PWPP

Plane wave plane wave (PWPP) code for predicting
reflection and transmission from random particle layers

D. W. Mackowski
Department of Mechanical Engineering
Auburn University, Auburn, AL 36849, USA

July 18, 2018

Contents

1 Plane wave plane parallel formulation

1.1 Overview

The plane parallel plane wave (PWPP) model is a mathematical formulation for numerically approximating solutions to the time harmonic (with periodic factor $\exp(-\mathsf{i}\omega t)$) Maxwell's wave equations (MWEs) for a plane parallel layer of a discretely inhomogeneous medium exposed to a linearly polarized monochromatic plane wave. In what follows all lengths are made dimensionless by scaling with the free space wavenumber $\mathrm{k}_0 = 2\pi/\lambda_0$ with λ_0 being the vacuum wavelength of the incident wave; the dimensional value corresponding to the dimensionless layer thickness H, for example, would be H/k_0. The normal directions of the layer boundaries are parallel to the $\hat{\mathbf{z}}$ direction, and the inhomogeneous medium is taken to reside entirely within $0 \le z \le H$. The incident field has a propagation direction $\hat{\mathbf{k}}_0$ specified by the polar and azimuth angles β_0, α_0, and it is assumed that $0 \le \beta_0 < \pi/2$, i.e., the incident field is propagating in the upwards direction. The polarization of the incident field is either parallel or perpendicular to the meridional plane formed by the $\hat{\mathbf{z}}$ and $\hat{\mathbf{k}}_0$ directions. The halfspaces $z < 0$ and $z > H$ are referred to as regions 0 and H, and have uniform permittivities of ϵ_0 and ϵ_H; permittivity is equivalent to the square of the complex medium refractive index. Medium 0 is assumed to have a real permittivity (i.e., non-absorbing material), whereas ϵ_H may be complex.

This formulation is described in detail in [1]; a brief synopsis will be given here. The plane parallel layer is assembled from a 2D infinite periodic lattice of $W \times W \times H$ unit cells, where W is the cell width, with each unit cell containing a microscopically-specified and identical inhomogeneous distribution of material. Within the layer the complex electric field amplitude will be governed by

$$(-\nabla \times \nabla \times + \epsilon(\boldsymbol{\rho}, z)\,)\ \mathbf{E}(\boldsymbol{\rho}, z) = 0 \tag{1}$$

in which $\boldsymbol{\rho} = \hat{\mathbf{x}}\, x + \hat{\mathbf{y}}\, y$ is the lateral position vector, z is the depth coordinate, and $\epsilon = \mathsf{m}^2$ is the position-dependent permittivity. The periodic conditions of the electric field amplitude in the lateral plane are identically satisfied by the expansion

$$\mathbf{E}(\boldsymbol{\rho}, z) = \sum_{\mathbf{s}} \mathbf{p}^{\mathbf{s}}(z)\, e^{\mathsf{i}\, \mathbf{k}_\rho^{\mathbf{s}} \cdot \boldsymbol{\rho}} \tag{2}$$

in which $\mathbf{k}_\rho^{\mathbf{s}}$ is the 2D reciprocal lattice (RL) vector given by

$$\mathbf{k}_\rho^{\mathbf{s}} = \mathbf{k}_\rho^0 + \frac{2\pi \mathbf{s}}{W}, \quad \mathbf{s} = s_x\, \hat{\mathbf{x}} + s_y \hat{\mathbf{y}}, \quad s_x,\ s_y = 0, \pm 1,\ \pm 2, \ldots \tag{3}$$

with $\mathbf{k}_\rho^0$ denoting the project of the incident propagation vector onto the lateral plane, and $\mathbf{p}^{\mathbf{s}}(z)$ is a vector amplitude function for the $\mathbf{s}$ RL mode. By utilizing the orthogonality properties of the exponential functions when integrated over the lateral area of the unit cell, a system of vector ordinary differential equations (ODEs) in z can be obtained from Eq. (1) for each RL mode index $\mathbf{s}$. This system

can be put in the form

$$\left(-\widetilde{\nabla}^{\mathbf{s}} \times \widetilde{\nabla}^{\mathbf{s}} \times + \bar{\epsilon}\right) \mathbf{p}^{\mathbf{s}}(z) = -\mathbf{a}^{\mathbf{s}}(z;\bar{\epsilon}) \tag{4}$$

where

$$\widetilde{\nabla}^{\mathbf{s}} \times = \begin{pmatrix} 0 & -d/dz & \mathrm{i}\,\mathrm{k}_y^{\mathbf{s}} \\ d/dz & 0 & -\mathrm{i}\,\mathrm{k}_x^{\mathbf{s}} \\ -\mathrm{i}\,\mathrm{k}_y^{\mathbf{s}} & \mathrm{i}\,\mathrm{k}_x^{\mathbf{s}} & 0 \end{pmatrix} \tag{5}$$

