

sparsix
corporation

Mathematics *empowered*

Analysis Applications

- Power Grid Modeling
- Oil Reservoir Modeling
- FEA / CFD Analysis
- Financial Markets Modeling
- Scheduling Optimization
- Pipeline Flow Optimization

Sparse Iterative Solvers

SparSol

Systems with real coefficients

LinCoS

Systems with complex coefficients

Preprocessing

- Scaling
- Reordering
- Filtration

Partitioning

- Multi-level, overlap
- METIS

Preconditioning

- Incomplete LU
- Incomplete Cholesky
- Algebraic multi-level
- Block and parallel

Iterative Methods

- BiCGStab
- MINRES
- TFQMR
- SVD
- CG

Library of optimized basic linear algebra operations

- Matrix-vector, vector-vector, custom "group" functions and more

SparSol

- Solver for systems with real coefficients
 - 200,000 lines of source code, 550 classes
 - Serial & parallel versions
- SparSol innovations:
 - New ILU type preconditioner: FILU – Fill-in Incomplete LU preconditioner
 - New Incomplete Cholesky preconditioner: IC2
 - New parallel partitioner: KWPT

LinCoS

- Solver for systems with complex coefficients
 - 40,000 lines of source code, 40 classes
 - Serial & parallel versions
- LinCoS innovations:
 - Uses ILUT type preconditioner of Yousef Saad with some modifications (mostly for parallel computing)
 - Special library for performing complex number operations

SparSol – Overview

- Solver for large sparse linear systems with real coefficients
 - Originally developed for ExxonMobil URC for oil reservoir simulation
 - Developed in close cooperation with the Russian Academy of Sciences
- World-class mathematics for world-class problems
 - Solution of very ill-conditioned systems
 - Solution of very large systems – millions of unknowns
 - Symmetrical and unsymmetrical systems including block structures
 - Incorporates widely used, public-domain algorithms
 - Includes a large set of custom algorithms for specific problems
 - Adaptive convergence schemes guarantee convergence
 - Highly-efficient and scalable parallel solution algorithms

- Object-oriented framework for a robust collection of tools
 - Framework allows easy inclusion of new and custom algorithms
 - Supported on both Windows and Unix/Linux operating systems
 - Well-documented APIs allow easy integration with external applications
 - Fully-parametric environment can be tuned for numerous applications
- High-performance computational core
 - Custom library of basic linear algebra operations and grouped functions
 - Robust set of highly-tuned preprocessing & preconditioning algorithms
 - High-performance implementations of multiple iterative methods
 - Highly-efficient and scalable parallel solution algorithms
 - Thread-safe serial and parallel versions
 - Optimized for multi-step applications

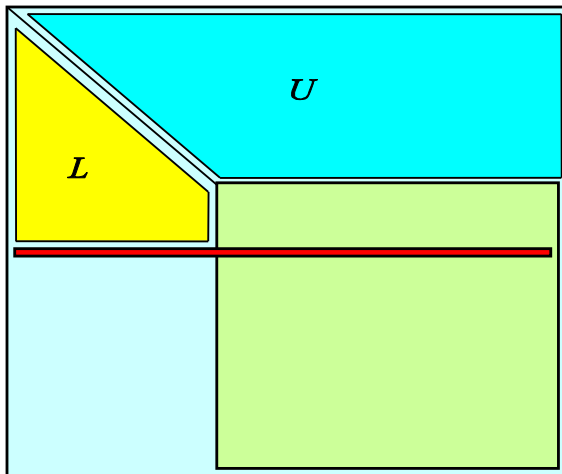
SparSol – Basic Algorithms

- Effective preprocessing algorithms
 - Scaling, reordering, filtration
- Several variants of preconditioning algorithms
 - Incomplete LU, incomplete Cholesky
 - Algebraic multi-level
 - Block and parallel
- Standard iterative methods
 - Conjugate gradient-type, minimal residual-type
 - Different convergence criteria and special convergence control scheme
- Specialized partitioners for parallel processing
 - KWPT
 - Multi-level partitioning
 - Partitioning with overlap

