HOFEM-AIRBUS 2D Emulation

Group of Radiofrequency, Electromagnetism, Microwaves and Antennas (GREMA)
http://grema.webs.tsc.uc3m.es/

Departamento de Teoría de la Señal y Comunicaciones (TSC)
Universidad Carlos III de Madrid, Spain

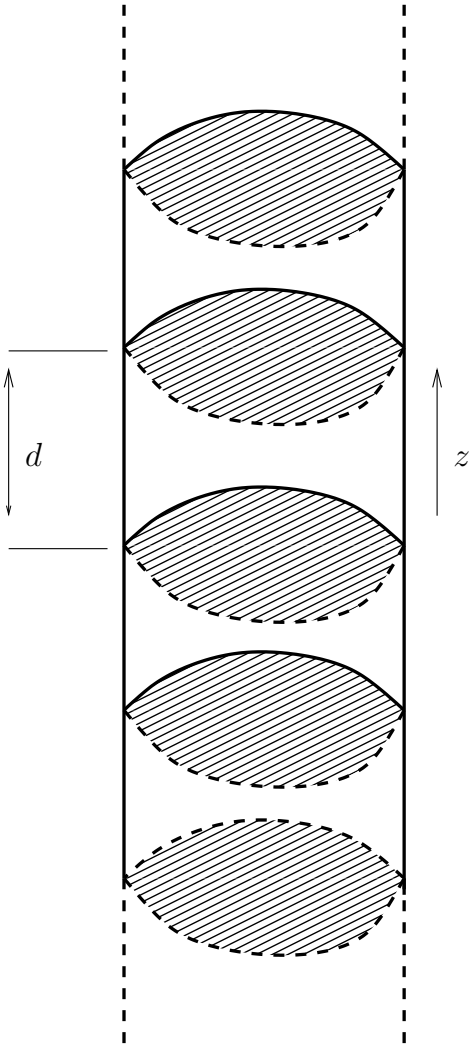
ilio García-Castillo legcasti@ing.uc3m.es, Adrián Amor aamor@ing.uc3m.es, Sergio Llorente sollor

uc3m | Universidad Carlos III de Madrid

GREMA
RADIOFREQUENCY, ELECTROMAGNETICS MICROWAVES & ANTENNAS

We start considering the use of `Green3D` or `Green2D` the scattering of inifinite cylinders using 3D simulator on a section ("slice") of the cylinder

HOFEM 2D Emulation



3D Green's Function

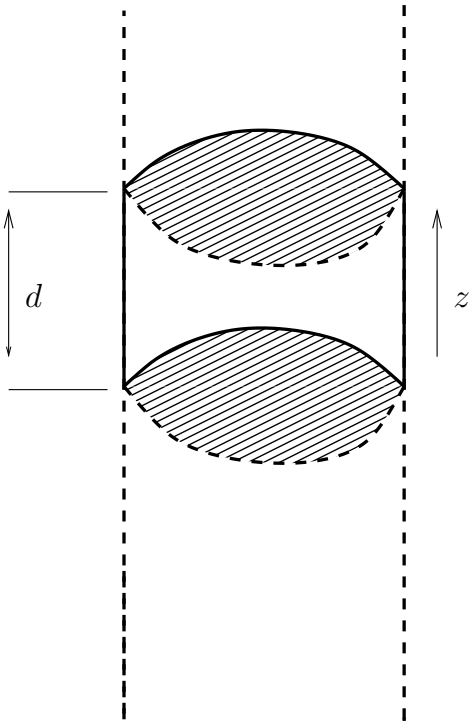
- Sum of infinite contributions (from each repeated cell)
- Ewald acceleration needed
  - It works great, though

$$\text{Green}(R) = \frac{1}{4\pi} \sum_{n=-\infty}^{\infty} \frac{e^{-jkR_n}}{R_n}$$

with

$$R_n = |\mathbf{r} - \mathbf{r'}_n| \, , \mathbf{r'}_n = \mathbf{r'}_0 + nd$$

HOFEM 2D Emulation (cont.)



2D Green's Function

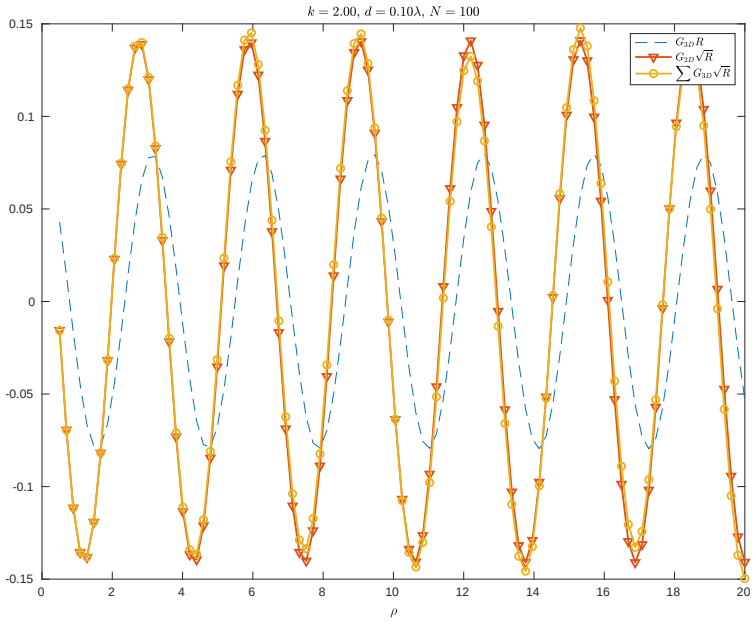
- No summation needed

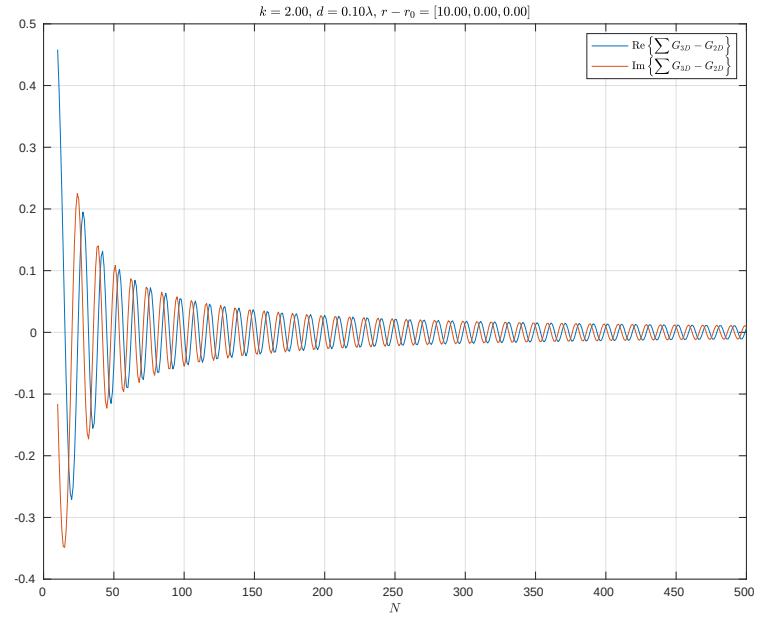
$$\text{Green}(\rho) = \frac{1}{4j} H_0^{(2)}(k|\rho|)$$

Near Field (FE-IIEE loop)

Sanity Checks (MATLAB)

- Equivalence of infinite sum of 3D Green's function with 2D Green's function on the unit cell

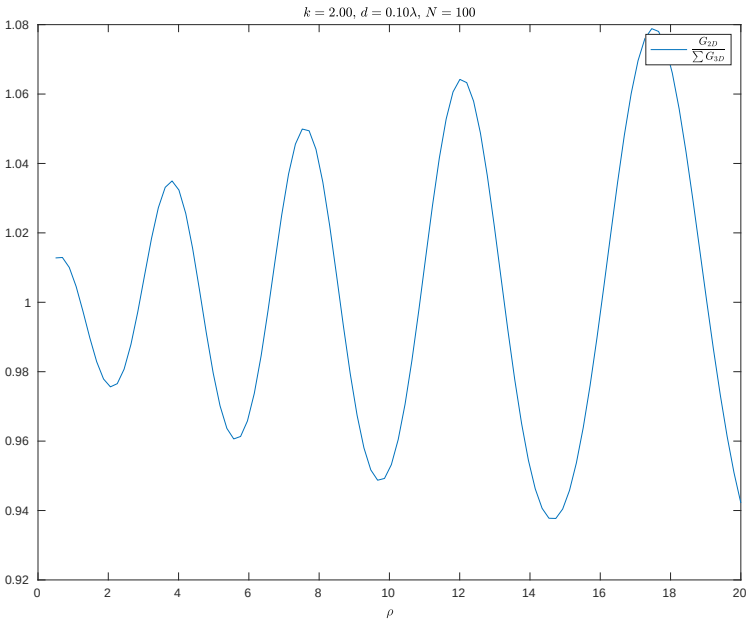


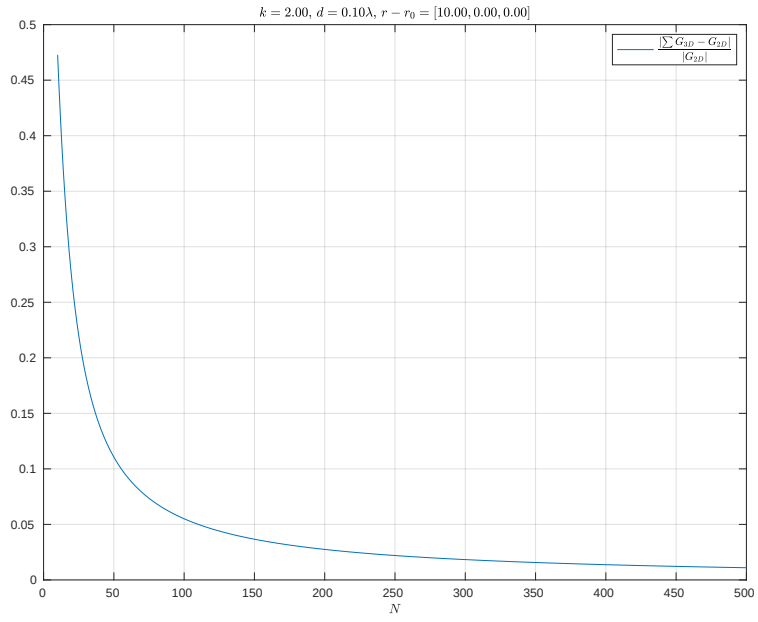


Near Field (FE-IIEE loop) (cont.)

Sanity Checks (MATLAB)

Equivalence of infinite sum of 3D Green's function with 2D Green's function on the unit cell





Near Field (FE-IIEE loop) (cont.)

HOFEM Implementation

- Coded
- Tested

```
    [#] EXECUTING IIEE TRUNCATION METHOD...

    ------------------------------------------------------------

    [#] CALCULATING SCATTERING FIELD...

    [#] SCATTERING FIELD CALCULATED IN
0.190 [sec]

    ------------------------------------------------------------

    [#] CALCULATING SCATTERING RHS...

    [#] SCATTERING RHS CALCULATED IN
0.014 [sec]

    ------------------------------------------------------------

            Iteration:
1. (err 1.19E-02)            0.235 [sec]

    ------------------------------------------------------------

    [#] CALCULATING SCATTERING FIELD...
```

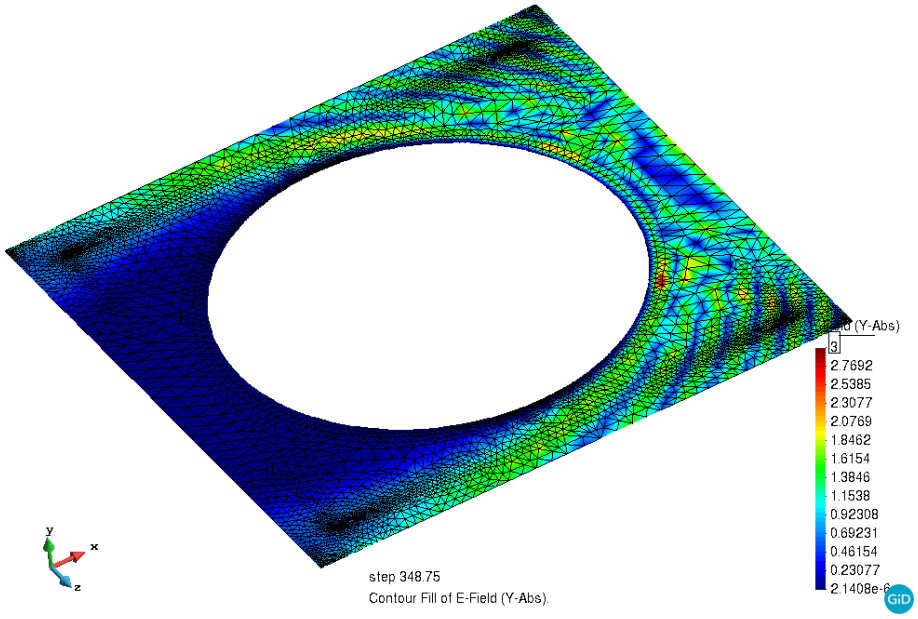
Near Field (FE-IIEE loop) (cont.)

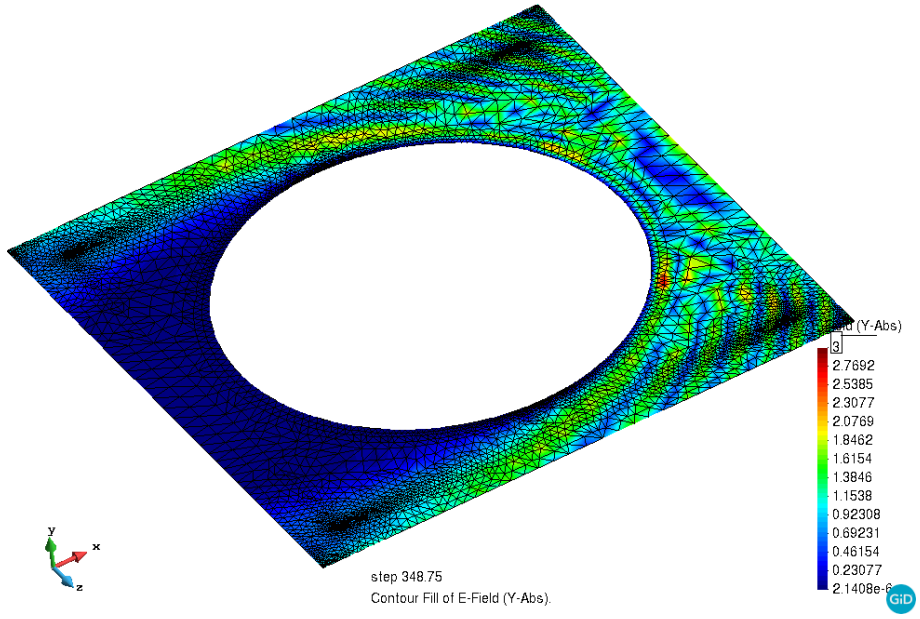
Code output (terminal):

```
Iteration:  1. (err 1.19E-02)
Iteration:  2. (err 8.31E-05)
Iteration:  3. (err 7.23E-07)
Iteration:  4. (err 6.10E-09)
```

```
Iteration:  1. (err 1.50E-02)
Iteration:  2. (err 1.02E-04)
Iteration:  3. (err 8.28E-07)
Iteration:  4. (err 7.00E-09)
```

Near Field (FE-IIEE loop) (cont.)





Far Field (postprocess)

Equivalence of Green's Functions for Far Field

- Can we use an infinite sum of 3D Green's far-field functions to model the 2D Green's far-field?
  - NO
  - and it makes sense!

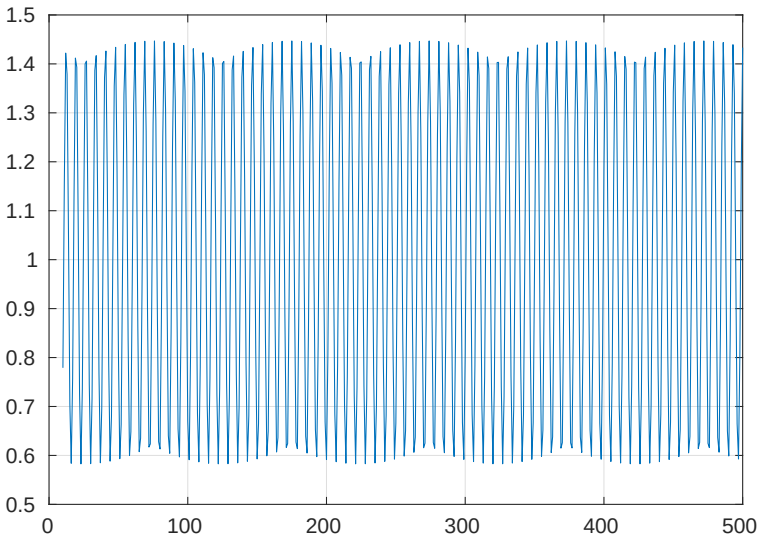
Far Field as Fourier Transform

The Fourier Transform of a constant current contribution along z is a delta function in the spatial frequency domain, i.e., variable θ:

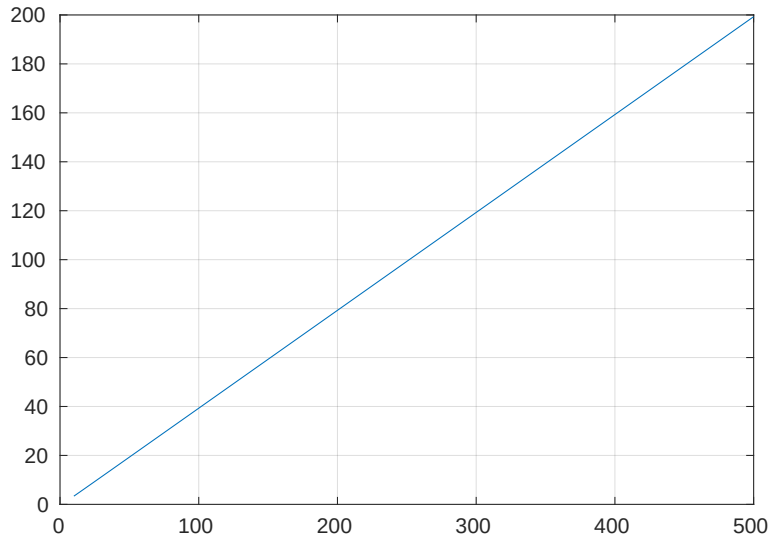
$$\text{Far Field}(\theta, \phi) = \delta(\theta - \pi/2) F(\phi) \tag{1}$$

Far Field (postprocess) (cont.)

$\theta \neq 90°$: Oscillations



$\theta = 90°$: Divergence



Far Field (postprocess) (cont.)

HOFEM Implementation

- Coded (in progress)
- Tested (in progress)

Rethinking...

AIRBUS: *hay un chico con nosotros que resuelve problemas de scattering de cilindros y no usa función de Hankel. Usa Green3D para calcular el campo (lejano) de scattering de una rodaja y le funciona.*

- Further study of differences between `Green2D` and `Green3D`
  - Obvious (after a maturation process) for far field
  - Not so obvious for near field (i.e., in its impact in FE-IIEE loop)

Far Field (postprocess)

Equivalence of Green's Functions for Far Field

- Can we use an infinite sum of 3D Green's far-field functions to model the 2D Green's far-field?
  - NO
  - and it makes sense!
  - No need of infinite sum (far field `Green3D`/`Green2D` is enough)
    - Note that far field versions of `Green3D` (with $r = \rho$) and `Green2D` are equal up to a constant

Far Field as Fourier Transform

The Fourier Transform of a constant current contribution along z is a delta function in the spatial frequency domain, i.e., variable θ:

$$\text{Far Field}(\theta, \phi) = \delta(\theta - \pi/2) F(\phi) \tag{2}$$

Far Field (postprocess) (cont.)

Green2D vs Green3D

$F(\phi)$ is the same (after normalization) considering asymptotic expressions of Green3D and Green3D

$$H_0^{(2)}(k\rho) \overset{\rho\to\infty}{\approx} \sqrt{\frac{2j}{\pi k\rho}}\, e^{-jk\rho} \qquad \frac{e^{-jkR}}{4\pi R} \overset{r\to\infty}{\approx} \frac{e^{-jkr}}{4\pi r} e^{jkr'\cos\psi}$$

Near Field (FE-IIEE loop)

Code output (terminal):

- From previous meeting

`Green2D`

```
Iteration:   1. (err 1.50E-02)
Iteration:   2. (err 1.02E-04)
Iteration:   3. (err 8.28E-07)
Iteration:   4. (err 7.00E-09)
```

`Green3D`

```
Iteration:   1. (err 1.19E-02)
Iteration:   2. (err 8.31E-05)
Iteration:   3. (err 7.23E-07)
Iteration:   4. (err 6.10E-09)
```

How can it work? Note we used `Green3D` itself (only one slide was considered)

Near Field (FE-IIEE loop) (cont.)

Code output (terminal):

- Large $S' - S$

`Green2D`

```
Iteration:  1. (err 1.50E-02)
Iteration:  2. (err 1.02E-04)
Iteration:  3. (err 8.28E-07)
Iteration:  4. (err 7.00E-09)
```

`Green3D`

```
Iteration:  1. (err 1.19E-02)
Iteration:  2. (err 8.31E-05)
Iteration:  3. (err 7.23E-07)
Iteration:  4. (err 6.10E-09)
```

- Small $S' - S$

`Green2D`

```
Iteration:  1. (err 5.16E-01)
Iteration:  2. (err 1.22E-01)
Iteration:  3. (err 2.53E-02)
Iteration:  4. (err 4.34E-03)
Iteration:  5. (err 6.86E-04)
Iteration:  6. (err 1.21E-04)
Iteration:  7. (err 2.46E-05)
Iteration:  8. (err 5.06E-06)
```

`Green3D`

```
Iteration:  1. (err 7.65E-02)
Iteration:  2. (err 6.08E-03)
Iteration:  3. (err 5.46E-04)
Iteration:  4. (err 5.05E-05)
Iteration:  5. (err 4.59E-06)
Iteration:  6. (err 4.24E-07)
Iteration:  7. (err 4.14E-08)
Iteration:  8. (err 4.13E-09)
```

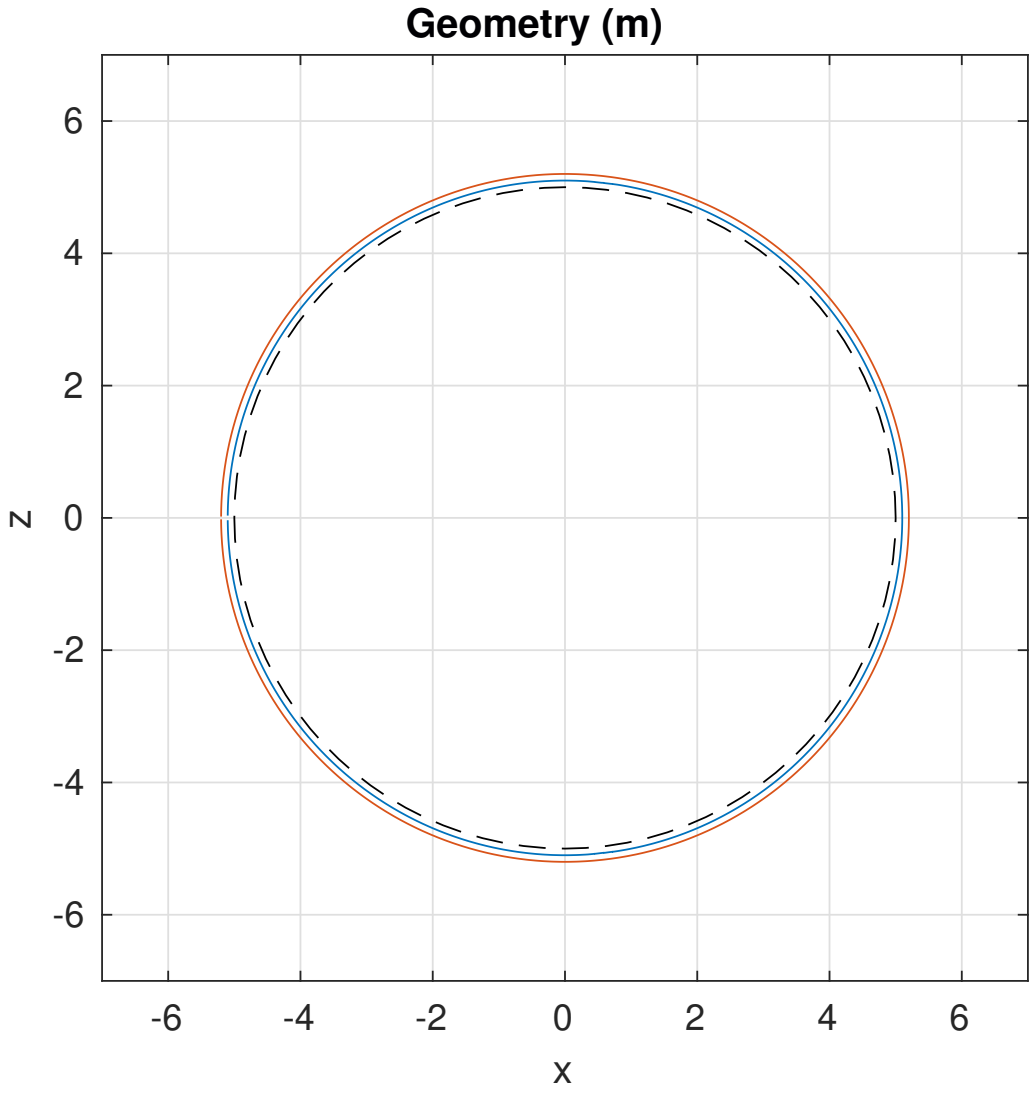
Green2D vs Green3D

Comparison with analytical solution

- Most results displayed correspond to TM and E field
- H field (i.e., RotE) identical conclusions
- By duality (satisfied by HOFEM) TE results "should" be identical (tested).
- Most results shown here correspond to both S' and S conformal to the cylinder (circular boundary). Analogous results/conclusions are obtained with rectangular boundaries for S' and S (and combinations of them).

