Multiple Precision Arithmetic Versions of SDP solvers; SDPA-GMP, SDPA-QD and SDPA-DD

NAKATA, Maho

Advanced Center of Computing and Communication,
RIKEN (The Institute of Physical and Chemical Research),
Saitama, Japan

2009/8/23-28, ISMP 2009

Collaborators

The SDPA project members in alphabetic order with WAKI, Hayato

- FUJISAWA, Katsuki
- FUKUDA, Mituhiro
- FUTAKATA, Yoshiaki
- KOBAYASHI, Kazuhiro
- KOJIMA, Masakazu
- NAKATA, Kazuhide
- (NAKATA, Maho)
- YAMASHITA, Makoto

Outline

1. Introduction
   - Abstract
   - What is number?
   - Semidefinite programming
   - Necessity of accurate solver
   - Origins of accuracy loss
2. Development of SDPA-GMP, SDPA-QD, DD, and MPACK
3. Results

Outline

Abstract

- I just want to solve SDP very very accurately!
- Problems from chemistry can be solved via SDP solvers: [Nakata-Nakatsuji-Ehara-Fukuda-Nakata-Fujisawa, J. Chem. Phys. 114, 8282 (2001)]
- Such problems require very high accuracy to SDP; relative gap $< 1.0 \times 10^{-8}$
- There are some inaccurate results.
- Multiple precision arithmetic version of SDPA; SDPA-GMP, SDPA-QD, SDPA-DD: http://sdpa.indsys.chuo-u.ac.jp/sdpa/.
- Multiple precision arithmetic version of BLAS/LAPACK: mpack http://mplapack.sourceforge.net/
- YES we can solve very very accurately!

Abstract

- I just want to solve SDP very very accurately!
- Problems from chemistry can be solved via SDP solvers:
  [Nakata-Nakatsuji-Ehara-Fukuda-Nakata-Fujisawa, J. Chem. Phys. 114, 8282 (2001)]
- Such problems require very high accuracy to SDP; relative gap $< 1.0 \times 10^{-8}$
- There are some inaccurate results.
- Multiple precision arithmetic version of SDPA; SDPA-GMP, SDPA-QD, SDPA-DD: http://sdpa.indsys.chuo-u.ac.jp/sdpa/.
- Multiple precision arithmetic version of BLAS/LAPACK: mpack http://mplapack.sourceforge.net/
- YES we can solve very very accurately!

Abstract

- I just want to solve SDP very very accurately!
- Problems from chemistry can be solved via SDP solvers:
  [Nakata-Nakatsuji-Ehara-Fukuda-Nakata-Fujisawa, J. Chem. Phys. 114, 8282 (2001)]
- Such problems require very high accuracy to SDP; relative gap $< 1.0 \times 10^{-8}$
- There are some inaccurate results.
- Multiple precision arithmetic version of SDPA; SDPA-GMP, SDPA-QD, SDPA-DD: http://sdpa.indsys.chuo-u.ac.jp/sdpa/.
- Multiple precision arithmetic version of BLAS/LAPACK: mpack http://mplapack.sourceforge.net/
- YES we can solve very very accurately!

Abstract

- I just want to solve SDP very very accurately!
- Problems from chemistry can be solved via SDP solvers:
  [Nakata-Nakatsuji-Ehara-Fukuda-Nakata-Fujisawa, J. Chem. Phys. 114, 8282 (2001)]
- Such problems require very high accuracy to SDP; relative gap $< 1.0 \times 10^{-8}$
- There are some inaccurate results.
- Multiple precision arithmetic version of SDPA; SDPA-GMP, SDPA-QD, SDPA-DD: http://sdpa.indsys.chuo-u.ac.jp/sdpa/.
- Multiple precision arithmetic version of BLAS/LAPACK: mpack http://mplapack.sourceforge.net/
- YES we can solve very very accurately!

Abstract

- I just want to solve SDP very very accurately!
- Problems from chemistry can be solved via SDP solvers:
  [Nakata-Nakatsuji-Ehara-Fukuda-Nakata-Fujisawa, J. Chem. Phys. 114, 8282 (2001)]
- Such problems require very high accuracy to SDP; relative gap $< 1.0 \times 10^{-8}$
- There are some inaccurate results.
- Multiple precision arithmetic version of SDPA; SDPA-GMP, SDPA-QD, SDPA-DD: http://sdpa.indsys.chuo-u.ac.jp/sdpa/.
- Multiple precision arithmetic version of BLAS/LAPACK: mpack http://mplapack.sourceforge.net/
- YES we can solve very very accurately!

Abstract

- I just want to solve SDP very very accurately!
- Problems from chemistry can be solved via SDP solvers:
  [Nakata-Nakatsuji-Ehara-Fukuda-Nakata-Fujisawa, J. Chem. Phys. 114, 8282 (2001)]
- Such problems require very high accuracy to SDP; relative gap $< 1.0 \times 10^{-8}$
- There are some inaccurate results.
- Multiple precision arithmetic version of SDPA; SDPA-GMP, SDPA-QD, SDPA-DD: http://sdpa.indsys.chuo-u.ac.jp/sdpa/.
- Multiple precision arithmetic version of BLAS/LAPACK: mpack http://mplapack.sourceforge.net/
- YES we can solve very very accurately!

Abstract

- I just want to solve SDP very very accurately!
- Problems from chemistry can be solved via SDP solvers:
  [Nakata-Nakatsuji-Ehara-Fukuda-Nakata-Fujisawa, J. Chem. Phys. 114, 8282 (2001)]
- Such problems require very high accuracy to SDP; relative gap $< 1.0 \times 10^{-8}$
- There are some inaccurate results.
- Multiple precision arithmetic version of SDPA; SDPA-GMP, SDPA-QD, SDPA-DD: http://sdpa.indsys.chuo-u.ac.jp/sdpa/.
- Multiple precision arithmetic version of BLAS/LAPACK: mpack http://mplapack.sourceforge.net/
- YES we can solve very very accurately!

Maho's philosophy

Do not think seriously. Take it easy!

Keep it sweet and simple

JUST ADD PRECISION

Maho's philosophy

Do not think seriously. Take it easy!

Keep it sweet and simple

JUST ADD PRECISION

Maho's philosophy

Do not think seriously. Take it easy!

