HELIOS: A surface integral equation software for light scattering in homogeneous, periodic, and stratified environments

Parmenion S. Mavrikakis[a,*], Olivier J. F. Martin[a]

[a]*Nanophotonics and Metrology Laboratory (NAM), École Polytechnique Fédérale de Lausanne (EPFL), CH-1015 Lausanne, Switzerland*

Abstract

We present HELIOS (HomogEneous and Layered medIa Optical Scattering), an open-source surface integral equation (SIE) software designed for modeling light scattering by particles embedded in homogeneous or layered media and periodic backgrounds. The code implements the Poggio-Miller-Chang-Harrington-Wu-Tsai (PMCHWT) formulation that has demonstrated exceptional reliability in solving scattering problems with penetrable objects. Domain boundaries are discretized using triangular meshes, upon which the electric and magnetic surface current densities are expanded using the Rao-Wilton-Glisson (RWG) basis functions. For periodic structures, such as photonic crystals and metasurfaces, HELIOS employs Ewald's transformation to efficiently evaluate the infinite series associated with 2D lattices. Regarding stratified media, the code utilizes a matrix-friendly approach for the layered media Green's tensor, computing Sommerfeld integrals and accelerating calculations through a tabulation-interpolation scheme. The source code is implemented in C++, while a Python interface manages the workflow, including simulation setup, solver run, and post-processing. The accuracy and versatility of HELIOS are demonstrated through various examples that cover all its functionalities.

PROGRAM SUMMARY

Program Title: HELIOS
CPC Library link to program files: (to be added by Technical Editor)
Developer's repository link: https://github.com/mavrikak/Helios
Licensing provisions: GNU General Public License Version 3
Programming language: C++ and Python
Nature of problem: The program simulates electromagnetic scattering by nanostructures embedded in homogeneous, stratified or periodic backgrounds. It models arbitrary 3D geometries, ranging from isolated scatterers to infinite periodic structures. The code calculates key optical quantities, such as scattering, absorption, and extinction cross-sections, reflectance and transmittance spectra, and near-field distributions, which are essential for analyzing nanophotonic devices and experiments.
Solution method: HELIOS uses the SIE method with the PMCHWT formulation and scatterers are discretized via triangular meshes and RWG functions. For periodic lattices, the code applies Ewald's transformation to evaluate the periodic Green's tensor. For stratified media, it uses a matrix-friendly formulation and a tabulation-interpolation acceleration scheme.
Additional comments including restrictions and unusual features: All examples (Sections 3, 4, and 5) were executed on a standard desktop equipped with a 13th Gen Intel i7-13700 CPU (24 threads, 2.1 GHz) and 32 GB of RAM. The provided high-quality results involve some heavy simulations; however, coarser meshes and larger discretization steps for near-field points can be used to significantly decrease the required computational time and resources. The use of `tmux` is suggested to facilitate long-running simulations within persistent sessions, ensuring execution continues independently of the user's terminal connection. Finally, the HELIOS graphical user interface (GUI) is not presented in this paper, but detailed documentation can be found in the repository.

[*] Corresponding author.
E-mail address: parmenion.mavrikakis@epfl.ch

1. Introduction

Computational electromagnetics has become an indispensable tool for the investigation of many phenomena [1–8], in both academia and industry. In the field of nanophotonics, it enables the rigorous design and analysis of complex optical systems ranging from isolated to periodic nanostructures [9–35]. Among the available numerical techniques, the SIE method stands out for its efficiency and accuracy in modeling 3D scatterers [36]. Unlike volume discretization methods, such as the discrete dipole approximation (DDA) [37–39], the finite-difference time-domain (FDTD) [40–43], and the finite element method (FEM) [44–46], SIE formulations require meshing only the boundaries of scattering bodies. This dimensionality reduction significantly lowers the computational burden. Furthermore, the SIE method inherently satisfies the Sommerfeld radiation condition, hence it does not require the implementation of artificial absorbing boundary conditions and allows precise computation of far-field quantities and extreme near-field interactions [47, 48].

Despite these advantages, the implementation of a robust and versatile SIE solver poses significant challenges, particularly when dealing with penetrable objects and complex backgrounds. For penetrable scatterers in a homogeneous background, the PMCHWT formulation is widely regarded as the standard choice, since it provides stable and accurate solutions [48]. However, modern nanophotonic devices often consist of periodic arrays, such as photonic crystals and metasurfaces, or structures fabricated in stratified media. Modeling these environments requires specialized Green's functions that are far more complex than their free-space counterparts. For periodic structures, the slow convergence of the periodic dyadic Green's tensor necessitates the employment of acceleration techniques, such as Ewald's transformation [49]. Similarly, for scatterers embedded in stratified media, Green's tensor must account for the reflection and transmission at every interface. This involves the evaluation of Sommerfeld integrals, which are computationally expensive and prone to numerical singularities, requiring robust computation schemes to become practical.

While the theoretical foundations for these problems are well-established, there remains a lack of open-source software that integrates all these functionalities into a single framework. To bridge this gap, we present HELIOS, an open-source SIE solver developed for the rigorous simulation of light scattering by penetrable bodies in homogeneous and stratified backgrounds and by periodic structures.

HELIOS implements the PMCHWT formulation using RWG functions on triangular meshes. It features a modular architecture with a C++ core for intensive operations and a Python interface for simulation management, execution and post-processing. Additionally, it provides a GUI which facilitates the whole simulation process. Key capabilities include the modeling of infinite periodic 2D lattices (with 3D unit cells) via Ewald's summation and the efficient handling of multilayered backgrounds using a matrix-friendly formulation with accelerated Green's function evaluation.

The remainder of this paper is organized as follows. Section 2 outlines the software structure, installation, and workflow. Section 3 details the formulation for isolated scatterers in homogeneous backgrounds. Section 4 extends this to periodic structures, while Section 5 addresses the challenges of stratified media. Finally, a rich variety of examples are provided for every functionality of the software.

2. HELIOS setup and structure

HELIOS is designed for Linux environments. Windows users may utilize the Windows Subsystem for Linux (WSL) to establish a fully compatible runtime. The installation process requires the git version-control system. To begin, the HELIOS repository is cloned from GitHub:

```
$ sudo apt install git git-lfs && git lfs install
$ git clone https://github.com/mavrikak/Helios.git
```

The project directory contains the source code, build scripts, and example input data. To ensure cross-platform consistency, the code relies on a reproducible environment managed by Miniconda. Hence, the dependencies in `environment.yml` are installed and the `Helios` environment is activated:

```
$ cd Helios
$ conda env create -f environment.yml
$ conda activate Helios
```

This establishes the necessary packages, including numerical libraries for data handling and post-processing.

Before execution, the C++ core of HELIOS must be compiled along with its auxiliary C++ [50, 51] and Fortran [52–55] files. The build system employs a standard `make` workflow, creating separate directories for applications and compiled objects:

```
$ mkdir apps
$ mkdir build
$ make clean && make -j
```

Successful compilation yields executables in the `apps/` directory and compiled files in `build/`. Each simulation is organized structurally in its own subdirectory under `sim_data/<SimName>/` and requires two inputs: (i) a `config.txt` file that specifies the simulation parameters (characteristic examples are provided in the `sim_data/` directory of HELIOS), and (ii) a `.mphtxt` or `.msh` mesh file generated with COMSOL or Gmsh, respectively.

Documentation may be rebuilt locally to reflect source code modifications or functional extensions. The documentation build system, located in `docs/`, utilizes Doxygen to generate HTML and PDF references:

```
$ cd docs
$ bash build.sh
```

Overall, completing these steps ensures a fully configured environment for compiling the project's code and running electromagnetic simulations. In Table 1 the main classes of the source C++ code are presented. A very detailed presentation of its structure and files is provided in the Doxygen documentation. Finally, the project's main structure is presented below:

Table 1: List of selected C++ classes. They appear categorized according to their main contribution to the source code.

Mesh and Geometry	**Simulation domains**	**Green's functions**	**SIE formulations**	**Incident fields**	**Quadrature and Singularity handling**	**Driver and Jobs**
Bsp3D	Domain	GreenF	SIEForm	Dipole	GaussQuad ⟨Type⟩	CLParser
RWGFun	DomainHom3D	GreenFHom3D	SIEFormPMCHW	Gaussian	SingCancellation	CLOption ⟨T⟩
SurfaceMesh	DomainHom3DPer	GreenFHom3DPer		IncidentField	SingSub	CLStringOption
Triangle	DomainLayered3D	GreenFHom3DPer2D		PlaneWave	TriQuadPol	CLFlag
	LayeredMediaUtils	GreenFLayered3D				JobParser
		LookUpTable				SimJob
		LookUpTableBin				
		SymmetryOperation				
		Translation				
		SommerfeldIntegrator				

```
Helios/
├── docs/ ........................ C++ code documentation and GUI user's guide
├── include/ ..................... header and blitz++ files
├── materials/ ................... materials epsilon tables
├── pytools/ ..................... project tools
│   ├── drude_model.py ........... Drude model
│   ├── jobwriter.py ............. creates jobfiles
│   ├── meshconvert.py ........... creates .mesh
│   ├── plot_mesh.py ............. plots mesh
│   └── visualization.py ......... plots results
├── sim_data/ .................... user data inputs
│   └── ⟨SimName⟩/ ............... simulation data folder
│       ├── config.txt ........... configuration file
│       ├── mesh.mphtxt .......... COMSOL mesh file
│       └── mesh.msh ............. Gmsh mesh file
└── sim_res/ ..................... simulations outputs
    └── ⟨SimName⟩/ ............... results folder
```

```
|
|   |__ jobs/ ........................................ job files
|   |__ logs/ ........................................ log files
|   |__ out/ ......................................... data outputs
|   |   |__ csc/ ..................................... cross-sections
|   |   |__ fields/ .................................. field files
|   |   |__ media/ ................................... PNG plots
|   |__ points/ ...................................... points files
|   |__ lambdalist.txt ............................... wavelength list
|   |__ mesh.mesh .................................... converted mesh file
|__ src/ ............................................. C++ source code
|__ third_party/ ..................................... Fortran files
|   |__ amos/ ........................................ amos library files
|   |__ legacy_f/ .................................... iterative solver
|   |__ quadpack/ .................................... quadpack routines
|   |__ regridpack/ .................................. regridpack routines
|   |__ erfc.bin ..................................... erfc lookup table
|__ dist_run_helios.py ............................... parallel job launcher
|__ environment.yml .................................. conda environment
|__ helios_gui.py .................................... GUI launcher
|__ makefile ......................................... compilation instructions
|__ README.md ........................................ README file
|__ run_sie.py ....................................... code run script
```

3. Isolated scatterers in homogeneous backgrounds

In this section, we present the theoretical formulation and numerical implementation for modeling isolated scatterers in homogeneous media. Following the theoretical overview, we provide two step-by-step examples demonstrating how to set up, run, and analyze simulations. For the rest of this work, time-harmonic fields with a $\exp(-j\omega t)$ time dependence are considered.

3.1. Theory

Let us focus on the two-region scattering problem illustrated in Fig. 1(a), which consists of a homogeneous body embedded in a homogeneous background. Each region Ω_i is characterized by its electric permittivity and magnetic permeability pair (ε_i, μ_i). The fields satisfy the standard vector Helmholtz equations,

$$\nabla \times \nabla \times \mathbf{E}_i(\mathbf{r}) - k_i^2 \mathbf{E}_i(\mathbf{r}) = j\omega\mu_i \mathbf{J}_i(\mathbf{r}) \,, \tag{1}$$

$$\nabla \times \nabla \times \mathbf{H}_i(\mathbf{r}) - k_i^2 \mathbf{H}_i(\mathbf{r}) = \nabla \times \mathbf{J}_i(\mathbf{r}) \,, \tag{2}$$

where ω denotes the angular frequency, $k_i = \omega\sqrt{\varepsilon_i \mu_i}$ symbolizes the wavenumber in Ω_i and $\mathbf{J}_i(\mathbf{r})$ is the volume current distribution in Ω_i. By introducing the dyadic Green's tensor

$$\overline{\mathbf{G}}\,(\mathbf{r}, \mathbf{r}') = \left(\overline{\mathbf{1}} + \frac{\nabla\nabla}{k^2}\right) \frac{\exp(jk\,|\mathbf{r} - \mathbf{r}'|)}{4\pi\,|\mathbf{r} - \mathbf{r}'|} \,, \tag{3}$$

where $g\,(\mathbf{r}, \mathbf{r}') = \exp(jk\,|\mathbf{r} - \mathbf{r}'|)/4\pi\,|\mathbf{r} - \mathbf{r}'|$ is the scalar Green's function in a homogeneous medium with wavenumber k, and the following equation [56] in each domain Ω_i

$$\nabla \times \nabla \times \overline{\mathbf{G}}_i\,(\mathbf{r}, \mathbf{r}') - k_i^2 \overline{\mathbf{G}}_i\,(\mathbf{r}, \mathbf{r}') = \overline{\mathbf{1}}\delta\,(\mathbf{r} - \mathbf{r}') \,, \tag{4}$$

where we have Kronecker's delta and $\overline{\mathbf{1}}\Big|_{\xi_1\xi_2} = \delta_{\xi_1\xi_2}$, one obtains the familiar electric field integral equation (EFIE) and magnetic field integral equation (MFIE) [47]. For each domain, they read respectively [36, 48]

$$j\omega\mu_i\, \mathcal{D}_i\left\{\mathbf{J}_{s,i}\right\}(\mathbf{r})\Big|_{\tan} + \mathcal{K}_i\left\{\mathbf{M}_{s,i}\right\}(\mathbf{r})\Big|_{\tan} = \mathbf{E}_i^{\mathrm{inc}}\,(\mathbf{r})\Big|_{\tan} \,, \tag{5}$$

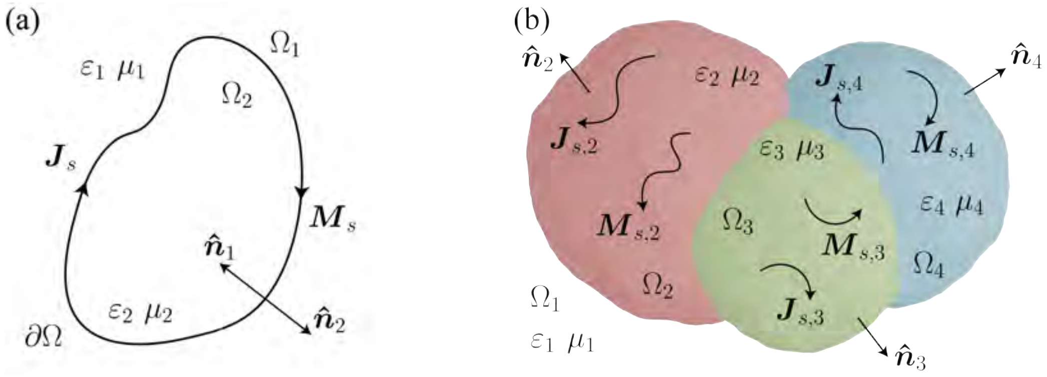


Figure 1: Penetrable scatterers in a homogeneous medium. (a) Single particle scenario (domain Ω_2), and (b) multiple attached particles case (domains Ω_2, Ω_3, Ω_4). In both cases the homogeneous background domain is Ω_1.