The source function $\mathbf{a}^{\mathbf{s}}(z;\bar{\epsilon})$ – which is equivalent to a displacement field – is defined by

$$\begin{aligned} \mathbf{a}^{\mathbf{s}}(z;\bar{\epsilon}) &= \sum_{\mathbf{s}'} \left(\epsilon^{\mathbf{s}-\mathbf{s}'}(z) - \bar{\epsilon}\,\delta_{\mathbf{s}-\mathbf{s}'}\right) \mathbf{p}^{\mathbf{s}'}(z) \\ &= \sum_{\mathbf{s}'} \hat{\epsilon}^{\,\mathbf{s}-\mathbf{s}'}(z;\bar{\epsilon})\,\mathbf{p}^{\mathbf{s}'}(z) \end{aligned} \tag{6}$$

where $\delta_{\mathbf{s}-\mathbf{s}'}$ denotes the Kronecker delta function, and $\epsilon^{\mathbf{s}}(z)$ denotes the lateral finite Fourier integral transform of the permittivity distribution in the unit cell:

$$\epsilon^{\mathbf{s}}(z) = \frac{1}{W^2} \int_{-W/2}^{W/2} dx \int_{-W/2}^{W/2} dy\, e^{-\mathrm{i}\,\mathbf{k}_\rho^{\mathbf{s}}\cdot\boldsymbol{\rho}}\, \epsilon(\boldsymbol{\rho}, z) \tag{7}$$

The complex–valued parameter $\bar{\epsilon}$ can be viewed as a reference (or effective) permittivity of the medium: it has no effect on the amplitude function $\mathbf{p}^{\mathbf{s}}(z)$ for a given spatial distribution of permittivity and a given incidence – which can be seen by combining Eqs. (6) and (4) – yet it will have a substantial effect on the numerical effort required to obtain a solution. It is implicitly assumed that the summation over the x and y RL indices in Eqs. (2) and (6) can be truncated at some appropriate maximum index s_{max}, which would result in $(2\,s_{max}+1)^2$ vector ODEs in all.

A 1D dyadic Green's function (DGF) can be derived for the ODE of Eq. (4). Unlike the free space DGF used in traditional discrete dipole (DD) methods, the DGF employed here explicitly accounts for the boundaries of the slab at $z = 0$ and H. Specifically, the DGF gives the $\mathbf{s}$-mode field at point z resulting from a source at point z', with both z and z' residing within a slab of thickness H and uniform permittivity $\bar{\epsilon}$ that is backed on both sides by media of permittivity ϵ_0 and ϵ_H. A 1D integral solution to Eq. (4) can be constructed from the DGF via

$$\mathbf{p}^{\mathbf{s}}(z) = \Phi^0(z;\bar{\epsilon})\,\delta_{\mathbf{s}} + \int_0^H \mathbf{G}^{\mathbf{s}}(z,z';\bar{\epsilon}) \cdot \mathbf{a}^{\mathbf{s}}(z';\bar{\epsilon})\,dz', \quad 0 < z < H \tag{8}$$

where $\Phi^0(z;\bar{\epsilon})$ is the complex vector amplitude existing in the homogeneous reference slab, of permittivity $\bar{\epsilon}$, when excited by the incident wave, and $\mathbf{G}^{\mathbf{s}}(z,z';\bar{\epsilon})$ is the DGF for the slab.

An integral equation for the source function $\mathbf{a}^{\mathbf{s}}(z;\bar{\epsilon})$ is obtained from the discrete convolution of Eq. (8) with the transformed permittivity, per Eq. (6). This equation is discretized by dividing the layer thickness into N_H elements,

each of thickness $d_H = H/N_H$, and within which the source function is assumed constant. Likewise, the transformed permittivity in Eq. (7) is calculated, at each layer element, by dividing the lateral area into N_W^2 square elements each of width $d_W = W/N_W$, assigning a uniform permittivity to the element based upon the material distribution within the element, and approximating the Fourier integrals with a discrete Fourier transform; this operation can be performed with a FFT. The working system of equations so derived appear as

$$\mathbf{a}^{\mathbf{s},j} - \sum_{\mathbf{s}'} \hat{\epsilon}^{\,\mathbf{s}-\mathbf{s}',j} \sum_{j=1}^{N_H} \mathbf{G}^{\mathbf{s}',j,j'} \cdot \mathbf{a}^{\mathbf{s}',j'} = \hat{\epsilon}^{\,\mathbf{s},j}\, \Phi^{0,j} \tag{9}$$

where superscript j denotes the depth coordinate. The application of the DFT to calculation of the permittivity spectrum constrains the largest RL mode index to $s_{max} = N_W/2$, and the associated maximum wavevector magnitude to $\mathrm{k}_{max} = \pi/d_W$. For most calculations – especially those modeling a random inhomogeneous medium – a significantly smaller range of RL modes can be used; this is discussed further at the end of this section.