Keep it sweet and simple

JUST ADD PRECISION

Outline

What is number?

There are several kinds of numbers.

- Natural number: $0, 1, 2, 3, 4, \cdots$
- Integer: $\cdots, -3, -2, -1, 0, 1, 2, 3, 4, \cdots$
- Rational number: a/b, where a, b are relatively prime
- Real number: convergence of Cauchy series. $\{x_n : x_n \in \mathbb{Q}\}_{n=0,1\cdots}$ s.t. $\forall \epsilon > 0, \exists N, \forall n, m > N \rightarrow |x_n - x_m| < \epsilon$ defines a real number x.
- Complex number: $z = a + bi$: two real numbers with i.
- floating point number: designed for computers, subset of rational numbers.

What is number?

There are several kinds of numbers.

- Natural number: $0, 1, 2, 3, 4, \cdots$
- Integer: $\cdots, -3, -2, -1, 0, 1, 2, 3, 4, \cdots$
- Rational number: a/b, where a, b are relatively prime
- Real number: convergence of Cauchy series.
  $\{x_n : x_n \in \mathbb{Q}\}_{n=0,1\cdots}$ s.t. $\forall\epsilon > 0, \exists N, \forall n, m > N \rightarrow |x_n - x_m| < \epsilon$ defines a real number x.
- Complex number: $z = a + bi$: two real numbers with i.
- floating point number: designed for computers, subset of rational numbers.

What is number?

There are several kinds of numbers.

- Natural number: $0, 1, 2, 3, 4, \cdots$
- Integer: $\cdots, -3, -2, -1, 0, 1, 2, 3, 4, \cdots$
- Rational number: a/b, where a, b are relatively prime
- Real number: convergence of Cauchy series. $\{x_n : x_n \in \mathbb{Q}\}_{n=0,1\cdots}$ s.t. $\forall \epsilon > 0, \exists N, \forall n, m > N \rightarrow |x_n - x_m| < \epsilon$ defines a real number x.
- Complex number: $z = a + bi$: two real numbers with i.
- floating point number: designed for computers, subset of rational numbers.

What is number?

There are several kinds of numbers.

- Natural number: $0, 1, 2, 3, 4, \cdots$
- Integer: $\cdots, -3, -2, -1, 0, 1, 2, 3, 4, \cdots$
- Rational number: a/b, where a, b are relatively prime
- Real number: convergence of Cauchy series.
  $\{x_n : x_n \in \mathbb{Q}\}_{n=0,1\dots}$ s.t. $\forall\epsilon > 0, \exists N, \forall n, m > N \to |x_n - x_m| < \epsilon$
  defines a real number x.
- Complex number: $z = a + bi$: two real numbers with i.
- floating point number: designed for computers, subset of rational numbers.

What is number?

There are several kinds of numbers.

- Natural number: $0, 1, 2, 3, 4, \cdots$
- Integer: $\cdots, -3, -2, -1, 0, 1, 2, 3, 4, \cdots$
- Rational number: a/b, where a, b are relatively prime
- Real number: convergence of Cauchy series.
  $\{x_n : x_n \in \mathbb{Q}\}_{n=0,1\cdots}$ s.t. $\forall \epsilon > 0, \exists N, \forall n, m > N \rightarrow |x_n - x_m| < \epsilon$ defines a real number x.
- Complex number: $z = a + bi$: two real numbers with i.
- floating point number: designed for computers, subset of rational numbers.

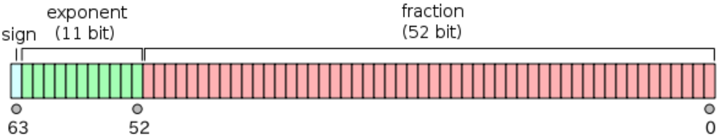
What is number?

There are several kinds of numbers.

- Natural number: $0, 1, 2, 3, 4, \cdots$
- Integer: $\cdots, -3, -2, -1, 0, 1, 2, 3, 4, \cdots$
- Rational number: a/b, where a, b are relatively prime
- Real number: convergence of Cauchy series.
  $\{x_n : x_n \in \mathbb{Q}\}_{n=0,1\cdots}$ s.t. $\forall \epsilon > 0, \exists N, \forall n, m > N \to |x_n - x_m| < \epsilon$
  defines a real number x.
- Complex number: $z = a + bi$: two real numbers with i.
- floating point number: designed for computers, subset of rational numbers.

IEEE 754: Standard for Binary Floating-Point Arithmetic

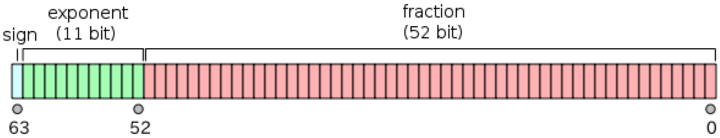
- The IEEE Standard for Floating-Point Arithmetic (IEEE 754) is the most widely-used standard for floating-point computation.
- Very well designed we feel as if we treat real numbers.
- IEEE 754 double is expressed in 64-bit (= 8 bytes)



- $a = \pm\left(\frac{1}{2} + \frac{d_2}{2^2} + \frac{d_3}{2^3} + \cdots + \frac{d_{52}}{2^{52}}\right) \times 2^e$, $d = 0$ or 1, $e = -1022 \sim 1023$
- about 16 significant digits ($\log_{10} 2^{53} = 15.955$).
- Implemented for popular CPUs.

IEEE 754: Standard for Binary Floating-Point Arithmetic

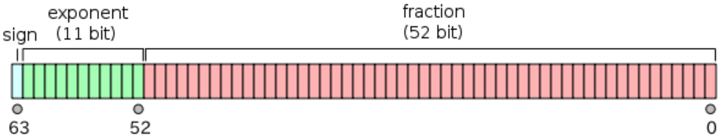
- The IEEE Standard for Floating-Point Arithmetic (IEEE 754) is the most widely-used standard for floating-point computation.
- Very well designed we feel as if we treat real numbers.
- IEEE 754 double is expressed in 64-bit (= 8 bytes)



- $a = \pm\left(\frac{1}{2} + \frac{d_2}{2^2} + \frac{d_3}{2^3} + \cdots + \frac{d_{52}}{2^{52}}\right) \times 2^e$, $d = 0$ or 1, $e = -1022 \sim 1023$
- about 16 significant digits ($\log_{10} 2^{53} = 15.955$).
- Implemented for popular CPUs.

