GT1 : Modélisation & simulation
webinar 14 Janvier 2025

Software Tool for Modeling Photonic Nanostructures

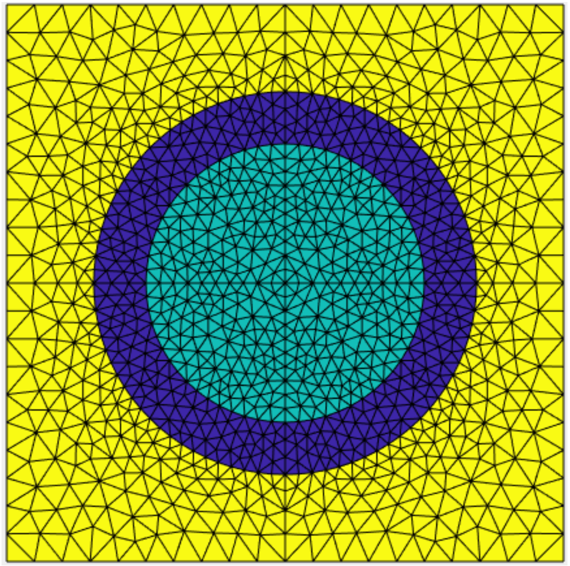
SimPhotonics_FMM Matlab Toolbox

MIT license

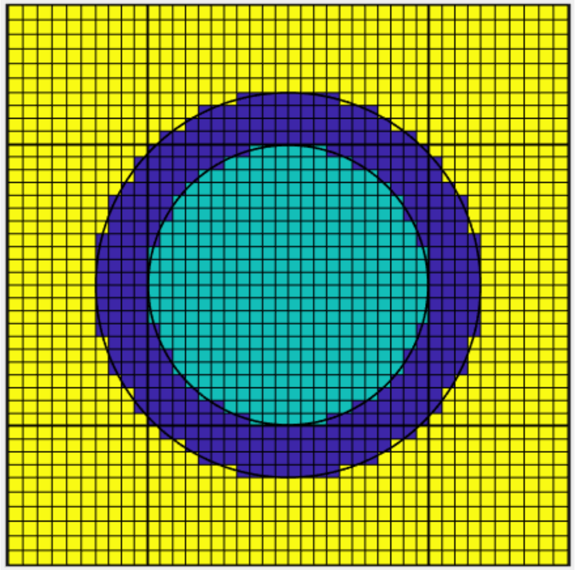
https://github.com/SimPhotonicsFMM/Sources

Mondher Besbes
Groupe Nanophotonique
Service Modélisation & Calcul Scientifique

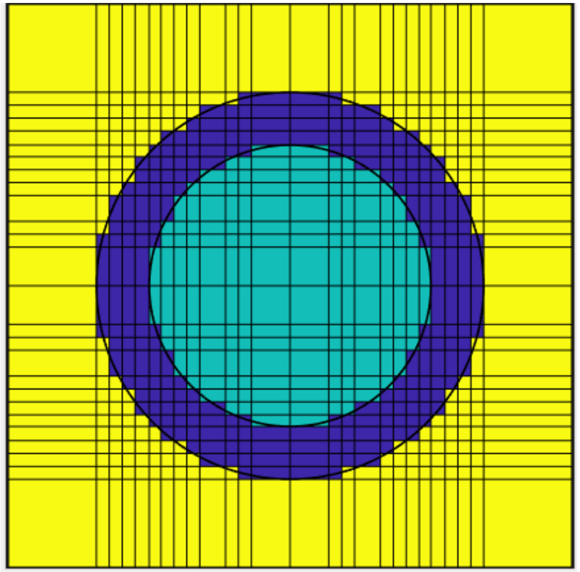
Numerical Solving of Maxwell’s Equations