The scattered far-field from the periodic layer takes the form of a diffraction pattern, i.e., a discrete superposition of transverse plane waves with propagation direction vectors $\mathbf{k}^{\mathbf{s}}$ given by

$$\mathbf{k}^{\mathbf{s}} = \mathbf{k}_\rho^{\mathbf{s}} + \sigma\, \mathrm{k}_z^{\mathbf{s}} \tag{10}$$

$$\mathrm{k}_z = \left(1 - \left|\mathbf{k}_\rho^{\mathbf{s}}\right|^2\right)^{1/2} \tag{11}$$

with $\sigma = \pm 1$ denoting whether the wave is propagating upwards (transmission) or downwards (reflection). The complex amplitude of the $\mathbf{s}$ mode reflected or transmitted wave will be a function solely of the corresponding $\mathbf{s}$ mode source coefficients. Only the propagating RL modes, corresponding to $\left|\mathbf{k}_\rho^{\mathbf{s}}\right| < 1$, will appear in the far-field reflection and transmission, yet the evanescent modes ($\left|\mathbf{k}_\rho^{\mathbf{s}}\right| > 1$) will contribute to the coupling of EM energy between lattice planes (per Eq. (9)) and as such will indirectly affect the far-field scattering. The polar angle increment between diffraction peaks is $\sin\Delta\theta = 2\pi/W$, or $\Delta\theta \sim 2\pi/W$ for $2\pi/W \ll 1$, and in this respect a criterion for the unit cell width W is that it be sufficiently large to provide a sufficiently fine RL mesh to resolve the directional features of reflection and transmission.

The discretized numerical model in Eq. (9) contains $3N_W^2 N_H$ degrees of freedom (DOF), with the 3 accounting for the 3D vector nature of the source function. A limited set of calculations presented in [1] indicated that – under random media conditions – the number of RL modes could be truncated to a value $N_R^2 \ll N_W^2$ with negligible effect on the calculated hemispherical reflectance and transmittance properties of the layer [2]. A convenient parameter for indicating the cutoff point is the wavenumber k_R, defined so that $N_R = W\mathrm{k}_R/\pi = N_W d\mathrm{k}_R/\pi$. Setting $\mathrm{k}_R = 1$ would include primarily the propagating modes in the calculation, and $\mathrm{k}_R = \pi/d$ would include all N_W^2 modes. Implementing this strategy is basically equivalent to performing a spectral windowing (or filtering) operation on the permittivity distribution in direct space.

That is, the convolution over index $\mathbf{s}'$ in Eq. (9) would be performed by the ubiquitous FFT procedure, and doing so requires storage of the permittivity distribution at $4N_R^2$ spatial points. This distribution is obtained by first generating permittivity distributions in the W^2H unit cell, which are then FFT'd, square windowed, and inverse FFT'd back to the spatial domain. The advantage of the method is that it retains the benefits of the large W – in that it can be chosen to provide a sufficiently small angular resolution – while reducing the size of the system of equations to a tractable level.

A Fortran-90+MPI code was developed to generate solutions to Eq. (9). For a given unit cell configuration and incident direction, calculation of the 2×2 complex amplitude matrix and the 4×4 Stokes matrix, for reflection and transmission and at each propagating RL mode, require a pair of solutions to Eq. (9) for two mutually orthogonal incident polarizations. The linear system in Eq. (9) is solved iteratively using the biconjugate gradient algorithm, and the convergence rate of this algorithm is strongly affected by the choice of the reference permittivity $\bar{\epsilon}$. In [1] it was shown that the optimum $\bar{\epsilon}$ corresponds to the effective permittivity of the medium, which describes the propagation and attenuation of the coherent field. The effective permittivity of the medium is not a quantity that can be easily calculated *a-priori*. The code employs a heuristic scheme to assign a value to $\bar{\epsilon}$, in which

$$\overline{\mathsf{m}} = \sqrt{\bar{\epsilon}} = \mathsf{m}_V + \mathsf{i}\,\frac{\kappa}{2} \tag{12}$$

where m_V is the volume-averaged complex refractive index of the unit cell, and κ is a dimensionless, real, and positive radiative transfer extinction coefficient. The extinction coefficient κ can be user–specified, or it can be internally estimated given the particle size, composition, and volume fraction of the medium via

$$\kappa = \frac{3Q_{ext}\, f_V}{4a_P} \tag{13}$$

with Q_{ext} the average dimensionless extinction efficiency for the particle of volume–mean dimensionless radius a_P, and f_V the particle volume fraction. The extinction efficiency is calculated in the code using Mie theory; this assumes the particles are spherical in shape.

Calculations of observable reflection and transmission, for random media, require configuration averages of the amplitude and Stokes matrices at the propagating RL directions. The number of configurations needed to produce acceptable averages depends on the sought radiative quantity being calculated. A relatively large number of configurations – on the order of several 100 to 1000 – will typically be required to produce directional distributions of polarized bidirectional reflection and transmission that are relatively free from speckle patterns associated with the individual configurations. On the other hand, hemispherical reflection and transmission – which are obtained by integration of the bidirectional quantities over the hemisphere, have much less sensitivity to fluctuations in individual configurations, and typically require around 10 configurations to produce stable averages.

