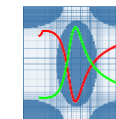


Tutorial SimPhotonics_FMM



Mondher Besbes
Laboratoire Charles Fabry
CNRS - IOGS

[SimPhotonics_FMM](#) is a useful and powerful Matlab toolbox for the simulation of nanophotonic structures. It is based on the **Modal Fourier Method** (well known as RCWA) which can be used to model **multilayer structure, 1D and 2D periodic metamaterials**.

[SimPhotonics_FMM](#) includes a specific feature that allows the design of nanoparticles by the use of the **Gielis's SuperFormula**.

This toolbox is the result of research works developed in the Charles Fabry laboratory and in particular the contributions of J.P. Hugonin.

With its advanced features, [SimPhotonics_FMM](#) is a valuable resource for researchers and engineers working in the field of nanophotonics.

Features

- ✓ Reflection and transmission of incident light
- ✓ Dispersive materials
- ✓ Reduced CPU time with y-axis symmetry
- ✓ Quick calculation of a Bragg mirror
- ✓ Intuitive field visualization
- ✓ Parallel computation
- ✓ User-friendly examples



<https://github.com/SimPhotonicsFMM/Sources>

MIT licence

IDDN.FR.001.510043.000.S.P.2023.000.31235.

DOI: [10.13140/RG.2.2.27149.92642](https://doi.org/10.13140/RG.2.2.27149.92642)

Main functions

- **SetGeom** : define geometric parameters of the structure
- **IndexVal** : refractive index library
- **Spectrum** : compute S-matrices and the spectrum of reflectivity-transmittivity

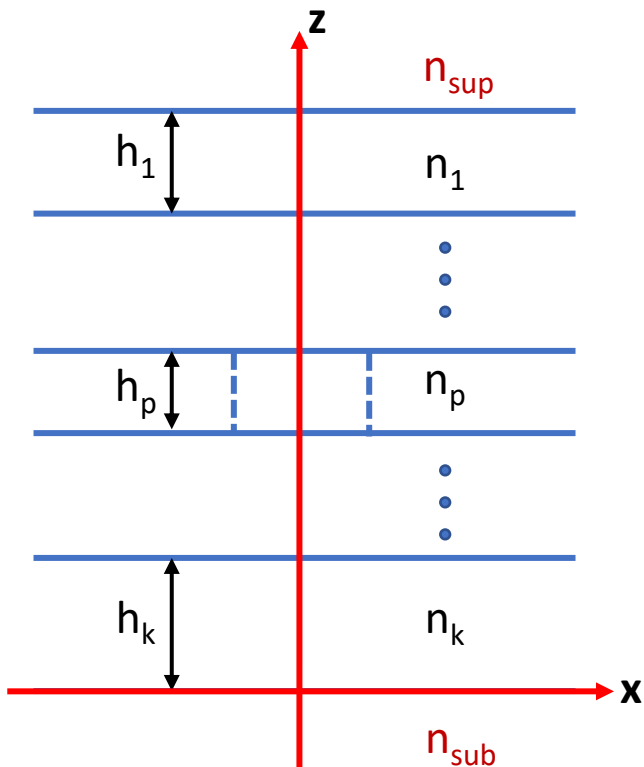
- **Field** : compute S-matrices and field distribution
- **CalculFieldFMM** : calculate the field distribution
- **VisuFieldFMM** : plot the distribution of the electromagnetic field

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

Geometrical and optical properties

geom : Layer width (μm)

index : Refractive index



➤ Homogenous multilayer

```
geom = [ h1 ... hp ... hk ];
```

```
index = [ nsup n1 ... np ... nk nsub ]; % constant refractive indices
```

for dispersive material (use handle function)

```
index = { nsup n1 ... @np ... nk nsub }; % cell array
```

```
index = { nsup n1 ... IndexVal('Material') ... nk nsub };
```

➤ Grating + multilayer : use [Gielis's SuperFormula](#)

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

```
index = { nsup n1 ... [ np1 np2 ... ] ... nk nsub }; % Constant value  
          { substrat, layers, { grating }, layers, substrat }
```

```
index = { nsup n1 ... { @np1 np2 ... } ... nk nsub };
```