SparSol – Preconditioners

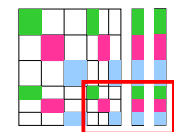
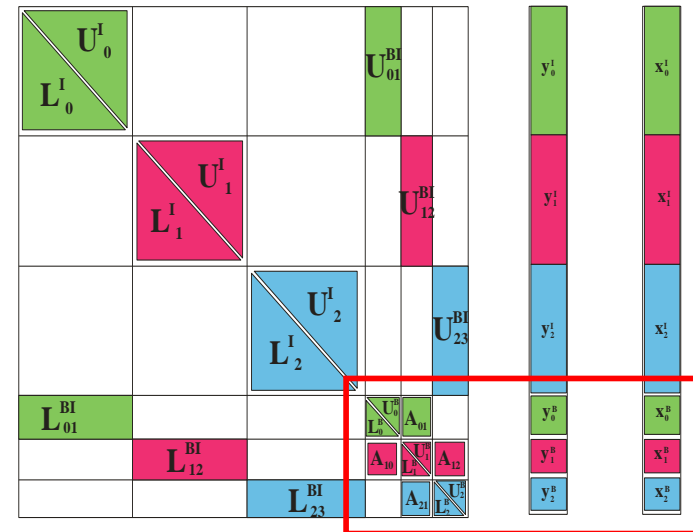
Standard Preconditioners

- Incomplete LU: RILU0, FILU, ILU2, ILUDP; multi-level ILU
- Incomplete Cholesky: RIC0, FIC, RIC2
- Nested factorization: NF, GNF, UNF



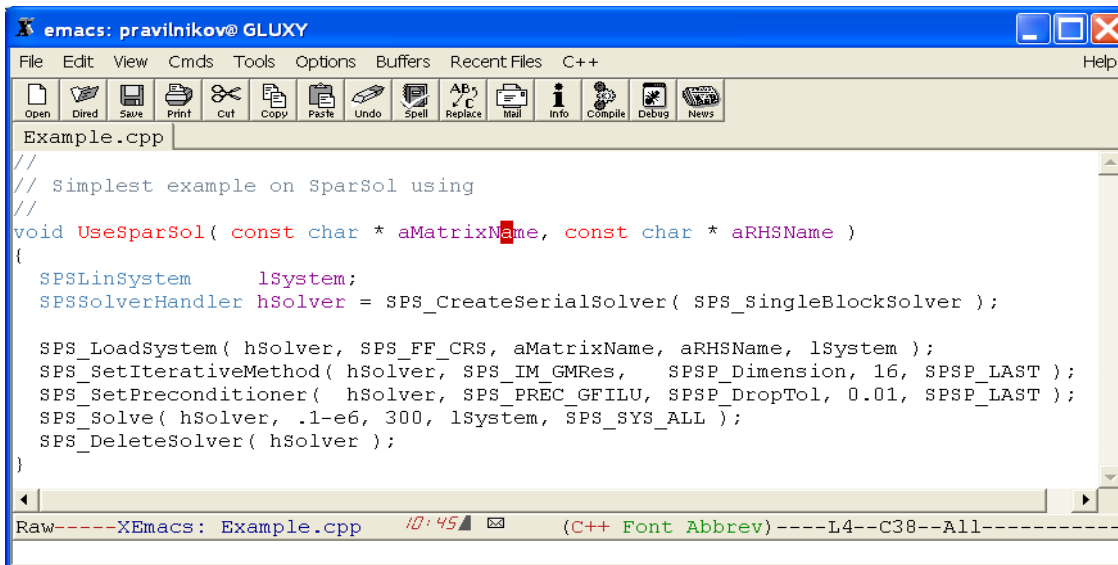
Parallel Preconditioners

- Parallel recursive FILU without overlap: RParFILU
- Parallel FILU with overlap: ParOFILU