IEEE 754: Standard for Binary Floating-Point Arithmetic

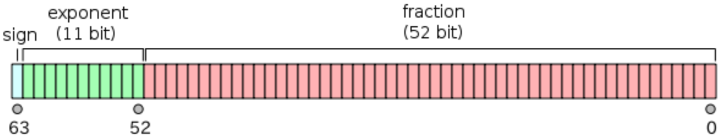
- The IEEE Standard for Floating-Point Arithmetic (IEEE 754) is the most widely-used standard for floating-point computation.
- Very well designed we feel as if we treat real numbers.
- IEEE 754 double is expressed in 64-bit (= 8 bytes)



- $a = \pm\left(\frac{1}{2} + \frac{d_2}{2^2} + \frac{d_3}{2^3} + \cdots + \frac{d_{52}}{2^{52}}\right) \times 2^e$, $d = 0$ or 1, $e = -1022 \sim 1023$
- about 16 significant digits ($\log_{10} 2^{53} = 15.955$).
- Implemented for popular CPUs.

IEEE 754: Standard for Binary Floating-Point Arithmetic

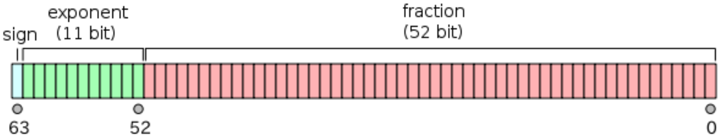
- The IEEE Standard for Floating-Point Arithmetic (IEEE 754) is the most widely-used standard for floating-point computation.
- Very well designed we feel as if we treat real numbers.
- IEEE 754 double is expressed in 64-bit (= 8 bytes)



- $a = \pm\left(\frac{1}{2} + \frac{d_2}{2^2} + \frac{d_3}{2^3} + \cdots + \frac{d_{52}}{2^{52}}\right) \times 2^e$, $d = 0$ or 1, $e = -1022 \sim 1023$
- about 16 significant digits ($\log_{10} 2^{53} = 15.955$).
- Implemented for popular CPUs.

IEEE 754: Standard for Binary Floating-Point Arithmetic

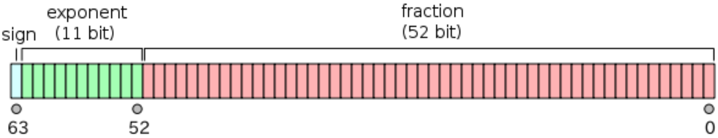
- The IEEE Standard for Floating-Point Arithmetic (IEEE 754) is the most widely-used standard for floating-point computation.
- Very well designed we feel as if we treat real numbers.
- IEEE 754 double is expressed in 64-bit (= 8 bytes)



- $a = \pm\left(\frac{1}{2} + \frac{d_2}{2^2} + \frac{d_3}{2^3} + \cdots + \frac{d_{52}}{2^{52}}\right) \times 2^e$, $d = 0$ or 1, $e = -1022 \sim 1023$
- about 16 significant digits ($\log_{10} 2^{53} = 15.955$).
- Implemented for popular CPUs.

IEEE 754: Standard for Binary Floating-Point Arithmetic

- The IEEE Standard for Floating-Point Arithmetic (IEEE 754) is the most widely-used standard for floating-point computation.
- Very well designed we feel as if we treat real numbers.
- IEEE 754 double is expressed in 64-bit (= 8 bytes)



- $a = \pm\left(\frac{1}{2} + \frac{d_2}{2^2} + \frac{d_3}{2^3} + \cdots + \frac{d_{52}}{2^{52}}\right) \times 2^e$, $d = 0$ or 1, $e = -1022 \sim 1023$
- about 16 significant digits ($\log_{10} 2^{53} = 15.955$).
- Implemented for popular CPUs.

IEEE 754: Standard for Binary Floating-Point Arithmetic

- Arithmetic operations with rounding errors.

$$A \oplus B \neq A + B$$

  Almost every manipulation include rounding error.

- In this study, still we suffer from the rounding error. We just add precision.

IEEE 754: Standard for Binary Floating-Point Arithmetic

- Arithmetic operations with rounding errors.

$$A \oplus B \neq A + B$$

  Almost every manipulation include rounding error.

- In this study, still we suffer from the rounding error. We just add precision.

Outline

Semidefinite programming

$$
\begin{cases}
\text{primal} & \text{minimize:} & A_0 \bullet X & \\
 & \text{s.t.:} & A_i \bullet X = b_i & (i = 1, 2, \cdots, m) \\
 & & X \succeq 0 & \\
\text{dual} & \text{maximize:} & \sum_{i=1}^{m} b_i z_i & \\
 & \text{s.t.:} & \sum_{i=1}^{m} A_i z_i + Y = A_0 & \\
 & & Y \succeq 0 &
\end{cases}
$$

A_i is $n \times n$ real symmetric matrices, X $n \times n$ real symmetric variable matrix, b_i are constant vectors of m-dimension, Y is $n \times n$ a real symmetric variable matrix, $X \bullet Y := \sum X_{ij} Y_{ij}$. $X \succeq 0$ means X is positive semidefinite.

Outline

Do we need to solve SDP problems *accurately?*

There are some questions for SDP results.

- Some SDPs are hard to solve. The results may have large gaps, not feasible.
- Simply we may not trust the results: “Strange Behaviors of Interior-point Methods for Solving Semidefinite Programming Problems in Polynomial Optimization” [Waki-Nakata-Muramatsu submitted]
- Users seldom care about the input file: try to solve ill-posed SDPs.

Do we need to solve SDP problems *accurately?*

There are some questions for SDP results.

- Some SDPs are hard to solve. The results may have large gaps, not feasible.
- Simply we may not trust the results: "Strange Behaviors of Interior-point Methods for Solving Semidefinite Programming Problems in Polynomial Optimization" [Waki-Nakata-Muramatsu submitted]
- Users seldom care about the input file: try to solve ill-posed SDPs.

Do we need to solve SDP problems *accurately?*

There are some questions for SDP results.