Finite Element Method

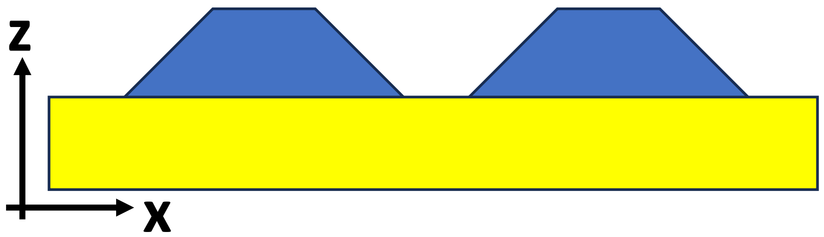


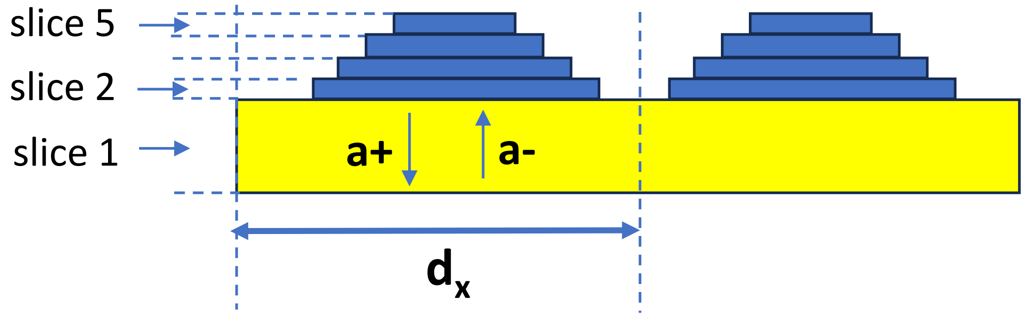
Finite Difference Method



Fourier Modal Method

Fourier Modal Method … RCWA - Reminder





Decomposition of the field into Floquet series

$$E(x,y,z) = \sum_{\substack{-m_x \to m_x \\ -m_y \to m_y}} \tilde{E}_{m_x,m_y}(z) e^{\left(i\ (\beta^x_{m_x} x + \beta^y_{m_y} y)\right)}$$

$$\beta^x_{m_x} = \beta^x + m_x \frac{2\pi}{d_x}$$

$$\beta^y_{m_y} = \beta^y + m_y \frac{2\pi}{d_y}$$

Maxwell's Equations in Fourier space

$$\boxed{\frac{d}{dz}\begin{pmatrix} \tilde{E}^x \\ \tilde{E}^y \\ \tilde{H}^x \\ \tilde{H}^y \end{pmatrix} = \begin{pmatrix} [0] & [A] \\ [B] & [0] \end{pmatrix} \begin{pmatrix} \tilde{E}^x \\ \tilde{E}^y \\ \tilde{H}^x \\ \tilde{H}^y \end{pmatrix}}$$

$$\boxed{\frac{d^2 \tilde{E}}{dz^2} = [AB]\tilde{E}}$$

Eigenvalue calculation of [AB]
Matrices T or S

Features

- ✓ Model multilayer structure, 1D and 2D periodic metamaterials
- ✓ Reflection and transmission of incident light
- ✓ Design of nanoparticles using the Gielis's SuperFormula
- ✓ Dispersive materials … *handle functions*
- ✓ Reduced CPU time with y-axis symmetry
- ✓ Finite difference coupled to S-matrix approach
- ✓ Reduced model … Proper Orthogonal Decomposition
- ✓ Quasi Normal Mode calculation (ω or $k_{//}$)
- ✓ Intuitive field visualization
- ✓ Parallel computation
- ✓ User-friendly examples

Main functions

- **SetGeom** : define geometric parameters of the structure
- **Spectrum** : compute S-matrices and the spectrum of reflectivity-transmittivity
- **Field** : compute S-matrices and field distribution
  - CalculFieldFMM : calculate the field distribution
  - VisuFieldFMM : plot the distribution of the electromagnetic field

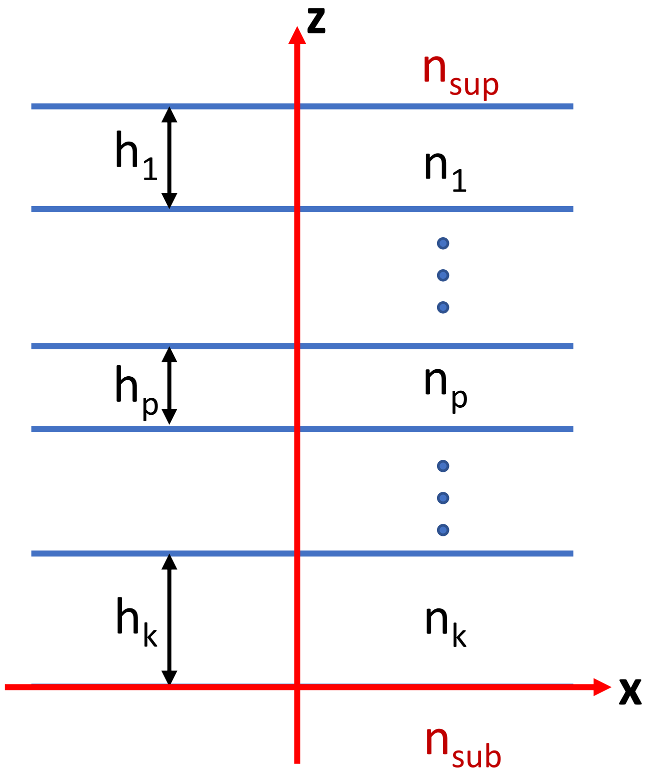
- MeshLayer : discretize different layers of the structure
- VisuMesh : plot the discretization of the structure