Green2D vs Green3D (cont.)

Comparison with analytical solution



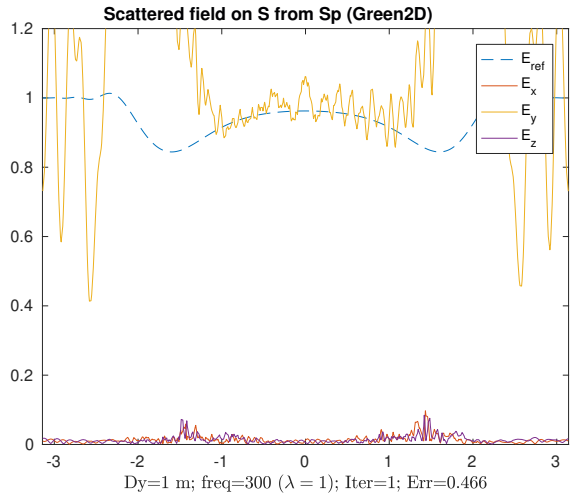
- 300 MHz
- Small $S - S'$
- Thick slice

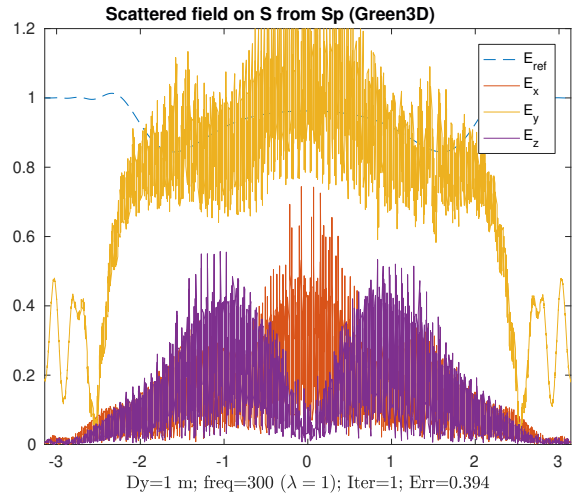
Green2D vs Green3D (cont.)

Comparison with analytical solution

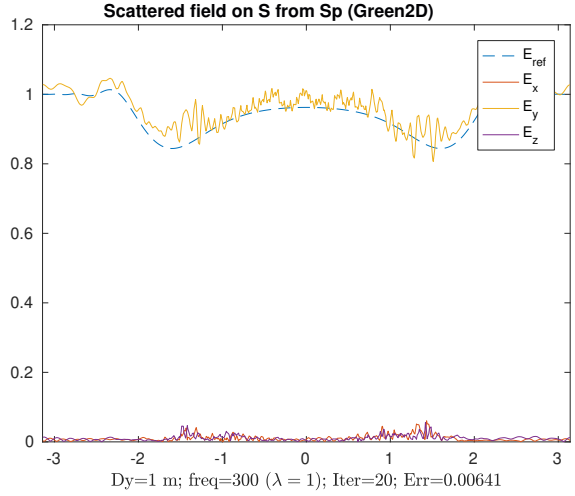
Evolution scattered electric field (S' over S)

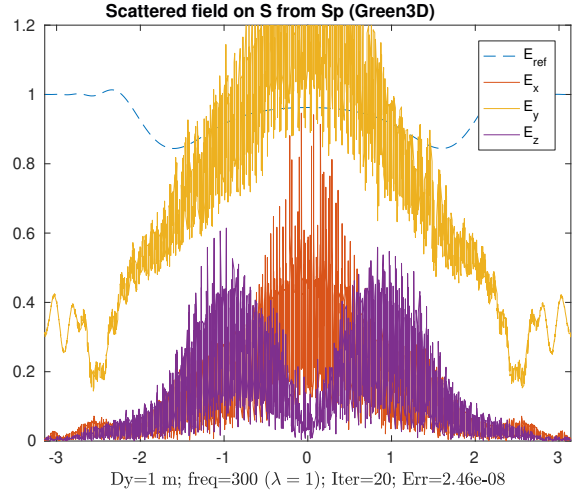
Iteration 1:





Iteration 20:



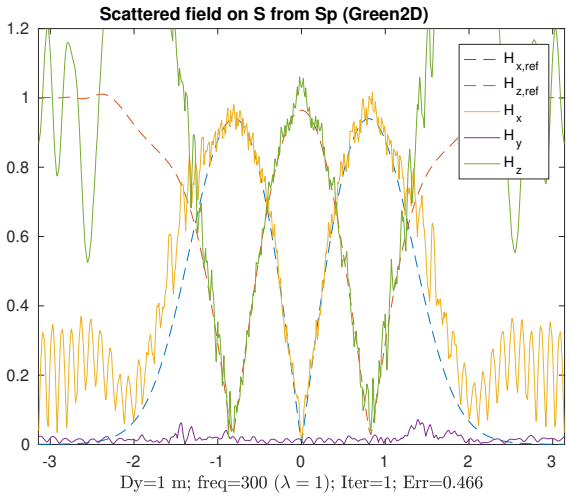


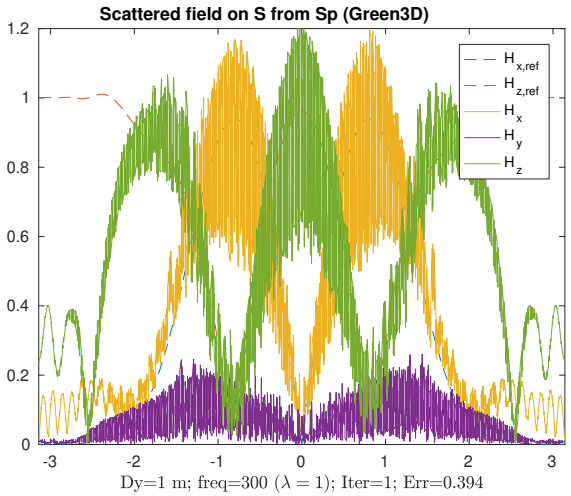
Green2D vs Green3D (cont.)

Comparison with analytical solution

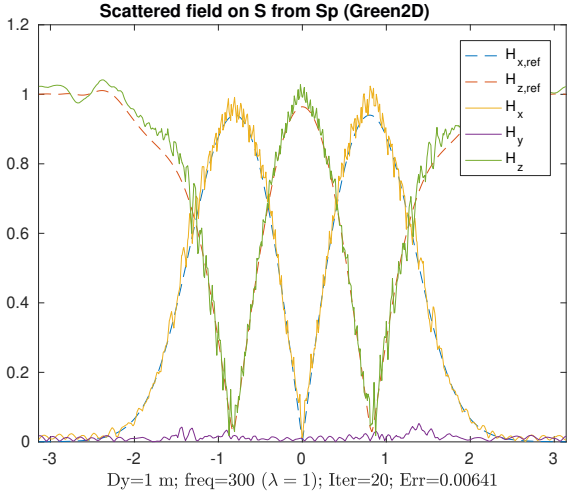
Evolution scattered magnetic field (S' over S)

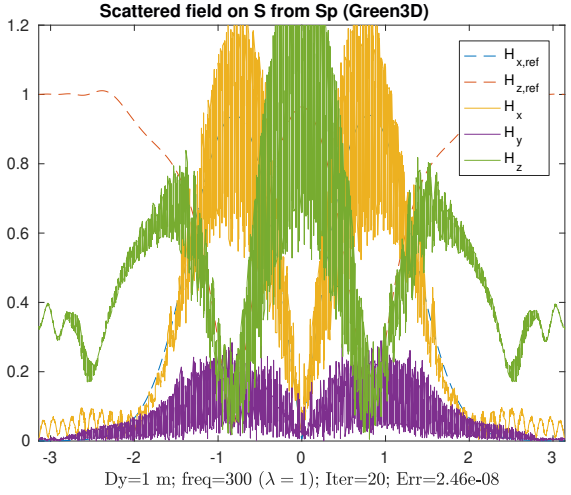
Iteration 1:





Iteration 20:



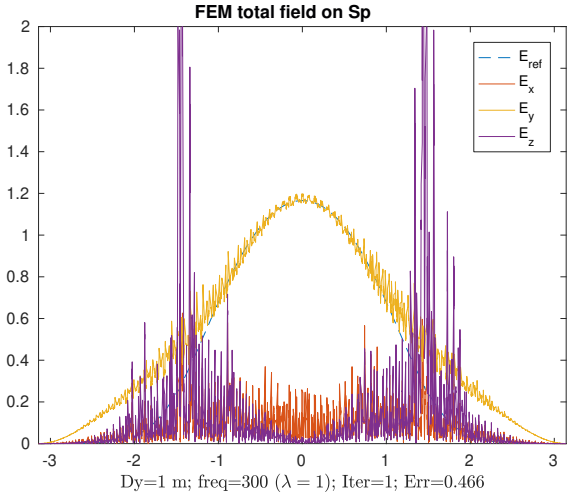


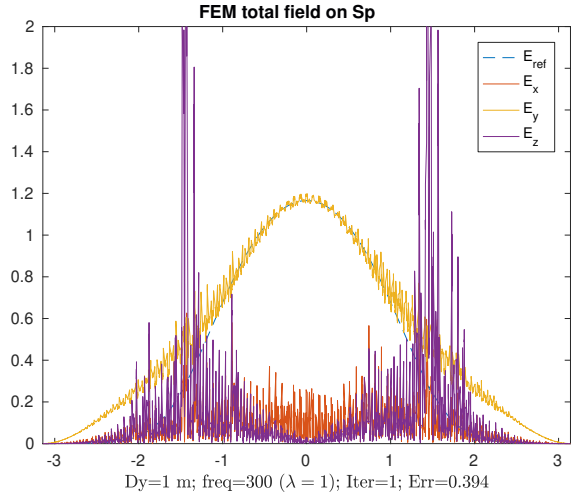
Green2D vs Green3D (cont.)

Comparison with analytical solution

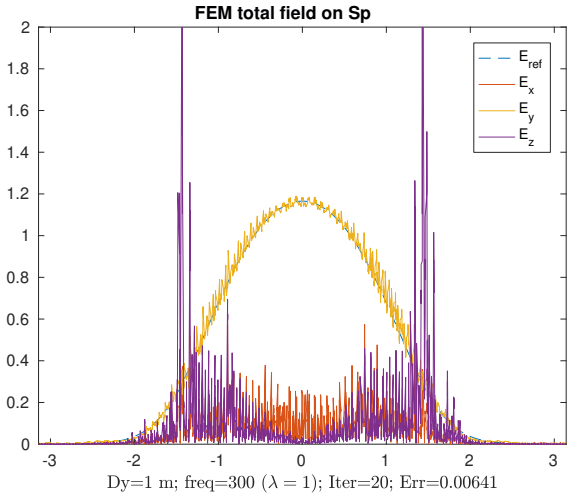
Evolution electric field on S'

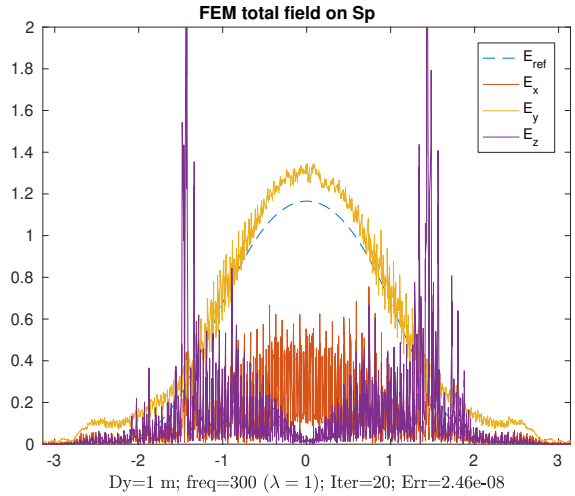
Iteration 1:





Iteration 20:





Green2D vs Green3D (cont.)

Comparison with analytical solution

- Higher error with Green3D
  - This is a case with small distance $S - S'$ and large thickness
  - Nevertheless, the error is always significantly higher than Green2D
- Green3D does not converge to right solution
- Components E_x, E_z, H_y are not null
  (issue of $(*)$ certainly added confusion on the debugging/verification of the code)
  - Much higher levels with Green3D
  - Green3D (the Green's function itself) generates non null E_x, E_z, H_y from E_z on S'
  - Green2D (the Green's function itself) does not generate any E_x, E_z, H_y
    - ⋆ Numerically, non-zero levels of E_x, E_z, H_y are generated because numerical FEM solution has non zero levels of E_x, E_z, H_y (tested!)
- Numerical noise always present due to discretization
  - Clearly visible in this case
  - Decreases with finer discretization —FEM mesh— (tested!)

$(*)$ Issue of GiD/HOFEM-GUI representation of fields on surface

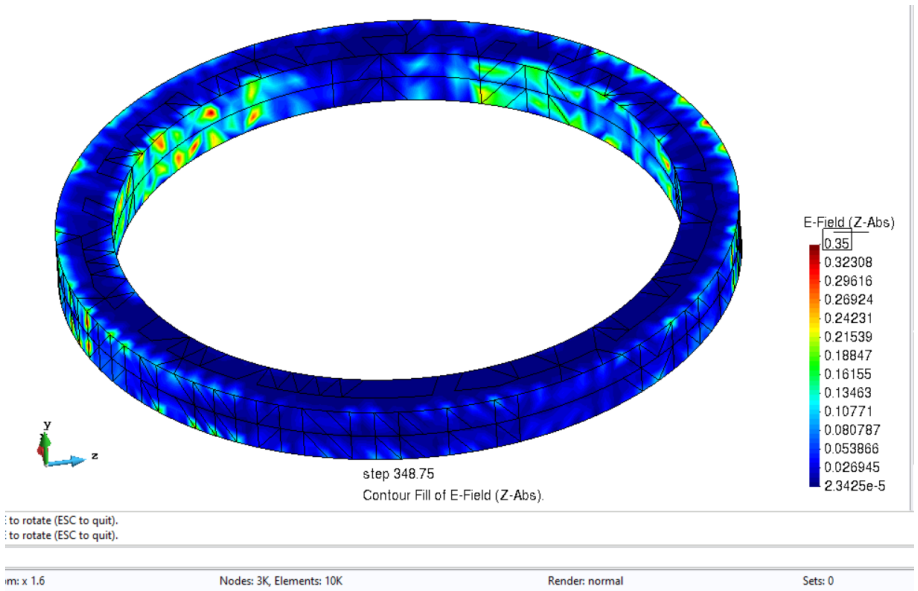
Rethinking...

(*) Issue of GiD/HOFEM-GUI representation of fields on surface

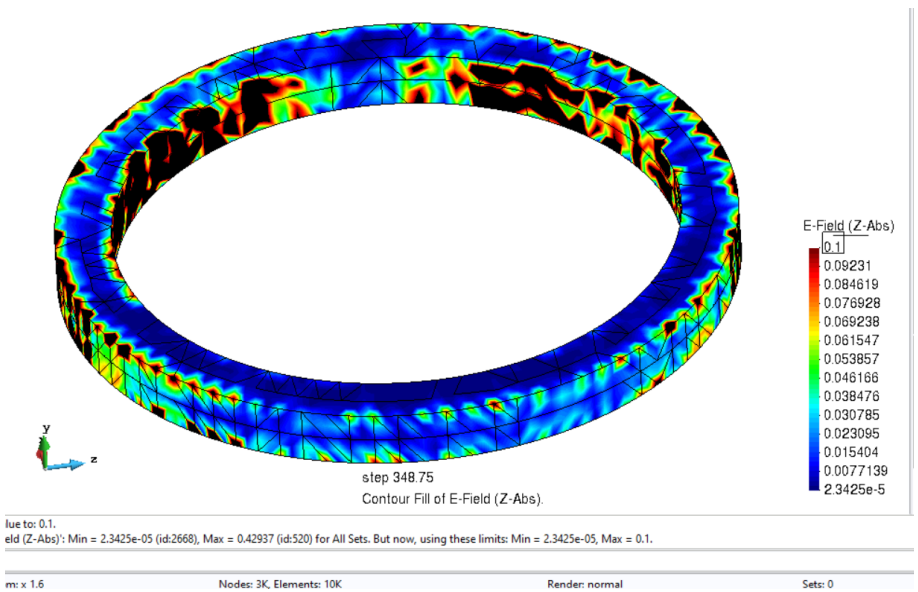
GiD/HOFEM-GUI Field Representation on Surfaces

Examples with PEC surfaces

- E_z should be null on the caps (top and bottom $y = \text{cte}$)



$|E_z|$ (default color scale)

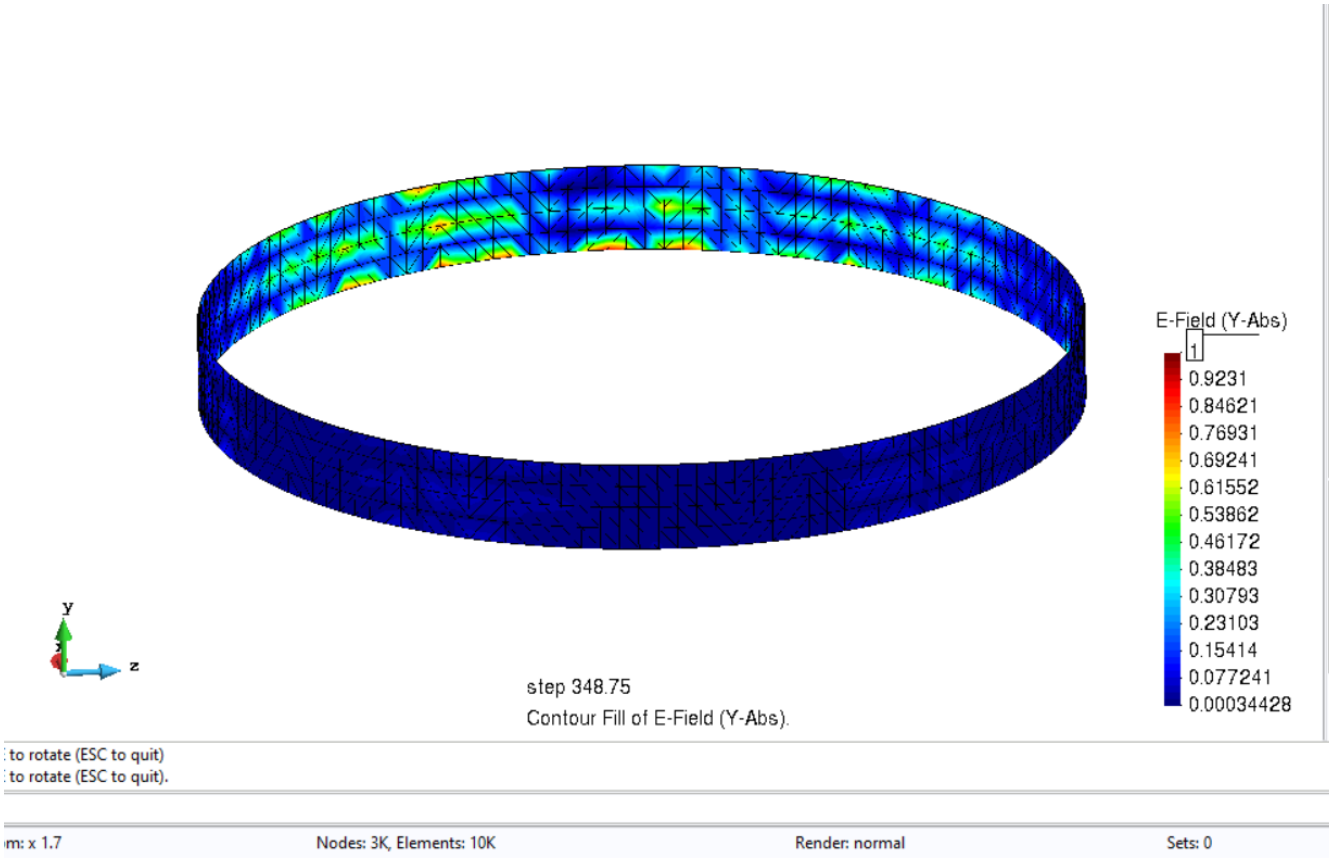


$|E_z|$ (saturated color scale)

GiD/HOFEM-GUI Field Representation on Surfaces (cont.)

Examples with PEC surfaces

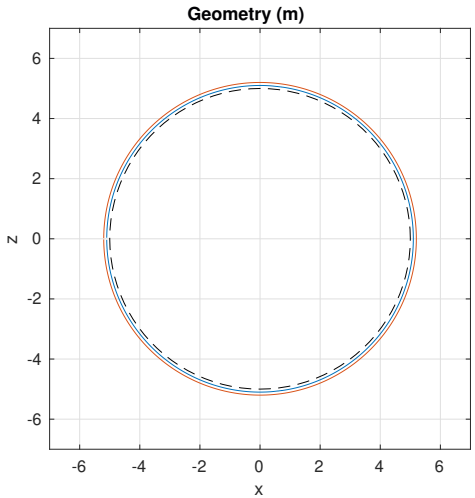
- E_y should be null on the cylinder "sides" (i.e., on the PEC cylinder)

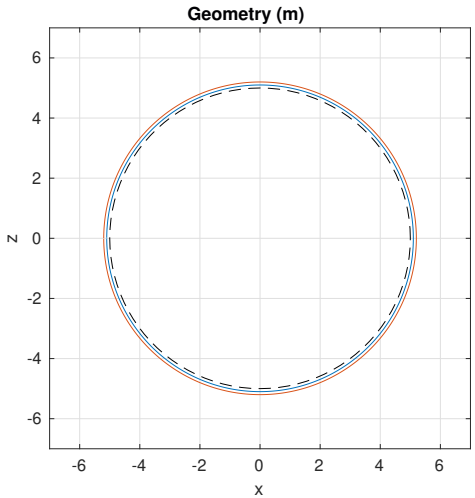


$|E_y|$ (only PEC cylinder surface shown)

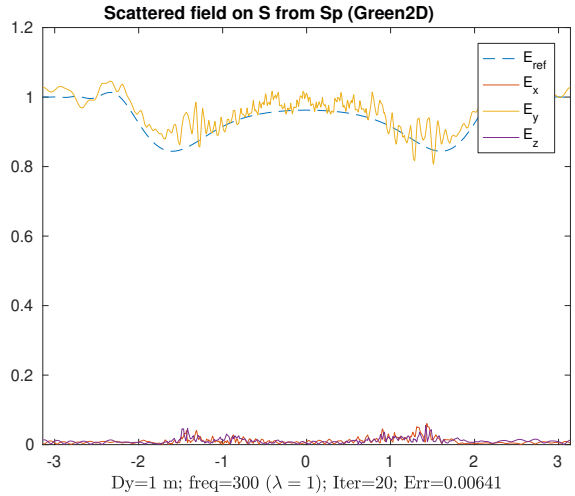
Effect of "Thickness"

Scattered near field —Note error in figure captions: thickness is 0.05 λ for the thin slice—