- Some SDPs are hard to solve. The results may have large gaps, not feasible.
- Simply we may not trust the results: “Strange Behaviors of Interior-point Methods for Solving Semidefinite Programming Problems in Polynomial Optimization” [Waki-Nakata-Muramatsu submitted]
- Users seldom care about the input file: try to solve ill-posed SDPs.

Outline

Sources of the evil (I)

- IEEE 754 double arithmetic: done in 16 significant digits. accuracy losses in manipulations

$$1 \oplus 1.0 \times 10^{-17} = 1$$

- Condition number of matrix A; $||A||||A^{-1}||$. when it becomes 10^{16}, solution to the linear equation is inaccurate with IEEE 754 double.
- $X \bullet Y = 0$ at the optimum (complementarity slackness theorem for SDP)
  variable matrix becomes singular at the optimum; condition number becomes infinite!

Sources of the evil (I)

- IEEE 754 double arithmetic: done in 16 significant digits. accuracy losses in manipulations

$$1 \oplus 1.0 \times 10^{-17} = 1$$

- Condition number of matrix A; $||A||||A^{-1}||$. when it becomes 10^{16}, solution to the linear equation is inaccurate with IEEE 754 double.
- $X \bullet Y = 0$ at the optimum (complementarity slackness theorem for SDP)
  variable matrix becomes singular at the optimum; condition number becomes infinite!

Sources of the evil (I)

- IEEE 754 double arithmetic: done in 16 significant digits. accuracy losses in manipulations

$$1 \oplus 1.0 \times 10^{-17} = 1$$

- Condition number of matrix A; $||A||||A^{-1}||$. when it becomes 10^{16}, solution to the linear equation is inaccurate with IEEE 754 double.
- $X \bullet Y = 0$ at the optimum (complementarity slackness theorem for SDP)
  variable matrix becomes singular at the optimum; condition number becomes infinite!

Sources of the evil (II)

- Z^{-1}: Primal-dual interior point method calculates Z^{-1}; *"It is seldom necessary to compute the inverts of matrix explicitly, and it is certainly not recommended as a means of solving linear systems."* by LAPACK Users' Guide Third Edition, p.14.
- Human factor: users try to solve SDPs which do not satisfy Slater's condition, i.e., *no interior points* etc, NO GUARANTEE! DO NOT BLAME SOLVERS!

Sources of the evil (II)

- Z^{-1}: Primal-dual interior point method calculates Z^{-1}; *"It is seldom necessary to compute the inverts of matrix explicitly, and it is certainly not recommended as a means of solving linear systems."* by LAPACK Users' Guide Third Edition, p.14.
- Human factor: users try to solve SDPs which do not satisfy Slater's condition, i.e., *no interior points* etc, NO GUARANTEE! DO NOT BLAME SOLVERS!

A brute force method for accurate SDP solutions

- Use multiple precision arithmetic; GMP, QD rather than IEEE 754 double.
- Simple answer to obtain high accuracy.
- Do not solve all the problems, but many!

A brute force method for accurate SDP solutions

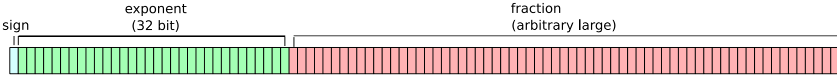
- Use multiple precision arithmetic; GMP, QD rather than IEEE 754 double.
- Simple answer to obtain high accuracy.
- Do not solve all the problems, but many!

A brute force method for accurate SDP solutions

- Use multiple precision arithmetic; GMP, QD rather than IEEE 754 double.
- Simple answer to obtain high accuracy.
- Do not solve all the problems, but many!

What is GMP?

- GMP is a free library for arbitrary precision arithmetic, operating on signed integers, rational numbers, and floating point numbers.
- significant digits: arbitrary (I usually use $60 \sim 72$ digits)



Strategy and features

- Using existing multiple precision libraries.
- Based on SDPA; http://sdpa.indsys.chuo-u.ac.jp/sdpa/
- No changes in algorithm.
- Changes from SDPA should be minimal to reduce the maintenance cost.
- Matrix-vector manipulations and eigenvalues etc. → Multiple precision version of LAPACK and BLAS.
  - 49 routines from MPACK; Rpotrf (dpotrf.f; cholesky), Rsyev (dsyev.f eigenvalue), Rsterf, Rsteqr (dsterf.f, dsteqr.f) etc..
- Introduction of “precision” parameter; controls number of significant bits used in the calculations.
- Actually I did was replacing “double” to “mpf_class” carefully.

Another MP library: Quad-Double library

- Usually quadruple or octuple precision are enough.
- Double-Double and Quad-Double Arithmetic; by Y. Hida, Xiaoye S. Li, David H Bailey, and faster than GMP.
- Four/two unevaluated IEEE 754 double $\sim$ approx octuple/quadruple precision.

$$A = (a_0, a_1, a_2, a_3)$$

- Utilize exact transformations [Dekker, Knuth, Priest, Shewcheck].

$$a = x \oplus y, b = x + y - (x \oplus y)$$

  Error by IEEE754 add $x \oplus y$ can be *exactly* evaluated.
- Replace “mpf_class” to “dd_real” and “qd_real” $\rightarrow$ SDPA-QD, SDPA-DD.

What is MPACK?

- MPACK is a multiple precision version of BLAS and LAPACK. http://mplapack.sourceforge.net/
- What is the BLAS? The BLAS (Basic Linear Algebra Subprograms) are routines that provide standard building blocks for performing basic vector and matrix operations.
- What is LAPACK? This provides routines for solving systems of simultaneous linear equations, least-squares solutions of linear systems of equations, eigenvalue problems, and singular value problems.
- Written C++, but look very similar to reference BLAS implementation.
- Very portable and no optimization at this moment.
- Pretty good compatibility with BLAS and LAPACK.

What is MPACK?

- MPACK is a multiple precision version of BLAS and LAPACK. http://mplapack.sourceforge.net/
- What is the BLAS? The BLAS (Basic Linear Algebra Subprograms) are routines that provide standard building blocks for performing basic vector and matrix operations.
- What is LAPACK? This provides routines for solving systems of simultaneous linear equations, least-squares solutions of linear systems of equations, eigenvalue problems, and singular value problems.
- Written C++, but look very similar to reference BLAS implementation.
- Very portable and no optimization at this moment.
- Pretty good compatibility with BLAS and LAPACK.

