Outline of High-Speed Quad-Precision Arithmetic Package ASLQUAD

OGATA Ryusei, KUBO Yoshiyuki, TAKEI Toshifumi

Abstract

The ASLQUAD high-speed quad-precision arithmetic package reduces numerical errors in the traditional double-precision numerical simulations and enables calculations with higher reliability. ASLQUAD also pursues improvement of the user program portability and abundance of functions at the same time as achieving the high-speed operational capability of NEC's Supercomputer SX Series.

This paper summarizes the functions and features of ASLQUAD and explains the usefulness of its high-precision arithmetic operations by taking an eigenvalue calculation as an example.

Keywords

quad-precision arithmetic operations, multi-precision arithmetic operations
high-precision arithmetic operations, mathematical library, numerical error

1. Introduction

One of the problems likely to be encountered in the use of supercomputers is the loss of precision due to the accumulation of numerical rounding errors. For example, convergence calculations and evaluation of the precision of calculation results are sensitive to numerical errors. A reduction in numerical errors is therefore an important issue. To achieve such issues in traditional large-scale simulations, developments have been based on quad-precision arithmetic operations supported by hardware and compilers at the same time as improving the precision of the numerical calculation algorithms in use.

In this trend, the demands for high-speed, high-precision calculations are increasing more than ever. Nevertheless, simply using the given hardware and compilers as they are without software control is not meeting the demands for high-speed, high-precision calculations because it does not allow a flexible selection of the factors determining the high-precision calculation performance. These factors are the internal data format, arithmetical method and guaranteed precision. Such usage of the hardware and compilers also results in an insufficient performance optimization.

At NEC, we are developing the ASLQUAD high-speed, quad-precision arithmetic package in order to support high-speed, high-precision arithmetic operations with software control. In the following sections of this paper, we will summarize the functions and features available when ASLQUAD is used in the Supercomputer SX Series and also demonstrate the usefulness of high-precision calculations by taking the eigenvalue calculation with ASLQUAD as an example.

2. Features of ASLQUAD

ASLQUAD is a development tool that powerfully supports the creation of numerical simulation programs with high-precision calculations. The use of ASLQUAD makes it possible to create high-speed, high-precision numerical simulation programs without being concerned about detailed matters in the multi-precision algorithms. The program development efficiency is thereby significantly improved.

ASLQUAD is composed of the basic arithmetic function group, the subroutine library function group gathering numerical calculation functions and the auxiliary function group for supporting users in programming (**Fig. 1**). These function groups are designed to be easily incorporated into user programs, that are written in Fortran90, and can be used from not only programs parallelized for shared-memory but also MPI programs.

Vector Arithmetic Register (VAR) of the SX Series has a width of 64 bits. To achieve high-speed quad-precision arithmetic operations using VARs, ASLQUAD adopts the double-double format, that is well known in the field of multi-precision arithmetics [1)-2)] , as the quad-precision data format. The double-double format represents the quad-precision data (128-bit)

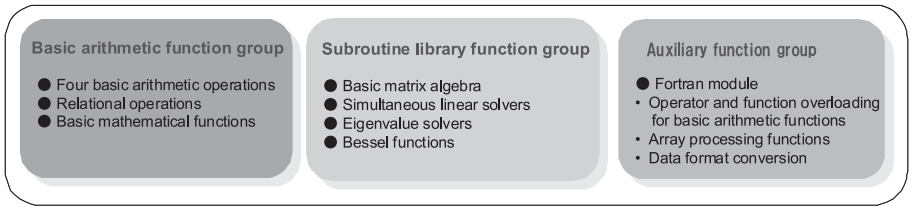


Fig. 1 Function groups of high-speed quad-precision arithmetic package ASLQUAD.

