Computer Physics Communications 204 (2016) 209–213

Contents lists available at ScienceDirect

Computer Physics Communications

journal homepage: www.elsevier.com/locate/cpc

OpenMP Fortran and C programs for solving the time-dependent Gross–Pitaevskii equation in an anisotropic trap

Luis E. Young-S.[a,*], Dušan Vudragović[b], Paulsamy Muruganandam[c], Sadhan K. Adhikari[a], Antun Balaž[b]

[a] Instituto de Física Teórica, UNESP – Universidade Estadual Paulista, 01.140-70 São Paulo, São Paulo, Brazil
[b] Scientific Computing Laboratory, Institute of Physics Belgrade, University of Belgrade, Pregrevica 118, 11080 Belgrade, Serbia
[c] School of Physics, Bharathidasan University, Palkalaiperur Campus, Tiruchirappalli 620024, Tamil Nadu, India

ARTICLE INFO

Article history:
Received 22 February 2016
Received in revised form
9 March 2016
Accepted 22 March 2016
Available online 6 April 2016

Keywords:
Bose–Einstein condensate
Gross–Pitaevskii equation
Split-step Crank–Nicolson scheme
Real- and imaginary-time propagation
C program
Fortran program
OpenMP
Partial differential equation

ABSTRACT

We present new version of previously published Fortran and C programs for solving the Gross–Pitaevskii equation for a Bose–Einstein condensate with contact interaction in one, two and three spatial dimensions in imaginary and real time, yielding both stationary and non-stationary solutions. To reduce the execution time on multicore processors, new versions of parallelized programs are developed using Open Multi-Processing (OpenMP) interface. The input in the previous versions of programs was the mathematical quantity nonlinearity for dimensionless form of Gross–Pitaevskii equation, whereas in the present programs the inputs are quantities of experimental interest, such as, number of atoms, scattering length, oscillator length for the trap, etc. New output files for some integrated one- and two-dimensional densities of experimental interest are given. We also present speedup test results for the new programs.

New version program summary

Program title: BEC-GP-OMP package, consisting of: (i) imag1d, (ii) imag2d, (iii) imag3d, (iv) imagaxi, (v) imagcir, (vi) imagsph, (vii) real1d, (viii) real2d, (ix) real3d, (x) realaxi, (xi) realcir, (xii) realsph.

Catalogue identifier: AEDU_v4_0.

Program Summary URL: http://cpc.cs.qub.ac.uk/summaries/AEDU_v4_0.html

Program obtainable from: CPC Program Library, Queen's University of Belfast, N. Ireland.

Licensing provisions: Apache License 2.0

No. of lines in distributed program, including test data, etc.: 130308.

No. of bytes in distributed program, including test data, etc.: 929062.

Distribution format: tar.gz.

Programming language: OpenMP C; OpenMP Fortran.

Computer: Any multi-core personal computer or workstation.

Operating system: Linux and Windows.

RAM: 1 GB.

Number of processors used: All available CPU cores on the executing computer.

Classification: 2.9, 4.3, 4.12.

Catalogue identifier of previous version: AEDU_v1_0, AEDU_v2_0.

Journal reference of previous version: Comput. Phys. Commun. 180 (2009) 1888; ibid. 183 (2012) 2021.

Does the new version supersede the previous version?: No. It does supersedes versions AEDU_v1_0 and AEDU_v2_0, but not AEDU_v3_0, which is MPI-parallelized version.

* Corresponding author.
E-mail addresses: luisevery@gmail.com (L.E. Young-S.), dusan.vudragovic@ipb.ac.rs (D. Vudragović), anand@cnld.bdu.ac.in (P. Muruganandam), adhikari@ift.unesp.br (S.K. Adhikari), antun.balaz@ipb.ac.rs (A. Balaž).

http://dx.doi.org/10.1016/j.cpc.2016.03.015
0010-4655/© 2016 Elsevier B.V. All rights reserved.