- IndexVal : refractive index library

Geometrical and optical properties

geom : Layer width (μm)

index : Refractive index



➢ Homogenous multilayer

geom = [h_1 … h_p … h_k];

index = [n_{sup} n_1 … n_p … n_k n_{sub}]; % constant refractive indices

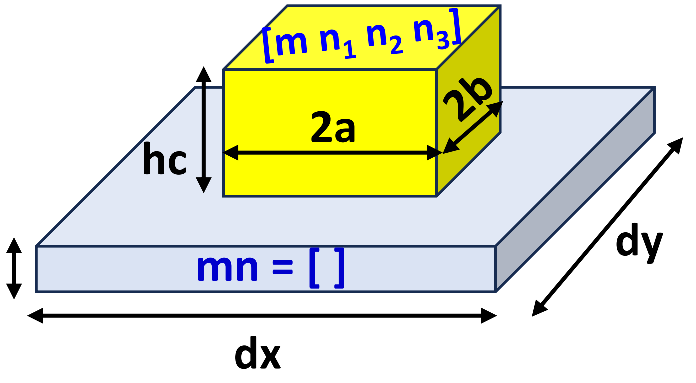
for dispersive material (use handle function)

index = {n_{sup} n_1 … @n_p … n_k n_{sub} }; % cell array

index = {n_{sup} n_1 … **IndexVal**('Material') … n_k n_{sub} };

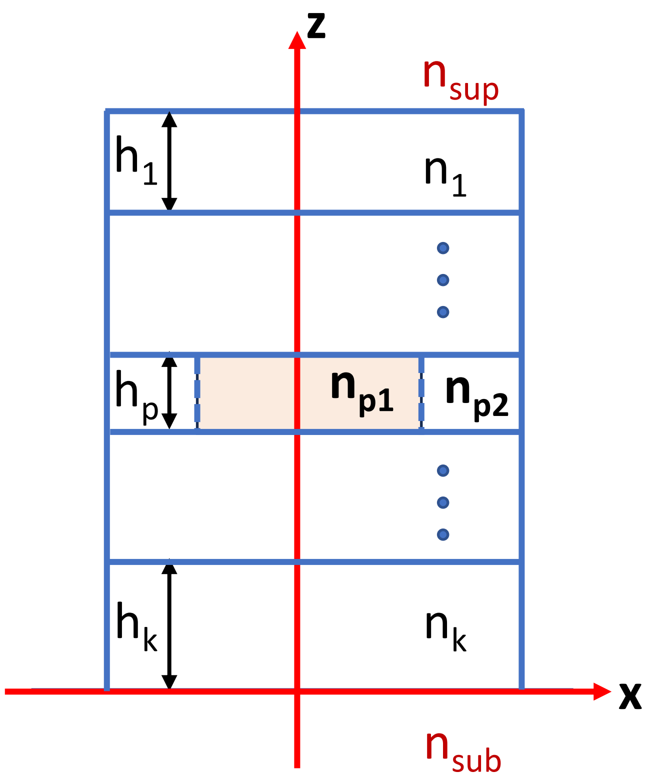
➢ Grating + multilayer : use Gielis's SuperFormula

$$r(\varphi) = \left(\left| \frac{\cos\left(\frac{m\varphi}{4}\right)}{a} \right|^{n_2} + \left| \frac{\sin\left(\frac{m\varphi}{4}\right)}{b} \right|^{n_3} \right)^{-\frac{1}{n_1}}$$



geom = SetGeom('dx',...,'dy',...,'hc',...,'mn',[m n1 n2 n3],'ab',[a b],'Angle',...,'Dep',...,'NumSD',...,'npx',..,'npy',..,'Plot', ..)

Optional parameters



➢ Grating + multilayer

geom = SetGeom('Param1',Val1,'Param2',Val2,...);

index = {n_{sup} n_1 … [n_{p1} n_{p2} …] … n_k n_{sub} }; % Constant value

{supstrat, layers, {grating}, layers, substrat}

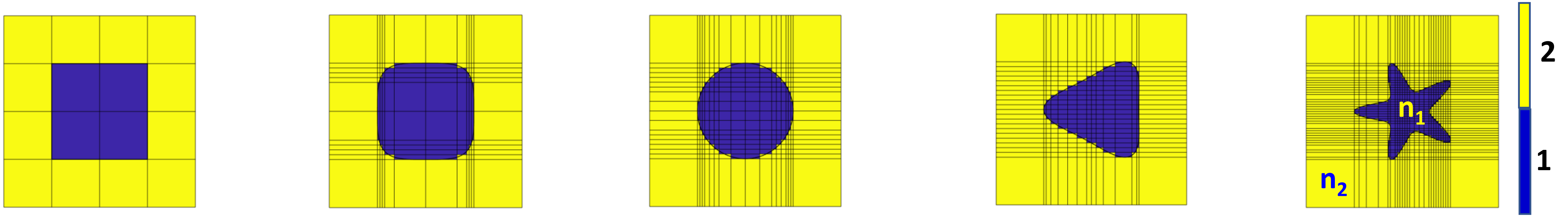
index = {n_{sup} n_1 … {@n_{p1} n_{p2} … }… n_k n_{sub} };