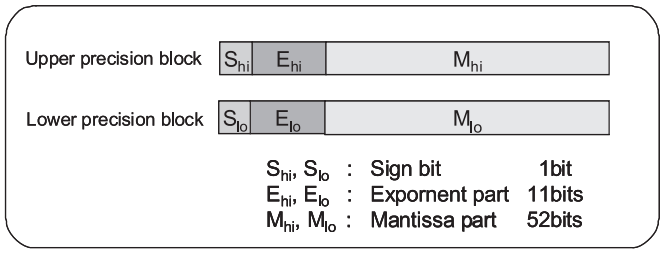


Fig. 2 Double-double format.

by using a pair of double-precision data (64-bit). **Fig. 2** shows the data format of the double-double format.

Arithmetic operations based on the double-double format can be performed by emulating a single quad-precision arithmetic operation in multiple double-precision arithmetic operations. As an example, an addition (C = A + B) based on double-double format arithmetic operation is shown below.

```
subroutine  A_add_B (Ahi、Alo、Bhi、Blo、Chi、Clo)
real(kind=8)  Ahi, Alo, Bhi, Blo, Chi, Clo,   z, r, zz
   z  =  Ahi + Bhi
   r  =  Ahi - z
  zz  =  ( ( ( r + Bhi ) + ( Ahi - ( r + z ) ) ) + Alo ) + Blo
   Chi = z + zz
   Clo = (z - Chi) + zz
end subroutine
```

Here, (Ahi, Bhi, Chi) represent respectively the upper precision blocks of (A, B, C), while (Alo, Blo, Clo) represent the lower precision blocks.For other calculations such as the four basic arithmetic operations and mathematical functions, a single quad-precision operation is emulated with the use of multiple double-precision arithmetic operations in the same way.

Some multi-precision arithmetic softwares applying this technique are also available on the web [3)-4)] . But those softwares have the disadvantage that the function calls for double-double precision arithmetic operations hinder optimization by the compiler. Therefore, the traditional multi-precision arithmetic softwares cannot make full use of the powerful vector pipelines of the SX Series. In order to overcome this disadvantage, ASLQUAD is designed to reduce the function calls by utilizing the nested inline expansion function of FORTRAN90/SX. For the four basic arithmetic operations of double-double format arithmetic operations and the elementary mathematical functions combining them, we have also made attempts such as a reduction in the number of operations and the simplification of conditional branching.

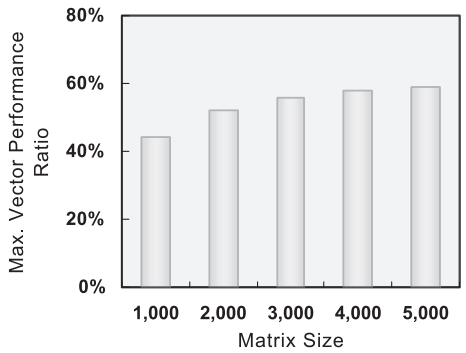
2.1 Basic Arithmetic Function Group

The basic arithmetic function group consists of the functions corresponding to the quad-precision four basic arithmetic operations and built-in functions supported by general commercial compilers. The basic arithmetic function group of the latest version R2.0 supports the following functions that use the double-double format arithmetic operations.

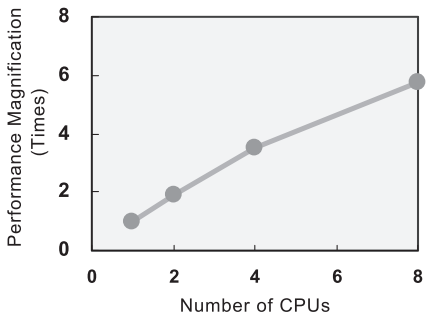
1) Four basic arithmetic operations and power operation (+, -, *, /, **)
2) Relational operations (>, < , = =, < =, >=, /=)
3) Type conversions (INT, DBLE, QEXT)
4) Summation/inner product (SUM, DOT_PRODUCT)
5) Absolute value/maximum value/minimum value (ABS/MAX/MIN)
6) Square root, inverse square root, cube root (SQRT, RSQRT, CBRT)
7) Trigonometric functions (SIN, COS, TAN)
8) Inverse trigonometric functions (ASIN, ACOS, ATAN)
9) Exponential functions (EXP, EXP2, EXP10)
10)Logarithmic functions (LOG, LOG2, LOG10)
11)Hyperbolic functions (SINH, COSH, TANH)

2.2 Subroutine Library Function Group

The subroutine library function group consists of sophisticated numerical calculation functions combining the functions in the basic arithmetic function group. Among the



(a) Maximum vector performance ratio (per CPU).