$$j\omega\varepsilon_i\ \mathcal{D}_i\left\{\mathbf{M}_{s,i}\right\}(\mathbf{r})\Big|_{\tan} - \mathcal{K}_i\left\{\mathbf{J}_{s,i}\right\}(\mathbf{r})\Big|_{\tan} = \mathbf{H}_i^{\text{inc}}(\mathbf{r})\Big|_{\tan}\ , \tag{6}$$

where "tan" stands for tangential, $\mathbf{J}_{s,i}(\mathbf{r})$ is the electric surface current density and $\mathbf{M}_{s,i}(\mathbf{r})$ is the magnetic surface current density defined on the boundary $\partial\Omega_i$ of domain Ω_i. The depicted linear integral operators $\mathcal{D}_i\{\mathcal{X}\}(\mathbf{r})$ and $\mathcal{K}_i\{\mathcal{X}\}(\mathbf{r})$ can be expressed as [36, 48]

$$\mathcal{D}_i\{\mathcal{X}\}(\mathbf{r}) = \int_{\partial\Omega_i} \mathrm{d}S'\overline{\mathbf{G}}_i(\mathbf{r},\mathbf{r}')\cdot\mathcal{X}(\mathbf{r}')\ , \tag{7}$$

$$\mathcal{K}_i\{\mathcal{X}\}(\mathbf{r}) = \int_{\partial\Omega_i} \mathrm{d}S'\left[\nabla'\times\overline{\mathbf{G}}_i(\mathbf{r},\mathbf{r}')\right]\cdot\mathcal{X}(\mathbf{r}')\ . \tag{8}$$

By discretizing the scatterer's surface with a triangular mesh $\mathcal{M}$ consisting of N_e edges, expanding the unknown surface current densities of Eqs. (5) and (6) with the RWG basis functions [36, 57, 58] as

$$\mathbf{J}_{s,i}(\mathbf{r}) = \sum_{\ell=1}^{N_e}\alpha_\ell\mathbf{f}_{\ell,i}(\mathbf{r})\ , \tag{9}$$

$$\mathbf{M}_{s,i}(\mathbf{r}) = \sum_{\ell=1}^{N_e}\beta_\ell\mathbf{f}_{\ell,i}(\mathbf{r})\ , \tag{10}$$

and applying the Galerkin testing procedure [59, chap. 3], the EFIE and MFIE become respectively

$$\left[j\omega\mu_i\mathbf{D}_i \quad \mathbf{K}_i\right]\cdot\begin{bmatrix}\boldsymbol{\alpha}\\ \boldsymbol{\beta}\end{bmatrix} = \mathbf{q}_i^E\ , \tag{11}$$

$$\left[-\mathbf{K}_i \quad j\omega\varepsilon_i\mathbf{D}_i\right]\cdot\begin{bmatrix}\boldsymbol{\alpha}\\ \boldsymbol{\beta}\end{bmatrix} = \mathbf{q}_i^H\ , \tag{12}$$

where $\mathbf{D}_i$, $\mathbf{K}_i$ are square matrices of side N_e, $\boldsymbol{\alpha}, \boldsymbol{\beta}$ are the column vectors that contain the expansion coefficients, and $\mathbf{q}_i^E$, $\mathbf{q}_i^H$ are the electric and magnetic incident field column vectors. The elements of the aforementioned quantities are written as

$$(D_i)_{\xi_1\xi_2} = \int_{\partial\Omega_i}\mathrm{d}S\,\mathbf{f}_{\xi_1,i}(\mathbf{r})\cdot\int_{\partial\Omega_i}\mathrm{d}S'\overline{\mathbf{G}}_i(\mathbf{r},\mathbf{r}')\cdot\mathbf{f}_{\xi_2,i}(\mathbf{r}')\ , \tag{13}$$

$$(K_i)_{\xi_1\xi_2} = \int_{\partial\Omega_i}\mathrm{d}S\,\mathbf{f}_{\xi_1,i}(\mathbf{r})\cdot\int_{\partial\Omega_i}\mathrm{d}S'\left[\nabla'\times\overline{\mathbf{G}}_i(\mathbf{r},\mathbf{r}')\right]\cdot\mathbf{f}_{\xi_2,i}(\mathbf{r}')\ , \tag{14}$$

$$(q_i^E)_{\xi_1} = \int_{\partial\Omega_i}\mathrm{d}S\,\mathbf{f}_{\xi_1,i}(\mathbf{r})\cdot\mathbf{E}_i^{\text{inc}}(\mathbf{r})\,, \tag{15}$$

$$(q_i^H)_{\xi_1} = \int_{\partial\Omega_i} \mathrm{d}S\, \mathbf{f}_{\xi_1,i}(\mathbf{r}) \cdot \mathbf{H}_i^{\mathrm{inc}}(\mathbf{r}) \,, \tag{16}$$

$$\boldsymbol{\alpha} = [\alpha_1 \quad ... \quad \alpha_{N_e}]^T \,, \tag{17}$$

$$\boldsymbol{\beta} = [\beta_1 \quad ... \quad \beta_{N_e}]^T \,. \tag{18}$$

However, both the EFIE and MFIE suffer from internal resonances, which produce inaccurate results [60, 61]. To overcome this difficulty, the PMCHWT scheme [62–64] combines the equations from all regions into a single coupled system [47]

$$\sum_i \begin{bmatrix} j\omega\mu_i\mathbf{D}_i & \mathbf{K}_i \\ -\mathbf{K}_i & j\omega\varepsilon_i\mathbf{D}_i \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{\alpha} \\ \boldsymbol{\beta} \end{bmatrix} = \sum_i \begin{bmatrix} \mathbf{q}_i^E \\ \mathbf{q}_i^H \end{bmatrix} \,, \tag{19}$$

thus providing a stable and reliable formulation for scattering problems with multiple penetrable scattering bodies in homogeneous backgrounds, as shown in Fig. 1(b).

Finally, the singularities of the Green's function that occur during numerical integration are handled using the singularity subtraction technique, as described in [65]. This method requires the Taylor series expansion of Green's function within domain Ω_i, which can then be decomposed into smooth and singular components [66] as follows

$$g_i\left(\mathbf{r}, \mathbf{r}'\right) = g_i^{sm}\left(\mathbf{r}, \mathbf{r}'\right) + g_i^{sn}\left(\mathbf{r}, \mathbf{r}'\right) \,, \tag{20}$$

where $g_i^{sm}(\mathbf{r}, \mathbf{r}')$ is suitable for numerical integration, while $g_i^{sn}(\mathbf{r}, \mathbf{r}')$ is evaluated semi-analytically using closed-form expressions. Detailed formulas for these calculations with RWG functions are provided in [65].

3.2. Simulating nanoparticles in homogeneous backgrounds

To illustrate the core workflow of HELIOS and establish a performance baseline, we present two scenarios of light scattering in homogeneous backgrounds (`--mode 0` is the corresponding command line argument). These configurations facilitate direct comparison with analytical solutions or well-established benchmarks, serving as an ideal validation of the solver's accuracy and post-processing capabilities. Throughout this paper, the following input arguments are used: (i) `-l 2` to solve the final system with the direct lower-upper (LU) factorization, `-th 0` to automatically utilize the maximum available threads on the host machine, and `-a` to increase the number of numerical integration points per mesh triangle [67].

3.2.1. Homogeneous gold sphere in vacuum

We first examine the scattering response of a spherical Au nanoparticle in free-space, with radius $R = 75$ nm. This benchmark system is routinely employed in nanophotonics and serves as a fundamental test of the program's accuracy [89]. The simulation is initialized using the `prepare` command of the `run_sie.py` script, which generates the results folder under `sim_res/Au_sphere_R_75nm/` with all the required data:

```
$ python3 run_sie.py prepare Au_sphere_R_75nm --mode 0 --materials eps --spline
```

If an earlier version exists, it may be overwritten with:

```
$ python3 run_sie.py prepare Au_sphere_R_75nm --mode 0 --materials eps --spline --overwrite
```

Prior to execution, it is beneficial to inspect the generated mesh. This can be done interactively:

```
$ python3 pytools/plot_mesh.py --sim Au_sphere_R_75nm --show
```

or by saving an image, see Fig. 2(a), to the results directory:

```
$ mkdir sim_res/Au_sphere_R_75nm/out/media/
$ python3 pytools/plot_mesh.py --sim Au_sphere_R_75nm --save  \
    ./sim_res/Au_sphere_R_75nm/out/media/mesh.png
```

The `--show` and `--save` options are available for all visualizations. For the remainder of this work, we will focus on commands that save PNG images. Once the simulation's geometry and settings are validated, the SIE solver assembles and solves the final linear system:

```
$ python3 run_sie.py solve Au_sphere_R_75nm --mode 0 -l 2 -th 0 -a
```

The desired illumination conditions are defined in `config.txt`.

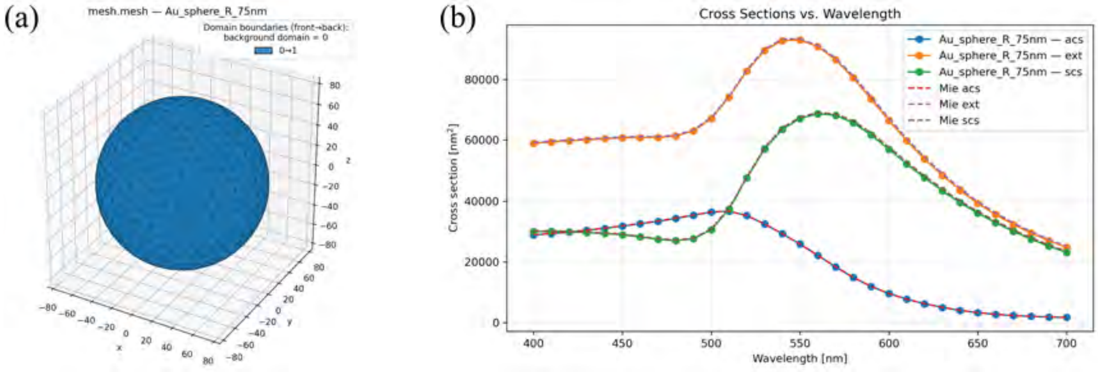


Figure 2: Au sphere (R = 75 nm) in free-space. (a) Triangular mesh, and (b) cross-section results comparison between the SIE method and Mie theory for an incident plane wave that propagates towards $+z$ and is x-polarized.

Key quantities of interest for spherical nanoparticles include the scattering (C_{scs}), absorption (C_{acs}), and extinction (C_{ext}) cross-sections. These can be illustrated in PNG format:

```
$ python3 pytools/visualization.py Au_sphere_R_75nm --root sim_res --plot-csc --subjob 1 --csc  \
    scs --csc acs --csc ext --save ./sim_res/Au_sphere_R_75nm/out/media/cross_sections.png
```

To validate against analytical solutions, HELIOS allows the overlay of corresponding Mie theory results using identical radii and tabulated material properties, as depicted in Fig. 2(b):

```
$ python3 pytools/visualization.py Au_sphere_R_75nm --root sim_res --plot-csc --subjob 1 --csc  \
    scs --csc acs --csc ext --mie-radius 75 --mie-material Au --mie-background eps=1,0  \
    --mie-spline --materials-root materials --save  \
    ./sim_res/Au_sphere_R_75nm/out/media/cross_sections_with_Mie.png
```

To visualize near-field, a set of sampling points is required. Here, the points file `points_xz.pos` is created for a 2D slice in the $y = 0$ plane, covering the region $x, z \in [-100, 100]$ nm with a step size of 0.5 nm:

```
$ python3 run_sie.py make-points Au_sphere_R_75nm xz --x -100 100 --z -100 100 --step 0.5 --y0=0  \
    -o points_xz.pos
```

Field values at these locations are computed during post-processing for the resonant wavelength of 560 nm:

```
$ python3 run_sie.py post Au_sphere_R_75nm -th 0 -a --lambdas 560 -p points/points_xz.pos
```

Finally, two near-field distributions at the sampling points are visualized in Fig. 3. The latter presents the total electric field intensity in the xz-plane for plane wave ($\mathbf{k} = k_z\hat{\mathbf{z}}$, with $k_z > 0$, and $\mathbf{E}^{\mathrm{inc}} = E_x^{inc}\hat{\mathbf{x}}$) and dipole ($x = 75$ nm, $z = 75$ nm, $\theta = \pi/4$, $\varphi = \pi$) illuminations:

```
$ python3 pytools/visualization.py Au_sphere_R_75nm --root sim_res --points points_xz.pos  \
    --field e --part tot --quantity total --function intensity --mode 2d --subjob 1 --lambda  \
    560 --save ./sim_res/Au_sphere_R_75nm/out/media/PW_560nm_Etot_intensity.png
$ python3 pytools/visualization.py Au_sphere_R_75nm --root sim_res --points points_xz.pos  \
    --field e --part tot --quantity total --function intensity --mode 2d --subjob 2 --log  \
    --lambda 560 --save ./sim_res/Au_sphere_R_75nm/out/media/dip_560nm_Etot_intensity.png
```

The dipolar source implementation can be useful when studying fluorescence since it can model fluorescent molecules [68, 69]. Treating a fluorophore as a dipole helps explain key phenomena such as polarization effects, emission patterns, energy transfer, and how fluorescence depends on molecular orientation relative to the excitation light.

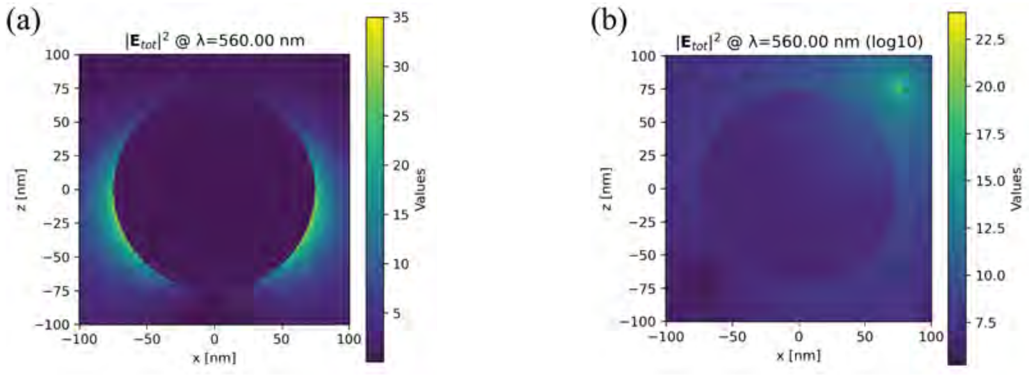


Figure 3: Near-field for (a) an incident plane wave that propagates towards $+z$ and is x-polarized, and (b) an in-plane source dipole at (x = 75 nm, z = 75 nm) with direction ($\theta = \pi/4$, $\varphi = \pi$). The dipole orientation is defined in the standard spherical coordinates.

