SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 6785

IMPLEMENTACIJA METODE KONAČNIH ELEMENATA U RAZVOJNOM OKRUŽENJU FENICS

Igor Kladarić

Zagreb, lipanj 2020.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 6785

IMPLEMENTACIJA METODE KONAČNIH ELEMENATA U RAZVOJNOM OKRUŽENJU FENICS

Igor Kladarić

Zagreb, lipanj 2020.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 13. ožujka 2020.

ZAVRŠNI ZADATAK br. 6785

Pristupnik:	**Igor Kladarić (0036505415)**
Studij:	Elektrotehnika i informacijska tehnologija
Modul:	Bežične komunikacijske tehnologije
Mentor:	doc. dr. sc. Dario Bojanjac

Zadatak: **Implementacija metode konačnih elemenata u razvojnom okruženju FEniCS**

Opis zadatka:

Metoda konačnih elemenata je numerička metoda za rješavanje parcijalnih diferencijalnih jednadžbi s varijabilnim koeficijentima na nesimetričnim geometrijama. Odabir numeričke metode ovisi o primjenama, vrsti jednadžbe te željenim svojstvima numeričke metode. U radu je potrebno opisati teorijsku pozadinu slabe formulacije eliptičke diferencijalne jednadžbe te objasniti prostor konačnih elemenata za njenu diskretizaciju. Primijeniti metodu na različite rubne uvjete i desne strane jednadžbe. Opisati osnovne metode diskretizacije konačnim razlikama u vremenu te primijeniti prostorno vremensku diskretizaciju na jednadžbu provođenja. Implementirati navedene metode u razvojnom okruženju FEniCS.

Rok za predaju rada: 12. lipnja 2020.

SADRŽAJ

1. Uvod **1**

2. Metoda konačnih elemenata **2**

2.1. Prostor neprekidnih linearnih funkcija po dijelovima 2

2.2. L^2 - projekcija 3

2.3. Slaba formulacija 4

2.4. Rješavanje jednadžbi 5

3. Rješavanje Poissonove jednadžbe u razvojnom okruženju FEniCS **7**

4. Računanje i vizualizacija elektromagnetskih modova u valovodu **13**

4.1. Problem svojstvene vrijednosti 13

4.2. Implementacija problema svojstvene vrijednosti 16

4.2.1. Longitudinalne komponente 16

4.2.2. Transverzalne komponente 19

4.3. Pravokutni valovod 20

4.4. Cilindrični valovod 27

5. Interferencija elektromagnetskih valova u vremenu **32**

5.1. Vremenska diskretizacija 32

5.2. Simulacija interferencije 33

6. Zaključak **37**

Literatura **38**

1. Uvod

Prirodne pojave poput širenja elektromagnetskog vala ili širenja topline u materijalu najčešće možemo opisati postavljanjem i rješavanjem parcijalnih diferencijalnih jednadžbi. U rijetkim slučajevima, kada su te jednadžbe jednostavne i kada je geometrija prostora koji promatramo simetrična, rješenja tih jednadžbi možemo pronaći analitički, no kad je sustav kompliciraniji, moramo se okrenuti numeričkim metodama rješavanja jednadžbi.

Jedna od takvih metoda je metoda konačnih elemenata (eng. FEM - *Finite Element Method*). Ova metoda omogućuje podjelu promatranog sustava na dijelove koji se zatim opisuju konačnim brojem algebarskih izraza te tako rješavaju. Na taj način dobivamo, ne u potpunosti točno, ali dovoljno dobro aproksimativno rješenje zadanog sustava. Programska implementacija metode konačnih elemenata često se provodi u razvojnom okruženju FEniCS.

U ovome radu bit će opisana teorijska podloga slabe formulacije eliptičke diferencijalne jednadžbe koja čini osnovu metode konačnih elemenata te prikazana njena implementacija uz pomoć FEniCS-a na dva primjera - za računanje dozvoljenih elektromagnetskih modova u savršenom metalnom valovodu, te za simulaciju interferencije elektromagnetskih valova u vremenu i prostoru.

2. Metoda konačnih elemenata

Metodu konačnih elemenata moguće je primjeniti na različite načine, no zbog jednostavnosti, bavit ćemo se tzv. Lagrangeovim elementima na kojim je najlakše prikazati osnovni koncept FEM-a.. Pritom, metoda će biti prikazana na primjeru rješavanja Poissonove jednadžbe u dvodimenzionalnom prostoru Ω,

$$-\triangle u = f, \quad \text{na } \Omega \tag{2.1}$$

$$u = 0, \quad \text{na } \partial\Omega \tag{2.2}$$

gdje je $\partial\Omega$ rub promatranog prostora.

Ovaj oblik metode konačnih elemenata ćemo prikazati ukratko, a detaljniji postupak opisan je u Larson i Bengzon (2010).

2.1. Prostor neprekidnih linearnih funkcija po dijelovima

Triangulirajmo prostor Ω, odnosno podjelimo ga na konačan broj trokuta. Triangulacijom dobivamo n_p čvorova kojim su trokuti definirani, stoga imenujmo svaki čvor N_i, gdje je $i = 1, 2, ...n_\mathrm{p}$ te nazovimo ovaj triangulirani prostor $\mathscr{K}$.

Neka je $K \in \mathscr{K}$ jedan trokut u prostoru. Definirajmo sada prostor linearnih funkcija:

$$\mathscr{P}(K) = \{v : v = c_0 + c_1 x_1 + c_2 x_2, (x_1, x_2) \in K, c_0, c_1, c_0 \in \mathbb{R}\}$$

Odnosno, vrijednosti na svakom trokutu su definirane linearnim funkcijama. Vrijednosti funkcija po segmentima (trokutima) smo mogli definirati i drukčije, polinomima višeg stupnja, no ovdje smo ih u skladu s Lagrangeovim elementima definirali kao polinome prvog stupnja.

Definirajmo sada prostor kontinuiranih linearnih funkcija po dijelovima V_h tako da funkcija na cijelom prostoru $\mathscr{K}$ bude neprekidna:

$$V_\mathrm{h} = \{v : v \in \mathscr{C}(\Omega), v|_\mathrm{K} \in \mathscr{P}(K) \forall K \in \mathscr{K}\},$$

gdje je $\mathscr{C}(\Omega)$ prostor svih mogućih neprekidnih funkcija na Ω. Definirajmo još funkcijski prostor $V_{\mathrm{h},\,0}$ kao podskup prostora V_h funkcija koje iščezavaju na rubu:

$$V_{\mathrm{h},\,0} = \{v \in V_\mathrm{h}, v|_{\partial\Omega} = 0\},$$

Na kraju, definirajmo funkcije $\phi_\mathrm{j} \in V_\mathrm{h}$:

$$\phi_j = \begin{cases} 1, i = j \\ 0, i \neq j \end{cases} \quad i, j = 1, 2, ..., n_\mathrm{p} \tag{2.3}$$

Ovako možemo svaku funkciju $v \in V_\mathrm{h}$ zapisati kao

$$v = \sum_{i=1}^{n_\mathrm{p}} \mathcal{E}_\mathrm{i} \phi_\mathrm{i}, \tag{2.4}$$

pri čemu $\mathcal{E}_\mathrm{i} = v(N_\mathrm{i})$, $i = 1, 2, ..., n_\mathrm{p}$.

2.2. L^2 - projekcija

L^2 - projekciju ($P_\mathrm{h} f$) definiramo kao projekciju funkcije $f : \Omega \to \mathbb{R}^n$ na prostor V_h koja najbolje aproksimira funkciju f na tom prostoru. Za nju vrijedi:

$$\int_\Omega (f - P_\mathrm{h} f) v dx = 0, \qquad \forall v \in V_\mathrm{h} \tag{2.5}$$

Može se pokazati da s obzirom da je V_h prostor dimenzije n_p, iz (2.5) slijedi

$$\int_\Omega (f - P_\mathrm{h} f) \phi_\mathrm{i} dx = 0, \qquad i = 1, 2, ..., n_\mathrm{p}. \tag{2.6}$$

Uzimajući u obzir činjenicu da je $\mathrm{P_h f} \in \mathrm{V_h}$, možemo umjesto $\mathrm{P_h f}$ pisati

$$P_\mathrm{h} f = \sum_{i=1}^{n_\mathrm{p}} \mathcal{E}_\mathrm{i} \phi_\mathrm{i}, \tag{2.7}$$

i dobiti

$$\int_\Omega f\phi_\mathrm{i}dx = \int_\Omega (P_\mathrm{h}f)\phi_\mathrm{i}dx = \int_\Omega (\sum_{j=1}^{n_\mathrm{p}} \mathcal{E}_\mathrm{j}\phi_\mathrm{j})\phi_\mathrm{i}dx \tag{2.8}$$

$$= \sum_{i=1}^{n_\mathrm{p}} \mathcal{E}_\mathrm{j} \int_\Omega \phi_\mathrm{j}\phi_\mathrm{i}dx \tag{2.9}$$

Definiramo li

$$M_\mathrm{ij} = \int_\Omega \phi_\mathrm{j}\phi_\mathrm{i}dx, \qquad i = 1, 2, ..., n_\mathrm{p}, \tag{2.10}$$

$$b_\mathrm{i} = \int_\Omega f\phi_\mathrm{i}dx, \qquad i, j = 1, 2, ..., n_\mathrm{p}, \tag{2.11}$$

onda (2.8) i (2.9) možemo u matričnom obliku zapisati kao

$$M\mathcal{E} = b, \tag{2.12}$$

Rješavajući ovu jednadžbu po $\mathcal{E}$ i uvrštavajući u (2.7), dobivamo našu L^2 - projekciju funkcije f.