Gielis's SuperFormula

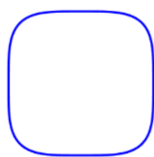
$$r(\varphi) = \left(\left| \frac{\cos\left(\frac{m\varphi}{4}\right)}{a} \right|^{n_2} + \left| \frac{\sin\left(\frac{m\varphi}{4}\right)}{b} \right|^{n_3} \right)^{-\frac{1}{n_1}}$$

Optional parameters

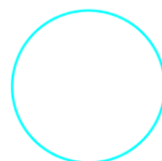
`geom = SetGeom('dx',..., 'dy',..., 'hc',..., 'mn', [m n1 n2 n3], 'ab', [a b], 'Angle',..., 'Dep',..., 'Num',..., 'npx',..., 'npy',..., 'Plot', ...)`



mn : 4



mn : [4 4 4 4]



mn : [4 2 2 2]

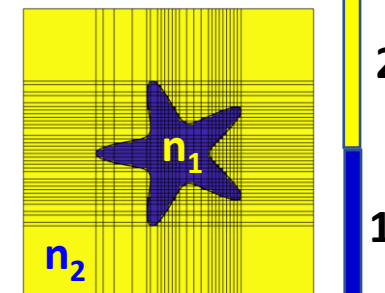
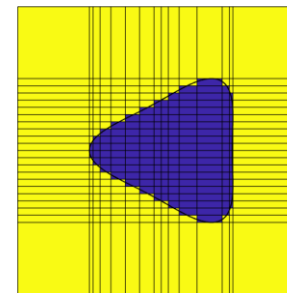
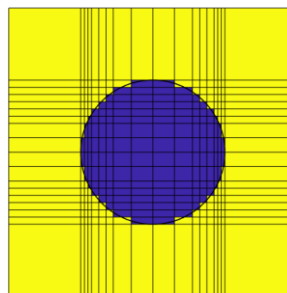
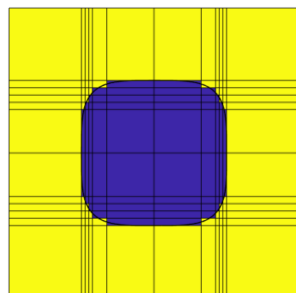
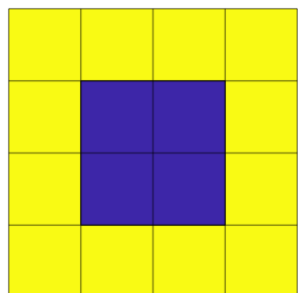


mn : [3 6 6 6]



mn : [5 2 7 7]

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

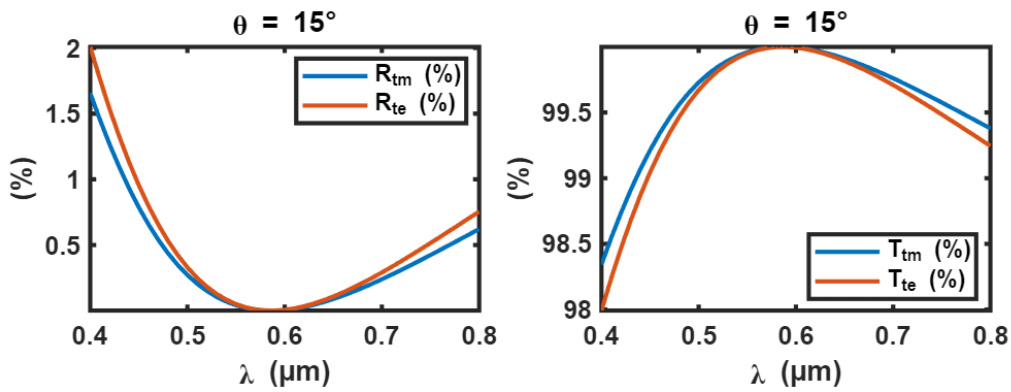
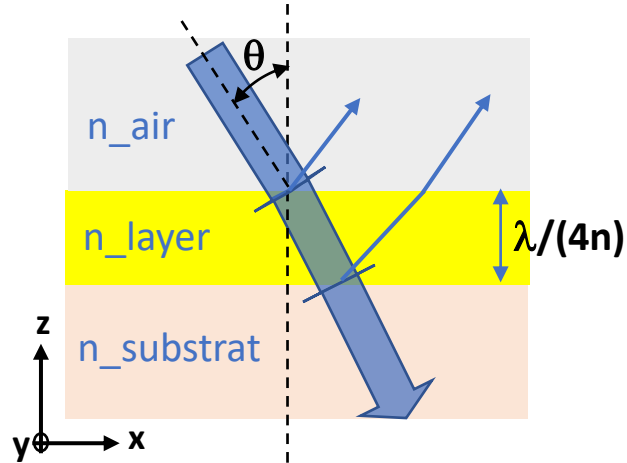