3.2.2. φ-type Janus ring

Next, we consider an inhomogeneous toroidal nanoparticle wherein the permittivity varies along the azimuthal direction φ. In contrast to the homogeneous Au sphere, the Janus ring [70–72] demonstrates the flexibility of HELIOS in handling different materials and discretizing junctions within the SIE framework. The simulation setup follows the standard workflow. The results directory is first generated using the prepare command:

```
$ python3 run_sie.py prepare phi_type_Janus_ring --mode 0 --materials eps --spline
```

A visual inspection of the mesh is recommended to verify that the Janus ring geometry and the azimuthally varying material patches are modeled correctly:

```
$ mkdir sim_res/phi_type_Janus_ring/out/media/
$ python3 pytools/plot_mesh.py --sim phi_type_Janus_ring --save  \
    ./sim_res/phi_type_Janus_ring/out/media/mesh.png
```

In Fig. 4(a), the crystalline Silicon (cSi) section ($\varphi \in [-\pi/4, \pi/4]$) is shown in brown, while the Au domain is depicted in blue. Additionally, the interfaces between these domains are highlighted in light blue. The minor radius is R_1 = 25 nm, while the major radius is R_2 = 75 nm. Upon geometrical confirmation, the system is assembled and solved for an incident plane wave that propagates towards $+z$ and is x-polarized, and a dipole at the origin with direction ($\theta = \pi/2$, $\varphi = 0$):

```
$ python3 run_sie.py solve phi_type_Janus_ring --mode 0 -l 2 -th 0 -a
```

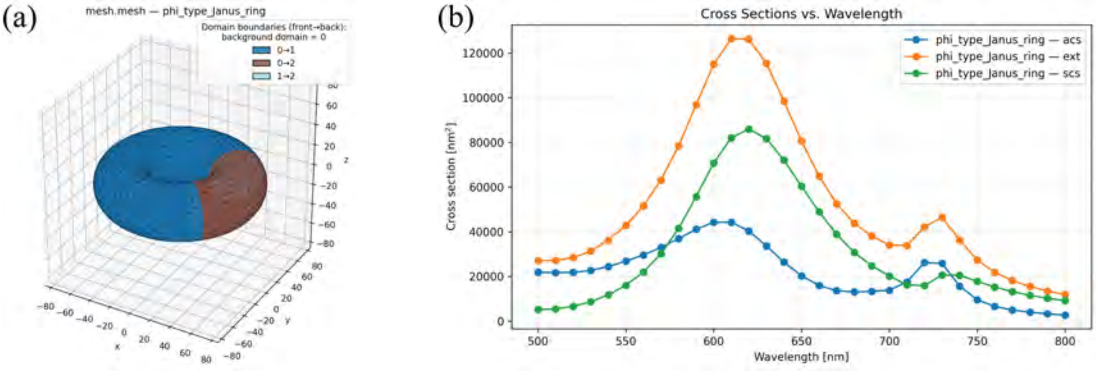


Figure 4: Au-cSi Janus ring in free-space. (a) Triangular mesh, and (b) cross-section results for an incident plane wave that propagates towards $+z$ and is x-polarized.

To examine the optical response of the Janus ring for plane wave illumination, C_{scs}, C_{acs} and C_{ext} are visualized in Fig. 4(b):

```
$ python3 pytools/visualization.py phi_type_Janus_ring --root sim_res --plot-csc --subjob 1  \
    --csc scs --csc acs --csc ext --save  \
    ./sim_res/phi_type_Janus_ring/out/media/cross_sections.png
```

For a detailed inspection of the near-field patterns, a grid of sampling points is generated in the $z = 0$ plane, covering the domain $x \in [-100,\ 100]$ nm and $y \in [-105,\ 105]$ nm with a step size of 0.5 nm:

```
$ python3 run_sie.py make-points phi_type_Janus_ring xy --x -100 100 --y -105 105 --step 0.5  \
    --z0=0 -o points_xy.pos
```

Post-processing is then performed at the resonance, $\lambda = 620$ nm:

```
$ python3 run_sie.py post phi_type_Janus_ring -th 0 -a --lambdas 620 -p points/points_xy.pos
```

Finally, two near-field illustrations are presented for different scenarios. More specifically, the total electric field intensity in the xy-plane at $\lambda = 620$ nm for an incident plane wave and the intensity of its x-component for dipole illumination are obtained as shown in Fig. 5:

```
$ python3 pytools/visualization.py phi_type_Janus_ring --root sim_res --points points_xy.pos  \
    --field e --part tot --quantity total --function intensity --mode 2d --subjob 1 --lambda  \
    620 --save ./sim_res/phi_type_Janus_ring/out/media/PW_620nm_Etot_intensity.png
$ python3 pytools/visualization.py phi_type_Janus_ring --root sim_res --points points_xy.pos  \
    --field e --part tot --quantity x --function intensity --mode 2d --subjob 3 --log --lambda  \
    620 --save ./sim_res/phi_type_Janus_ring/out/media/dipx_620nm_Ex_intensity.png
```

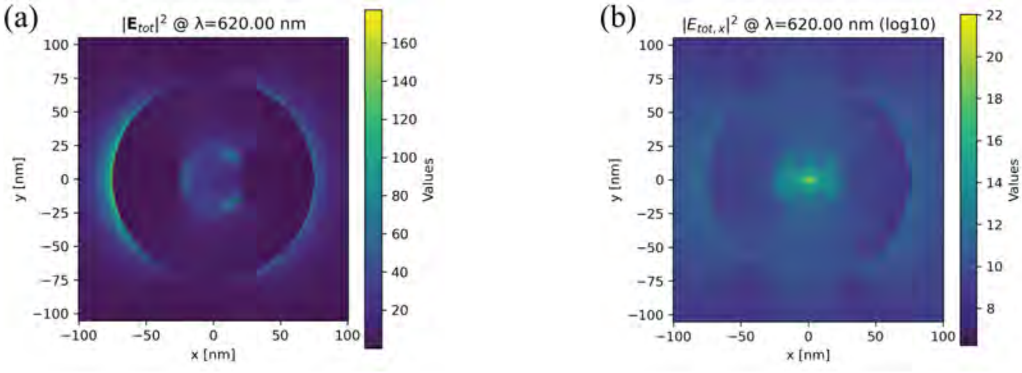


Figure 5: Near-field for (a) an incident plane wave that propagates towards $+z$ and is x-polarized, and (b) a dipole at the origin with direction $(\theta = \pi/2,\ \varphi = 0)$.

This example highlights the ability of HELIOS to simulate complex inhomogeneous particles, while maintaining robust numerical accuracy and providing clear physical insights.

4. Periodic structures in homogeneous backgrounds

This section extends the previous formulation to treat periodic structures embedded in homogeneous media. We first outline the necessary theoretical adaptations; specifically, the application of Floquet-periodic boundary conditions and the use of Ewald's transformation for Green's tensor. Since Ewald's summation relies heavily on the complementary error function, the solver utilizes a precomputed binary lookup table (`erfc.bin`) for acceleration. This file is provided with the source code; however, we also include a dedicated utility to regenerate it locally if any compatibility issues are encountered. Hence, the user can make a new `erfc.bin` with desired maximum and increment values:

```
$ make table
$ python run_sie.py make-table erfc.bin --maxRe 10.0 --maxIm 10.0 --incRe 0.1 --incIm 0.1
```

Subsequently, we present two numerical examples, i.e. a photonic crystal and a fishnet structure, to illustrate the handling of periodic lattices and the extraction of scattering parameters.

4.1. Theory

The modeling of periodic structures relies on the definition of a 2D lattice. In the present framework, the lattice possesses translational symmetry along the x and y axes. A translation vector $\mathbf{t}$ within this lattice is expressed as a linear combination $\mathbf{t} = c_1\boldsymbol{\alpha}_1 + c_2\boldsymbol{\alpha}_2$, where $c_1, c_2 \in \mathbb{Z}$ and $\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2$ denote the primitive lattice vectors. The fundamental repeating unit of the lattice is the unit cell, labeled as V in Fig. 6. Additionally, the irreducible representations of the translation group are defined by wavevectors $\mathbf{k}$ situated within the first Brillouin zone [73], while the 1D spaces forming the bases of these representations consist of Bloch functions that satisfy the Floquet-periodic boundary conditions

$$\mathbf{U}_{\mathbf{k}}(\mathbf{r} - \mathbf{t}) = \mathbf{U}_{\mathbf{k}}(\mathbf{r}) \exp(-j\mathbf{k} \cdot \mathbf{t}) \,. \tag{21}$$

The projections of the incident electric and magnetic fields onto this space can be written as $\mathbf{E}^{\text{inc}}_{i,\mathbf{k}}$ and $\mathbf{H}^{\text{inc}}_{i,\mathbf{k}}$, respectively. Should the volume current distributions in Eqs. (1) and (2) not be Floquet-periodic, the incident fields $\mathbf{E}^{\text{inc}}_{i}$ and $\mathbf{H}^{\text{inc}}_{i}$ may be expressed as linear combinations of Floquet-periodic components $\mathbf{E}^{\text{inc}}_{i,\mathbf{k}}$ and $\mathbf{H}^{\text{inc}}_{i,\mathbf{k}}$.

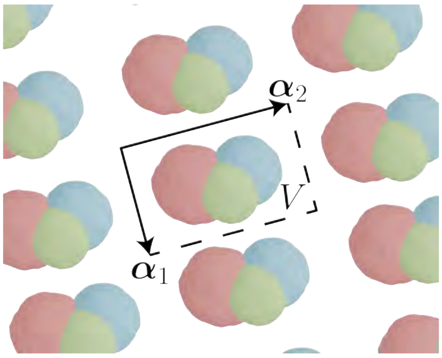


Figure 6: Multiple homogeneous scatterers in a unit cell of a periodic structure embedded in a homogeneous background.

The evaluation of the EFIE and MFIE is restricted to the boundaries lying within the volume V of the unit cell. Hence, for periodic structures Eqs. (5) and (6) can be rewritten as

$$j\omega\mu_i \; \mathcal{D}_{i,\mathbf{k}} \left\{\mathbf{J}_{s,i,\mathbf{k}}\right\}(\mathbf{r})\Big|_{\tan} + \mathcal{K}_{i,\mathbf{k}} \left\{\mathbf{M}_{s,i,\mathbf{k}}\right\}(\mathbf{r})\Big|_{\tan} = \mathbf{E}^{\text{inc}}_{i,\mathbf{k}}(\mathbf{r})\Big|_{\tan} \,, \tag{22}$$

$$j\omega\varepsilon_i \; \mathcal{D}_{i,\mathbf{k}} \left\{\mathbf{M}_{s,i,\mathbf{k}}\right\}(\mathbf{r})\Big|_{\tan} - \mathcal{K}_{i,\mathbf{k}} \left\{\mathbf{J}_{s,i,\mathbf{k}}\right\}(\mathbf{r})\Big|_{\tan} = \mathbf{H}^{\text{inc}}_{i,\mathbf{k}}(\mathbf{r})\Big|_{\tan} \,. \tag{23}$$

Each Floquet component of the surface current densities, $\mathbf{J}_{s,i,\mathbf{k}}$ and $\mathbf{M}_{s,i,\mathbf{k}}$, is solved independently. The dyadic pseudo-periodic Green's tensor is defined as

$$\overline{\mathbf{G}}_{i,\mathbf{k}}(\mathbf{r}, \mathbf{r}') = \sum_{\mathbf{t}} \exp(j\mathbf{k} \cdot \mathbf{t}) \overline{\mathbf{G}}_i(\mathbf{r} - \mathbf{t}, \mathbf{r}') \,. \tag{24}$$

Expanding the right-hand side, we obtain

$$\overline{\mathbf{G}}_i(\mathbf{r}, \mathbf{r}') = \left(\overline{\mathbf{1}} + \frac{\nabla\nabla}{k_i^2}\right) \sum_{\mathbf{t}} \frac{\exp(jk_i \left|\mathbf{R}_{\mathbf{t}}\right| + j\mathbf{k} \cdot \mathbf{t})}{4\pi \left|\mathbf{R}_{\mathbf{t}}\right|} \,, \tag{25}$$

where $\mathbf{R}_{\mathbf{t}} = \mathbf{r} - \mathbf{r}' - \mathbf{t}$. Due to the slow convergence of this series, Ewald's transformation [74, 75] is employed to decompose it into two rapidly convergent sums,

$$g_{i,\mathbf{k}}(\mathbf{r}, \mathbf{r}') = g^{sp}_{i,\mathbf{k}}(\mathbf{r}, \mathbf{r}') + g^{rp}_{i,\mathbf{k}}(\mathbf{r}, \mathbf{r}') \,, \tag{26}$$

where $g_{i,\mathbf{k}}(\mathbf{r}, \mathbf{r}')$ represents the sum in Eq. (25), $g^{sp}_{i,\mathbf{k}}(\mathbf{r}, \mathbf{r}')$ is the spatial domain term, and $g^{rp}_{i,\mathbf{k}}(\mathbf{r}, \mathbf{r}')$ is the reciprocal space term. More specifically, the spatial domain term is given by

$$g^{sp}_{i,\mathbf{k}}(\mathbf{r}, \mathbf{r}') = \frac{1}{8\pi} \sum_{\mathbf{t}} \exp(j\mathbf{k} \cdot \mathbf{t}) \sum_{\pm} \frac{\exp(\pm jk_i|\mathbf{R}_{\mathbf{t}}|)}{|\mathbf{R}_{\mathbf{t}}|} \operatorname{erfc}\left(|\mathbf{R}_{\mathbf{t}}|f \pm \frac{jk_i}{2f}\right) . \tag{27}$$

The reciprocal space term involves the reciprocal lattice vectors $\boldsymbol{\beta}_1$ and $\boldsymbol{\beta}_2$, which can be expressed as $\boldsymbol{\beta}_1 = (2\pi/A^2)\boldsymbol{\alpha}_2 \times (\boldsymbol{\alpha}_1 \times \boldsymbol{\alpha}_2)$ and $\boldsymbol{\beta}_2 = (2\pi/A^2)\boldsymbol{\alpha}_1 \times (\boldsymbol{\alpha}_2 \times \boldsymbol{\alpha}_1)$, where $A = |\boldsymbol{\alpha}_1 \times \boldsymbol{\alpha}_2|$. By defining $\mathbf{u} = d_1\boldsymbol{\beta}_1 + d_2\boldsymbol{\beta}_2$ $(d_1, d_2 \in \mathbb{Z})$, one obtains

$$g^{rp}_{i,\mathbf{k}}(\mathbf{r}, \mathbf{r}') = \frac{1}{4A} \sum_{\mathbf{u}} \exp[j(\mathbf{k} - \mathbf{u}) \cdot (\mathbf{r} - \mathbf{r}')] \sum_{\pm} \frac{\exp(\pm\gamma_{i,\mathbf{k},\mathbf{u}} R_\perp)}{\gamma_{i,\mathbf{k},\mathbf{u}}} \operatorname{erfc}\left(\frac{\gamma_{i,\mathbf{k},\mathbf{u}}}{2f} \pm R_\perp f\right), \tag{28}$$

where $\gamma_{i,\mathbf{k},\mathbf{u}} = \sqrt{|\mathbf{k} - \mathbf{u}|^2 - k_i^2}$ and $R_\perp = (\mathbf{r} - \mathbf{r}') \cdot (\boldsymbol{\alpha}_1 \times \boldsymbol{\alpha}_2)/A$. The parameter f governs the convergence of both series; an optimal value is typically $f = \pi/\sqrt{A}$ as reported in [74].

From Eqs. (22) and (23), it is evident that applying the PMCHWT formulation yields a linear system analogous to Eq. (19), following the discretization of the unit cell's scatterer(s) with a triangular mesh $\mathcal{M}$ and the application of the Galerkin scheme with RWG functions. A critical implementation detail arises when elements of $\mathcal{M}$ extend to one or more pairs of opposite boundaries of the unit cell. In such cases, translational symmetry must be explicitly enforced to maintain current continuity. Thus, the Floquet-periodic boundary conditions impose

$$\mathbf{J}_{s,i,\mathbf{k}}(\mathbf{r} - \mathbf{t}) = \exp(-j\mathbf{k} \cdot \mathbf{t})\mathbf{J}_{s,i,\mathbf{k}}(\mathbf{r}), \tag{29}$$

$$\mathbf{M}_{s,i,\mathbf{k}}(\mathbf{r} - \mathbf{t}) = \exp(-j\mathbf{k} \cdot \mathbf{t})\mathbf{M}_{s,i,\mathbf{k}}(\mathbf{r}), \tag{30}$$

ensuring consistent flow across periodic boundaries. A detailed analysis is provided in [76].