2 PWPP code

2.1 Structure and compilation

The code is organized into the following five components:

`pwpp-modules.f90:` Contains modules for special functions.

`pwpp-solver.f90:` Subroutines for setting up and solving the interaction equations and calculation of the scattering field and associated quantities.

`pwpp-input.f90:` Subroutines for input of calculation parameters and generation of configuration–averaged quantities.

`pwpp-target_generation.f90:` Subroutines for generation of random particle configurations in the unit cell, and for simulating diffusion, agglomeration, and restructuring of particle configurations.

`pwpp-main.f90:` Initializes MPI and calls the main drivers.

`mpidefs-parallel.f90:` A module which defines the MPI commands for use on multiprocessor platforms.

`mpidefs-serial.f90:` A module which defines MPI commands for use on single processor (serial) platforms.

`pwpp-intrinsics.f90:` Compiler–specific (non–standard Fortran) functions for command–line argument retrieval and system time operations. The users must modify this module to suit their specific compiler.

`fftmod.f90:` Fast Fourier transform subroutines. Contained is the 2-3-5 generalized prime fft code of Templeton and ancillary subroutines.

Compilation of the code using the GNU *gfortran* on a MS-Windows, single processor machine would involve

```
gfortran  -o pwpp.exe pwpp-intrinsics mpidefs-serial pwpp-modules
  fftmod pwpp-solver pwpp-target\_generation pwpp-input pwpp-main
```

This places the executable in the file `pwpp.exe`. Compilation using the MPICH2 package for execution on a parallel machine would use

```
mpif90 -I/opt/mpich2-1.2.1p1/include -g -o pwpp.out
         pwpp-intrinsics mpidefs-parallel pwpp-modules fftmod
         pwpp-solver pwpp-target\_generation pwpp-input pwpp-main
```

and would put the executable in `pwpp.out`.

Other compilers follow the same basic plan. It is important to compile the module files in the order they are given. And remember that `pppw-intrinsics.f90` must be modified to match the command–line recognition and retrieval intrinsic functions of the compiler. The distribution has the intrinsics set up for *gfortran*.

2.2 Execution

Parameters to the code are passed to the code by use of an input file, in which the input file is designated via a command-line argument. in a windows command shell, the serial code would be launched via

```
pppw pppw_01.inp
```

where, for this example, the executable code and the input file are named `pppw.exe` and `pppw_01.inp`. The default filename `pppw.inp` is used for the input file if no command line argument is given; this file must be present in such cases.

2.2.1 Parallel operation

When using parallel platforms, the number of processors in the parallel run must be some integer multiple of four. Each group of 4 is dedicated to solving Eq. (9) for a particular configuration: 2 processors for parallel incidence, and 2 for perpendicular. The biconjugate gradient iteration algorithm used to solve Eq. (9) requires two matrix-vector multiplications for each iteration, and two processors allow these multiplications to be performed simultaneously. Four processors, in this regard, can accelerate the determination of the scattering matrix for a particular configuration by a factor close to four when compared to a serial computation. Application of additional processors to the solution of Eq. (9) would result in a considerably smaller scaling factor. Because of this, separate groups of 4 are used to perform the configuration averaging processes.

Launching the code in parallel mode would involve a statement of the form

```
mpirun -n 16 pppw.out pppw_01.inp
```

which would start the code on 16 processors.

2.3 Input file

The input file consists of paired lines; the first line of a pair representing a parameter ID, and the second representing the value or option for that parameter. Order of the option pairs in important only for situations in which conflicting options are chosen; in this case, the last option pair appearing in the input file will be the one in effect. If a pair corresponding to a particular parameter is not present, the code will use the default value.

Character variables should be written with no enclosing quotes. Logical variables appear as character `t` or `f`, no enclosing quotes. Integer and real variables appear in conventional form. Complex variables appear as (`re part`, `im part`), with `re part`, `im part` both representing real numbers.

2.4 General options

`output_file`: name of the output file: character, no enclosing quotes (ppdd.dat).

`append_output_file`: logical variable (character `t` or `f`, no enclosing quotes). If `t` and `output_file` exists, output will be appended to the `output_file`, otherwise the file is created or overwritten.

`run_file`: file name for intermediate output. If this option is absent the intermediate output is written to the standard output (unit 6).

`dipole_map_input_file`: The coordinates and property information of the dipole target used in the calculations are read from this file. The format is described below. If this option is absent the dipole map is generated internally, using options described below.

`dipole_map_output_file`: The coordinates and property information of the dipole target used in the calculations are written to this file. If absent no dipole map information is written.

`target_width`: The dimensionless width of the unit cell (100.)

`target_thickness`: The dimensionless thickness of the layer (20.)