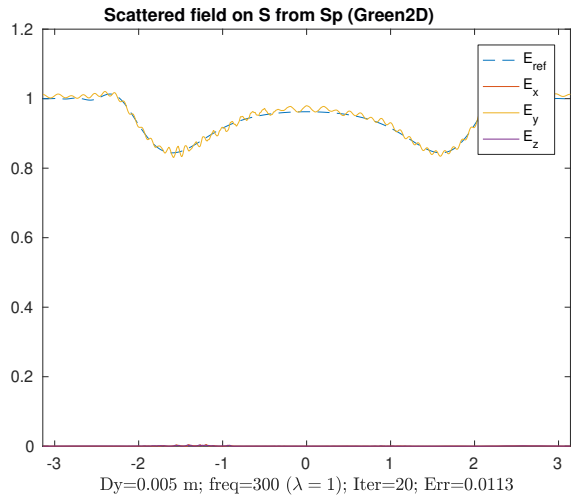




Green2D

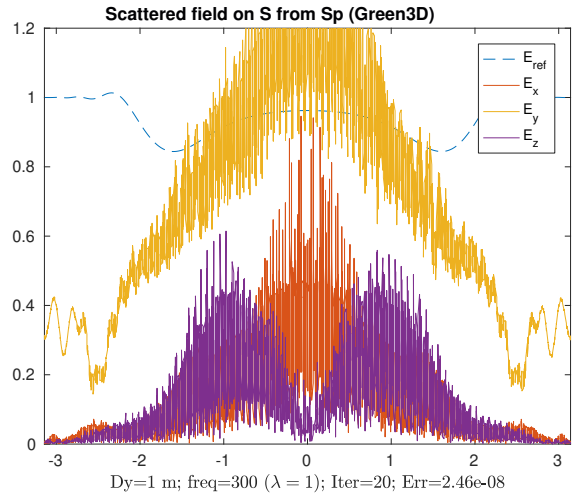


Dy=1 m; freq=300 ($\lambda = 1$); Iter=20; Err=0.00641

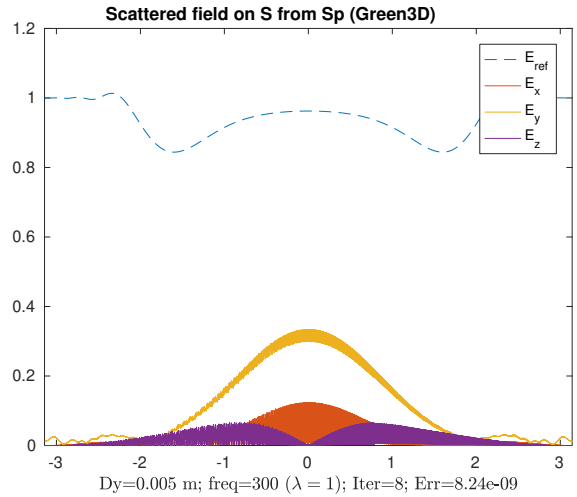


Dy=0.005 m; freq=300 ($\lambda = 1$); Iter=20; Err=0.0113

Green3D



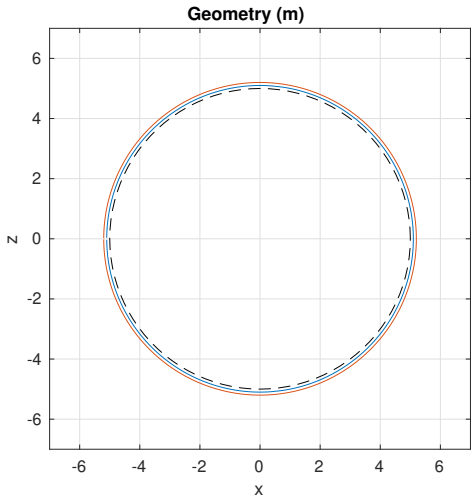
Dy=1 m; freq=300 ($\lambda = 1$); Iter=20; Err=2.46e-08

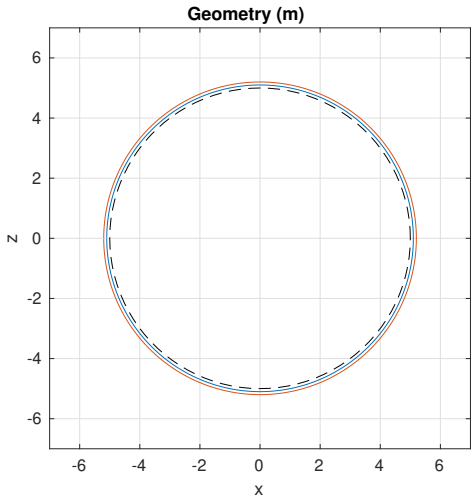


Dy=0.005 m; freq=300 ($\lambda = 1$); Iter=8; Err=8.24e-09

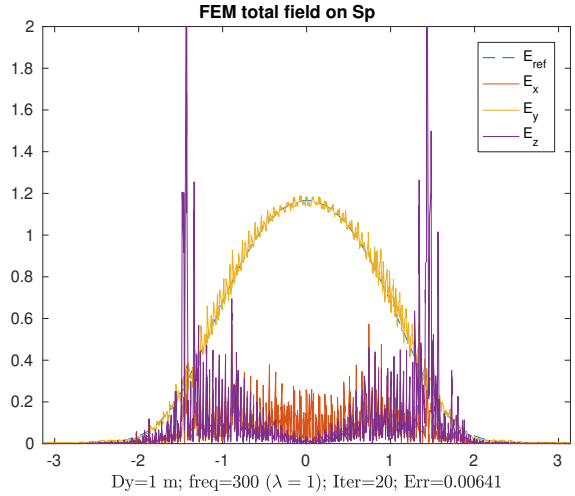
Effect of "Thickness"

FEM solution —Note error in figure captions: thickness is 0.05 λ for the thin slice—

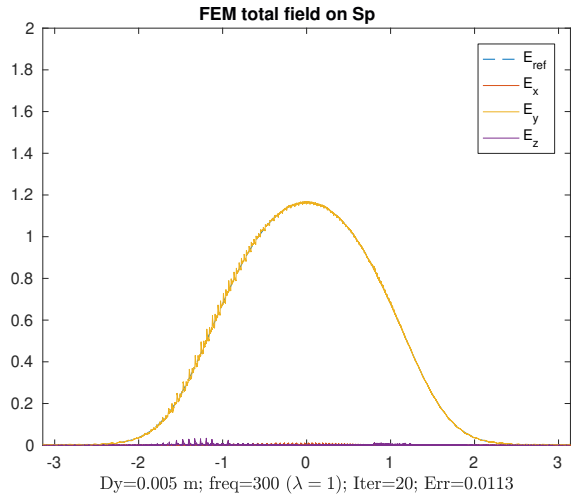




Green2D

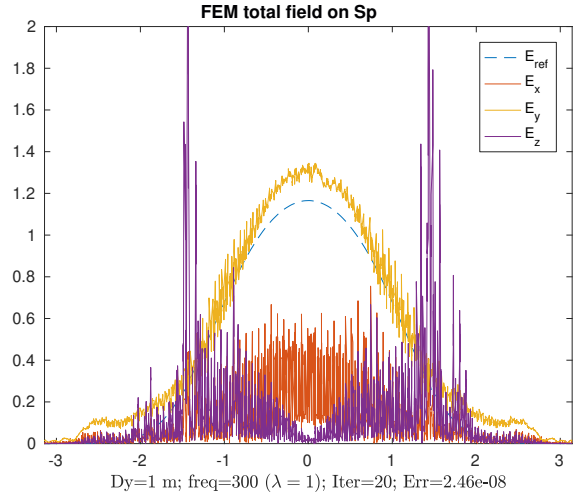


Dy=1 m; freq=300 ($\lambda = 1$); Iter=20; Err=0.00641

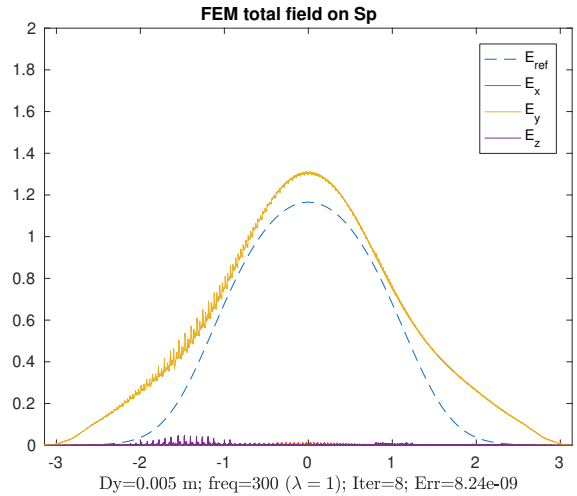


Dy=0.005 m; freq=300 ($\lambda = 1$); Iter=20; Err=0.0113

Green3D



Dy=1 m; freq=300 ($\lambda = 1$); Iter=20; Err=2.46e-08



Dy=0.005 m; freq=300 ($\lambda = 1$); Iter=8; Err=8.24e-09

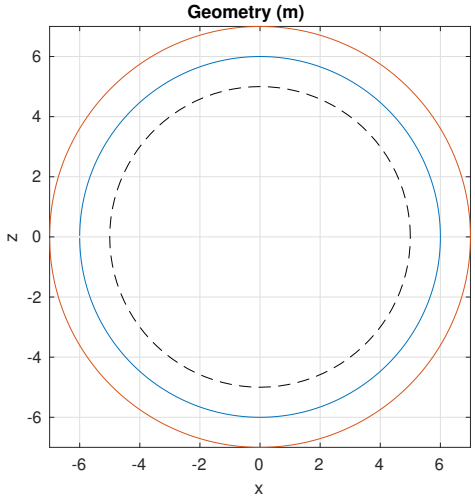
Effect of "Thickness" (cont.)

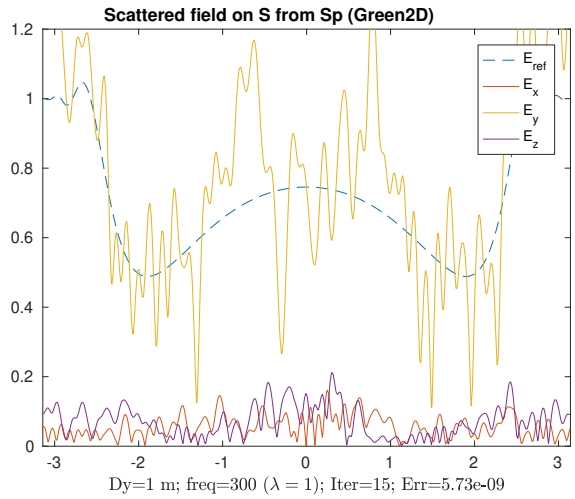
FEM solution —Note error in figure captions: thickness is 0.05 λ for the thin slice—

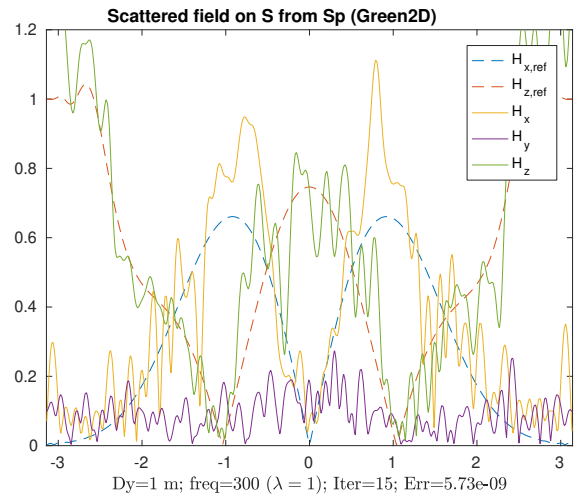
- The thickness is the main factor for error with `Green3D`
- `Green2D` behaves great!
- Always lower error with thin slices

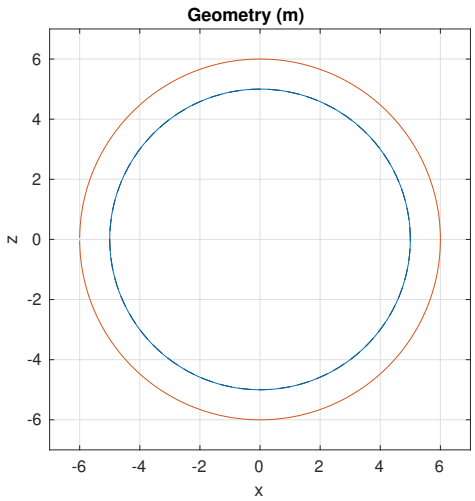
Other Effects (paranoic mode)

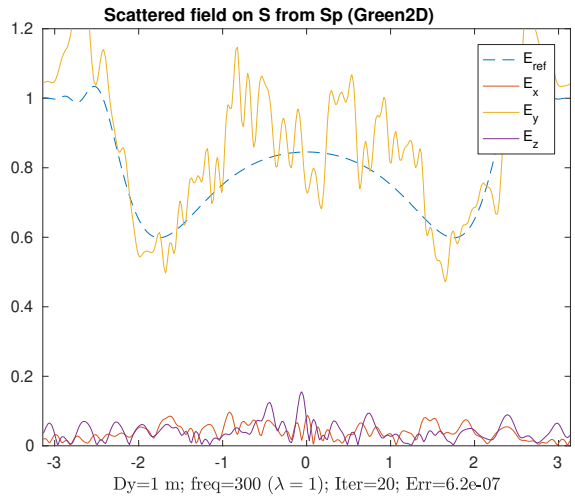
Effect of S' on PEC cylinder

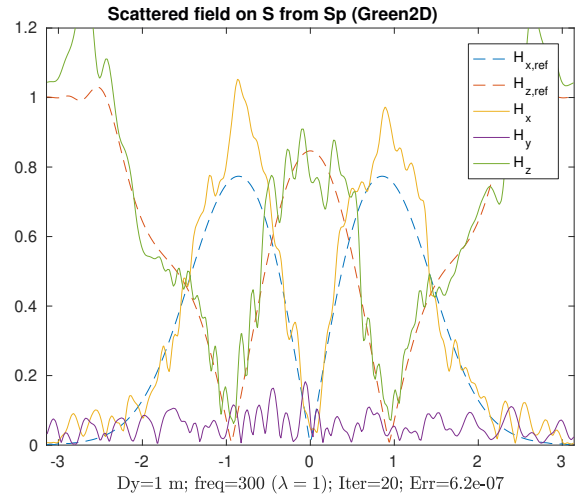






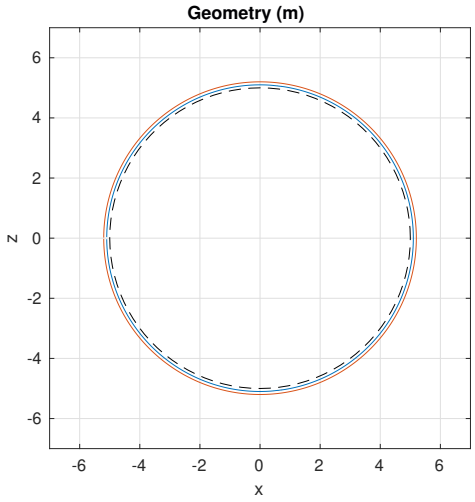


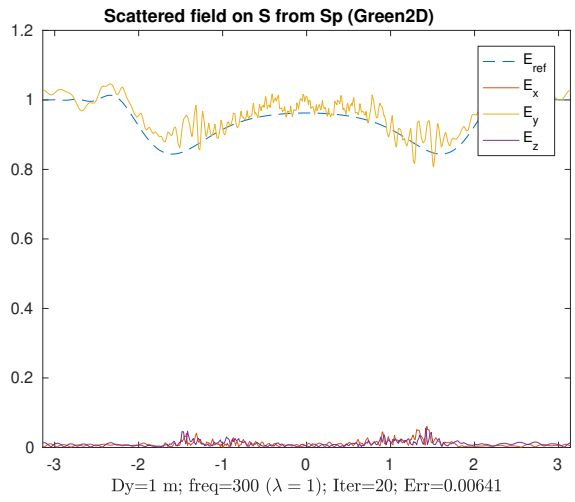


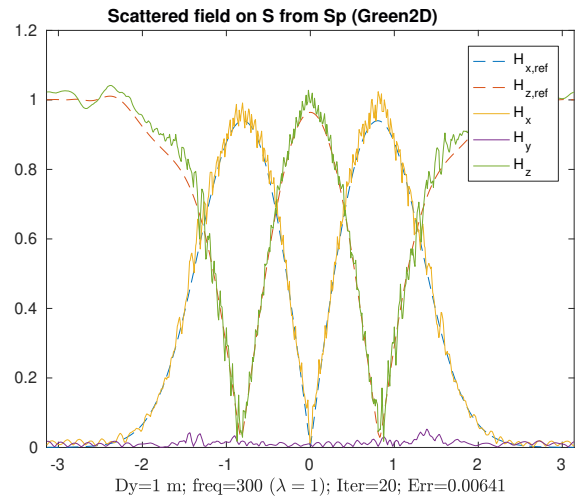


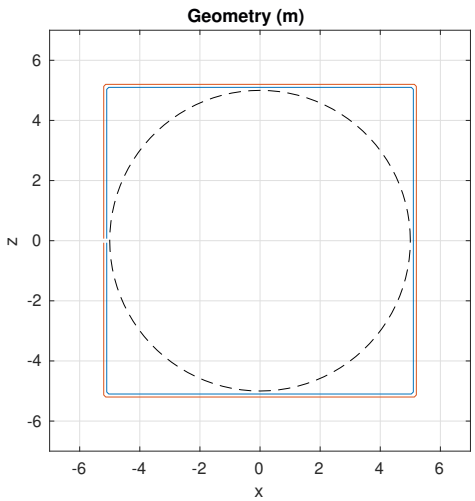
Other Effects (paranoic mode)

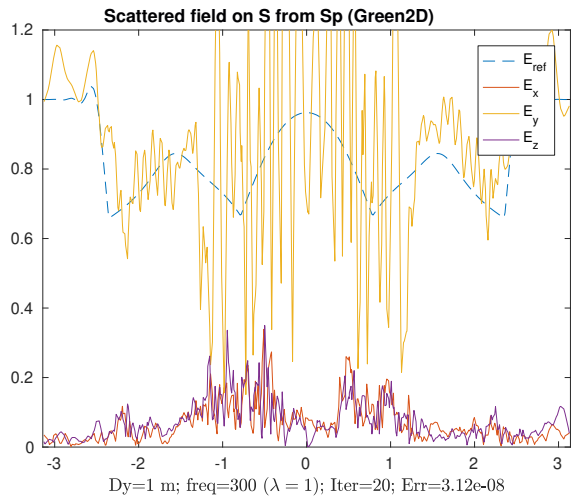
Effect of S', S Curved

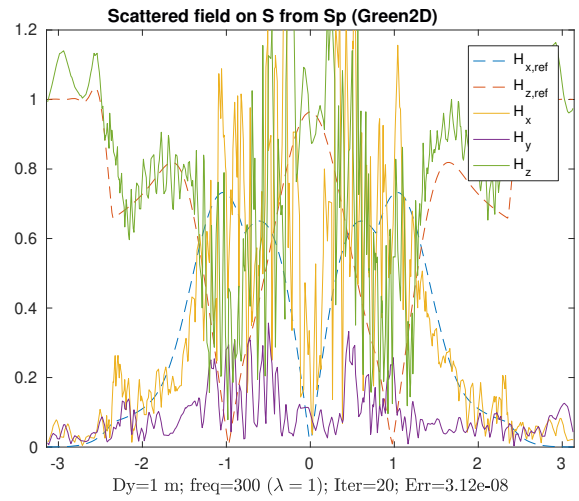












Conclusions

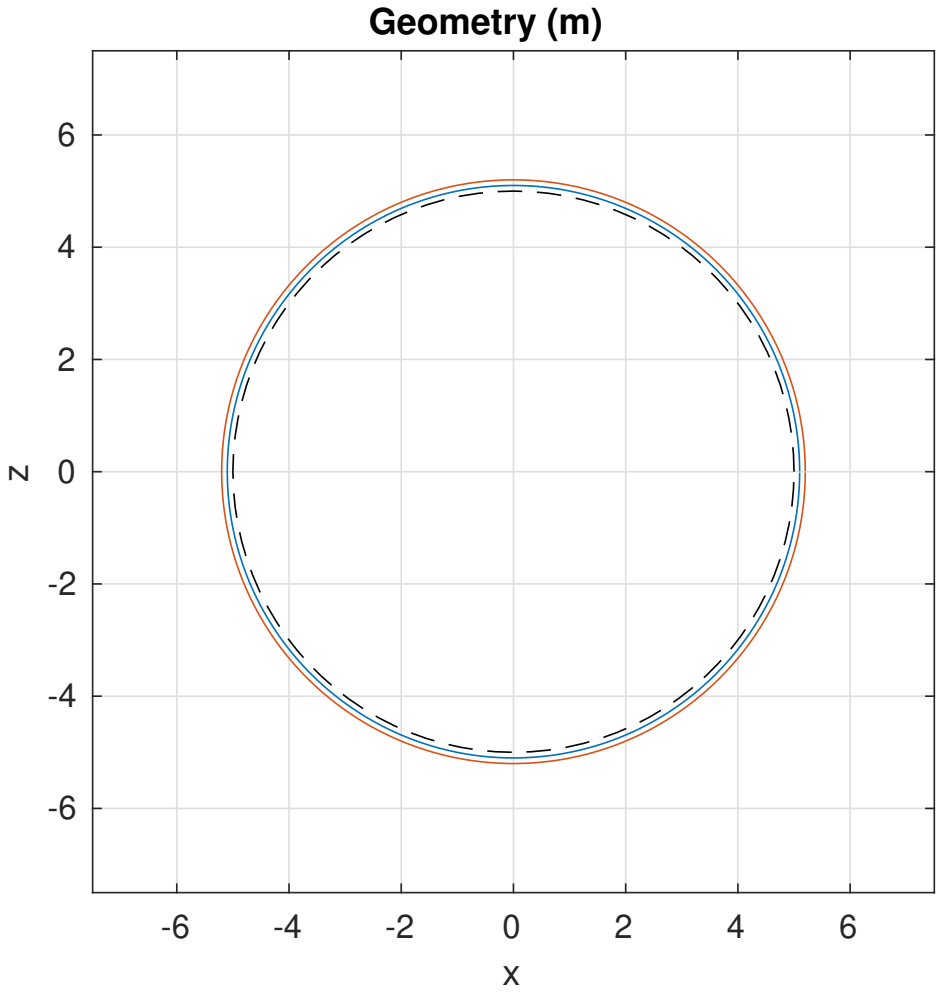
- Green3D itself not valid for 2D Emulation
  - FE-IIEE converges but
    - Quality of solution far from great
    - It may guide you to wrong solution
- Need to activate contributions of slice replicas along direction of cylinder (translation symmetry) Green3D_Ewald

From now on we only use `Green2D`

- We check TE polarization for PEC cylinder
- We check for the dielectric coated cylinder

Check TE polarization

Comparison with TM results

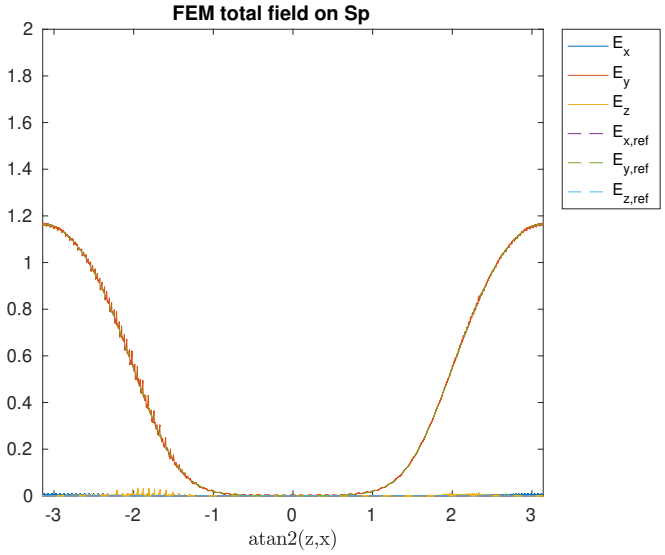


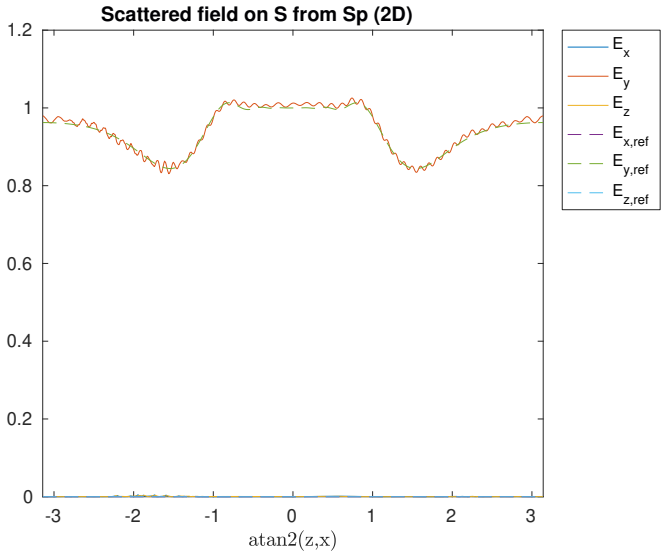
- Freq: 300 MHz. ($\lambda = 1$ m)
- Thickness: 0.05 m

Check TE polarization (cont.)