Nature of problem: The present OpenMP Fortran and C programs solve the time-dependent nonlinear partial differential Gross–Pitaevskii (GP) equation for a Bose–Einstein condensate in one (1D), two (2D), and three (3D) spatial dimensions in a harmonic trap with six different symmetries: axial- and radial-symmetry in 3D, circular-symmetry in 2D, and fully anisotropic in 2D and 3D.

Solution method: The time-dependent GP equation is solved by the split-step Crank–Nicolson method by discretizing in space and time. The discretized equation is then solved by propagation, in either imaginary or real time, over small time steps. The method yields the solution of stationary and/or non-stationary problems.

Reasons for the new version: Previously published Fortran and C programs [1,2] for solving the GP equation are recently enjoying frequent usage [3] and application to a more complex scenario of dipolar atoms [4]. They are also further extended to make use of general purpose graphics processing units (GPGPU) with Nvidia CUDA [5], as well as computer clusters using Message Passing Interface (MPI) [6]. However, a vast majority of users use single-computer programs, with which the solution of a realistic dynamical 1D problem, not to mention the more complicated 2D and 3D problems, could be time consuming. Now practically all computers have multicore processors, ranging from 2 up to 18 and more CPU cores. Some computers include motherboards with more than one physical CPU, further increasing the possible number of available CPU cores on a single computer to several tens. The present programs are parallelized using OpenMP over all the CPU cores and can significantly reduce the execution time. Furthermore, in the old version of the programs [1,2] the inputs were based on the mathematical quantity nonlinearity for the dimensionless form of the GP equation. The inputs for the present versions of programs are given in terms of phenomenological variables of experimental interest, as in Refs. [4,5], i.e., number of atoms, scattering length, harmonic oscillator length of the confining trap, etc. Also, the output files are given names which make identification of their contents easier, as in Refs. [4,5]. In addition, new output files for integrated densities of experimental interest are provided, and all programs were thoroughly revised to eliminate redundancies.

Summary of revisions: Previous Fortran [1] and C [2] programs for the solution of time-dependent GP equation in 1D, 2D, and 3D with different trap symmetries have been modified to achieve two goals. First, they are parallelized using OpenMP interface to reduce the execution time in multicore processors. Previous C programs [2] had OpenMP-parallelized versions of 2D and 3D programs, together with the serial versions, while here all programs are OpenMP-parallelized. Secondly, the programs now have input and output files with quantities of phenomenological interest. There are six trap symmetries and both in C and in Fortran there are twelve programs, six for imaginary-time propagation and six for real-time propagation, totaling to 24 programs. In 3D, we consider full radial symmetry, axial symmetry and full anisotropy. In 2D, we consider circular symmetry and full anisotropy. The structure of all programs is similar.

For the Fortran programs the input data (number of atoms, scattering length, harmonic oscillator trap length, trap anisotropy, etc.) are conveniently placed at the beginning of each program. For the C programs the input data are placed in separate input files, examples of which can be found in a directory named input. The examples of output files for both Fortran and C programs are placed in the corresponding directories called output. The programs then calculate the dimensionless nonlinearities actually used in the calculation. The provided programs use physical input parameters that give identical nonlinearity values as the previously published programs [1,2], so that the output files of the old and new programs can be directly compared. The output files are conveniently named so that their contents can be easily identified, following Refs. [4,5]. For example, file named <code>-out.txt, where <code> is a name of the individual program, is the general output file containing input data, time and space steps, nonlinearity, energy and chemical potential, and was named fort.7 in the old Fortran version. The file <code>-den.txt is the output file with the condensate density, which had the names fort.3 and fort.4 in the old Fortran version for imaginary- and real-time propagation, respectively. Other density outputs, such as the initial density, are commented out to have a simpler set of output files. The users can re-introduce those by taking out the comment symbols, if needed.

