User’s guide

T-matrix program based on the null-field method with discrete sources (NFM-DS)

Update on 6th March 2009.

Vladimir Schmidt[1], Adrian Doicu[2], Thomas Wriedt[3]

[1] Universität Bremen, Germany, e-mail: vschmidt@iwt.uni-bremen.de
[2] Institute für Methodik der Fernerkundung, Germany, e-mail: adrian.doicu@dlr.de
[3] Universität Bremen, Germany, e-mail: thw@iwt.uni-bremen.de

Abstract

This is an update of the FORTRAN programs based on the Null-Field Method with Discrete Source which originally was included on CD-ROM with the book by A. Doicu, T. Wriedt, Y.A. Eremin *Light scattering by systems of particles*, Springer, Berlin, 2006.

1. Introduction

The T-matrix method or null-field method is one of the most popular light scattering theories to accurately compute scattering by nonspherical particles [1]. Recent reviews of the literature on this method have been published by Mishchenko et al. [2]. It is based on the expansion of incident, internal and scattered field in the terms of suitable basis of vector wave functions, which are classically spherical vector wave functions. The T-matrix relates the expansion coefficients of the incident field to the expansion coefficients of the scattered field. Using a computed T-matrix various light scattering problems (multiple scattering, scattering by a particle located near a plane interface, scattering by a rotated or a translated particle or orientation averaged scattering) can easily be calculated.

The NFM-DS combines approaches of the Generalized Multipole Techniques (GMT) [3] with the T-Matrix method. For expansion of the internal field within a scatterer discrete sources are used. This helps to compute the T-Matrix of particles having a high aspect ratio (up to 100:1) such as fibres and flat discs. The full scope of the NFM-DS has recently been presented in a review paper [4]. For a full description of the theory we refer the interested reader to the published monograph by Doicu et al. [5], which includes the original FORTRAN90 program code on CD-ROM.

The theory and corresponding program has been developed at Bremen University over the last 10 years. It is intended for science professionals, engineers and graduate students working in optics, electromagnetics, biomedical optics, atmospheric radiation and remote sensing. The program can be used to compute the scattering and absorption of electromagnetic waves by particles with arbitrary geometries using the NFM-DS. Various scattering problems are implemented in the program

single particle (homogeneous)
• homogeneous, dielectric (isotropic, chiral) and perfectly conducting, axisymmetric particles of arbitrary shape incl. having a high aspect ratio and concavities;
• homogeneous, dielectric (isotropic, uniaxial anisotropic, chiral) and perfectly conducting, nonaxisymmetric particles of arbitrary shape;

single particle (inhomogeneous)
• axisymmetric, composite particles of arbitrary shapes;
• axisymmetric, layered particles of arbitrary shapes;
• inhomogeneous, dielectric, axisymmetric particles with an arbitrarily shaped inclusion;
• inhomogeneous, dielectric spheres with a spherical inclusion;
• inhomogeneous, dielectric spheres with an arbitrarily shaped inclusion;
• inhomogeneous, dielectric spheres with multiple spherical inclusions;
• concentrically layered spheres;

multiple particles
• clusters of arbitrarily shaped particles;
• two homogeneous, dielectric spheres;
• clusters of homogeneous, dielectric spheres;

particle on or near a plane
• homogeneous, dielectric or perfectly conducting, axisymmetric particles on or near a plane surface.

Using the computed T-matrix the scattering characteristics describing the scattered field in the far-field region for the considered scatterer in a fixed or averaged orientation can be easily computed. These include the far-field pattern, the differential scattering cross section (DSCS), the amplitude matrix, the optical cross sections and the phase and extinction matrices. To ensure convergence of the results corresponding routines are integrated into the software. Additionally the effective wave number of a medium with randomly distributed spheroidal particles can be calculated.

2. Update to the original code

The original program code can be found on CD-ROM enclosed in the monograph by Doicu et al [5]. Here we would like to familiarize the reader with improvements implemented in the NFM-DS program code since the book was published (version 1.1).