Reference number of subdomains is used to establish refractive index table: `index = { nsup ...[n1 n2]...nsub }`

Tutorials & Examples

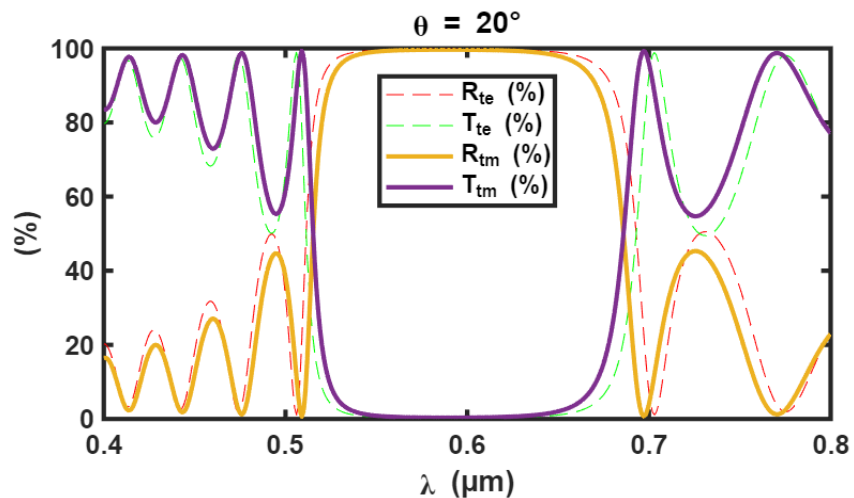
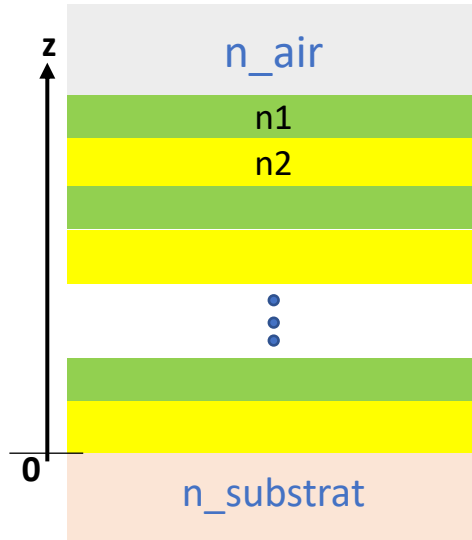
- [TutoStepByStep.mlx](#): Discover step by step the different stages of a simulation with SimPhotonics_FMM
- [TutoSuperFormula.mlx](#): Interactive generation of different shapes of nanoparticles with Gielis's Superformula
- [TutoGeomMesh.mlx](#): Some examples of geometry generation and the associated mesh
- [TutoSpectrum.mlx](#): How to use '*Spectrum.m*' with different examples
 - ✓ [AntiReflectionCoating.mlx](#)
 - ✓ [BlazedGrating1D.mlx](#)
 - ✓ [SPRBioSensor.mlx](#)
 - ✓ [SPRBioSensorGrating1D.mlx](#)
 - ✓ [SPRBioSensorGrating2D.mlx](#)

