Hue University Journal of Science: Natural Science
Vol. 128, No. 1B, 29–34, 2019

pISSN 1859–1388
eISSN 2615–9678

IMPROVEMENT OF PRECISION OF NUMERICAL CALCULATIONS USING "MULTIPLE PRECISION COMPUTATION" PACKAGE

Hanh H. Nguyen, Tran Duong Anh Tai, Duc T. Hoang, Uyen N. Le, Tang Thi Bich Van, Vinh N. T. Pham*

Department of Physics, Ho Chi Minh University of Education, 280 An Duong Vuong St., Dist. 5, Ho Chi Minh City, Vietnam

Correspondence to Vinh N. T. Pham (email: vinhpnt@hcmue.edu.vn)

(Received: 15–5–2019; Accepted: 6–6–2019)

Abstract. In this work, the program so-called "Multiple Precision Computation" (MPC) proposed by Smith in 2003 is introduced and embedded into conventional codes for considerably improving the precision of numerical calculations. The procedure is evaluated for improvement and validated by using the comparison between calculations incorporating MPC and those using regular double-precision declarations and results obtained with well-known software Mathematica, respectively. Several representatively fundamental problems are taken into account for illustration.

Keywords: multiple precision computation, computational physics, FM package

1 Introduction

Recently, numerical computation is an efficient tool assisting scientists and engineers to not only evaluate the accuracy of experimental results and theoretical models but also predict those in extreme regimes which are not able to be considered in experiments or computed analytically due to the limitation of current technologies or mathematical techniques. There exist a number of programming languages for computation such as FORTRAN, Python, C++, Java. Among them, FORTRAN is still widely used in the community of computational scientists due to its simplicity and advantages regarding the execution time, supporting libraries such as LAPACK and Intel MKL. Note that it is the limitation of random access memory (RAM) assigned to variables that reduce the precision of the numerical results and in several cases prevent us from achieving reasonable convergence. For instance, it is impossible to obtain an accurate ionization rate below 10^{-10} au of atomic or molecular systems as the electric field strength is extremely small [1].

To achieve highly precise results, each intermediate step has to be neatly considered. One may use various commercial software such as Mathematica or Matlab. However, their cost is relatively high, and it is not easy to integrate such programs to our own ones written in different programming languages. An alternative way to overcome this obstacle is to scrutinize the essence of high-precision computation using open-source programming languages such as FORTRAN due to its simplicity and popularity. In 1978, Brent introduced the Multiple-Precision Arithmetic Package (MP package) to support high-precision calculations [2]. The MP package includes four main modules for converting default variable declarations in FORTRAN to high-precision declarations. The MP package also provides high-precision constants and special functions. Nonetheless, the execution is more time-consuming compared with programs using default declarations, and the MP package does not either support complex numbers. In 1991, Smith introduced the Multiple-precision package (FM package) based on subroutines in MP, which is improved in terms of speed and precision due to the use of improved algorithms for computing the elementary functions in multiple-precision [3, 4]. Initially, the FM package was written in FORTRAN 77 and supported for integer, real variables, and several fundamental functions. In 1998, the FM package was extended for computing in complex number sets [5]. Thereafter, the FM package was updated and upgraded consecutively [6–8], and the latest version written in FORTRAN 95 can be found on Smith's website [9].

It is interesting to note that the FM package is free to use. Hence, we provide in this paper a brief introduction to the FM package and our evaluation of the differences between the results computed with and without the FM package in terms of precision and stability. We consider four fundamental problems, including derivatives, integrals, finding roots, and solving ordinary differential equations. We note that the results presented in this paper are preliminary evaluation on the ability of applying the FM package for further scrutinizing the interaction between atoms, molecules, and laser fields at extremely weak intensity regime and the thermodynamic properties of the ideal Fermi gas confined harmonically at the vicinity of 0 K that are not able to be computed due to the limitation of the accuracy. It is also important to have benchmarks for self-assessment. We choose well-known Mathematica software since it supports fundamental functions at high precision and is reliable to be used in academic communities [8, 10].

2 An introduction to the FM package

In this section, we present the structure of the FM package. This package consists of three systematic files named FMSAVE.f95, FM.f95, and FMZM90.f95 whose description and contents are shown in Table 1. Each of them contains libraries for multiple-precision operations, global variables, and modules for type interfaces. These files can be downloaded from the website [9]. Then, the users could modify these files to suit their desires. However, due to its complexity, the users are encouraged to know how to embed these files into their codes and make use of them.

To compile and run a program utilizing the FM package, these systematic files, together with the main file, have to be put into a similar directory. We then compile these files to create object files (*.o) before linking them to the main program. Note that all programs in this paper were compiled by the gfortran compiler on Ubuntu OS.