There are three important changes in the program. Firstly, the formulae of the T-matrix for an uniaxial anisotropic, nonaxisymmetric particle of arbitrary shape were obtained and implemented by A. Doicu [6]. Secondly, the program was parallelized for the usage on multi-core or multi-processors computers. Thirdly, some improvements in usage of the program for multiple simulations were realized.

Here we would like to present detailed information about these additional features.

3. Light scattering by uniaxialy anisotropic particles

3.1. T-matrix method for uniaxialy anisotropic particles

The T-matrix method is based on the expansion of incident, internal and scattered field in the terms of a suitable basis of vector wave functions. For an isotropic medium the spherical vector wave functions (SVWF) are used classically. A. Doicu obtained the basis of the so called quasi spherical vector wave functions (qSVWF) for an uniaxial anisotropic medium by solving Maxwell's equations in Fourier's space [6]. Using this basis the scattering problem for an uniaxial anisotropic, non-axisymmetric particle using the T-matrix method is solved. We refer the interested reader to the paper [6] for more detailed theoretical derivation.

3.2. Description of program code

The main program "TMATRIX.f90" calls a T-matrix routine for solving a specific scattering problem. For an uniaxial anisotropic nonaxisymmetric particle the code is included in the file "TNONAXSYM.f90", which was used originally only for isotropic dielectric, chiral or perfect conducting particles.

3.3. Input parameters of the scattering problem

For the scattering problem by an uniaxial anisotropic particle the input parameters are specified in three input files:

• "/INPUTFILES/InputNONAXSYM.dat" provides the variables specifying the optical properties, geometry and error tolerances,
• "/INPUTFILES/InputSCT.dat" provides the variables specifying the scattering characteristics calculation,

- “/INPUTFILES/Input.dat” specifies the model control parameters.

The input parameters are divided between several groups, each specified by a keyword that is recognized by the program. A detailed description of the parameters required by the input files can be found on the CD enclosed in the book [5] or in the comment lines of each T-matrix routine. Here we notice some changes in the structure of “InputNONAXSYM.dat” relative to the original version (only isotropic dielectric, chiral or perfect conducting particles). Five new parameters were added into the file “InputNONAXSYM.dat”, see Table 1.

Table 1. The new input parameters for uniaxial anisotropic particles.

Group name	**Parameter name**	**Examples for the value**	**Comment**
MatProp	anisotropic	.true. .false.	If true, then the particle is an uniaxial anisotropic crystal
AnSVWF	ind_refRelZ *	(1.7,0.0)	The second relative refractive index of the uniaxial anisotropic particle
AnSVWF	alphaPR *	120	Azimuthal angle specifying the orientation of the principal coordinate system with respect to the particle coordinate system
AnSVWF	betaPR *	90	Zenith angle specifying the orientation of the principal coordinate system with respect to the particle coordinate system
AnSVWF	Nbeta *	181	Number of integration points for computing the vector quasispherical wave functions

* these variables must be provided if anisotropic = .true.

The angles *alphaPR* and *betaPR* identify the orientation of the optical axis of a crystal while the parameter *ind_refRelZ* defines the refractive index along the Z-axis. If *alphaPR*=0 and *betaPR*=0, the refractive index of a particle would have the components *(ind_refRel, ind_refRel, ind_refRelZ)* where *ind_refRel* is the parameter from the *MatProp* group (see [5]). The parameter *Nbeta* defines the number of points in a gauss integration rule for computing the vector quasispherical wave functions (see [6]). We don’t advise to change its default value for non-extreme refractive indices, because a value of *Nbeta*=181 already gives the exact integration for polynomials of the degree up to 363.

3.4. Summary

To compute the electromagnetic scattering by an uniaxial anisotropic particle with the TMATRIX program the user has to:

- specify the optical properties, geometry and error tolerances in the “/INPUTFILES/InputNONAXSYM.dat”,
- provide the scattering characteristics in the “/INPUTFILES/InputSCT.dat”;
- set the model control parameters in the input file “/INPUTFILES/Input.dat”;
- run the main program “TMATRIX.f90” and call the routine “TNONAXSYM.f90”;
- perform a convergence test and analyse the results written to the file “/OUTPUTFILES/Output.dat”; if convergence is achieved the program will write the T-matrix to the file FileTmat in the directory TMATFILES, and the differential scattering cross sections and the scattering characteristics to the files FileDSCS and FileScat in the directory OUTPUTFILES;
- to change the scattering characteristics calculation, modify the input file “/INPUTFILES/ InputSCT.dat” and run the main program “TMATRIX.f90” by calling the routine “SCT.f90”.