Anti-reflection coating



```
%
% Refractive index
n_air = 1; n_subtrat = 1.5; n_layer = sqrt(n_air*n_subtrat);
index = [n_air n_layer n_subtrat];
%
% Geometry
wl = .6; % Wavelength reference (μm)
geom = wl/(4*n_layer); % Width of anti-reflection layer (μm)
%
% Incident plane wave
lambda = 0.4:0.01:0.8; % Wavelength (μm)
theta = 15*pi/180; % Angle of incidence (rd)
inc = +1; % Incidence from top =+1, from down =-1
%
% Spectrum calculation for TM and TE polarization
[R_tm,T_tm,R_te,T_te] = Spectrum(index,geom,lambda,theta,inc);
%
% Plot results (with PlotCoefRTA or simply with matlab function plot)
figure
subplot(121), PlotCoefRTA(lambda,theta,R_tm,R_te),
    legend('R_{tm} (%)','R_{te} (%)'), axis tight
subplot(122), PlotCoefRTA(lambda,theta,T_tm,T_te),
    legend('T_{tm} (%)','T_{te} (%)'), axis tight
```

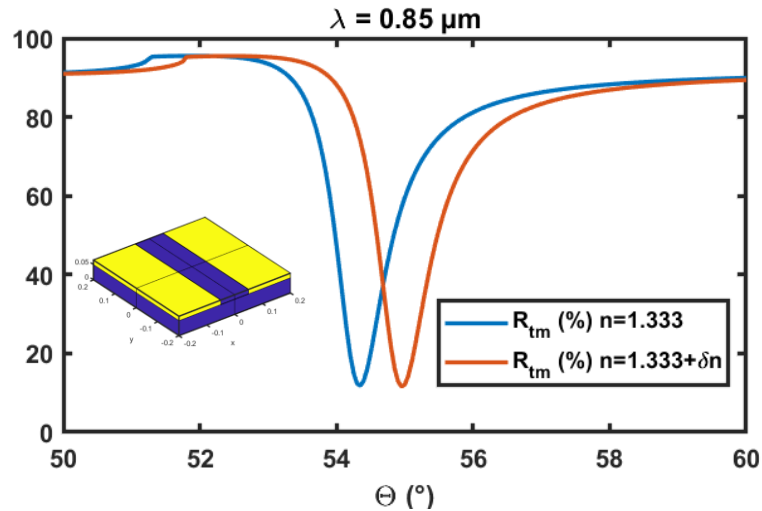
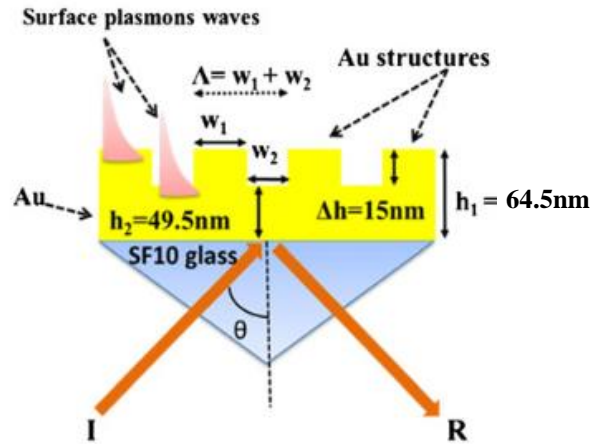
Bragg mirror



```
% Calculation with the optional parameter 'Nper' Number of periods to
% reduce CPU time

Np = 10; % Number of periods
%
n_air = 1; n_subtrat = 1.5; n1 = 1.5; n2 = 2.2;
index = [n_air n1 n2 n_subtrat];
% index = [n_air repmat([n1 n2],1,Np) n_subtrat]; % Second method (slow)
%
wl0 = .6; % Wavelength reference
geom = [wl0/4/n1, wl0/4/n2]; % Width of different layers from top to down
% geom = repmat([wl0/4/n1 , wl0/4/n2],1,Np); % Second method
%
lambda = .4:.001:.8;
theta = 20*pi/180;
inc = +1;
%
[R_tm,T_tm,R_te,T_te] = Spectrum(index,geom,lambda,theta,inc,'Nper',Np);
% [R_tm,T_tm,R_te,T_te] = Spectrum(index,geom,lambda,theta,inc);
%
figure, hold on
    plot(lambda,R_te*100,'--r',lambda,T_te*100,'--g'),
    PlotCoefRTA(lambda,theta,R_tm,T_tm),
    legend('R_{te} (%)','T_{te} (%)','R_{tm} (%)','T_{tm} (%)')
```