4.2. *Simulating periodic structures in homogeneous backgrounds*

In this subsection, we demonstrate the simulation of periodic structures embedded in homogeneous backgrounds (`--mode 1`). The following examples illustrate the handling of infinite 2D periodic lattices and extraction of spectral quantities, such as reflectance and transmittance. We consider two distinct cases: a dielectric photonic crystal and a fishnet structure. Along with the standard solver arguments introduced previously (namely `-l`, `-th`, and `-a`), the flag `--etm` is utilized here to specify the number of terms in the evaluation of Green's function with Ewald's method. The only limitation is the code's singular behavior when the wavelength is equal to the period.

4.2.1. *Photonic crystal*

To demonstrate the capabilities of HELIOS with periodic structures, we first simulate a photonic crystal. The simulation environment is initialized with the `prepare` command:

```
$ python3 run_sie.py prepare photonic_crystal --mode 1 --materials eps --spline
```

Before launching the solver, the mesh of the unit cell is inspected. The visualization, as shown in Fig. 7(a), is saved to the results directory:

```
$ mkdir sim_res/photonic_crystal/out/media/
$ python3 pytools/plot_mesh.py --sim photonic_crystal --save  \
    ./sim_res/photonic_crystal/out/media/mesh.png
```

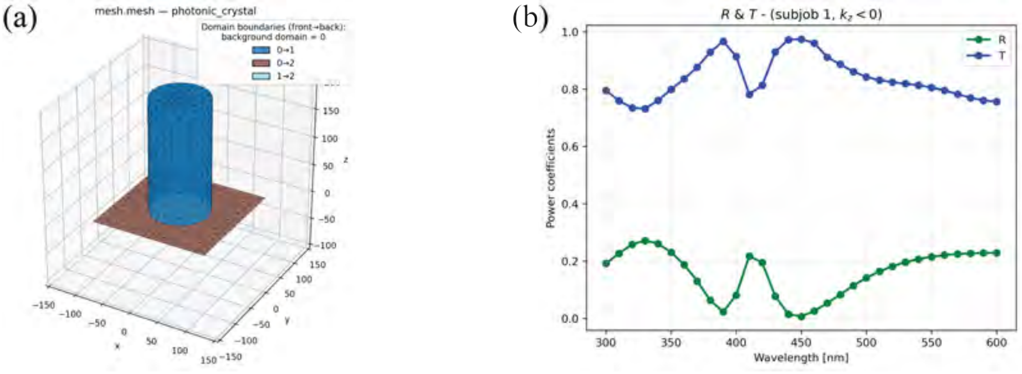


Figure 7: Photonic crystal example. (a) Unit cell's triangular mesh (the dimensions of the visualization cell exceed those of the unit cell), and (b) reflectance and transmittance as a function of wavelength for a plane wave illumination that propagates towards $-z$ and is x-polarized.

The periodic structure is an infinite square array of pillars with refractive index $n = 3.36$, diameter $D = 100$ nm, height $h = 200$ nm, and a substrate of the same material. Also, the lattice periods are $p_x = p_y = 200$ nm. Proper definition of the extra periodic simulation parameters `px py cx cy z` in `config.txt` is essential. Specifically, `px` and `py` define the lattice periodicity along the x and y axes, respectively, while `cx` and `cy` set the center of the unit cell in the xy-plane. The final parameter `z` establishes the vertical position of the mesh cutting plane below which everything is clipped, here set at $z = -100$ nm. Consequently, any mesh elements lying on or outside the boundaries defined by these parameters (i.e. `px`, `py`, and `z`) are clipped. In this example, although the original input mesh includes a finite slab spanning $x, y \in [-100, 100]$ nm and $z \in [-100, 0]$ nm, the lateral periodic boundaries and the cutoff plane trim the geometry, leaving only the top planar interface included in the computation, as in Fig. 7(a). Next, the problem is solved by using `--mode 1` to indicate a periodic simulation:

```
$ python3 run_sie.py solve photonic_crystal --mode 1 -l 2 -th 0 -a --etm 4
```

A primary quantity of interest for periodic structures is the spectral response, specifically the reflectance (R) and transmittance (T). Hence, we generate the points file `rt.pos`, defining the far-field measurement planes within the limits of the unit cell's xy-plane ($x, y \in [-100, 100]$ nm and $z \in \{-1000, 1200\}$ nm), and then compute the desired coefficients using the `post` command:

```
$ python3 run_sie.py make-points photonic_crystal xy --x -100 100 --y -100 100 --step 5  \
    --z0=-1000,1200 -o rt.pos
$ python3 run_sie.py post photonic_crystal --mode 1 -p points/rt.pos --etm 4 -a -th 0
```

The resulting R and T spectra, for an incident plane wave that propagates towards $-z$ and is x-polarized, can be plotted and saved, as displayed in Fig. 7(b), with the following command:

```
$ python pytools/visualization.py photonic_crystal --root sim_res --plot-rt --points rt.pos  \
    --subjob 1 --rt-prop auto --save ./sim_res/photonic_crystal/out/media/RTcoeffs.png
```

To visualize the near-field distribution, we define a slice in the $y = 0$ plane and perform post-processing at $\lambda = 410$ nm:

```
$ python3 run_sie.py make-points photonic_crystal xz --x -100 100 --z -100 390 --stepx 2 --stepz  \
    3.5 --y0=0 -o points_xz.pos
$ python3 run_sie.py post photonic_crystal --mode 1 -p points/points_xz.pos --etm 4 -a -th 0  \
    --lambdas 410
```

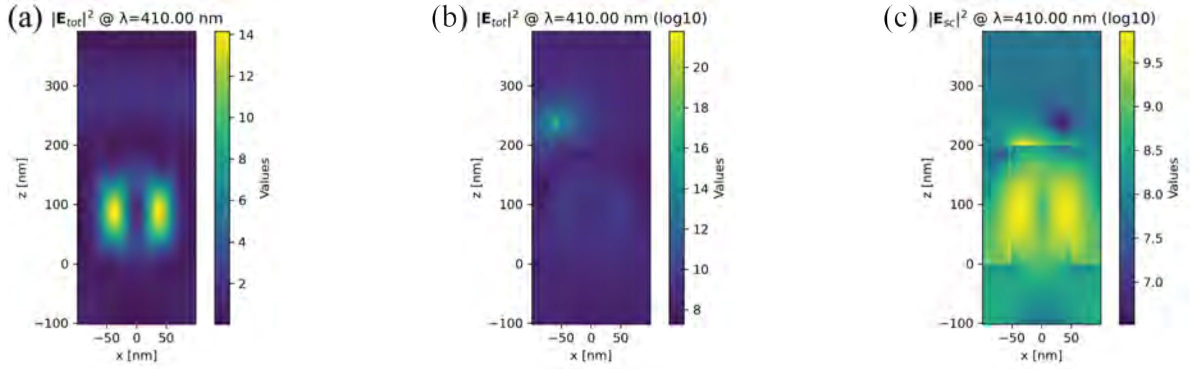


Figure 8: Near-field intensity distributions at $\lambda = 410$ nm for (a) an incident plane wave that propagates towards $-z$ and is x-polarized, and (b), (c) an in-plane dipole excitation ($x = -60$ nm, $z = 240$ nm, $\theta = 76°$, $\varphi = 0°$).

Finally, the total electric field intensity is visualized for both plane wave (`--subjob 1`) and dipole excitations (`--subjob 2`) as depicted in Figs. 8(a) and 8(b), respectively. For the dipole illumination scenario, the scattered electric field intensity is also shown in Fig. 8(c). All these PNG images are produced with the commands:

```
$ python3 pytools/visualization.py photonic_crystal --root sim_res --points points_xz.pos  \
    --field e --part tot --quantity total --function intensity --mode 2d --subjob 1 --lambda  \
    410 --save ./sim_res/photonic_crystal/out/media/PW_410nm_Etot_intensity.png
$ python3 pytools/visualization.py photonic_crystal --root sim_res --points points_xz.pos  \
    --field e --part tot --quantity total --function intensity --mode 2d --subjob 2 --log  \
    --lambda 410 --save ./sim_res/photonic_crystal/out/media/dip_410nm_Etot_intensity.png
$ python3 pytools/visualization.py photonic_crystal --root sim_res --points points_xz.pos  \
    --field e --part sc --quantity total --function intensity --mode 2d --subjob 2 --log  \
    --lambda 410 --save ./sim_res/photonic_crystal/out/media/dip_410nm_Esc_intensity.png
```

4.2.2. Fishnet structure

The second example considers a fishnet structure. The workflow is identical to the previous case. We begin by preparing the simulation and verifying the mesh, which is depicted in Fig. 9(a):

```
$ python3 run_sie.py prepare fishnet_structure --mode 1 --materials eps --spline
$ mkdir sim_res/fishnet_structure/out/media/
$ python3 pytools/plot_mesh.py --sim fishnet_structure --save  \
    ./sim_res/fishnet_structure/out/media/mesh.png
```

All the details of the system can be found in [76]. Notably, the resulting geometry is a continuous fishnet structure that extends over the entire space, since the scatterers occupy the entire unit cell. The solver is then executed using the periodic mode:

```
$ python3 run_sie.py solve fishnet_structure --mode 1 -l 2 -th 0 -a --etm 4
```

We subsequently compute the R and T coefficients to identify spectral features. The system is excited with a p-polarized plane wave that propagates towards the direction ($\theta = 6°$, $\varphi = 0°$). Results are visualized in Fig. 9(b):

```
$ python3 run_sie.py make-points fishnet_structure xy --x -150 150 --y -150 150 --step 5  \
    --z0=-1000,1000 -o rt.pos
$ python3 run_sie.py post fishnet_structure --mode 1 -p points/rt.pos --etm 4 -a -th 0
$ python3 pytools/visualization.py fishnet_structure --root sim_res --plot-rt --points rt.pos  \
    --subjob 1 --rt-prop auto --save ./sim_res/fishnet_structure/out/media/RTcoeffs.png
```

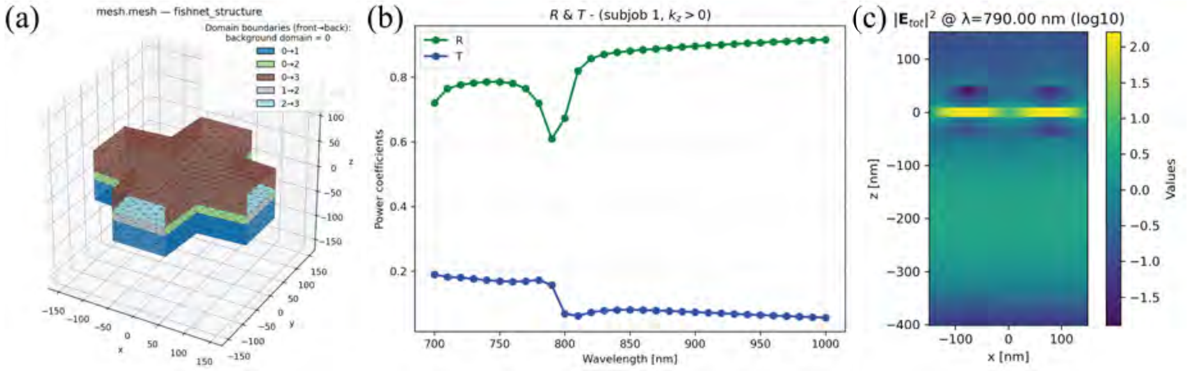


Figure 9: Fishnet structure example. (a) Unit cell's triangular mesh, (b) R and T spectra, and (c) total electric near-field intensity at $\lambda = 790$ nm for a p-polarized incident plane wave propagating towards ($\theta = 6°$, $\varphi = 0°$).

A near-field map is generated at $\lambda = 790$ nm. We define the sampling grid in the $y = 0$ plane with a step of 2.5 nm and run the post-processing:

```
$ python3 run_sie.py make-points fishnet_structure xz --x -150 150 --z -400 150 --step 2.5  \
    --y0=0 -o points_xz.pos
$ python3 run_sie.py post fishnet_structure --mode 1 -p points/points_xz.pos --etm 4 -a -th 0  \
    --lambdas 790
```

Finally, the total electric field intensity distribution for the selected plane wave illumination is visualized, as in Fig. 9(c):

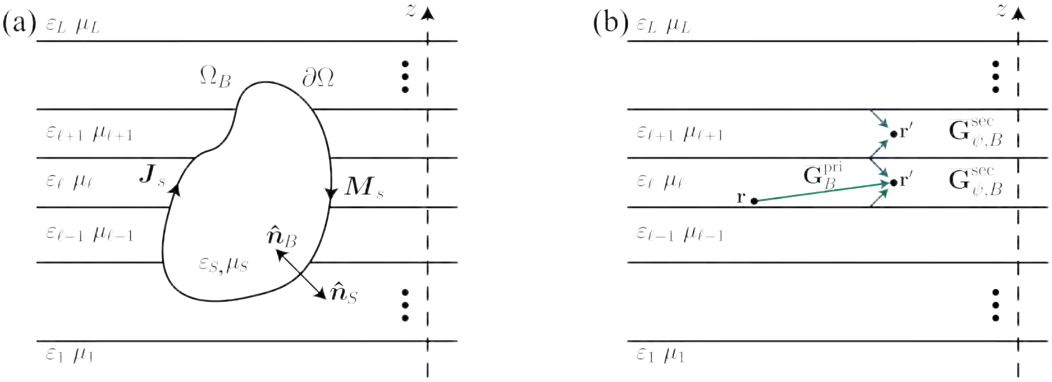
```
$ python3 pytools/visualization.py fishnet_structure --root sim_res --points points_xz.pos  \
    --field e --part tot --quantity total --function intensity --mode 2d --subjob 1 --log  \
    --lambda 790 --save ./sim_res/fishnet_structure/out/media/PW_790nm_Etot_intensity.png
```

5. Isolated scatterers in layered backgrounds

In this section, we address the modeling of isolated scatterers embedded in stratified media. This configuration corresponds to `--mode 2` in HELIOS. We first derive the necessary Green's tensor components for layered environments, including the evaluation of Sommerfeld integrals. Subsequently, we provide three examples, i.e. a sphere on a substrate, a hole in a metallic film, and a hole in a multilayer stack, to demonstrate the solver's capabilities.