An exemplary calculation of the normalized differential scattering cross section for an uniaxial anisotropic ellipsoid is presented in Fig. 1.

4. Parallelization using OpenMP

The T-matrix method is one of the most effective methods for accurate solving the light scattering problem. But the numerical scheme for the calculation of a T-matrix of a single particle consists of a huge number of integrals over the particle surface which contain products of the spherical vector wave functions. There are some numerical problems in so called “difficult” cases:

- particle with a complex shape (due to difficulties by approximation of the particle shape),
- particle with a high aspect ratio (due to a large divergence of the radius vector over a the particle surface and therefore due to a large radial divergence of SVWFs under the integral),
- particle with a high refractive index (due to a larger radial divergence of SWVFs for a larger refractive index),
- if nonsingularities of the scattered field continued inside the particle are placed close or outside the sphere inscribed in the particle.

This can be resolved by increasing of the number of integration points which leads to longer calculation time.

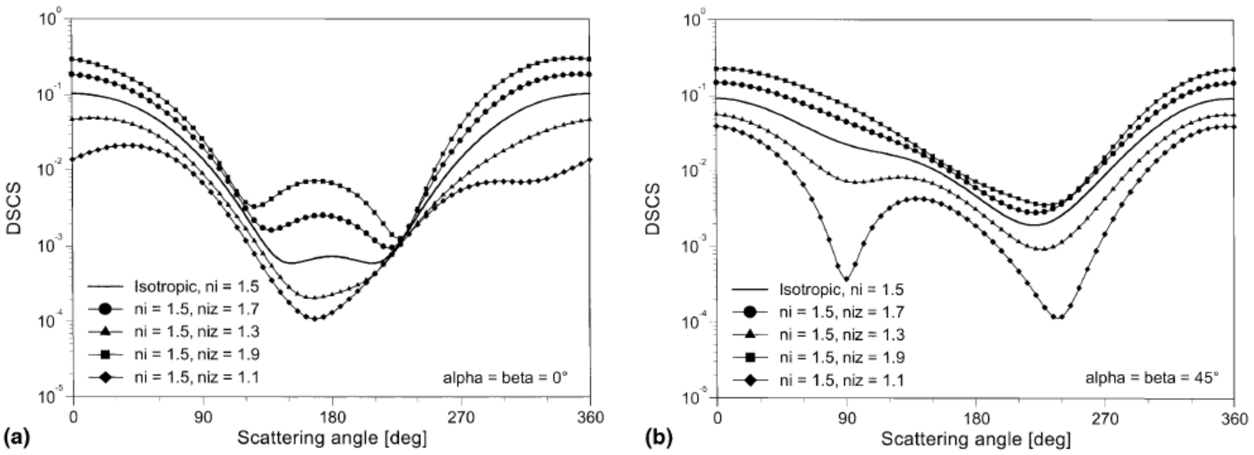


Fig. 1. Normalized differential scattering cross sections for positive uniaxial anisotropic ellipsoids with n_i=1.5, n_{iz}=1.7 and n_i=1.5, n_{iz} =1.9 and negative uniaxial anisotropic ellipsods with n_i =1.5, n_{iz} =1.3 and n_{iz}=1.5, n_{iz}=1.1. The orientation of the particle is: (a) α=0, β=0, γ=0 and (b) α=45, β=45, γ=0. The dimensions of the ellipsoids are $k_s a$=1.0, $k_s b$=1.5 and $k_s c$=2.0. The maximum number of azimuthal modes is M_{max}=8, while the maximum expansion order is N_{max}=10 [6].

