JMATRIX – A program for scattering phase shifts calculations

P. Syty *, P. Horodecki and J.E. Sienkiewicz

Department of Theoretical and Mathematical Physics
Gdańsk University of Technology, Narutowicza 11/12, 80-952 Gdańsk, Poland

Abstract

We present a computer code, which is an implementation of the J-matrix method for a scattering problem on the potential vanishing faster than Coulomb one. Non-relativistic as well as relativistic cases are implemented. In this version of the program, scattering potential can be easily modelled as the square-well, as the truncated Coulomb potential or may be given by any analytical formula. Specific properties of the J-matrix method allows for calculatig phase shifts for many projectile energies with relatively small computational time.

Key words: atomic, phase shift, relativistic, Jacobi, J-matrix method
PACS: 34.80.-i, 11.80.m, 02.60.x

PROGRAM SUMMARY

Title of program: JMATRIX

Catalogue identifier:

Program obtainable from: CPC Program Library, Queen's University of Belfast, North Ireland, and home page of the first author (http://www.mif.pg.gda.pl/homepages/sylas/)

Licensing provisions: none

* Corresponding author
Email address: sylas@task.gda.pl (P. Syty).
URL: http://www.mif.pg.gda.pl/homepages/sylas/ (P. Syty).

Computer for which the program is designed and others on which it has been tested:

Computers: All machines with Fortran 90/95 compiler

Installations: Gdańsk University of Technology (IBM SP/2 and PC Pentium)

Operating systems under which the program has been tested: Linux, IBM AIX, MS Windows

Program language used: ANSI standard Fortran 90/95; a few numerical libraries written in Fortran 77 are also used

Memory required to execute with typical data: 1 – 30 MB (it is strongly dependent on the maximal basis size N used in calculations). Additionally, approximately 20MB of hard disk space is required for the output files.

No. of processors used: 1

Has the code been vectorised or parallelized? no

Distribution format: compressed tar file

No. of bytes in distributed program, including test data, etc.: approx. 200kB (uncompressed)

Nature of physical problem
Projectile of a given energy is elastically scattered on the radial potential vanishing faster than the Coulomb one. Our task is to obtain phase shift of the wavefunction of the scattered projectile. These phase shifts can be used in calculations of differential cross sections for elastic scattering, and spin polarization of the projectile.

Method of solution
Physical scattering problem is replaced by well-defined model which is solved exactly. Radial kinetic operator is tridiagonal in some suitable bases, such as Gaussian or Laguerre basis set. Scattering potential (vanishing faster than the Coulomb one) is truncated in N elements of the selected basis. Then, using some algebraic methods, one can find formula for tangent to the approximate phase shift, $\tan\delta_N$. We expect that for $N \to \infty$, this approximate value converges to the exact value, $\tan\delta$.

Restrictions on the complexity of the problem
In the presented code, the maximum basis size N is limited for these two reasons: (1) When basis size N approaches value about 500, the convergence process fails. It seems that this fact is a consequence of loss of preciseness in numerical integration. (2) For Laguerre basis, maximum value of N is 497,

due to limitations of the largest number representable in the double precision variables. For Gaussian basis, there is no such restriction. In future versions of the program these restrictions will be removed by using more precise and stable integration procedures and applying some numerical techniques.

Typical running time:
1 – 180 minutes, strongly dependent on basis size N and scheme used in computations (relativistic or non-relativistic).

Unusual features of the program:
At the end of its execution, program saves calculated elements of truncated potential to a file. This allows for repetition of calculations for different projectile energies with a relatively small computational time. It is possible to create data base of these files for a different parameters for a future use.

LONG WRITE-UP

1 Introduction

The J-matrix method is an algebraic method in quantum scattering theory. It is based on fact that the radial kinetic energy operator is tridiagonal in some suitable bases. Non-relativistic version of the method was introduced in 1974 by Heller and Yamani [1], [2] and developed by Yamani and Fishman [3] a year after. Recently, relativistic version was introduced by Horodecki [4] and extended by Alhaidari *et al* [5]. Theoretical basis of the method is described in section 2.1. Presented program JMATRIX implements both non-relativistic and relativistic versions of the method. In general, program allows for calculations of scattering phase shifts in all cases when the scattering potential is given through the analitycal formulas. However, the main task of the present work was to perform some calculations for a relatively simple cases, for the purpose of testing the relativistic version of the method, as it has never been (numerically) tested before. For a start, we selected the scattering potential modelled as square-well and a truncated Coulomb potential. Performed test calculations of scattering phase shifts show, that: (i) numerical phase shifts converge to results obtained using an analytical formula, as we increase size of basis used to truncate scattering potential, (ii) non-relativistic limit in relativistic computations is correctly satisfied.

