
Almost the same as the performance of original Blas and Lapack

Portability: platforms and compilers supported
Platform independent

Installation
Requires Blas and Lapack

Miscellaneous
– A few bugs and unsupported Blas and Lapack functions

Limitations
– Templates not supported, float not supported
– Alpha version of sparse matrix classes with bugs (the authors advise developers not to use these classes in their code)
– No sparse vectors
– No separation between headers and sources

Part III
Eigen

Eigen is a C++ template library for linear algebra, part of the KDE project.

Eigen	Distribution 2.0-beta6 – soon released	Doc. updated in Jan. 2009
http://eigen.tuxfamily.org/		
in development	7 contributors, 2 developers	First release: Dec. 2006
License: LGPL and GPL (GPL compatible)		

Types of matrices and vectors available
– Dense and sparse matrices and vectors
– Plain matrices/vectors and abstract expressions
– Triangular and diagonal matrices
– Column-major (the default) and row-major matrix storage

Linear algebra operations available
– Triangular, SVD, Cholesky, QR and LU solvers
– Eigen values/vectors solver for non-selfadjoint matrices
– Hessemberg decomposition
– Tridiagonal decomposition of a selfadjoint matrix

Interface with other packages
For sparse matrices: TAUCS, umfpack, cholmod and SuperLU

Performance
Very efficient, see benchmark: http://eigen.tuxfamily.org/index.php?title=Benchmark

Portability: platforms and compilers supported
Standard C++ 98, compatible with any compliant compiler such as
– GCC, version 3.3 and newer
– MSVC (Visual Studio), 2005 and newer
– ICC, recent versions
– MinGW, recent versions

Installation
No dependency

Miscellaneous
– Templates supported
– Todo includes: interface to Lapack and eigensolver in non-selfadjoint case
– Examples of users:
KDE related projects such as screensavers, kgllib, kglengine2d, solidkreator, painting and image editing
Avogadro, an opensource advanced molecular editor
VcgLib, C++ template library for the manipulation and processing of triangular and tetrahedral meshes
MeshLab, for the processing and editing of unstructured 3D triangular meshes and point cloud
The Yujin Robot company uses Eigen for the navigation and arm control of their next generation robots (switched from blitz, ublas and tvmet)

Limitations
– Sparse matrices and vectors still experimental
– Eigen 2 is a beta version (Eigen 1 is the old stable version)

Part IV
Flens

Flens (Flexible Library for Efficient Numerical Solutions) is a C++ library for scientific computing providing interface for Blas and Lapack. Flens intends to be the building block of choice for the creation of serious scientific software in C++.

Flens	Distribution RC1 – Jul. 2007	Doc. updated in Feb. 2008
http://flens.sourceforge.net		
in development	9 developers	First release: 2004
License: BSD License (GPL compatible)		

Types of matrices and vectors available
– General, triangular and symmetric matrix types
– Storage formats: full storage (store all elements), band storage (store only diagonals of a banded matrix), packed storage (store only the upper or lower triangular part)
– Sparse matrix types: general and symmetric, compressed row storage; random access for initialization

Linear algebra operations available
– Linear systems solving using QR factorization
– Cg and Pcg methods

Interface with other packages
Blas and Lapack

Performance
– Natural mathematical notation: e. g. `y += 2 * transpose(A) * x + 1.5 * b + c` without sacrificing performances (see section 3 in Appendix)
– Very efficient, see benchmarks: http://flens.sourceforge.net/session2/tut4.html, http://grh.mur.at/misc/sparselib_benchmark/report.html, http://flens.sourceforge.net/session1/tut9.html and section 2.2

Portability: platforms and compilers supported
– Tested on Mac OS X, Ubuntu Linux and a SUSE Opteron cluster
– GCC: version 4 or higher
– Intel C++ compiler (icc): version 9.1
– Pathscale (pathCC): GNU gcc version 3.3.1, PathScale 2.3.1 driver

Installation
Requires Blas, Lapack and CBlas

Miscellaneous
– Extensible: e. g. easy integration of user-defined matrix/vector types
– Flexible: e. g. generic programming of numerical algorithms
– Flens implements a view concept for vectors and dense matrices: a vector can reference a part of a vector, or also a row, column or diagonal of a matrix; you can apply to these views the same operations as for regular vectors and matrices.
– Templated matrices and vectors with several storage formats

Limitations
– Lack of portability: not recently tested on Windows (once compiled with Microsoft Visual Studio Express compiler with minor modifications in the Flens code)
– No eigenvalues computation
– No sparse vectors
– No hermitian matrices

Part V

Gmm++

Gmm++ is a generic matrix template library inspired by MTL and ITL.

Gmm++	Distribution 3.1 – Sep. 2008	Doc. updated in Sep. 2008
http://home.gna.org/getfem/gmm_intro.html		
in development	2 contributors and 2 developers	First release: Jun. 2002
License: LGPL (GPL compatible)		

Types of matrices and vectors available
Sparse, dense and skyline vectors and matrices

Linear algebra operations available
– Triangular solver, iterative generic solvers (Cg, BiCgStag, Qmr, Gmres) with preconditioners for sparse matrices (diagonal, based on MR iterations, ILU, ILUT, ILUTP, ILDLT, ILDLTT)
– Reference to sub-matrices (with sub-interval, sub-slice or sub-index) for any sparse dense or skyline matrix for read or write operations
– LU and QR factorizations for dense matrices
– Eigenvalues computation for dense matrices

Interface with other packages
– Blas, Lapack or Atlas for better performance
– SuperLU 3.0 (sparse matrix direct solver) for sparse matrices

Performance
Very efficient, see benchmarks: http://grh.mur.at/misc/sparselib_benchmark/report.html and http://eigen.tuxfamily.org/index.php?title=Benchmark

Portability: platforms and compilers supported
– Linux/x86 with g++ 3.x and g++ 4.x
– Intel C++ Compiler 8.0
– Linux/Itanium with g++
– MacOS X Tiger (with the python and matlab interface)
– Windows with MinGW and MSys (without the Python and Matlab interface)

Installation
No special requirement

Miscellaneous
– Templates supported
– Examples of users: IceTools, an open source model for glaciers; EChem++: a problem solving environment for electrochemistry
– Gmm++ is included in Getfem++, a generic and efficient C++ library for finite element methods, awarded by the second price at the "Trophées du Libre 2007" in the category of scientific softwares
– Provides a high-level interface to Python and Matlab via Mex-Files for Getfem++, covering some functionalities of Gmm++

Limitations
– No separation between headers and sources (only header files)
– No eigenvalues computation for sparse matrices
– Gmm++ primary aim is not to be a standalone linear algebra library, but is more aimed at interoperability between several linear algebra packages.

Part VI
GNU Scientific Library (GSL)

GSL is a numerical library for C and C++ programmers. The library provides a wide range of mathematical routines covering subject areas such as linear algebra, Blas support and eigensystems.

GSL	Distribution GSL-1.12 – Dec. 2008	Doc. updated in Dec. 2008
http://www.gnu.org/software/gsl/		
in development	18 developers	First release: 1996
License: GPL (GPL compatible)		

Types of matrices and vectors available
General vectors and matrices

Linear algebra operations available
– Eigenvalues and eigenvectors computation
– Functions for linear systems solving: LU Decomposition, QR Decomposition, SVD Decomposition, Cholesky Decomposition, Tridiagonal Decomposition, Hessenberg Decomposition, Bidiagonalization, Householder Transformations, Householder solver for linear systems, Tridiagonal Systems, Balancing

Interface with other packages
– Blas (level 1, 2 and 3)
– CBlas or Atlas
– Many extensions such as Marray, NEMO, LUSH (with full interfaces to GSL, Lapack, and Blas) and PyGSL

Performance
Not evaluated

Portability: platforms and compilers supported
– GNU/Linux with gcc – SunOS 4.1.3 and Solaris 2.x (Sparc) – Alpha GNU/Linux, gcc – HP-UX 9/10/11, PA-RISC, gcc/cc – IRIX 6.5, gcc – m68k NeXTSTEP, gcc – Compaq Alpha Tru64 Unix, gcc – FreeBSD, OpenBSD and NetBSD, gcc – Cygwin – Apple Darwin 5.4 – Hitachi SR8000 Super Technical Server, cc

Installation
Easy to compile without any dependencies on other packages

Miscellaneous

Limitations
– Can be called from C++ but written in C
– No sparse matrices and vectors
– Templates not supported

Part VII
IT++

IT++ is a C++ library of mathematical, signal processing and communication routines. Templated vector and matrix classes are the core of the IT++ library, making its functionality similar to that of MATLAB and GNU Octave. IT++ makes an extensive use of existing open source libraries (e. g. Blas, Lapack, ATLAS and FFTW).