On modern computers an effective method to accelerate the algorithm is the usage of parallel computing. Parallel computing is a form of computation in which many calculations are carried out simultaneously, operating on the principle that large problems can often be divided into smaller ones, which are then solved concurrently.

There are a lot of approaches for the parallelization of programs, each of them is preferable for specific problems due to their advantages and disadvantages. The well known are Open Multi-Processing (OpenMP) [7], which is mostly used on shared memory systems (multi-processors computers), and Message Passing Interface (MPI) [8], which is mostly used on distributed memory systems (computer clusters).

4.1. Parallelisation algorithm

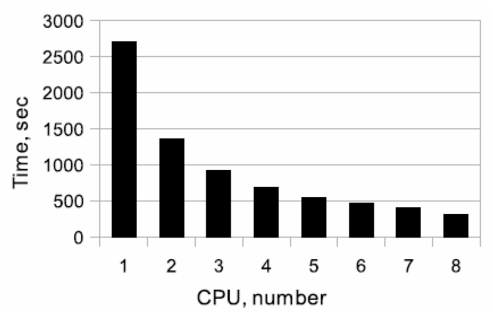
In a first step, we choose the OpenMP paradigm to parallelize the NFM-DS program. Here, the integration process for the Q and the RegQ matrixes elements was parallelized.

Using the gauss integration rule the calculation of integrals can be approximated as a weighted sum of function values at specified points within the domain of integration

$$\int_S f(x)dx = \sum_{i\in A} w_i f(x_i),\ \ A = \{1,..N_{\text{int}}\},$$

where S is particle surface, x_i and w_i are points and weights of a particle surface element, N_{int} is the number of series items. For the parallelization on a computer with N processors we divide the particle surface into N areas $A = \bigcup_{j=1}^{N} A_j$, provide summation simultaneously for each area $I_k = \sum_{i \in A_k} w_i f(x_i),\ k = \overline{1, N}$ and then collect the results $\sum_{k=1}^{N} I_k$. For example, on a computer with six processors each face of the cube would be integrated separately. Thereby the size of the total transferred data between the calculation kernels is minimal. Because transfer time is normally the restricting factor for parallel programs, here the efficiency of the developed program is relative high. On the other hand this improvement increases the memory needed (N+1)-fold. The memory is required to store the calculated Q-matrices for each computing thread and for the master thread. An exemplary results for the calculation time depending on the number of used cores is printed in Fig. 2.

Fig. 2: An exemplary calculation time on computers with different number of processors computing scattering by a cube with size parameter $k_s a$=4 and refractive index m_r=(1.1; 1.1; 1.5).



4.2. Parallelized subroutines

Using the OpenMP technology the following subroutines from the module Proces1.f90, which provides integration over the particle surface, were parallelized: matrix_Q, matrix_Q_m, matrix_Q_sym, incident_matrix_m. These subroutines are used for the calculation of the T-matrix using localized and distributed sources for the scattering problems presented in Table 2.

4.3. Usage of the program

To start the developed program in the parallel mode it has to be compiled with a OpenMP compatible Fortran compiler. Usually all modern, both commercial (eq. Intel® Fortran Compiler 11.0) and open-source (eq. GNU Compiler Collection 4.3.3, Gfortran 4.3) compilers support this option. The compilation instructions are described in the Sec. 7. Further the compiled program can be started on a arbitrary multi-processor computer without any special customization.

Table 2. The scattering problems for which parallelized subroutines can be used.

Routine	**Light scattering by …**
TAXSYM.f90	a homogeneous, dielectric (isotropic, chiral) and perfectly conducting, axisymmetric particle
TNONAXSYM.f90	a homogeneous, dielectric (isotropic, chiral) and perfectly conducting, nonaxisymmetric particle
TINHOM.f90	an inhomogeneous, dielectric, axisymmetric particle with an arbitrarily shaped inclusion
TPARTSUB.f90 TPARTSUBFILM.f90	a particle on or near a plane surface

The command line to start the program (on Windows or Unix systems) is

```
<program > -num_threads <N>
```

where <program> is the name of a executable file, <N> is the number of parallel threads to be started. The parameter <N> can have any positive value, but we advise to define a number, which is equal or less than the number of computational cores on the computer. This parameter is valid also only for the scattering problems, which are listed above and where the OpenMP technology was implemented (see Table 2). It has also to be noted, that the memory requirements are increased (N+1)-fold by using <N> computational cores.