2 Theortical method

2.1 The J-matrix method

In this section we give only a short review of the J-matrix theory of scattering, but it should be sufficient for understanding the main idea of the method. Detailed description of the method can be found in publications [1] – [6].

Our task is to find an approximate solution of the scattering problem on the radial potential $V = V(r)$ vanishing faster than the Coulomb one. Let us replace this scattering potential by a truncated potential operator:

$$V^N = P_N^\dagger V P_N \tag{1}$$

with the generalized projection operator

$$P_N = \sum_{n=0}^{N-1} \left|\phi_n^l\right\rangle \left\langle\phi_n^l\right|. \tag{2}$$

Then, using expansion of the solution of the new problem in the basis $\{\phi_n^l\}$, one can find that tangent of approximated phase shift is given by the formula

$$\tan\delta_N = -\frac{s_{N-1}^l(k) + g_{N-1,N-1}(\mathcal{E}) J_{N,N-1}(k) s_N^l(k)}{c_{N-1}^l(k) + g_{N-1,N-1}(\mathcal{E}) J_{N,N-1}(k) c_N^l(k)}, \tag{3}$$

where s_n^l and c_n^l are coefficients of sine-like and cosine-like solutions of the following equation

$$\left(H_0 - \frac{k^2}{2}\right) \sum_{n=0}^{\infty} u_n^l \phi_n^l(\lambda r) = \Omega_u \bar{\phi}_n^l(\lambda r);\ u = s, c;\quad \Omega_s = 0;\quad \Omega_c = -\frac{k}{2s_0^l}. \tag{4}$$

Here, $k \equiv \sqrt{\frac{2M\mathcal{E}}{\hbar^2}}$ is the wave number related to the energy $\mathcal{E}$ and mass M of the projectile. Basis set $\{\bar{\phi}_n^l\}$ is biorthonormal to set $\{\phi_n^l\}$ with respect to unitary scalar product, i.e. $\left\langle\bar{\phi}_m^l|\phi_n^l\right\rangle = \delta_{mn}$, where δ_{mn} is, as usual, Kronecker delta.

$J_{N,N-1}$ is an element of the following matrix

$$J_{mn} \equiv \left\langle\phi_m^l\right| H_0 - \frac{k^2}{2} \left|\phi_n^l\right\rangle \equiv \left\langle\phi_m^l\right| - \frac{1}{2}\frac{\mathrm{d}^2}{\mathrm{d}r^2} + \frac{l(l+1)}{2r^2} - \frac{k^2}{2} \left|\phi_n^l\right\rangle. \tag{5}$$

Table 1
Elements of Laguerre and Gaussian basis sets and elements of expansion of sine-like and cosine-like solutions in these bases.

	Laguerre set	Gaussian set
ϕ_n^l	$(\lambda r)^{l+1}\exp\left(-\frac{\lambda r}{2}\right)L_n^{(2l+1)}(\lambda r)$	$(\lambda r)^{l+1}\exp\left(-\frac{\lambda^2 r^2}{2}\right)L_n^{(l+\frac{1}{2})}(\lambda^2 r^2)$
$\bar{\phi}_n^l$	$\frac{n!}{\lambda^2\Gamma(n+2l+2)}\frac{1}{r}\phi_n^l(\lambda r)$	$\frac{2n!}{\lambda^2\Gamma(n+l+\frac{3}{2})}\phi_n^l(\lambda r)$
s_n^l	$\frac{2^l\Gamma(l+1)n!(\sin\theta)^{l+1}}{\Gamma(n+2l+2)}C_n^{(l+1)}(\cos\theta)$	$\frac{\sqrt{2\pi}n!(-1)^n}{\Gamma(n+l+\frac{3}{2})}\exp\left(-\frac{\eta^2}{2}\right)\eta^{l+1}L_n^{(l+\frac{1}{2})}(\eta^2)$
c_n^l	$\frac{-2^l\Gamma(l+\frac{1}{2})n!}{\sqrt{\pi}\Gamma(n+2l+2)(\sin\theta)^l}$ $\times{}_2F_1\left(-n-2l-1,n+1,\frac{1}{2}-l;\sin^2\left(\frac{\theta}{2}\right)\right)$ $\sin\theta\equiv\frac{k\lambda^{-1}}{k^2\lambda^{-2}+\frac{1}{4}};\ \cos\theta\equiv\frac{k^2\lambda^{-2}-\frac{1}{4}}{k^2\lambda^{-2}+\frac{1}{4}}$	$\frac{\sqrt{\frac{2}{\pi}}\Gamma\left(l+\frac{1}{2}\right)(-1)^n n!}{\Gamma(n+l+\frac{3}{2})}\exp\left(-\frac{\eta^2}{2}\right)\eta^{-l}$ $\times{}_1F_1\left(-n-l-\frac{1}{2},\frac{1}{2}-l;\eta^2\right)$ $\eta\equiv\frac{k}{\lambda}$