IT++	Distribution 4.0.6 – Oct. 2008	Doc. updated in Oct. 2008
http://itpp.sourceforge.net/		
in development	19 contributors and 11 developers	First release: 2001
License: GPL (not LGPL compatible)		

Types of matrices and vectors available
– Diagonal, Jacobsthal, Hadamard and conference matrices
– Templated vectors and matrices
– Sparse vectors and matrices

Linear algebra operations available
– Matrix decompositions such as eigenvalue, Cholesky, LU, Schur, SVD and QR
– Linear systems solving: over- and underdetermined, LU factorization and Cholesky factorization

Interface with other packages
– Blas, Lapack and FFTW
– Optionally Atlas, MKL and ACML

Performance
Not evaluated

Portability: platforms and compilers supported
GNU/Linux, Sun Solaris, Microsoft Windows (with Cygwin, MinGW/MSYS or Microsoft Visual C++) and Mac OS X

Installation
Packages available (Fedora RPM, Debian GNU/Linux and openSUSE)

Miscellaneous
– Templates supported
– Separation between headers and sources

Limitations
– Its main use is in simulation of communication systems and for performing research in the area of communications. It is also used in areas such as machine learning and pattern recognition.
– One important high-performance feature missing in IT++ is the ability to create ”submatrix-views” and ”shallow copies” of matrices, i.e. pass submatrices by reference -instead of by value (compared to Lapack++).

Part VIII
Lapack++

Lapack++ is a library for high performance linear algebra computations.

Lapack++	Distribution beta 2.5.2 – Jul. 2007	Doc. updated in Jul. 2007
http://lapackpp.sourceforge.net/		
in development	7 developers	First release: 1993
License: LGPL (GPL compatible)		

Types of matrices and vectors available
– Int, long int, real and complex vectors and matrices
– Symmetric positive definite matrix
– Symmetric, banded, triangular and tridiagonal matrices

Linear algebra operations available
– Linear systems solving for non-symmetric matrices, symmetric positive definite systems and solving linear least-square systems; using LU, Cholesky and QR matrix factorizations
– Symmetric eigenvalues computation
– SVN and QR decompositions

Interface with other packages
– Blas
– Lapack

Performance
High performance linear algebra computation

Portability: platforms and compilers supported
– Linux/Unix: gcc2.95.x, gcc3.x and gcc4.x
– Windows 9x/NT/2000: MinGW and gcc3.x
– Windows 9x/NT/2000: Microsoft Visual Studio, .NET and MSVC
– Mac OS X

Installation
Requires Blas, Lapack and a Fortran compiler

Miscellaneous
– Template functions for matrices

Limitations
– Templates not supported, float supported only for general matrices
– No sparse matrices
– No sparse vectors

Part IX
Matrix Template Library (MTL)

MTL is a generic component library developed specially for high performance numerical linear algebra. MTL includes matrix formats and functionality equivalent to level 3 Blas.

MTL 4	Distribution alpha 1 – Oct. 2007	Doc. updated in Nov. 2008
http://www.osl.iu.edu/research/mtl/mtl4/		
in development	4 developers	First release: 1998 (MTL2)
License: Copyright Indiana University (can be modified to become GPL compatible)		

Types of matrices and vectors available
– Dense2D, morton_dense and sparse matrices
– Arbitrary types can be used for matrix elements (float, double, complex)

Linear algebra operations available
– Preconditioners: diagonal inversion, incomplete LU factorization without fill-in and incomplete Cholesky factorization without fill-in
– Solvers: triangular, conjugate gradient, BiCg, CgSquared and BiCgStab
– Iterative methods for solving linear systems thanks to the Iterative Template Library (ITL, last release in Oct. 2001): Chebyshev and Richardson iterations, generalized conjugate residual, generalized minimal residual and (transpose free) quasi-minimal residual without lookahead

Interface with other packages
Blas (optionally Blitz++ thanks to ITL)

Performance
– Natural mathematical notation without sacrificing performances: A = B * C dispatched to the appropriate Blas algorithm if available; otherwise an implementation in C++ is provided (also reasonably fast, usually reaching 60 percent peak)
– See benchmarks: http://grh.mur.at/misc/sparselib_benchmark/report.html, http://eigen.tuxfamily.org/index.php?title=Benchmark and http://projects.opencascade.org/btl/

Portability: platforms and compilers supported
Can be compiled and used on any target platform with an ANSI C++ compiler
– Linux: g++ 4.0.1, g++ 4.1.1, g++ 4.1.2, g++ 4.2.0, g++ 4.2.1, g++ 4.2.2, icc 9.0
– Windows: VC 8.0 from Visual Studio 2005
– Macintosh: g++ 4.0.1

Installation
Requires the Boost library included, optionally scons and a Blas library installed

Miscellaneous
– Templates and generic programming
– A generic library has been built on top of MTL (ITL: http://www.osl.iu.edu/research/itl/)
– Developed from scratch but inspired by the design and implementation details of MTL 2 (interfacing Lapack; supporting sparse, banded, packed, diagonal, tridiagonal, triangle, symmetric matrices)

Limitations
– No sparse vectors
– No eigenvalues computation

– No release since the alpha version in Oct. 2007

Part X

PETSc

PETSc, the Portable, Extensible Toolkit for Scientific computation, is a suite of data structures and routines for the scalable (parallel) solution of scientific applications modeled by partial differential equations.

PETSc	Distribution 2.3.3 – May 2007	Doc. updated in Jul. 2008
http://www.mcs.anl.gov/petsc/petsc-2/		
in development	11 developers	First release: Mar. 1999
License: Copyright University of Chicago (GPL compatible)		

Types of matrices and vectors available
– Parallel vectors and matrices
– Several sparse matrices storages
– Symmetric, block diagonal and sequential matrices

Linear algebra operations available
– Preconditioners: ILU, LU, Jacobi, block Jacobi, additive Schwartz and ICC
– Direct solvers: LU, Cholesky and QR
– Krylov subspace methods: GMRES, Chebychev, Richardson, conjugate gradients (Cg), CGSquared, BiCgStab, two variants of TFQMR, conjugate residuals and Lsqr
– Nonlinear solvers: Newton-based methods, line search and trust region
– Parallel timestepping solvers: Euler, Backward Euler and pseudo time stepping

Interface with other packages
Blas, Lapack, ADIC/ADIFOR, AnaMod, BlockSolve95, BLOPEX, Chaco, DSCPACK, ESSL, Euclid, FFTW, Hypre, Jostle, Mathematica, Matlab, MUMPS, ParMeTiS, Party, PaStiX, PLapack, Prometheus, Scotch, SPAI, SPOOLES, SPRNG, Sundial/CVODE, SuperLU, Trilinos/ML, UMFPACK

Performance
Optimal on parallel systems with high per-CPU memory performance

Portability: platforms and compilers supported
Any compiler supporting ANSI C standard on Unix or Windows

Installation
Requires Blas, Lapack, MPI and optional packages

Miscellaneous
– Related packages using PETSc such as TAO, SLEPc, Prometheus, OpenFVM, OOFEM, DEAL.II and Python bindings (petsc4py and LINEAL)
– Scientific applications in many fields such as nano-simulations, cardiology, imaging and surgery, geosciences, environment, computational fluid dynamics, wave propagation and optimization

Limitations
– No sparse vectors
– Not coded in C++ but in C
– Templates not supported, polymorphism used instead

Part XI
Seldon

Seldon is a C++ library for linear algebra. Seldon is designed to be efficient and convenient, which is notably achieved thanks to template classes.

Seldon	Distribution 2008-11-12 – Nov. 2008	Doc. updated in Nov. 2008
http://seldon.sourceforge.net/		
in development	2 developers	First release: Nov. 2004
License: GPL (GPL compatible)		

Types of matrices and vectors available
– Dense and sparse vectors
– Dense matrices: several formats for rectangular, symmetric, hermitian and triangular
– Two sparse matrix forms: Harwell-Boeing and array of sparse vectors
– 3D arrays

Linear algebra operations available
– Preconditioner of your own or by successive over-relaxations (SOR)
– Direct dense solvers: LU, QR, LQ and SVD decomposition
– Iterative solvers: BiCg, BiCgcr, BiCgStab, BiCgStabl, Cg, Cgne, Cgs, CoCg, Gcr, Gmres, Lsqr, MinRes, QCgs, Qmr, QmrSym, SYMmetric, Symmlq and TfQmr
– Eigenvalues and eigenvectors computation

Interface with other packages
– Fully interfaced with Blas (level 1, 2 and 3) and Lapack, except for functions with banded matrices
– Direct sparse solvers: MUMPS, SuperLU and UmfPackLU

Performance
Very efficient, see benchmarks in section 2.2

Portability: platforms and compilers supported
– Code fully compliant with the C++ standard
– OS Portable
– GNU GCC $\geq$ 3 (from 3.2 to 4.3 tested); Intel C++ compiler icc (icc 7.1 and 8.0 tested); compile with Microsoft Visual

Installation
Requires Blas and CBlas

Miscellaneous
– Thanks to templates, the solvers can be used for any type of matrix and preconditioning, not only Seldon matrices: very useful to perform a matrix-vector product when the matrix is not stored
– Provides a Python interface generated by SWIG
– Exception handling and several debug levels helpful while coding
– Good coverage of the interface to Blas and Lapack routines (see section 4)
– A few alternative functions provided in C++ if Blas is not available

Limitations
– No band matrices

Part XII

SparseLib++

SparseLib++ is a C++ class library for efficient sparse matrix computations across various computational platforms.