5. Automation of simulations

The NFM-DS is the good program code based on a good theoretical basis for simulations of light scattering by an ensemble of single particles or aggregates. Practically a lot of simulations have to be provided before a suitable result will be obtained. Firstly, because there are no automatic built-in convergence criteria in the NFM-DS program, the user has always to perform own convergence tests. Secondly, the practical interest exhibits not a single simulation, but multiple simulations.

In the original NFM-DS program the input parameters can be specified only through the keyboard and in three input files, which are different for each kind of the scattering problem. It can reduce the practical effect of the NFM-DS program for multiple simulations. To decrease this disadvantage we have developed some improvements in the program such as

- parameters for the scattering program can be provided not only through input file, but also through the command line which is used to start the program;
- wavelength dependencies of the refractive index for common materials (such as silver, etc.) are included in the program;
- parameters entered normally through the keyboard can be provided through the command line which is used to start the program.

Below we describe how to use these features.

5.1. Variables in the input files

The input parameters in the NFM-DS program for each scattering problem are specified in the corresponding input files. They all are called like “InputXXX.dat”, where XXX denotes the scattering problem (AXSYM, NONAXSYM, et all.). The structure of each input files is very simple and was explained in detail in the program description on the CD enclosed in the book [5].

The input parameters in input files are divided between several groups each specified by a keyword. Once the program have recognized the group through the corresponding keyword, it expects a sequence of parameter values. Each line consists only one value of the corresponding parameter. The sentence of these parameters are strictly defined for each group in the program and described in comment lines in the input files. An exemplary input text for the *OptProp* group (optical properties of material) is presented in the first line of Table 3.

In the current NFM-DS program values of parameters can be specified not only directly in the input file, but also outside of them using so-called a variable mechanism. In this case instead of a explicit value the following sentence have to be written in the input file

@<variable_name>|<default_value>@

or

@<variable_name>@

where <variable_name> is the name of the variable, which value should be defined outside the file, <default_value> is the default value, in cases when no value is defined outside. The name is case sensitive.

The value of the variable can be provided in the command line. During reading the input file by simulation this sentence will be automatically replaced with it or <default_value>, if no value is provided in the command line. Otherwise, the sentence will be not replaced.

This feature is available for the following scattering problems: AXSYM (T-matrix for a homogeneous, isotropic, chiral and perfectly conducting, axisymmetric particle), NONAXSYM (T-matrix for a homogeneous, isotropic, chiral, perfectly conducting and uniaxial anisotropic, non-axisymmetric particle) and SCT(scattering characteristics using a previously calculated T-matrix).

Some examples of the usage of variables in input files are presented below in Table 3.

Table 3. Exemplary usage of variables in the InputAXSYM.dat file, here the group OptProp (properties of a particle and of the ambient medium).

1. parameters without variables	OptProp 0.628318530717959 1.0 (1.5, 0) Variables: - wavelength - wavelength of the incident light in vacuo. - ind_refMed - refractive index of the ambient medium. - ind_refRel - relative refractive index of the particle.
2. parameters with and without the default value	OptProp @wave|0.628318530717959@ 1.0 @ref_index@ Variables: - wavelength - wavelength of the incident light in vacuo. - ind_refMed - refractive index of the ambient medium. - ind_refRel - relative refractive index of the particle.
3. complex parameters	OptProp 0.628318530717959 1.0 (@ReRef@, @ImRef@) Variables: - wavelength - wavelength of the incident light in vacuo. - ind_refMed - refractive index of the ambient medium. - ind_refRel - relative refractive index of the particle.

The value of variables can be specified in the command line by starting of the program. The format of the command line is the following

```
<program > -<variable1> <value1> -<variable2> <value2> …
```

where <program> is the name of a executable file, <variableX> and <valueX> are names and values of corresponding variables. There are some rules about possible values of variables. The value of the variable must be not longer then 80 characters. If the value contains a space character or some other special characters, it must be to quoted.