In some suitable bases, such as Gaussian or Laguerre set, the above matrix is tridiagonal (and is called Jacobi or J-matrix). This enables us to find coefficients s_n^l and c_n^l, using three-term recursion relation between them and the J-matrix (see [2] for details). The explicit forms of these coefficients as well as elements of basis sets are collected in Table 1. In the table, $L_n^{(\alpha)}$ and $C_n^{(\alpha)}$ are Laguerre and Gegenbauer polynomials, respectively; ${}_2F_1$ and ${}_1F_1$ are hypergeometric functions, $\lambda > 0$ is a scaling parameter ($\lambda \neq 0.5$).

In above formulas, N is the quantity of base functions ϕ_n^l used to truncate scattering potential, $g_{N-1,N-1}(\mathcal{E})$ is a matrix element of the inverse of the truncated operator $P_N^\dagger\left(H_0+V^N-\frac{k^2}{2}\right)P_N$, restricted to the N-dimensional space, where it doesn't vanish. In short, this matrix can be viewed as the matrix approximating the Green function.

For $N\to\infty$, what is connected with reduction of inaccuracy in approximating of the scattering potential, $\tan\delta_N$ should converge to the exact value $\tan\delta$, and, simultaneously, approximate δ_N should approach the exact scattering phase, δ.

In the relativistic case we have very similar formula for tangent of the approximated phase shift:

$$\tan\tilde{\delta}_N=-\frac{s_{N-1}^l(\tilde{k})+\frac{2\epsilon}{\tilde{k}}\mathcal{G}_{N-1,N-1}^{++}(E)J_{N,N-1}(\tilde{k})s_N^l(\tilde{k})}{c_{N-1}^l(\tilde{k})+\frac{2\epsilon}{\tilde{k}}\mathcal{G}_{N-1,N-1}^{++}(E)J_{N,N-1}(\tilde{k})c_N^l(\tilde{k})}. \tag{6}$$

Here, we have the same coefficients of the expansion and J-matrix element as in the non-relativistic case, only taken with the relativistic number $\tilde{k}\equiv\frac{\sqrt{(E-Mc^2)(E+Mc^2)}}{c\hbar}$, related to the total energy $E=\mathcal{E}+Mc^2$. See [4] for detailed explanation of the symbol $\mathcal{G}_{N-1,N-1}^{++}$. As in the non-relativistic case, it can be viewed as a matrix element of the inverse of some truncated operator,

but here restricted not to N (as it is in the non-relativistic case), but to the $2N$-dimensional space. To complete definitions, $\epsilon \equiv \sqrt{\frac{E-Mc^2}{E+Mc^2}}$.

2.2 Test case – square well

Here we shall consider spherically symmetric potential $V(r)$ defined by the square-well with respect to the radial coordinate:

$$V(r) = \begin{cases} 0 & \text{for } r \in (0, a) \\ V_0 & \text{for } r \in [a, b) \\ 0 & \text{for } r \in [b, \infty) \end{cases} .$$

and is defined by three parameters: (i) depth V_0, (ii) left bound a, (iii) right bound b.

We will consider only the relativistic case, in which the analytical formula for tangent of the phase shift can be simply found to be

$$\tan\tilde{\delta} = \frac{\tilde{B}}{\tilde{A}} \tag{7}$$

where the numbers A, B (depending on the energy of the projectile, the relativistic number κ, and parameters of the potential) can be constructed as coordinates of the following vector

$$\begin{bmatrix} \tilde{A} \\ \tilde{B} \end{bmatrix} = N(b)^{\tilde{k},\kappa} M(b)^{\tilde{k}',\kappa} N(a)^{\tilde{k}',\kappa} M(a)^{\tilde{k},\kappa} \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \tag{8}$$

where the 2×2 matrices M and N, depending on the position, are defined with aid of Ricatti-Bessel and Ricatti-Neumann functions $j_l(r)$, $n_l(r)$ as follows