`ri_front_surface`: The complex refractive index of medium 0, $\mathtt{m}_0$ (1.+0.i)

`ri_back_surface`: The complex refractive index of medium H, $\mathtt{m}_H$ (1.+0.i)

`ri_binder`: The complex refractive index of the material, within the layer, that is not occupied by particles (1.+0.i)

`ri_medium`: The complex reference refractive index, $\overline{\mathtt{m}}$, used to define the dyadic Green's function. This quantity is typically set automatically using the options below. Default is 1. + 0i, for which the DGF becomes equivalent to the free space dyadic Green's function.

`auto_ri_medium`: Logical, `t` will have the code calculate $\overline{\mathtt{m}}$ using Eq. (13), and `f` will use the value specified by `ri_medium`.

`precon_absorption`: The value of κ in Eq. (12), used to set `ri_medium`. A typical value is 0.05. One should experiment to find the optimum value which, for a specific target and particle properties, minimizes the number of iterations required for a solution. Set to an arbitrary negative number to implement the approximation scheme in Eq. (13).

`incident_beta_deg`: polar angle of incident field in degrees. Normal incidence is 0. (0).

`incident_alpha_deg`: azimuthal angle of incident field in degrees (0).

`max_iterations`: integer, maximum number of iterations in the BiCG solution method. (1000)

`solution_eps`: solution error tolerance. (1.d-6)

`k_cut`: real valued truncation parameter for the system of equations in reciprocal space. The number of reciprocal lattice points in the lateral direction will be set by N_R^2, with N_R=min[N_W, ceiling[`k_cut`$\times W/\pi$]]. The standard value is 2.5.

`print_scattering_matrix`: logical, `t` will print the directional transmittance and reflectance matrix elements to the output file.

`azimuth_average`: logical, when `print_scattering_matrix` is true *and* when incident is normal, then setting `t` will average the scattering matrix elements over the 2π azimuthal directions.

2.5 Specification of target properties

2.5.1 `dipole_map_input_file` option

The code allows for two methods of specifying the properties of the target. The first method is via the `dipole_map_input_file`. The first line of this file contains N_W and N_H: the number of volume elements (referred to as dipoles) in the lateral and depth coordinates of the unit cell. The remaining lines list the integer coordinates and integer component index of occupied dipole sites. An integer dipole coordinate is a triplet of integers $i_x,\ i_y,\ j$ with $-N_W/2 \le i_x, i_y \le (N_W-1)/2$ and $-N_H/2 \le j \le (N_H-1)/2$. Standard Fortran integer arithmetic is assumed here: if $N_W = 3$ (i.e., odd), then $N_W/2 = 1$. The component index is an integer ranging from 1 to 5, and this index will be used with subsequent information provided in the input file to assign a refractive index to the dipole site. An example of the first few lines of a `dipole_map_input_file` is

```
400 100
-36 -24 -12 1
-36 -23 -12 1
-36  12 -12 2
```

The listing of the dipole coordinates does not need to be in any particular order. In addition, the entire set of $N_W^2 N_H$ dipole sites does not need to be included in the file: sites that are not listed are assumed to have a refractive index equal to `ri_binder`, which is specified in the input file. The value of N_W – but does not have to – conform to the form $2^a 3^b 5^c$, where a, b, c are integers. If this is not the case, N_W will be adjusted to the nearest larger values satisfying this criteria, and the dipole coordinates of the additional sites will be taken to have a refractive index of `ri_binder`.

The complex refractive index associated with a specific component index is specified in the input file via the parameter `part_ri`. The first occurrence of `part_ri` is taken to be that for component 1. Additional components are specified by including the option `new_component` following the first `part_ri`, followed then by an additional `part_ri` and the associated refractive index of

component 2, and so on up to a maximum of 5 components. The specification of refractive index in the input file must appear as (`re part`, `im part`). An example of specifying a three component system in the input file would appear as

```
part_ri
(1.33d0,0.d0)
new_component
part_ri
(1.54d0,0.d0)
new_component
part_ri
(1.9d0,0.1d0)
```

If the `dipole_map_input_file` contains $N_{c,f}$ distinct component indices, and the input file specifies $N_{c,i} < N_{c,f}$ separate refractive index values, the refractive index for the sites associated with $N_{c,i}+1, N_{c,i}+2, \ldots N_{c,f}$ will be set to that corresponding to component $N_{c,i}$ in the input file.

2.5.2 Random configuration option

The second method of specifying the target configuration is by an internal algorithm. This algorithm creates a random configuration of particles by a Monte Carlo method, details of which are described below. The algorithm allows for the configuration to be composed of up to five individual particle components, which each component specifying a set of mean size, shape, size distribution, volume fraction, and refractive index parameters. The properties of each component are specified by the input file options:

`part_mean_radius`: dimensionless volume mean radius of the particle component. (1.d0).

`part_sigma`: dimensionless standard deviation of a log normal particle size distribution for the particle component. Set to 0. for a monodispersion. (0.d0).

`part_fv`: volume fraction of the particle component.