What is MPACK?

- MPACK is a multiple precision version of BLAS and LAPACK. http://mplapack.sourceforge.net/
- What is the BLAS? The BLAS (Basic Linear Algebra Subprograms) are routines that provide standard building blocks for performing basic vector and matrix operations.
- What is LAPACK? This provides routines for solving systems of simultaneous linear equations, least-squares solutions of linear systems of equations, eigenvalue problems, and singular value problems.
- Written C++, but look very similar to reference BLAS implementation.
- Very portable and no optimization at this moment.
- Pretty good compatibility with BLAS and LAPACK.

What is MPACK?

- MPACK is a multiple precision version of BLAS and LAPACK. http://mplapack.sourceforge.net/
- What is the BLAS? The BLAS (Basic Linear Algebra Subprograms) are routines that provide standard building blocks for performing basic vector and matrix operations.
- What is LAPACK? This provides routines for solving systems of simultaneous linear equations, least-squares solutions of linear systems of equations, eigenvalue problems, and singular value problems.
- Written C++, but look very similar to reference BLAS implementation.
- Very portable and no optimization at this moment.
- Pretty good compatibility with BLAS and LAPACK.

What is MPACK?

- MPACK is a multiple precision version of BLAS and LAPACK. http://mplapack.sourceforge.net/
- What is the BLAS? The BLAS (Basic Linear Algebra Subprograms) are routines that provide standard building blocks for performing basic vector and matrix operations.
- What is LAPACK? This provides routines for solving systems of simultaneous linear equations, least-squares solutions of linear systems of equations, eigenvalue problems, and singular value problems.
- Written C++, but look very similar to reference BLAS implementation.
- Very portable and no optimization at this moment.
- Pretty good compatibility with BLAS and LAPACK.

What is MPACK?

- MPACK is a multiple precision version of BLAS and LAPACK. http://mplapack.sourceforge.net/
- What is the BLAS? The BLAS (Basic Linear Algebra Subprograms) are routines that provide standard building blocks for performing basic vector and matrix operations.
- What is LAPACK? This provides routines for solving systems of simultaneous linear equations, least-squares solutions of linear systems of equations, eigenvalue problems, and singular value problems.
- Written C++, but look very similar to reference BLAS implementation.
- Very portable and no optimization at this moment.
- Pretty good compatibility with BLAS and LAPACK.

MBLAS Rgemm.cpp and BLAS dgemm.f

Rgemm.cpp

```
//Start the operations.
if (notb) {
    if (nota) {
        //Form C := alpha*A*B + beta*C.
        for (int j = 0; j < n; j++) {
            if (beta == Zero) {
                for (int i = 0; i < m; i++) {
                    C[i + j * ldc] = Zero;
                }
            } else if (beta != One) {
                for (int i = 0; i < m; i++) {
                    C[i + j * ldc] = beta * C[i + j * ldc];
                }
            }
            for (int l = 0; l < k; l++) {
                if (B[l + j * ldb] != Zero) {
                    temp = alpha * B[l + j * ldb];
                    for (int i = 0; i < m; i++) {
                        C[i + j * ldc] =
                            C[i + j * ldc] + temp * A[i + l * lda];
                    }
                }
            }
        }
    } else {
//Form  C := alpha*A'*B + beta*C.
```

dgemm.f

```
*
*     Start the operations.
*
      IF (NOTB) THEN
          IF (NOTA) THEN
*
*           Form  C := alpha*A*B + beta*C.
*
              DO 90 J = 1,N
                  IF (BETA.EQ.ZERO) THEN
                      DO 50 I = 1,M
                          C(I,J) = ZERO
   50                 CONTINUE
                  ELSE IF (BETA.NE.ONE) THEN
                      DO 60 I = 1,M
                          C(I,J) = BETA*C(I,J)
   60                 CONTINUE
                  END IF
                  DO 80 L = 1,K
                      IF (B(L,J).NE.ZERO) THEN
                          TEMP = ALPHA*B(L,J)
                          DO 70 I = 1,M
                              C(I,J) = C(I,J) + TEMP*A(I,L)
   70                     CONTINUE
                      END IF
   80             CONTINUE
   90         CONTINUE
          ELSE
*
*           Form  C := alpha*A'*B + beta*C
```

How MBLAS is used in SDPA-GMP?

From sdpa_linear.cpp from SDPA-GMP 7.1.2.

```
    if (scalar==NULL) {
      scalar = &MONE;
      // scalar is local variable
    }
    // The Point is the first argument is "Transpose".
    Rgemm("Transpose","NoTranspose",retMat.nRow,retMat.nCol,aMat.nCol,
           *scalar,aMat.de_ele,aMat.nCol,bMat.de_ele,bMat.nRow,
           0.0,retMat.de_ele,retMat.nRow);
    break;
  case DenseMatrix::COMPLETION:
    rError("no support for COMPLETION");
    break;
  }

  return _SUCCESS;
```

Results (I)

Some results from SDPLIB, on Opteron 250 (2.4GHz), 16G Mem, FreeBSD 7/amd64.
"precision" is 250 for GMP.

instance	arch8(GMP)	arch8(QD)	arch8(DD)	arch8(double)
iter	47	47	37	25
rel. gap	$3.57e-31$	$3.58e-31$	$3.80e-21$	$1.65e-08$
p feas. error	$3.11e-76$	$1.02e-61$	$5.05e-29$	$1.14e-12$
d feas. error	$5.66e-72$	$9.01e-52$	$4.85e-21$	$1.10e-07$
time (s)	634.766	497.289	55.445	9.35

instance	mcp500-4(GMP)	mcp500-4(QD)	mcp500-4(DD)	mcp500-4(double)
iter	38	38	28	15
rel. gap	$1.36e-31$	$1.36e-31$	$1.36e-21$	$1.16e-08$
p feas. error	$1.28e-76$	$6.08e-64$	$6.41e-31$	$4.88e-15$
d feas. error	$1.67e-75$	$7.72e-59$	$1.68e-28$	$1.02e-13$
time (s)	5711.6	4678.1	455.0	10.2

instance	maxG32(GMP)	maxG32(QD)	maxG32(DD)	maxG32(double)
iter	40	40	30	17
rel. gap	$2.07e-31$	$2.07e-31$	$2.04e-21$	$1.65e-08$
p feas. error	$1.74e-76$	$1.09e-64$	$1.23e-31$	$1.14e-12$
d feas. error	$1.90e-72$	$2.47e-53$	$8.53e-25$	$1.10e-07$
time (s)	348564.8	315969.5	30472.0	9.35

Results (II) 1D-Hubbard model

1D Hubbard model Strong correlation limit: $|U/t| \to \infty$:[Nakata et al. JCP 2008]; with SDPA-GMP 6.