$$M(r)^{\tilde{k},\kappa} = \begin{bmatrix} j_l(kr) & -n_l(kr) \\ \mp\epsilon(\tilde{k}) j_{l\pm1}(\tilde{k}r) & \pm\epsilon(\tilde{k}) n_{l\pm1}(\tilde{k}r) \end{bmatrix}, \tag{9}$$

$$N(r)^{\tilde{k},\kappa} = \begin{bmatrix} \pm\epsilon(\tilde{k}) n_{l\pm1}(\tilde{k}r) & n_l(\tilde{k}r) \\ \mp\epsilon(\tilde{k}) j_{l\pm1}(\tilde{k}r) & j_l(\tilde{k}r) \end{bmatrix}, \tag{10}$$

with the relativistic quantum number $\kappa = l$ ($\kappa = -l - 1$) for upper (lower) sign of indices in the above formula. Here, $\tilde{k}'$ is defined in the same way as $\tilde{k}$ (defined in previous section), but with shifted energy $E' = \mathcal{E} + V_0 + Mc^2$ instead of E. The number ϵ in the above matrices is defined in the same way as in the previous section.

3 Numerical computations

3.1 A general view of the program

We expect that for $N \to \infty$, approximate phase shift δ_N (or $\tilde{\delta}_N$ in the relativistic case) converges to the exact value δ (or $\tilde{\delta}$). Using presented program, one is able to study this convergence by systematic calculations of phase shifts for progressively increased basis size N. Moreover, in the case of square-well potential, numerical results obtained in relativistic calculations can be compared with result obtained using analytical formula (7).

The program computes all required mathematical functions (such as Gegenbauer and Laguerre polynomials, hypergeometric functions, Bessel, spherical Bessel, Neumann and spherical Neumann functions and their derivatives, gamma function and more) to evaluate basis functions ϕ_n^l and coefficients s_n^l and c_n^l (see Table 1). Then the program truncates the scattering potential in selected basis by numerical integration, and forms the matrix approximating Green function. This matrix is constructed as sum of the matrix of elements of truncated potential and the elements of the J-matrix. Then program inverses this matrix approximating Green function by diagonalisation (using some orthogonal matrix; in the present code this orthogonal matrix is a matrix of eigenvectors of the Green matrix) and finally, computes approximate phase shift for given number N. Also, there has been written an additional procedure to calculate phase shifts using an analytical formula for potentials with a shape of potential square-well, using relativistic formulas from section 2.2

Greater part of the mathematical procedures and functions used in program have been written by authors using useful formulas and relations included in [7] and [8]. Some procedures (for Bessel, Neumann and Gamma functions, for numerical integration and for searching eigenvalues and eigenvectors of real, symmetric matrix) have been taken from public Fortran 77 libraries.

In the relativistic case, most of the formulas are written as function of the total energy of the projectile, E. For user convenience and for unification with the non-relativistic case, we used a rescaled kinetic energy $\mathcal{E}$ in our code, instead the total energy E. Due to this rescaling, implemented formulas are slightly

different than written in section 2.1. In the non-relativistic case such rescaling was not necessary, because in this case all formulas are written as function of the kinetic energy from the beginning.

Coefficients s_n^l and c_n^l for $N = 1$ and $N = 2$ have been calculated using explicit formulas (see Table 1), but for $N > 2$, to avoid numerical instabilities, using three-term recursion relation (see [2]). In these formulas, factorials and gamma functions for a big arguments appear. To minimize the numerical errors, these functions have been replaced by exponents of their logarithms. This move was possible only in the case of Gaussian basis; in the case of Laguerre basis, explicit formulas have been used in calculations, so the maximum number of N is limited to 497 in this case.

In the J-matrix method there are many integrals to be calculated, especially in the relativistic case. As they are used to truncate the scattering potential (1), their quantity is $N \times N$ in the non-relativistic case, and $2N \times 2N$ in the relativistic one. Calculation of these integrals has the major contribution in the computational time of the present code. To minimize their quantity (and therefore overall time of computations), we applied a few tricks, such as utilization of the symmetry of the J-matrix and properties of the scattering potential. Moreover, integrals calculated in each iteration are saved to be used in the next iterations. The current integration method is the Double Exponential (DE) Transformation method.

Once the phase shift for given energy and requested values of N have been calculated, program starts calculations of phase shifts for another projectile energies (defined in the input file – see section 3.3), with using previously calculated elements of the truncated potential, as they do not depend on energy of the projectile. Such approach allows for saving great amount of time, because – as it has been noticed before – calculations of the elements of the trunated potential are the most time-consuming operations in the present code. Additionally, these elements are saved to the external file on the hard disk, so it is possible to use them in next runs of the program, i.e. for calculating phase shifts for yet another energies of the projectile.