Table 1
Wall-clock execution times (in seconds) for runs with 1, 6 and 20 CPU cores with different programs using the Intel Fortran ifort (F-1, F-6 and F-20, respectively) and Intel C icc (C-1, C-6 and C-20, respectively) compilers using a workstation with two Intel Xeon E5-2650 v3 CPUs, with a total of 20 CPU cores, and obtained speedups (speedup-F = F-1/F-20, -speedupC = C-1/C-20) for 20 CPU cores.

	F-1	F-6	F-20	speedup-F	C-1	C-6	C-20	speedup-C
imag1d	32	26	26	1.2	45	28	27	1.7
imagcir	15	15	15	1.0	21	15	15	1.4
imagsph	12	12	12	1.0	19	12	10	1.9
real1d	194	84	72	2.7	304	110	98	3.1
realcir	132	62	57	2.3	182	78	64	2.8
realsph	119	68	67	1.8	191	76	61	3.1
imag2d	190	66	52	3.7	394	77	33	11.9
imagaxi	240	74	56	4.3	499	113	55	9.1
real2d	269	70	47	5.7	483	96	35	13.8
realaxi	132	37	25	5.3	237	51	22	10.8
imag3d	1682	472	366	4.6	2490	545	202	12.3
real3d	15,479	3494	2082	7.4	22,228	4558	1438	15.5

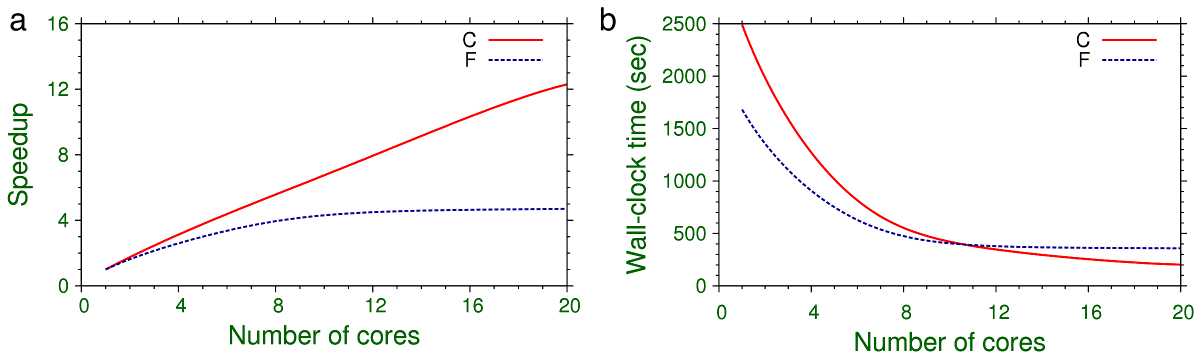

Fig. 1. (a) Speedup of the C and Fortran (F) imag3d programs as a function of the number of CPU cores, measured in a workstation with two Intel Xeon E5-2650 v3 CPUs. The speedup for the run with N CPU cores was calculated as the ratio between wall-clock execution times with one and N CPU cores. (b) Wall-clock time of the same runs as a function of the number of CPU cores.

Also, some new output files are introduced in this version of programs. The files <code>-rms.txt are the output files with values of root-mean-square (rms) sizes in the multi-variable cases. There are new files with integrated densities, such as imag2d-den1d_x.txt, where the first part (imag2d) denotes that the density was calculated with the 2D program imag2d, and the second part (den1d_x) stands for the 1D density in the x-direction, obtained after integrating out the 2D density $|\phi(x,y)|^2$ in the x–y plane over y-coordinate,

$$n_{1D}(x) = \int_{-\infty}^{\infty} dy|\phi(x,y)|^2. \tag{1}$$

Similarly, imag3d-den1d_x.txt and real3d-den1d_x.txt represent 1D densities from a 3D calculation obtained after integrating out the 3D density $|\phi(x,y,z)|^2$ over y- and z-coordinate. The files imag3d-den2d_xy.txt and real3d-den2d_xy.txt are the integrated 2D densities in the x–y plane from a 3D calculation obtained after integrating out the 3D density over the z-coordinate, and similarly for other output files. Again, calculation and saving of these integrated densities is commented out in the programs, and can be activated by the user, if needed.