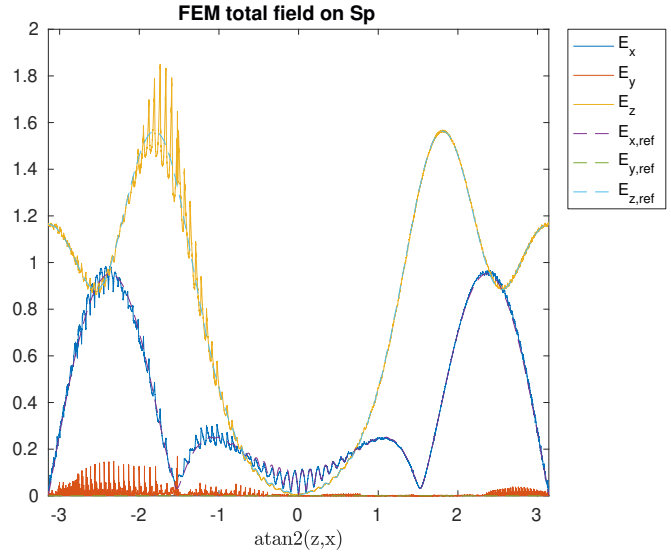
Comparison with TM results

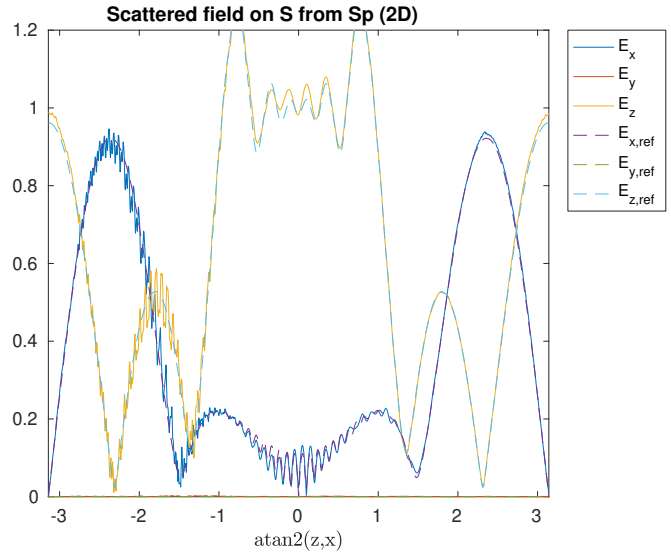
TM incident field





TE incident field

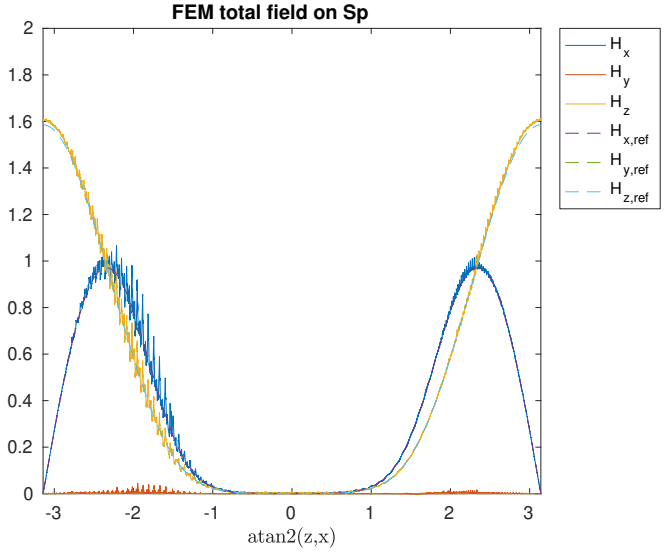


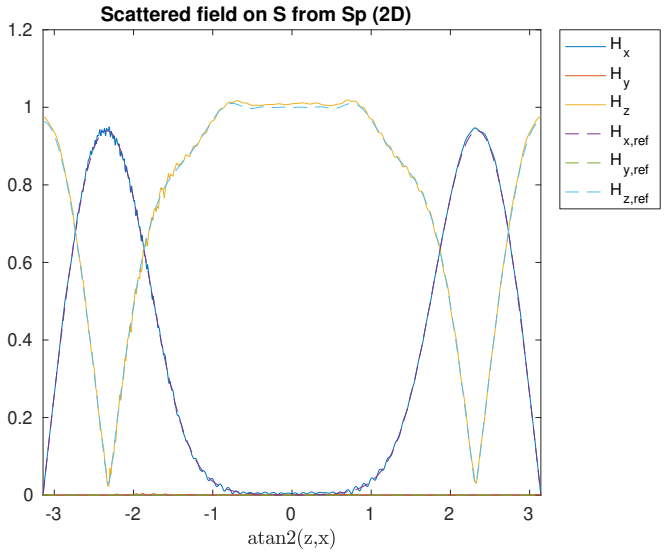


Check TE polarization (cont.)

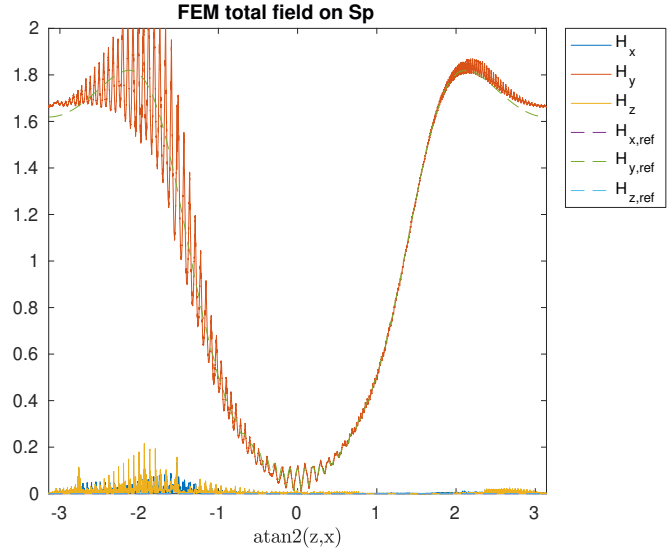
Comparison with TM results

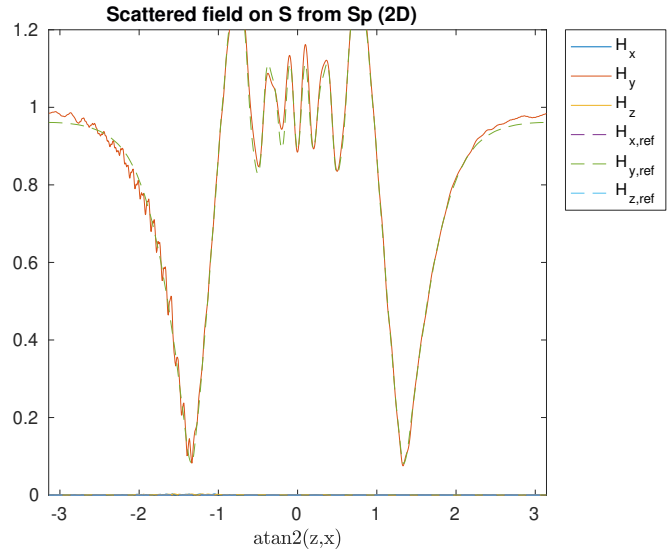
TM incident field





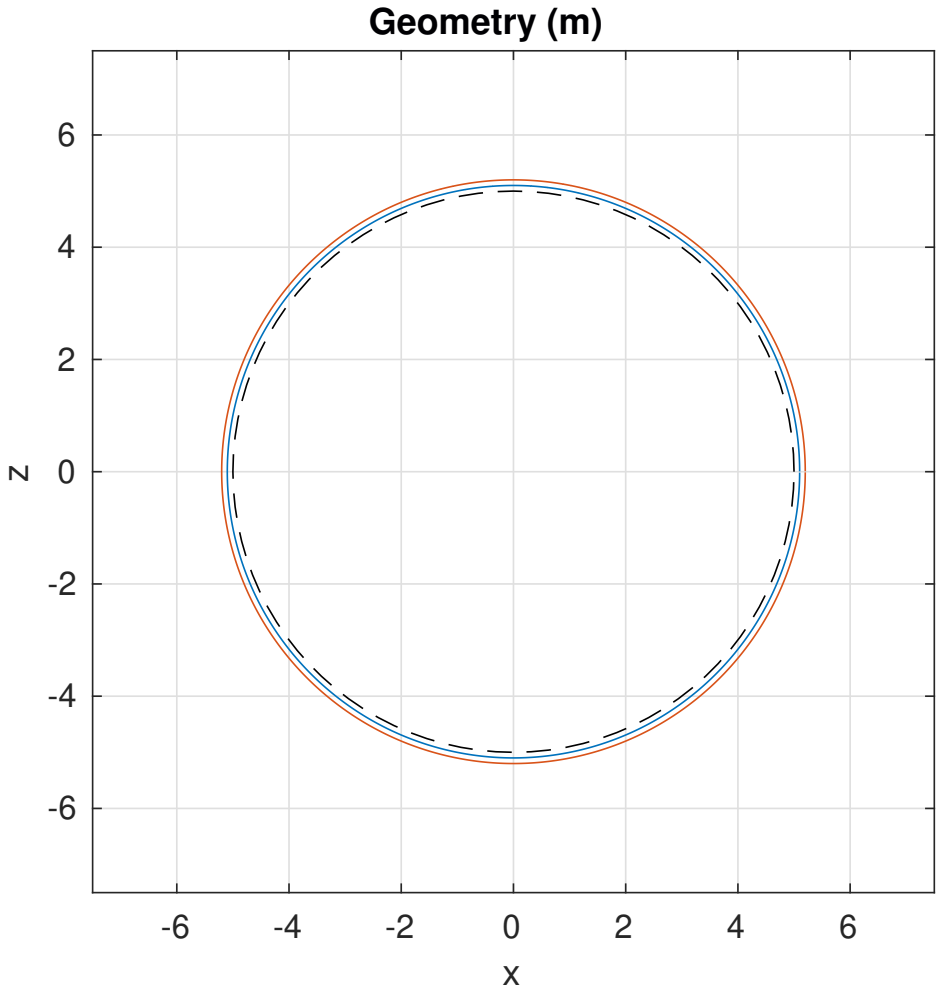
TE incident field





Dielectric Coated Cylinders

Comparison with the analytical solution

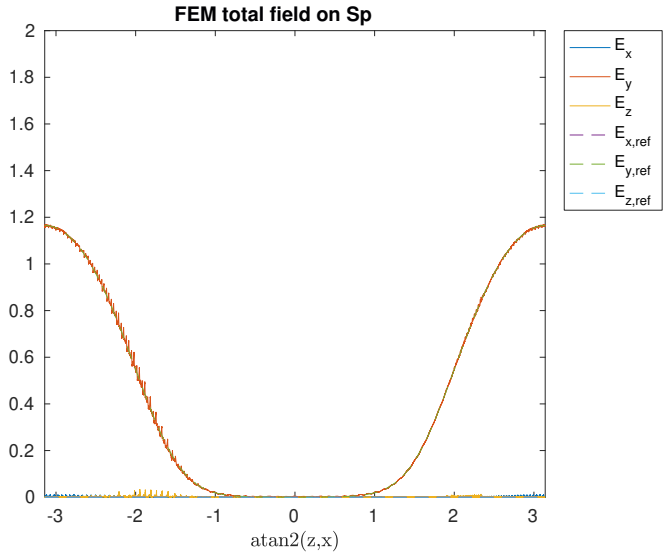


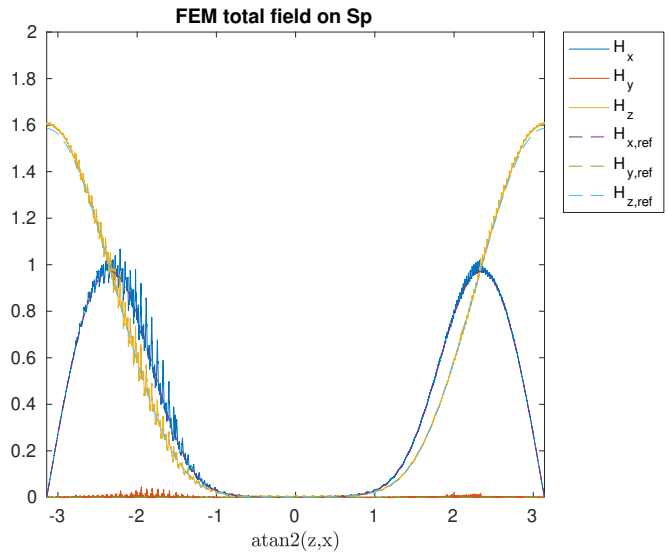
- Freq: 300 MHz. ($\lambda = 1$ m)
- Thickness: 0.05 m
- Dielectric substrate between S and S' with relative permittivity ε_r.

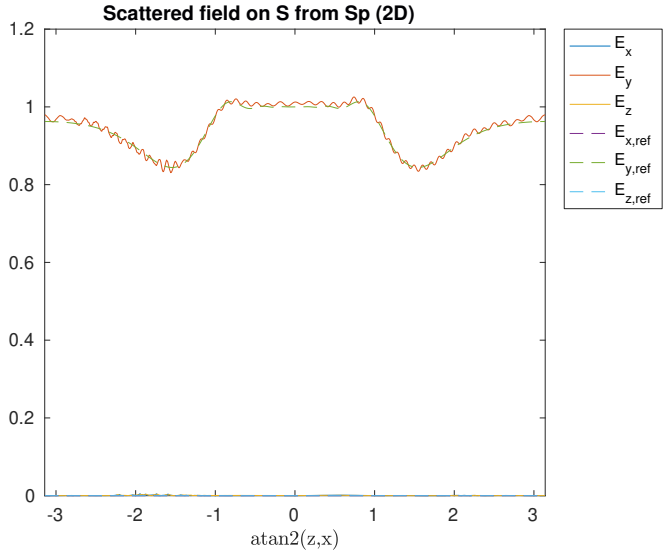
Dielectric Coated Cylinders (cont.)

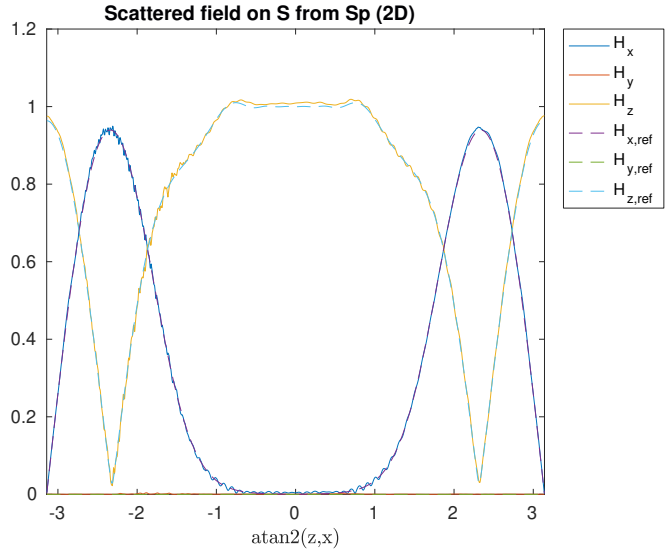
Comparison with the analytical solution

TM incident field $\varepsilon_r = 1$





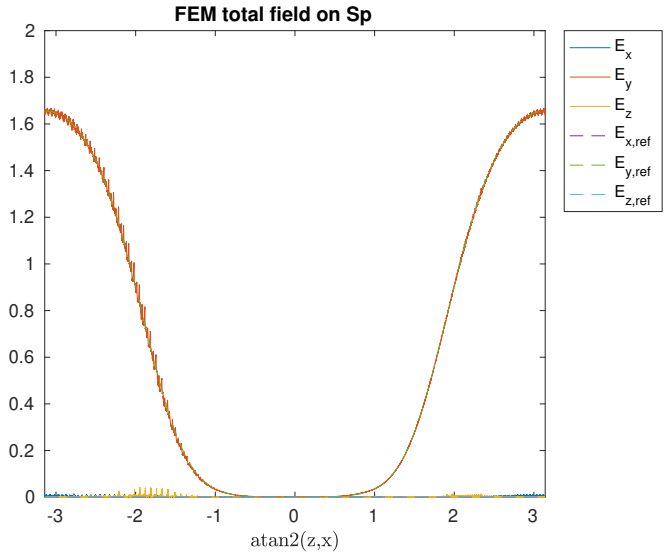


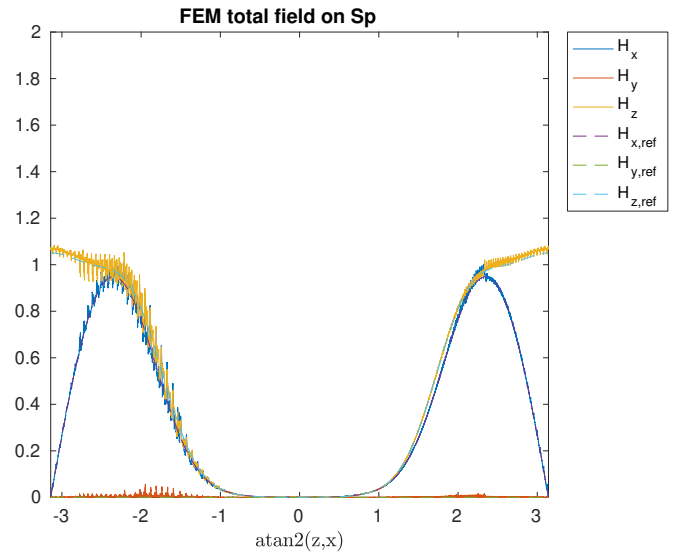


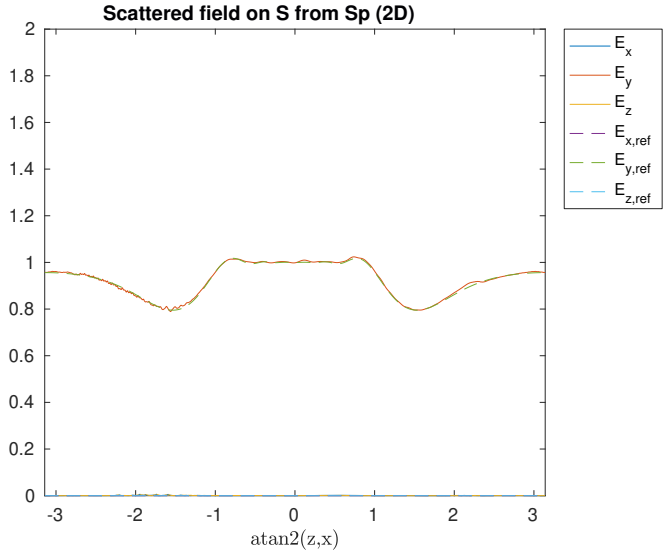
Dielectric Coated Cylinders (cont.)

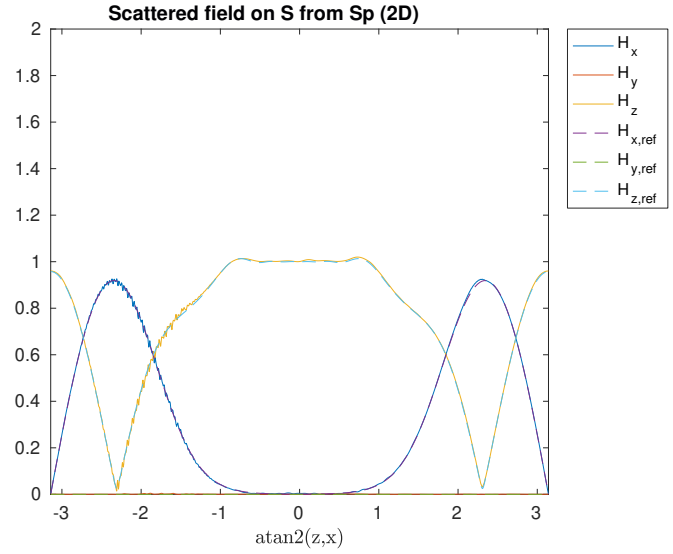
Comparison with the analytical solution

TM incident field $\varepsilon_r = 4$





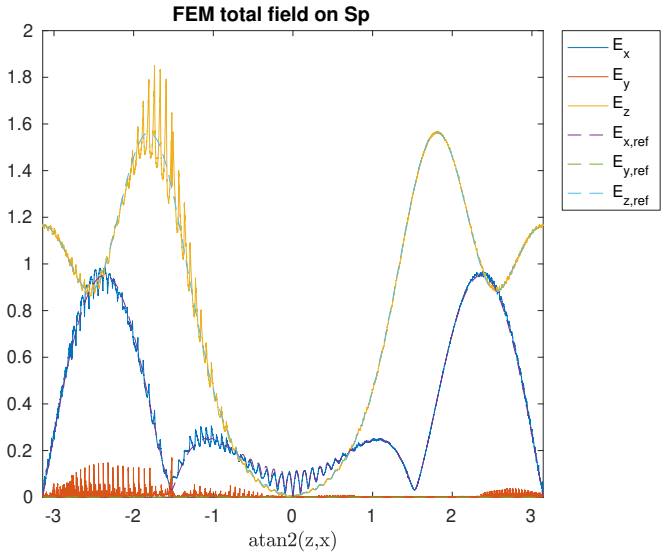


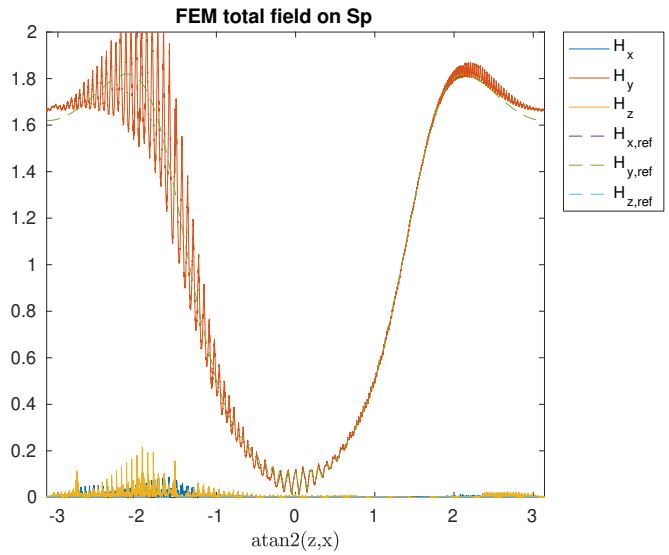


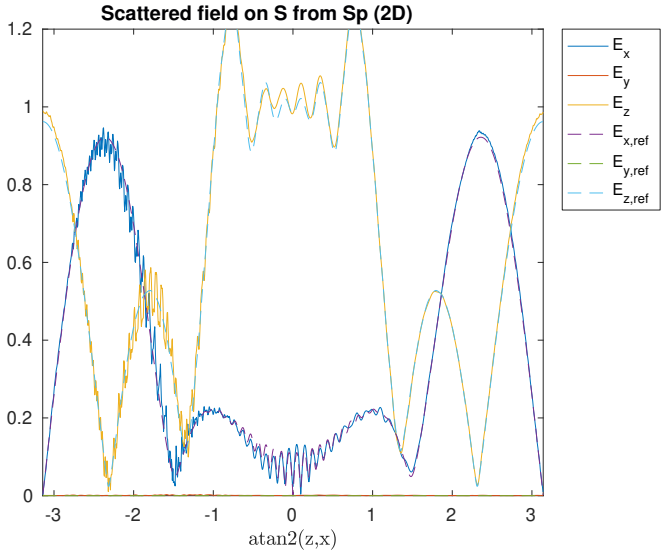
Dielectric Coated Cylinders (cont.)

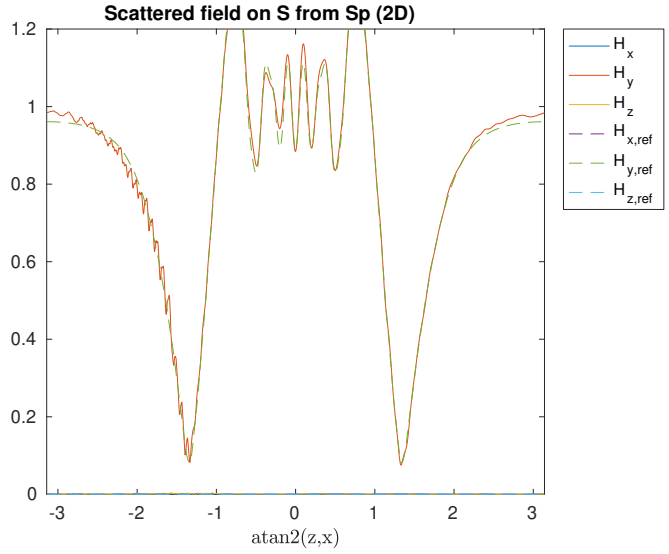
Comparison with the analytical solution

TE incident field $\varepsilon_r = 1$





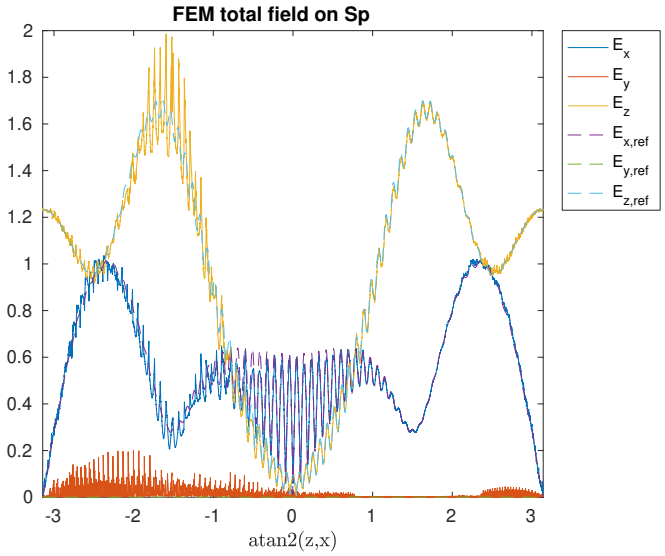


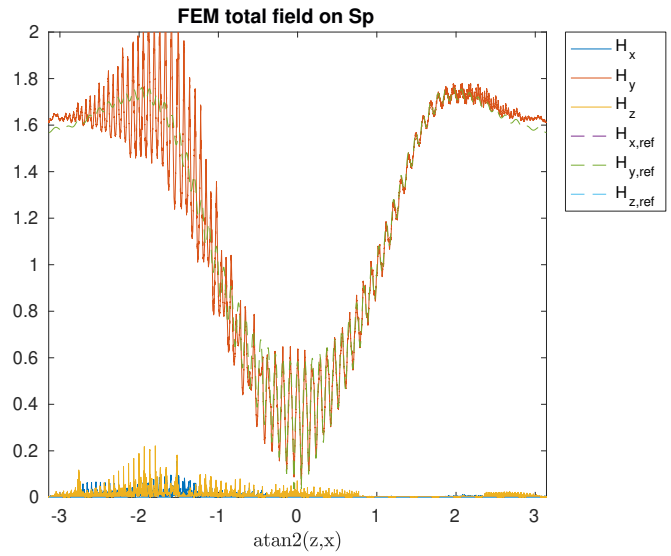


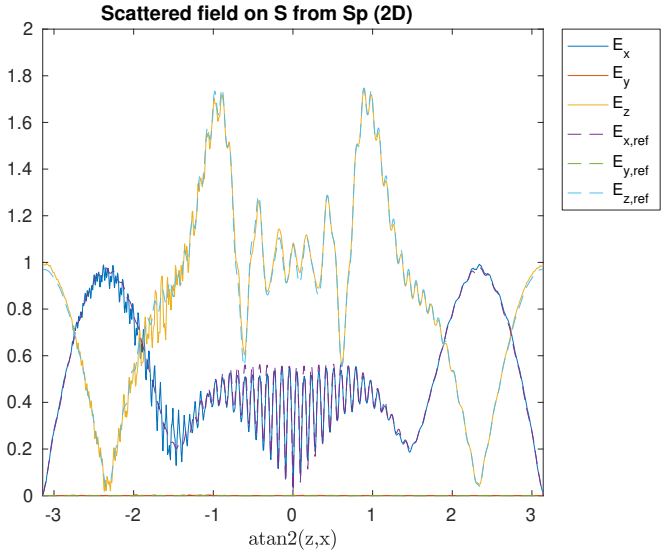
Dielectric Coated Cylinders (cont.)