3.2 Structure of the program and compilation

During unpacking provided gzipped tar file, directory *jmatrix* is created, and two directiories within it: *source*, in which all source files are stored, and *bin*, into which the executable file, created during compilation, will be moved. For a clarity, source code has been splitted into several files. In each file there is a part of the code responsible for a specific task, i.e. procedures included in file *jm_specfun.f90* calculates the special functions used in calculations, *jm_v.f90*

truncates the scattering potential, *jm_analytic.f90* calculates the analytical value of phase shift in case of potential square-well, etc. The main procedure is included in the file *jmatrix.f90*. There are two special modules in the program: *jm_constants.f90* and *jm_global.f90*. In the first one there are defined constants and types used in program, the second one is the main module which controls calculations and in which the global variables are defined. Also, short *readme* file is provided.

Program is written in ANSI Fortran 90/95 and should be compiled with any Fortran 90/95 compiler without any problems. Additionally, program uses some numerical libraries written in Fortran 77, they also should be succesfully compiled with use of the same F90/95 compiler. In some rare cases it is convenient to use separate F90/95 and F77 compilers, i.e. when using VAST/f90 compiler. To make compilation as easy as possible, it is controlled through the proper *makefile*. Before compilation, user should modify this makefile by specifying the executable names of the F77 and F90/F95 compilers, and flags passed to the compilers, then run the command *make* from within the directory *source*. The executable file *jm* will be created and moved to directory *../bin*. For user's convenience, we prepared and included a set of makefiles for various compilers: Portland Group PGI, VAST/f90, Intel Fortran Compiler and Digital Visual Fortran. These makefiles are also stored in the directory *source*.

3.3 Input and output files

All calculations are controlled through the parameters read from the external file. The name of the input file is of one's choice, program asks for its name at start of execution, but the default is *jm.inp*. The input file should be composed of a set of lines in the following form:

; comment
keyword = value ;comment

Blank lines, lines beginning with ";" and lines without proper keyword are ignored. Keywords and values are case sensitive, there is a free choice of their order in a file. Not all keywords are mandatory, what will be discussed later. There *have to* be spaces around the "=" sign and before the ";" sign (except the case when whole line is a comment). Below is an example of the input file, in which all of the keywords recognized by the program are included. The senses of the particular parameters are discussed later, but they can be easily presumed from their names and included comments.

```
; PROPERTIES OF THE PROJECTILE
estart   = 3.0          ; Projectile energy range
eend     = 4.0
```

```
estep    = 0.1          ; Step size
l        = 1            ; Quantum numbers
kappa    = 1

; TYPE OF CALCULATIONS
lambda   = 1.0          ; Scaling parameter
basis    = laguerre     ; Basis set (gauss, laguerre)
scheme   = relativistic ; Scheme (relativistic, non-relativistic)
pot_type = well         ; Potential type (well, coulomb, other)
v_light  = finite       ; Velocity of light (finite, infinite)
nstart = 1              ; Initial value of N
nend   = 400            ; Final value of N

; OTHER PARAMETERS
screen = .T.            ; Display results on screen
shift  = .F.            ; Shift results to range [0,pi]

; PARAMETERS OF POTENTIAL SQUARE-WELL
v0 = -1.0               ; Depth
a  = 0.8                ; Left bound
b  = 1.0                ; Right bound

; PARAMETERS OF TRUNCATED COULOMB POTENTIAL V(r) = - z / (r^alpha)
r0    = 1.0             ; Truncating parameter
z     = 30.0            ; The z parameter
alpha = 1.0             ; The alpha parameter
```

Program calculates phase shifts for projectile energies from range [**estart**, **eend**] with step size **estep** in a single run of the program. Energies are given in atomic units. Parameters **estart** and **eend** are mandatory; **estart** should be less or equal than **eend** (in the latter case phase shift will be calculated for only one energy of the projectile). When no **estep** is specified, program takes **estep** = (**eend** − **estart**)/10. The mass of the projectile is assumed to be unitary in the present code (what conforms i.e. to electrons), but it can be easily changed in the source file *jm_global.f90*. Option which will allow for change mass of the projectile will be added to the future versions of the program.

Next, orbital quantum number l and quantum number κ, which describe the projectile have to be specified, while $\kappa = l$ or $\kappa = -l - 1$. To specify these numbers, keywords **l** and **kappa**, respectively, should be used. If no one from the above dependencies between κ and l is not satisfied, program stops with a proper message. This restriction does matter only in relativistic calculations, because κ is not used in the non-relativistic scheme, so the keyword **kappa** is ignored in that case.