5.1. Theory

As demonstrated in the preceding sections, the cornerstone of the SIE method is the dyadic Green's tensor. For stratified media, this tensor must incorporate the reflection and transmission effects occurring at each background interface. Let us consider the system depicted in Fig. 10(a), where a homogeneous scatterer (domain Ω_S) is embedded in a layered background (domain Ω_B). The dyadic Green's tensor describing Ω_B is expressed as [77]



(a)
$\varepsilon_L\ \mu_L$
z
Ω_B
$\partial\Omega$
$\varepsilon_{\ell+1}\ \mu_{\ell+1}$
$\varepsilon_\ell\ \mu_\ell$
$\boldsymbol{J}_s$
$\boldsymbol{M}_s$
$\varepsilon_{\ell-1}\ \mu_{\ell-1}$
$\hat{\boldsymbol{n}}_B$
ε_S, μ_S
$\hat{\boldsymbol{n}}_S$
$\varepsilon_1\ \mu_1$
(b)
$\varepsilon_L\ \mu_L$
z
$\varepsilon_{\ell+1}\ \mu_{\ell+1}$
$\mathbf{r}'$
$\mathbf{G}^{\mathrm{sec}}_{\psi,B}$
$\varepsilon_\ell\ \mu_\ell$
$\mathbf{G}^{\mathrm{pri}}_B$
$\mathbf{r}$
$\mathbf{r}'$
$\mathbf{G}^{\mathrm{sec}}_{\psi,B}$
$\varepsilon_{\ell-1}\ \mu_{\ell-1}$
$\varepsilon_1\ \mu_1$

Figure 10: Homogeneous scatterer in a stratified background. (a) Geometry for the scattering problem analysis, and (b) contributions of primary (green) and secondary (blue) dyadic Green's tensor terms in layered media.

$$\overline{\mathbf{G}}_{\psi,B}(\mathbf{r},\mathbf{r}') = \overline{\mathbf{G}}^{\mathrm{pri}}_{B}(\mathbf{r},\mathbf{r}') + \overline{\mathbf{G}}^{\mathrm{sec}}_{\psi,B}(\mathbf{r},\mathbf{r}'), \tag{31}$$

where $\overline{\mathbf{G}}^{\mathrm{pri}}_{B}(\mathbf{r},\mathbf{r}')$ represents the primary contribution, $\overline{\mathbf{G}}^{\mathrm{sec}}_{\psi,B}(\mathbf{r},\mathbf{r}')$ denotes the secondary contribution arising from the layered medium, and the subscript ψ indicates the field type ($\psi = E$ for electric and $\psi = H$ for magnetic). The primary term corresponds to the dyadic Green's tensor of a homogeneous medium, whereas the secondary term accounts for all scattering interactions between layers and is essential for satisfying the boundary conditions of Maxwell's equations at the interfaces. In the following analysis, the layer indices of the source and observation points are denoted by m and n, respectively. We focus on the electric-type scenario ($\psi = E$), noting that the magnetic-type case can be retrieved via the duality principle [78, chap. 7]. Hence, the primary term is given by

$$\overline{\mathbf{G}}^{\mathrm{pri}}_{B}(\mathbf{r},\mathbf{r}') = \overline{\mathbf{G}}^{\mathrm{pri}}_{E,B}(\mathbf{r},\mathbf{r}') = \overline{\mathbf{G}}^{\mathrm{pri}}_{H,B}(\mathbf{r},\mathbf{r}') = \left(\overline{\mathbf{1}} + \frac{\nabla\nabla}{k_m^2}\right)\frac{\exp(jk_m\,|\mathbf{r}-\mathbf{r}'|)}{4\pi\,|\mathbf{r}-\mathbf{r}'|}, \tag{32}$$

where $m = n$, since this term contributes exclusively to interactions within the same layer, as shown in Fig. 10(b). The secondary term admits a decomposition into transverse electric (TE) and transverse magnetic (TM) wave contributions [79], which reads as

$$\overline{\mathbf{G}}^{\mathrm{sec}}_{E,B}(\mathbf{r},\mathbf{r}') = \overline{\mathbf{G}}^{\mathrm{TE}}_{E,B}(\mathbf{r},\mathbf{r}') + \frac{1}{k_{nm}^2}\overline{\mathbf{G}}^{\mathrm{TM}}_{E,B}(\mathbf{r},\mathbf{r}'), \tag{33}$$

where $k_{nm}^2 = \omega^2 \varepsilon_n \mu_m$ and

$$\overline{\mathbf{G}}_{E,B}^{\mathrm{TE}}(\mathbf{r},\mathbf{r}') = (\nabla \times \hat{\mathbf{z}})(\nabla' \times \hat{\mathbf{z}})\, g_B^{\mathrm{TE}}(\mathbf{r},\mathbf{r}')\ , \tag{34}$$

$$\overline{\mathbf{G}}_{E,B}^{\mathrm{TM}}(\mathbf{r},\mathbf{r}') = (\nabla \times \nabla \times \hat{\mathbf{z}})(\nabla' \times \nabla' \times \hat{\mathbf{z}})\, g_B^{\mathrm{TM}}(\mathbf{r},\mathbf{r}')\ . \tag{35}$$

The scalar Green's functions that appear in Eqs. (34) and (35) are written in compact form as [79]

$$g_B^{\nu}(\mathbf{r},\mathbf{r}') = \frac{j}{4\pi}\int_0^{+\infty} \frac{dk_\rho}{k_{mz}k_\rho} J_0\left(k_\rho \rho\right) F_B^{\nu}\left(k_\rho, z, z'\right)\ , \tag{36}$$

where $\nu \in \{\mathrm{TE},\mathrm{TM}\}$, k_ρ is the radial wavenumber, $k_{mz} = \sqrt{k_m^2 - k_\rho^2}$, ρ is the radial distance between $\mathbf{r}$ and $\mathbf{r}'$ and $F_B^{\nu}(k_\rho, z, z')$ is the propagation factor [78, chap. 2]. The latter determines the reflected or transmitted fields for primary TE or TM fields and is computed using the transfer matrix method [80, chap. 8].

The analytical structure of the secondary Green's tensor, as shown in Eqs. (33) to (35), allows for the transfer of differential operators from the scalar Green's function to RWG test and basis functions. This property is crucial, since curl and divergence operations appear inside surface integrals. However, the existence of k_ρ in the denominator of the integrand in Eq. (36) leads to a singularity as $k_\rho \to 0$, which is handled as in [81, chap. 6]. Regarding the singularities of the primary term, they are treated using the same semi-analytical approach employed for homogeneous backgrounds, as mentioned in Section 3.1.

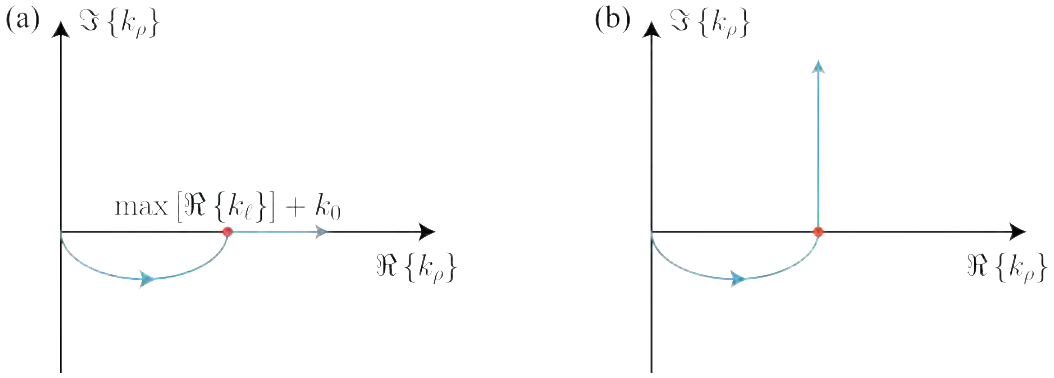


Figure 11: Integration paths for Sommerfeld integrals. The first part of the integration follows the semi-ellipse from the origin to the highlighted point of the real axis with value $\max[\Re\{k_\ell\}] + k_0$, where $\max[\Re\{k_\ell\}]$ is the maximum of the real parts of the wavenumbers in the layers of the stratified medium ($\ell = 1, ..., L$) and k_0 is the free-space wavenumber, which is used as a safety margin. (a) For large $|z - z'|$ values the contour continues along the real axis $\Re\{k_\rho\}$, while (b) for smaller values of $|z - z'|$ the path follows the parallel to the imaginary axis $\Im\{k_\rho\}$.

The evaluation of layered medium Green's function requires the numerical computation of Sommerfeld integrals, as depicted in Eq. (36). For this purpose, we adopt the integration path proposed in [82], which adapts to the vertical distance $|z - z'|$ between source and observation points. As shown in Fig. 11(a), for sufficiently large $|z - z'|$, the path consists of a semi-elliptical contour followed by the remainder of the real axis. Conversely, for small values of $|z - z'|$, the contour continues along a parallel to the imaginary axis, as illustrated in Fig. 11(b). This selection ensures an accelerated convergence of the required Sommerfeld integrals.

The final linear system is derived via the implementation of the matrix-friendly formulation [77, 79, 81, 83], inserting the Green's tensor decomposition from Eq. (33) into the PMCHWT formulation. Wherever feasible, derivative operators acting on Green's functions are shifted onto the test and basis RWG functions via partial integration. This results in expressions that are significantly more amenable to numerical evaluation. The final linear system for multiple penetrable scattering bodies in stratified backgrounds is written as

$$\left(\begin{bmatrix} \mathbf{D}_B^{\mathrm{pri}} + \mathbf{D}_{E,B}^{\mathrm{sec}} & \mathbf{K}_B^{\mathrm{pri}} + \mathbf{K}_{E,B}^{\mathrm{sec}} \\ -\mathbf{K}_B^{\mathrm{pri}} - \mathbf{K}_{H,B}^{\mathrm{sec}} & \mathbf{D}_B^{\mathrm{pri}} + \mathbf{D}_{H,B}^{\mathrm{sec}} \end{bmatrix} + \sum_{i,\, i\neq B} \begin{bmatrix} j\omega\mu_i \mathbf{D}_i & \mathbf{K}_i \\ -\mathbf{K}_i & j\omega\varepsilon_i \mathbf{D}_i \end{bmatrix}\right) \cdot \begin{bmatrix} \boldsymbol{\alpha} \\ \boldsymbol{\beta} \end{bmatrix} = \sum_i \begin{bmatrix} \mathbf{q}_i^E \\ \mathbf{q}_i^H \end{bmatrix}\ , \tag{37}$$

where the first matrix represents the stratified background domain Ω_B and the summation covers the homogeneous domain(s) of the scatterer(s). The main difference with Eq. (19) is that the matrix elements for the background domain differ from the homogeneous case and are expressed as [77, 79, 81, 83]

$$(D_B^{\text{pri}})_{\xi_1\xi_2} = j\omega\mu_m \int_{\partial\Omega_B} \mathrm{d}S\, \mathbf{f}_{\xi_1,B}(\mathbf{r}) \cdot \int_{\partial\Omega_B} \mathrm{d}S' \overline{\mathbf{G}}_B^{\text{pri}}(\mathbf{r},\mathbf{r}') \cdot \mathbf{f}_{\xi_2,B}(\mathbf{r}') \,, \tag{38}$$

$$(K_B^{\text{pri}})_{\xi_1\xi_2} = \int_{\partial\Omega_B} \mathrm{d}S\, \mathbf{f}_{\xi_1,B}(\mathbf{r}) \cdot \int_{\partial\Omega_B} \mathrm{d}S' \left[\nabla' \times \overline{\mathbf{G}}_B^{\text{pri}}(\mathbf{r},\mathbf{r}')\right] \cdot \mathbf{f}_{\xi_2,B}(\mathbf{r}') \,, \tag{39}$$

$$\begin{aligned}(D_{E,B}^{\text{sec}})_{\xi_1\xi_2} = j\omega\mu_m \Big\{ &\Big\langle \nabla \cdot \mathbf{f}_{\xi_1,B}(\mathbf{r}) \Big| k_{nm}^{-2}\partial_z\partial_{z'} g_B^{\text{TM}}(\mathbf{r},\mathbf{r}') - g_B^{\text{TE}}(\mathbf{r},\mathbf{r}') \Big| \nabla' \cdot \mathbf{f}_{\xi_2,B}(\mathbf{r}') \Big\rangle \\ &- \Big\langle \hat{\mathbf{z}} \cdot \mathbf{f}_{\xi_1,B}(\mathbf{r}) \Big| \mu_n\mu_m^{-1}\partial_{z'} g_B^{\text{TM}}(\mathbf{r},\mathbf{r}') + \partial_z g_B^{\text{TE}}(\mathbf{r},\mathbf{r}') \Big| \nabla' \cdot \mathbf{f}_{\xi_2,B}(\mathbf{r}') \Big\rangle \\ &- \Big\langle \nabla \cdot \mathbf{f}_{\xi_1,B}(\mathbf{r}) \Big| \varepsilon_m\varepsilon_n^{-1}\partial_z g_B^{\text{TM}}(\mathbf{r},\mathbf{r}') + \partial_{z'} g_B^{\text{TE}}(\mathbf{r},\mathbf{r}') \Big| \hat{\mathbf{z}} \cdot \mathbf{f}_{\xi_2,B}(\mathbf{r}') \Big\rangle \\ &+ \Big\langle \hat{\mathbf{z}} \cdot \mathbf{f}_{\xi_1,B}(\mathbf{r}) \Big| k_{mn}^2 g_B^{\text{TM}}(\mathbf{r},\mathbf{r}') - \partial_z\partial_{z'} g_B^{\text{TE}}(\mathbf{r},\mathbf{r}') \Big| \hat{\mathbf{z}} \cdot \mathbf{f}_{\xi_2,B}(\mathbf{r}') \Big\rangle \\ &+ \Big\langle \hat{\mathbf{x}} \cdot \mathbf{f}_{\xi_1,B}(\mathbf{r}) \Big| g_{B,s}^{\text{TE}}(\mathbf{r},\mathbf{r}') \Big| \hat{\mathbf{x}} \cdot \mathbf{f}_{\xi_2,B}(\mathbf{r}') \Big\rangle \\ &+ \Big\langle \hat{\mathbf{y}} \cdot \mathbf{f}_{\xi_1,B}(\mathbf{r}) \Big| g_{B,s}^{\text{TE}}(\mathbf{r},\mathbf{r}') \Big| \hat{\mathbf{y}} \cdot \mathbf{f}_{\xi_2,B}(\mathbf{r}') \Big\rangle \Big\} \,, \end{aligned} \tag{40}$$

$$\begin{aligned}(K_{E,B}^{sec})_{\xi_1\xi_2} = &\Big\langle \hat{\mathbf{z}} \cdot \mathbf{f}_{\xi_1,B}(\mathbf{r}) \Big| k_{mn}^2 \partial_\rho g_B^{\text{TM}}(\mathbf{r},\mathbf{r}') \Big| \hat{\mathbf{z}} \cdot \left[\mathbf{f}_{\xi_2,B}(\mathbf{r}') \times (\mathbf{r}-\mathbf{r}')\right]\rho^{-1} \Big\rangle \\ - &\Big\langle \hat{\mathbf{z}} \cdot \left[\mathbf{f}_{\xi_1,B}(\mathbf{r}) \times (\mathbf{r}-\mathbf{r}')\right]\rho^{-1} \Big| k_m^2 \partial_\rho g_B^{\text{TE}}(\mathbf{r},\mathbf{r}') \Big| \hat{\mathbf{z}} \cdot \mathbf{f}_{\xi_2,B}(\mathbf{r}') \Big\rangle \\ + &\Big\langle \hat{\mathbf{z}} \cdot \left[\mathbf{f}_{\xi_1,B}(\mathbf{r}) \times (\mathbf{r}-\mathbf{r}')\right]\rho^{-1} \Big| \partial_\rho\partial_{z'} g_B^{\text{TE}}(\mathbf{r},\mathbf{r}') \Big| \nabla' \cdot \mathbf{f}_{\xi_2,B}(\mathbf{r}') \Big\rangle \\ - &\Big\langle \nabla \cdot \mathbf{f}_{\xi_1,B}(\mathbf{r}) \Big| \varepsilon_m\varepsilon_n^{-1}\partial_\rho\partial_z g_B^{\text{TM}}(\mathbf{r},\mathbf{r}') \Big| \hat{\mathbf{z}} \cdot \left[\mathbf{f}_{\xi_2,B}(\mathbf{r}') \times (\mathbf{r}-\mathbf{r}')\right]\rho^{-1} \Big\rangle \,, \end{aligned} \tag{41}$$

where the following simplified notation is used

$$\Big\langle \Xi(\mathbf{r}) \Big| \Phi(\mathbf{r},\mathbf{r}') \Big| \Psi(\mathbf{r}') \Big\rangle = \int_{\partial\Omega_B}\int_{\partial\Omega_B} \mathrm{d}S \;\mathrm{d}S'\; \Xi(\mathbf{r})\Phi(\mathbf{r},\mathbf{r}')\Psi(\mathbf{r}') \,. \tag{42}$$

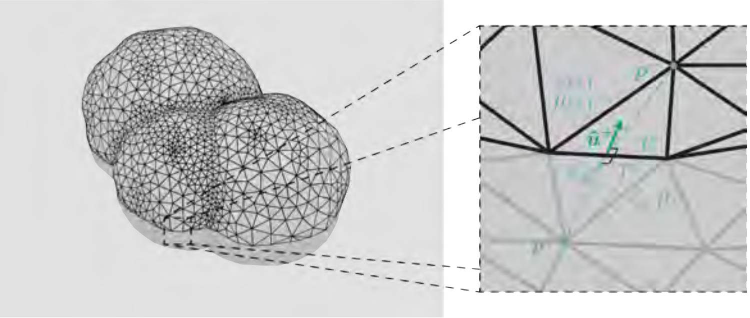


Figure 12: Close-up view of a geometric configuration relevant to the line-integral terms. The scatterers go through an interface between two layers with properties $(\varepsilon_\ell, \mu_\ell)$ and $(\varepsilon_{\ell+1}, \mu_{\ell+1})$. All triangle pairs that are modeling this feature have their common edge exactly on the interface.