Gielis's SuperFormula

$$r(\varphi) = \left(\left| \frac{\cos\left(\frac{m\varphi}{4}\right)}{a} \right|^{n_2} + \left| \frac{\sin\left(\frac{m\varphi}{4}\right)}{b} \right|^{n_3} \right)^{-\frac{1}{n_1}}$$

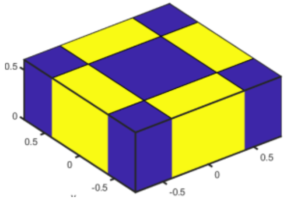
mn :[4 2 2 2] **[4 4 4 4]** **[3 6 6 6]** **[5 2 7 7]** **[4 1 1 1]** **[8 10 10 10]** **mn : [4 0 0 0]**
mn : 4

```
Mesh = MeshLayer(geom);  % discretization of the structure according to npx and npy value
figure, VisuMesh(Mesh)      % plot mesh
```

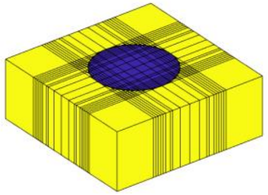


Reference number of subdomains is used to establish refractive index table: index = { nsup …[n_1 n_2]…nsub}

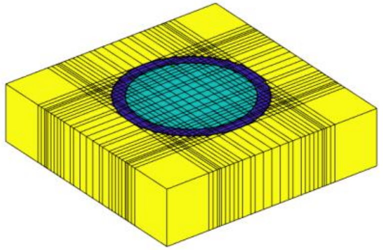
Examples of structure



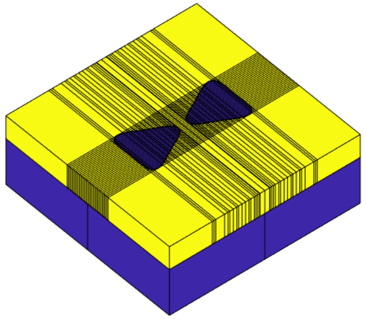
Checkerboard



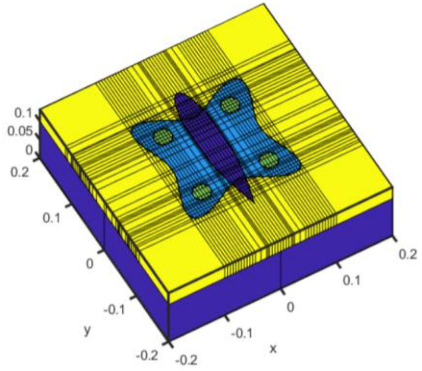
Pilar



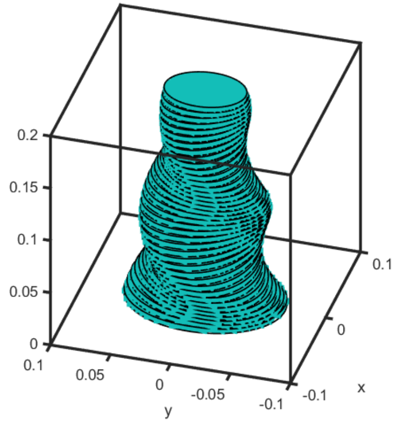
Core-shell



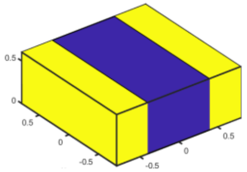
Bowtie



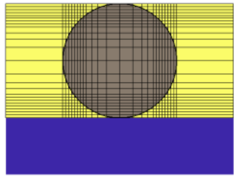
Morpho



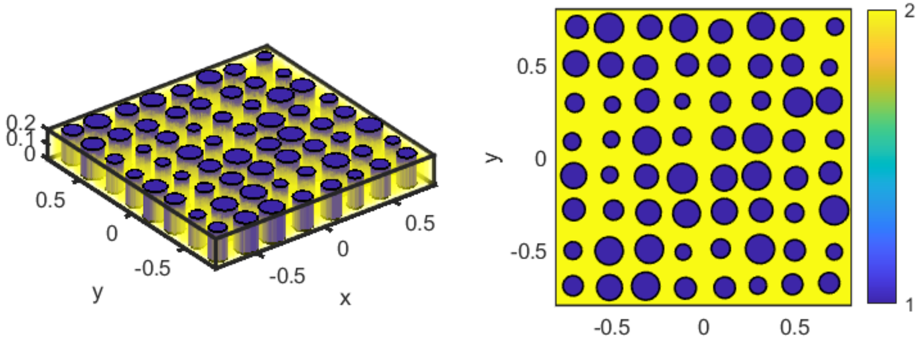
Babel tower



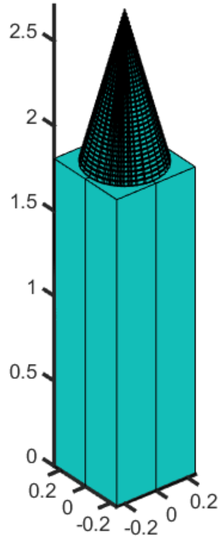
Slits



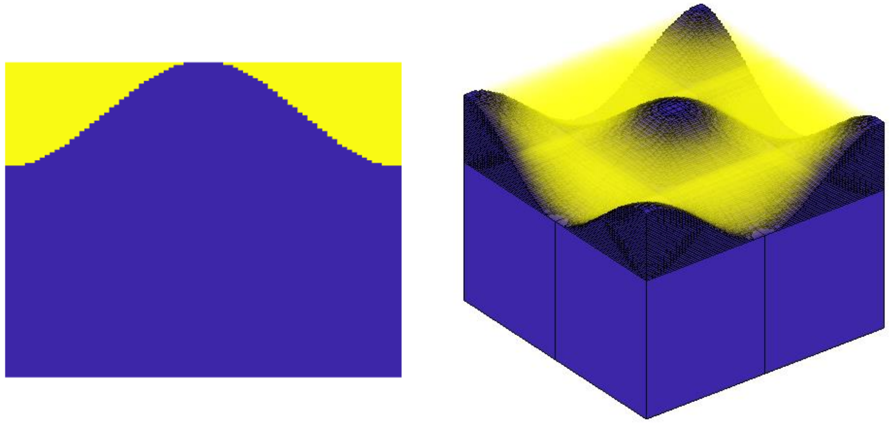
Pipes



Super cell, random size



Black Si



Free-Form

Refractive index library

```
load('IndexData.mat')
```

56 materials

Name	n	k	Fct	Ref
'Ag'	*49x2 double*	*49x2 double*	@(x)IndexVal('Ag',x)	'P. B. Johnson and R. W. Christy. Optical c...
'Air'	*101x2 doub...*	[]	@(x)1+0.05792105./(238.0185-x.^-2)+...	'P. E. Ciddor. Refractive index of air: new ...
'Al'	*200x2 doub...*	*200x2 doub...*	@(x)IndexVal('Al',x)	'A. D. Rakić. Algorithm for the determina...
'Al2O3'	*101x2 doub...*	[]	@(x)sqrt(1+1.4313493./(1-(0.0726631./...	'1) I. H. Malitson and M. J. Dodge. Refrac...