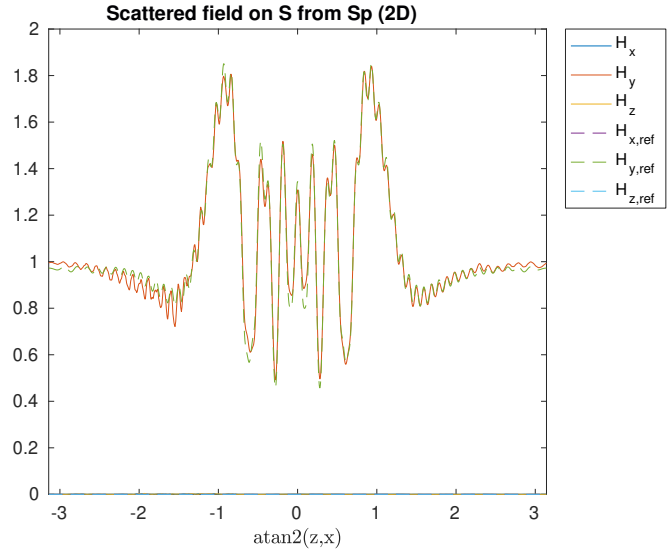
Comparison with the analytical solution

TE incident field $\varepsilon_r = 1.5$









Green3D_Ewald

Ewald Method in HOFEM

where s is a complex variable. The periodic Green's function can be written in two parts by using the previous identity and splitting the path integration at the parameter E as

$$G_p(\mathbf{r},\mathbf{r}_s) = G_{p1}(\mathbf{r},\mathbf{r}_s) + G_{p2}(\mathbf{r},\mathbf{r}_s) \tag{5.41}$$

where $G_{p1}(\mathbf{r},\mathbf{r}_s)$ is given by

$$G_{p1}(\mathbf{r},\mathbf{r}_s) = \frac{1}{4\pi}\sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty} e^{-j(k_x m \mathrm{D_x} + k_y n \mathrm{D_y})} \times \frac{2}{\sqrt{\pi}}\int_0^{\mathrm{E}} e^{\left(-R_{mn}^2 s^2 + \frac{k_0^2}{4s^2}\right)} ds \tag{5.42}$$

and $G_{p2}(\mathbf{r},\mathbf{r}_s)$ is given by

$$G_{p2}(\mathbf{r},\mathbf{r}_s) = \frac{1}{4\pi}\sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty} e^{-j(k_x m \mathrm{D_x} + k_y n \mathrm{D_y})} \times \frac{2}{\sqrt{\pi}}\int_{\mathrm{E}}^{\infty} e^{\left(-R_{mn}^2 s^2 + \frac{k_0^2}{4s^2}\right)} ds \tag{5.43}$$

```
!Distancia de la celda unidad en X
 Dx =  &
       SQRT(DOT_PRODUCT(PBC_structure%offset_vector(
 !Distancia de la celda unidad en Y
 Dy =  &
       SQRT(DOT_PRODUCT(PBC_structure%offset_vector(

 E = SQRT(PI/Dx/Dy)

 !Sum using Ewald transformation
 DO m=-2,2
    DO n=-2,2

    !Calcular alpha_mn
    alpha_mn = SQRT(CMPLX((PI*m/Dx)**2 +  &
          (PI*n/Dy)**2 + (PI*m/Dx)*Kx + &
    (PI*n/Dy)*Ky +  &
          (1.0_DBL/4.0_DBL)*(Kx**2+Ky**2-K**2),0.0_DB
```

Ewald 1D periodicity

- Naively we thought if was simply setting either $m = 0$ or $n = 0$
- But it is NOT . . . see next section on Ewald 1D

Details on Ewald (1D periodicity)

Ewald sum

- Technique for summing contribution from an infinite set of sources (in this case along z axis):

$$\sum_n G(R_n) = \frac{1}{4\pi} \sum_n e^{-j(k\cos\theta)nd} \frac{e^{-jkR_n}}{R_n}$$

where $R_n = \sqrt{(x-x')^2 + (y-y')^2 + (z-z'+nd)^2}$, θ is elevation angle.

- The Green function is decomposed in two terms:

$$\sum_n G(R_n) = \sum_n G_1(R_n) + \sum_n G_2(R_n)$$

  - One of them decays quickly with R_n: $|G_2(R_{n+1})| \ll |G_2(R_n)|$
  - For the other one the Poisson summation formula is applied:

$$\sum_n G_1(R_n) = \sum_n \hat{G}_1(k_n)$$

where $\hat{G}_1(k)$ is the Fourier transform of $G_1(R)$[1].

[1] $\hat{G}_1(k)$ is narrow in k because $G_1(R)$ is wide in R

1D from 2D periodicity Ewald sum

- No problem with *spatial* term:

$$\sum_m \sum_n G_2(R_{n,m}) \rightarrow \sum_n G_2(R_{n,0})$$

- But the *spectral* term...

$$\sum_m \sum_n \hat{G}_1(k_{n,m}) \rightarrow ??$$

The Fourier Transform of $G_1(n,0)$ must be computed from scratch:

$$\hat{G}_{1,n} = \frac{e^{j(z-z')(k_z - 2\pi\frac{n}{d})}}{2\pi d} g\left(\frac{\alpha_n^2}{4E^2}, \rho^2 E^2\right)$$

$$\rho^2 = (x-x')^2 + (y-y')^2 \qquad \alpha_n^2 = \left(k\cos\theta + \frac{2\pi n}{d}\right)^2 - k^2$$

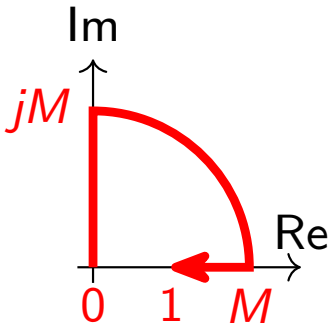
1D from 2D periodicity Ewald sum (cont.)

where we need the following numerical integration

$$g(a,b) = \int_0^1 \frac{e^{-\frac{a}{z^2} - bz^2}}{z}\, dz$$

Notes about g(a,b)

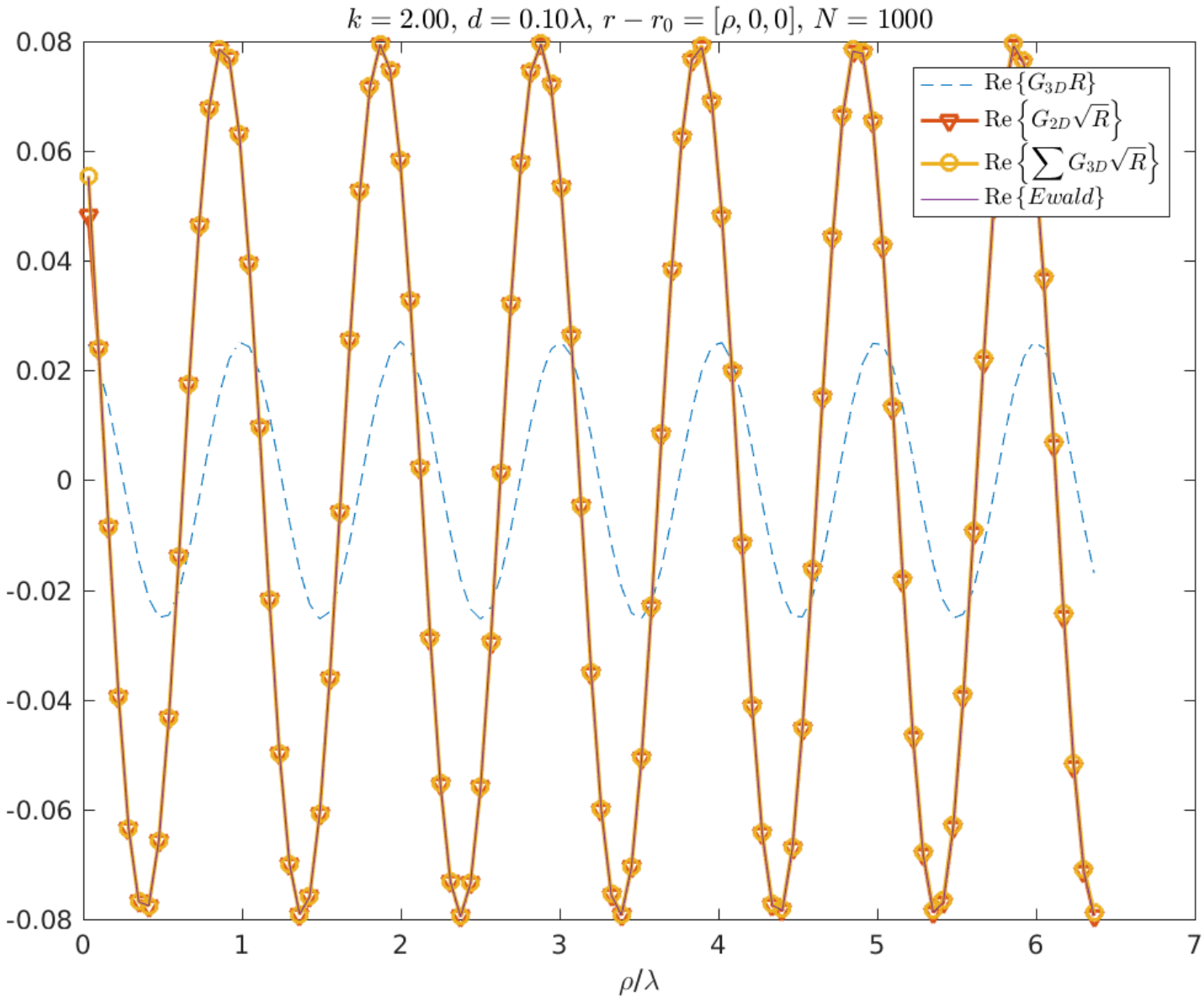
- It is a complex integral whose value (and convergence) depends on the chosen path.
- For calculating $\hat{G}_{1,n}$:
  - $a, b \in \mathbb{R}$
  - $b > 0$,
  - but a is negative for the first n values (At each n term in the sum $a = \alpha_n^2/4E^2$).
- If $a > 0$, the integral can be done through real axis.
- If $a < 0$, the integral must be done through the following path:



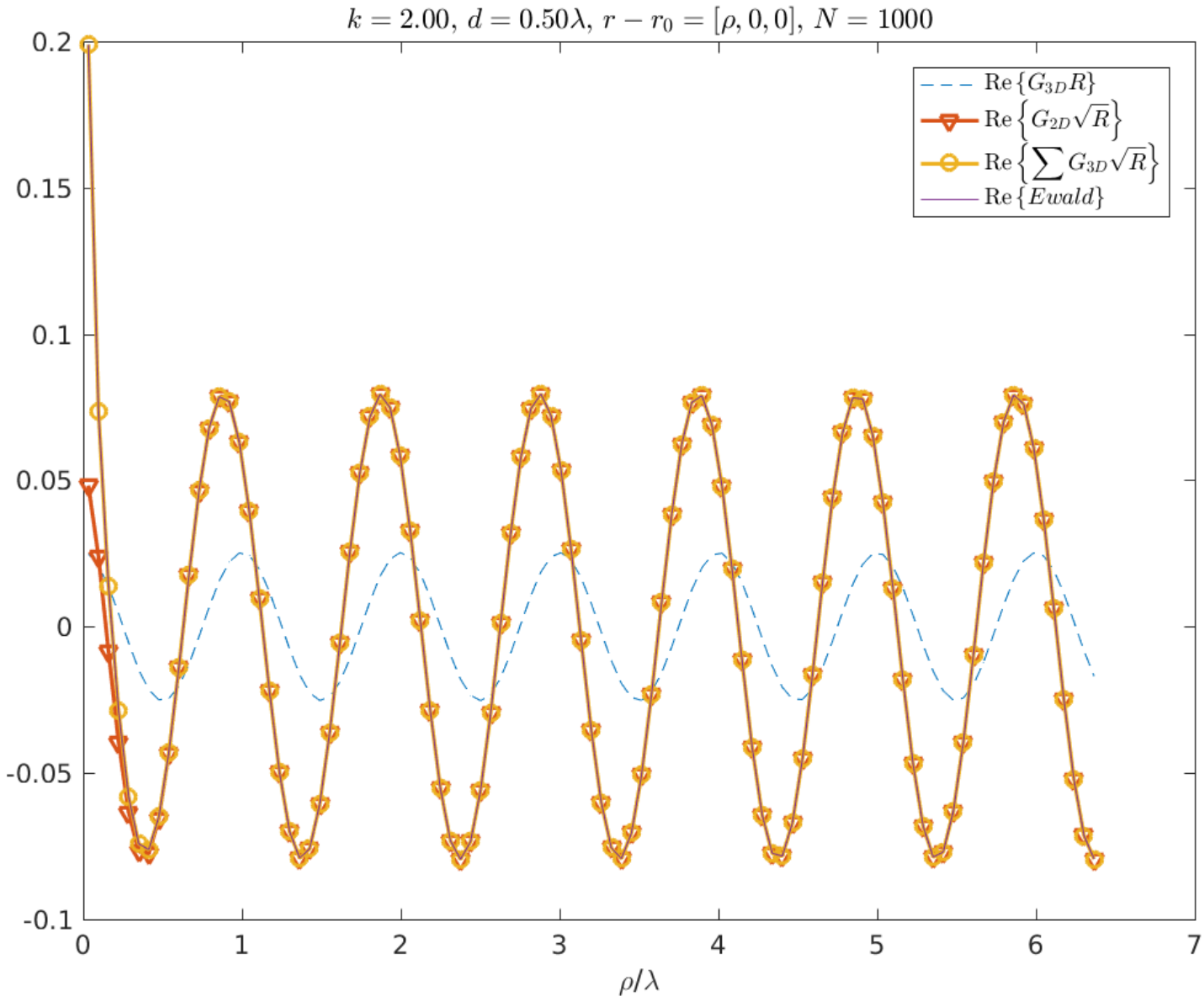
$$M = \left|\frac{a}{b}\right|^{\frac{1}{4}}$$

- If $a = 0$, integral does not converge. It happens for specific values of k and d (not a problem: Greens function is singular there).

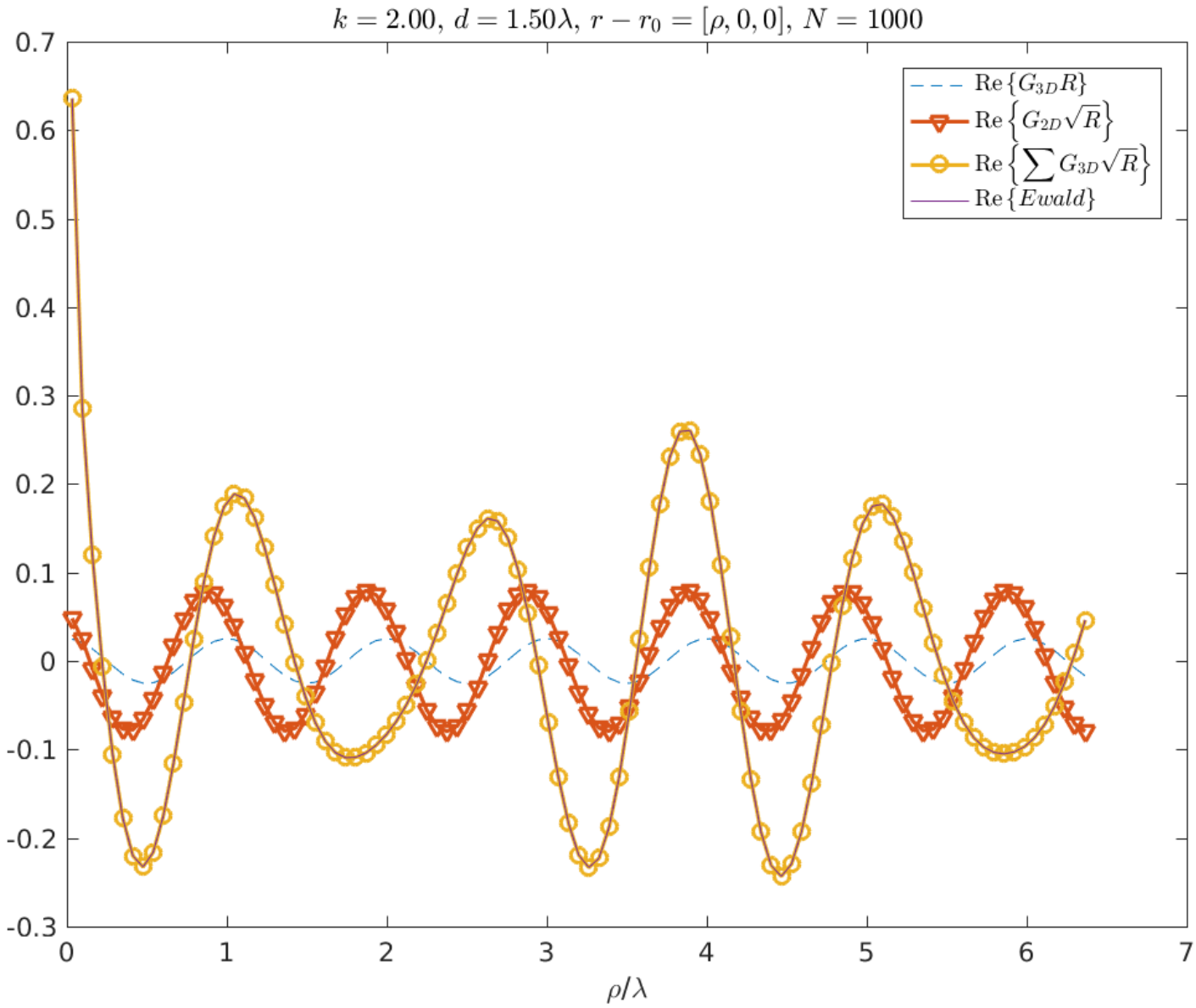
Checking results (thin slice)



Checking results (medium slice)

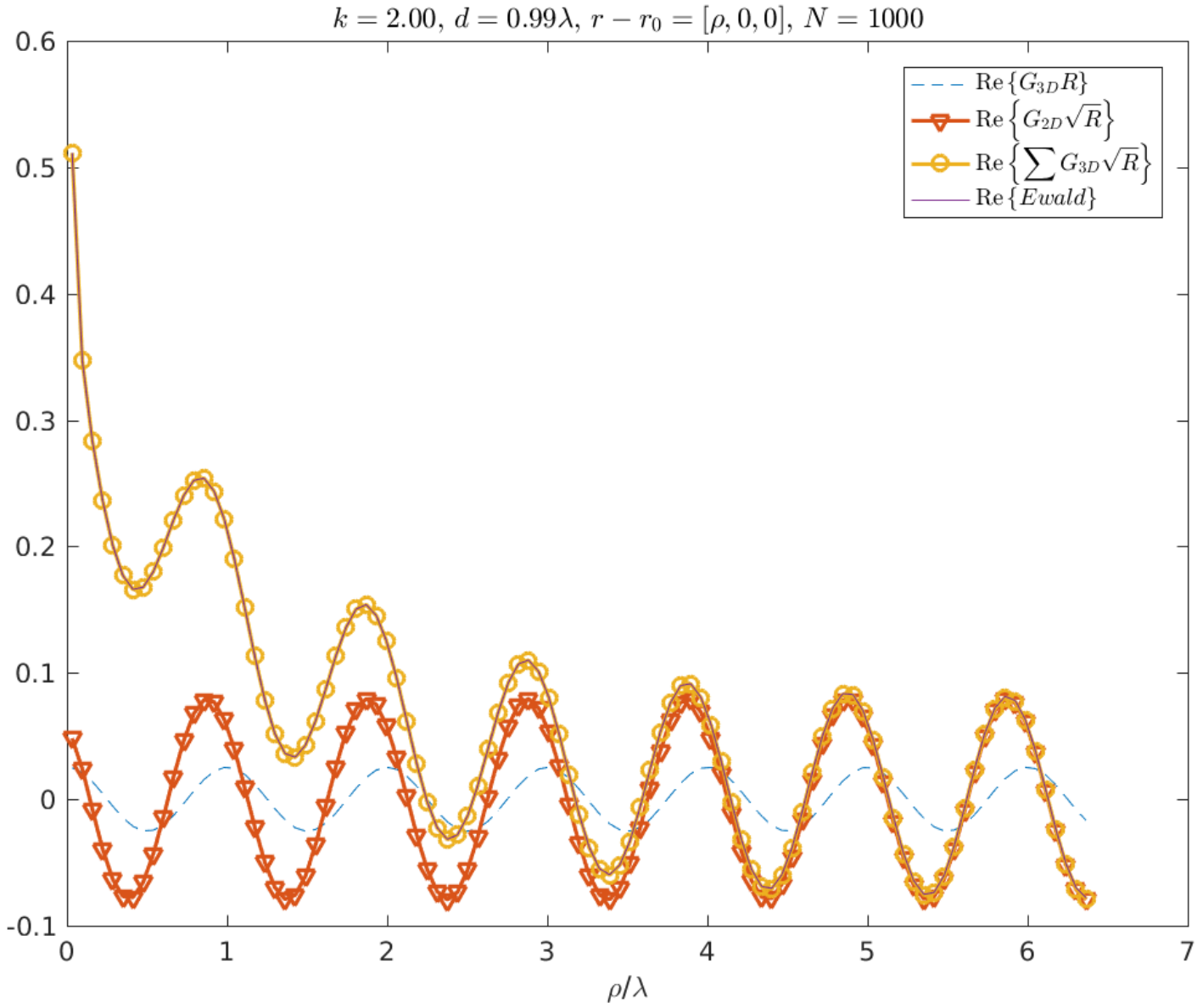


Checking results (thick slice)



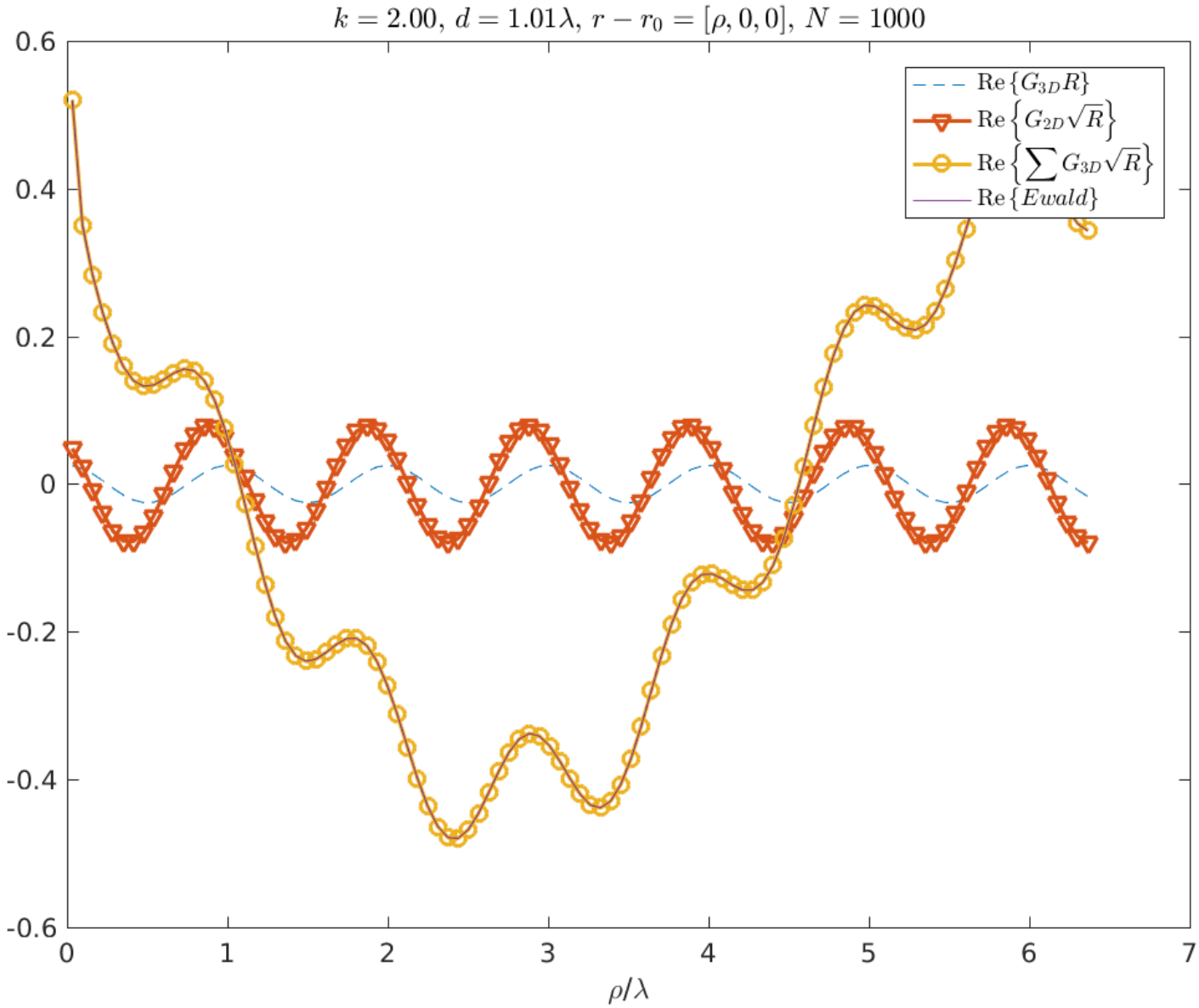
Checking results (close to transverse Floquet resonance)