SPR Biosensor

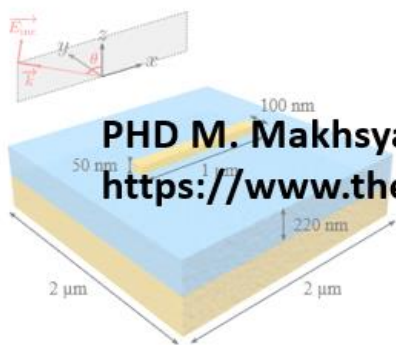


```
% Geometry (μm)
h1 = 0.0645; % Au thickness (layer+grating)
h2 = 0.0495; % Gold layer thickness
dx = 0.4;    % Period x-axis
w1 = dx/4;   % Grating width
%
geom = SetGeom('dx',dx,'mn',{4 []},'ab',{[w1/2 inf] []},'hc',[h1-h2, h2]);
%
% Refractive indices
n_H2O = 1.333;
% {supstrat , {grating} , layers , substrat}
index1 = {n_H2O , {IndexVal('Au') n_H2O}, IndexVal('Au'), IndexVal('SF10')};
%
% Incident Plane Wave
lambda = .85; % Wavelength(μm)
theta = linspace(50,60,201)*pi/180; % Incident angle (rd)
inc = -1;
%
% Spectrum Computation versus incident angle for a defined wavelength
R_tem1 = Spectrum(index1,geom,lambda,theta,inc,'mx',20);
%
% n_H2O + Δn : reflectivity variation
index2 = index1;
index2{1} = n_H2O + 1e-2; index2{2}{2} = n_H2O + 1e-2;
R_tem2 = Spectrum(index2,geom,lambda,theta,inc,'mx',20);
%
% Plot Reflection versus incident angle in TM polarisation
figure,
PlotCoefRTA(lambda,theta,R_tem1(:,1),R_tem2(:,1)),
legend('R_{tm} (%) n=1.333','R_{tm} (%) n=1.333+Δn')
```

[Enhanced SPR sensitivity with nano-micro-ribbon grating - an exhaustive simulation mapping](#)

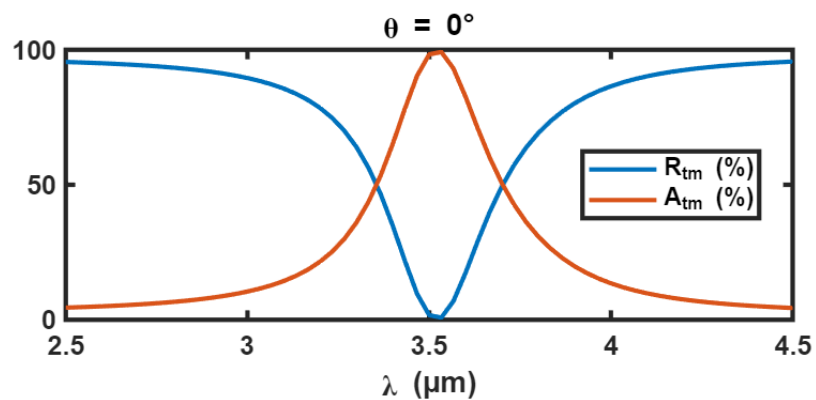
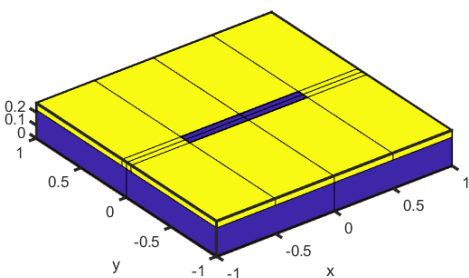
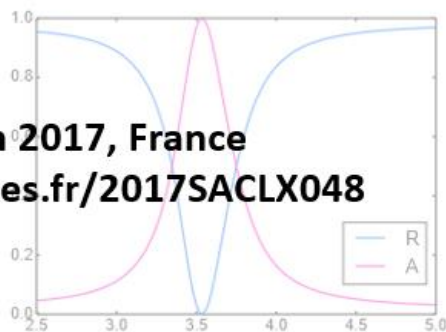
M. Chamtouri, A. Dhawan, M. Besbes, J. Moreau, H. Ghalila, T. Vo-Dinh, M. Canva
Plasmonics, 2013, pp.10.1007/s11468-013-9600-4. [10.1007/s11468-013-9600-4](#)