Ground state energy of 1D Hubbard model

PBC, # of sites:4, # of electrons: 4, spin 0

U/t	SDPA (16 digits)	SDPA-GMP (60 digits)	fullCI
10000.0	0	$-1.1999998800000251 \times 10^{-3}$	$-1.199999880 \times 10^{-3}$
1000.0	-1.2×10^{-2}	$-1.1999880002507934 \times 10^{-2}$	$-1.1999880002 \times 10^{-2}$
100.0	-1.1991×10^{-1}	$-1.1988025013717993 \times 10^{-1}$	$-1.19880248946 \times 10^{-1}$
10.0	-1.1000	-1.0999400441222934	-1.099877772750
1.0	-3.3417	-3.3416748070259956	-3.340847617248

PBC, # of sites:6, # of electrons: 6, spin 0

U/t	SDPA (16 digits)	SDPA-GMP (60 digits)	fullCI
10000.0	0	$-1.7249951195749525 \times 10^{-3}$	$-1.721110121 \times 10^{-3}$
1000.0	-1×10^{-2}	$-1.7255360310431304 \times 10^{-2}$	$-1.7211034713 \times 10^{-2}$
100.0	-1.730×10^{-1}	$-1.7302157140594339 \times 10^{-1}$	$-1.72043338097 \times 10^{-1}$
10.0	-1.6954	-1.6953843276854447	-1.664362733287
1.0	-6.6012	-6.6012042217806286	-6.601158293375

Results (III) Kissing number

Kissing number from A New Library of Structured Semidefinite Programming Instances; the optimal values *were* uncertain or only known with low accuracy. Powered by Fujisawa-san (2008/12/21); precision is 128bit for GMP.

instance	opt (double)	opt (GMP)
kissing_3_10_10	-11.4385	-11.43814328
kissing_4_10_10	-23.14	-23.13553364
kissing_5_10_10	-44.15	-44.158868754
kissing_6_10_10	-77.9	-77.912852357
kissing_7_10_10	-134.3	-134.32853967
kissing_8_10_10	-238.929	-238.99981527
kissing_9_10_10	-365	-365.21946909
kissing_10_10_10	-562.9	-562.89594739
kissing_11_10_10	-889.74	-889.74203646
kissing_12_10_10	-1369.485	-1369.5287720

History

- YAMASHITA-san told me NAKATA Kazuhide-san's student implemented MP version of SDP solver in Java using fixed point numbers.
- I started to implement SDPA-GMP6 based on SDPA6. The first working version: 2006/12/5. Lot of discussions with NAKATA Kazuhide-san.
- First appearance of SDPA-GMP 6 is Journal of chemical physics 128, 16 164113 (2008); to solve strong correlation limit of Hubbard model.
- I have been implementing general purpose multiple precision version of BLAS/LAPACK routines; MPACK (MBLAS/MLAPACK).
- SDPA-GMP 6, 7.0.2, 3, 5 etc. internal versions.
- SDPA-GMP 7.1.0 has been released in 2008/4/10.
- MPACK (MBLAS/MLAPACK) 0.0.8 has been released in 2009/1/8.
- SDPA-GMP 7.1.2 supports MPACK 0.0.9 in 2009/2/5.
- QD and DD version are requested by Hans D. Mittelmann and Fujisawa-san.
- SDPA-GMP, QD and DD 7.1.2 have been released in 2009/3/21.
- SDPA-GMP, QD and DD is on NEOS server
- Extensive tests with Waki-san and Fujisawa-san.

History

- YAMASHITA-san told me NAKATA Kazuhide-san's student implemented MP version of SDP solver in Java using fixed point numbers.
- I started to implement SDPA-GMP6 based on SDPA6. The first working version: 2006/12/5. Lot of discussions with NAKATA Kazuhide-san.
- First appearance of SDPA-GMP 6 is Journal of chemical physics 128, 16 164113 (2008); to solve strong correlation limit of Hubbard model.
- I have been implementing general purpose multiple precision version of BLAS/LAPACK routines; MPACK (MBLAS/MLAPACK).
- SDPA-GMP 6, 7.0.2, 3, 5 etc. internal versions.
- SDPA-GMP 7.1.0 has been released in 2008/4/10.
- MPACK (MBLAS/MLAPACK) 0.0.8 has been released in 2009/1/8.
- SDPA-GMP 7.1.2 supports MPACK 0.0.9 in 2009/2/5.
- QD and DD version are requested by Hans D. Mittelmann and Fujisawa-san.
- SDPA-GMP, QD and DD 7.1.2 have been released in 2009/3/21.
- SDPA-GMP, QD and DD is on NEOS server
- Extensive tests with Waki-san and Fujisawa-san.

History

- YAMASHITA-san told me NAKATA Kazuhide-san's student implemented MP version of SDP solver in Java using fixed point numbers.
- I started to implement SDPA-GMP6 based on SDPA6. The first working version: 2006/12/5. Lot of discussions with NAKATA Kazuhide-san.
- First appearance of SDPA-GMP 6 is Journal of chemical physics 128, 16 164113 (2008); to solve strong correlation limit of Hubbard model.
- I have been implementing general purpose multiple precision version of BLAS/LAPACK routines; MPACK (MBLAS/MLAPACK).
- SDPA-GMP 6, 7.0.2, 3, 5 etc. internal versions.
- SDPA-GMP 7.1.0 has been released in 2008/4/10.
- MPACK (MBLAS/MLAPACK) 0.0.8 has been released in 2009/1/8.
- SDPA-GMP 7.1.2 supports MPACK 0.0.9 in 2009/2/5.
- QD and DD version are requested by Hans D. Mittelmann and Fujisawa-san.
- SDPA-GMP, QD and DD 7.1.2 have been released in 2009/3/21.
- SDPA-GMP, QD and DD is on NEOS server
- Extensive tests with Waki-san and Fujisawa-san.