$\alpha_0^2 \simeq 0$

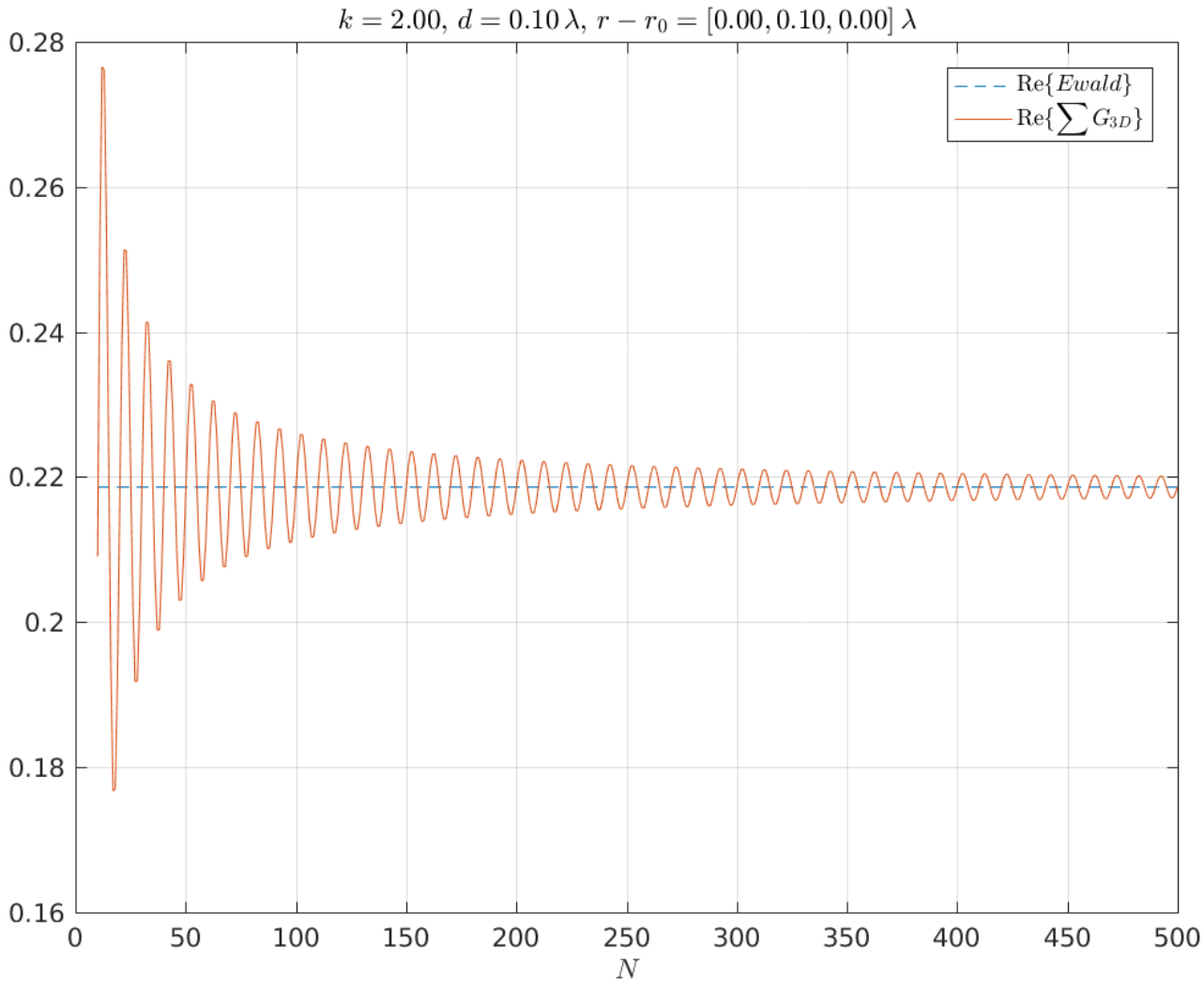


Checking results (close to transverse Floquet resonance)

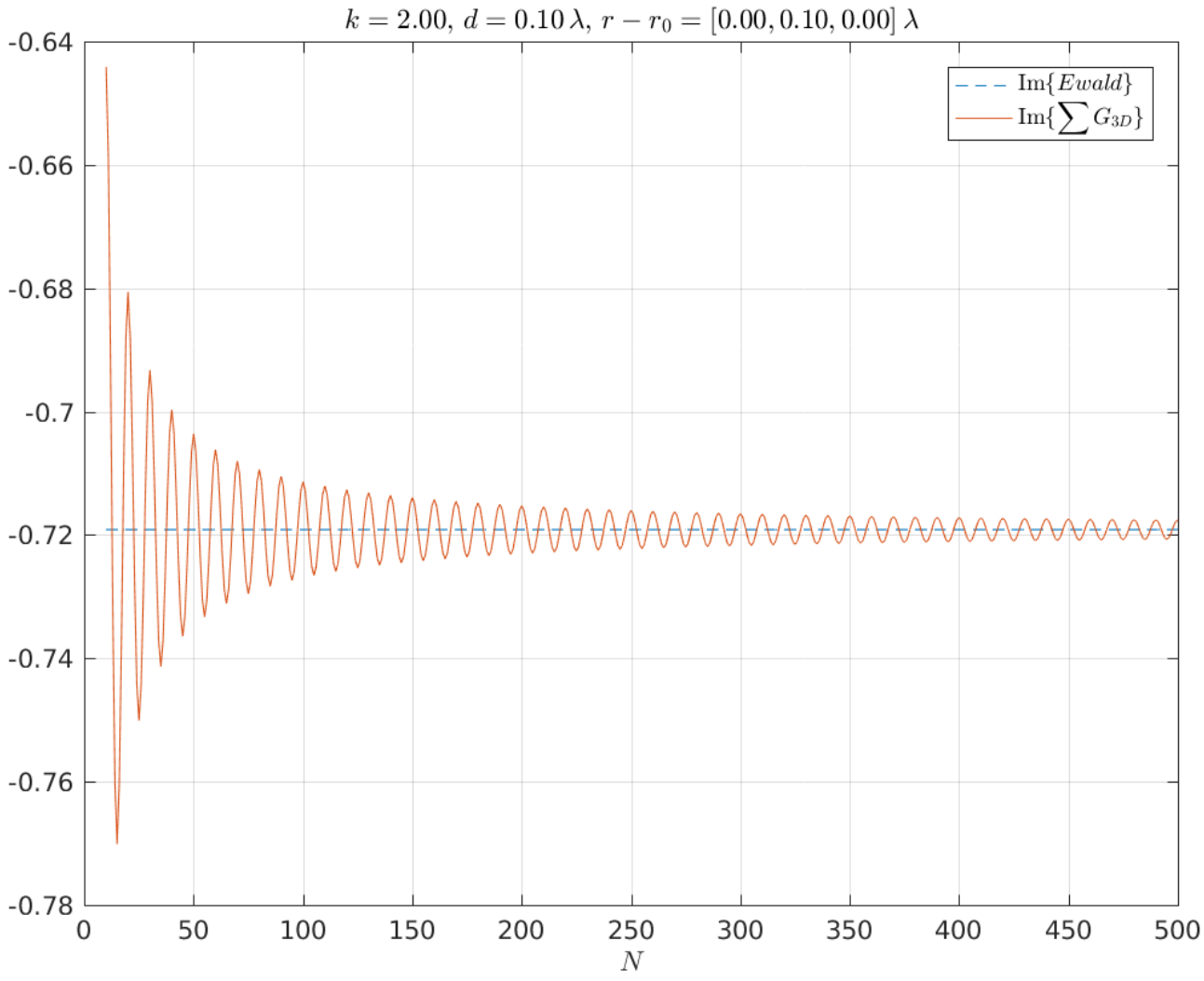
$\alpha_0^2 \simeq 0$



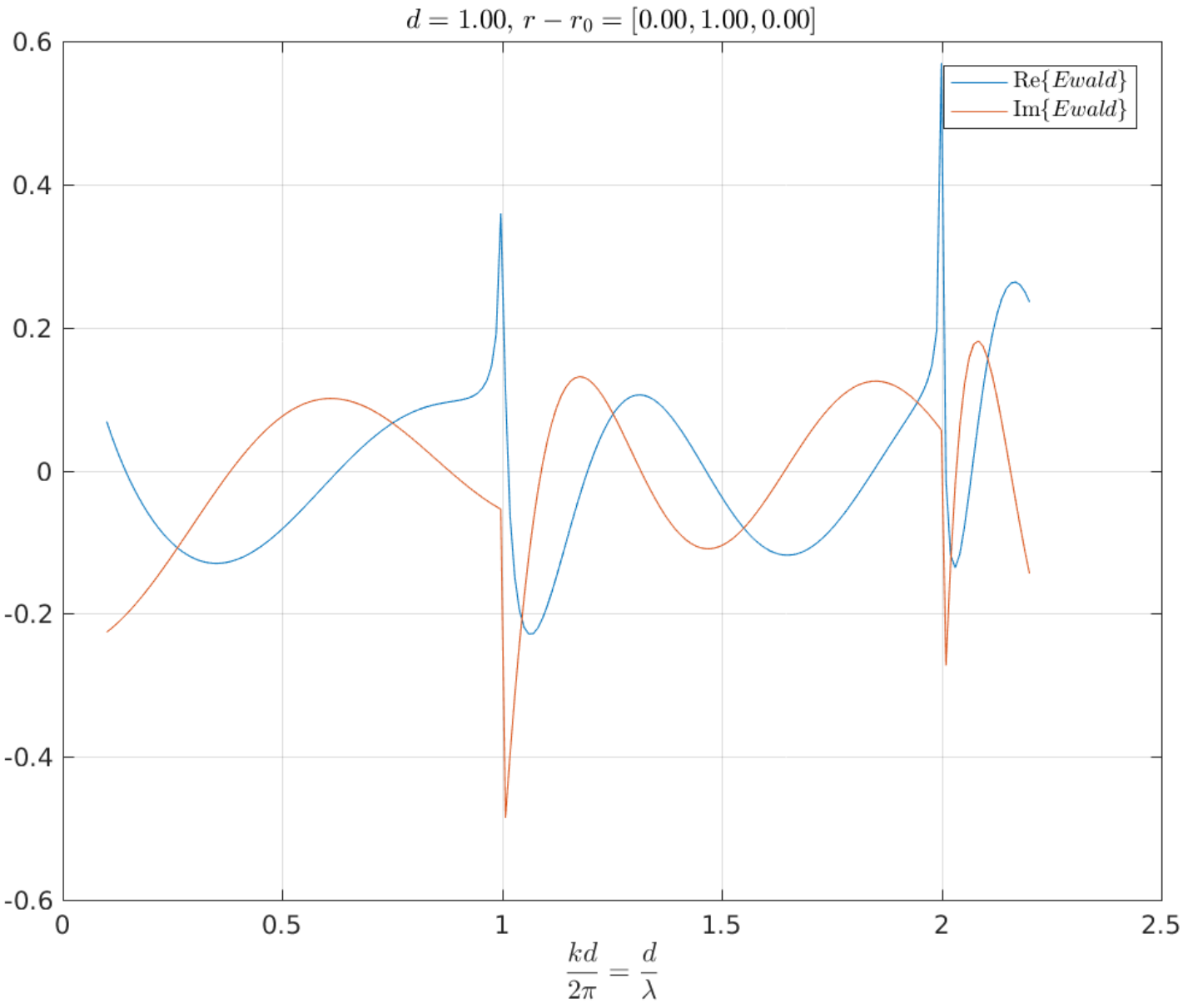
Convergence of the direct sum



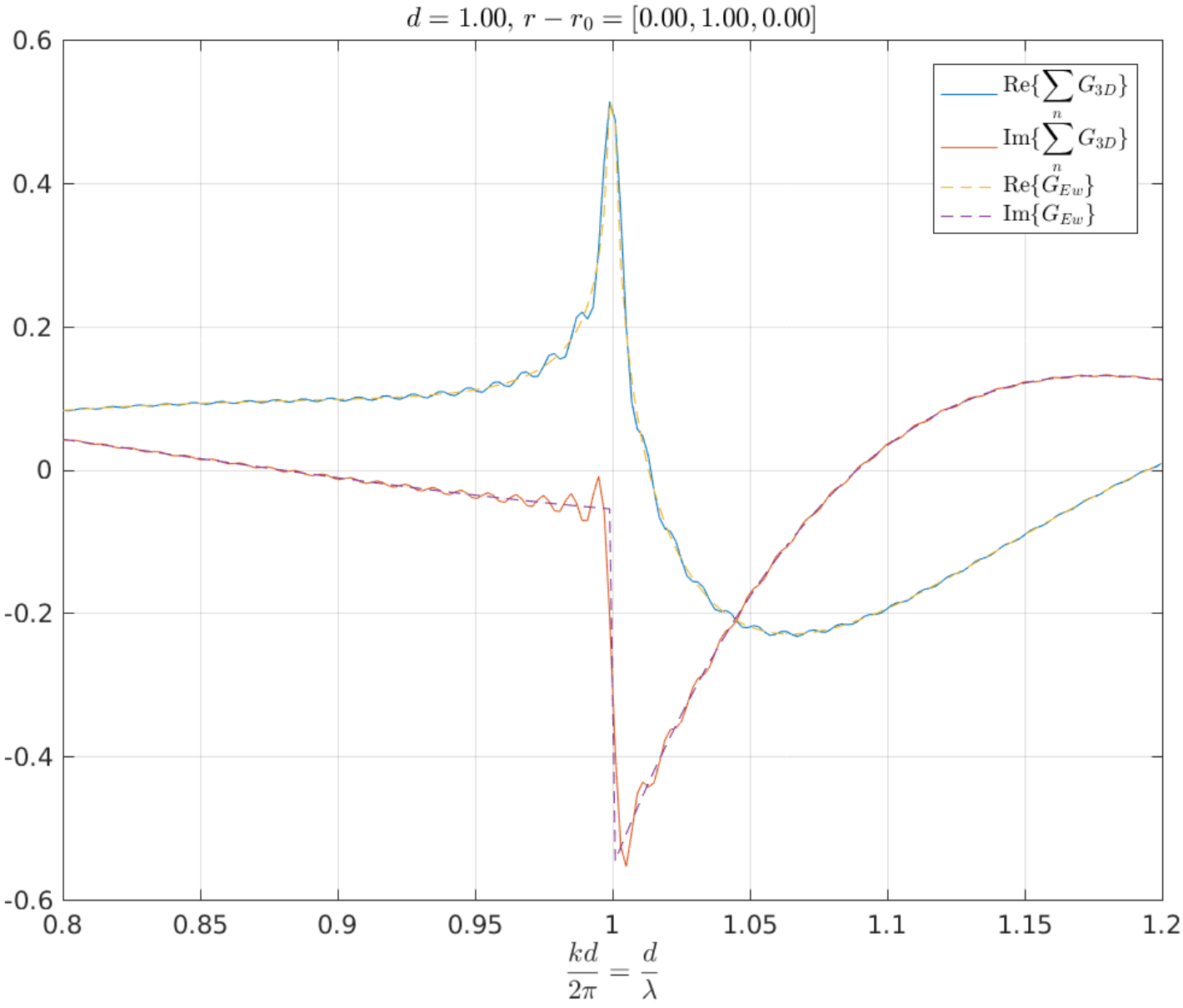
Convergence of the direct sum (cont.)



Dependence with frequency



Singularities when $\alpha_n^2 \to 0$



Additional notes

- CPU time cost similar to sum 100 terms of G_{3D}.
- Matlab prototype.
- Numerical integral may be improved using classic algorithms for calculate exponential integrals (see [2] and [3]) Done!: Calculation 10 times faster, but inestable for large values of ρ ($\rho > d$)
- Checked:

$$\int_{-\frac{d}{2}}^{\frac{d}{2}} G_{3D}(\rho, z)\, dz = G_{2D}(\rho)$$

where G_{3D} is numerically evaluated with Ewald, and G_{2D} is the Hankel function: $\frac{1}{4j} H_0^{(2)}(k|\rho|)$.

[2] F. Capolino *et al.*, "Efficient Computation of the 2-D Green's Function for 1-D Periodic Structures Using the Ewald Method", *IEEE-TAP*, 2005. Equation (23).

[3] F. Capolino *et al.*, "Efficient Computation of the 3D Green's Function with One Dimensional Periodicity Using the Ewald Method", *IEEE-APS*, 2006.

Fortran Module

`EMULATION_2D_scattering_module.F90`

Fortran Module

EMULATION_2D_scattering_module.F90

```
TYPE :: EMULATION_2D_scattering_type
   PRIVATE
   LOGICAL :: enabled_EMULATION_2D_scattering = &
        DEFAULT_ENABLED_EMULATION_2D_SCATTERING
   CHARACTER(len=2) :: green_function_type = DEFAULT_GREEN_FUNCTION_TYPE
   LOGICAL :: enabled_normal_incidence = DEFAULT_ENABLED_NORMAL_INCIDENCE
   ! Thickness (height) of 3D slice used to emulate 2D
   REAL(KIND=DBL) :: slice_thickness = DEFAULT_SLICE_THICKNESS
END TYPE EMULATION_2D_scattering_type

PUBLIC ::  &
   is_EMULATION_2D_scattering_enabled, &
   enable_EMULATION_2D_scattering, &
   disable_EMULATION_2D_scattering, &
   get_EMULATION_2D_scattering_green_function_type, &
   set_EMULATION_2D_scattering_green_function_type, &
   is_EMULATION_2D_scattering_normal_incidence, &
   enable_EMULATION_2D_scattering_normal_incidence, &
   disable_EMULATION_2D_scattering_normal_incidence, &
   get_EMULATION_2D_scattering_slice_thickness, &
   set_EMULATION_2D_scattering_slice_thickness, &
   EMULATION_2D_scattering_print, &
   EMULATION_2D_scattering_sanitycheck
```

Fortran Module (cont.)

EMULATION_2D_scattering_module.F90

Input file .em

```
---------------------------------------
--      EMULATION 2D properties      --
---------------------------------------
-- EMULATION 2D activation flag
EMULATION_2D = true
-- Set the type of green function used
EMULATION_2D_green_function_type = "2D"
-- Activation flag for normal incidence
EMULATION_2D_normal_incidence = true
-- Thickness of the slice used as pseudo2D problem
EMULATION_2D_slice_thickness = 1.0
```

- The case of "normal incidence" is kept as a separate case
  - It does not require periodic meshes
- Value of "slice thickness" set up manually (desirable to be set automatically, e.g., GUI or HOFEM mesh preprocessing)

Fortran Module

EMULATION_2D_scattering_module.F90

Input file .em

```
----------------------------------------
--      EMULATION 2D properties       --
----------------------------------------
-- Orientation of the cylinder (axis unit vector)
EMULATION_2D_cylinder_axis = {0.0,1.0,0.0}
```

- Cylinder axis in arbitrary direction
  - Cylinder axis is set up manually (desirable to be set automatically, e.g., GUI or HOFEM mesh preprocessing)

Fortran Module (cont.)

EMULATION_2D_scattering_module.F90

Input file .em

```
----------------------------------------
--     EMULATION 2D properties        --
----------------------------------------
-- Number of incidence angles to be analized
EMULATION_2D_num_exterior_excitations = 1
-- Type of plane wave ("Single", "Multiple")
EMULATION_2D_exterior_type_1 = "Single"
-- Angle of incidence (with respect to local spherical coordinate
-- system with z axis along the cylinder axis)
--    * First number is 'theta' (angle with respect to cylinder axis;
--      90 corresponds to normal incidence)
--    * Second number is 'phi' (angle of associated local cylindrical
--      coordinate system)
EMULATION_2D_exterior_angle_1 = {90,45}
-- Polarization (TM or TE)
EMULATION_2D_polarization_type_1 = "TM"
```

Fortran Module (cont.)

`EMULATION_2D_scattering_module.F90`

- Definition of angles for excitations with respect to local spherical coordinate system with z axis along the cylinder axis
  - Angles are transformed back and forth between local and global axis.
- Definition of polarization as either “TM” or “TE”
  - Vector components (polarization) are transformed back and forth between local and global axis.

Fortran Module (cont.)

EMULATION_2D_scattering_module.F90

Input file farfield_EMULATION_2D.conf

```
!- Farfield mode (bistatic,monostatic)
Bistatic
!- Farfield component (|rE-longitudinal|, rE-longitudinal-real, rE-lor
rE-transverse|, rE-transverse-real, rE-transverse-imag, rE-transverse-
inal, RCS-dB, RCS-longitudinal-dB, RCS-transverse-dB, Monostatic-RCS,
RCS
!- Frequency index and number of rhs
1 1
!- Rhs index
1
!- Rhs phase
0
!- Rhs amplitude
1
```

Fortran Module (cont.)

EMULATION_2D_scattering_module.F90

Input file farfield_EMULATION_2D.conf

```
!- Monitor number
1
!- Phi_cyl sampling points (num init stop)
37 0 360
!- RCS units
sigma-lambda
!- Array analysis (flag,num_elem_u,num_elem_v)
0 1 1
```

Fortran Module (cont.)