In real-time propagation programs there are additional results for the dynamics saved in files, such as real2d-dyna.txt, where the first column denotes time, the second, third and fourth columns display rms sizes for the x-, y-, and r-coordinate, respectively. The dynamics is generated by multiplying the nonlinearity with a pre-defined factor during the NRUN iterations, and starting with the wave function calculated during the NPAS iterations. Such files were named fort.8 in the old Fortran versions of programs. There are similar files in the 3D real-time programs as well.

Often it is needed to get a precise stationary state solution by imaginary-time propagation and then use it in the study of dynamics using real-time propagation. For that purpose, if the integer number NSTP is set to zero in real-time propagation, the density obtained in the imaginary-time simulation is used as initial wave function for real-time propagation, as in Refs. [4,5]. In addition, at the end of output files <code>-out.txt, we have introduced two new outputs, wall-clock execution time and CPU time for each run.

We tested our programs on a workstation with two 10-core Intel Xeon E5-2650 v3 CPUs, and present results for all programs compiled with the Intel compiler. In Table 1 we show different wall-clock execution times for runs on 1, 6 and 20 CPU cores for Fortran and C. The corresponding columns "speedup-F" and "speedup-C" give the ratio of wall-clock execution times of runs on 1 and 20 CPU cores, and denote the actual measured speedup for 20 CPU cores. For the programs with effectively one spatial variable, the Fortran programs turn out to be quicker for small number of cores, whereas for larger number of CPU cores and for the programs with three spatial variables the C programs are faster. We also studied the speedup of the programs as a function of the number of available CPU cores. The performance for the imag3d Fortran and C programs is illustrated in Fig. 1(a) and (b), where we plot the speedup and actual wall-clock time of the imag3d C and Fortran programs as a function of number of CPU cores in a workstation with two Intel Xeon E5-2650 v3 CPUs, with a total of 20 CPU cores. The plot in Fig. 1(a) shows that the C program parallelizes more efficiently than the Fortran program. However, as the wall-clock time in Fortran for a single CPU core is less than that in C, the wall-clock times in both cases are comparable, viz. Fig. 1(b). A saturation of the speedup with the increase of the number of CPU cores is expected in all cases. However, the saturation is attained quicker in Fortran than in C programs, and therefore the use of C programs could be recommended for larger number of CPU cores. For a small number of CPU cores the Fortran programs should be preferable. For example, from Table 1 we see that for 6 CPU cores the Fortran programs are faster than the C programs. In Fig. 1(a) the saturation of the speedup of the Fortran program is achieved for approximately 10 CPU cores, when the wall-clock time of the C program crosses that of the Fortran program.

Additional comments:
This package consists of 24 programs, see Program title above. For the particular purpose of each program, please see descriptions below.

Running time:
Example inputs provided with the programs take less than 30 min in a workstation with two Intel Xeon Processors E5-2650 v3, 2 QPI links, 10 CPU cores (25 MB cache, 2.3 GHz).

Program summary (i), (v), (vi), (vii), (xi), (xii)

Program title: imag1d, imagcir, imagsph, real1d, realcir, realsph.

Title of electronic files in C: (imag1d.c and imag1d.h), (imagcir.c and imagcir.h), (imagsph.c and imagsph.h), (real1d.c and real1d.h), (realcir.c and realcir.h), (realsph.c and realsph.h).

Title of electronic files in Fortran 90: imag1d.f90, imagcir.f90, imagsph.f90, real1d.f90, realcir.f90, realsph.f90.

Maximum RAM memory: 1 GB for the supplied programs.