There is an example of a command line

```
./double_par_TMATRIX -wave 0.6283 -size_a 1.2 -ref_index ”(1.5, 0.001)”
```

5.2. Refractive index for common materials

The refractive index of some materials such as silver, which are frequently used in simulations, depends on the wavelength of the incident light. We have included the dependences for some materials in the program.

For such material the user can write in the line responsible for the refractive index (see the *OptProp* group in the *Input.dat*, *InputAXSYM.dat* or *InputSCT.dat* files) instead of a explicit value the following sentence

```
@$<name_of_material>@
```

where <name_of_material> is the name of the material. During the calculation this sentence will be automatically replaced with the corresponding value of the refractive index for the current wavelength.

In the program code the dependence of the refractive index $m_r(\lambda)$ is defined through its values for the set of the wavelengths $m_i = m_r(\lambda_i),\ \ i = \overline{1,N}$. Between them the refractive index $m_r(\lambda)$ is calculated using an approximation to a line function

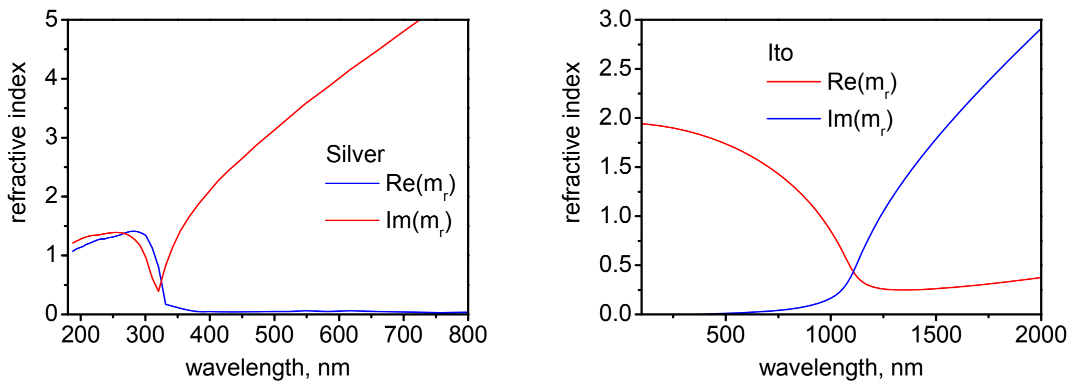
$$m_r(\lambda) = \frac{m_{i+1} - m_i}{\lambda_{i+1} - \lambda_i}(\lambda - \lambda_i) + m_i \,.$$

In Table 4 materials included in the program are presented while on the Fig. 3 the dependences are plotted. For more detailed information we advise the interested user to check the ref_index.f90 and interpretator.f90 modules.

Table 4. Materials which refractive index dependence is included in the program.

Material	**Command**	**Range for wavelength**	**Reference**
Silver	silber	187.85 … 1937.25 nm	Johnson and Christy [9]
Ito	Ito	100 … 2000 nm	Franzen [10]

Fig. 3. The refractive index dependences of Silver (left) and Ito (right) which is included in the program.



5.3. Input from console

In the original NFM-DS program some input parameters like N_{rank}, M_{rank} has to be specified through console input. Now it is also possible to provide these parameters through the command line and moreover to save information printed during simulation on the screen to the specified file.

To use these possibilities the user must start the program with options (both or one of them)

```
<program > -input “<in_data>” -output “<out_file>”
```

where <program> is the name of a executable file, <in_data> is the sequence of parameters to be entered into the program like they would be entered through keyboard, <out_file> is the filename to save the output information of the simulation process. The character of new line (hex code 0x13) is given by writing the "\n" in the sequence <in_data>.