One can specify scaling parameter **lambda**. If omitted, the standard value 1.0 will be taken. Changing this parameter should improve the convegence in some particular cases, but it is recommended to leave this parameter untouched as its influence has not been investigated in details yet. Moreover, due to some numerical reasons, it is not allowed to put the value 0.5 for **lambda**.

J-matrix (5) can be calculated in two different basis sets, Laguerre or Gauss. This is detemined through keyword **basis**. It can assume two values, **laguerre**

or **gauss**. Also, **scheme** of computations (**relativistic** or **non-relativistic**) should be specified. If no keyword **scheme** is found in the input file, calculations will be performed using the non-relativistic scheme.

Keyword **pot_type** is responsible for kind of scattering potential used in computations. There are two types of potentials pre-defined in program: square-well (value: **well**) and truncated Coulomb potential (value: **coulomb**). The parameters of the above potentials should also be specified in input file. If square-well is selected, the depth **v0**, left and right bounds (**a** and **b**, respectively) are required (all of them in atomic units). The depth of square-well is normally a negative number. It is allowed to put the positive number for **v0**, but in that case it should not be greater than the energy of the projectile. The Coluomb potential has the form $V(r) = -z/r^{\alpha}$, so the parameters **z** and **alpha** should be specified. Parameter **r0** (in atomic units) specifies at which r potential will be truncated, so that $V(r > \mathbf{r0}) = 0$.

Moreover, program is able to use whichever scattering potential given in any analytical form, by using the value **other** for keyword **pot_type** in the input file. Requested formula describing the potential should be specified in the adequate place in the module *jm_v.f90*, then the program have to be recompiled.

To verify that the relativistic results converge to the non-relativistic limit as the speed of light approaches infinity, we introduced additional keyword, **v_light**. Standard value is **finite**, it is responsible for the constant and finite speed of light (137.036 in atomic units). By specifying **v_light** = **infinite**, one can get non-relativistic limit in relativistic computations and compare it with pure non-relativistic calculations. This setting should not be used in real calculations, it has been added only for testing purposes. Keyword **v_light** is ignored in non-relativistic scheme of calculations.

Next two parameters, **screen** and **shift** are logical-type and are responsible for displaying (default) or not results on screen and for shifting (or not, which is default) calculated phase shifts to range $[0, \pi]$. At last, the initial (**nstart**) and final (**nend**) values of N should be specified, so calculations will be performed for N form range [**nstart**, **nend**] with step 1. If **nstart** is omitted, it is taken as 1. When **nstart** = **nend**, calculations will be performed for only this one value of N. It does not allow for studying the convergence, but it may be useful in case when one need only to calculate phase shifts for different energies with use of previously saved elements of truncated potential.

Program creates three output files which contain results of calculations. In the first file (program asks for its name; the default name is *jm.out*), calculated phase shifts as function of the basis size are saved. In the second file, calculated phase shifts as function of energy of the projectile (for the final value of N) are saved. The name of this file is automatically choosen to be the same as the first

file, but with suffix *.en*. Additionally, the file *saved.vn* is created. It contains the elements of the truncated potential and can be utilized, to save computational time, in separate runs of the program. Program will utilize this file when saved informations will correspond to the actual set of parameters, i.e. calculations are performed in the same basis, scheme, with the same quantum numbers describing the projectile and scaling paremeter, and, of course, for the same basis size N (so adequate and the same values for **nstart** and **nend** should be put in the input file). When these informations do not conform to current calculations or the **nstart** is not equal to **nend**, the file *saved.vn* will be deleted and created for the new set of parameters.

Below is an (shortened) example of the first output file, with phase shifts as functions of N:

```
; OUTPUT FROM THE JMATRIX PROGRAM
(...)
; ANALYTICAL RESULT
; delta =    0.158163581281253
;
; NUMERICAL RESULTS:  N   tan(delta)   delta
  1    0.0001153729311281    0.0001153729306162
  2    0.0016942699360635    0.0016942683149037
  3    0.0084487948013789    0.0084485937789887
(...)
 13    0.1377207522226093    0.1368598121024684
 14    0.1547320896298930    0.1535146646821944
 15    0.1751206279063803    0.1733627075052276
(...)
203    0.1623105129055408    0.1609072868221379
204    0.1640488400677350    0.1626005266770787
205    0.1650570056215622    0.1635821130890733
(...)
300    0.1621137854778999    0.1607156031262297
301    0.1615424542908467    0.1601588523320820
302    0.1605951402488267    0.1592354932033918
(...)
398    0.1595807653598821    0.1582464653476224
399    0.1597952178393117    0.1584555852164212
400    0.1599136302723107    0.1585710471988183
```