SparseLib++	Distribution v.1.7 – after 1996	Doc. updated in 1996
http://math.nist.gov/sparselib++/		
minimal maintenance, not in development	3 authors, 0 developers	First release: 1994
License: Public domain (GPL compatible)		

Types of matrices and vectors available
– Sparse double vectors
– Sparse double matrices with several storage formats: compressed row/column, compressed diagonal, coordinate formats, jagged diagonal, block compressed row and skyline

Linear algebra operations available
– Preconditioners: incomplete LU, incomplete Cholesky and diagonal scaling
– Iterative solvers: SOR, Chebyshev iteration, BiCg, BiBgStab, Cg, Cgne, Cgne, Cgnr, Gmres, MinRes and Qmr
– Sparse triangular system solver

Interface with other packages
Blas

Performance
Not evaluated

Portability: platforms and compilers supported
Various computational platforms

Installation
Requires Blas

Miscellaneous
– Built upon sparse Blas (level 3)
– SparseLib++ matrices can be built out of nearly any C++ matrix/vector classes (it is shipped with the MV++ classes by default)

Limitations
– Templates not supported (only double elements)
– No new feature since 1996, only maintenance
– No eigenvalues computation
– No dense vectors and matrices

Part XIII

Template Numerical Toolkit (TNT)

TNT is a collection of interfaces and reference implementations of numerical objects useful for scientific computing in C++. JAMA/C++ library (Java Matrix Package translated into C++) utilizes TNT for the lower-level operations to perform linear algebra operations.

TNT	Distribution v.3.0.11 – Jan. 2008	Doc. updated in 2003
http://math.nist.gov/tnt/		
active maintenance, not in development	2 authors, 0 developers	First release: Sep. 1999
License: Public domain (GPL compatible)		

Types of matrices and vectors available
– Sparse matrices
– 1D, 2D and 3D arrays (row-major and column-major)

Linear algebra operations available
Provided by JAMA/C++:
– SVD decomposition
– SVD, LU, QR and Cholesky solvers
– Eigenvalues computation

Interface with other packages
None

Performance
Not evaluated

Portability: platforms and compilers supported
ANSI C++ compatibility: should work with most updated C++ compilers (tested by the authors with Microsoft Visual C++ v.5.0)

Installation
By including header files

Miscellaneous
Templates supported

Limitations
– No sparse vectors
– No separation between headers and sources (only header files)
– Beta release since Jan. 2008

Part XIV
Trilinos

Trilinos provides algorithms and enabling technologies within an object-oriented software framework for large-scale, complex multi-physics engineering and scientific problems. Trilinos is a collection of interacting independent packages (package names are in italic).

Trilinos	Distribution 9.0.1 – Oct. 2008	Doc. updated in Oct. 2008
http://trilinos.sandia.gov/		
in development	34 developers	First release: Sept. 2003
License: LGPL (GPL compatible)		

Types of matrices and vectors available
– Core kernel package (*Kokkos*)
– Dense, symmetric dense, sparse, block sparse, jagged-diagonal sparse matrices (*Epetra* and *EpetraExt*)
– Dense vectors and multivectors (*Epetra* and *EpetraExt*), sparse vectors (*Tpetra*)
– Integer and double elements (*Epetra* and *EpetraExt*), templates (*Tpetra*)

Linear algebra operations available
– Preconditioners: ILU-type (*AztecOO* and *IFPACK*), Multilevel (*ML*), Block (*Meros*)

– Linear solvers:
Direct dense solvers (*Epetra* and *Teuchos*: wrappers for selected Blas and Lapack routines, *Pliris*: LU solver on parallel platforms);
Krylov solvers (*AztecOO*: preconditioned Krylov solver, *Belos*: iterative solver, *Komplex*: for complex values);
Direct sparse solvers (*Amesos*: for DSCPACK, SuperLU, SuperLUDist and UMFPACK);
SVD decomposition (*Epetra*)

– Nonlinear solvers:
System solvers (*NOX*: globalized Newton methods such as line search and trust region methods, *LOCA*: computing families of solutions and their bifurcations for large-scale applications);
Optimization (*MOOCHO*: reduced-space successive quadratic programming (SQP) methods);
Time integration/Differential-algebraic equations (*Rythmos*)

– Eigensolver: block Krylov-Schur, block Davidson and locally-optimal block preconditioned conjugate gradient (*Anasazi*)

Interface with other packages
– Uses Blas and Lapack
– Provides interfaces for Metis/ParMetis, SuperLU, Aztec, Mumps, Umfpack and soon PETSc
– Conjunction possible with SWIG, MPI, Expat (XML parser), METIS and ParMETIS

Performance
Epetra provides classes to distribute vectors, matrices and other objects on a parallel (or serial) machine

Portability: platforms and compilers supported
Linux, MAC OS X, Windows (under Cygwin), SGI64, DEC and Solaris

Installation
Requires Blas and Lapack

Miscellaneous
– Examples of users: SIERRA (Software Environment for Developing Complex Multiphysics Applications), SALINAS (structural dynamics finite element analysis), MPSalsa (high resolution 3D simulations with an equal emphasis on fluid flow and chemical kinetics modeling), Sundance (finite-element solutions of partial differential equations), DAKOTA (Design Analysis Kit for Optimization and Terascale Applications ; coding for uncertainty quantification, parameter estimation and sensitivity/variance analysis)
– Most of Trilinos functionalities available in a Python script
– A package of basic structures with templated types (*Tpetra*, first release distributed with 9.0.1)
– *EpetraExt* enables objects to be easily exported to MATLAB
– Trilinos is based on established algorithms at Sandia. The effort required to develop new algorithms and enabling technologies is substantially reduced because a common base provides an excellent starting point. Furthermore, many applications are standardizing on the Trilinos APIs: these applications have access to all Trilinos solver components without any unnecessary interface modifications.

Limitations
– Not yet available: templates (except in an isolated package, *Teuchos*) and sparse vectors (Tpetra is still under heavy development, release planned in Mar./Apr. 2009)
– Impossible to build Trilinos under Windows without Cygwin (improved Windows support in a further release)
– Trilinos is a complex collection of interoperable packages and requires some careful configuration (with a suitable set of packages and options)

Part XV
uBlas

uBlas is a C++ template class library that provides Blas level 1, 2, 3 functionality for dense, packed and sparse matrices.

uBlas	Distribution 1.33.0 – Jul. 2008	Doc. updated in 2008
http://www.boost.org/doc/libs/1_35_0/libs/numeric/ublas/		
in development	5 developers	First release: 2008
License: Boost Software License (GPL compatible)		

Types of matrices and vectors available
– Dense, unit and sparse (mapped, compressed or coordinate) vectors
– Dense, identity, triangular, banded, symmetric, hermitian, packed and sparse (mapped, compressed or coordinate) matrices

Linear algebra operations available
– Submatrices and subvectors operations
– Triangular solver
– LU factorization

Interface with other packages
Blas (level 1, 2 and 3)

Performance
Optimized for large vectors and matrices, see benchmarks: http://flens.sourceforge.net/session2/tut4.html, http://flens.sourceforge.net/session1/tut9.html, http://eigen.tuxfamily.org/index.php?titl http://projects.opencascade.org/btl/ and and section 2.2

Portability: platforms and compilers supported
OS Independent, requires a modern (ISO standard compliant) compiler such as GCC 3.2.3, 3.3.x, 3.4.x, 4.0.x; MSVC 7.1, 8.0; ICC 8.0, 8.1; Visual age 6; Codewarrior 9.4, 9.5

Installation
Requires Blas

Miscellaneous
– Templates supported
– Included in Boost C++ libraries
– Mathematical notation to ease the development (use of operator overloading)

Limitations
– No eigenvalues computation
– Only basic linear algebra (no linear solving except triangular solver)
– The implementation assumes a linear memory address model
– Tuning focussed on dense matrices
– No separation between headers and sources (only header files)

Part XVI
Other Libraries

Armadillo++ is a C++ linear algebra library providing matrices and vectors, interfacing Lapack and Atlas. : http://arma.sourceforge.net/.
Limitations — No templates (only double), early development (first release in Apr. 2008), no portability under Windows without Cygwin.