`part_ri`: complex refractive index of the particle component. Should appear as (`re part`, `im part`).

`part_shape`: integer value specifying particle shape. Options are: 1: spheroid, 2: rectangular solid, 3: circular cylinder, and 4: hexagonal cylinder. (1).

`part_ary`: Aspect ratio of y axis of particle; relevant only for shapes 1 and 2. (1.d0).

`part_arz`: Aspect ratio of z axis of particle; relevant for all shapes, and for shapes 3 and 4 this will be the half length/radius of the particle. (1.d0).

`part_shell_thickness`: dimensionless thickness of a shell (or, equivalently, coating) surrounding the particle. Default is zero (no shell), and the thickness should be equal to or greater than the volume element size to be meaningful. See notes below for more information.

`part_shell_ri`: complex refractive index of the shell.

The first set of component options appearing in the input file will be associated with component 1. Additional components can be specified by including the option `new_component` following the first set of component options; the component options following `new_component` will then be associated with component 2, and so on.

Additional options for automatic target generation are as follows

`boundary_type`: integer: 0 sets a natural boundary, where all particles are intact and all points within the particles are constrained to lie within the depth boundaries of the target. When using this option the actual volume fraction of the particles in the target – defined by ratio of the total particle volume to the volume within the unit cell – will typically be somewhat smaller than that specified in the input file by the particle component properties. 1 sets a sheared boundary, in which the target represents a slice, of thickness given by `target_thickness`, of an infinite medium. The particles can now be sheared at the boundary, and the actual volume fraction will correspond closely to the sum of the volume fractions for the components.

`max_number_particles`: integer: the maximum number of particles will be limited to this value. This will only work for a single component system. If not present the number of particles will depend on the particle mean radius, volume fraction, and target volume. Default is 10^6 – which means that the maximum is never reached for any feasible calculation.

`number_particles_specified`: logical, when `t`, *and* when the number of components is 1, the number of particles used to generate the target will be fixed at the specified `max_number_particles`. The thickness of the target will be automatically set using the number of particles, the mean particle radius, the particle volume fraction, and the target width, and the boundary type will be set to 0 (the natural boundary).

`n_configurations`: the number of random configurations used to compute configuration–averaged properties.

The basic algorithm for assembling the configuration is as follows

1. The number of particles, within the unit cell, for each component is estimated via

$$N_{P,i} = \frac{3 f_{V,i} W^2 H}{4\pi\, a_{V,i}^3} \tag{14}$$

where $f_{V,i}$ and $a_{V,i}$ are `part_fv` and `part_mean_radius` for component i: Note that the shell of the particle, if present, does not enter into the calculation of $N_{P,i}$.

2. The algorithm samples a particle radius from a log-normal PSD distribution, based on the given `part_mean_radius` and `part_sigma` for the component, and assembles a volume-discretized representation of the particle based on the specified particle shape. The orientation of the particle is randomly sampled (relevant only for non-spherical particles). The discretized particle is now randomly inserted into the volume-discretized unit cell, subject to the constraint that the particle not overlap with previously–inserted particles.

3. The previous step is repeated until all $N_{P,i}$ particles for all components have been inserted into the unit cell.

Please note that the algorithm can fail for specified particle volume fractions in excess of around 0.45-0.5. This is because the probability of finding a sufficiently large unoccupied volume in the unit cell – to place a sampled particle within without overlap with existing particles – decreases as the number of particles in the cell increases. The algorithm will exit when such sites cannot be found after 3000 random attempts. The code will print a message in the output file when this error occurs: understand that the calculation results for such cases are not representative of the particle conditions specified in the input file.

The code allows for a simulation of particle agglomeration effects. The basic procedure is to let each particle in the configuration initially created by the above process randomly move one volume element, i.e., a random walk. When two particles come into contact they can be allowed to stick together (aggregate), based upon a set sticking probability. The aggregated particle, when formed, will move as a single unit in subsequent random walk steps. This process will continue for a specified number of random walk steps, or until a specified fraction of the initial number of sampled monomer particles have aggregated. Parameters associated with the agglomeration simulation are

`number_diffusion_cycles`: integer, number of random walk steps in the simulation. The simulation will exit when this number is reached, or when all initial particles have aggregated together, whichever comes first. Default is zero.

`aggregation_fraction`: real number $0 \leq f \leq 1$ specifying final fraction of aggregated particles in unit cell: 0 would return the initially sampled configuration, and 1 continue the simulation until a single aggregate is formed. When using this option, set `number_diffusion_cycles` to a large number, i.e., 1000000.

`sticking_probability`: real number specifying probability that two particles coming into contact will aggregate. Default is 1.

Options which are relevant for both options of target specification are

`dipole_spacing`: dimensionless spacing of the volume elements in the depth direction, d_H. (0.2d0).

`lateral_dipole_spacing`: dimensionless spacing of the volume elements in the lateral direction, d_W. When this option is absent the code defaults to $d_W = d_H$.