'AlAs'	*101x2 doub...*	[]	@(x)sqrt(1+1.0792+6.0840./(1-(0.2822....	'R. E. Fern and A. Onton. Refractive index ...
'AlGaAs'	*46x2 double*	*46x2 double*	@(x)IndexVal('AlGaAs',x)	'D. E. Aspnes, S. M. Kelso, R. A. Logan an...
'AlGaSb'	*56x2 double*	*56x2 double*	@(x)IndexVal('AlGaSb',x)	'D. E. Aspnes, S. M. Kelso, R. A. Logan an...
'AlSb'	*45x2 double*	*45x2 double*	@(x)IndexVal('AlSb',x)	'S. Zollner, C. Lin, E. Schönherr, A. Böhrin...
'Au'	*50x2 double*	*50x2 double*	@(x)IndexVal('Au',x)	'P. B. Johnson and R. W. Christy. Optical c...
'BK7'	*101x2 doub...*	*25x2 double*	@(x)sqrt(1+1.03961212./(1-0.0060006...	'SCHOTT Zemax catalog 2017-01-20b (o...
'BaTiO3'	*101x2 doub...*	[]	@(x)sqrt(1+4.187./(1-(0.223./x).^2))	'S.H. Wemple, M. Didomenico Jr., and I. C...
'Bi'	*150x2 doub...*	*150x2 doub...*	@(x)IndexVal('Bi',x)	'1) H.-J. Hagemann, W. Gudat, and C. Kun...
'C'	*176x2 doub...*	*176x2 doub...*	@(x)IndexVal('C',x)	'H. R. Phillip and E. A. Taft. Kramers-Kron...
'C3H6O'	*101x2 doub...*	[]	@(x)1.34979+0.00306.*x.^-2+0.00006....	'J. Rheims, J Köser and T Wriedt. Refracti...
'CO2'	*101x2 doub...*	[]	@(x)1+6.99100e-2./(166.175-x.^-2)+1....	'A. Bideau-Mehu, Y. Guern, R. Abjean and...

```
IndexVal('Au',0.6)

ans = 0.2487 + 3.0740i
```

```
IndexVal({'Au' 'BK7'},0.6)

ans = 1×2 complex
    0.2487 + 3.0740i   1.5163 + 0.0000i
```

```
IndexVal('BaTiO3')

ans = function_handle with value:
    @(x)sqrt(1+4.187./(1-(0.223./x).^2))
```

```
IndexVal({ 'Au' 'BK7'})

ans = 1×2 cell
```

	1	2
1	*@(x)IndexV...*	*@(x)sqrt(1...*

Spectrum calculation

Reflectance and transmittance calculation versus wavelength and/or incidence angle.