Blitz++ is a C++ class library for scientific computing providing high performance by using template techniques. The current versions provide dense arrays and vectors, random number generators, and small vectors and matrices: http://www.oonumerics.org/blitz/.
Limitations — No sparse matrices and vectors. Barely relevant for linear algebra.

CPPScaLapack is a C++ class wrapper for BLACS, PBlas and ScaLapack with MPI. CPPScaLapack provides a user-friendly interface of high-speed parallel matrix calculation with Blas and Lapack techonologies for programers concerning with large-scale computing: http://cppscalapack.sourceforge.net/.
Limitations — Still an alpha program, no sparse matrices and vectors, no templates (only double-precision vectors and general matrices).

CVM Class Library provides vector and different matrices including square, band, symmetric and hermitian ones in Euclidean space of real and complex numbers. It utilizes Blas and Lapack. Contains different algorithms including solving of linear systems, singular value decomposition, non-symmetric and symmetric eigenvalue problem (including Cholesky and Bunch-Kaufman factorization), LU factorization, QR factorization: http://www.cvmlib.com/.
Limitations — Templates supported but distinction between real and complex types. No sparse matrices and vectors.

IML++ (Iterative Methods Library) is a C++ templated library for solving linear systems of equations, capable of dealing with dense, sparse, and distributed matrices: http://math.nist.gov/iml++/.
Limitations — No matrices and vectors, only iterative solvers.

LA library is a C++ interface to Blas and Lapack providing also a general (sparse) matrix diagonalization, linear solver and exponentiation templates : http://www.pittnerovi.com/la/.
Limitations — Portability: tested only on Linux with a code not fully ANSI C++ compliant.

LinAl The library is based on STL techniques and uses STL containers for the storage of matrix data furthermore STL algorithms are used where feasible. Low level, algebraic operators as well as linear solvers and eigenvalue solvers are implemented, based on calls to Blas, Lapack and CGSOLX and LANCZOS: http://linal.sourceforge.net/LinAl/Doc/linal.html.
Limitations — No vectors, no sparse matrices.

LinBox is a C++ template library for exact, high-performance linear algebra computation with dense, sparse, and structured matrices over the integers and over finite fields: http://www.linalg.org/.
Limitations — Does not suit to real and complex values.

MV++ is a small set of concrete vector and matrix classes specifically designed for high performance numerical computing, including interfaces to the computational kernels found in Blas: http://math.nist.gov/mv++/.
Limitations — Only building blocks to a larger-user level library such as SparseLib++ and Lapack++.

Newmat C++ library supports various matrix types. Only one element type (float or double) is supported. The library includes Cholesky decomposition, QR, triangularisation, singular value decomposition and eigenvalues of a symmetric matrix: http://www.robertnz.net/nm_intro.htm.
Limitations — No sparse matrices and vectors, templates not supported.

RNM by Frédéric Hecht provides C++ classes for arrays with vectorial operations: http://www.ann.jussieu.fr/~hecht/.
Limitations — Only general matrices, no sparse matrices, no vectors, only one linear system solver (conjugate gradient), no English documentation.

SL++ (Scientific Library) is a C++ numerical library, composed of modules specialized in various fields of numerical computations: http://ldeniau.home.cern.ch/ldeniau/html/sl++.html.
Limitations — Not developed since 1998.

TCBI (Temporary Base Class Idiom) templated C++ numerical library implements basic data structures like complex numbers, dynamic vectors, static vectors, different types of matrices like full matrices, band matrices, sparse matrices, etc. It also includes some standard matrix solvers like Gauss-Jordan, LU-decomposition and Singular Value Decomposition and a set of powerful iterative solvers (Krylov subspace methods along with preconditioners). Also interfaces to netlib libraries such as CLapack or SuperLU. Its specificity is being exclusively written in C++, without needing to interface to Fortran code. The usual loss of performance associated with object-oriented languages has been avoided through not as obvious implementations of numerical base classes, avoiding unnecessary copying of objects. It can be thought of as some sort of reference counting done by the compiler at compile time. Supported on Linux/Unix with the GNU compiler, on Windows with the Microsoft Visual C++ (6, 7) compiler and with the Intel compiler: http://plasimo.phys.tue.nl/TBCI/.
Limitations — No interface to Blas and Lapack.

Part XVII
Links and Benchmarks

1 Links

Here are some Web portals related to numerical analysis software and linear algebra libraries:

– List of numerical analysis software (Wikipedia) http://en.wikipedia.org/wiki/List_of_numerical_analysis_software

– Numerical computing resources on the Internet (Indiana University) http://www.indiana.edu/~statmath/bysubject/numerics.html

– Scientific computing in object-oriented languages (community resources) http://www.oonumerics.org/oon/

– Scientific computing software (Master's Degrees in Applied Mathematics at École Centrale Paris) http://norma.mas.ecp.fr/wikimas/ScientificComputingSoftware

2 Benchmarks

2.1 Benchmarks for Linear Algebra Libraries

– Freely available software for linear algebra on the web – 2006 comparative statement http://www.netlib.org/utk/people/JackDongarra/la-sw.html

– Benchmark sparse matrices (tests for residual and random order initialization) – 2008 http://flens.sourceforge.net/session2/tut4.html

– Some Blas Benchmarks – 2007 http://flens.sourceforge.net/session1/tut9.html

– Benchmark 2008 http://eigen.tuxfamily.org/index.php?title=Benchmark

– Benchmark of C++ Libraries for Sparse Matrix Computation – 2007 http://grh.mur.at/misc/sparselib_benchmark/report.html

– Benchmark for Templated Libraries project – 2003 http://projects.opencascade.org/btl/

2.2 Benchmarks including Seldon

Platform: Intel Core 2 Duo CPU P9500, 2.53GHz, 6 MB cache, 4 GB Ram.
Compiler: gcc version 4.3.2 (Ubuntu 4.3.2-1ubuntu12).
Date: March 2009.

2.2.1 Benchmarks for Dense Matrix

Adapted from http://flens.sourceforge.net/session1/tut9.html

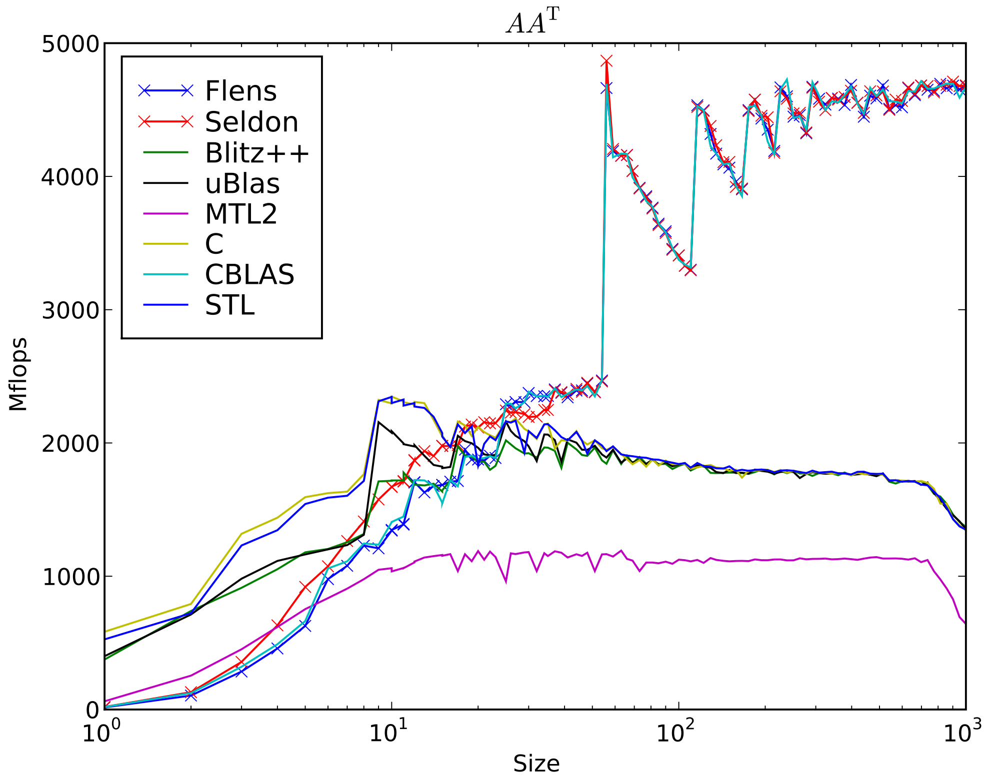

Figure 1: A x At product for dense matrices.