During the matrix-filling process, if the surface of the scattering body passes through one or more interfaces, as shown in Fig. 12, then the evaluation of Eq. (41) requires the addition of testing (ℓ_t) and basis (ℓ_b) line integrals

$$\ell_t = \varepsilon_m\varepsilon_n^{-1} \int_{\partial T_{\xi_1}} \mathrm{d}\ell\, \mathbf{f}_{\xi_1,B}(\mathbf{r}) \cdot \hat{\mathbf{u}}_{\xi_1}^{\pm} \int_{T_{\xi_2}^{\pm}} \mathrm{d}S'\, \hat{\mathbf{z}} \cdot \left[\mathbf{f}_{\xi_2,B}(\mathbf{r}') \times (\mathbf{r}-\mathbf{r}')\right]\rho^{-1}\partial_\rho\partial_z g_B^{\text{TM}}(\mathbf{r},\mathbf{r}') \,, \tag{43}$$

$$\ell_b = -\int_{\partial T_{\xi_2}} \mathrm{d}\ell'\, \mathbf{f}_{\xi_2,B}(\mathbf{r}') \cdot \hat{\mathbf{u}}_{\xi_2}^{\pm} \int_{T_{\xi_1}^{\pm}} \mathrm{d}S\, \hat{\mathbf{z}} \cdot \left[\mathbf{f}_{\xi_1,B}(\mathbf{r}) \times (\mathbf{r}-\mathbf{r}')\right]\rho^{-1}\partial_\rho\partial_{z'} g_B^{\text{TE}}(\mathbf{r},\mathbf{r}') \,. \tag{44}$$

In these line integral terms, ∂T_{ξ_1} (∂T_{ξ_2}) denotes the common edge of the triangle pair supporting the RWG test (basis) function $\mathbf{f}_{\xi_1,B}$ ($\mathbf{f}_{\xi_2,B}$), and $\hat{\mathbf{u}}_{\xi_1}^{\pm}$ ($\hat{\mathbf{u}}_{\xi_2}^{\pm}$) is the corresponding outward unit normal on that edge. The ± sign depends on which of the two triangles is being processed by the code at a given moment. The inner surface integral is performed over $T^{\pm}_{\xi_2}$ ($T^{\pm}_{\xi_1}$), i.e. one of the two triangles supporting the basis (test) RWG function $\mathbf{f}_{\xi_2,B}$ ($\mathbf{f}_{\xi_1,B}$). This formulation is particularly convenient as the implementation evaluates matrix entries via half-RWG interactions. The testing line integral is needed only when the testing RWG function straddles the interface and the corresponding RWG or half-RWG basis functions (on either positive or negative triangles) lie within the same two adjacent layers [77]. An analogous argument determines the requirement of the basis line integral. Notably, "straddling" implies that the shared edge of a triangle pair lies exactly on the interface. Finally, the corresponding magnetic-type quantities $\mathbf{D}^{\text{sec}}_{H,B}$ and $\mathbf{K}^{\text{sec}}_{H,B}$ are obtained via the duality principle [78], implemented by the interchanges $(\dots)_E \leftrightarrow (\dots)_H$, $\varepsilon \leftrightarrow \mu$ and $(\dots)^{\text{TE}} \leftrightarrow (\dots)^{\text{TM}}$.

A significant numerical challenge in evaluating the layered medium Green's function and its derivatives, that appear in Eqs. (40) and (41), arises from the integrands of the underlying Sommerfeld representations. For many configurations, the dominant contribution to these integrals originates from large transverse wavenumbers k_ρ, corresponding to the quasi-static regime [84]. This behavior compromises the efficiency of direct numerical quadrature and interpolation-based acceleration strategies, due to the strong and non-smooth dependence on both the radial separation ρ and the vertical coordinates of source and observation points, z and z' respectively [84]. Following the approach of [84, 85], the integrands are decomposed as $g^{v}_{B}(\mathbf{r},\mathbf{r}') = g^{v,sm}_{B}(\mathbf{r},\mathbf{r}') + g^{v,qs}_{B}(\mathbf{r},\mathbf{r}')$, where $g^{v,qs}_{B}(\mathbf{r},\mathbf{r}')$ represents the quasi-static part and $g^{v,sm}_{B}(\mathbf{r},\mathbf{r}')$ the smooth part. The latter is a numerically well behaved quantity and its evaluation is accurate; however, to further accelerate repeated Green's function calculations, $g^{v,sm}_{B}(\mathbf{r},\mathbf{r}')$ and its required derivatives are tabulated on structured grids, which are adapted to the geometry of the problem. When both source and observation points lie in the uppermost or lowermost layer, a 2D grid $g^{v,sm}_{B}(\rho, z + z')$ suffices. If the points are located within the same interior layer, the use of the two independent 2D grids $g^{v,sm}_{B,+}(\rho, z + z')$ and $g^{v,sm}_{B,-}(\rho, z - z')$ is required. For other configurations, in which the points reside in different layers, a full 3D grid $g^{v,sm}_{B}(\rho,\ z,\ z')$ is used. This tabulation-interpolation scheme accelerates the computational process, while retaining the required accuracy for surface integral calculations. Hence, it ensures that interpolated values remain stable in parameter regions strongly influenced by the quasi-static contributions, which are calculated analytically and then added back to obtain the full result. Finally, all small distance singularities – which are isolated in the quasi-static part – are handled with singularity cancellation techniques [86–88].

5.2. Simulating nanostructures in stratified backgrounds

This section details the simulation of isolated scatterers within stratified media, enabled by the `--mode 2` flag. These examples demonstrate the solver's ability to account for substrate effects and complex multilayer environments using the formulation described in the previous section.

5.2.1. Silver sphere on SiO$_2$

We begin with the example of a plasmonic nanoparticle on a dielectric substrate; more specifically, an Ag sphere of radius $R = 30$ nm located on top of a SiO$_2$ half-space. This configuration allows us to validate the solver's handling of layered media Green's tensor. The simulation environment is initialized and the mesh is saved for inspection, as shown in Fig. 13(a):

```
$ python run_sie.py prepare Ag_sphere_over_SiO2 --mode 2 --materials eps --spline
$ mkdir sim_res/Ag_sphere_over_SiO2/out/media/
$ python3 pytools/plot_mesh.py --sim Ag_sphere_over_SiO2 --save  \
    ./sim_res/Ag_sphere_over_SiO2/out/media/mesh.png
```

Then, the system is assembled and solved using LU decomposition:

```
$ python3 run_sie.py solve Ag_sphere_over_SiO2 --mode 2 -l 2 -th 0 -a
```

Scattering, absorption, and extinction cross-sections are calculated and plotted. To illustrate the influence of the substrate, we compare these results against the analytical Mie theory for an identical sphere in free-space. As seen in Fig. 13(b), the presence of the SiO$_2$ substrate breaks the symmetry, leading to a spectral red-shift and modification of the resonance compared to the free-space scenario:

```
$ python3 pytools/visualization.py Ag_sphere_over_SiO2 --root sim_res --plot-csc --subjob 1  \
    --csc scs --csc acs --csc ext --save  \
    ./sim_res/Ag_sphere_over_SiO2/out/media/cross_sections.png
```

```
$ python3 pytools/visualization.py Ag_sphere_over_SiO2 --root sim_res --plot-csc --subjob 1  \
    --csc scs --csc acs --csc ext --mie-radius 30 --mie-material Ag --mie-background eps=1,0  \
    --materials-root materials --mie-spline --save  \
    ./sim_res/Ag_sphere_over_SiO2/out/media/cross_sections_with_Mie.png
```

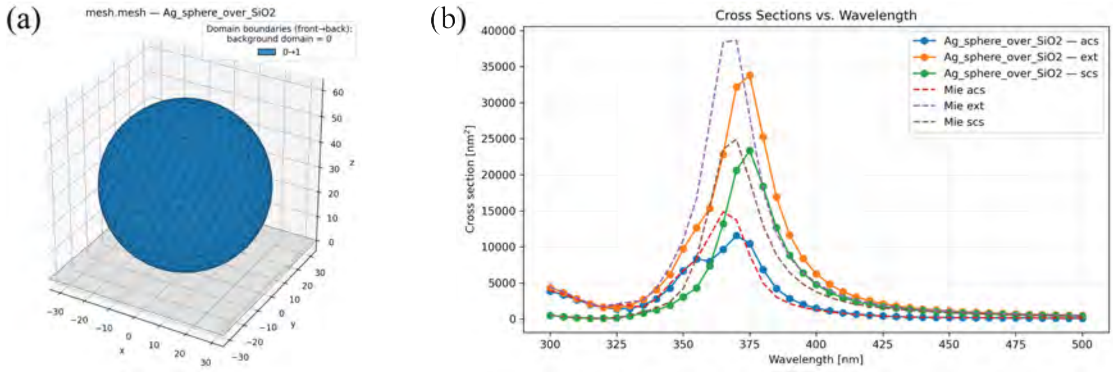


Figure 13: Ag sphere (R = 30 nm) on a SiO_2 substrate. (a) Triangular mesh with the air-SiO_2 interface, and (b) cross-section results for an incident plane wave that propagates towards $-z$ and is x-polarized, including a comparison with Mie theory for the same sphere in free-space.

Near-field intensities are computed at $\lambda = 375$ nm. We define a sampling grid in the $y = 0$ plane and visualize, as depicted in Fig. 14, the total electric field for a plane wave illumination that propagates towards $-z$ and is x-polarized, and the scattered field for a dipole illumination (x = 35 nm, z = 70 nm, $\theta = \pi/4$, $\varphi = \pi$). The post-processing and visualization commands are:

```
$ python3 run_sie.py make-points Ag_sphere_over_SiO2 xz --x -120 120 --z -90 150 --step 0.6  \
    --y0=0 -o points_xz.pos
$ python3 run_sie.py post Ag_sphere_over_SiO2 -th 0 -a -p points/points_xz.pos --lambdas 375
$ python3 pytools/visualization.py Ag_sphere_over_SiO2 --root sim_res --points points_xz.pos  \
    --field e --part tot --quantity total --function intensity --mode 2d --subjob 1 --log  \
    --hlines 0 --lambda 375 --save  \
    ./sim_res/Ag_sphere_over_SiO2/out/media/PW_375nm_Etot_intensity.png
$ python3 pytools/visualization.py Ag_sphere_over_SiO2 --root sim_res --points points_xz.pos  \
    --field e --part sc --quantity total --function intensity --mode 2d --subjob 2 --log  \
    --hlines 0 --lambda 375 --save  \
    ./sim_res/Ag_sphere_over_SiO2/out/media/dip_375nm_Esc_intensity.png
```

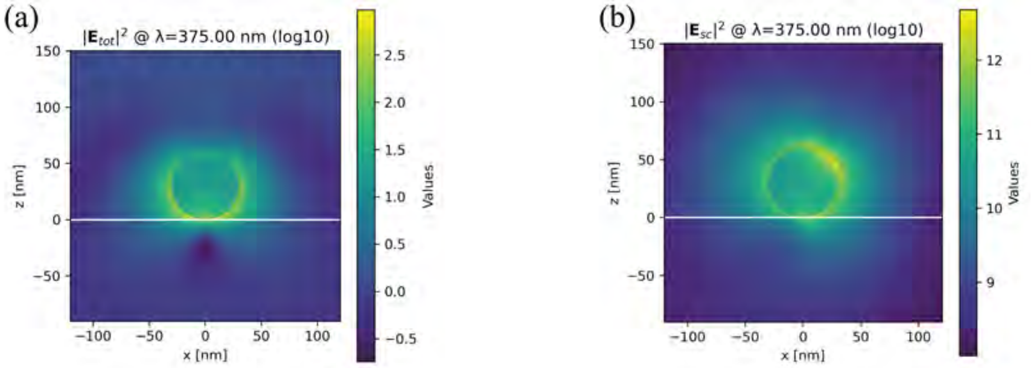


Figure 14: Electric near-field intensity at λ = 375 nm. (a) Total field for plane wave excitation (propagation towards $-z$ and is polarization along x), and (b) scattered field for in-plane dipole excitation (x = 35 nm, z = 70 nm, $\theta = \pi/4$, $\varphi = \pi$).

5.2.2. Triangular hole in gold film

Next, we investigate a triangular aperture in a gold film with thickness $d = 50$ nm. This example is particularly relevant for verifying the code implementation for bodies that are located solely within intermediate layers. The geometry is prepared and meshed, see Fig. 15(a), as follows:

```
$ python3 run_sie.py prepare triangular_hole --mode 2 --materials eps --spline
$ mkdir sim_res/triangular_hole/out/media/
$ python3 pytools/plot_mesh.py --sim triangular_hole --save  \
    ./sim_res/triangular_hole/out/media/mesh.png
```

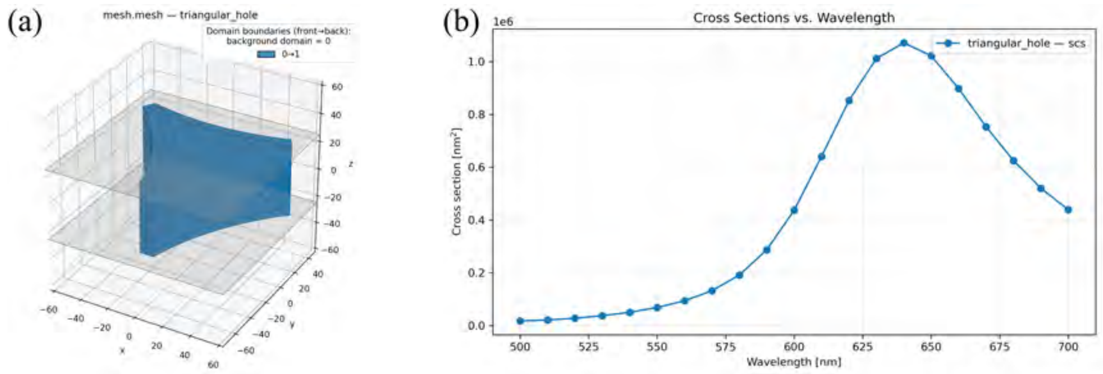


Figure 15: Triangular hole in an Au film. (a) Triangular mesh (provided in `sim_data/triangular_hole/`) with interfaces, and (b) C_{scs} for an incident plane wave that propagates towards $-z$ and is x-polarized.

After solving the problem, C_{scs} is computed to identify spectral features, as in Fig. 15(b):

```
$ python3 run_sie.py solve triangular_hole --mode 2 -l 2 -th 0 -a
$ python3 pytools/visualization.py triangular_hole --root sim_res --plot-csc --subjob 1 --csc  \
    scs --save ./sim_res/triangular_hole/out/media/cross_sections.png
```

The calculation of all cross-sections is realized via the decomposition and integration of the Poynting vector.