An example of the usage for the calculation of light scattering by a dielectric axisymetric particle:

command line

```
./double_par_TMATRIX -input "1\n100 20\n2\n"
```

screenshot

```
T-Matrix Code for Light Scattering Calculation
- enter an integer specifying the scattering problem:
   1 - dielectric, perfectly conducting or chiral, axisymmetric particle;
   2 - dielectric, perfectly conducting or chiral, nonaxisymmetric particle;
   3 - axisymmetric, composite particle;
   4 - axisymmetric, layered particle;
   5 - inhomogeneous, axisymmetric particle with an arbitrary inclusion;
   6 - inhomogeneous sphere with a spherical inclusion;
                                          ....
         1

Convergence Test for an Axisymmetric Particle
---------------------------------------------

Nrank estimate:
the estimated value of Nrank from Wiscombe's criterion is  10;

- enter the estimated values of Nint and Nrank, where Nint = Ndgs * Nrank
  and Ndgs = 5,6,...;
       100          20

- enter the type of convergence test: 1 - Nint, 2 - Nrank, 3 - Mrank;
         2

progress of main calculation:
-    1  /    9;
-    2  /    9;
-    3  /    9;
-    4  /    9;
-    5  /    9;
-    6  /    9;
-    7  /    9;
-    8  /    9;
-    9  /    9;

Convergence criterion for Nrank is satisfied;
```

6. Obtaining and installing of the source code

The source code can be downloaded from the internet portal http://www.scattport.org, which is provided by the our research group [11]. The package must be downloaded and unzipped in any folder. It contains the following directories:

- BIN
- TMATSOURCES,
- TMATFILES,
- INPUTFILES,
- GEOMFILES, and
- OUTPUTFILES.

The executable program must be created in the directory BIN and must include the main program “TMATRIX.f90” and all F90 routines contained in the directory TMATSOURCES. To compile the code on a Unix system use the makefile supplied in the directory BIN, but edit the makefile to provide the required compiler options. To compile the code using for instance Compaq Visual Fortran or Microsoft Developer Studio, create a project in the directory TMATSOURCES and add all F90 files to the project. The input parameters for each scattering problem are provided in text files in the directory INPUTFILES like as the geometry files are contained in the directory GEOMFILES. The directory TMATFILES contains text files with calculated T-matrices.

7. Compiling

OpenMP is a standard for the support of shared-memory parallel programming and can provide a performance advantage when using the NFM-DS on multi-cores or multi-processors platforms.

To compile the developed program an OpenMP compatible Fortran compiler is required. All modern, both commercial (eq. Intel® Fortran Compiler 11.0) and open-source (eq. GNU Compiler Collection 4.3.3, Gfortran 4.3) compilers support OpenMP. In the downloadable source file we also offer a makefile for easy compilation of the program using the Intel® Fortran Compiler and Makefile utility.

7.1. OpenMP

If an *Integrated Development Environment (IDE)* like the Microsoft Visual Studio is to be used to compile the program, one must find in the project properties options like “Parallelization” and “OpenMP Conditional Compilation” and set them to True. If the user uses the command line compiler eq. the Intel Fortran Compiler on Linux System, then the corresponding options like “-openmp” and “-parallel” must be included. Please see your compiler documentation for more detailed information.

7.2. Double vs. Extended

The NFM-DS program is written to allow an easy generation of either double- or extended-precision versions of the executable file. The precision control parameter O is defined in the file “Parameters.f90”. For double-precision calculation, must be set O = kind(1.d0), while for extended-precision calculation, the statement O = kind(1.q0).

This mode can be chosen also through pre-processor definitions. This is a typical feature of modern compilers which helps to automatically change the program code before the compilation step. For extended-precision calculation the pre-defined variable PRECISION_QUAD must be set, otherwise double-precision calculation will be used.

If an IDE like the Microsoft Visual Studio is to be used, the user must find in the project properties a option like “Pre-processor Definitions” and enter PRECISION_QUAD. For a command line compiler eq. the Intel Fortran Compiler on Linux System the corresponding option like “-DPRECISION_QUAD” must be included. For more detailed information we advise the user to look his compiler documentation.

Of course, the extended precision version will demand about twice as much memory as the double-precision version. The extended precision codes are also slower than the double-precision codes by a factor of 5-6, but allow computations for larger scatterers.