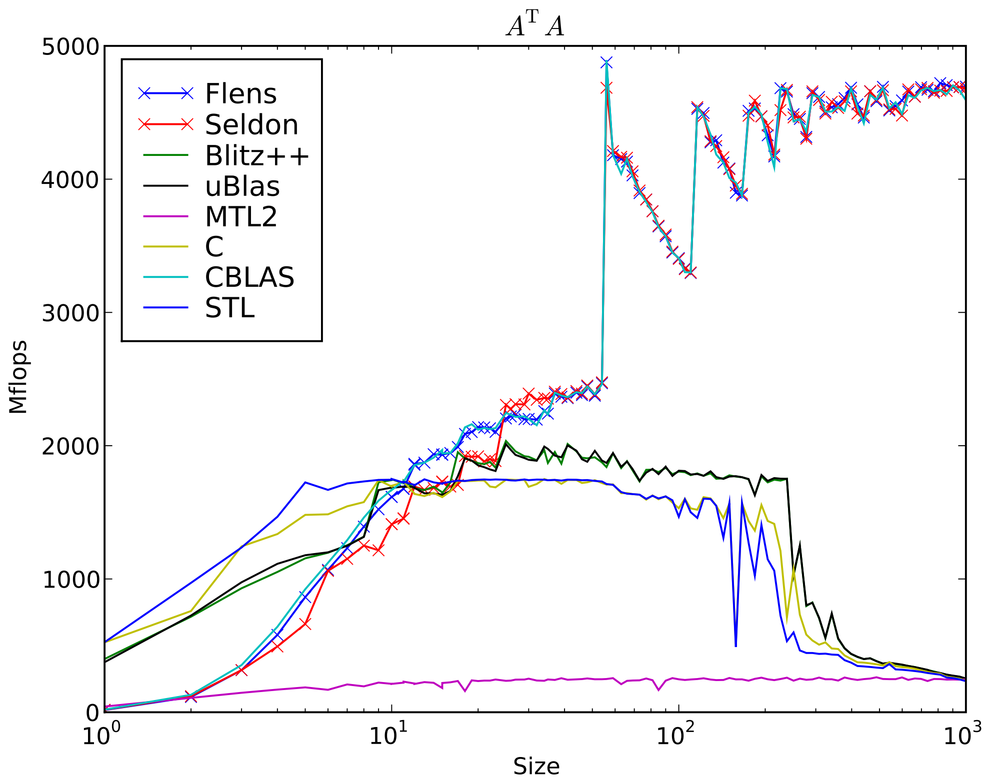

Figure 2: At x A product for dense matrices.

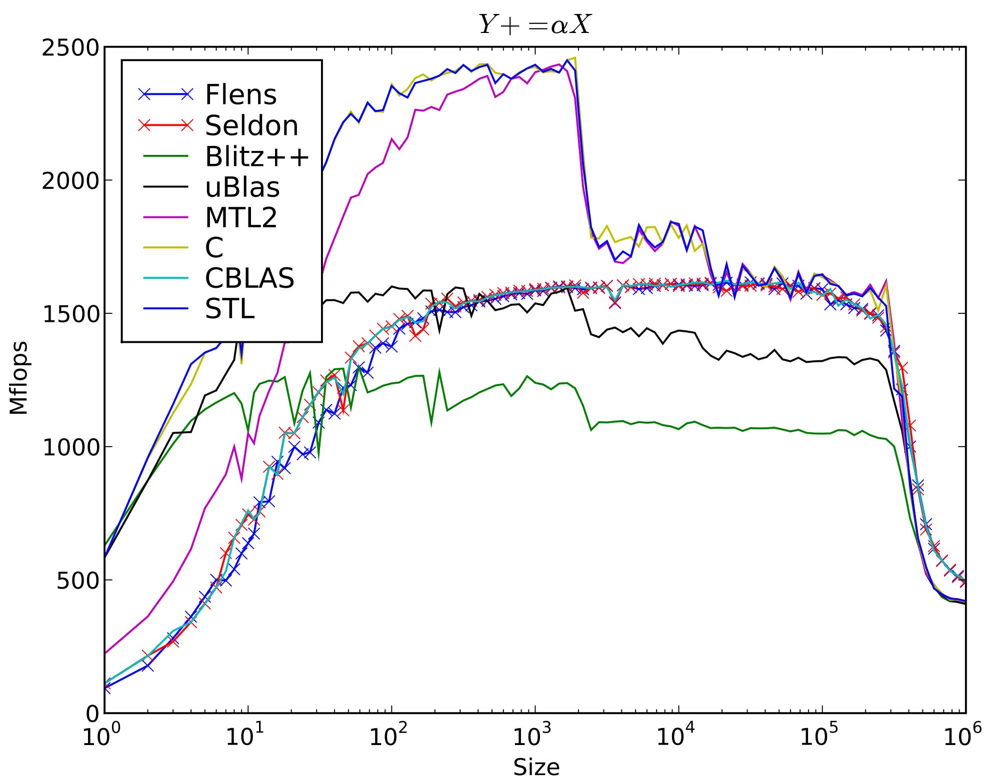

Figure 3: Y += alpha * X for dense vectors.

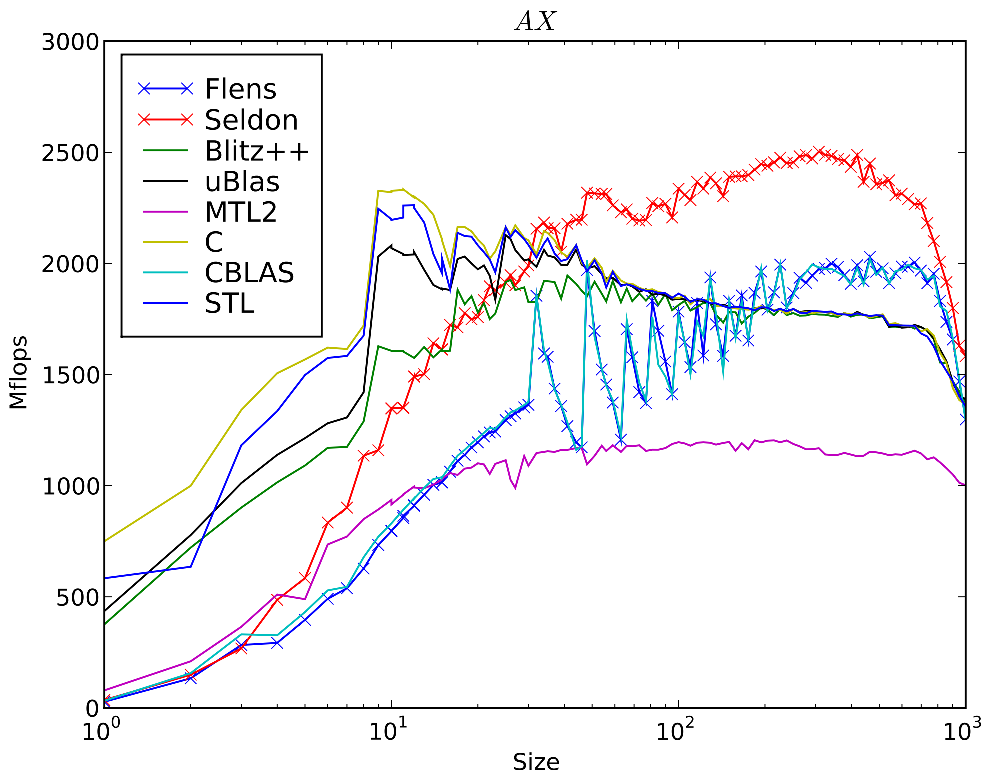

Figure 4: Matrix vector product for dense matrices and vectors.