Near-field maps are generated at the resonance wavelength of $\lambda = 640$ nm. We sample the field in the xy-plane (at $z = 0$) for both plane wave (propagating towards $-z$ while polarized along x) and dipole ($x = 0$ nm, $y = -30$ nm, $\theta = \pi/2$, $\varphi = 0$) excitations. The resulting total and scattered field intensity distributions, as shown in Fig. 16(a) and Fig. 16(b) respectively, demonstrate the strong field confinement at the lower sharp vertex of the aperture for plane wave illumination, and the near-field response for a dipole situated within the hole ($z = 0$):

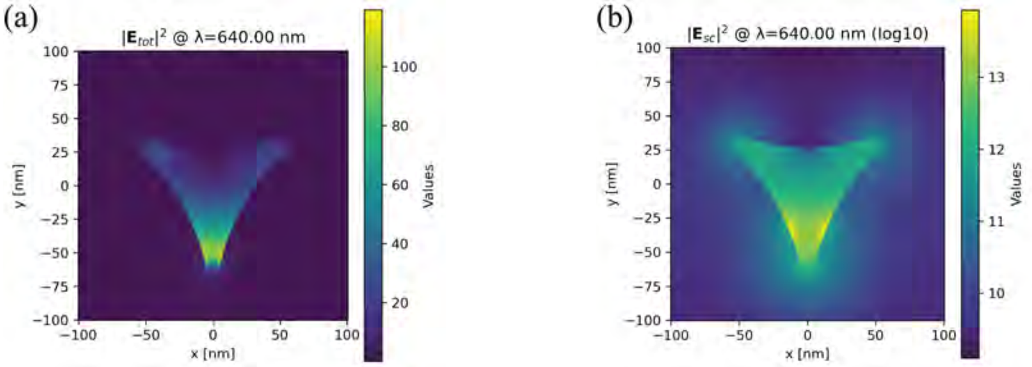


Figure 16: Total and scattered electric field intensities at $\lambda = 640$ nm in the $z = 0$ plane for (a) a plane wave propagating towards $-z$ while polarized along x, and (b) a dipole illumination ($x = 0$ nm, $y = -30$ nm, $\theta = \pi/2$, $\varphi = 0$).

```
$ python3 run_sie.py make-points triangular_hole xy --x -100 100 --y -100 100 --step 0.5 --z0=0  \
    -o points_xy.pos
$ python3 run_sie.py post triangular_hole -th 0 -a -p points/points_xy.pos --lambdas 640
$ python3 pytools/visualization.py triangular_hole --root sim_res --points points_xy.pos --field  \
    e --part tot --quantity total --function intensity --mode 2d --subjob 1 --lambda 640 --save  \
    ./sim_res/triangular_hole/out/media/PW_640nm_Etot_intensity.png
$ python3 pytools/visualization.py triangular_hole --root sim_res --points points_xy.pos --field  \
    e --part sc --quantity total --function intensity --mode 2d --subjob 2 --log --lambda 640  \
    --save ./sim_res/triangular_hole/out/media/dipx_640nm_Esc_intensity.png
```

In general, when triangular elements lie on an interface of the stratified background, the tangential electromagnetic fields in the final linear system are parallel to the boundary. Due to their continuity, any side of the interface can be considered for field computation. To avoid any extra computationally expensive inter-layer calculations, the aforementioned triangles are identified as part of the layer that is closer to the scatterer's center.

5.2.3. Cylindrical hole in multilayer stack

For the final example, we consider a more complex stratified environment, i.e. a cylindrical hole penetrating a multilayer stack. This scenario tests the implementation of the line integral corrections, see Eqs. (43) and (44), which are required when the mesh straddles through one or more interfaces between layers. Additionally, it verifies the robustness of the tabulation-interpolation scheme for Green's function, in the case of multiple layers with varying optical properties. As always, the simulation preparation and mesh inspection, see Fig. 17(a), are done first:

```
$ python3 run_sie.py prepare multilayer_hole --mode 2 --materials eps --spline
$ mkdir sim_res/multilayer_hole/out/media/
$ python3 pytools/plot_mesh.py --sim multilayer_hole --save  \
    ./sim_res/multilayer_hole/out/media/mesh.png
```

The planar interfaces are located at $z = -330, -30, 0$ nm. Considering the $+z$ direction, the materials of the different layers are SiO_2, amorphous Si, Au, and air. Regarding the cylindrical hole, it has diameter $D = 100$ nm and height $h = 330$ nm. Following the solution step we compute C_{scs}, as in Fig. 17(b), which reveals two distinct resonant wavelengths at approximately 520 nm and 685 nm:

```
$ python3 run_sie.py solve multilayer_hole --mode 2 -l 2 -th 0 -a
$ python3 pytools/visualization.py multilayer_hole --root sim_res --plot-csc --subjob 1 --csc  \
    scs --save ./sim_res/multilayer_hole/out/media/cross_sections.png
```

The system is illuminated by a plane wave that propagates towards $-z$ and is x-polarized.

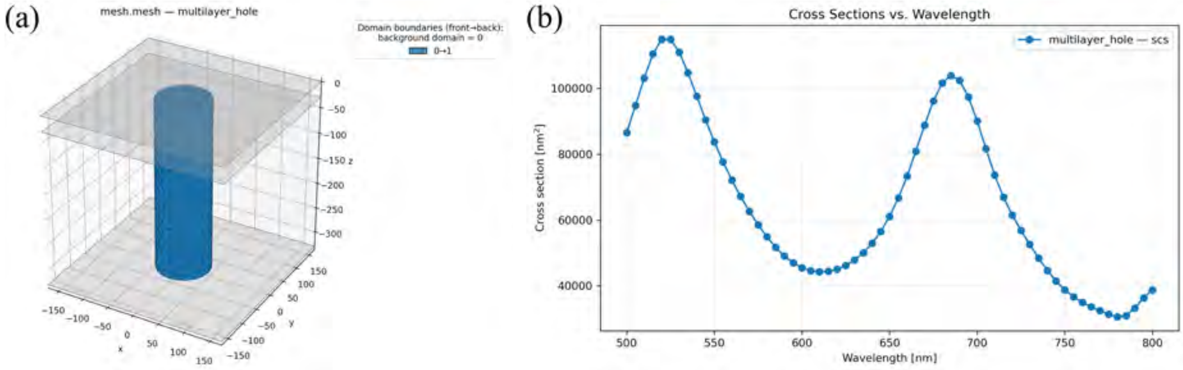


Figure 17: Cylindrical hole in a multilayer stack. (a) Triangular mesh with interfaces, and (b) C_{scs} for an incident plane wave that propagates towards $-z$ and is x-polarized. The latter shows two distinct resonances.

Near-field distributions are analyzed in the xz-plane at the two resonant wavelengths:

```
$ python3 run_sie.py make-points multilayer_hole xz --x -150 150 --z -450 100 --step 1.5 --y0=0  \
    -o points_xz.pos
$ python3 run_sie.py post multilayer_hole -th 0 -a -p points/points_xz.pos --lambdas 520
$ python3 run_sie.py post multilayer_hole -th 0 -a -p points/points_xz.pos --lambdas 685
```

To clearly visualize the intensity profiles relative to the layer boundaries, the `--hlines` argument is used to overlay the interface positions in Fig. 18. For plane wave and dipole illuminations at $\lambda = 520$ nm the commands are:

```
$ python3 pytools/visualization.py multilayer_hole --root sim_res --points points_xz.pos --field  \
    e --part tot --quantity total --function intensity --mode 2d --subjob 1 --log  \
    --hlines=-330,-30,0 --lambda 520 --save  \
    ./sim_res/multilayer_hole/out/media/PW_520nm_Etot_intensity.png
$ python3 pytools/visualization.py multilayer_hole --root sim_res --points points_xz.pos --field  \
    e --part tot --quantity total --function intensity --mode 2d --subjob 2 --log  \
    --hlines=-330,-30,0 --lambda 520 --save  \
    ./sim_res/multilayer_hole/out/media/dipx_520nm_Etot_intensity.png
```

The same distributions are extracted also for $\lambda = 685$ nm:

```
$ python3 pytools/visualization.py multilayer_hole --root sim_res --points points_xz.pos --field  \
    e --part tot --quantity total --function intensity --mode 2d --subjob 1 --log  \
    --hlines=-330,-30,0 --lambda 685 --save  \
    ./sim_res/multilayer_hole/out/media/PW_685nm_Etot_intensity.png
$ python3 pytools/visualization.py multilayer_hole --root sim_res --points points_xz.pos --field  \
    e --part tot --quantity total --function intensity --mode 2d --subjob 2 --log  \
    --hlines=-330,-30,0 --lambda 685 --save  \
    ./sim_res/multilayer_hole/out/media/dipx_685nm_Etot_intensity.png
```

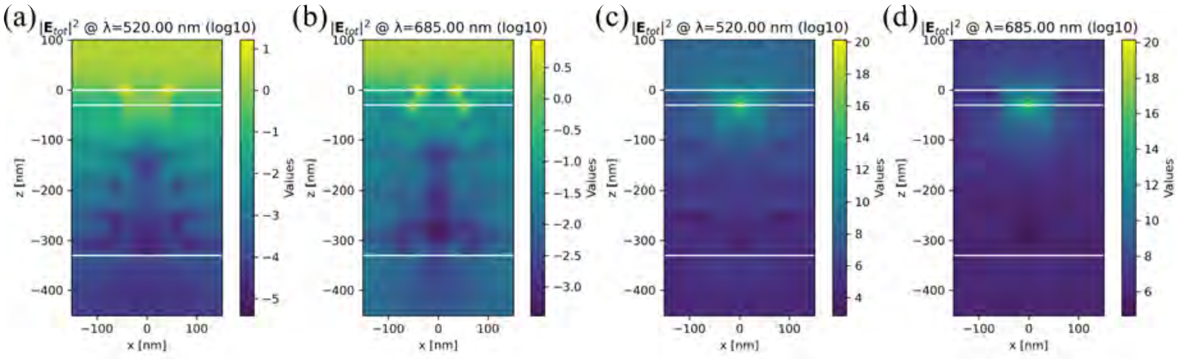


Figure 18: Total electric field intensity distributions in the xz-plane. (a, c) Results at $\lambda = 520$ nm, and (b, d) at $\lambda = 685$ nm for plane wave and dipole illuminations. The incident plane wave propagates towards $-z$ and is x-polarized, while the dipole is located at ($x = y = 0$ nm, $z = -30$ nm) with direction ($\theta = \pi/2$, $\varphi = 0$). The horizontal lines indicate the interfaces of the multilayer stack.

The continuity of tangential electromagnetic fields is verified in Fig. 19, where the magnitude of the x-component of the total electric field is plotted along a vertical line cut of the $y = 0$ plane, at $x = 60$ nm. Since the incident field is x-polarized and all interfaces are perpendicular to the z-axis, $E_{tot,x}$ represents the tangential component of the total electric field, which must remain continuous across all boundaries. As shown in both panels, the field profiles exhibit perfect continuity at the interfaces, confirming the validity of the line integral implementation in our model:

```
$ python3 pytools/visualization.py multilayer_hole --root sim_res --points points_xz.pos --field  \
    e --part tot --quantity x --function magnitude --mode 1d --subjob 1 --line x=60  \
    --hlines=-330,-30,0 --lambda 520 --save  \
    ./sim_res/multilayer_hole/out/media/abs_Etot_x_520nm_1d.png
$ python3 pytools/visualization.py multilayer_hole --root sim_res --points points_xz.pos --field  \
    e --part tot --quantity x --function magnitude --mode 1d --subjob 1 --line x=60  \
    --hlines=-330,-30,0 --lambda 685 --save  \
    ./sim_res/multilayer_hole/out/media/abs_Etot_x_685nm_1d.png
```

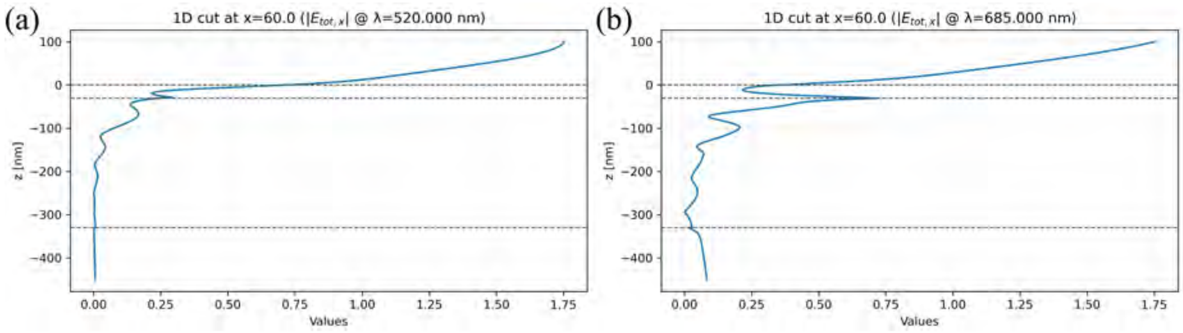


Figure 19: Vertical line profiles of the tangential electric field magnitude $|E_{tot,x}|$ at $x = 60$ nm. The profiles for (a) $\lambda = 520$ nm and (b) $\lambda = 685$ nm demonstrate field continuity across all interfaces (dashed lines).

Finally, the calculations in this scattering problem are computationally more expensive compared to previous stratified media examples, due to inter-layer interactions and 3D tabulation-interpolation.

6. Conclusion

In this paper, we have presented HELIOS, a versatile open-source software package based on the SIE method for simulating electromagnetic scattering by nanostructures in homogeneous, periodic and stratified environments. By implementing the PMCHWT formulation, the solver provides stable and accurate solutions for arbitrary 3D geometries.

A key feature of HELIOS is its comprehensive handling of diverse scenarios. For periodic systems, the integration of Ewald's transformation enables the efficient evaluation of periodic Green's tensors, facilitating the rigorous analysis of photonic crystals and metasurfaces. Furthermore, the code addresses the complexities of stratified media through the matrix-friendly formulation. The combination of advanced singularity cancellation techniques, optimized Sommerfeld integration paths, and a tabulation-interpolation scheme ensures both numerical precision and computational efficiency, even for structures penetrating multiple interfaces.

The software architecture leverages a C++ core for intensive matrix operations, coupled with a user-friendly Python interface that streamlines the simulation workflow, from pre-processing to visualization. The accuracy and flexibility of the code have been demonstrated through a series of examples, including isolated nanoparticles, periodic arrays, and apertures in multilayer stacks. These tests not only validate the numerical implementation but also highlight the code's capability to extract key optical quantities, such as cross-sections, near-field distributions, and scattering parameters.

We believe that HELIOS constitutes a valuable resource for the research community, offering a reliable tool for the design and analysis of complex nanophotonic devices. Its open-source nature encourages collaborative developments, paving the way for future extensions.

CRediT authorship contribution statement

Parmenion S. Mavrikakis: Conceptualization, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing - original draft, Writing – review and editing. **Olivier J. F. Martin:** Conceptualization, Funding acquisition, Project administration, Resources, Supervision, Writing – review and editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

It is a pleasure to acknowledge contributions from A.M. Kern (free-space scattering) and B. Gallinet (periodic systems) in the initial development of HELIOS. This work was supported by the Swiss National Science Foundation (SNSF) under project 200021 212758.

References

[1] M.I. Sancer, R.L. McClary, K.J. Glover, Electromagn. 10 (1990) 85–103. https://doi.org/10.1080/02726349008908230.

[2] E. Newman, P. Alexandropoulos, E. Walton, IEEE Trans. Antennas Propag. 32 (1984) 742–747. https://doi.org/10.1109/TAP.1984.1143408.

[3] K. Yee, IEEE Trans. Antennas Propag. 14 (1966) 302–307. https://doi.org/10.1109/TAP.1966.1138693.