Programming language used: OpenMP C and Fortran 90.

Typical running time: Minutes on a modern four-core PC.

Nature of physical problem: These programs are designed to solve the time-dependent nonlinear partial differential GP equation in one spatial variable.

Method of solution: The time-dependent GP equation is solved by the split-step Crank–Nicolson method by discretizing in space and time. The discretized equation is then solved by propagation in imaginary time over small time steps. The method yields the solution of stationary problems.
Program summary (ii), (iv), (viii), (x)

Program title: imag2d, imagaxi, real2d, realaxi.

Title of electronic files in C: (imag2d.c and imag2d.h), (imagaxi.c and imagaxi.h), (real2d.c and real2d.h), (realaxi.c and realaxi.h).

Title of electronic files in Fortran 90: imag2d.f90, imagaxi.f90, real2d.f90, realaxi.f90.

Maximum RAM memory: 1 GB for the supplied programs.

Programming language used: OpenMP C and Fortran 90.

Typical running time: Hour on a modern four-core PC.

Nature of physical problem: These programs are designed to solve the time-dependent nonlinear partial differential GP equation in two spatial variables.

Method of solution: The time-dependent GP equation is solved by the split-step Crank–Nicolson method by discretizing in space and time. The discretized equation is then solved by propagation in imaginary time over small time steps. The method yields the solution of stationary problems.
Program summary (iii), (ix)

Program title: imag3d, real3d.

Title of electronic files in C: (imag3d.c and imag3d.h), (real3d.c and real3d.h).

Title of electronic files in Fortran 90: imag3d.f90, real3d.f90.

Maximum RAM memory: 1 GB for the supplied programs.

Programming language used: OpenMP C and Fortran 90.

Typical running time: Few hours on a modern four-core PC.

Nature of physical problem: These programs are designed to solve the time-dependent nonlinear partial differential GP equation in three spatial variables.

Method of solution: The time-dependent GP equation is solved by the split-step Crank–Nicolson method by discretizing in space and time. The discretized equation is then solved by propagation in imaginary time over small time steps. The method yields the solution of stationary problems.

Acknowledgments

L.E. Y.-S. acknowledges support by the FAPESP of Brazil under project 2012/21871-7 and 2014/16363-8. D.V. and A.B. acknowledge support by the Ministry of Education, Science, and Technological Development of the Republic of Serbia under projects OI1611005, ON171017 and III43007. P.M. acknowledges support by the Science and Engineering Research Board, Department of Science and Technology, Government of India under project No. EMR/2014/000644. S.K.A. acknowledges support by the CNPq of Brazil under project 303280/2014-0, and by the FAPESP of Brazil under project 2012/00451-0.

References

[1] P. Muruganandam, S.K. Adhikari, Comput. Phys. Commun. 180 (2009) 1888.
[2] D. Vudragović, I. Vidanović, A. Balaž, P. Muruganandam, S.K. Adhikari, Comput. Phys. Commun. 183 (2012) 2021.
[3] S. Gautam, (2016) e-print arXiv:1601.06020;
I. Vasić, A. Balaž, (2016) e-print arXiv:1602.03538;
T. Khellil, A. Balaž, A. Pelster, New J. Phys. (2016) e-print arXiv:1510.04985;
T. Khellil, A. Pelster, J. Stat. Mech.-Theory Exp. (2016) e-print arXiv:1511.08882;
J. Akram, A. Pelster, Phys. Rev. A 93 (2016) 023606;
J. Akram, A. Pelster, Phys. Rev. A 93 (2016) 033610;
T. Mithun, K. Porsezian, B. Dey, Phys. Rev. A 93 (2016) 013620;
L. Salasnich, S.K. Adhikari, Acta Phys. Pol. A 128 (2015) 979;
Y.H. Wang, A. Kumar, F. Jendrzejewski, R.M. Wilson, M. Edwards, S. Eckel, G.K. Campbell, C.W. Clark, New J. Phys. 17 (2015) 125012;
J.B. Sudharsan, R. Radha, H. Fabrelli, A. Gammal, B.A. Malomed, Phys. Rev. A 92 (2015) 053601;
V.S. Bagnato, D.J. Frantzeskakis, P.G. Kevrekidis, B.A. Malomed, D. Mihalache, Rom. Rep. Phys. 67 (2015) 5;
K.-T. Xi, J. Li, D.-N. Shi, Phys. B 459 (2015) 6;
H.L.C. Couto, W.B. Cardoso, J. Phys. B: At. Mol. Opt. Phys. 48 (2015) 025301;
E. Chiquillo, J. Phys. A: Math. Theor. 48 (2015) 475001;
S. Sabari, C.P. Jisha, K. Porsezian, V.A. Brazhnyi, Phys. Rev. E 92 (2015) 032905;
W. Wen, T.K. Shui, Y.F. Shan, C.P. Zhu, J. Phys. B: At. Mol. Opt. Phys. 48 (2015) 175301;
P. Das, P.K. Panigrahi, Laser Phys. 25 (2015) 125501;