2.3. Slaba formulacija

Vratimo se natrag na naš sustav (2.1) i (2.2). Kako bismo ga mogli riješiti pomoću metode konačnih elemenata, moramo ga zapisati u tzv. slaboj formulaciji (eng. *weak formulation*) o kojoj se više može pronaći također u Larson i Bengzon (2010).
Slabu formulaciju dobivamo množeći obje strane jednadžbe s probnom funkcijom $v \in V_\mathrm{h}$ i integrirajući obje strane po čitavom prostoru Ω.

$$\int_\Omega (-\triangle u)vdx = \int_\Omega fvdx \tag{2.13}$$

Provedemo li parcijalnu integraciju nad lijevom stranom jednadžbe dobijemo

$$\int_\Omega (-\triangle u)vdx = \int_\Omega \nabla u\nabla vdx - \int_{\partial\Omega} n \cdot \nabla uvdx. \tag{2.14}$$

S obzirom da prema rubnom uvjetu (2.2) u iščezava na rubu, pretpostavka je da to isto čini i probna funkcija v pa se prethodni izraz pojednostavljuje na

$$\int_\Omega \nabla u \nabla v dx = \int_\Omega f v dx, \qquad \forall v \in V_{\mathrm{h,0}}. \tag{2.15}$$

Ovu jednadžbu nazivamo slabom formulacijom.

Nas zanima rješenje u koje možemo dobiti iz ove jednadžbe, što će biti prikazano u sljedećem poglavlju.

2.4. Rješavanje jednadžbi

Potpuno analitički točno rješenje u ne možemo dobiti iz dobivene jednadžbe, no ono što možemo izračunati je njenu L^2 - projekciju u_h.
Za L^2 - projekciju vrijedi izraz (2.5), što znači da varijablu u izvodu u prethodnom poglavlju možemo zamijeniti s L^2 - projekcijom. To znači da vrijedi:

$$\int_\Omega \nabla u_\mathrm{h} \nabla v dx = \int_\Omega f v dx, \qquad \forall v \in V_{\mathrm{h,0}} \tag{2.16}$$

Odnosno, kao i kod (2.6) vrijedi:

$$\int_\Omega \nabla u_\mathrm{h} \nabla \phi_\mathrm{i} dx = \int_\Omega f \phi_\mathrm{i} dx, \qquad i = 1, 2, ..., n_\mathrm{p} \tag{2.17}$$

u_h je L^2 - projekcija što znači da je $\mathrm{u_h} \in \mathrm{V_{h,0}}$, odnosno, možemo ju zapisati kao

$$u_\mathrm{h} = \sum_{j=1}^{n_\mathrm{p}} \mathcal{E}_\mathrm{j} \phi_\mathrm{j}. \tag{2.18}$$

Uvrštavanjem (2.18) u (2.17) dobivamo

$$\sum_{j=1}^{n_\mathrm{p}} \mathcal{E}_\mathrm{j} \int_\Omega \nabla \phi_\mathrm{j} \nabla \phi_\mathrm{i} dx = \int_\Omega f \phi_\mathrm{i} dx. \tag{2.19}$$

Zamjenimo li

$$A_{\mathrm{ij}} = \int_{\Omega} \nabla \phi_{\mathrm{j}} \nabla \phi_{\mathrm{i}} dx, \tag{2.20}$$

$$b_{\mathrm{i}} = \int_{\Omega} f \phi_{\mathrm{i}} dx, \tag{2.21}$$

dobivamo

$$b_{\mathrm{i}} = \sum_{j=1}^{n_{\mathrm{p}}} A_{\mathrm{ij}} \mathcal{E}_{\mathrm{j}}. \tag{2.22}$$

Odnosno, u matričnom obliku

$$A\mathcal{E} = b. \tag{2.23}$$

Rješavanjem jednandžbe (2.23) po $\mathcal{E}$, možemo dobiti u_{h} prema (2.18), što je upravo naše traženo rješenje sustava u prostoru koji smo definirali triangulacijom prostora Ω.

3. Rješavanje Poissonove jednadžbe u razvojnom okruženju FEniCS

Razvojno okruženje FEniCS je platforma za rješavanje parcijalnih diferencijalnih jednadžbi, a mi ćemo ju uz pomoć Python-a koristiti za rješavanje problema u elektromagnetizmu. Detaljnije upute o osnovama FEniCS-a mogu se pronaći u Langtangen i Logg (2017).

Osnovne dijelove programa prikazat ćemo na primjeru rješavanja Poissonove jednadžbe (2.1). Jedan primjer pojave Poissonove jednadžbe u prirodi je raspodjela električnog potencijala u prostoru s nabojima.

Gaussova jednadžba za električno polje u praznom prostoru glasi:

$$\nabla \cdot E = \frac{\rho}{\epsilon_0}. \tag{3.1}$$

Pretpostavimo li da je sustav u elektrostatici, možemo električno polje zapisati pomoću električnog potencijala

$$E = -\nabla\phi \tag{3.2}$$

te tako dobijemo Poissonovu jednadžbu

$$-\triangle\phi = \frac{\rho}{\epsilon_0}. \tag{3.3}$$

Zamjenimo li $\phi = u$ i $\frac{\rho}{\epsilon_0} = f$, dobivamo upravo jednadžbu (2.1) koju smo naučili riješiti metodom konačnih elemenata uz rubni uvjet (2.2).

Stoga, riješimo ovu jednadžbu na primjeru jednog točkastog naboja u prostoru.

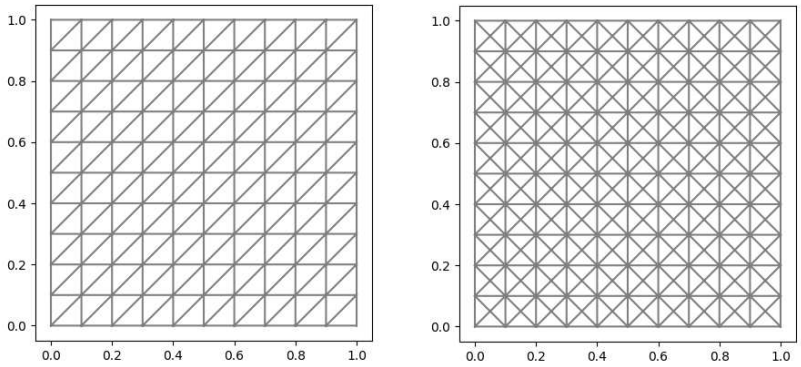
Svaki program započinjemo s jednom od sljedeće dvije linije kako bismo mogli koristiti FEniCS paket:

```
from fenics import *
from dolfin import *
```

Prvo, stvorimo trianguliziran prostor u obliku pravokutnika na kojem ćemo računati raspodjelu električnog potencijala:

```
mesh = RectangleMesh(Point(0, 0), Point(1, 1),
                     40, 40, "crossed")
```

Prva dva člana u funkciji definiraju dvije točke između kojih stvaramo pravokutnik, a druga dva člana određuju broj trokuta koje stvaramo na x i y osi. Posljednjim članom ("crossed") naznačujemo da želimo da se prostor podjeli na trokutove tako da održimo simetričnost te dobijemo točniji rezultat.



Slika 3.1: Mreža bez člana "crossed" (lijevo) i s naznačenim članom "crossed" (desno)

Zatim, definirajmo funkcijski prostor V_h:

```
V = FunctionSpace(mesh, "Lagrange", 1)
```

Ovako stvaramo Lagrangeove elemente prog stupnja kao što smo definirali u prethodnoj cjelini.

Definirajmo sad rubne uvjete:

```
def boundary(x, on_boundary):
    return on_boundary

u0 = Constant(0)
bc = DirichletBC(V, u0, boundary)
```

Prvo smo definirali funkciju `boundary()` koja definira što smatramo granicom, u ovom slučaju to je rub prostora koji smo definirali ranije. Zatim smo sa zadnje dvije linije odredili prema (2.1) da rubni uvjeti budu 0.

Definirajmo sada probnu i testnu funkciju:

```
u = TrialFunction(V)
v = TestFunction(V)
```

Funkcija u nam u ovom slučaju predstavlja električni potencijal ϕ.

Sad, moramo definirati pobudu koja je u našem slučaju točkasti naboj. S obzirom da se radi o točkastom naboju dok je ostatak prostora prazan, pobudu ćemo definirati kao 0, odnosno da je nema na čitavom prostoru, a točkasti naboj ćemo dodati naknadno. Neka se točkasti naboj pritom nalazi točno na sredini našeg pravokutnog prostora, u točki (0.5, 0.5):

```
f = Constant(0)
charge = PointSource(V, Point(0.5, 0.5), 1)
```

Nakon ovoga, konačno možemo zapisati slabu formulaciju našeg problema (2.15):

```
a = inner(grad(u), grad(v))*dx
L = Constant(0)*v*dx
```

Ovako smo definirali lijevu i desnu stranu jednadžbe slabe formulacije, odnosno što se nalazi unutar integrala s obje strane.