[R_tm,T_tm,R_te,T_te] = Spectrum(index,geom,lambda,theta,inc) % Multilayer

inc = -1 incidence from bottom, **+1** from top

[R_tm,T_tm,R_te,T_te] = Spectrum(index,geom,lambda,theta,inc, ’mx’,…) % 1D Grating

[R_tm,T_tm,R_te,T_te] = Spectrum(index,geom,lambda,theta,inc, ’mx’,…,’my’,…) % 2D Grating

mx and **my** number of Fourier terms

[R,T] = Spectrum(index,geom,lambda,theta,inc, ’Param’,Val)

S = Spectrum(index,geom,lambda(1),theta(1),inc, ’Param’,Val) ← **S-matrix calculation**

Optional parameters:

SymY : Symmetry in y, =0 :TM (PMC), =1 : TE (PEC), by default =2
Num : Diffracted order numbers, by default Num=[]
Phi0 : Azimuthal angle (rd), by default Phi0=0
Nper: Possible number of periods (Bragg mirror), by default Nper=1

Field calculation

Electric and magnetic field calculation for a giving value of wavelength and incidence angle.

- + at a point
- + along a line // axis
- + cross section (xy) – (xz) – (yz)
- + 3D grid
- + coordinate table

1st method

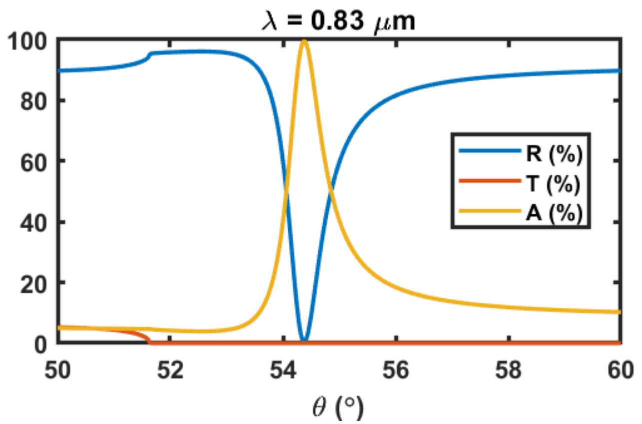
```
geom = SetGeom(geom, 'x',…,'y',…,'z',….)    % Update "geom" – x=y=z=0 per default

[E_tm,H_tm,E_te,H_te] = Field(index,geom,lambda,theta,inc , 'Param',Val)
```

2d method

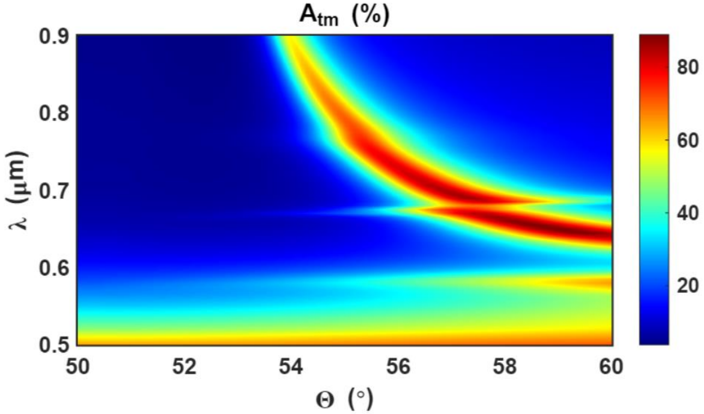
```
S = Spectrum(index,geom,lambda,theta,inc, 'Param',Val)

[E_tm,H_tm,E_te,H_te] = CalculFieldFMM(S,geom)

[E_tm,H_tm,E_te,H_te] = CalculFieldFMM(S,x,y,z)

[E_tm,H_tm,E_te,H_te] = CalculFieldFMM(S,CoorXYZ)
```

Plot results

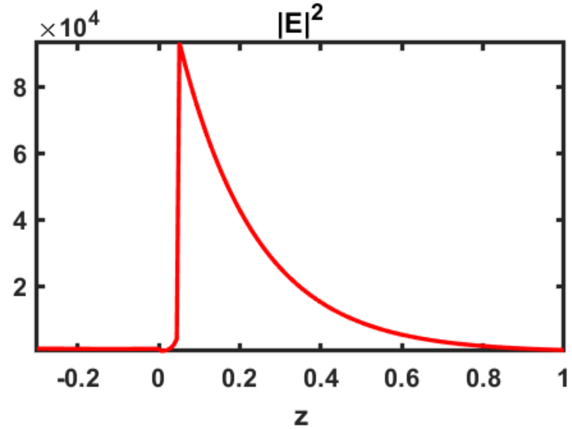
```
figure
PlotCoefRTA(lambda,theta,Rtm)
PlotCoefRTA(lambda,theta,Rtm,Rte)
PlotCoefRTA(lambda,theta,Rtm,Ttm,1-Rtm-Ttm)
```