Below is an (shortened) example of the second output file, with phase shifts as functions of energy of the projectile:

```
; OUTPUT FROM THE JMATRIX PROGRAM
(...)
; NUMERICAL RESULTS: energy  tan(delta)  delta
  3.000000    0.1599136771879867    0.1585710929446643
  3.100000    0.1596260888605002    0.1582906629904541
  3.200000    0.1611806782827603    0.1598062570466153
  3.300000    0.1607058305437033    0.1593433986129410
  3.400000    0.1603631246415301    0.1590093028032292
  3.500000    0.1611399619369947    0.1597665714411470
  3.600000    0.1597165659930660    0.1583788907618623
  3.700000    0.1586909568144603    0.1573786340075637
  3.800000    0.1588798233241868    0.1575628557703182
  3.900000    0.1573651299127394    0.1560851102673489
  4.000000    0.1551291838287373    0.1539024506549712
```

3.4 Test case – Results and discussion

We performed some test calculations of phase shifts in scattering on the square-well potential, using the relativistic scheme of computations, in both bases, Laguerre and Gauss. We present a few figures ilustrating convergence of the calculated phase shift to the value obtained using analytical formulas. In Figure 1 results obtained using Laguerre basis functions are presented. As we can see, the convergence in this basis is rather slow but systematic. Results obtained using Gaussian basis functions are presented in Figure 2. In this case we have much faster convergence. One can notice that the convergence in Laguerre basis set has completely different nature if compared to convergence in Gaussian set. The convergence in Laguerre set appears to be more stable and regular, but is slower. In Gaussian basis, we have rather a quick convergence, but the numerical results "jump" around the analytical result.

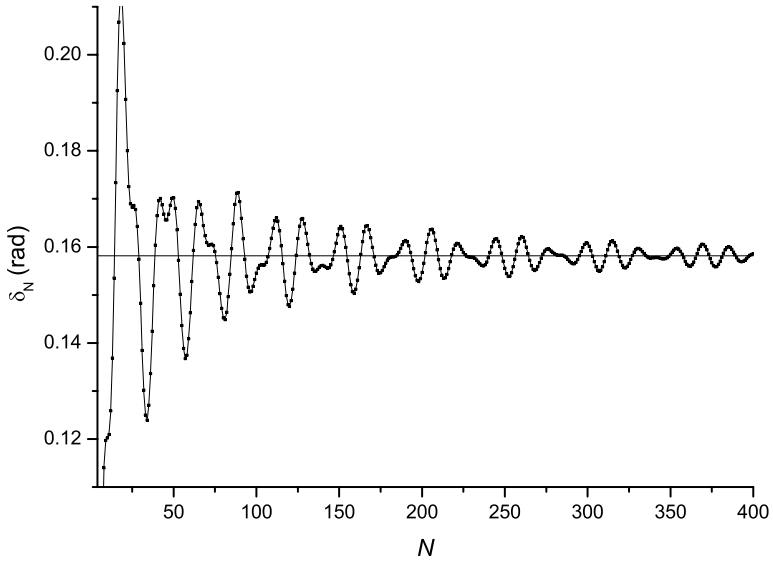

Fig. 1. Convergence of the phase shift versus number of Laguerre basis function N used to truncate the scattering potential. Straight line – analytical result.

In general, in both investigated cases it is not difficult to see that phase shifts computed numerically converge to phase shift obtained using an analytical formula. This is clearly shown in Figure 3, where root-mean-square deviations of numerical values from the analytical one are presented.

Now let us see that the non-relativistic limit in relativistic calculations is properly achieved. It is illustrated in Table 2. One can see, that relativistic calculations performed with substituded infinite speed of light instead of finite, give result very close to that one obtained from non-relativistic calculations.