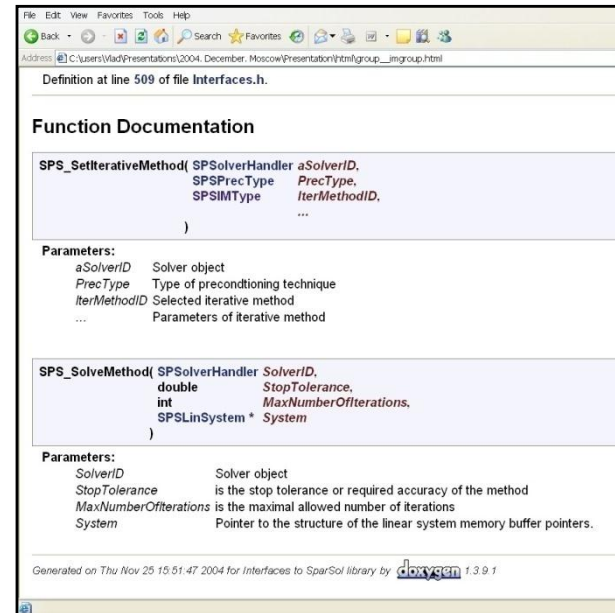
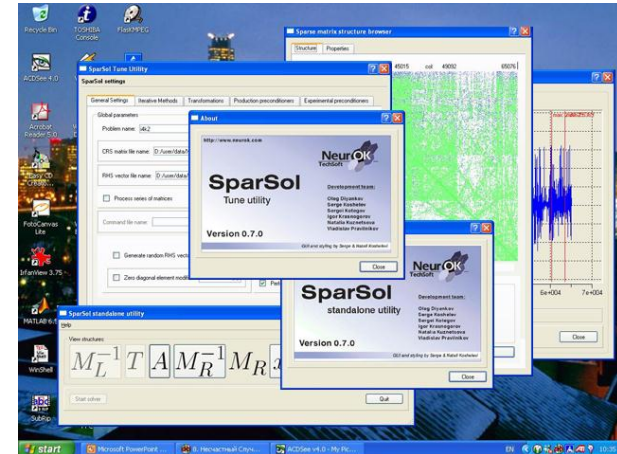
SparSol – Usage Model

- Library of algorithms and functions
- Accessed through application program interface
- May be used as a stand-alone utility from command line
- GUI interface for finding and tuning optimal parameters and for data visualization
- HTML documentation includes description of program interfaces, command-line options, variants of usage from users applications



```
Example.cpp
//
// Simplest example on SparSol using
//
void UseSparSol( const char * aMatrixName, const char * aRHSName )
{
    SPSLinSystem      lSystem;
    SPSSolverHandler hSolver = SPS_CreateSerialSolver( SPS_SingleBlockSolver );

    SPS_LoadSystem( hSolver, SPS_FF_CRs, aMatrixName, aRHSName, lSystem );
    SPS_SetIterativeMethod( hSolver, SPS_IM_GMRes, SPS_Dimension, 16, SPSP_LAST );
    SPS_SetPreconditioner( hSolver, SPS_PREC_GFILU, SPSP_DropTol, 0.01, SPSP_LAST );
    SPS_Solve( hSolver, .1-e6, 300, lSystem, SPS_SYS_ALL );
    SPS_DeleteSolver( hSolver );
}
Raw-----XEmacs: Example.cpp 10:45 (C++ Font Abbrev)-----L4--C38--A11-----
```



The screenshot shows a web browser displaying HTML documentation for SparSol. The page title is 'Definition at line 509 of file Interfaces.h.' The main content is titled 'Function Documentation' and lists two functions: `SPS_SetIterativeMethod` and `SPS_SolveMethod`. Each function is followed by its parameters and their descriptions.

Function Documentation

SPS_SetIterativeMethod(SPSolverHandler aSolverID, SPSPrecType PrecType, SPSIMType IterMethodID, ...)

Parameters:

- aSolverID Solver object
- PrecType Type of preconditioning technique
- IterMethodID Selected iterative method
- ... Parameters of iterative method

SPS_SolveMethod(SPSolverHandler SolverID, double StopTolerance, int MaxNumberOfIterations, SPSLinSystem * System)

Parameters:

- SolverID Solver object
- StopTolerance is the stop tolerance or required accuracy of the method
- MaxNumberOfIterations is the maximal allowed number of iterations
- System Pointer to the structure of the linear system memory buffer pointers.