Napokon, možemo sastaviti sustav jednadžbi (2.23) funkcijom `assemble_system()`:

```
A, b = assemble_system(a, L, bc)
```

S obzirom da u slaboj formulaciji nismo uzeli u obzir točkasti naboj - ključni dio našeg zadatka, naknadno ga dodajemo pobudi - vektoru `b`:

```
charge.apply(b)
```

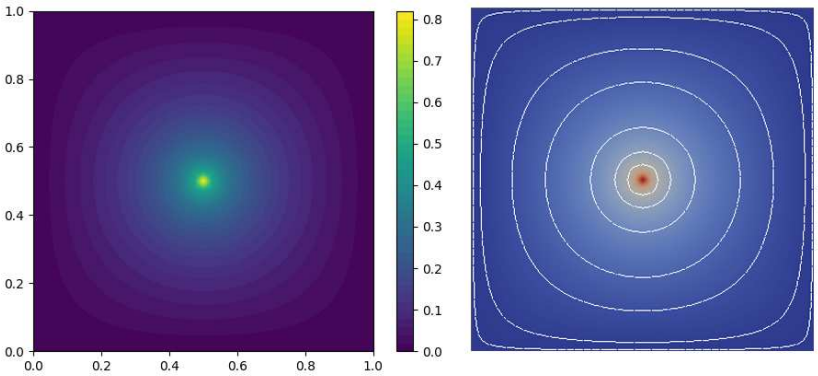
Sad možemo riješiti sustav po varijabli `u`:

```
u = Function(V)
solve(A, u.vector(), b)
```

Nakon što smo riješili sustav jednadžbi, možemo vizualizirati rješenje direktno:

```
import matplotlib.pyplot as plt
c = plot(u)
plt.colorbar(c)
plt.show()

vkkfile = File('poisson.pvd')
vtkfile << (u)
```

Pomoću prve četiri linije dobivamo prikaz električnog potencijala odmah nakon pokretanja programa, dok pomoću preostale dvije spremamo rezultat u dokument $poisson.pvd$, koji zatim možemo pregledati u programu $Paraview$ i vidjeti kako izgledaju ekvipotencijalne linije.



Slika 3.2: Električni potencijal u okolici pozitivnog točkastog naboja s rubnim uvjetima jednakima 0 na granici.

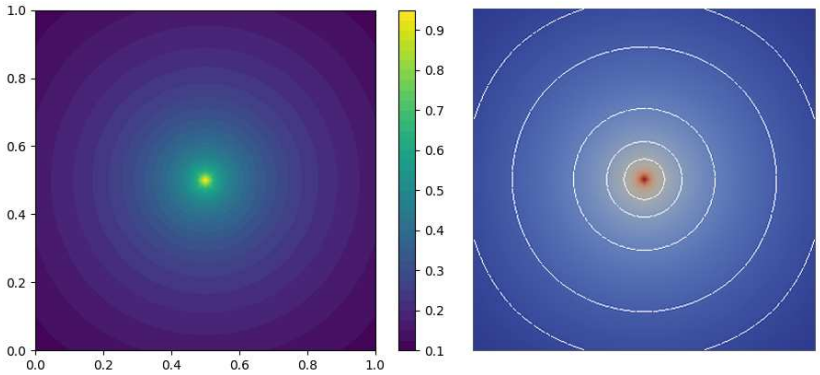
Vidimo da je električni potencijal maksimalan u sredini, ali iščezava na rubu zbog zadanog rubnog uvjeta. Zbog tog rubnog uvjeta prikaz električnog potencijala nije isti kao onakav kakvog ga intuitivno zamišljamo u okolici točkastog naboja. Da bismo dobili takav prikaz, moramo definirati da je električni potencijal jednak nuli u beskonačnosti, no mi možemo definirati samo rubne uvjete na granici našeg prostora.
Stoga uzmimo izraz za potencijal točkastog naboja

$$\phi = \frac{Q}{4\pi\epsilon_0} \cdot \frac{1}{r}, \tag{3.4}$$

gdje je r udaljenost od točkastog naboja i primjenimo ga na rubu prostora na kojem računamo. Pritom, zbog jednostavnosti zapisa, uzmimo da je $\epsilon_0 = 1$.

```
def boundary(x, on_boundary):
    return on_boundary

u0 = Expression('1/(4*3.14159*sqrt((x[0]-0.5)*(x[0]-0.5)
                +(x[1]-0.5)*(x[1]-0.5)))', degree=2)
bc = DirichletBC(V, u0, boundary)
```

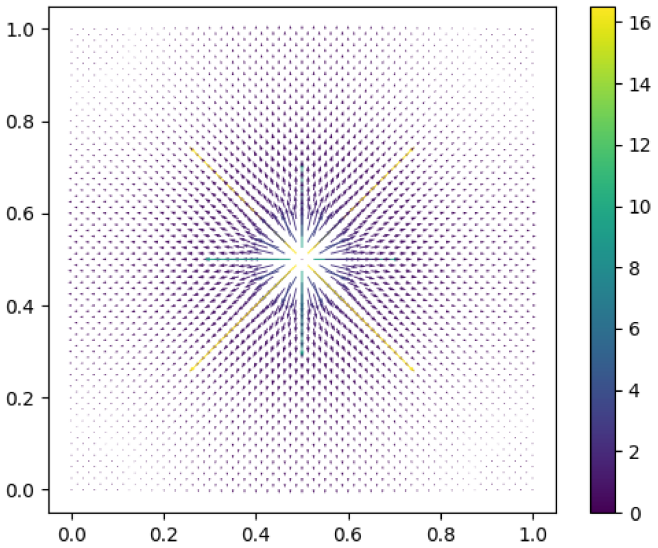
S ovako definiranim rubnim uvjetima dobivamo očekivani rezultat:



Slika 3.3: Električni potencijal u okolici pozitivnog točkastog naboja s rubnim uvjetima jednakima 0 u beskonačnosti.

Kad bi nas zanimalo električno polje u ovom sustavu, umjesto da ispisujemo funkciju u koja predstavlja električni potencijal, tražili bismo prema (3.2) ispisivati `-grad(u)`:

```
import matplotlib.pyplot as plt

c = plot(-grad(u))
plt.colorbar(c)
plt.show()
```



Slika 3.4: Električno polje u okolici pozitivnog točkastog naboja s rubnim uvjetima jednakima 0 u beskonačnosti.

Vidimo da je električno polje usmjereno radijalno od točkastog naboja te se smanjuje s udaljenošću od njega što upravo i očekujemo.

Ovim primjerom smo pokazali osnovne funkcije i sposobnosti FEniCS paketa.

4. Računanje i vizualizacija elektromagnetskih modova u valovodu

Valovod je struktura koja omogućava vođenje elektromagnetskih valova. Ovisno o obliku i svojstvima materijala od kojih se valovod sastoji i kako su oni raspoređeni, valovod omogućuje propagaciju samo nekih oblika i frekvencija elektromagnetskih valova. Primjerice, savršeni metalni valovod osigurava da električno polje paralelno s rubom valovoda na rubu bude nula zbog svojstva refleksije savršenog vodiča.

Upravo ćemo za takav savršeni metalni valovod pokušati pomoću FEniCS-a izračunati dozvoljene elektromagnetske modove te prikazati raspodjelu električnog i magnetskog polja tih modova unutar valovoda.

4.1. Problem svojstvene vrijednosti

Električno i magnetsko polje moraju zadovoljavati Maxwellove jednadžbe. one u diferencijalnom obliku glase:

$$\nabla \cdot E = \frac{\rho}{\mathcal{E}} \tag{4.1}$$

$$\nabla \cdot B = 0 \tag{4.2}$$

$$\nabla \times E = -\frac{\partial B}{\partial t} \tag{4.3}$$

$$\nabla \times B = \mu J + \mu \mathcal{E} \frac{\partial E}{\partial t} \tag{4.4}$$

Uzimat ćemo da je naš prostor prazan, odnosno da ne sadrži nikakvu raspodjelu naboja niti gustoća struja, $\rho = 0, J = 0$. Također, uzmimo da vrijedi $B = \mu H$.
Ako pretpostavimo sinusnu pobudu i zapišemo jednadžbe u frekvencijskoj domeni, dobijemo:

$$\nabla \cdot E = 0 \tag{4.5}$$

$$\nabla \cdot B = 0 \tag{4.6}$$

$$\nabla \times E = -j\omega\mu H \tag{4.7}$$

$$\nabla \times H = j\omega\epsilon E \tag{4.8}$$

Djelujemo li operatorom rotacije na obje strane jednadžbe (4.7)

$$\nabla \times \nabla \times E = -j\omega\mu\nabla \times H, \tag{4.9}$$

te uvrštavanjem (4.8), dobivamo:

$$\nabla \times \nabla \times E = \mu\epsilon\omega^2 E, \tag{4.10}$$

Uzmemo li u obzir da vrijedi

$$\nabla \times \nabla \times E = \nabla(\nabla \cdot E) - \triangle E, \tag{4.11}$$

te uvrstimo (4.5), dobivamo:

$$\triangle E + \mu\epsilon\omega^2 E = 0 \tag{4.12}$$

što nazivamo Helmholtzovom jednadžbom.
$\mu\epsilon\omega^2$ je nekakva vrijednost koja ovisi o frekvenciji ω pa ju možemo zamijeniti s arbitrarnom varijablom λ. Tako dobivamo jednadžbu

$$\triangle E + \lambda E = 0, \tag{4.13}$$

odnosno problem svojstvene vrijednosti.
Ista jednadžba se može dobiti za magnetsko polje H.

$$\triangle H + \lambda H = 0, \tag{4.14}$$

Vrijednost λ nazivamo svojstvenom vrijednosti sustava koja nam govori o frekvenciji moda koji može postojati u valovodu kojeg promatramo, a električno polje E i magnetsko polje H su svojstveni vektori koji ovise o dobivenoj svojstvenoj vrijednosti.
Naš zadatak je pronaći koje svojstvene vrijednosti omogućuju postojanje rješenja ove jednadžbe za određeni oblik valovoda te kako ta rješenja izgledaju.