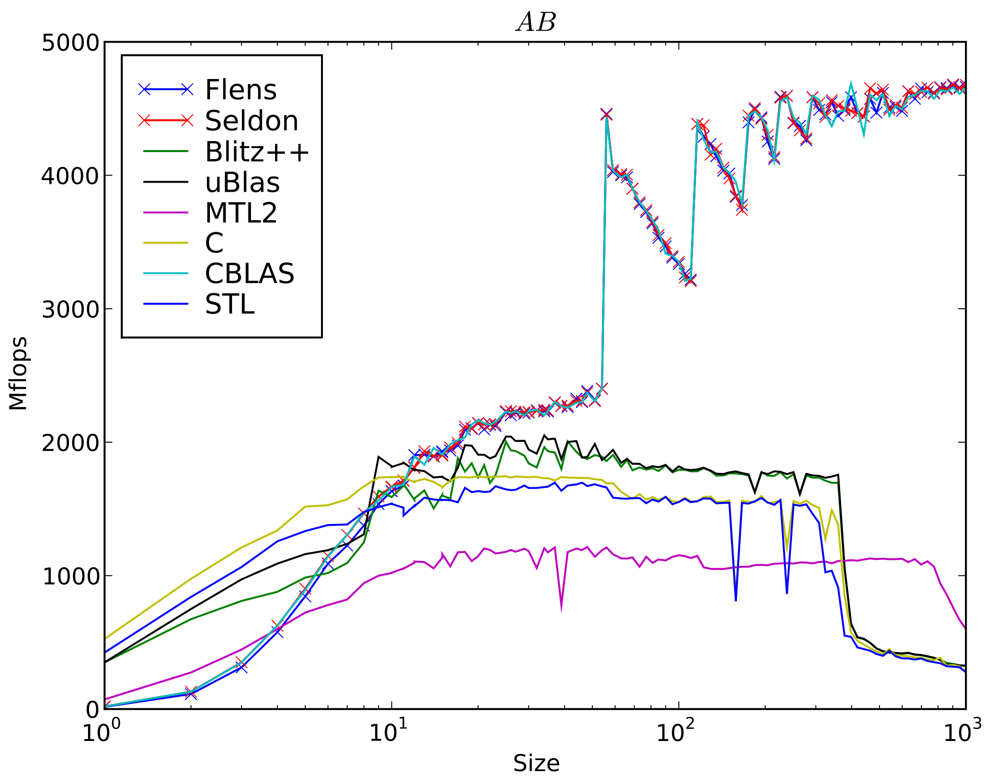

Figure 5: Matrix matrix product for dense matrices.

2.2.2 Benchmarks for Sparse Matrix

Adapted from http://flens.sourceforge.net/session2/tut4.html.
Compiled with -DNDEBUG -O3 options for g++.
Matrices of 1 000 000 lines.

1. Benchmark for sparse matrix, vector product with random initialization
   5 non zero elements per line on average. In CRS Flens format, the number of non zero values stored corresponds to the number of non zero values plus the number of empty lines (one 0 is inserted on the diagonal).

	Flens	Seldon
initialization finalization	0.35s 1.23s	1.48s
y = Ax y = A'x	1.28s 1.61s	1.3s 1.49s

2. Benchmark with initialization in order and tridiagonal matrix
   3 non zero elements per line (except on first and last lines with only 2 non zero elements).

	Flens	Seldon (matrix built by hand)	Seldon (matrix built with a generic algorithm *)
initialization finalization	0.28s 0.16s	0.08s	0.1s
y = Ax y = A'x	0.19s 0.22s	0.18s 0.22s	0.18s 0.22s

   *using MergeSort on already sorted vectors, instead of QuickSort used in random initialization case.

3. Benchmark for computation of the residual
   5 non zero elements per line on average.

	Flens	Seldon
initialization finalization	1.34s 1.25s	2.46s
computations r = b - Ax	1.39s	1.35s

Acknowledgement

This document benefits from discussions with Vivien Mallet, Dominique Chapelle and Philippe Moireau, and also from corrections thanks to Xavier Clerc.

Part XVIII
Appendix

These sections deal with Flens distribution RC1 (Jul. 2007), Seldon distribution 2008-11-12 and Trilinos distribution 9.0.1 (Oct. 2008). The first two libraries satisfy the main requirements exposed in section I, and Trilinos was supposed to be a good reference as for the coverage of Blas and Lapack.

3 Flens Overloaded Operator Performance Compared to Seldon

Flens implements a user-friendly interface to Blas routines by overloading operators. For example, one can add vectors by using mathematical symbols: `y = a + b + c`. Here are a few tests to check if this does not imply any loss of performance in the computation.

In table 1 are presented CPU times measured with `std::clock()` for several operations, for RowMajor (R.) and ColMajor (C.) matrices. In the following, lower-case letters denote vectors and upper-case letters denote matrices. `Y = A + B` cannot be used for sparse matrices in Flens, so this has been tested only on general dense matrices.

In ColumnMajor format for Flens, the affectation operation `Y = A` costs much more (1.48s) than in RowMajor format (0.35s). This lack of performance slows down sum operations such as `Y = A + B`. This is due to a direct call to Blas to copy the data (a memory block copy in C++ is used in Seldon).

In ColumnMajor for Seldon, matrix vector product such as `Y += transpose(A)*B` is three times longer than in Flens. This is due to a call to a generic function instead of the Blas routine. Taking advantage of a local knowledge of Seldon (one of its authors, Vivien Mallet, is part of my team!), we have fixed this bug and got the same results as in Flens for `y += transpose(A)*b` and `y +=2*transpose(A)*b + 1.5*c + d`.

Table 1: Flens and Seldon performance benchmarks using overloaded operators.

Operation	Flens code	R.(s)	C.(s)	Seldon code	R.(s)	C.(s)
y = a + b + c	y = a + b + c;	1.16	1.17	y = a; Add(b,y); Add(c,y);	1.13	1.12
	y = a; y += b; y += c;	1.16	1.18			
	y = a + b; y += c;	1.17	1.18			
y = a + b + c + d	y=a+b+c+d;	1.58	1.59	y=a; Add(b,y); Add(c,y); Add(d,y)	1.52	1.54
	y = a; y += b; y += c; y += d;	1.6	1.58			
y += 4a	y += 4a;	4.11	4.07	Add(4.,a,y);	4.09	4.09
	axpy(4, a, y);	4.09	4.1			
Y = A	Y = A;	0.35	1.48	Y = A;	0.32	0.32
Y = A + B	Y = A + B;	0.78	2.92	Y = A; Add(B,Y);	0.76	1.89
	Y = A; Y += B;	0.78	2.93			
Y = A + B + C	Y = A + B + C;	1.21	4.41	Y = A; Add(B,Y); Add(C,Y);	1.24	3.44
	Y = A + B; Y += C;	1.22	4.35			
	Y = A; Y += B; Y += C;	1.21	4.35			
Y = A + B + C + D	Y = A + B + C + D;	1.63	5.78	Y = A; Add(B,Y); Add(C,Y); Add(D,Y);	1.7	4.95
	Y = A; Y+=B; Y+=C; Y+=D;	1.62	5.78			
y = Ax + b	y = Ax + b;	1.64	2.09	Copy(b,y); MltAdd(1.,A,x,1.,y);	1.61	2.08
y += A*b	y += A*b;	1.63	2.07	MltAdd(1.,A,b,1.,y);	1.63	2.08
y += transpose(A)*b	y += transpose(A)*b;	2.05	1.63	MltAdd(1.,SeldonTrans,A,b,1.,y);	6.64/2.05*	1.53
y+=2transpose(A)*b+1.5c+d	y += 2*transpose(A)*b+1.5c+d;	2.06	1.64	MltAdd(2.,SeldonTrans,A,b,1.,y);Add(1.5,c,y);Add(d,y);	6.66/2.05*	1.52
C += 1.5A*transpose(B)	C += 1.5A*transpose(B);	43.18	43.3	MltAdd(1.5,SeldonNoTrans, A,SeldonTrans,B,1.,C);	43.1	43.26

* with the correction (see above for the explanation).

4 Flens, Seldon and Trilinos Content Comparisons

4.1 Available Matrix Types from Blas (Flens and Seldon)

F stands for Flens, S stands for Seldon. A black cell for an existing structure; a gray cell for no structure.

	F	S		F	S		F	S
GE - General			GB - General Band					
SY - SYmmetric			SB - Sym. Band			SP - Sym. Packed		
HE - HErmitian			HB - Herm. Band			HP - Herm. Packed		
TR - TRiangular			TB - Triang. Band			TP - Triang. Packed		

4.2 Available Interfaces to Blas and Lapack Routines (Flens and Seldon)

A black cell for an existing interface to a given routine; a gray cell for no interface.

