

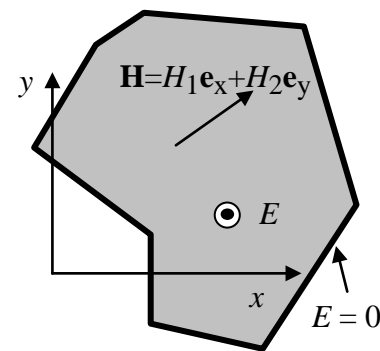
Lab 3: Finite elements for time-harmonic waveguide cross-section analysis

We consider the cross-section (x,y) -plane in a homogeneous (possibly dielectric-filled) wave guide with metallic outer surface. The field is independent of the longitudinal coordinate z , and the only non-zero electric field component is $E (=E_z)$, d.o for the \mathbf{H} -field H_1 and H_2 : a TM-field. The Maxwell equations for time-harmonic (angular frequency ω) excitation by an externally driven source current J (in z -direction) are:

$$\nabla \times \mathbf{H} = \mathbf{J} \mathbf{e}_z + \epsilon j \omega \mathbf{E} \mathbf{e}_z : \partial_x H_2 - \partial_y H_1 = J + \epsilon j \omega E$$

$$\nabla \times (E \mathbf{e}_z) = -\mu j \omega \mathbf{H} : \begin{aligned} \partial_y E &= -\mu j \omega H_1 \\ -\partial_x E &= -\mu j \omega H_2 \end{aligned}$$

It is your job to write matlab codes to do the calculations as explained below. Please formulate answers, with some form of plot or other computed documentation, as relevant for the questions in *italics* below.



1 Nodal finite elements for scalar equation

Eliminating the H -variables by the two last equations one obtains

$$H_1 = -\frac{1}{\mu j \omega} \partial_y E, H_2 = \frac{1}{\mu j \omega} \partial_x E;$$

$$\partial_x (\partial_x E) + \partial_y (\partial_y E) = \mu j \omega (J + \epsilon j \omega E) = \mu j \omega J - k^2 E$$

$$\Delta E + k^2 E = \mu j \omega J, \text{ Helmholtz' equation}$$

$$E = 0 \text{ on outer boundary.}$$

This scalar equation is approximated by taking E in the space spanned by the standard piece-wise linear “tent” basis functions on a triangularization of a polygonal approximation to the boundary, exactly as explained in the CEMbook pp. 92-102.

Your task is to:

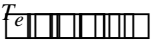
1. Include computation and assembly of the contributions from the terms

$k^2 E$ - the mass-matrix

$j \omega \mu J$ - the source function

The mass matrix elements are

$$m_{ij} = \int_{\Omega} \varphi_i \varphi_j d\Omega = \sum_e \int_{T_e} \varphi_i \varphi_j d\Omega ;$$


 element matrix, element e

$$\int_{T_e} \varphi_i \varphi_j d\Omega = \frac{A_e}{12} (1 + \delta_{ij}) \quad \text{where } i \text{ and } j \text{ run over the nodes of triangle } T_e$$

Use a point source or constant for the source J . Everything in the discretized equations is real, except the factor $j \omega \mu$ in the source, so replace it by 1.

2. Create the variables `no2xy`, `el2no`, `noInt` and `noExt` from data provided by the Matlab `PDEToolBox`; In the `PDEToolBox` documentation `no2xy` is called `p(1:2, 1:nnode)` (for points) and `el2no` is called `t(1:3, 1:nel)` (for

triangles). Do not attempt to extract boundary condition information, the encoding is complicated. Set up `noInt` and `noExt` yourself - node numbers of interior points, and exterior points (= on the conducting outer surface), the latter should have homogeneous Dirichlet conditions (see code p 102, `no_ess` etc.).

Use the file `pdedemo1` in the `pde` toolbox. It needs the function `circleg` which you want to modify for your own geometry; the original code creates a unit circle in four boundary segments each spanning 90° . Simply change the definition of the boundary curves from quarter-circles to the shape of your choice.

Use the `p` and `t` produced by the call to `initmesh`, set up `noInt` and `noExt` and call the solver routines from the book instead of the `PDEToolBox` solver routine `assemblpde`. Keep the loop which does the successive global refinements. You can use the `pdesurf` plotting routine or some graphics of your own device.