[4] W.C. Chew, E. Michielssen, J.M. Song, J.M. Jin, Fast and Efficient Algorithms in Computational Electromagnetics, Artech House, Norwood, MA, 2001.

[5] A.M. Kern, A.J. Meixner, O.J.F. Martin, ACS Nano 6 (2012) 9828–9836. https://doi.org/10.1021/nn3033612.

[6] G. Quaranta, G. Basset, O.J.F. Martin, B. Gallinet, Laser Photonics Rev. 12 (2018). https://doi.org/10.1002/lpor.201800017.

[7] K. Achouri, O.J.F. Martin, IEEE Trans. Antennas Propag. 68 (2020) 432–442. https://doi.org/10.1109/TAP.2019.2943423.

[8] K. Ntokos, P. Mavrikakis, A.C. Iossifides, T.V. Yioultsis, Int. J. Electron. Commun. 170 (2023) 154780. https://doi.org/10.1016/j.aeue.2023.154780.

[9] J.P. Kottmann, O.J.F. Martin, D.R. Smith, S. Schultz, New J. Phys. 2 (2000) 27. https://doi.org/10.1088/1367-2630/2/1/327.

[10] Y. Ekinci, A. Christ, M. Agio, O.J.F. Martin, H.H. Solak, J.F. Löffler, Opt. Express 16 (2008) 13287–13295. https://doi.org/10.1364/OE.16.013287.

[11] A. Devilez, B. Stout, N. Bonod, ACS Nano 4 (2010) 3390–3396. https://doi.org/10.1021/nn100348d.

[12] U. Hohenester, A. Trügler, Comput. Phys. Commun. 183 (2012) 370–381. https://doi.org/10.1016/j.cpc.2011.09.009.

[13] D. Darvill, A. Centeno, F. Xie, Phys. Chem. Chem. Phys. 15 (2013) 15709–15726. https://doi.org/10.1039/C3CP50415H.

[14] U. Hohenester, Comput. Phys. Commun. 185 (2014) 1177–1187. https://doi.org/10.1016/j.cpc.2013.12.010.

[15] Y. Yang, I.I. Kravchenko, D.P. Briggs, J. Valentine, Nat. Commun. 5 (2014) 5753. https://doi.org/10.1038/ncomms6753.

[16] J. Waxenegger, A. Trügler, U. Hohenester, Comput. Phys. Commun. 193 (2015) 138–150. https://doi.org/10.1016/j.cpc.2015.03.023.

[17] J. Wang, J. Du, Appl. Sci. 6 (2016) 239. https://doi.org/10.3390/app6090239.

[18] F. Monticone, A. Alù, Rep. Prog. Phys. 80 (2017) 36401. https://doi.org/10.1088/1361-6633/aa518f.

[19] M. Fruhnert, I. Fernandez-Corbaton, V. Yannopapas, C. Rockstuhl, Beilstein J. Nanotechnol. 8 (2017) 614–626. https://doi.org/10.3762/bjnano.8.66.

[20] N. Bontempi, K.E. Chong, H.W. Orton, I. Staude, D.-Y. Choi, I. Alessandri, Y.S. Kivshar, D.N. Neshev, Nanoscale 9 (2017) 4972–4980. https://doi.org/10.1039/C6NR07904K.

[21] R.N.S. Suryadharma, M. Fruhnert, I. Fernandez-Corbaton, C. Rockstuhl, Phys. Rev. B 96 (2017) 045406. https://doi.org/10.1103/PhysRevB.96.045406.

[22] U. Hohenester, Comput. Phys. Commun. 222 (2018) 209–228. https://doi.org/10.1016/j.cpc.2017.08.010.

[23] D. Tzarouchis, A. Sihvola, Appl. Sci. 8 (2018) 184. https://doi.org/10.3390/app8020184.

[24] S. Sun, R. Li, M. Li, Q. Du, C.E. Png, P. Bai, J. Phys. Chem. C 122 (2018) 14771–14780. https://doi.org/10.1021/acs.jpcc.8b01978.

[25] S. Sun, T. Zhang, Q. Liu, L. Ma, Q. Du, H. Duan, J. Phys. Chem. C 123 (2019) 21150–21160. https://doi.org/10.1021/acs.jpcc.9b06280.

[26] C. Sideris, E. Garza, O.P. Bruno, ACS Photonics 6 (2019) 3233–3240. https://doi.org/10.1021/acsphotonics.9b01137.

[27] D. Ray, T.V. Raziman, C. Santschi, D. Etezadi, H. Altug, O.J.F. Martin, Nano Lett. 20 (2020) 8752–8759. https://doi.org/10.1021/acs.nanolett.0c03613.

[28] D. Beutel, A. Groner, C. Rockstuhl, I. Fernandez-Corbaton, J. Opt. Soc. Am. B 38 (2021) 1782-1791. https://doi.org/10.1364/JOSAB.419645.

[29] D. Ray, A. Kiselev, O.J.F. Martin, Opt. Express 29 (2021) 24056–24067. https://doi.org/10.1364/OE.427911.

[30] B. Zerulla, M. Krstić, D. Beutel, C. Holzer, C. Wöll, C. Rockstuhl, I. Fernandez-Corbaton, Adv. Mater. 34 (2022). https://doi.org/10.1002/adma.202200350.

[31] S.S. Kruk, L. Wang, B. Sain, Z. Dong, J. Yang, T. Zentgraf, Y. Kivshar, Nat. Photonics 16 (2022) 561–565. https://doi.org/10.1038/s41566-022-01018-7.

[32] U. Hohenester, N. Reichelt, G. Unger, Comput. Phys. Commun. 276 (2022) 108337. https://doi.org/10.1016/j.cpc.2022.108337.

[33] E. Garza, C. Sideris, ACS Photonics 10 (2023) 824-835. https://doi.org/10.1021/acsphotonics.2c01072.

[34] B. Zerulla, C. Li, D. Beutel, S. Oßwald, C. Holzer, J. Bürck, S. Bräse, C. Wöll, I. Fernandez-Corbaton, L. Heinke, C. Rockstuhl, M. Krstić, Adv. Funct. Mater. 34 (2024) 2301093. https://doi.org/10.1002/adfm.202301093.

[35] S. Athanasiou, O.J.F. Martin, J. Phys. Chem. C 128 (2024) 18574–18581. https://doi.org/10.1021/acs.jpcc.4c05322.

[36] P. Mavrikakis, O.J.F. Martin, J. Opt. Soc. Am. A 42 (2025) 315–326. https://doi.org/10.1364/JOSAA.546221.

[37] E.M. Purcell, C.R. Pennypacker, Astrophys. J. 186 (1973) 705. https://doi.org/10.1086/152538.

[38] B.T. Draine, Astrophys. J. 333 (1988) 848. https://doi.org/10.1086/166795.

[39] O.J.F. Martin, N.B. Piller, Phys. Rev. E 58 (1998) 3909–3915. https://doi.org/10.1103/PhysRevE.58.3909.

[40] S.S. Zivanovic, K.S. Yee, K.K. Mei, IEEE Trans. Microw. Theory Tech. 39 (1991) 471–479. https://doi.org/10.1109/22.75289.

[41] A. Taflove, Computational Electrodynamics: The Finite-Difference Time-Domain Method, Artech House, Norwood, MA, 1995.

[42] K.S. Yee, J.S. Chen, IEEE Trans. Antennas Propag. 45 (1997) 354–363. https://doi.org/10.1109/8.558651.

[43] K.S. Kunz, R.J. Luebbers, The Finite Difference Time Domain Method for Electromagnetics, CRC Press, Boca Raton, FL, 2018.

[44] J.-M. Jin, J.L. Volakis, J.D. Collins, IEEE Antennas Propag. Mag. 33 (1991) 22–32. https://doi.org/10.1109/74.88218.

[45] J.-M. Jin, The Finite Element Method in Electromagnetics, John Wiley & Sons, Hoboken, NJ, 1993.

[46] P.P. Silvester, R.L. Ferrari, Finite Elements for Electrical Engineers, third ed., Cambridge University Press, Cambridge, 2012.

[47] A.M. Kern, O.J.F. Martin, J. Opt. Soc. Am. A 26 (2009) 732–740. https://doi.org/10.1364/JOSAA.26.000732.

[48] P.S. Mavrikakis, O.J.F. Martin, Appl. Comput. Electromagn. Soc. J. 40 (2025) 279–301. https://doi.org/10.13052/2024.ACES.J.400402.

[49] P.P. Ewald, Ann. Phys. 369 (1921) 253–287. https://doi.org/10.1002/andp.19213690304.

[50] T.L. Veldhuizen, in: D. Caromel, R.R. Oldehoeft, M. Tholburn (Eds.), Computing in Object-Oriented Parallel Environments, Springer Berlin Heidelberg, Berlin, Heidelberg, 1998, pp. 223–230.

[51] E. Anderson, Z. Bai, C. Bischof, L.S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, D. Sorensen, LAPACK Users' Guide, third ed., Society for Industrial and Applied Mathematics, 1999.

[52] R. Piessens, E. Doncker-Kapenga, C. Uberhuber, D.K. Kahaner, Quadpack, Springer, Berlin, Germany, 1983.

[53] D.E. Amos, ACM Trans. Math. Softw. 12 (1986) 265–273. https://doi.org/10.1145/7921.214331.

[54] J. Williams, regridpack v1.0.1 [software], GitHub, February 20, 2024. https://github.com/jacobwilliams/regridpack.

[55] P.J. Flatau, arXiv E-Prints (2012). https://doi.org/10.48550/arXiv.1208.4869.

[56] A.M. Kern, O.J.F. Martin, in: S. Kawata, V.M. Shalaev, D.P. Tsai (Eds.), Plasmonics: Nanoimaging, Nanofabrication, and Their Applications V, SPIE, 2009, p. 739518. https://doi.org/10.1117/12.825833.

[57] B.M. Kolundzija, IEEE Trans. Microw. Theory Tech. 47 (1999) 1021–1032. https://doi.org/10.1109/22.775434.

[58] P. Ylä-Oijala, M. Taskinen, J. Sarvas, Electromagn. Waves (Camb.) 52 (2005) 81–108. http://doi.org/10.2528/PIER04071301.

[59] R.F. Harrington, Field Computation by Moment Methods, Wiley-IEEE Press, 1993.

[60] R.F. Harrington, J. Electromagn. Waves Appl. 3 (1989) 1–15. https://doi.org/10.1163/156939389X00016.

[61] X.Q. Sheng, J.-M. Jin, J. Song, W.C. Chew, C.-C. Lu, IEEE Trans. Antennas Propag. 46 (1998) 1718–1726. https://doi.org/10.1109/8.736628.

[62] A.J. Poggio, E.K. Miller, in: R. Mittra (Ed.), Computer Techniques for Electromagnetics, Pergamon, 1973, pp. 159–264.

[63] Y. Chang, R.F. Harrington, IEEE Trans. Antennas Propag. 25 (1977) 789–795. https://doi.org/10.1109/TAP.1977.1141685.

[64] T. Wu, L.L. Tsai, Radio Sci. 12 (1977) 709–718. https://doi.org/10.1029/RS012i005p00709.

[65] I. Hanninen, M. Taskinen, J. Sarvas, Prog. In Electromagn. Res. 63 (2006) 243–278. http://doi.org/10.2528/PIER06051901.

[66] J.P. Kottmann, O.J.F. Martin, IEEE Trans. Antennas Propag. 48 (2000) 1719–1726. https://doi.org/10.1109/8.900229.

[67] T.V. Raziman, W.R.C. Somerville, O.J.F. Martin, E.C. Le Ru, J. Opt. Soc. Am. B 32 (2015) 485–492. https://doi.org/10.1364/JOSAB.32.000485.

[68] C. Girard, A. Cuche, E. Dujardin, A. Arbouet, A. Mlayah, Opt. Lett. 40 (2015) 2116. https://doi.org/10.1364/OL.40.002116.

[69] S. Athanasiou, O.J.F. Martin, J. Phys. Chem. C 129 (2025) 4282–4289. https://doi.org/10.1021/acs.jpcc.4c08617.

[70] T. Honegger, O. Lecarme, K. Berton, D. Peyrade, Microelectron. Eng. 87 (2010) 756–759. https://doi.org/10.1016/j.mee.2009.11.145.

[71] T. Zentgraf, J. Valentine, N. Tapia, J. Li, X. Zhang, Adv. Mater. 22 (2010) 2561–2564. https://doi.org/10.1002/adma.200904139.

[72] K.V. Garapati, M. Salhi, S. Kouchekian, G. Siopsis, A. Passian, Phys. Rev. B 95 (2017) 165422. https://doi.org/10.1103/PhysRevB.95.165422.

[73] W. Ludwig, C. Falter, Symmetries in Physics: Group Theory Applied to Physical Problems, Springer, Berlin, Germany, 1996.

[74] K.E. Jordan, G.R. Richter, P. Sheng, J. Comput. Phys. 63 (1986) 222–235. https://doi.org/10.1016/0021-9991(86)90093-8.

[75] I. Stevanovic, P. Crespo-Valero, K. Blagovic, F. Bongard, J.R. Mosig, IEEE Trans. Microw. Theory Tech. 54 (2006) 3688–3697. https://doi.org/10.1109/TMTT.2006.882876.

[76] B. Gallinet, A.M. Kern, O.J.F. Martin, J. Opt. Soc. Am. A Opt. Image Sci. Vis. 27 (2010) 2261–2271. https://doi.org/10.1364/JOSAA.27.002261.

[77] Y.P. Chen, W.C. Chew, L. Jiang, IEEE Trans. Antennas Propag. 60 (2012) 4766–4776. https://doi.org/10.1109/TAP.2012.2207332.

[78] W.C. Chew, Waves and Fields in Inhomogenous Media, Wiley-IEEE Press, New York, NY, 1995.

[79] W.C. Chew, J.L. Xiong, M.A. Saville, IEEE Antennas Wirel. Propag. Lett. 5 (2006) 490–494. https://doi.org/10.1109/LAWP.2006.886306.

[80] U. Hohenester, Nano and Quantum Optics, 1st ed., Springer, Cham, Switzerland, 2019.

[81] W.C. Chew, M.S. Tong, B. Hu, Integral Equation Methods for Electromagnetic and Elastic Waves, Morgan & Claypool, San Rafael, CA, 2008.

[82] M. Paulus, P. Gay-Balmaz, O.J.F. Martin, Phys. Rev. E 62 (2000) 5797–5807. https://doi.org/10.1103/PhysRevE.62.5797.

[83] Y.P. Chen, W.E.I. Sha, W.C.H. Choy, L. Jiang, W.C. Chew, Opt. Express 20 (2012) 20210–20221. https://doi.org/10.1364/OE.20.020210.

[84] U. Hohenester, Comput. Phys. Commun. 294 (2024) 108949. https://doi.org/10.1016/j.cpc.2023.108949.

[85] S. Pratama, D. van Oosten, Opt. Express 23 (2015) 21741–21760. https://doi.org/10.1364/OE.23.021741.

[86] D.J. Taylor, IEEE Trans. Antennas Propag. 51 (2003) 1630–1637. https://doi.org/10.1109/TAP.2003.813623.

[87] J. D’Elía, L. Battaglia, A. Cardona, M. Storti, Int. J. Numer. Method Biomed. Eng. 27 (2011) 314–334. https://doi.org/10.1002/cnm.1309.

[88] S. Sarraf, E. López, G. Ríos Rodríguez, J. D’Elía, Comput. and Appl. Math. 33 (2014) 63–80. https://doi.org/10.1007/s40314-013-0043-5.

[89] A.M. Kern, O.J.F. Martin, IEEE Trans. Antennas Propag. 58 (2010) 2158–2161. https://doi.org/10.1109/TAP.2010.2046870.