2.6 Input file commenting, termination, and multiple calculation options

Any input file line beginning with a ! or a % will be treated as a comment and skipped. An `end_of_options` statement will terminate the reading of input options and start a calculation using the options assigned up to that point, and the program will end at the completion of the calculations. A `new_run` statement will terminate input file reading and start a calculation, and at the completion of the calculation the program will read additional options from the input file immediately following the `new_run` statement, up to an `end_of_options` statement or an additional `new_run` statement. Output from the `new_run` calculations will be appended to the output file, except for the situation in which one of the parameters following `new_run` is a new output file name. This option allows one to execute multiple calculations using the same input file. For example, a series of calculations involving different target thicknesses could be performed using the options

```
output_file
test.dat
target_thickness
10.d0
⋮
(more options)
⋮
new_run
target_thickness
15.d0
new_run
target_thickness
20.d0
end_of_options
```

A looped calculation can be performed using the `loop_variable` option. This option allows a numerical input file parameter to be looped over a set starting, ending, and increment value. An example of using this option to iterate the value of `target_thickness` from a value of 5. to 30. with a 2.5 increment is as follows:

```
output_file
test.dat
⋮
(more options)
⋮
loop_variable
target_thickness
5.d0,30.d0,2.5d0
end_of_options
```

Note that using this option will always involve at least 3 lines in the input file: the `loop_variable` command, the parameter ID of the looped value, and the starting, stopping, and increment values. When the looped variable is associated with a component-specific property, such as the mean radius or the refractive index, it is necessary to include the option `component_number`, followed by a line containing the component number, prior to the specific parameter ID. An example of a loop which will vary the $z-$aspect ratio of the second component from 0.5 to 1.5 in 0.1 increments is

```
output_file
test.dat
⋮
(more options)
⋮
loop_variable
component_number
2
part_arz
0.5d0,1.5d0,0.1d0
end_of_options
```

A total of three `loop_variable` commands can be included for any one run to create a nested loop. The first `loop_variable` will represent the outer loop of this nest. The `loop_variable` commands can also be part of options included following a `new_run` command.

2.7 Output

The code writes output to the `output_file` following each solution to the PPDD equations. In the cases where configuration–averaged properties of a random medium are sought, the written output corresponds to the current average following the solution for each randomly-sampled configuration. Information written to `output_file` includes

- The time in seconds required for a single solution to the PWPP equations, and the current elapsed time for the specific run.

- The number of particles in the target sample, and the target–averaged particle volume fraction.

- The effective refractive index of the medium. This quantity is meaningful only for random media, configuration–averaged calculations. It is obtained by a least-square error fit of a plane wave model to the configuration–averaged coherent field in the medium.

- The dimensionless extinction coefficient and optical depth of the medium. These quantities are obtained from the averaged coherent transmittance of the target, t_c, via

$$\mathrm{OD} = -\ln t_c, \quad \kappa_{ext} = \mathrm{OD}/(\mathrm{k}_0 H) \tag{15}$$

- The quantities r_d, t_d, r_c, t_c and a for parallel, perpendicular, and unpolarized incidence, where r, t, and a denote hemispherical-directional reflectance, transmittance, and absorptance, and subscripts d and c denote diffuse and coherent. The coherent transmittance is equivalent to the line–of–sight transmittance using in the above formula. Unpolarized quantities are the average of parallel and perpendicular. The absorptance is calculated via energy conservation: $a = 1 - r_d - r_c - t_d - t_c$. The PWPP model can only strictly guarantee energy conservation when $N_R = N_W$, i.e., when truncation in RS is not employed. Because of this, practical calculations, using `k_cut` ≈ 2.5, will show a non-zero absorptance even in non-absorbing materials (zero imaginary part for all refractive indices). Typically this "phantom" absorption is less than 0.01. Larger values of absorption for non-absorbing materials should be viewed as an indication that the calculation results are not accurate, and a larger value of `k_cut` should be used.

- The diffuse bidirectional reflection and transmission matrix elements for the specific incident direction, providing `print_scattering_matrix` = `t`. When incidence is normal and when `azimuth_average` is set to `t`, the matrix elements are averaged over 2π azimuthal directions, and are listed as a function of polar angle θ. For all other cases the elements are listed according to reciprocal lattice directions $\mathrm{k}_x^{\mathbf{s}} = \sin\theta^{\mathbf{s}} \cos\phi^{\mathbf{s}}$ and $\mathrm{k}_y^{\mathbf{s}} = \sin\theta^{\mathbf{s}} \sin\phi^{\mathbf{s}}$.