Q1: With your code, first try it (with $k^2 = 0$) on the unit circle with $\Delta E = 1$ and compare with the exact solution, just as `pdedemo1` does. Plot i) l_2 -norm of error ii) max-norm of error vs. number of elements in log-log plot. What order of convergence do you observe?

Q2: Then run a sequence of k -values and plot the l_2 -norm of the solution vs. k . Check that the resonances are correct. Hint: Avoid re-assembling the matrices for every k -value, as follows:

$$\Delta E + k^2 E = f$$

$\mathbf{S}\mathbf{e} + k^2 \mathbf{M}\mathbf{e} = \mathbf{F}$, $\mathbf{e} = (e_1, e_2, \dots, e_{N_{\text{node}}})^T$ is the vector of approximate E -fields at the nodes,

$$E_h(x, y) = \sum_{n=1}^{N_{\text{node}}} e_n \varphi_n(x, y)$$

$$\mathbf{F} = (f_1, f_2, \dots, f_{N_{\text{node}}})^T, f_k = \int_{\Omega} f(x, y) \varphi_k(x, y) d\Omega \text{ is the load vector,}$$

\mathbf{S} the stiffness matrix of the Laplace operator, and \mathbf{M} the mass matrix

Compute the stiffness matrix \mathbf{S} and the mass-matrix \mathbf{M} and the load vector \mathbf{F} once, then for the list of k 's form $\mathbf{S} + k^2 \mathbf{M}$ and solve.

Q3: Compute the smallest magnitude eigenvalues of the generalized eigenvalue problem

$$\Delta E + k^2 E = 0$$

$$\mathbf{S}\mathbf{e} + k^2 \mathbf{M}\mathbf{e} = 0: \mathbf{S}\mathbf{e} = \lambda \mathbf{M}\mathbf{e}, \lambda = -k^2$$

by matlab's `eigs` for sparse matrices; here is part of the help:

```
EIGS Find a few eigenvalues and eigenvectors of a matrix using
ARPACK.
...
EIGS(A,B) solves the generalized eigenvalue problem A*V == B*V*D.
B must be symmetric (or Hermitian) positive definite and the same
size as A.
EIGS(A,[],...) indicates the standard eigenvalue problem A*V == V*D.
...
```

For the circle case the eigensolutions are known in terms of Bessel functions. What order of convergence do you observe for the eigenvalues, and for the **E**-field?

Q4: **H** has to be computed from *E* by differentiation. We have

$$\mathbf{H}_h = \frac{1}{j\omega\mu} \mathbf{e}_z \times \nabla E_h, \nabla E_h = \sum_{n=1}^{N_{node}} e_n \nabla \varphi_n$$

and you can obtain the gradients of the basis functions from element geometry (see CEMbook pp. xxx). Compute the l2-norm of difference of **H** computed at triangle centroids to exact eigenfields and comment on the order of convergence observed. First confine your analysis to eigenmodes with only radial variation. For shapes with symmetries, an eigenvalue may have multiple eigenvectors, spanning a linear subspace. The numerically computed eigensolutions can be approximations to any vector in the subspace. For the circle, eigenfunctions with angular variation are multiple (double). To resolve this problem, you could e.g. find the best linear combination in the exact eigenspace to a computed eigenvector by least squares, and take the residual as error.

2 Edge finite elements for time harmonic curl-curl vector equation

Use your nodal element code as template to solve the sequence of *k*-values and the eigenvalue problem with the matrices and approximate fields computed by the **edgeFEM2D** edge element code in the book. You can export element meshes from PDEToolBox as before, but now you have to use the graphics code from the book because PDEToolBox does not know about edge elements. You need to write code to compute the load vector, so first derive the necessary formulas and see if you can re-use pieces from the CEM Book code. If you are unfamiliar with vector elements, please take time to study the code which uses Matlab vectorized operations to become quite compact. Think about signs, element areas, and determinants to get things right.

Q5: How do you compute **E** from **H**?

Q6: For the current-excitation model, note that the **H**-field is excited by the curl of the current. How do you propose to implement a point source?

Q7: What order of convergence do you now observe for eigenvalues and **H**-fields? The **H**-field accuracy should be better than with the nodal elements, but how much?

Q8: Try another geometry of your choice, one with sharp internal corners so the **E**-field becomes singular. What order of convergence do you observe for the **H**-field?

Good Luck !