`EMULATION_2D_scattering_module.F90`

- Specialization to EMULATION 2D mode
  - Farfield-mode: only scattering
  - Farfield-component: "longitudinal" and "transverse" components with respect to cylinder
  - ...
- Only the angles "phi" need to be defined for the plot (plot asociated to the transversal plane to the cylinder (perpendicular to its axis)
- Array analysis: we keep it as it might be of interest to consider finite 1D/2D periodic structures based on pseudo2D cylinders

Fortran Module

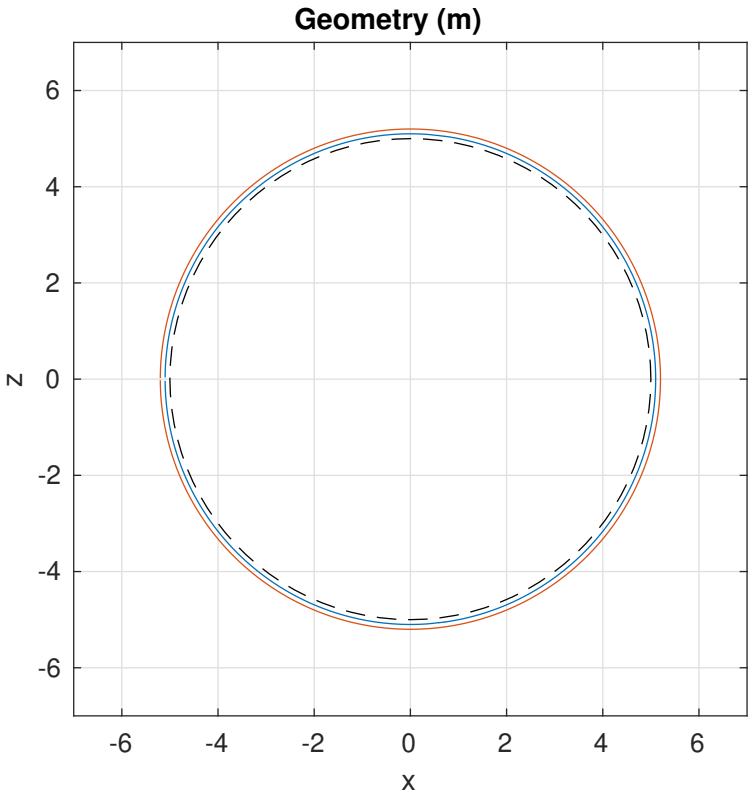
`EMULATION_2D_scattering_module.F90` —Update 8 Nov 2022—

- Near field: GreenD derivatives are rewritten in a compact way supporting computation of the scattering near field (within FE-IIEE loop) for arbitraily oriented cylinders
- Far field: Perfect agreement between numerical and analytical solutions
  - Well... agreement up to a constant (not worried at all about it)

Far-field Bistatic RCS calculation

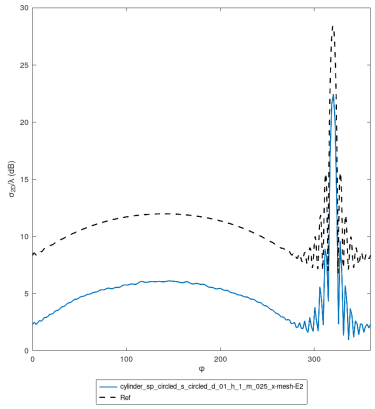
Reference cases

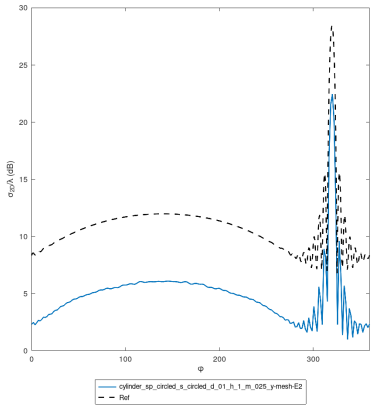
The same geometry is analized with different orientations of the longitudinal axis:

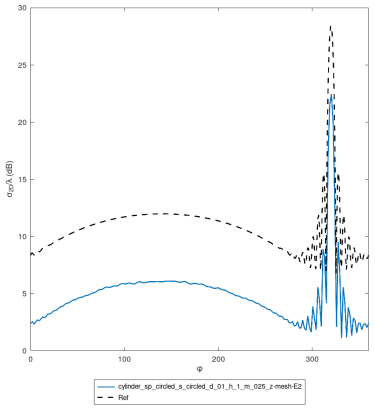


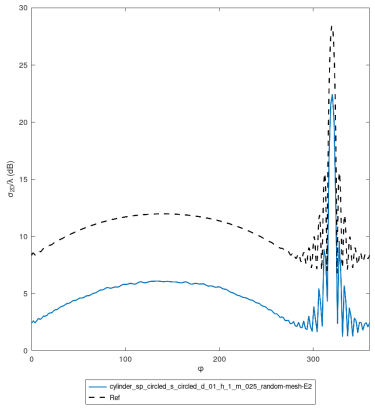
- Normal incidence
- Dashed black line:
- Blue line: S'
- Red line: S
- Different values of permittivity between PEC and S': ε_r
- $f = 300\,\text{MHz}$ ($\lambda = 1\,\text{m}$, cylinder radius is 5λ).

TM polarization, $\varepsilon_r = 1$ (reference case)

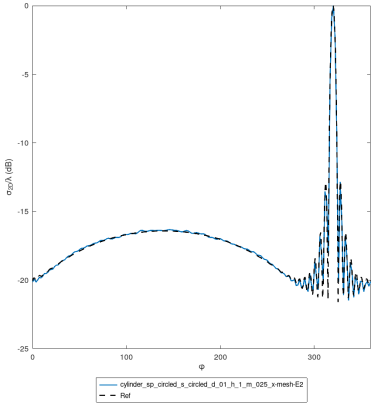


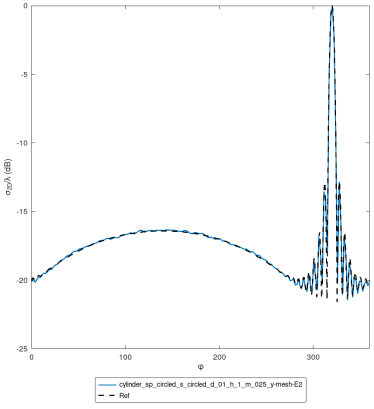


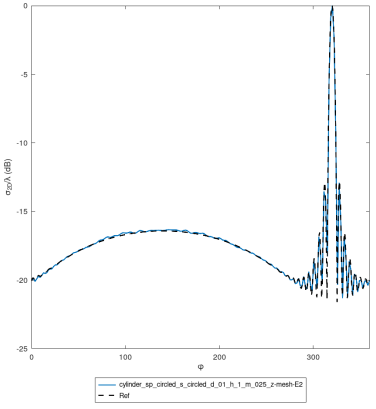


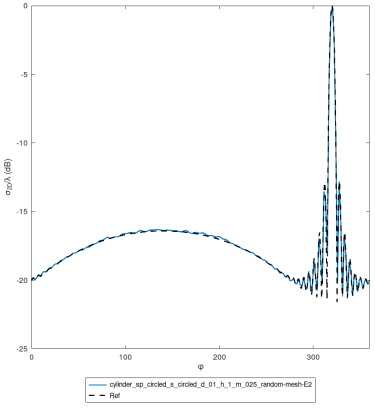


Analytical curves and FEM curves agree except for a constant factor.
Normalized bistatic-RCS:

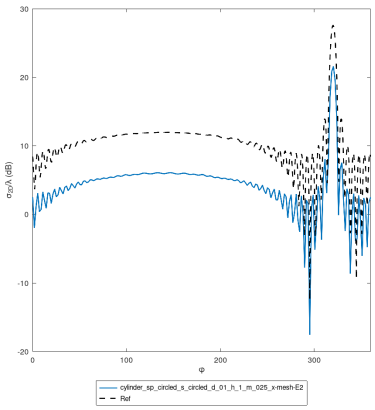


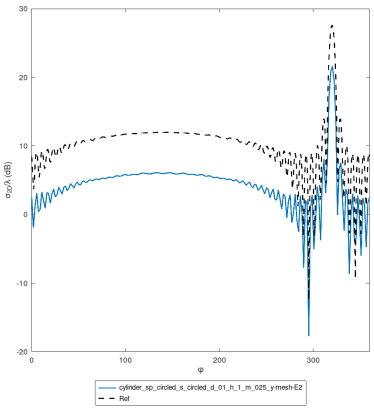


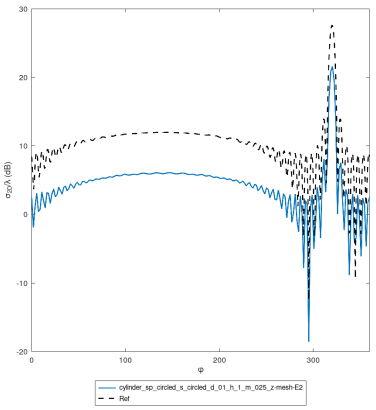


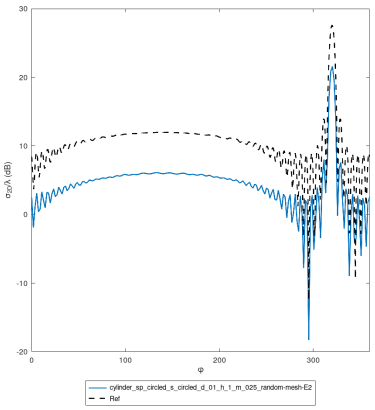


TE polarization, $\varepsilon_r = 1$

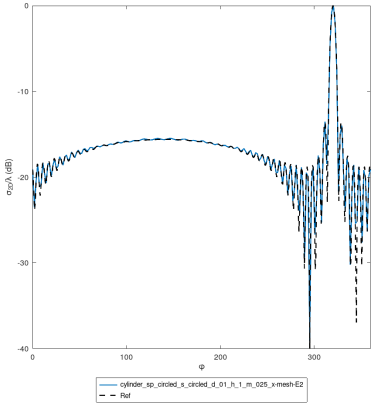


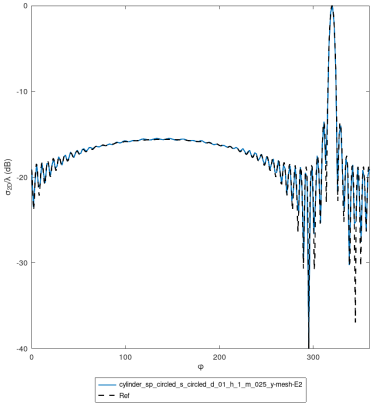


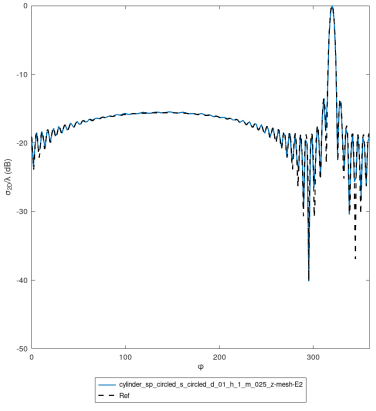


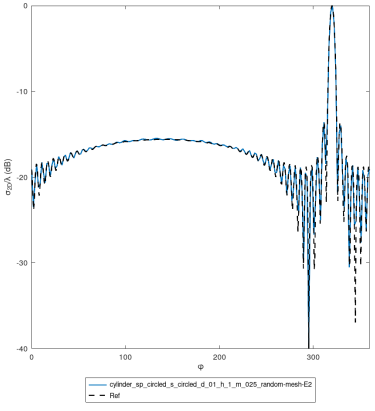


Analytical curves and FEM curves agree except for a constant factor.
Normalized bistatic-RCS:

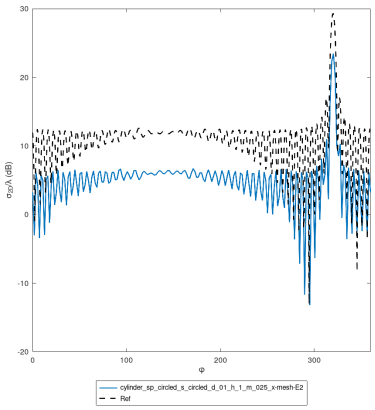


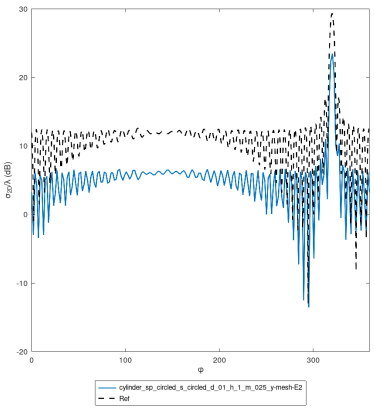


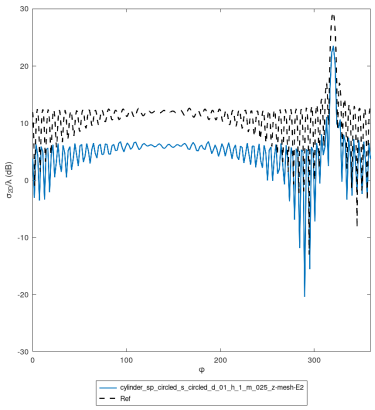


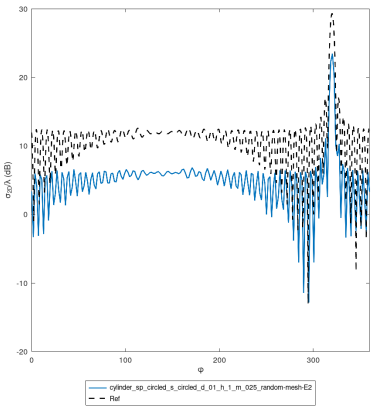


TM polarization, $\varepsilon_r = 8$

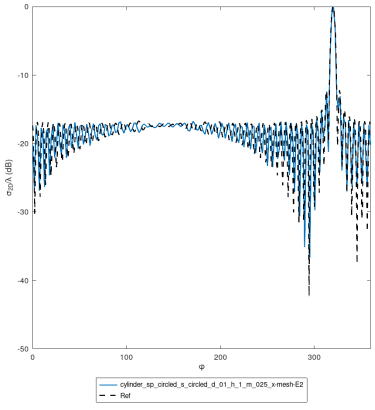


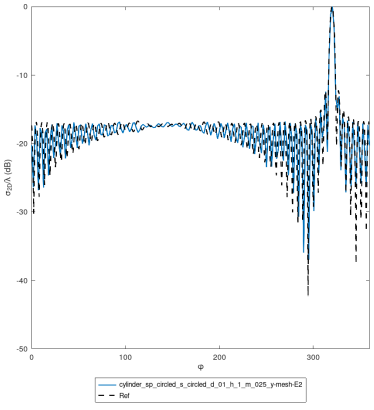


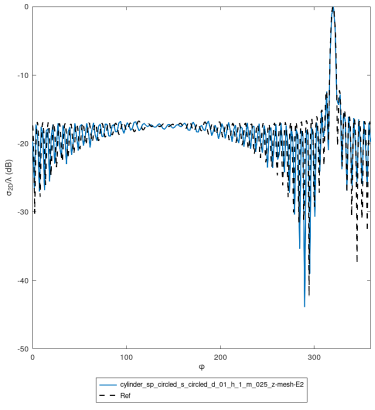


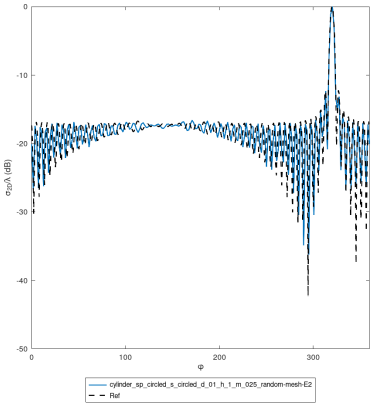


Analytical curves and FEM curves agree (minor discrepancies) except for a constant factor.
Normalized bistatic-RCS:

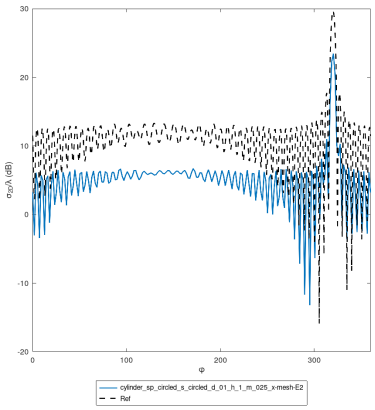


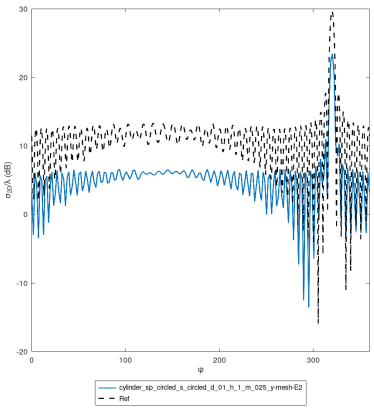


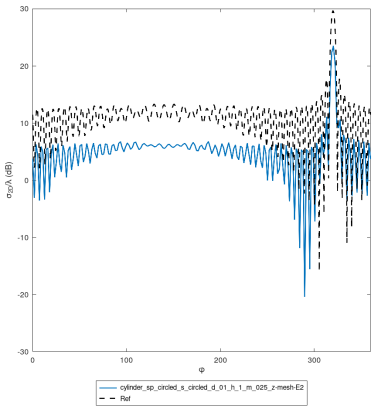


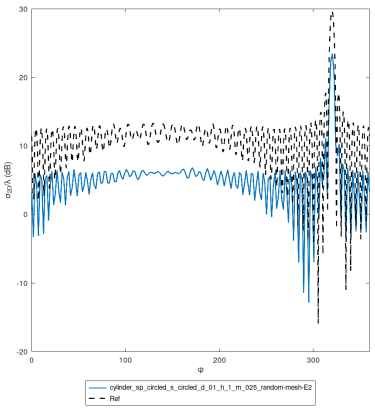


TE polarization, $\varepsilon_r = 2$

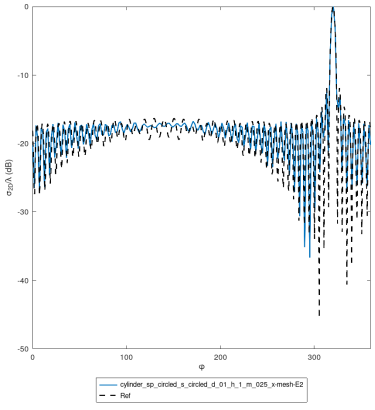


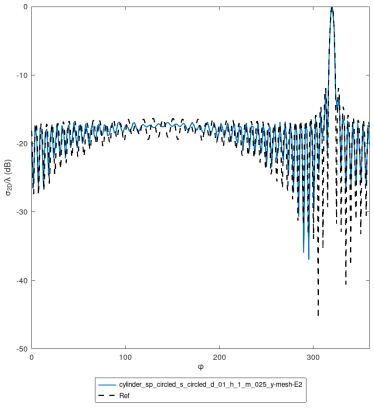


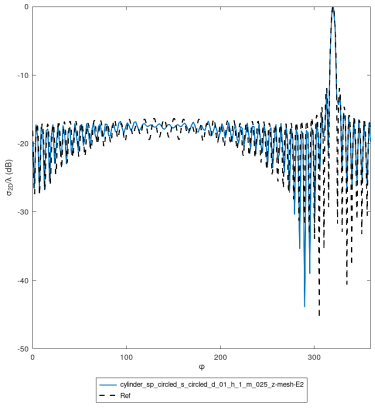


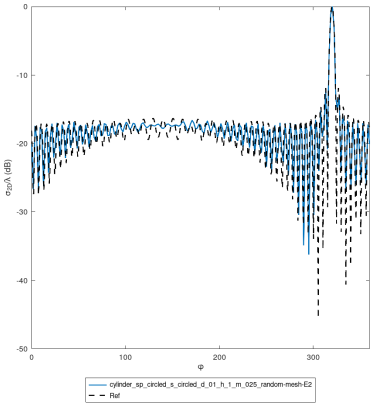


Analytical curves and FEM curves agree (some discrepancies) except for a constant factor.
Normalized bistatic-RCS:

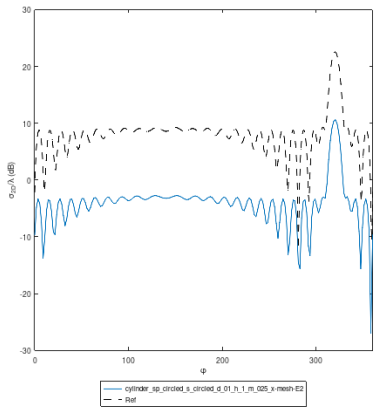


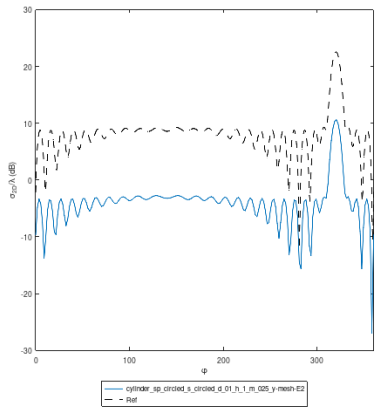


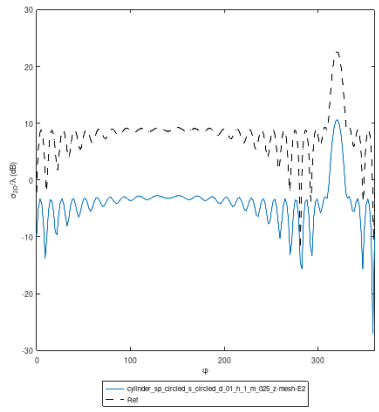


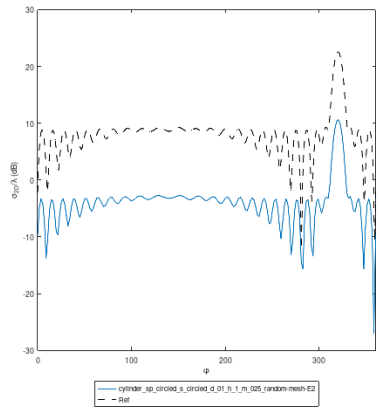


TE polarization, $\varepsilon_r = 2$, $f = 150$ MHz

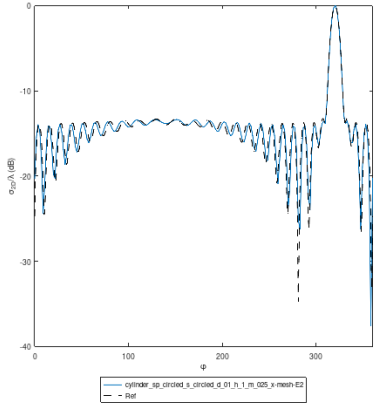


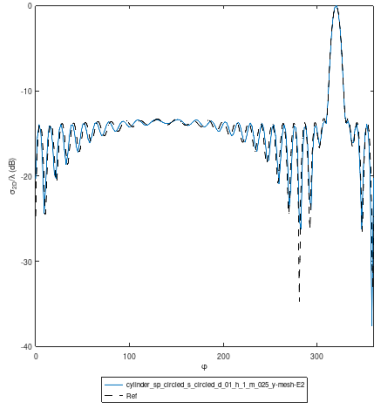


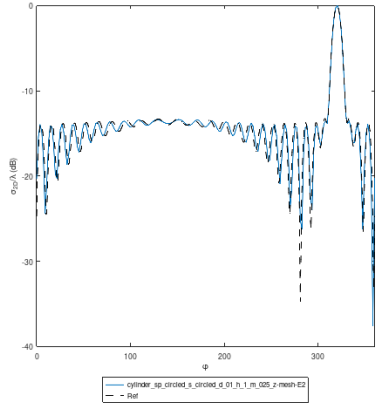


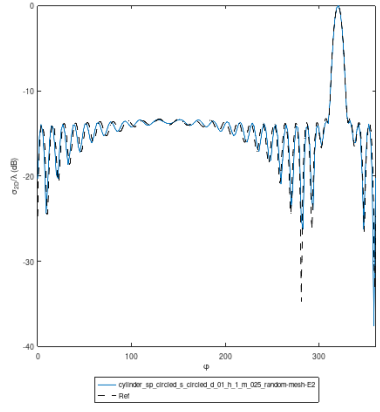


For a smaller problem (same mesh, double wavelength), almost full agree.
Normalized bistatic-RCS:







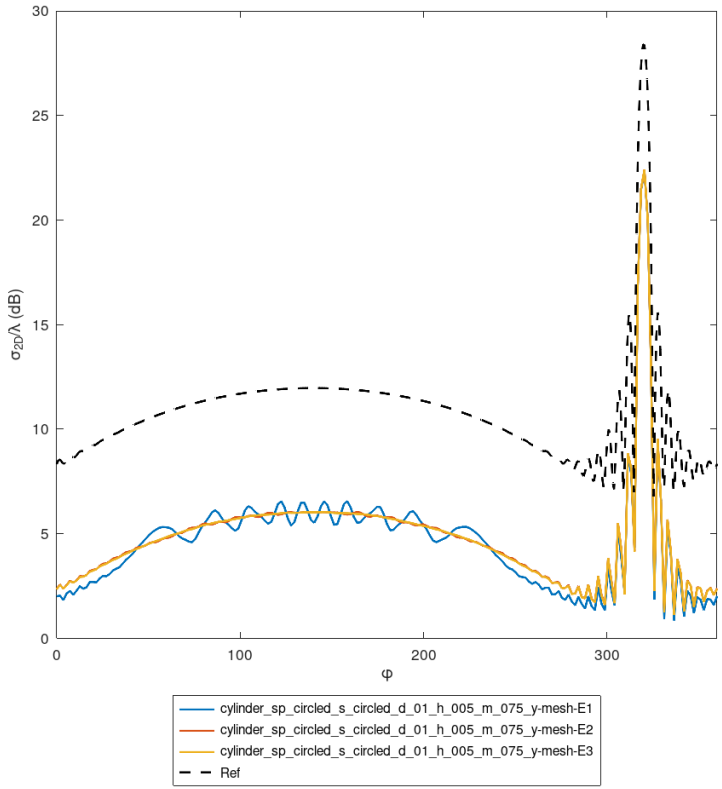


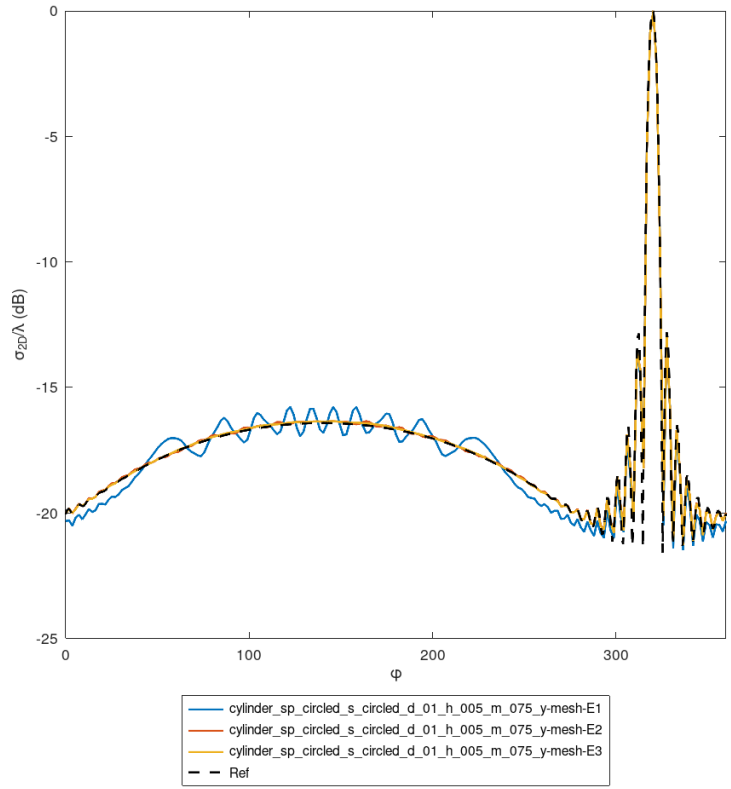
Effect of IIEE iterative method in Far Field

TM polarization, $\varepsilon_r = 1$, (reference case)

Results for the following residual errors:

- M1: 10^{-1} (blue)
- M2: 10^{-2} (red)
- M3: 10^{-3} (orange)

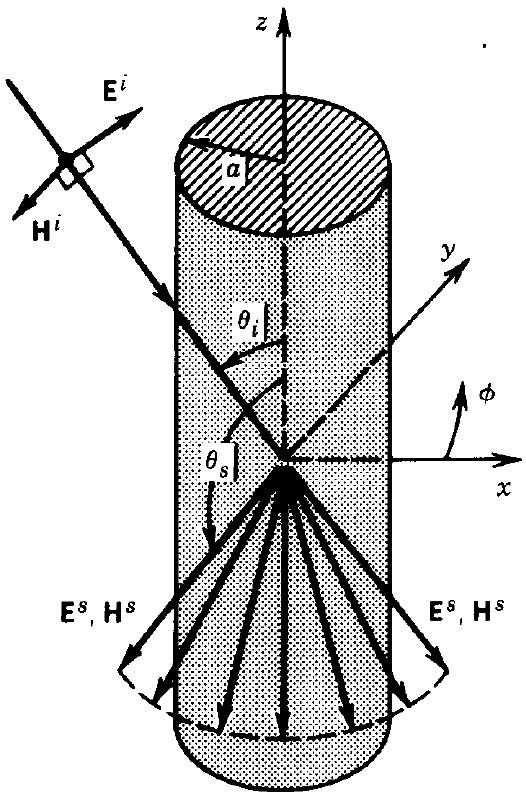




Note: Orange and red lines overlap.