MIM plasmonic nanoantenna



PHD M. Makhryan 2017, France

<https://www.theses.fr/2017SACLX048>



% Geometry (μm)

dx = 2;

dy = 2;

L = 1;

w = 0.1;

h = [0.05 0.22];

%

% With Superformula

geom = SetGeom('dx',dx,'dy',dy,'hc',h,'mn',{4 , []},'ab',{[L/2 w/2] , []});

%

% Drude model for Au

xr = 0.159; g = 0.0077;

nAu = @(x) sqrt(1-1./(xr./x.*(xr./x+1i*g)));

%

index = {1 , {nAu 1} , IndexVal('SiO2'), nAu};

%

lambda = linspace(2.5,4.5,61);

theta = 0;

inc = +1;

%

% With symmetry in y-axis (θ : TM - 1: TE)

[R,T] = Spectrum(index,geom,lambda,theta,inc,'mx',15,'my',15,'SymY',0);

% Without Symmetry

%[R_tm,T_tm,R_te,T_te] = Spectrum(index,geom,lambda,theta,inc,'mx',15,'my',15);

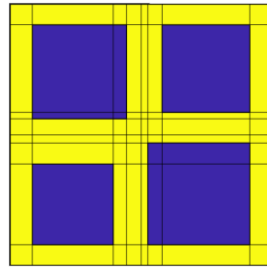
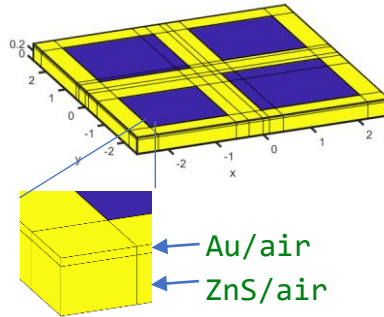
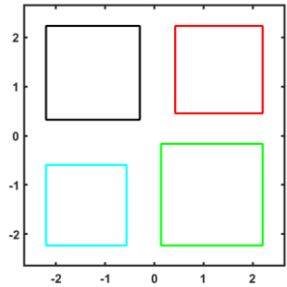
%

% Plot Reflection and absorption versus wavelength in TM polarisation (SymY=0)

Figure, PlotCoefRTA(lambda,theta,R,1-R-T), legend('R_{tm} (%)','A_{tm} (%)')

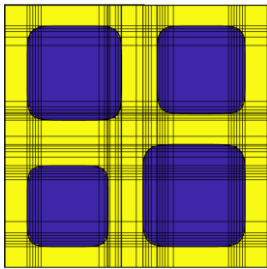
% Geometric data

```
d = 5.3;
h_ZnS = 0.29; h_Au = 0.05;
w1 = 1.64; w2 = 1.78; w3 = 1.91; w4 = 2.07;
cx = 4.41; cy = 4.47;
%
Tab_mn = repmat([4],4,1); % 4 squared particles
TabR = 0.5*[w1 w1; w2 w2; w3 w3; w4 w4];
Dep_XY = 0.5*[-cx+w1 -cy+w1; cx-w2 cy-w2; -cx+w3 cy-w3; cx-w4 -cy+w4];
%
% Use cell array for multi-layers (2 layers: Au/air + ZnS/Air)
%
geom = SetGeom('dx',d,'dy',d,'hc',[h_Au h_ZnS],'mn',{Tab_mn , Tab_mn},...
    'ab',{TabR , TabR},'Dep',{Dep_XY , Dep_XY},'Plot',1);
```