Polja E i H imaju longitudinalnu i transverzalnu komponentu. Longitudinalna komponenta je paralelna, a transverzalna komponenta okomita s osi valovoda. Ako mod ima samo transverzalnu komponentu električnog polja nazivamo ga TE modom (*Transverse Electric*) dok ako ima samo transverzalnu komponentu magnetskog polja, nazivamo ga TM modom (*Transverse Magnetic*). Može se pokazati da se svi modovi u savršenom metalnom valovodu mogu prikazati kao linearna kombinacija TE i TM modova.

Kako bismo pronašli zadovoljavajuće modove, moramo jednadžbe (4.13) i (4.14) zapisati u slaboj formulaciji te definirati rubne uvjete sustava. Provedemo li isti postupak kao u poglavlju 2.3, dobivamo slabu formulaciju:

$$\int_\Omega \nabla u \nabla v dx = \lambda \int_\Omega uv dx, \qquad \forall v \in V_{\mathrm{h},0}, \tag{4.15}$$

pri čemu u predstavlja ili polje E, ili polje H, ovisno o tome po kojoj varijabli rješavamo sustav.

Rubni uvjeti sustava također ovise o polju kojeg računamo. Neka je $\vec{n}$ normala na rub $\partial\Omega$ prostora Ω. Tangencijalna komponenta električnog polja na površini savršenog vodiča mora iščezavati. Stoga, ako tražimo električno polje, ono mora zadovoljavati Dirichletov rubni uvjet

$$E \times \vec{n} = 0 \quad na\, \partial\Omega. \tag{4.16}$$

Ako tražimo magnetsko polje, ono mora zadovoljavati rubni uvjet

$$H \cdot \vec{n} = 0 \quad na\, \partial\Omega, \tag{4.17}$$

iz kojega se, zajedno s (4.16) i (4.8), može dobiti Neumannov rubni uvjet za magnetsko polje

$$\frac{\partial H}{\partial \vec{n}} = 0 \quad na\, \partial\Omega. \tag{4.18}$$

4.2. Implementacija problema svojstvene vrijednosti

Kako bismo pronašli modove elektromagnetskih valova unutar valovoda, moramo riješiti problem svojstvene vrijednosti prikazan u prethodnom poglavlju. S obzirom da su TE i TM modovi definirani postojanjem longitudinalnih komponenti električnog i magnetskog polja, a preostale komponente se mogu izraziti preko njih, upravo možemo samo njih tražiti te tako pojednostaviti postupak računanja. Stoga ćemo prvo prikazati kako za TE modove računati longitudinalnu komponentu magnetskog, a za TM modove longitudinalnu komponentu električnog polja, a nakon toga kako direktno računati i transverzalne komponente.

4.2.1. Longitudinalne komponente

Prvo započnimo s pozivanjem svih potrebnih paketa:

```
from dolfin import *
import matplotlib.pyplot as plt
from dolfin.cpp.mesh import *
from mshr import *
```

Prva linija poziva FEniCS paket, druga `matplotlib` paket za iscrtavanje dobivenih rješenja, a posljednje dvije nam omogućuju kreiranje željenih mreža koje definiraju prostor na kojemu računamo, odnosno oblik valovoda u ovom zadatku.

Ako računamo TM modove, odnosno rješavamo po longitudinalnoj komponenti električnog polja, koristimo jednadžbu (4.13) te sustav mora zadovoljiti Dirichletov rubni uvjet (4.16). Prostor funkcija definiranih na mreži `mesh`, koju ćemo definirati kasnije, i rubne uvjete u ovom slučaju definiramo na sljedeći način:

```
V = FunctionSpace(mesh, "Lagrange", 1)
bcs = DirichletBC (V, 0, DomainBoundary())
```

Ako računamo TE modove, odnosno rješavamo po longitudinalnoj komponenti magnetskog polja, koristimo jednadžbu (4.14) te sustav mora zadovoljiti Neumannov rubni uvjet (4.17). Ovaj Neumannov rubni uvijet ne moramo ručno implementirati jer ga razvojno okruženje FEniCS automatski pretpostavi ako ne definiramo nikakav rubni uvjet, stoga bismo drugu liniju jednostavno izostavili u slučaju računanja TE modova.

Kao na primjeru rješavanja Poissonove jednadžbe definiramo probnu i testnu funkciju,

```
u = TrialFunction(V)
v = TestFunction(V)
```

a zatim i slabu formulaciju jednadžbe (4.13), pritom zasad zanemarujemo svojstvenu vrijednost λ.

```
a = inner(grad(u), grad(v))*dx
b = inner(u, v)*dx
```

U nastavku koristimo *eigensolver* alat koji upravo služi rješavanju problema svojstvenih vrijednosti.

```
dummy = v*dx
A = PETScMatrix()
assemble_system(a, dummy, bcs, A_tensor=A)
B = PETScMatrix()
assemble_system(b, dummy, bcs, A_tensor=B)
bcs.zero(B)
```

Matrica A predstavlja lijevu, dok matrica B desnu stranu jednadžbe. Ovim dijelom sastavljamo sustav jednadžbi potrebnih za rješavanje problema svojstvene vrijednosti za električno polje. Kod magnetskog polja, u funkciji `assemble_system` umjesto `bcs` napisali bismo `bcs=None` i izostavili posljednju liniju kako bismo osigurali da se postave Neumannovi rubni uvjeti. Nakon ovako definiranog sustava, možemo iskoristiti *eigensolver*.

```
solver = SLEPcEigenSolver(A, B)
solver.parameters["solver"] = "krylov-schur"
solver.parameters["problem_type"] = "gen_hermitian"
solver.parameters["spectrum"] = "target magnitude"
solver.parameters["spectral_transform"] = "shift-and-invert"
solver.parameters["spectral_shift"] = 2.0
neigs = 12
solver.solve(neigs)
```

Ovime smo odredili parametre *eigensolvera* kao tip solvera, tip problema koji rješavamo, kakva rješenja tražimo i oko koje centralne vrijednosti te koliko rješenja želimo dobiti.

Sada kad smo pokrenuli *eigensolver*, samo trebamo izvući rješenja koja je izračunao.

```
computed_eigenvalues = []
computed_eigenvectors = []
for i in range(min(neigs, solver.get_number_converged())):
r, c, rx, cx = solver.get_eigenpair(i)
computed_eigenvalues.append(r)
computed_eigenvectors.append(rx)
print(computed_eigenvalues)
```

Funkcija `solver.get_number_converged()` nam daje broj dobivenih rješenja solvera, a funkcija `solver.get_eigenpair()` daje realne i kompleksne komponente dobivenih svojstvenih vrijednosti i svojstvenih vektora. Nas zanimaju samo realne komponente rješenja jer u ovom slučaju samo one i postoje, stoga samo njih spremamo. Varijablom `neigs` definirali smo da želimo izračunati 12 mogućih svojstvenih vrijednosti, a postavljanjem `"spectral_shift"` parametra solvera na 2.0 definirali da tražimo svojstvene vrijednosti u okolici vrijednosti 2.0. Posljednja linija nam govori da ćemo rezultat dobiti u obliku liste svojstvenih vrijednosti.

Ako nas zanima i izgled rješenja, odnosno i određeni svojstveni vektor koji pripada nekoj svojstvenoj vrijednosti, recimo prvoj vrijednosti u dobivenoj listi, to dobivamo na sljedeći način:

```
vec = Function(V)
vec.vector()[:] = computed_eigenvectors[0]
c = plot(vec)
plt.colorbar(c)
plt.show()
```

4.2.2. Transverzalne komponente

Osim da tražimo longitudinalnu komponentu magnetskog polja u TE modu, možemo računati transverzalnu komponentu električnog polja. Jednako tako, kod TM modova možemo računati transverzalnu komponentu magnetskog polja. Transverzalne komponente su u ovom slučaju dvodimenzionalni vektori te ih ne možemo predstaviti običnim Lagrangeovim elementima, stoga moramo se okrenuti nekoj drugoj metodi. Jedno rješenje je transverzalnu komponentu rastaviti na dvije, x i y komponentu, te svaku predstaviti jednim Lagrangeovim elementom, no može se pokazati da takva metoda često dovodi do pogrešnih rezultata. Iz tog razloga se koriste Nedelecovi elementi koji puno bolje služe opisivanju dvodimenzionalnih vektora u ovakvim problemima što se može vidjeti u Arnold (2003).