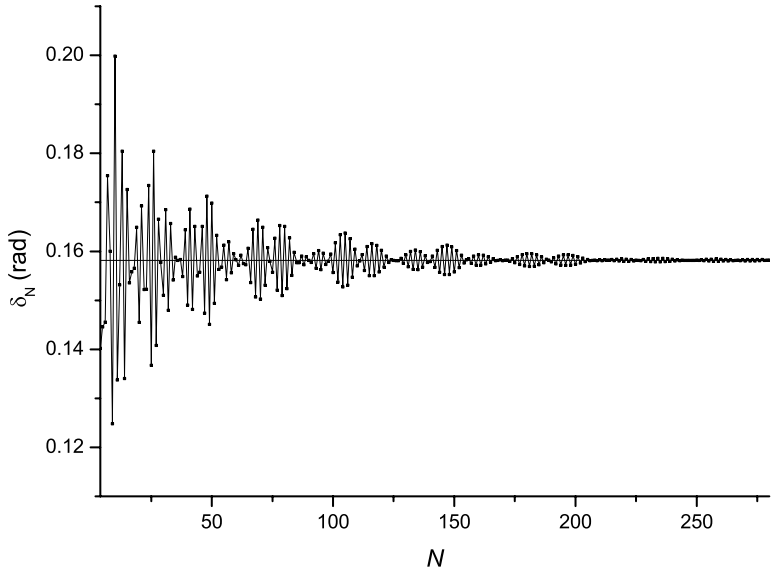

Fig. 2. Convergence of the phase shift versus number of Gaussian basis function N used to truncate the scattering potential. Straight line – analytical result.

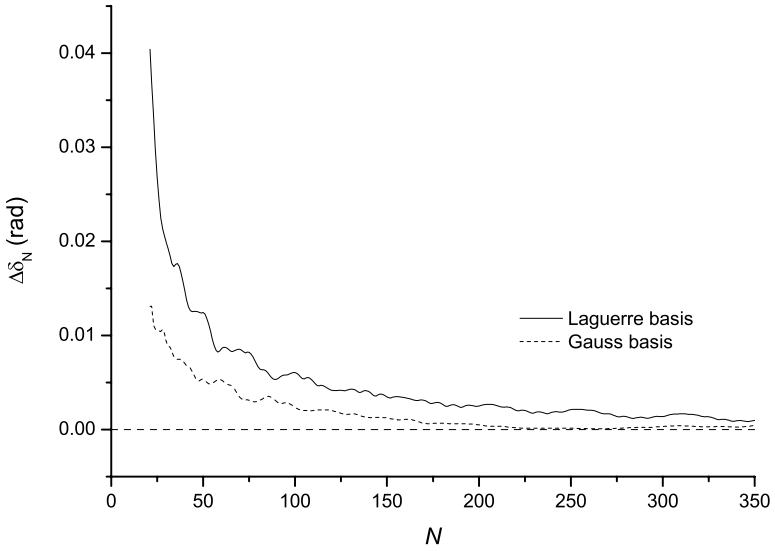

Fig. 3. Root-mean-square error, averaging 20 points backwards and 20 forwards.

4 Conclusions

In last few years the interest of the J-matrix method is significantly increased, mainly due to relativistic extension of the method. We proposed a program

Table 2
Illustration of the non-relativistic limit in relativistic calculations.

scheme	speed of light	$\delta_{N=100}$
relativistic	finite	0.1545573335662396
relativistic	infinite	0.1545390923421738
non-relativistic	—	0.1545390774387093

for scattering phase shifts calculations using the J-matrix method, both relativistic as well as non-relativstic versions. It allows for applying any scattering potential vanishing faster than the Coulomb one and given in analytical form. An example of scattering on the potential in shape of square-well has been presented. In this case, results of numerical computations agree with analytical results. Moreover, non-relativistic limit is properly achieved.

Acknowledgements

Authors thanks R. Szmytkowski for valuable suggestions concerning some numerical issues. All computations were performed in the Tri-City Academic Computer Centre in Gdańsk TASK.

References

[1] E. Heller, H. Yamani, New L2 approach to quantum scattering: Theory, Phys. Rev. A **9** (1974) 1201.

[2] E. Heller, H. Yamani, J-matrix method: Application to s-wave electron-hydrogen scattering, Phys. Rev. A **9** (1974) 1209.

[3] H. Yamani, L. Fishman, J-matrix method: Extensions to arbitrary angular momentum and to Coulomb scattering, J. Math. Phys. **16** (1975) 410.

[4] P. Horodecki, The relativistic J-matrix method, Phys. Rev. A **62** (2000) 052716.

[5] D. Alhaidari, H.A. Yamani, and M.S. Abdelmonem, Phys. Rev. A **63**, (2001) 062708.

[6] H.A. Yamani, A.D. Alhaidari, M.S. Abdelmonem, Phys. Rev. A **64**, (2001) 042703.

[7] H. Bateman, A. Erdély (ed.), *Higher Transcendental Functions*, vol. I, II, (McGraw-Hill, New York, 1953).

[8] G. Arfken, *Mathematical Methods For Physicists*, (Academic Press, New York, 1970).