1. Blas routines

	Flens	Seldon
srotg		■
srotmg		■
csrot		
sswap		■
sscal	■	■
csscal		■
ccopy		■
caxpy		■
sdsdot		■
cdotc		■
scnrm2		■
scasum		■
icamax		■
cgemv		■
cgbmv		
chbmv		
ssymv	■	■
sspmv	■	■
ctrmv		■
ctbmv		
ctpmv		■
ctrsv		■
ctbsv		
ctpsv		■
cgeru		■
cher		■
cher2		
ssyr		■
ssyr2		
sgemm	■	■
ssymm	■	■
chemm		■
csyrk		
ssyr2k		
cher2k		
ctrmm		■
ctrsm		■

	Flens	Seldon
drotg		■
drotmg		■
zdrot		
dswap		■
dscal	■	■
zdscal		■
zcopy	■	■
zaxpy	■	■
dsdot		■
zdotc		■
dznrm2	■	■
dzasum		■
izamax	■	■
zgemv	■	■
zgbmv		
zhbmv		
dsymv	■	■
dspmv	■	■
ztrmv		■
ztbmv		
ztpmv		■
ztrsv		■
ztbsv		
ztpsv		■
zgeru		■
zher		■
zher2		
dsyr		■
dsyr2		
dgemm	■	■
dsymm	■	■
zhemm		■
zsyrk		
dsyr2k		
zher2k		
ztrmm		■
ztrsm		■

	Flens	Seldon
crotg		
srot		■
srotm		■
cswap		■
cscal		■
scopy	■	■
saxpy	■	■
sdot	■	■
cdotu		■
snrm2	■	■
sasum		■
isamax	■	■
sgemv	■	■
sgbmv	■	
chemv		■
chpmv		■
ssbmv	■	
strmv	■	■
stbmv	■	
stpmv	■	■
strsv		■
stbsv		
stpsv		■
sger		■
cgerc		■
chpr		■
chpr2		■
sspr		■
sspr2		■
cgemm		■
csymm		■
ssyrk		
cherk		
csyr2k		
strmm	■	■
strsm	■	■

	Flens	Seldon
zrotg		
drot		■
drotm		■
zswap		■
zscal	■	■
dcopy	■	■
daxpy	■	■
ddot	■	■
zdotu		■
dnrm2	■	■
dasum		■
idamax	■	■
dgemv	■	■
dgbmv	■	
zhemv		■
zhpmv		■
dsbmv	■	
dtrmv	■	■
dtbmv	■	
dtpmv	■	■
dtrsv	■	■
dtbsv		
dtpsv		■
dger		■
zgerc		■
zhpr		■
zhpr2		■
dspr		■
dspr2		■
zgemm	■	■
zsymm		■
dsyrk		
zherk		
zsyr2k		
dtrmm	■	■
dtrsm	■	■

Blas	Flens	Seldon
Total	44	110
Coverage	30%	75%

2. Single precision real Lapack routines

	Flens	Seldon		Flens	Seldon		Flens	Seldon		Flens	Seldon
sgesv	■	■	sgbsv	■		sgtsv			sposv		
sppsv			spbsv			sptsv			ssysv		
sspsv			sgels	■		sgelsd			sgglse		
sggglm			ssyev		■	ssyevd			sspev		■
sspevd			ssbev			ssbevd			sstev		
sstevd			sgees			sgeev	■	■	sgesvd	■	■
sgesdd			ssygv		■	ssygvd			sspgv		■
sspgvd			ssbgv			ssbgvd			sgegs		
sgges			sgegv			sggev		■	sggsvd		
sgesvx			sgbsvx			sgtsvx			sposvx		
sppsvx			spbsvx			sptsvx			ssysvx		
sspsvx			sgelsx			sgelsy			sgelss	■	
ssyevx			ssyevr			ssygvx			sspevx		
sspgvx			ssbevx			ssbgvx			sstevx		
sstevr			sgeesx			sggesx			sgeevx		
sggevx			sbdsdc			sbdsqr			sdisna		
sgbbrd			sgbcon			sgbequ			sgbrfs		
sgbtrf	■		sgbtrs	■		sgebak			sgebal		
sgebrd			sgecon		■	sgeequ		■	sgehrd		
sgelqf		■	sgeqlf			sgeqp3			sgeqpf		
sgeqrf	■	■	sgerfs		■	sgerqf			sgetrf	■	■
sgetri	■	■	sgetrs	■	■	sggbak			sggbal		
sgghrd			sggqrf			sggrqf			sggsvp		
sgtcon			sgtrfs			sgttrf			sgttrs		
shgeqz			shsein			shseqr			sopgtr		
sopmtr			sorgbr			sorghr			sorglq		
sorgql			sorgqr	■		sorgrq			sorgtr		
sormbr			sormhr			sormlq		■	sormql		
sormqr	■	■	sormr3			sormrq			sormrz		
sormtr			spbcon			spbequ			spbrfs		
spbstf			spbtrf			spbtrs			spocon		
spoequ			sporfs			spotrf			spotri		
spotrs			sppcon			sppequ			spprfs		
spptrf			spptri			spptrs			sptcon		
spteqr			sptrfs			spttrf			spttrs		
ssbgst			ssbtrd			sspcon		■	sspgst		
ssprfs		■	ssptrd			ssptrf		■	ssptri		■
ssptrs		■	sstebz			sstedc			sstegr		
sstein			ssteqr			ssterf			ssycon		■
ssygst			ssyrfs		■	ssytrd			ssytrf		■
ssytri		■	ssytrs		■	stbcon			stbrfs		
stbtrs			stgevc			stgexc			stgsen		
stgsja			stgsna			stgsyl			stpcon		■
stprfs		■	stptri		■	stptrs		■	strcon		■
strevc			strexc			strrfs		■	strsen		
strsna			strsyl			strtri		■	strtrs	■	■
stzrqf			stzrzf								

Single real Lapack	Flens	Seldon
Total	15	36
Coverage	8%	19%

3. Double precision real Lapack routines

	Flens	Seldon		Flens	Seldon		Flens	Seldon		Flens	Seldon
dgesv	■	■	dsgesv			dgbsv	■		dgtsv		
dposv			dppsv			dpbsv			dptsv		
dsysv			dspsv			dgels	■		dgelsd		
dgglse			dggglm			dsyev		■	dsyevd		
dspev		■	dspevd			dsbev			dsbevd		
dstev			dstevd			dgees			dgeev	■	■
dgesvd	■	■	dgesdd			dsygv		■	dsygvd		
dspgv		■	dspgvd			dsbgv			dsbgvd		
dgegs			dgges			dgegv			dggev		■
dggsvd			dgesvx			dgbsvx			dgtsvx		
dposvx			dppsvx			dpbsvx			dptsvx		
dsysvx			dspsvx			dgelsx			dgelsy		
dgelss	■		dsyevx			dsyevr			dsygvx		
dspevx			dspgvx			dsbevx			dsbgvx		
dstevx			dstevr			dgeesx			dggesx		
dgeevx			dggevx			dbdsdc			dbdsqr		
ddisna			dgbbrd			dgbcon			dgbequ		
dgbrfs			dgbtrf	■		dgbtrs	■		dgebak		
dgebal			dgebrd			dgecon		■	dgeequ		■
dgehrd			dgelqf		■	dgeqlf			dgeqp3		■
dgeqpf			dgeqrf	■	■	dgerfs		■	dgerqf		
dgetrf	■	■	dgetri	■	■	dgetrs	■	■	dggbak		
dggbal			dgghrd			dggqrf			dggrqf		
dggsvp			dgtcon			dgtrfs			dgttrf		
dgttrs			dhgeqz			dhsein			dhseqr		
dopgtr			dopmtr			dorgbr			dorghr		
dorglq			dorgql			dorgqr	■	■	dorgrq		
dorgtr			dormbr			dormhr			dormlq		■
dormql			dormqr	■	■	dormr3			dormrq		
dormrz			dormtr			dpbcon			dpbequ		
dpbrfs			dpbstf			dpbtrf			dpbtrs		
dpocon			dpoequ			dporfs			dpotrf		
dpotri			dpotrs			dppcon			dppequ		
dpprfs			dpptrf			dpptri			dpptrs		
dptcon			dpteqr			dptrfs			dpttrf		
dpttrs			dsbgst			dsbtrd			dspcon		■
dspgst			dsprfs		■	dsptrd			dsptrf		■
dsptri		■	dsptrs		■	dstebz			dstedc		
dstegr			dstein			dsteqr			dsterf		
dsycon		■	dsygst			dsyrfs		■	dsytrd		
dsytrf		■	dsytri		■	dsytrs		■	dtbcon		
dtbrfs			dtbtrs			dtgevc			dtgexc		
dtgsen			dtgsja			dtgsna			dtgsyl		
dtpcon		■	dtprfs		■	dtptri		■	dtptrs		■
dtrcon		■	dtrevc			dtrexc			dtrrfs		■
dtrsen			dtrsna			dtrsyl			dtrtri		■
dtrtrs	■	■	dtzrqf			dtzrzf					