Nedelecove elemente i rubne uvjete koji vrijede za njih kod TE modova definiramo na sljedeći način:

```
V = FunctionSpace(mesh, "N1curl", 1)
bcs = [DirichletBC(V, Constant((0.0, 0.0)),
       DomainBoundary())]
```

Za TM modove, kao i kod longitudinalnih komponenti možemo jednostavno izbrisati rubne uvjete.

S obzirom da se radi o dvodimenzionalnim vektorima, ne možemo više koristiti funkciju `grad` nad njima pa tako ni ne možemo rješavati jednadžbe (4.13) i (4.14), nego se vratimo na oblik (4.10) i njegov magnetski ekvivalent. Takvu jednadžbu, odnosno njenu slabu formulaciju možemo zapisati kao:

```
a = inner(curl(u), curl(v))*dx
b = inner(u, v)*dx
```

S obzirom da se sve ostale linije odnose na postavljanje i pokretanje *eigensolvera* te obradu rješenja, sve ostalo možemo ponovno iskoristiti kao i u prošlom poglavlju te dobiti tražene transverzalne komponente.

Ovako izgleda općenita implementacija problema svojstvene vrijednosti za savršeni metalni valovod.

4.3. Pravokutni valovod

Prvo pogledajmo kakve rezultate bismo trebali očekivati u pravokutnom valovodu. Točni izrazi za sve komponente električnog i magnetskog polja u pravokutnom valovodu se mogu analitički izvesti separacijom varijabli i može se pokazati da svojstvena vrijednost poprima oblik:

$$\lambda = \sqrt{\left(\frac{m\pi}{a}\right)^2 + \left(\frac{n\pi}{b}\right)^2} \tag{4.19}$$

gdje su m i n brojevi koji označavaju određeni mod elektromagnetskog vala, a a i b duljine stranica pravokutnika. Pojednostavnimo rješenja tako da $a = b = \pi$, odnosno:

$$\lambda = \sqrt{m^2 + n^2} \tag{4.20}$$

Prema ovim izrazima, svojstvene vrijednosti koje bismo trebali dobiti iznose 1, 1, 2, 4, 4, 5, 5, 8, 9, 9, 10, 10...
Pogledajmo što dobijemo metodom konačnih elemenata. Kako bismo dobili nekakva rješenja, moramo definirati prostor na kojemu računamo, tj. pravokutni valovod. Stvorimo mrežu kvadratnog oblika sa stranicom duljine π:

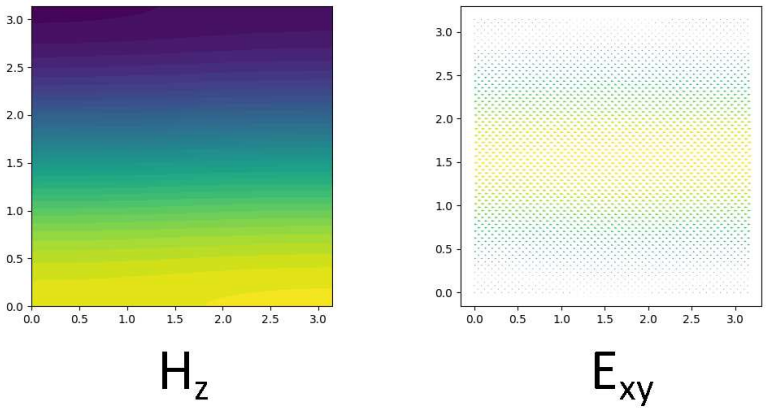
```
mesh = RectangleMesh(Point(0,0), Point(pi, pi), 80, 80,
                     "crossed")
```

Nakon što smo definirali prostor, možemo primjeniti postupke iz prethodnih poglavlja. Izlaz nakon pokretanja čitavog koda za longitudinalnu komponentu magnetskog polja TE modova izgleda ovako:

```
[1.0000856756673306, 1.0000856756675411, 2.000214179284142,
 4.0013708952709575, 4.001370924631881, 5.001628040541056,
 5.001628049102299, 8.003427868933048, 9.00694103446049,
 9.006941072364683, 10.007412734290334, 10.007412782318115]
```

Tu je izlistano 12 svojstvenih vrijednosti koje zadovoljavaju jednadžbu u okolici svojstvene vrijednosti 1.0 kao što je definirano u kodu. Vidimo da su rezultati vrlo slični onima koje dobijemo koristeći (4.19).

Sad nas zanima kako izgleda raspodjela električnih i magnetskih polja u valovodu za modove koje smo dobili. Različite TE modove možemo dobiti mijenjajući centralnu vrijednost *eigensolvera*. Za TE_{10} mod, odnosno mod za koji vijedi $m = 1, n = 0$, dobivamo ovakve rezultate:



Slika 4.1: Rezultati računanja raspodjele longitudinalne komponente magnetskog i transverzalne komponente električnog polja TE_{10} moda u pravokutnom valovodu

Ove rezultate možemo provjeriti iscrtavajući točne matematičke izraze za iste komponente. Da bismo usporedili dobivene rezultate s točnim izrazima, tražiti ćemo samo kvalitativna rješenja, odnosno samo izgled rješenja, a ne i točne vrijednosti polja u svakoj točki pa ćemo u izrazu za polja uzeti samo u obzir dio koji definira oblik rješenja. bez strogo definiranih amplitudi:

$$H_z = \cos(mx)\cos(ny) \tag{4.21}$$

$$E_x = m \cdot \cos(mx)\sin(ny) \tag{4.22}$$

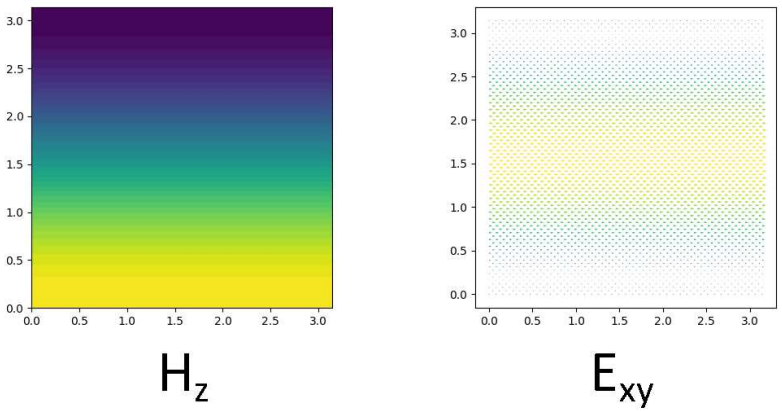
$$E_z = -n \cdot \sin(mx)\cos(ny) \tag{4.23}$$

Primjerice, matematički izraz za longitudinalnu komponentu magnetskog polja TE_{10} moda možemo nacrtati tako da interpoliramo točan matematički izraz za tu komponentu na istu mrežu i funkcijski prostor koji su definirani u zadatku:

```
from dolfin import *
import numpy as np
import matplotlib.pyplot as plt
```

```
mesh = RectangleMesh(Point(0,0), Point(pi, pi), 80, 80,
                     "crossed")
V = FunctionSpace(mesh, 'Lagrange', 1)

E = Expression(('cos(m*x[0])*cos(n*x[1])'),
               degree=2, m=1, n=0)

u = interpolate(E, V)

plot(u)
plt.show()
```

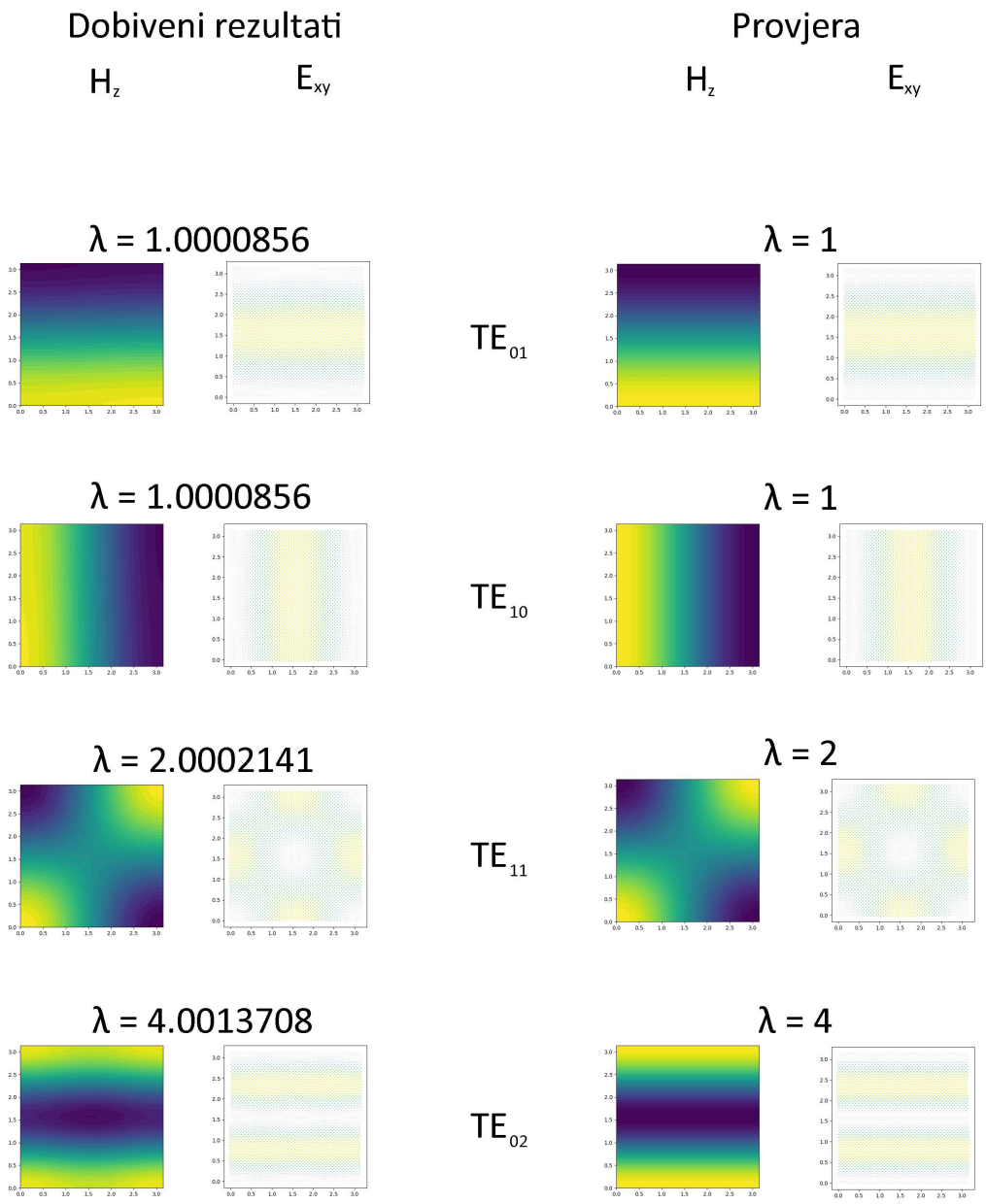
Isto tako možemo nacrtati i transverzalnu komponentu. Rezultati koje dobijemo:



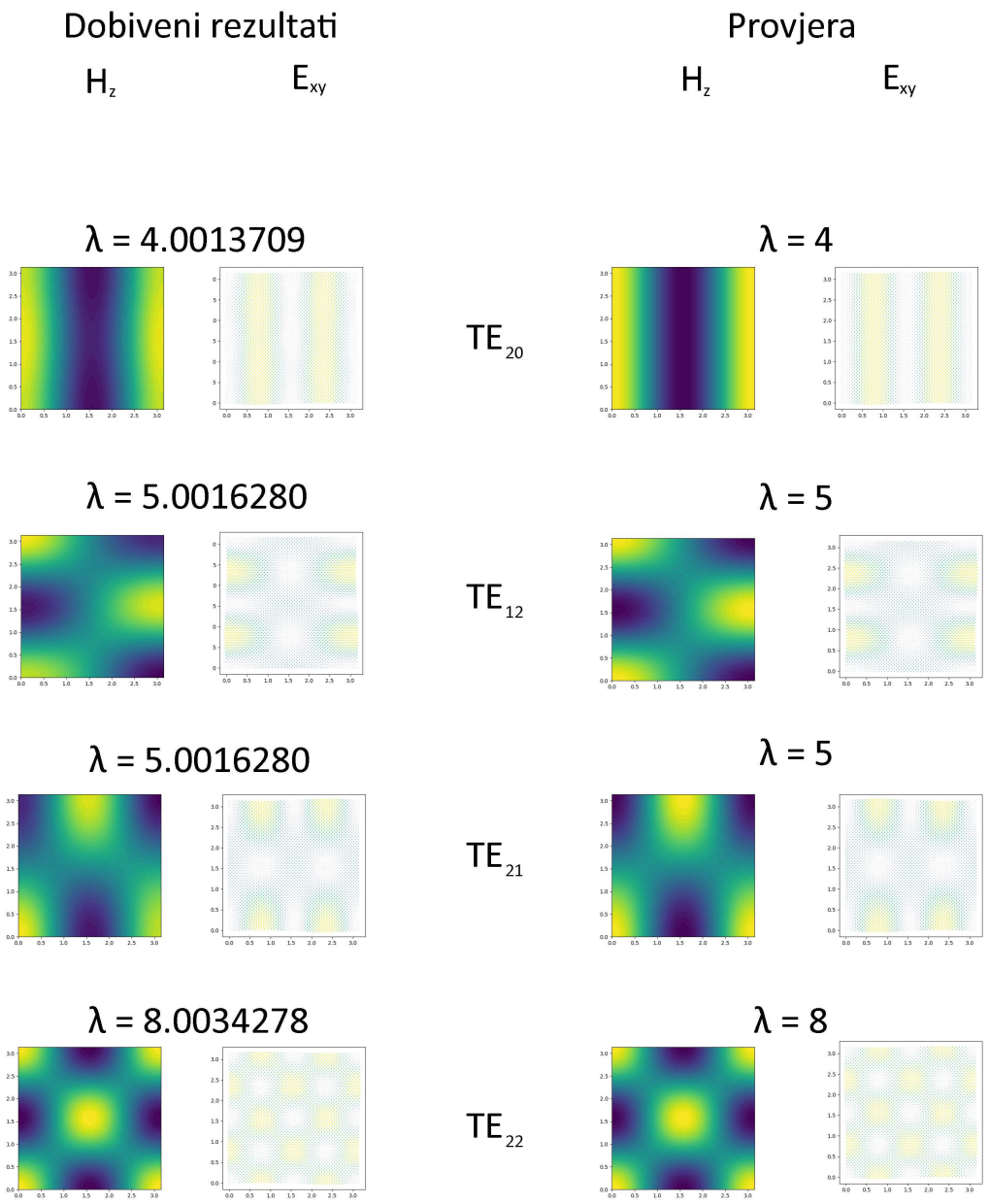
Slika 4.2: Očekivane raspodjele longitudinalne komponente magnetskog i transverzalne komponente električnog polja TE_{10} moda u pravokutnom valovodu

Vidimo da su očekivani rezultati gotovo identični dobivenima.

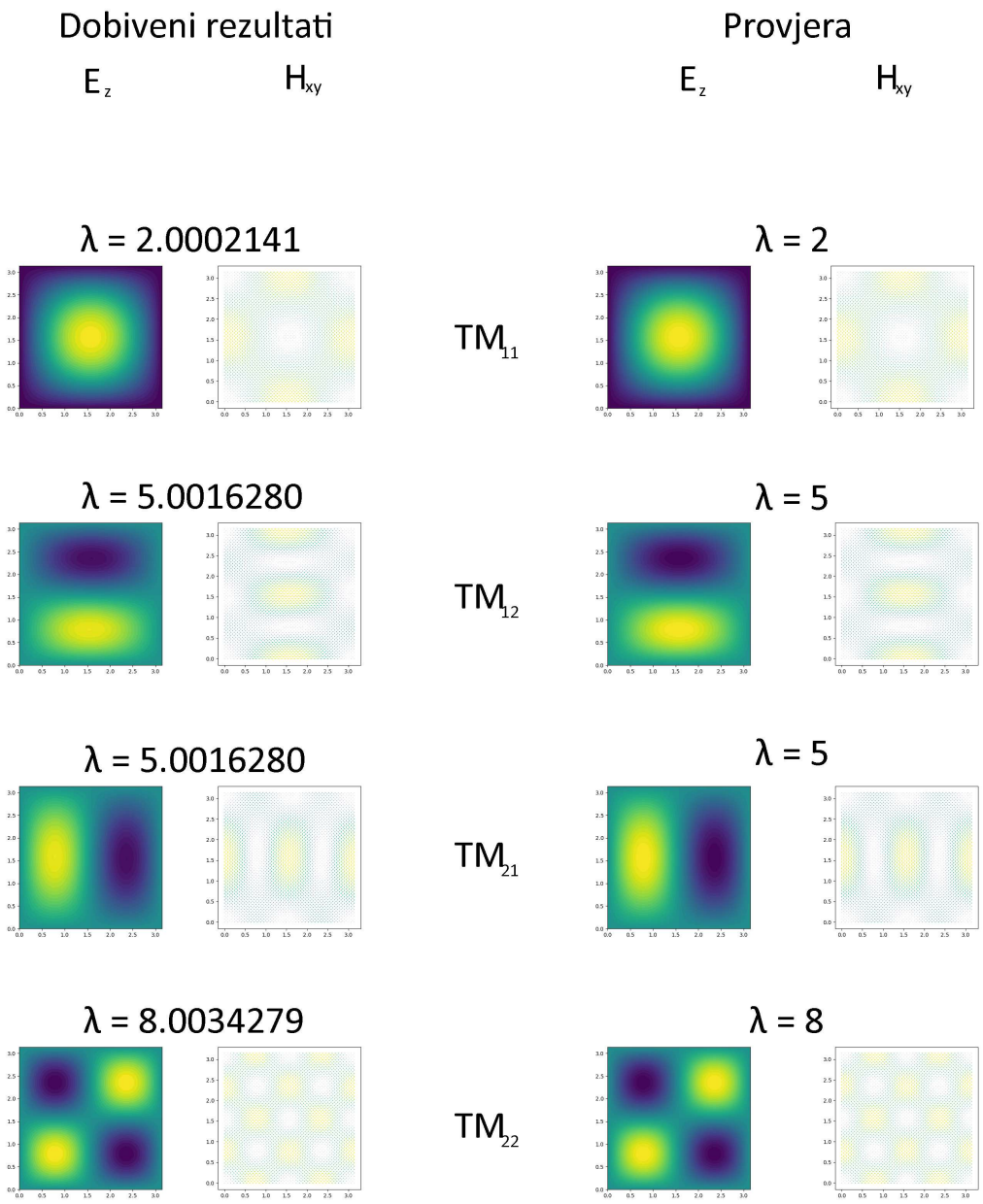
Isti proces možemo obaviti i za ostale TE i TM modove. Prvih 8 TE i TM modova prikazani su na slikama:



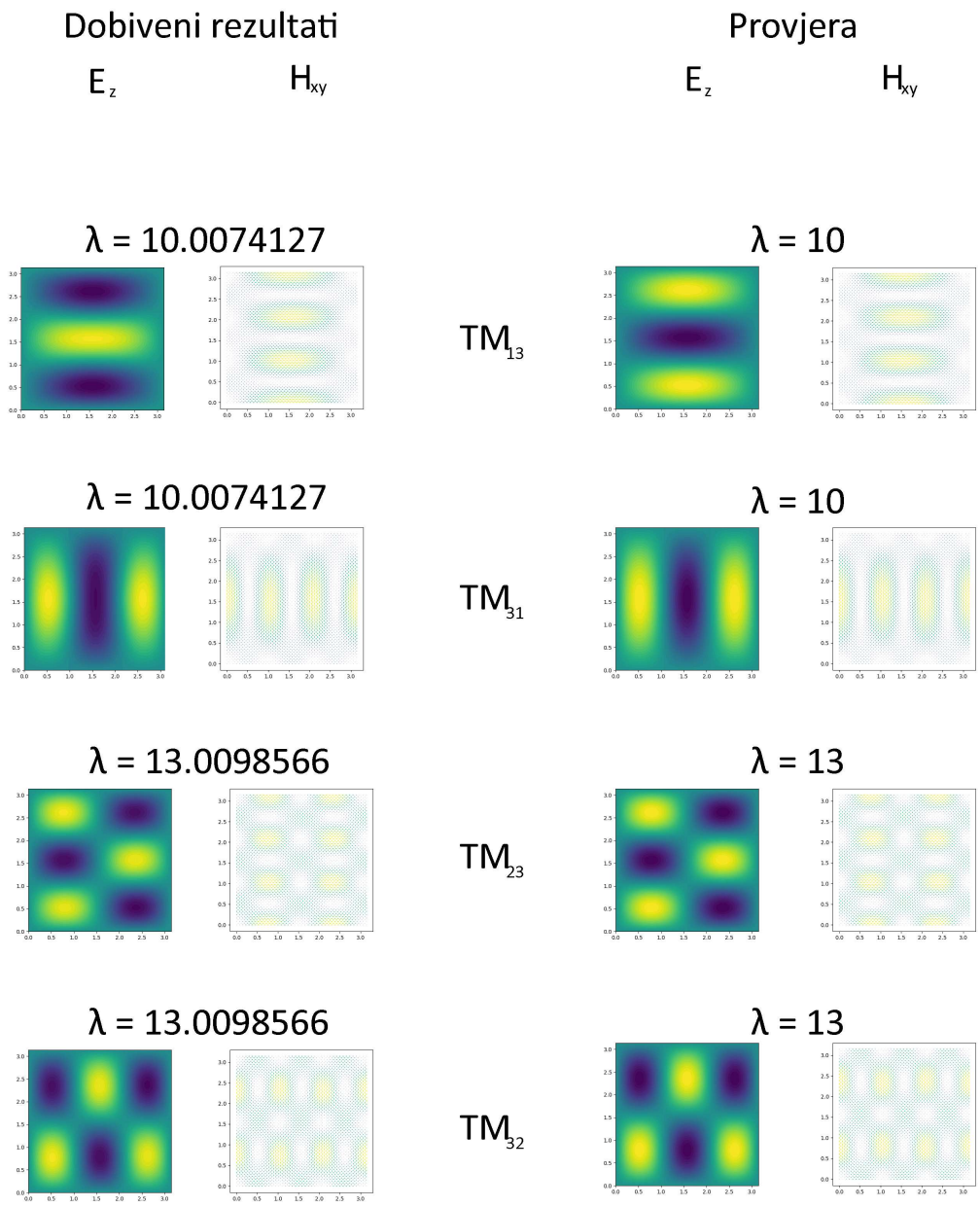
Slika 4.3: Raspodjela longitudinalne komponente magnetskog i transverzalne komponente električnog polja nekih TE modova pravokutnog valovoda.



Slika 4.4: Raspodjela longitudinalne komponente magnetskog i transverzalne komponente električnog polja nekih TE modova pravokutnog valovoda.



Slika 4.5: Raspodjela longitudinalne komponente električnog i transverzalne komponente magnetskog polja nekih TM modova pravokutnog valovoda.



Slika 4.6: Raspodjela longitudinalne komponente električnog i transverzalne komponente magnetskog polja nekih TM modova pravokutnog valovoda.

4.4. Cilindrični valovod

Za cilindrični valovod provest ćemo isti postupak kao i za pravokutni osim što ćemo promijeniti oblik valovoda. Definirat ćemo mrežu kružnog oblika radijusa $r = 0.5$.

```
base = Circle(Point(0.5, 0.5), 0.5)
mesh = generate_mesh(base, 40)
```

Izrazi za modove cilindričnog valovoda nisu jednostavni kao za pravokutni jer su rezultati opisani Besselovim funkcijama. No svojstvene vrijednosti dane su izrazima:

$$\lambda = \left(\frac{p_{\text{vn}}}{r}\right), \quad \text{za TM modove} \tag{4.24}$$

$$\lambda = \left(\frac{p'_{\text{vn}}}{r}\right), \quad \text{za TE modove} \tag{4.25}$$

gdje p_{vn} označava n-tu nulu Besselove funkcije v-tog stupnja, a p'_{vn} n-tu nulu prve derivacije Besselove funkcije v-tog stupnja. Prvi izraz nam daje da su očekivane svojstvene vrijednosti za TM modove 13.56, 37.33, 58.75, 70.64, 113.18...

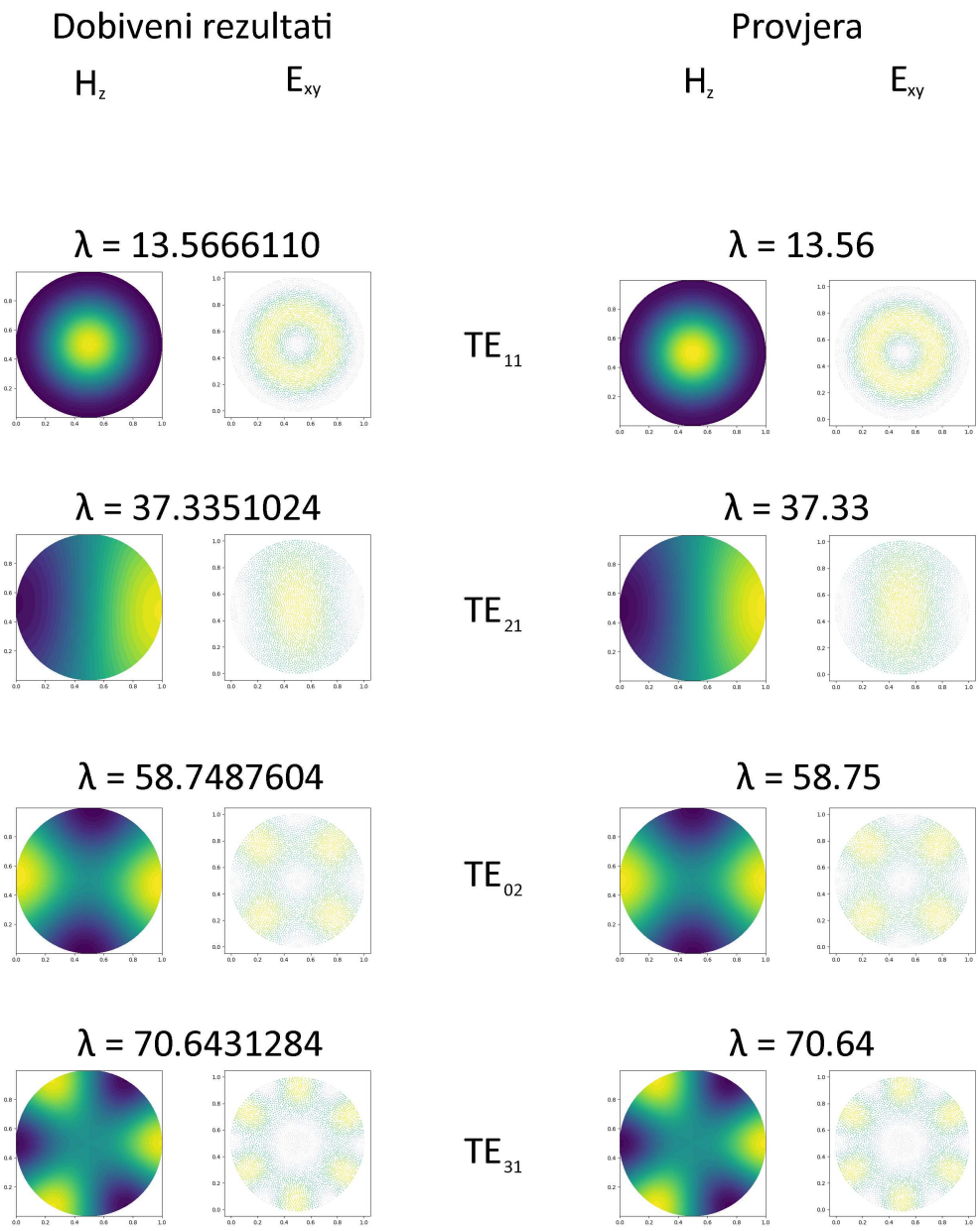
Program nam daje rezultate:

```
[13.566611073661946, 13.566621892172883, 37.334472058389984,
 37.33510249422398, 58.74876042934757, 70.64312849428886,
 70.64423843790433, 113.17787823994261]
```

