Electromagnetic scattering I (53919, 5 cr)
Exercise 4: FEM

1. Consider the wave equation in a bounded source free region Ω with the PEC boundary condition on the boundary $\partial\Omega$

$$\begin{aligned} \nabla \times \mu_r^{-1} \nabla \times \mathbf{E} - k^2 \epsilon_r \mathbf{E} &= 0, \text{ in } \Omega \\ \mathbf{n} \times \mathbf{E} &= 0, \text{ on } \partial\Omega \end{aligned} \tag{1}$$

Derive the corresponding weak formulation in $H_{0,curl}$.

2. Discretize the derived weak equation using the lowest order curl-conforming edge-elements as basis $\mathbf{b}_n$ and testing $\mathbf{t}_m$ functions. Write a code that computes the required stiffness and mass matrices

$$S_{mn} =< \nabla \times \mathbf{t}_m, \mu_r^{-1} \nabla \times \mathbf{b}_n >_{L^2(\Omega)} \tag{2}$$

$$M_{mn} =< \mathbf{t}_m, \epsilon_r \mathbf{b}_n >_{L^2(\Omega)}, \tag{3}$$

and solve the corresponding generalized eigenvalue problem

$$S_{mn} c_n = \omega^2 M_{mn} c_n. \tag{4}$$

"cube170.mat"-file contains the data structures for the tetrahedral mesh (a unit cube discretized with 170 tetrahedra), and the Gaussian quadrature points $P0(\xi;\eta;\zeta)$ and weights $w0$ for the reference tetrahedron. The generalized eigenvalue problem can be solved with Matlab's EIG(S,M) function. Find a few lowest (e.g. 50) non-zero eigenvalues and compare them with the exact eigenvalues $k^2 = \pi^2(l^2 + m^2 + n^2)$ $l, m, n = 0, 1, \ldots$ $st.$ $lm + ln + mn > 0$. Try different discretization densities("cube638.mat","cube2433.mat") What can you say about the accuracy of the FEM solution?

Pseudo-code for the problem 2

Algorithm 1 FEM matrix assembly

```
Calculate shape functions N0 and their gradients dN0 in the reference element
[N0_1; N0_2; N0_3; N0_4] = ...
[dN0_1; dN0_2; dN0_3; dN0_4] = ...
Allocate sparse matrices S(N_e, N_e) M(N_e, N_e)
S = spare(Ne,Ne); M = spare(Ne,Ne);

//Loop over elements
for k = 1 to N_k do
    // Compute mapping and Jacobian
    𝓕_k = ... ; J_{𝓕_k} = ...
    // Compute gradients of shape functions
    dN = [∇N_1, ∇N_2, ∇N_3, ∇N_4]
    //Compute integral
    for i = 1 to 4 do
        for j = 1 to 4 do
            I(i, j) ← ∫_{T_k} N_i N_j dV
        end for
    end for
    // Loop over edges of tetrahedron k
    for et = 1 to 6 do
        m = mesh.etopol2(et,k) //global index of testing function
        tk = ..., // local index for node1 [1]
        tl = ..., // local index for node2
        signt = ... [2]
        for eb = 1 to 6 do
            n = mesh.etopol2(eb,k) //global index of basis function
            bk = ..., // local index for node1
            bl = ..., // local index for node2
            signb = ... [2]
            // Calculate matrix elements
            S(m, n) = S(m, n) + signt * signb * ... [3]
            M(m, n) = M(m, n) + signt * signb * ... [4]
        end for
    end for

end for
Enforce the PEC boundary condition
```

1. Local nodal indices are defined as

localedge	1	2	3	4	5	6
*node*1	1	1	1	2	2	3
*node*2	2	3	4	3	4	4

2. If the global and local edge have the same orientation (mesh.edges(1,m) = mesh.etopol(tk,k)) the sign of the testing/basis function should be positive and negative otherwise

3. Lecture notes: equation (68)

4. Lecture notes: equation (69)

5. Remove rows and columns of matrices S and M associated with the boundary edges