8. Some problems with memory allocation

There are some problems, which have been observed using the Intel Fortran Compiler. For large problems (large Q- and T-matrices) a memory allocation error can arise during running the program. Even when the computer should have enough memory for such simulation. It has to be noted that this problem has not been occurred with other compilers. Up to now the exact explanation of this error isn’t known to us.

There is an excerpt from the Intel Fortran Documentation and internet forums devoted to high performance simulation using OpenMP.

There are three options in Intel Fortran Compiler, which have an influence on how the variables are allocated during the execution: on the call stack, in static memory or in heap.

/save

causes all variables to be placed in static memory

/auto

causes all local variables, which do not have SAVE attribute, to be allocated to the run-time stack. It does not affect variables that have the SAVE attribute or ALLOCATABLE attribute.

/auto-scalar

causes scalar variables of intrinsic types INTEGER, REAL, COMPLEX, and LOGICAL that do not have the SAVE attribute to be allocated to the run-time stack. All other variables are allocated statically.

If you are not using OpenMP, Intel Fortran Compiler default uses the /auto-scalar. But if you say -openmp, this changes to /automatic and all local variables are allocated on the stack. This is required to allow for thread-safety.

So, if the program uses OpenMP, then the size of run-time stack might be not enough for the memory allocation of Q-matrices for each threads and we need increase it. Therefore it should be increased.

8.1. on a Linux/Unix system

There are some utilities for Linux/Unix systems, which can change the stack size used by a program, or set it unbounded. We advise to use the ulimit utility, which controls the resources available to a process started by the shell. To change the maximum size of stack this utility has to be started once prior to the NFM-DS program with the option -s and the maximal size of call stack

An example of the command line to start this utility

```
ulimit -s 10000000
```

8.2. on a Windows system

For Windows there are no such utilities. The stack size is defined by the compilation of a program exactly by linking the compiled object files in the executable file. The necessary parameter in the Intel Fortran Compiler used by compilation in the command line is “/STACK:reserve,commit”, where reserve and commit are the total stack allocation sizes in virtual and physical memory respectively. In an IDE there have to be also the corresponding options like “Stack Reserve Size” and “Stack Commit Size”. Please see your compiler documentation for more detailed information.

Bibliography

1. P.C. Waterman, “Symmetry, unitary and geometry in electromagnetic scattering,” Physical Review D 3 (1971), 825–839.
2. M.I. Mishchenko, G. Videen, N. G. Khlebtsov, T. Wriedt, N. T. Zakharova “Comprehensive T-matrix reference database: A 2006–07 update,” Journal of Quantitative Spectroscopy & Radiative Transfer 109 (2008), 1447-1460.
3. T. Wriedt (Editor), *Generalized Multipole Techniques for Electromagnetic and Light Scattering*. Elsevier, Amsterdam (1999).
4. Wriedt T. Review of the null-field method with discrete sources. JQSRT 2007; 106:535-545.
5. A. Doicu, T. Wriedt, Y.A. Eremin, *Light Scattering by Systems of Particles. Null-field Method with Discrete Sources. Theory an Programs*. Springer, Berlin Heidelberg New York (2006).
6. A. Doicu, “Null-field method to electromagnetic scattering from uniaxial anisotropic particles,” Optics Communications 218 (2003), 11–17.
7. OpenMP Architecture Review Board, “The OpenMP API specification for parallel programming,” http://openmp.org
8. University of Tennessee, “Message Passing Interface (MPI) Forum Home Page,” http://www.mpi-forum.org/
9. P.B. Johnson, R.W. Christy, “Optical constants of the noble metals,” Phys Rev B 6 (12) 1972, 4370–4379.
10. S. Franzen, “Surface Plasmon Polaritons and Screened Plasma Absorption in Indium Tin Oxide Compared to Silver and Gold,” J. Phys. Chem. C 112 (2008), 6027–6032.
11. J. Hellmers, T. Wriedt, “New approaches for a light scattering Internet information portal and categorization schemes for light scattering software,” doi:10.1016/j.jqsrt.2009.01.023