Table 1. List of systematic files of the FM package

File	Content
FMSAVE.f95	- Including 488 lines of code. - Module for FM internal global variables.
FM.f95	- Including 64,738 lines of code. - Subroutine library for multiple-precision operations.
FMZM90.f95	- Including 49,781 lines of code. - Module for derived type interfaces.

pISSN 1859–1388
eISSN 2615–9678

3 Results and discussion

We proceed to illustrate the improvement of precision as the FM package is incorporated. The results with and without the presence of the FM package are shown together with the benchmarks obtained by Mathematica [10]. We consider several fundamental problems, including first-order derivative, integral, root finding, and ordinary differential equations, since they are vital for further numerical calculations in computational physics. For the sake of simplicity, in the following, we refer to the results obtained using conventional double-precision declarations (without FM package) and with FM package as double precision and FM, respectively.

The first-order derivative is considered using central-difference formula as

$$f'(x_0) = \frac{f(x_0 + H) - f(x_0 - H)}{2H} \qquad (1)$$

where H is the step size. The two representative problems are taken into account to make assessments, as shown in Table 2.

In principle, the smaller the step size H is, the better the convergence should be. This fact holds well for the case of FM, while the results exhibit poor stability for conventional declarations, especially at a very small step size due to round-off errors. For the smallest step size $H = 10^{-12}$, the FM result is well consistent with that obtained by Mathematica up to 24 significant digits. Note that such convergence is similar for all considered cases that are not shown here.

The next fundamental problem considered is solving ordinary differential equations (ODEs) using the four-order Runge-Kutta algorithm [11]. Like the previous example, we also take into consideration two ODEs with various step sizes H, and the results are presented in Table 3.

Table 2. Numerical results for the first-order derivative in Eq. (1) when using conventional declarations and taking into account the FM package. [*] Results obtained by Mathematica

H	Double precision	FM
$f(x) = 2e^{2x}$ $f'(x)\big\|_{x_0=0.25} =$ **6.594885082800512587394603**15125665428661 [*]		
10^{-4}	**6.594885**12676498	**6.594885**12676641322732982158830541565345
10^{-8}	**6.59488508247818**	**6.5948850828005130270536086712908355727**
10^{-12}	**6.59494681087835**	**6.59488508280051258739460754784670948700**
$f(x) = 3\sin(x)$ $f'(x)\big\|_{x_0=1} =$ **1.620906917604419152202809**82232892981119 [*]		
10^{-4}	**1.62090691490313**	**1.62090691490290762421286699967298755213**
10^{-8}	**1.62090691979699**	**1.62090691760441912518769452892194407623**
10^{-12}	**1.62092561595273**	**1.62090691760441915220280955217777687715**

Table 3. Numerical results for the ODEs when using conventional declarations and taking into account the FM package. [*] Results obtained by Mathematica

H	Double precision	FM
$y'+0.25y-9.8=0, y(0)=0$ $y(1)=$ **8.671009303600929164789325**534449830625967 [*]		
10^{-3}	**8.671009303**598439	**8.67100930360092**89162919981883029299 12347
10^{-4}	**8.671009303600931**	**8.6710093036009291647644804606185699361 14**
10^{-5}	**8.671009303601004**	**8.67100930360092916478932304998903159185 5**
$y''+0.25y'-9.8=0, y(0)=2, y'(0)=0$ $y(1)=$ **6.5159627855962833408426**97862200677496133 [*]		
10^{-3}	**6.515962785596286**	**6.515962785596284334832007246788280350613**
10^{-4}	**6.515962785596270**	**6.515962785596283340942078157525720255543**
10^{-5}	**6.515962785596305**	**6.515962785596283340842707800043873632581**

It is apparent that the results with conventional declarations in the two examples match relatively well to the exact value, and the highest number of significant digits that can be reached is 14. However, for a smaller step size (i.e., $H = 10^{-5}$), the numerical results turn out to be unstable as previous cases. Meanwhile, as the FM package is incorporated, the results are more consistent with the benchmarks obtained by Mathematica at smaller step sizes, up to 24 significant digits for $H = 10^{-5}$.

We proceed to consider the numerical integration using the Gauss-Legendre method.