History

- YAMASHITA-san told me NAKATA Kazuhide-san's student implemented MP version of SDP solver in Java using fixed point numbers.
- I started to implement SDPA-GMP6 based on SDPA6. The first working version: 2006/12/5. Lot of discussions with NAKATA Kazuhide-san.
- First appearance of SDPA-GMP 6 is Journal of chemical physics 128, 16 164113 (2008); to solve strong correlation limit of Hubbard model.
- I have been implementing general purpose multiple precision version of BLAS/LAPACK routines; MPACK (MBLAS/MLAPACK).
- SDPA-GMP 6, 7.0.2, 3, 5 etc. internal versions.
- SDPA-GMP 7.1.0 has been released in 2008/4/10.
- MPACK (MBLAS/MLAPACK) 0.0.8 has been released in 2009/1/8.
- SDPA-GMP 7.1.2 supports MPACK 0.0.9 in 2009/2/5.
- QD and DD version are requested by Hans D. Mittelmann and Fujisawa-san.
- SDPA-GMP, QD and DD 7.1.2 have been released in 2009/3/21.
- SDPA-GMP, QD and DD is on NEOS server
- Extensive tests with Waki-san and Fujisawa-san.

History

- YAMASHITA-san told me NAKATA Kazuhide-san's student implemented MP version of SDP solver in Java using fixed point numbers.
- I started to implement SDPA-GMP6 based on SDPA6. The first working version: 2006/12/5. Lot of discussions with NAKATA Kazuhide-san.
- First appearance of SDPA-GMP 6 is Journal of chemical physics 128, 16 164113 (2008); to solve strong correlation limit of Hubbard model.
- I have been implementing general purpose multiple precision version of BLAS/LAPACK routines; MPACK (MBLAS/MLAPACK).
- SDPA-GMP 6, 7.0.2, 3, 5 etc. internal versions.
- SDPA-GMP 7.1.0 has been released in 2008/4/10.
- MPACK (MBLAS/MLAPACK) 0.0.8 has been released in 2009/1/8.
- SDPA-GMP 7.1.2 supports MPACK 0.0.9 in 2009/2/5.
- QD and DD version are requested by Hans D. Mittelmann and Fujisawa-san.
- SDPA-GMP, QD and DD 7.1.2 have been released in 2009/3/21.
- SDPA-GMP, QD and DD is on NEOS server
- Extensive tests with Waki-san and Fujisawa-san.

History

- YAMASHITA-san told me NAKATA Kazuhide-san's student implemented MP version of SDP solver in Java using fixed point numbers.
- I started to implement SDPA-GMP6 based on SDPA6. The first working version: 2006/12/5. Lot of discussions with NAKATA Kazuhide-san.
- First appearance of SDPA-GMP 6 is Journal of chemical physics 128, 16 164113 (2008); to solve strong correlation limit of Hubbard model.
- I have been implementing general purpose multiple precision version of BLAS/LAPACK routines; MPACK (MBLAS/MLAPACK).
- SDPA-GMP 6, 7.0.2, 3, 5 etc. internal versions.
- SDPA-GMP 7.1.0 has been released in 2008/4/10.
- MPACK (MBLAS/MLAPACK) 0.0.8 has been released in 2009/1/8.
- SDPA-GMP 7.1.2 supports MPACK 0.0.9 in 2009/2/5.
- QD and DD version are requested by Hans D. Mittelmann and Fujisawa-san.
- SDPA-GMP, QD and DD 7.1.2 have been released in 2009/3/21.
- SDPA-GMP, QD and DD is on NEOS server
- Extensive tests with Waki-san and Fujisawa-san.

History

- YAMASHITA-san told me NAKATA Kazuhide-san's student implemented MP version of SDP solver in Java using fixed point numbers.
- I started to implement SDPA-GMP6 based on SDPA6. The first working version: 2006/12/5. Lot of discussions with NAKATA Kazuhide-san.
- First appearance of SDPA-GMP 6 is Journal of chemical physics 128, 16 164113 (2008); to solve strong correlation limit of Hubbard model.
- I have been implementing general purpose multiple precision version of BLAS/LAPACK routines; MPACK (MBLAS/MLAPACK).
- SDPA-GMP 6, 7.0.2, 3, 5 etc. internal versions.
- SDPA-GMP 7.1.0 has been released in 2008/4/10.
- MPACK (MBLAS/MLAPACK) 0.0.8 has been released in 2009/1/8.
- SDPA-GMP 7.1.2 supports MPACK 0.0.9 in 2009/2/5.
- QD and DD version are requested by Hans D. Mittelmann and Fujisawa-san.
- SDPA-GMP, QD and DD 7.1.2 have been released in 2009/3/21.
- SDPA-GMP, QD and DD is on NEOS server
- Extensive tests with Waki-san and Fujisawa-san.

History

- YAMASHITA-san told me NAKATA Kazuhide-san's student implemented MP version of SDP solver in Java using fixed point numbers.
- I started to implement SDPA-GMP6 based on SDPA6. The first working version: 2006/12/5. Lot of discussions with NAKATA Kazuhide-san.
- First appearance of SDPA-GMP 6 is Journal of chemical physics 128, 16 164113 (2008); to solve strong correlation limit of Hubbard model.
- I have been implementing general purpose multiple precision version of BLAS/LAPACK routines; MPACK (MBLAS/MLAPACK).
- SDPA-GMP 6, 7.0.2, 3, 5 etc. internal versions.
- SDPA-GMP 7.1.0 has been released in 2008/4/10.
- MPACK (MBLAS/MLAPACK) 0.0.8 has been released in 2009/1/8.
- SDPA-GMP 7.1.2 supports MPACK 0.0.9 in 2009/2/5.
- QD and DD version are requested by Hans D. Mittelmann and Fujisawa-san.
- SDPA-GMP, QD and DD 7.1.2 have been released in 2009/3/21.
- SDPA-GMP, QD and DD is on NEOS server
- Extensive tests with Waki-san and Fujisawa-san.