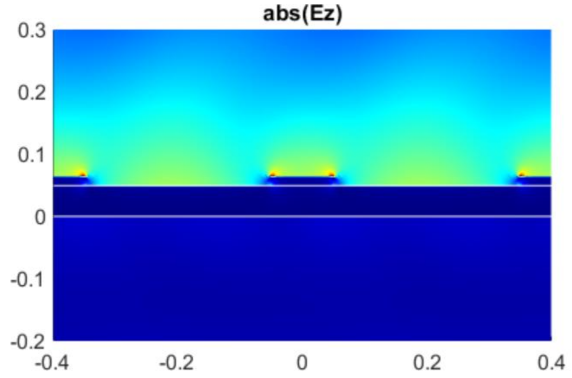



```
PlotCoefRTA(lambda,theta,1-Rtm-Ttm)
```

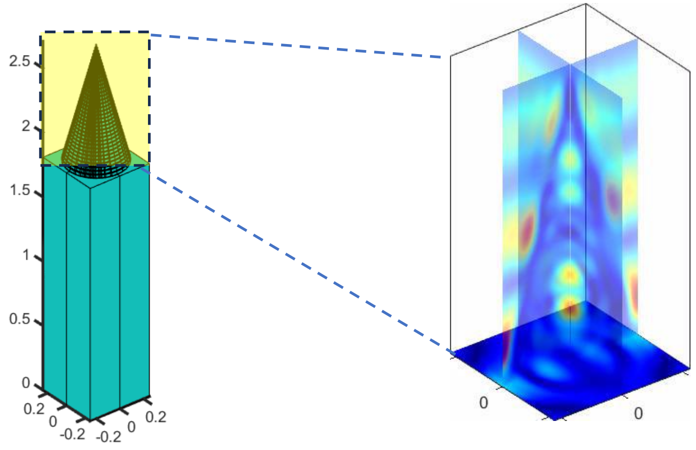



```
figure
VisuFieldFMM(V,geom)     % V = abs(E(:,3))
VisuFieldFMM(V,x,y,z)
```