Until now we were considering normal incidence to the cylinder
Now we consider the extension to case of “Oblique Incidence”

Oblique Incidence



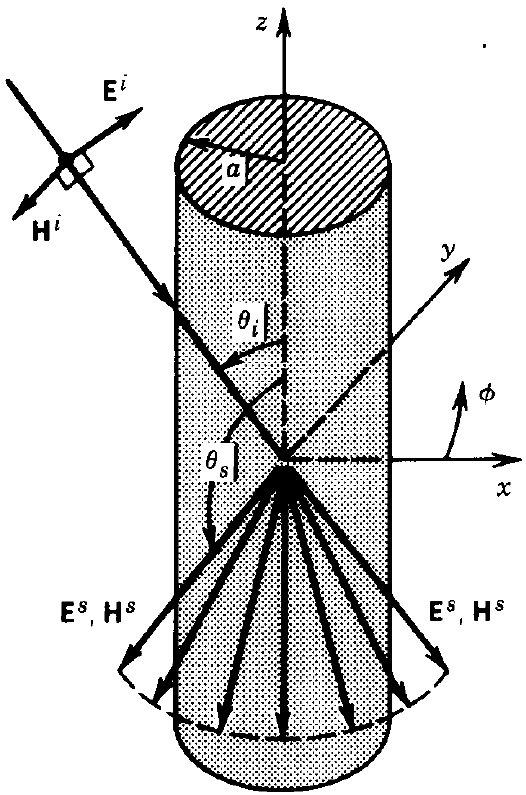
Green Function

$$G = \int_{-\infty}^{\infty} \frac{e^{-jk\sqrt{\rho^2+z^2}}}{4\pi\sqrt{\rho^2+z^2}} e^{-j\beta_z z}\, dz$$

$$= \begin{cases} \frac{1}{4j} H_0^{(2)}\left(\rho\sqrt{k^2-\beta_z^2}\right) & k^2 > \beta_z^2 \\ \frac{1}{2\pi} K_0^{(2)}\left(\rho\sqrt{k^2-\beta_z^2}\right) & \beta_z^2 > k^2 \end{cases}$$

where $\beta_z = k\cos\theta_i < k$

Oblique Incidence (cont.)



Green Function

- For any θ_i we have

$$G = \frac{1}{4j} H_0^{(2)} \left(\rho \sqrt{k^2 - \beta_z^2} \right) = \frac{1}{4j} H_0^{(2)} \left(k|\rho| \sin \theta_i \right)$$

  where $\beta_z = k \cos \theta_i < k$

- The particularization to $\theta_i = 90^\circ$ (normal incidence) yields to the well known result

$$G = \frac{1}{4j} H_0^{(2)}(k|\rho|)$$

Oblique Incidence (cont.)

FE-IIEE loop

- As in the case of normal incidence
  - We use `Green2D`[a] on the whole 3D slice and we divide the scattered field at each target point by the "slice thickness"
  - We work with ρ instead of r, i.e.,

$$|\boldsymbol{\rho} - \boldsymbol{\rho}'| = |(\mathbf{r} - \mathbf{r}') - ((\mathbf{r} - \mathbf{r}') \cdot \hat{\mathbf{z}})\hat{\mathbf{z}}|$$

    where $\hat{\mathbf{z}}$ stands for the direction along the cylinder axis[b]
- But for oblique incidence we need to take into account the relative "height" between $\mathbf{r}$ and $\mathbf{r}'$

$$e^{-j\beta_z|(\mathbf{r}-\mathbf{r}')\cdot\hat{\mathbf{z}}|}$$

[a] We may also use `Green3D_Ewald`

[b] The code supports arbitrary orientations for the cylinder

Oblique Incidence (cont.)

In other words,

$$G(|\mathbf{r}-\mathbf{r}'|) = \frac{1}{4j} H_0^{(2)}\left(k|\boldsymbol{\rho}-\boldsymbol{\rho}'|\sin\theta_i\right)\, e^{-jk\cos\theta_i|z-z'|}$$

Remarks:

- Symbol $\hat{\mathbf{z}}$ stands for the direction along the cylinder axis
- The code supports arbitrary orientations for the cylinder

- Implementation status:
  - Coded
  - Not tested for oblique cylinder yet (requires PBCs)

Oblique Incidence (cont.)

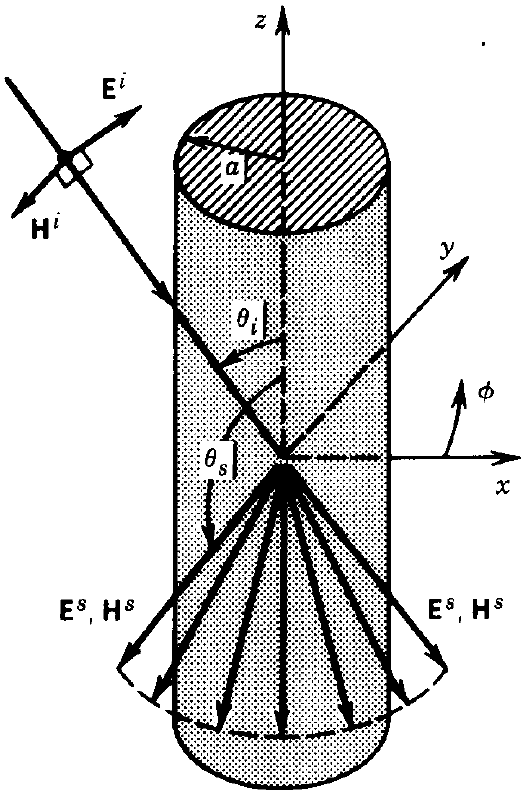
A remark on dielectric (and dieliectric coated PEC) cylinders...

Dielectric cylinders (also dielectric coated PEC)

- TM and TE are not longer solutions (polarizations are coupled) for oblique incidence
- In the limit, with angle tending to normal incidence, the polarizations are effectively decoupled
- Analogously, TM and TE coupling appears also when using IBC (even isotropic case) for oblique incidence

Oblique Incidence

Far Field —Update 8 Nov 2022—



Far Field Setup

Far Field

- Green's function

$$G_{\text{far}} \propto e^{+jk\rho \sin\theta_i} \underbrace{e^{-jk\cos\theta_i z}}_{F_{1z}(z)}$$

- Potential, fields

$$\text{Potentials, fields} \propto F_{2\phi}(k, \phi, \sin\theta_i) F_{2\theta}(\theta)$$

where

$$F_{2\theta}(\theta) = \int_{-\infty}^{\infty} F_{1z}(z) e^{+jk\cos\theta z} = \delta(\theta - (\pi - \theta_i))$$

Using far field version of `Green2D` on a 3D slice can be used to calculate $F_{2\phi}$ (the variation with ϕ of the scattering field: RCS, scattering width)

Oblique Incidence (cont.)

Far Field —Update 8 Nov 2022—

More precisely...

- Green's Function

$$G_{\text{far}} = \frac{1}{4j} H_0^{(2)}(k\rho \sin\theta_i)\Big|_{k\rho\to\infty} \approx \sqrt{\frac{2j}{\pi\rho\sin\theta_i}} e^{-jk\rho\sin\theta_i}$$

Oblique Incidence (cont.)

Far Field —Update 8 Nov 2022—

Far Field Computation using 3D Slice

- We use `Green2D` on the whole 3D slice and we divide the scattered field at each target point by the "slice thickness"
- We need to take into account the z-dependence of the 3D (FEM) solution, i.e.,

$$\mathbf{E}, \mathbf{H} \propto e^{jk \cos\theta_i z}$$

Oblique Incidence (cont.)

Far Field —Update 8 Nov 2022—

Far Field Algorithm:

1. We start from $\mathbf{J}$, $\mathbf{M}$ on S'
2. We "equalize" $\mathbf{J}$, $\mathbf{M}$ on z-dependence[4]

$$\mathbf{J} = \mathbf{J}\, e^{-jk\cos\theta_i z}$$
$$\mathbf{M} = \mathbf{M}\, e^{-jk\cos\theta_i z}$$

3. We transform $\mathbf{J}$, $\mathbf{M}$ to spherical coordinates on local coordinate system attached to the cylinder

$$\mathbf{J} \longrightarrow (J_\theta, J_\phi)$$
$$\mathbf{M} \longrightarrow (M_\theta, M_\phi)$$

   - Note that for normal incidence the local θ, ϕ components are longitudinal J_z, M_z and transverse J_ϕ, M_ϕ components, respectively

Oblique Incidence (cont.)

Far Field —Update 8 Nov 2022—

4. We compute vector potentials A_θ, A_ϕ, F_θ, F_ϕ

$$J_\theta \rightarrow A_\theta \qquad M_\theta \longrightarrow F_\theta$$
$$J_\phi \rightarrow A_\phi \qquad M_\phi \longrightarrow F_\phi$$

5. and from vector potentials we compute the far field components

$$(A_\theta, A_\phi, F_\theta, F_\phi) \longrightarrow (E_\theta, E_\phi, H_\theta, H_\phi)$$

6. We do not forget to divide by "slice thickness"

$$(E_\theta, E_\phi, H_\theta, H_\phi) \longrightarrow (E_\theta, E_\phi, H_\theta, H_\phi) \,/\, \text{"slice thickness"}$$

7. Due to decomposition of currents in local θ and ϕ components, we can naturally obtain the TM and TE components, i.e.,

$$(E_\theta, H_\phi) \longrightarrow \text{TM}$$
$$(E_\phi, H_\theta) \longrightarrow \text{TE}$$

Oblique Incidence (cont.)

Far Field —Update 8 Nov 2022—

Scattering Width σ

$$\sigma = \begin{bmatrix} \sigma_{(\mathrm{TM,TM})} & \sigma_{(\mathrm{TM,TE})} \\ \sigma_{(\mathrm{TE,TM})} & \sigma_{(\mathrm{TE,TE})} \end{bmatrix}$$

- Note that for PEC cylinders we have $\sigma_{(\mathrm{TM,TE})} = \sigma_{(\mathrm{TE,TM})} = 0$

- Implementation status:
  - Coded
  - Not tested for oblique cylinder yet (requires PBCs)

[4] The "equalization" can alternatively be applied on the Greens's Function

Integration EMULATION_2D and PBCs
Far Field

Logic for Oblique Incidence —In progress (barely initiated)—

- `cylinder_axis` and `slice_thickness` obtained from PBC module
- Angle of incidence set up in file `.em`
  - Discard PBC info in file `.em`: `periodic_phi_angle`, `periodic_theta_angle`
  - PBC $\longleftarrow$ EMULATION_2D info from file
    - Example: `EMULATION_2D_exterior_angles_1 = {45,140}`
- Normal incidence can be run as a particular case

- If `enabled_normal_incidence` there is no PBC interaction (coded and working)
  - `cylinder_axis` and `slice_thickness` read from file `.em`[5]
  - Angle of incidence set up in file `.em`:
    - Example: `EMULATION_2D_exterior_angles_1 = {90,140}`

[5] Nevertheless, that information could be obtained by preprocessing the mesh within HOFEM code (extra task, not considered at present)

Modifications relative to postprocessing (near field)

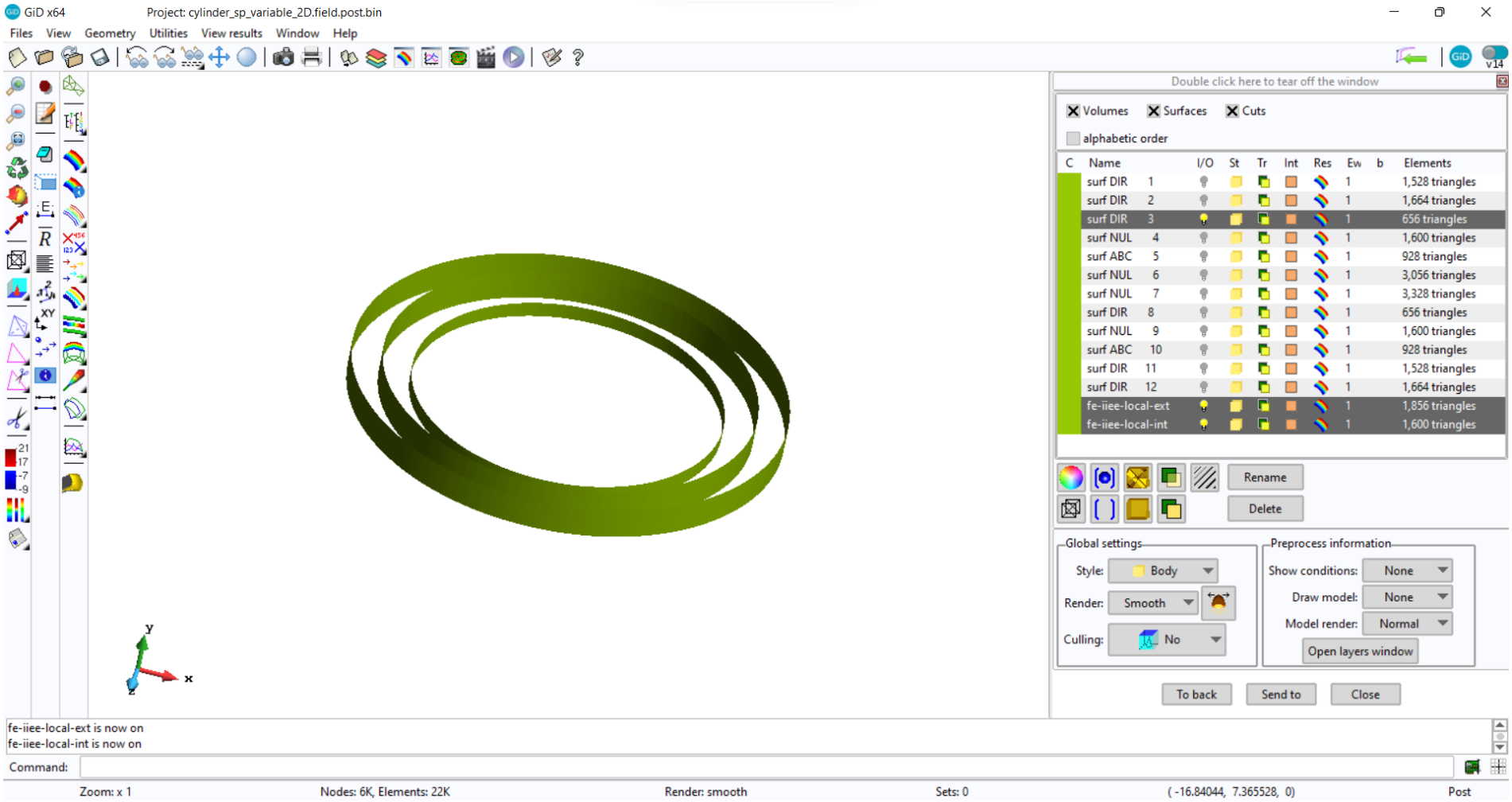
Postprocesssing changes
Extra information in nearfield

Changes in nearfield postprocessing

- Added a 3-character string after `surf` to denote the kind of boundary condition that surface is (DIR, NEU, NUL, ABC,...)
- Added to the surface set the surfaces involved in FE-IIEE method, denoted as `fe-iiee-local-ext`, and `fe-iiee-local-int`, corresponding to S and S' surfaces respectively.

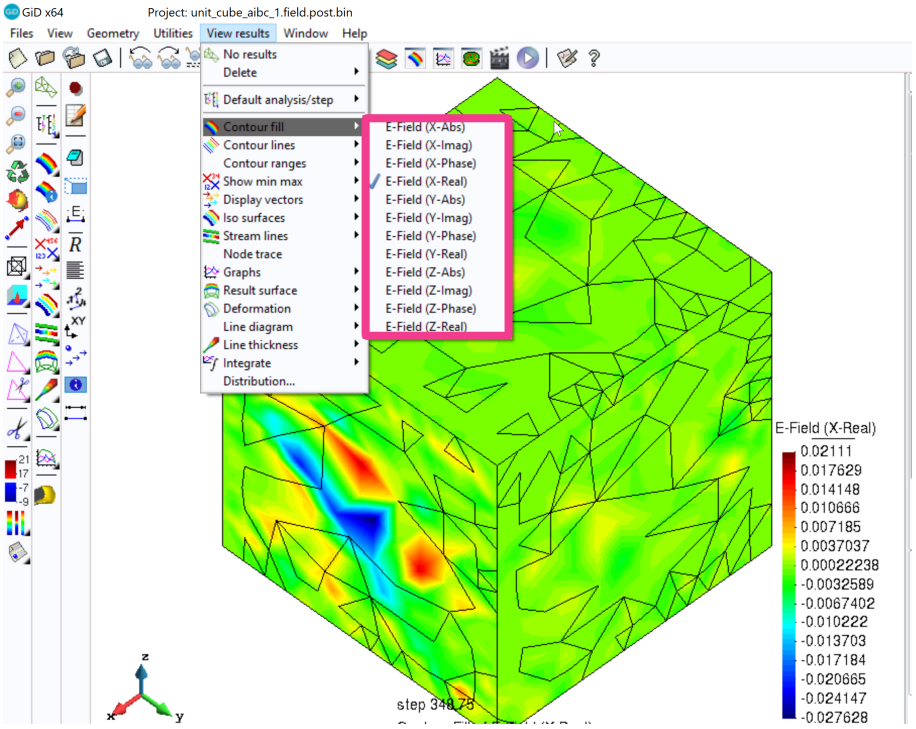
Postprocesssing changes (cont.)

Extra information in nearfield



Postprocesssing changes

Extra information in nearfield



- Included a new option to get all the available results for the near field.
- Implemented as a new option called `all-xyz`.

MPI related issues

EMULATION_2D MPI Broadcast

- From FREQ-SWEEP to POST-MODE we write and read from disk

Write to Disk

```
SUBROUTINE write_EMULATION_2D_data_structure(filename)

  CHARACTER(LEN=*), INTENT(IN) :: filename
  INTEGER:: ierror,fileunit

  OPEN(NEWUNIT=fileunit, FILE=filename, STATUS="REPLACE",  &
      ACTION="WRITE", FORM="UNFORMATTED",IOSTAT=ierror)
  IF (ierror /= 0) THEN
     CHKERRQ(ierror,'␣write_EMULATION_2D_data_structure:␣ &
         Error␣opening␣file')
  ENDIF
  WRITE(fileunit,IOSTAT=ierror)   &
      EMULATION_2D_scattering_structure

  CLOSE(fileunit)

END SUBROUTINE write_EMULATION_2D_data_structure
```

EMULATION_2D MPI Broadcast (cont.)

Read from Disk

```
SUBROUTINE read_EMULATION_2D_data_structure(filename)

  CHARACTER(LEN=*), INTENT(IN) :: filename
  INTEGER:: ierror,fileunit

  OPEN(NEWUNIT=fileunit, FILE=filename, STATUS="OLD",  &
      ACTION="READ", FORM="UNFORMATTED",IOSTAT=ierror)
  IF (ierror /= 0) THEN
     CHKERRQ(ierror,'␣read_EMULATION_2D_data_structure:␣Error␣ &
         opening␣file')
  ENDIF
  READ(fileunit,IOSTAT=ierror)   &
      EMULATION_2D_scattering_structure

  CLOSE(fileunit)

END SUBROUTINE read_EMULATION_2D_data_structure
```

EMULATION_2D MPI Broadcast (cont.)

- MPI communication between processes within POST-MODE
  - At present it is simply by all MPI processes reading the EMULATION_2D data structure from disk (and not only process 0)
  - Use of custom MPI datatypes (simple prototype under test at present)
    - One MPI call: low latency, readability of the code
    - Maintenability of the code: routines defined on each module to create MPI datatype associated to each derived datatype.

EMULATION_2D MPI Broadcast (cont.)

Fortran Derived DataType

```
! Derived type to test MPI_Type_create_struct
TYPE :: my_derived_type_def
   LOGICAL :: mylogical = .TRUE.
   CHARACTER(len=4) :: mystring = 'hola'
   REAL(KIND=DBL) :: myreal = 1.0_DBL
END TYPE my_derived_type_def

TYPE(my_derived_type_def) :: my_derived_type
```

EMULATION_2D MPI Broadcast (cont.)

MPI datatype creation (MPI_Type_create_struct) (cont.)

```
!! Creamos el MPI Datatype que se use para pasar el tipo  &
    derivado
!! entre procesos MPI

CALL MPI_GET_ADDRESS(my_derived_type,address(1), ierror)
CALL MPI_GET_ADDRESS(my_derived_type%mylogical,address(2),  &
    ierror)
CALL MPI_GET_ADDRESS(my_derived_type%mystring,address(3),  &
    ierror)
CALL MPI_GET_ADDRESS(my_derived_type%myreal,address(4), ierror)

DO i=1,COUNT
   ! MPI-3.0 & former: disp = iaddr(i+1)-iaddr(i)
   displacements(i) = MPI_Aint_diff(address(i+1),address(i))
ENDDO

typelist(1) = MPI_LOGICAL
typelist(2) = MPI_CHARACTER
typelist(3) = MPI_DOUBLE_PRECISION
```

EMULATION_2D MPI Broadcast (cont.)

MPI datatype creation (MPI_Type_create_struct) (cont.)

```
block_lengths(1)=1
block_lengths(2)=4
block_lengths(3)=1

! build the derived data type
call MPI_Type_create_struct(COUNT,block_lengths,displacements,&
     typelist, my_mpi_derived_type_def,ierr)
if (ierr /= 0 ) then
   print *, 'got an error in type create: ', ierr
   call MPI_Abort(MPI_COMM_WORLD, ierr, ierr)
endif
```

EMULATION_2D MPI Broadcast (cont.)

Use of MPI datatype in MPI calls

```
! commit it to the system, so it knows we ll use it
! for communication
call MPI_TYPE_COMMIT(my_mpi_derived_type_def)
if (ierr /= 0 ) then
   print *, 'got␣an␣error␣in␣type␣commit:␣', ierr
   call MPI_Abort(MPI_COMM_WORLD, ierr, ierr)
endif

! use it
call  &
    MPI_BCAST(my_derived_type,1,my_mpi_derived_type_def,0,MPI_COMM_
```

Free MPI datatype created

```
! We free the MPI Datatype created
call MPI_TYPE_FREE(my_mpi_derived_type_def,ierr)
```

MPI Fortran interfaces

USE mpi_f08

```
 USE mpi_f08
!! It requires compile-time argument checking with unique MPI
!! handle types and provides techniques to fully solve the
!! optimization problems with nonblocking calls. This is the  &
    only
!! Fortran support method that is consistent with the Fortran
!! standard (Fortran 2008 + TS 29113 and later). This method is
!! highly recommended for all MPI applications.
```

USE mpi

```
 USE mpi
!! It requires compile-time argument checking. Handles are  &
    defined
!! as INTEGER. This Fortran support method is inconsistent  &
    with the
!! Fortran standard, and its use is therefore not recommended.
```

MPI Fortran interfaces (cont.)

INCLUDE 'mpif.h'

```
 INCLUDE 'mpif.h'
!!  The use of the include file mpif.h is strongly discouraged
!!  starting with MPI-3.0, because this method neither  &
    guarantees
!!  compile-time argument checking nor provides sufficient
!!  techniques to solve the optimization problems with  &
    nonblocking
!!  calls, and is therefore inconsistent with the Fortran  &
    standard.
```