Double real Lapack	Flens	Seldon
Total	15	38
Coverage	8%	20%

4. Single precision complex Lapack routines

	Flens	Seldon		Flens	Seldon		Flens	Seldon		Flens	Seldon
cgesv		■	cgbsv			cgtsv			cposv		
cppsv			cpbsv			cptsv			csysv		
chesv			cspsv			chpsv			cgels		
cgelsd			cgglse			cggglm			cheev		■
cheevd			cheevr			chpev		■	chpevd		
chbev			chbevd			cgees			cgeev	■	■
cgesvd		■	cgesdd			chegv		■	chegvd		
chpgv		■	chpgvd			chbgv			chbgvd		
cgegs			cgges			cgegv			cggev		■
cggsvd			cgesvx			cgbsvx			cgtsvx		
cposvx			cppsvx			cpbsvx			cptsvx		
csysvx			chesvx			cspsvx			chpsvx		
cgelsx			cgelsy			cgelss			cheevx		
cheevr			chegvx			chpevx			chpgvx		
chbevx			chbgvx			cgeesx			cggesx		
cgeevx			cggevx			cbdsdc			cbdsqr		
cgbbrd			cgbcon			cgbequ			cgbrfs		
cgbtrf			cgbtrs			cgebak			cgebal		
cgebrd			cgecon		■	cgeequ		■	cgehrd		
cgelqf			cgeqlf			cgeqp3			cgeqpf		
cgeqrf			cgerfs		■	cgerqf			cgetrf		■
cgetri		■	cgetrs		■	cggbak			cggbal		
cgghrd			cggqrf			cggrqf			cggsvp		
cgtcon			cgtrfs			cgttrf			cgttrs		
chgeqz			chsein			chseqr			cupgtr		
cupmtr			cungbr			cunghr			cunglq		
cungql			cungqr			cungrq			cungtr		
cunmbr			cunmhr			cunmlq			cunmql		
cunmqr			cunmr3			cunmrq			cunmrz		
cunmtr			cpbcon			cpbequ			cpbrfs		
cpbstf			cpbtrf			cpbtrs			cpocon		
cpoequ			cporfs			cpotrf			cpotri		
cpotrs			cppcon			cppequ			cpprfs		
cpptrf			cpptri			cpptrs			cptcon		
cpteqr			cptrfs			cpttrf			cpttrs		
chbgst			chbtrd			cspcon		■	chpcon		■
chpgst			csprfs		■	chprfs		■	chptrd		
csptrf		■	chptrf		■	csptri		■	chptri		■
csptrs		■	chptrs		■	cstedc			cstegr		
cstein			csteqr			csycon		■	checon		■
chegst			csyrfs		■	cherfs		■	chetrd		
csytrf		■	chetrf		■	csytri		■	chetri		■
csytrs		■	chetrs		■	ctbcon			ctbrfs		
ctbtrs			ctgevc			ctgexc			ctgsen		
ctgsja			ctgsna			ctgsyl			ctpcon		■
ctprfs		■	ctptri		■	ctptrs		■	ctrcon		■
ctrevc			ctrexc			ctrrfs		■	ctrsen		
ctrsna			ctrsyl			ctrtri		■	ctrtrs		■
ctzrqf			ctzrzf								

Single complex Lapack	Flens	Seldon
Total	1	42
Coverage	1%	22%

5. Double precision complex Lapack routines

	Flens	Seldon		Flens	Seldon		Flens	Seldon		Flens	Seldon
zgesv		■	zgbsv			zgtsv			zposv		
zppsv			zpbsv			zptsv			zsysv		
zhesv			zspsv			zhpsv			zgels	■	
zgelsd			zgglse			zggglm			zheev		■
zheevd			zhpev		■	zhpevd			zhbev		
zhbevd			zgees			zgeev	■	■	zgesvd		■
zgesdd			zhegv		■	zhegvd			zhpgv		■
zhpgvd			zhbgv			zhbgvd			zgegs		
zgges			zgegv			zggev		■	zggsvd		
zgesvx			zgbsvx			zgtsvx			zposvx		
zppsvx			zpbsvx			zptsvx			zsysvx		
zhesvx			zspsvx			zhpsvx			zgelsx		
zgelsy			zgelss	■		zheevx			zheevr		
zhegvx			zhpevx			zhpgvx			zhbevx		
zhbgvx			zgeesx			zggesx			zgeevx		
zggevx			zbdsdc			zbdsqr			zgbbrd		
zgbcon			zgbequ			zgbrfs			zgbtrf		
zgbtrs			zgebak			zgebal			zgebrd		
zgecon		■	zgeequ		■	zgehrd			zgelqf		■
zgeqlf			zgeqp3			zgeqpf			zgeqrf		■
zgerfs		■	zgerqf			zgetrf	■	■	zgetri	■	■
zgetrs		■	zggbak			zggbal			zgghrd		■
zggqrf			zggrqf			zggsvp			zgtcon		
zgtrfs			zgttrf			zgttrs			zhgeqz		■
zhsein			zhseqr			zupgtr			zupmtr		
zungbr			zunghr			zunglq			zungql		
zungqr		■	zungrq			zungtr			zunmbr		
zunmhr			zunmlq		■	zunmql			zunmqr		■
zunmr3			zunmrq			zunmrz			zunmtr		
zpbcon			zpbequ			zpbrfs			zpbstf		
zpbtrf			zpbtrs			zpocon			zpoequ		
zporfs			zpotrf			zpotri			zpotrs		
zppcon			zppequ			zpprfs			zpptrf		
zpptri			zpptrs			zptcon			zpteqr		
zptrfs			zpttrf			zpttrs			zhbgst		
zhbtrd			zspcon		■	zhpcon		■	zhpgst		
zsprfs		■	zhprfs		■	zhptrd			zsptrf		■
zhptrf		■	zsptri		■	zhptri		■	zsptrs		■
zhptrs		■	zstedc			zstegr			zstein		
zsteqr			zsycon		■	zhecon		■	zhegst		
zsyrfs		■	zherfs		■	zhetrd			zsytrf		■
zhetrf		■	zsytri		■	zhetri		■	zsytrs		■
zhetrs		■	ztbcon			ztbrfs			ztbtrs		
ztgevc			ztgexc			ztgsen			ztgsja		
ztgsna			ztgsyl			ztpcon		■	ztprfs		■
ztptri		■	ztptrs		■	ztrcon		■	ztrevc		
ztrexc			ztrrfs		■	ztrsen			ztrsna		
ztrsyl			ztrtri		■	ztrtrs		■	ztzrqf		
ztzrzf											

Double complex Lapack	Flens	Seldon
Total	5	49
Coverage	3%	25%

4.3 Available Interfaces to Blas and Lapack Routines (Trilinos)

As a rough guide, here are some results for Trilinos. Several Trilinos packages offer at least a partial interface to Blas and Lapack routines: *Epetra*, *Teuchos*, *Amesos*, *AztecOO*, *ML*, *MOOCHO* and *Pliris*. The Trilinos column refers to all the packages tested together. This result is the maximum coverage and one should be careful for its interpretation. Indeed, some packages may not communicate together, with non compatible structures, and therefore could not be used together. Moreover, some of the interfaces may not be usable directly, only indirectly through other functions.

Blas	Trilinos	*Epetra*	*Teuchos*	*Amesos*	*AztecOO*	*ML*	*MOOCHO*	*Pliris*
Total	87	28	62	18	21	29	34	28
Coverage	60%	19%	42%	12%	14%	20%	23%	19%

Single real Lapack	Trilinos	*Epetra*	*Teuchos*	*Amesos*	*AztecOO*	*ML*	*MOOCHO*	*Pliris*
Total	57	44	44	4	8	32	1	1
Coverage	31%	24%	24%	2%	4%	17%	1%	1%

Double real Lapack	Trilinos	*Epetra*	*Teuchos*	*Amesos*	*AztecOO*	*ML*	*MOOCHO*	*Pliris*
Total	59	43	44	4	7	32	7	0
Coverage	32%	23%	24%	2%	4%	17%	4%	0%

Single complex Lapack	Trilinos	*Epetra*	*Teuchos*	*Amesos*	*AztecOO*	*ML*	*MOOCHO*	*Pliris*
Total	39	1	38	1	1	1	0	0
Coverage	20%	1%	20%	1%	1%	1%	0%	0%

Double complex Lapack	Trilinos	*Epetra*	*Teuchos*	*Amesos*	*AztecOO*	*ML*	*MOOCHO*	*Pliris*
Total	38	0	38	1	0	0	0	0
Coverage	20%	0%	20%	1%	0%	0%	0%	0%

5 Flens and Seldon Synoptic Comparison

	Flens		Seldon		See section
	—	+	—	+	
Portability	Not recently tested on Windows				
High-level interface				Python interface generated by SWIG	
C++ templates		Supported		Supported	
Matrix types	No hermitian matrices		No band matrices		
Sparse matrices		General and symmetric, compressed row storage		Harwell-Boeing and array of sparse vectors	
Sparse vectors	Not Supported			Supported	
Syntax		Natural mathematical notation			
Maintenance	Release candidate RC1 in Jul. 2007, last commit Jan. 2009			Latest release in Nov. 2008	
Blas and Lapack interface				Good coverage	4
Performance	Costly affectation in ColumnMajor format		One bug (fixed in later versions)	Better performance for dense matrix vector product	2.2 and 3
Vector and matrix views		Supported	Not supported		
Technical mastery				One author in my INRIA team!	
Eigenvalues computation	Not Supported			Eigenvalues and eigenvectors computation	