2.8 Example input and output files

The following input file (corresponding to `pwpp.inp` in this package) does a single-configuration calculation for hemispherical reflectance and transmission by ice particles. The calculation loops over particle volume fraction.

```
output_file
test.dat
run_file

!dipole_map_output_file
!dtest.dat
append_output_file
f
max_iterations
10000
solution_eps
1.d-4
target_width
100.
target_thickness
10.
precon_absorption
.01
auto_ri_medium
t
k_cut
2.5d0
ri_binder
(1.d0,.00d0)
ri_back_surface
(1.d0,0.0d0)
ri_front_surface
(1.d0,0.d0)
ri_medium
(1.0,0.0d0)
incident_alpha_deg
0.
incident_beta_deg
0.
max_number_particles
10000000
number_particles_specified
f
azimuth_average
t
part_shape
1
part_mean_radius
1.3d0
part_sigma
0.186
part_ary
1.d0
part_arz
1.d0
part_fv
.20d0
part_ri
(1.31d0,0.0d0)
part_shell_thickness
0.d0
part_shell_ri
(1.d0,0.d0)
number_diffusion_cycles
100000
aggregation_fraction
0.d0
sticking_probability
1.d0
boundary_type
0
dipole_spacing
.2d0
lateral_dipole_spacing
.2d0
n_configurations
1
print_scattering_matrix
f
loop_variable
part_fv
0.05,0.5,.05
end_of_options
```

The output, for the first run, is given below.

```
************************************************
************************************************
 ppgf calculation results
 date, time:
20180718 110209.435
 input file:
```

```
pwpp.inp
**************************************************
 input variables for run     1
 max_iterations,solution_eps
     10000  0.10000E-03
 incident alpha,beta(deg)
  0.00000E+00  0.00000E+00
 ri medium calculated automatically
 absorption factor
  0.10000E-01
 ndx_width, ndy_width,,dz,dlat,k_cut
  500  500  0.20000E+00  0.20000E+00  0.25000E+01
 number components
    1
  comp   shape rad     sigma   ary     arz    fv           re(m)        im(m)
     1     1   1.300   0.186   1.000   1.000  0.50000E-01  0.13100E+01  0.00000E+00
 particle shell properties
  thickness    re(m)        im(m)
    0.00  0.10000E+01  0.00000E+00
 particle positions generated automatically
 maximum diffusion cycles, sticking, rotation probabilities:
    100000  0.10000E+01  0.00000E+00
 binder refractive index
  0.10000E+01  0.00000E+00
 front medium refractive index
  0.10000E+01  0.00000E+00
 back medium refractive index
  0.10000E+01  0.00000E+00
 natural boundary (particles intact)
 ndsx, ndsy, lateral spacing, sample x, y width
    80    80  0.20000E+00  0.10000E+03  0.10000E+03
 nd thickness, vertical spacing, target thickness
    50  0.20000E+00  0.10000E+02
**************************************************
 calculation results for run
           1
```

Output is overwritten to the file following each solution for a configuration.

```
 current, total configurations
     1     1
 number iterations, error, time per solution, total time
    13  0.63704E-04  0.50031E+02  0.50031E+02
```

Number of monomers is the initial number of sampled particles. Number of aggregates is the total number of unconnected particles. Even without aggregation simulations, some of the particles initially placed in the cell will be touching, and are hence counted together as an aggregated particle.

```
number monomer particles, number aggregates,volume fraction
  543   499  0.49271E-01
DGF refractive index
 0.10175E+01  0.10015E-01
effective refractive index
 0.10575E+01  0.96674E-02
dimensionless extinction coefficient, optical thickness
 0.58373E-02  0.58373E-01
```

`par-par` refers to hemispherical reflectance and transmittance for parallel polarized incidence and parallel polarized scattering, and likewise for `par-per`, `per-par`, etc. For a homogeneous medium (no particles) these quantities would correspond to the Fresnel reflection and transmission values for a slab of refractive index `ri_binder` backed on either side by media of `ri_front_surface` and `ri_back_surface`.

`dif` and `coh` refer to diffuse and coherent, and `unp` denotes unpolarized incidence and scattering. Coherent is the reflection and transmission in the direction of the incident beam; the coherent transmission is the line-of-sight transmission (along the incident direction) through the slab, and optical thickness is -ln(`unp coh t`). `unp abs` is the absorptance of the layer, as calculated from energy conservation. It is non-zero here even though the medium is nonabsorbing: this is a numerical truncation error.

```
 par-par r    per-par r    par-par t   per-par t
  0.42178E-02  0.12221E-01  0.95507E+00  0.28030E-01
 par-per r    per-per r    par-per t   per-per t
  0.41570E-02  0.11025E-01  0.12965E-01  0.97126E+00
 unp dif r    unp dif t    unp coh r    unp coh t    unp abs
  0.15776E-01  0.40360E-01  0.34203E-04  0.94330E+00  0.53129E-03
************************************************
```

References

[1] D. W. Mackowski, B. Ramezan Pour, A plane wave model for direct simulation of reflection and transmission by discretely inhomogeneous plane parallel media, J. Quant. Spectrosc. Radiat. Transfer in press, available online.

[2] D. W. Mackowski, Van de Hulst Essay: The dda, the rte, and the computation of scattering by plane parallel layers of particles, J. Quant. Spectrosc. Radiat. Transfer 189 (2017) 43–59.