Table 4. Numerical results for the integrals when using conventional declarations and taking into account the FM package. [*] Results obtained by Mathematica

N	Double precision	FM
$I=\int_0^1 e^{x^2}dx$ I = **1.4626517459071816088040485868569881551 2** [*]		
10	**1.46265174590717**	**1.4626517459071816**0263058281805599785342
30	**1.46265174590716**	**1.46265174590718160880404858685698815512**
50	**1.46265174590712**	**1.46265174590718160880404858685698815512**
$I=\int_1^2 \frac{dx}{x(x+1)^2}$ I = **0.121015405785114260772552339327160764 83** [*]		
10	**0.121015405785113**	**0.12101540578511374723804916382155681231**
30	**0.121015405785113**	**0.12101540578511426077255233932716076483**
50	**0.121015405785109**	**0.12101540578511426077255233932716076483**

Two representative problems are evaluated and shown in Table 4.

Table 4 indicates that in the case of integration using the Gauss-Legendre method, the convergence is good and up to 14 significant digits even for conventional declarations at a small number of quadrature points since the Gauss-Legendre algorithm is one of the most stable algorithms for numerical integration [11]. However, several computational problems require much higher levels of convergence, such as the calculation of the ionization rate of atomic systems for extremely small electric field strength. Again, the FM package enables us to significantly improve the precision of numerical calculations, and the consistency between our calculations and Mathematica is around 30 significant digits.

The next consideration in the present work is root-finding problems. Here, we use the Newton-Ralphson algorithm due to its accuracy and rapid convergence compared with other methods [11]. We firstly use a bisector to minimize the interval of the root, then apply the Newton-Ralphson algorithm to precisely determine it. The number of loops used in bisection is 35, while that of Newton-Ralphson is only 1. The details are presented in Table 5. Again, the results obtained from double precision is also consistent up to 15 significant digits, while the consistency is improved up to 30 significant digits as the FM package is added to the main program. Obviously, the FM package enables us to extremely improve numerical results in comparison with those of declared double-precision type. Although these four problems cannot cover all features of computational physics, they are very fundamental and essential for higher levels of numerical study associating with more complex obstacles.

4 Conclusion

This study aims to present a tool so-called "Multiple Precision Computation" for high-precision scientific calculations. We focus on the illustration of the efficiency once the FM package is used via a set of four fundamental problems. The results deductively indicate a high improvement of precision while using the FM package in comparison with the cases of conventional variable declarations. This paves a way to overcome real obstacles in computational physics such as the calculation of the ionization rate of atomic systems for extremely weak electric field or consideration of entropy of fermion gas as the absolute temperature decreases very closed to 0 K. These problems are postponed to the next projects.

Table 5. Numerical results for root-finding problems when using conventional declarations and taking into account the FM package. [*] Results obtained by Mathematica

N	Double precision	FM
$13x^5 + x^4 + 2x^3 + 2x^2 + 4 = 0, \quad x \in [-1, 0]$ x = -**0.8154583398327741821685623818604230656853** [*]		
35	**-0.81545833983277**5	-**0.8154583398327741821685623818604230656853**
$3e^{-x} + x - 3 = 0, \quad x \in [2, 3]$ x = **2.8214393721220788934031913302944851953459** [*]		
35	**2.8214393721220**8	**2.8214393721220788934031913302944851953459**

References

1. Batishchev PA, Tolstikhin OI, Morishita T. Atomic Siegert states in an electric field: Transverse momentum distribution of the ionized electrons. Physical Review A. 2010 08 17;82(2).
2. Brent RP. A Fortran Multiple-Precision Arithmetic Package. ACM Transactions on Mathematical Software. 1978 03 01;4(1):57-70.
3. Smith DM. Algorithm 693: a FORTRAN package for floating-point multiple-precision arithmetic. ACM Transactions on Mathematical Software. 1991 06 01;17(2):273-283.
4. Smith DM. Efficient multiple-precision evaluation of elementary functions. Mathematics of Computation. 1989;52:131-134.
5. Smith DM. Algorithm 786: multiple-precision complex arithmetic and functions. ACM Transactions on Mathematical Software. 1998 Dec 01;24(4):359-367.
6. Smith D. Using multiple-precision arithmetic. Computing in Science & Engineering. 2003 07;5(4):88-93.
7. Smith DM. Algorithm 814: Fortran 90 software for floating-point multiple precision arithmetic, gamma and related functions. ACM Transactions on Mathematical Software. 2001 Dec 01;27(4):377-387.
8. Smith DM. Algorithm 911. ACM Transactions on Mathematical Software. 2011 02 01;37(4):1-16.
9. Smith DM. Multiple Precision Computation. https://dmsmith.lmu.build/ (accessed on 08/05/2019)
10. https://www.wolfram.com/mathematica/compare-mathematica/ (accessed on 08/05/2019)
11. Press WH, Teukolsky SA, Vetterling W.T, Flannery BP. Numerical Recipes in FORTRAN. Cambridge University Press, Cambridge; 1992.