Generated on Thu Nov 25 15:51:47 2004 for Interfaces to SparSol library by doxygen 1.3.9.1

SparSol Supported Formats

- Text Formats
 - CRS (Compress Row Storage)
 - CCS (Compress Column Storage)
 - Coordinate (Matlab-like)
 - Matrix-Block CRS
 - Matrix-Market
 - Harwell-Boeing
 - Rutherford-Boeing
 - Block Modified Sparse Row format
 - Block CRS format
- Other Formats
 - SparSol binary (CRS format)
 - OUTPUT4 format
 - XML format (with data compression)
 - Custom formats

SparSol – Performance Comparisons

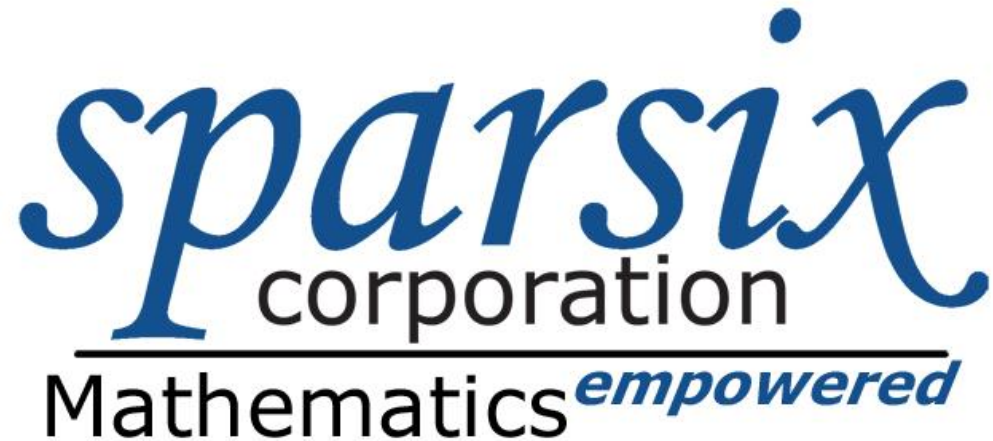


Performance comparison using ExxonMobil collection of matrices

N – number of rows Z – the number of non-zero elements

Name	N	Z	Z/N	PARDISO Time	SPARSEKIT2 Time	SparSol Time	Vs. PARDISO	Vs. SPARSEKIT2
CIT-1	17,436	344,245	19.7	2.62	*NA	0.27	9.7x	∞
CIT-2	249,428	5,613,978	22.5	*NA	*NA	5.31	∞	∞
SBO-1	21,700	145,122	6.7	0.86	0.78	0.62	1.4x	1.3x
SBO-2	111,756	888,190	7.9	9.88	6.28	3.47	2.8x	1.8x
SBO-3	216,051	1,849,317	8.6	39.65	15.31	8.24	4.8x	1.9x
SBO-4	93,264	667,882	7.2	12.10	3.41	2.23	5.4x	1.5x
SEO-1	22,421	204,784	9.1	2.26	0.90	0.46	4.9x	2.0x
Matrix 7	378,931	2,699,192	7.1	180.22	14.69	12.55	14.4x	1.2x
Matrix 8	411,087	5,454,618	13.3	*NA	*NA	4.75	∞	∞
Matrix 9	416,110	8,752,310	21.0	*NA	*NA	10.49	∞	∞
Matrix 13	753,876	10,752,287	14.3	*NA	*NA	13.22	∞	∞
Matrix 14	929,580	25,838,064	27.8	*NA	*NA	15.20	∞	∞
Matrix 16	1,735,863	61,750,422	35.6	*NA	*NA	47.75	∞	∞

*NA – Not available – solver could not solve system



Sparsix Corporation

236 West Portal Avenue, #221
San Francisco, CA 94127
(415) 367-3523
info@sparsix.com