(b) Parallel performance magnifications within a single node (Matrix size 5,000 × 5,000).

Fig. 3 Performances of real symmetric generalized eigenvalue solvers of ASLQUAD.

numerical calculation techniques and their relevant basic techniques, those functions are selected for ASLQUAD that tend to be affected by numerical error, and are optimized in an advanced manner in consideration of vectorization and parallelization. The latest version of R2.0 supports the following functions.

(1) Basic Matrix Operation

- Matrix product

(2) Simultaneous Linear Direct Solvers (for dense matrix)

- LU factorization and equation solution after LU factorization
- Determinant and inverse matrix calculations

(3) Real Symmetric Eigenvalue Solvers/Generalized Eigenvalue Solvers

- Eigenvalue and eigenvector calculations
- Eigenvalues and eigenvectors with section definition

(4) Bessel Functions

- First-kind and second-kind integer-order Bessel functions

Fig. 3 shows (a) the graphs of the maximum vector performance ratio [*1] and (b) the parallelization performance magnification within a single node with respect to the number of CPUs [*2] , when the generalized eigenvalue solver (for real symmetric matrix) of ASLQUAD is run on the SX-8. It shows that the vector performance of each CPU is utilized by almost 50% of the peak and that the parallel performance is about 6 times with 8 CPUs.

Fig. 3 shows the results of measurements on the SX-8. Performance optimization for the SX-9 is still underway. However, we expect that the SX-9 will achieve the maximum vector performance ratio and parallelization performance magnification equivalent to those of the SX-8.

2.3 Auxiliary Function Group

The auxiliary function group consists of functions for supporting the data compatibility and program portability. These functions are provided as Fortran modules that can facilitate the basic arithmetic function group. In the Fortran module, the double-double format data is defined as structures and the invocation of the basic arithmetic function group is defined by overloading operators and functions. The user defined operators (and user defined functions) of Fortran90 are used. The functions supported by the basic arithmetic function group can also be utilized in array processing of Fortran90. In addition, the function for format conversion between the quad-precision data format supported by the SX Series and the double-double format is also provided in the Fortran module.

These functions allow users to use the basic arithmetic function group by simply quoting the Fortran module in their programs with coding manner close to the normal Fortran90 grammar. An image of a Fortran90 program using ASLQUAD is shown below. Line “TYPE(quad)” means the array declaration of the structure representing the double-double format as defined in the Fortran module.

[*1] Maximum vector performance ratio: Value obtained by dividing the double-precision floating-point arithmetic performance in the double-double arithmetic by the floating-point arithmetic processing capabilities (theoretical peak value) using only the adder and multipliers inside the vector unit.
[*2] Performance magnification: Value obtained by dividing the overall processing time by the processing time obtained by execution of a single CPU.

```
USE  ASLQUAD                    ! modu;e
TYPE(quad) ,  dimension(256,256)  ::  A, B,C
～
A=1q0;  B=2q0;  C=3q0           ! array processing
do  K = 1 , 256
      do  J = 1 , 256
            do  I = 1 , 256
               C( I, J )=C( I, J ) + A( I, K) * B( K, J)
enddo;  enddo;  enddo
～
END
```

3. Precision of Eigenvalue Calculation Using ASLQUAD

In this section, we will discuss a large-scale eigenvalue calculation as an example of the case in which a high-precision operation is required. The eigenvalue calculation has traditionally been required in various research fields including structural analysis and vibration analysis. However, depending on the properties of the matrix to be solved, solutions with high-precision are sometimes not able to be obtained in the calculations at double-precision. **Fig. 4** shows the relative errors of double-precision and quad-precision calculations with the eigenvalue of the real symmetric matrix called the Frank matrix. Here, the Frank matrix refers to a matrix each component A_{ij} of which can be expressed as follows.