Y.S. Wang, S.T. Ji, Y.E. Luo, Z.Y. Li, J. Korean. Phys. Soc. 67 (2015) L1504;
S.K. Adhikari, J. Phys. B: At. Mol. Opt. Phys. 48 (2015) 165303;
F.I. Moxley III, T. Byrnes, B. Ma, Y. Yan, W. Dai, J. Comput. Phys. 282 (2015) 303;
S.K. Adhikari, Phys. Rev. E 92 (2015) 042926;
R.R. Sakhel, A.R. Sakhel, H.B. Ghassib, Physica B 478 (2015) 68;
S. Gautam, S.K. Adhikari, Phys. Rev. A 92 (2015) 023616;
D. Novoa, D. Tommasini, J.A. Nóvoa-López, Phys. Rev. E 91 (2015) 012904;
S. Gautam, S.K. Adhikari, Laser Phys. Lett. 12 (2015) 045501;
K.-T. Xi, J. Li, D.-N. Shi, Physica B 459 (2015) 6;
S. Gautam, S.K. Adhikari, Phys. Rev. A 91 (2015) 013624;
A.I. Nicolin, M.C. Raportaru, A. Balaž, Rom. Rep. Phys. 67 (2015) 143;
S. Gautam, S.K. Adhikari, Phys. Rev. A 91 (2015) 063617;
E.J.M. Madarassy, V.T. Toth, Phys. Rev. D 91 (2015) 044041;
X. Antoine, R. Duboscq, Comput. Phys. Commun. 185 (2014) 2969;
S.K. Adhikari, L.E. Young-S, J. Phys. B: At. Mol. Opt. Phys. 47 (2014) 015302;
K. Manikandan, P. Muruganandam, M. Senthilvelan, M. Lakshmanan, Phys. Rev. E 90 (2014) 062905;
S.K. Adhikari, Phys. Rev. A 90 (2014) 055601;
A. Balaž, R. Paun, A.I. Nicolin, S. Balasubramanian, R. Ramaswamy, Phys. Rev. A 89 (2014) 023609;
S.K. Adhikari, Phys. Rev. A 89 (2014) 013630;
J. Luo, Commun. Nonlinear Sci. Numer. Simul. 19 (2014) 3591;
S.K. Adhikari, Phys. Rev. A 89 (2014) 043609;
K.-T. Xi, J. Li, D.-N. Shi, Physica B 436 (2014) 149;
M.C. Raportaru, J. Jovanovski, B. Jakimovski, D. Jakimovski, A. Mishev, Rom. J. Phys. 59 (2014) 677;
S. Gautam, S.K. Adhikari, Phys. Rev. A 90 (2014) 043619;
A.I. Nicolin, A. Balaž, J.B. Sudharsan, R. Radha, Rom. J. Phys. 59 (2014) 204;
K. Sakkaravarthi, T. Kanna, M. Vijayajayanthi, M. Lakshmanan, Phys. Rev. E 90 (2014) 052912;
S.K. Adhikari, J. Phys. B: At. Mol. Opt. Phys. 47 (2014) 225304;
R.K. Kumar, P. Muruganandam, Numerical studies on vortices in rotating dipolar Bose-Einstein condensates, in: Proceedings of the 22nd International Laser Physics Workshop, J. Phys. Conf. Ser. 497 (2014) 012036;