History

- YAMASHITA-san told me NAKATA Kazuhide-san's student implemented MP version of SDP solver in Java using fixed point numbers.
- I started to implement SDPA-GMP6 based on SDPA6. The first working version: 2006/12/5. Lot of discussions with NAKATA Kazuhide-san.
- First appearance of SDPA-GMP 6 is Journal of chemical physics 128, 16 164113 (2008); to solve strong correlation limit of Hubbard model.
- I have been implementing general purpose multiple precision version of BLAS/LAPACK routines; MPACK (MBLAS/MLAPACK).
- SDPA-GMP 6, 7.0.2, 3, 5 etc. internal versions.
- SDPA-GMP 7.1.0 has been released in 2008/4/10.
- MPACK (MBLAS/MLAPACK) 0.0.8 has been released in 2009/1/8.
- SDPA-GMP 7.1.2 supports MPACK 0.0.9 in 2009/2/5.
- QD and DD version are requested by Hans D. Mittelmann and Fujisawa-san.
- SDPA-GMP, QD and DD 7.1.2 have been released in 2009/3/21.
- SDPA-GMP, QD and DD is on NEOS server
- Extensive tests with Waki-san and Fujisawa-san.

History

- YAMASHITA-san told me NAKATA Kazuhide-san's student implemented MP version of SDP solver in Java using fixed point numbers.
- I started to implement SDPA-GMP6 based on SDPA6. The first working version: 2006/12/5. Lot of discussions with NAKATA Kazuhide-san.
- First appearance of SDPA-GMP 6 is Journal of chemical physics 128, 16 164113 (2008); to solve strong correlation limit of Hubbard model.
- I have been implementing general purpose multiple precision version of BLAS/LAPACK routines; MPACK (MBLAS/MLAPACK).
- SDPA-GMP 6, 7.0.2, 3, 5 etc. internal versions.
- SDPA-GMP 7.1.0 has been released in 2008/4/10.
- MPACK (MBLAS/MLAPACK) 0.0.8 has been released in 2009/1/8.
- SDPA-GMP 7.1.2 supports MPACK 0.0.9 in 2009/2/5.
- QD and DD version are requested by Hans D. Mittelmann and Fujisawa-san.
- SDPA-GMP, QD and DD 7.1.2 have been released in 2009/3/21.
- SDPA-GMP, QD and DD is on NEOS server
- Extensive tests with Waki-san and Fujisawa-san.

History

- YAMASHITA-san told me NAKATA Kazuhide-san's student implemented MP version of SDP solver in Java using fixed point numbers.
- I started to implement SDPA-GMP6 based on SDPA6. The first working version: 2006/12/5. Lot of discussions with NAKATA Kazuhide-san.
- First appearance of SDPA-GMP 6 is Journal of chemical physics 128, 16 164113 (2008); to solve strong correlation limit of Hubbard model.
- I have been implementing general purpose multiple precision version of BLAS/LAPACK routines; MPACK (MBLAS/MLAPACK).
- SDPA-GMP 6, 7.0.2, 3, 5 etc. internal versions.
- SDPA-GMP 7.1.0 has been released in 2008/4/10.
- MPACK (MBLAS/MLAPACK) 0.0.8 has been released in 2009/1/8.
- SDPA-GMP 7.1.2 supports MPACK 0.0.9 in 2009/2/5.
- QD and DD version are requested by Hans D. Mittelmann and Fujisawa-san.
- SDPA-GMP, QD and DD 7.1.2 have been released in 2009/3/21.
- SDPA-GMP, QD and DD is on NEOS server
- Extensive tests with Waki-san and Fujisawa-san.

History

- YAMASHITA-san told me NAKATA Kazuhide-san's student implemented MP version of SDP solver in Java using fixed point numbers.
- I started to implement SDPA-GMP6 based on SDPA6. The first working version: 2006/12/5. Lot of discussions with NAKATA Kazuhide-san.
- First appearance of SDPA-GMP 6 is Journal of chemical physics 128, 16 164113 (2008); to solve strong correlation limit of Hubbard model.
- I have been implementing general purpose multiple precision version of BLAS/LAPACK routines; MPACK (MBLAS/MLAPACK).
- SDPA-GMP 6, 7.0.2, 3, 5 etc. internal versions.
- SDPA-GMP 7.1.0 has been released in 2008/4/10.
- MPACK (MBLAS/MLAPACK) 0.0.8 has been released in 2009/1/8.
- SDPA-GMP 7.1.2 supports MPACK 0.0.9 in 2009/2/5.
- QD and DD version are requested by Hans D. Mittelmann and Fujisawa-san.
- SDPA-GMP, QD and DD 7.1.2 have been released in 2009/3/21.
- SDPA-GMP, QD and DD is on NEOS server
- Extensive tests with Waki-san and Fujisawa-san.

History

- YAMASHITA-san told me NAKATA Kazuhide-san's student implemented MP version of SDP solver in Java using fixed point numbers.
- I started to implement SDPA-GMP6 based on SDPA6. The first working version: 2006/12/5. Lot of discussions with NAKATA Kazuhide-san.
- First appearance of SDPA-GMP 6 is Journal of chemical physics 128, 16 164113 (2008); to solve strong correlation limit of Hubbard model.
- I have been implementing general purpose multiple precision version of BLAS/LAPACK routines; MPACK (MBLAS/MLAPACK).
- SDPA-GMP 6, 7.0.2, 3, 5 etc. internal versions.
- SDPA-GMP 7.1.0 has been released in 2008/4/10.
- MPACK (MBLAS/MLAPACK) 0.0.8 has been released in 2009/1/8.
- SDPA-GMP 7.1.2 supports MPACK 0.0.9 in 2009/2/5.
- QD and DD version are requested by Hans D. Mittelmann and Fujisawa-san.
- SDPA-GMP, QD and DD 7.1.2 have been released in 2009/3/21.
- SDPA-GMP, QD and DD is on NEOS server
- Extensive tests with Waki-san and Fujisawa-san.

Summary

- We developed multiple precision version of SDP solver. SDPA-GMP, SDPA-QD and SDPA-DD.
- Can solve SDPs very accurately.
- MPACK 0.0.9: Multiple precision version of LAPACK/BLAS: development ongoing.

- Outlook
  - Faster SDPA-GMP, QD, DD and MPACK, parallel and multicore versions.
  - More routines for MPACK.
  - Higher accuracy to SDPA; minimal use of multiple precision arithmetic.