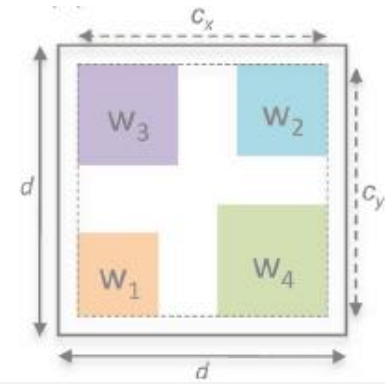
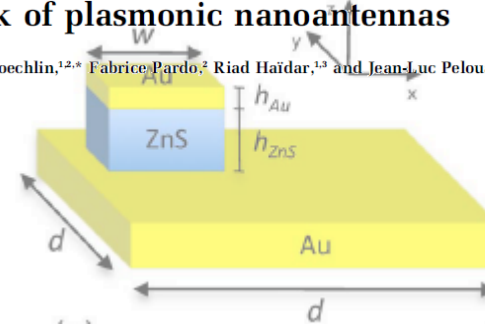
% With rounded squared shape

```
Tab_mn1 = repmat([4 6 6 6],4,1);
geom1 = SetGeom(geom,'mn',{Tab_mn1 , Tab_mn1}, ...
    'npx',80, 'npy',80);
%
mesh = MeshLayer(geom1);
figure, VisuMesh(mesh),
```



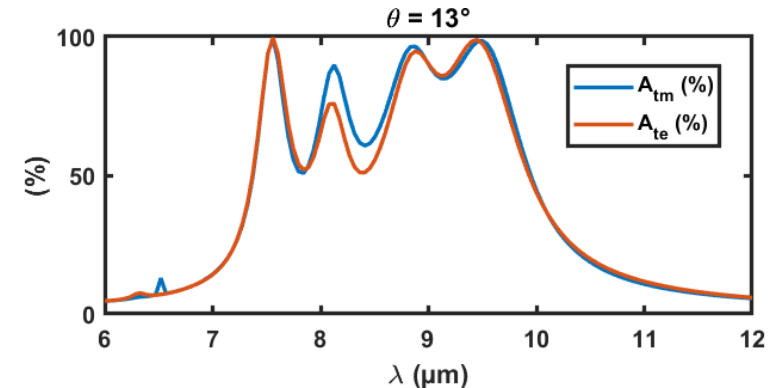
Wideband omnidirectional infrared absorber with a patchwork of plasmonic nanoantennas

Patrick Bouchon,¹ Charlie Koechlin,^{1,2,*} Fabrice Pardo,² Riad Haïdar,^{1,3} and Jean-Luc Pelouard²



% Drude model for Au

```
xr = 0.159; g = 0.0048; nAu = @(x) sqrt(1-1./(xr./x.*(xr./x+1i*g)));
n_ZnS = 2.2; n_air = 1;
%
index = {n_air , {nAu n_air } ,{n_ZnS n_air } , nAu};
%
lambda = linspace(6,12,128);
theta = 13*pi/180;
inc = +1;
%
% Spectrum calculation
[R_tm,T_tm,R_te,T_te] = Spectrum(index,geom,lambda,theta,inc,'mx',15,'my',15);
%
figure, PlotCoefRTA(lambda,theta, 1-R_tm-T_tm, 1-R_te-T_te)
legend('A_{tm} (%)','A_{te} (%)')
```



Field calculation

```
lambda = 9.5;
% S-Matrix calculation
s = Spectrum(index,geom,lambda,theta,inc,'mx',15,'my',15);

% Field calculation at (xy) plan
geom = SetGeom(geom,'z',h_ZnS/2,'x',linspace(-d/2,d/2,100),'y',linspace(-d/2,d/2,110));
[E,H] = CalculFieldFMM(s,geom.x,geom.y,geom.z);

% Second method
[E,H] = Field(index,geom,lambda,theta,inc,'mx',15,'my',15);

% Plot field (E:[Etm,Ete] H:[Htm,Hte])
figure, VisuFieldFMM(abs(H(:,2)).^2,geom.x,geom.y,geom.z) % TM
figure, VisuFieldFMM(abs(H(:,4)).^2,geom.x,geom.y,geom.z) % TE
```

```
geom = SetGeom(geom,'z',linspace(-.1,2,100),'x',0,'y',0);

% Field calculation along z-axis
[E_tm,H_tm,E_te,H_te] = Field(index,geom,lambda,theta,inc,'mx',15,'my',15);

% Plot field
figure, VisuFieldFMM(abs(E_tm(:,1)).^2,geom.x,geom.y,geom.z)
```