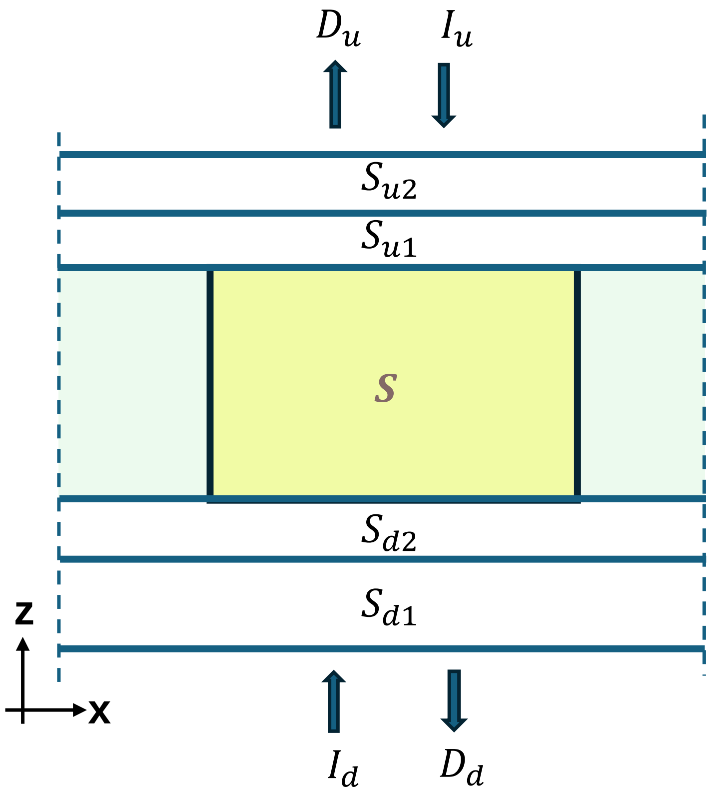


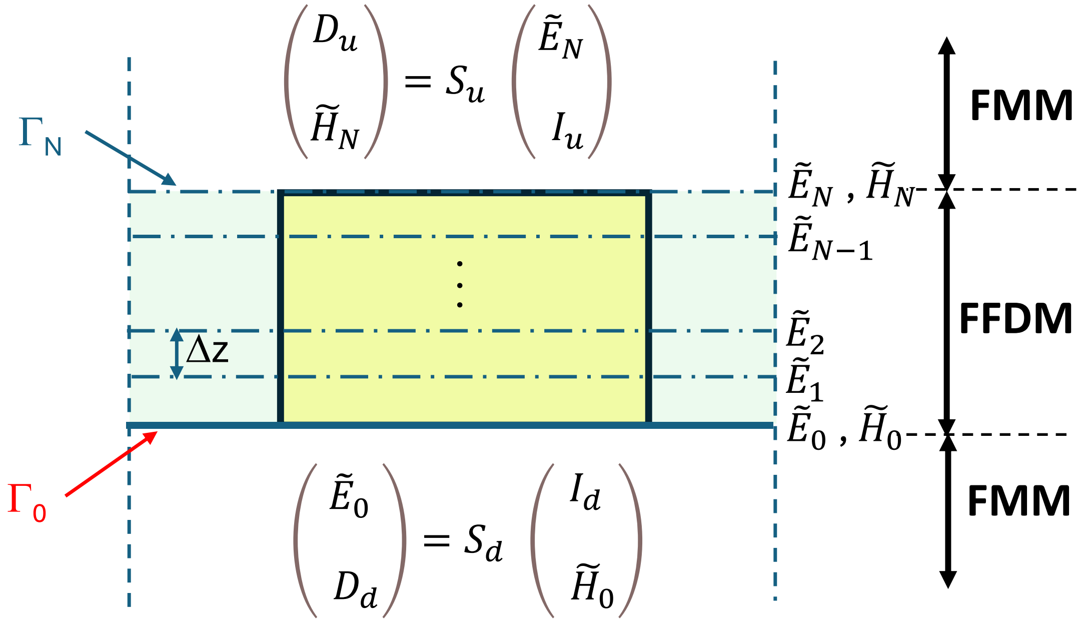


3D grid



New Features…Save CPU time





$$\frac{d^2\tilde{E}}{dz^2} = [AB]\tilde{E}$$

Fourier Finite Difference Method

$$\tilde{E}_{n-1} - 2\tilde{E}_n + \tilde{E}_{n+1} = \Delta z^2 AB\tilde{E}_n$$

N subdivisions … for m_x=m_y=50 and N = 1, SpeedUp ~ 30

[R_tm,T_tm,R_te,T_te] = Spectrum(index,geom,lambda,theta,inc, ’mx’,…,’my’,…, ’Nsub’,[0 0 N 0 0])

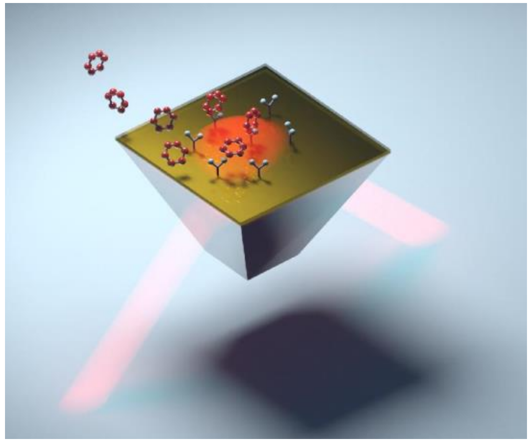
Tutorials & Examples

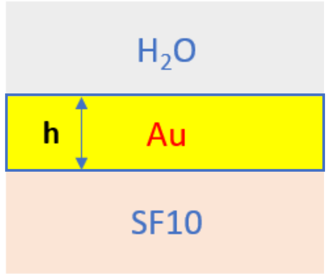
- TutoStepByStep.mlx: Discover step by step the different stages of a simulation with SimPhotonics_FMM
- TutoSuperFormula.mlx: Interactive generation of different shapes of nanoparticles with Gielis's Superformula
- TutoGeomMesh.mlx: Some examples of geometry generation and the associated mesh
- TutoSpectrum.mlx: How to use '*Spectrum.m*' with different examples

- ✓ AntiReflectionCoating.mlx
- ✓ BlazedGrating1D.mlx
- ✓ SPRBioSensor.mlx
- ✓ SPRBioSensorGrating1D.mlx
- ✓ SPRBioSensorGrating2D.mlx

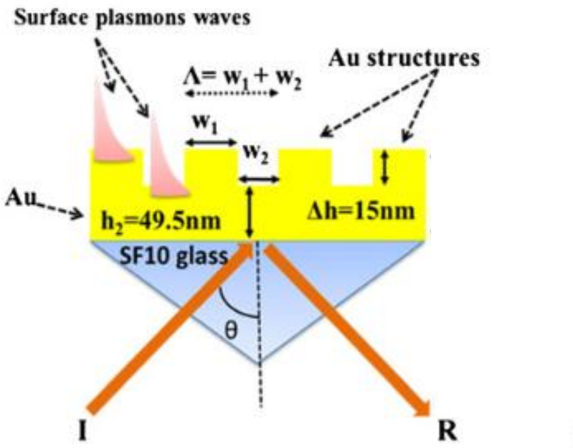
Case study : SPR Biosensor

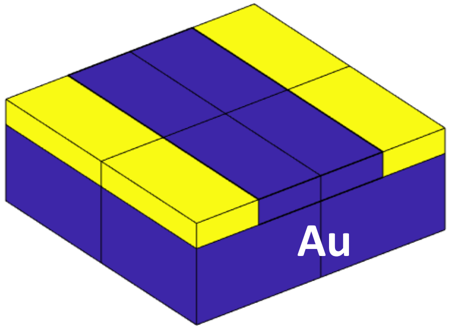
0D Multilayer structure





1D Grating





2D Grating