A.I. Nicolin, I. Rata, Density waves in dipolar Bose-Einstein condensates by means of symbolic computations, in: High-Performance Computing Infrastructure for South East Europe's Research Communities: Results of the HP-SEE User Forum 2012, in: Springer Series: Modeling and Optimization in Science and Technologies, vol. 2, 2014, p. 15;
S.K. Adhikari, Phys. Rev. A 89 (2014) 043615;
R.K. Kumar, P. Muruganandam, Eur. Phys. J. D 68 (2014) 289;
I. Vidanović, N.J. van Druten, M. Haque, New J. Phys. 15 (2013) 035008;
S. Balasubramanian, R. Ramaswamy, A.I. Nicolin, Rom. Rep. Phys. 65 (2013) 820;
L.E. Young-S, S.K. Adhikari, Phys. Rev. A 87 (2013) 013618;
H. Al-Jibbouri, I. Vidanović, A. Balaž, A. Pelster, J. Phys. B: At. Mol. Opt. Phys. 46 (2013) 065303;
X. Antoine, W. Bao, C. Besse, Comput. Phys. Commun. 184 (2013) 2621;
B. Nikolić, A. Balaž, A. Pelster, Phys. Rev. A 88 (2013) 013624;
H. Al-Jibbouri, A. Pelster, Phys. Rev. A 88 (2013) 033621;
S.K. Adhikari, Phys. Rev. A 88 (2013) 043603;
J.B. Sudharsan, R. Radha, P. Muruganandam, J. Phys. B: At. Mol. Opt. Phys. 46 (2013) 155302;
R.R. Sakhel, A.R. Sakhel, H.B. Ghassib, J. Low Temp. Phys. 173 (2013) 177;
E.J.M. Madarassy, V.T. Toth, Comput. Phys. Commun. 184 (2013) 1339;
R.K. Kumar, P. Muruganandam, B.A. Malomed, J. Phys. B: At. Mol. Opt. Phys. 46 (2013) 175302;
W. Bao, Q. Tang, Z. Xu, J. Comput. Phys. 235 (2013) 423;
A.I. Nicolin, Proc. Rom. Acad. Ser. A-Math. Phys. 14 (2013) 35;
R.M. Caplan, Comput. Phys. Commun. 184 (2013) 1250;
S.K. Adhikari, J. Phys. B: At. Mol. Opt. Phys. 46 (2013) 115301;
Ž. Marojević, E. Göklü, C. Lämmerzahl, Comput. Phys. Commun. 184 (2013) 1920;
L.E. Young-S, S.K. Adhikari, Phys. Rev. A 86 (2012) 063611;
S.K. Adhikari, J. Phys. B: At. Mol. Opt. Phys. 45 (2012) 235303.

[4] R. Kishor Kumar, L.E. Young-S, D. Vudragović, A. Balaž, P. Muruganandam, S.K. Adhikari, Comput. Phys. Commun. 195 (2015) 117.

[5] V. Lončar, A. Balaž, A. Bogojević, S. Skrbić, P. Muruganandam, S.K. Adhikari, Comput. Phys. Commun. 200 (2016) 406.

[6] B. Satarić, V. Slavnić, A. Belić, A. Balaž, P. Muruganandam, S.K. Adhikari, Comput. Phys. Commun. 200 (2016) 411.