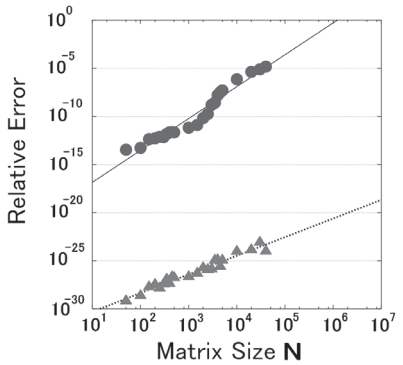
$$A_{ij} = N - \max(i, j) + 1$$

It is also known that the exact solution of eigenvalue, E_i, can be expressed as follows [5].

$$E_i = \frac{1}{\sqrt{1 - \cos\dfrac{(2i-1)\pi}{2N+1}}}$$

We used the eigenvalue solver of ASLQUAD (ASLQ_DCSMAN) in the eigenvalue calculations at quad-precision, and the eigenvalue solver of NEC's Advanced Scientific Library (ASL) in the double-precision eigenvalue calculations. The eigenvalue at each precision was calculated by using the Householder transformation and the root-free QR decomposition.

When matrix size N exceeds 100,000, the relative errors in double-precision calculations were below O (10^{-5}). In other words, when it is required to obtain the eigenvalue of a matrix with matrix size N over 100,000 to 5 or more significant digits, the precision of the calculation should be higher than the double-precision (for example, quad-precision). Also, if operation is continued using the results of double-precision calculations, the accumulation of numerical errors would eventually deteriorate the computing accuracy of the entire program. Although the results shown here were obtained from a Frank matrix, the quad-precision operation is regarded as being effective whenever a high-precision eigenvalue is required for any ill-conditioned matrices as well as for Frank matrices.



(●: Double precision. ▲: Quad precision.)

Fig. 4 Relative error in eigenvalue of Frank matrix.

4. Conclusion

In this paper, we summarized the functions and features of the ASLQUAD high-speed, quad-precision arithmetic package. We also described the usefulness of the quad-precision arithmetic operation by actually comparing the relative errors by taking a large-scale eigenvalue calculation as an example. The advantages of this package are that it can optimize the quad-precision arithmetic operations of double-double format data using the nested inline expansion function of FORTRAN90/SX and that it has been optimized in an advanced manner for use with NEC's Supercomputer SX Series.

In the future, we intend to expand the high-precision arithmetic function, tune the subroutine library performance and support octo-precision arithmetic operations so that the hardware performance of the SX Series can achieve maximum utilization in high-precision arithmetic operations.

References

1) Dekker, T.J.: A floating-point technique for extending the available precision, Numerische Mathematik, Vol.18, pp.224-242, 1971
2) Knuth, D. E.: The art of computer programming Vol.2 Seminumerical algorithms, 3rd Edition, Addison- Wesley, 1998
3) DDFUN90; http://crd.lbl.gov/~dhbailey/mpdist/index.html
4) QD;http://crd.lbl.gov/~dhbailey/mpdist/index.html
5) Frank, Werner L. “Computing Eigenvalues of Computing Eigenvalues of Complex Matrices by Determinant Evaluation and by Methods of Danilewski and Wielandt,” Jour. SIAM, 6, 378-392, 1961

Authors' Profiles

OGATA Ryusei
HPC Marketing Promotion Division,
1st Computers Operations Unit,
NEC Corporation

KUBO Yoshiyuki
HPC Solution Manager,
HPC Marketing Promotion Division,
1st Computers Operations Unit,
NEC Corporation

TAKEI Toshifumi
Senior Manager,
HPC Marketing Promotion Division,
1st Computers Operations Unit,
NEC Corporation