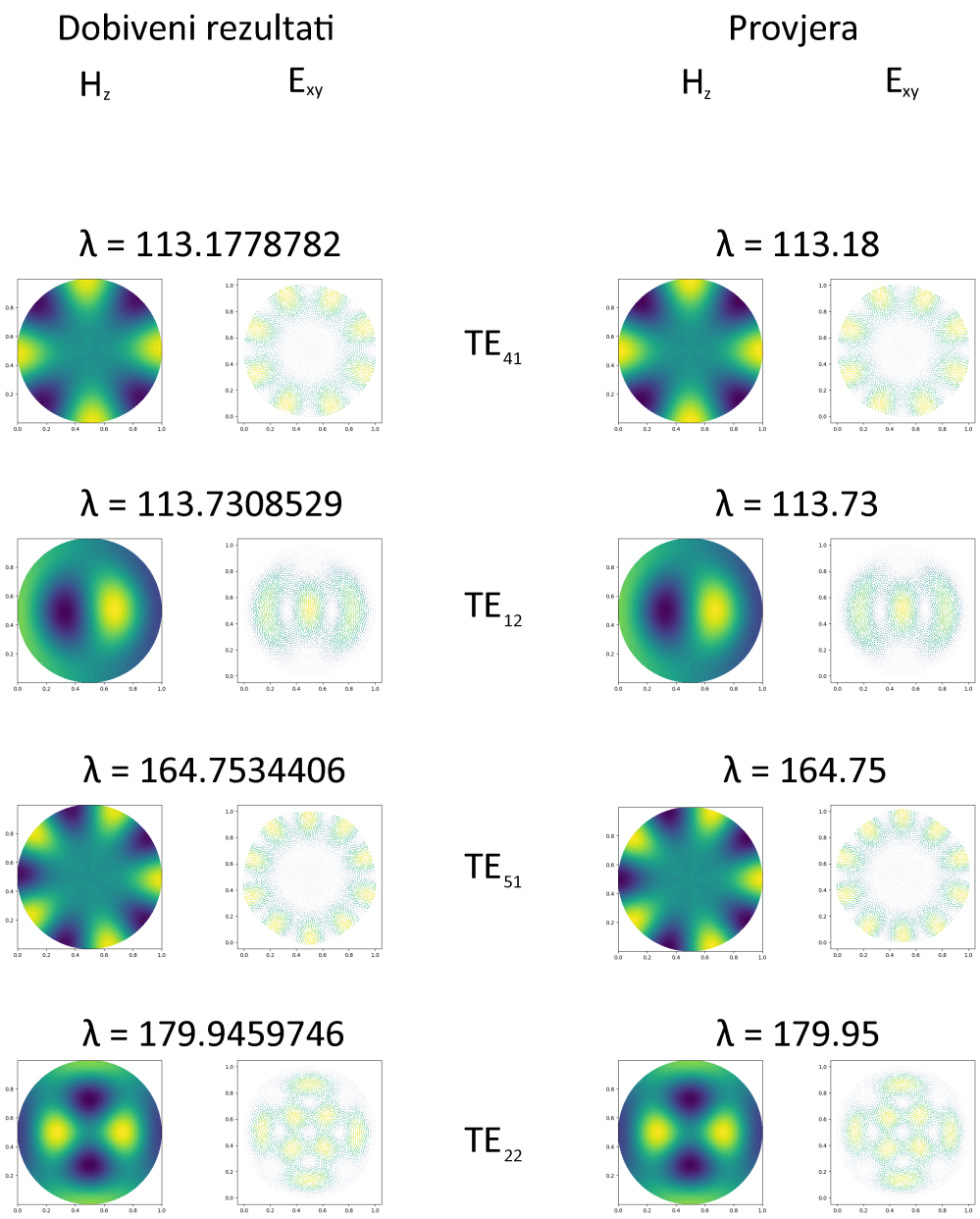
Neke vrijednosti se ponavljaju dva puta zbog činjenice da ti modovi mogu imati bilo koju rotaciju pa FEniCS pronalazi dvije nasumične rotacije tog moda pomoću kojih je moguće opisati i sve ostale. Unatoč tome, možemo vidjeti da su rezultati približno jednaki očekivanima

Kao i za pravokutni valovod, iscrtat ćemo modove i usporediti ih s točnim izrazima za sve komponente. Implementacija izraza za komponente električnog i magnetskog polja kod cilindričnog valovoda je kompliciranija od one za pravokutni jer nije jednostavno implementirati izraze za Besselove funkcije i njene derivacije kao sinusnu i kosinusnu funkciju, ali se kao i kod pravokutnog valovoda svodi na interpolaciju tih izraza na istu mrežu i funkcijski prostor kao i u zadatku. Stoga nećemo detaljno prolaziti kroz implementaciju koda za provjeru.

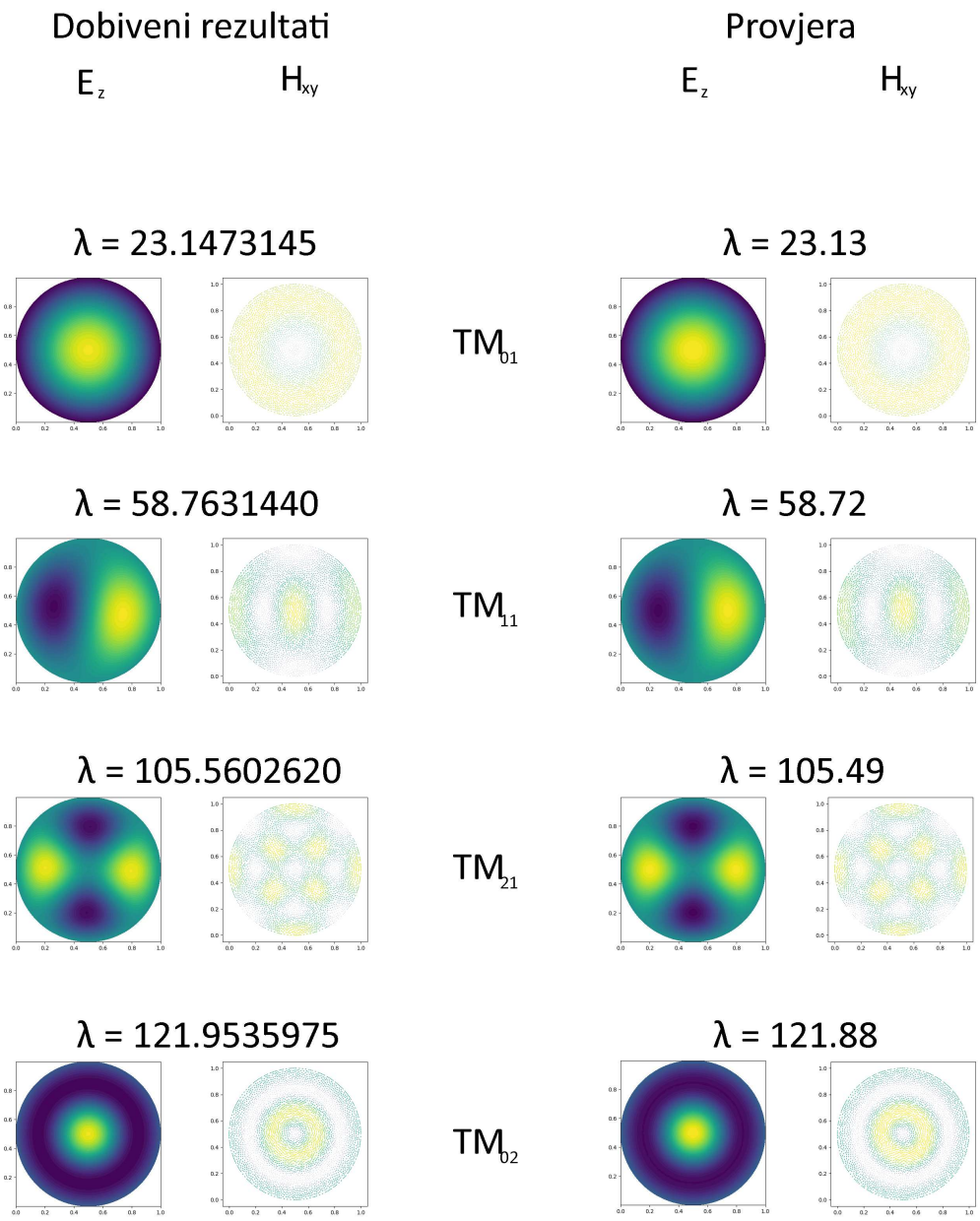
Rezultati za prvih 8 TE i TM modova i usporedba s izrazima za provjeru prikazani su na slikama:



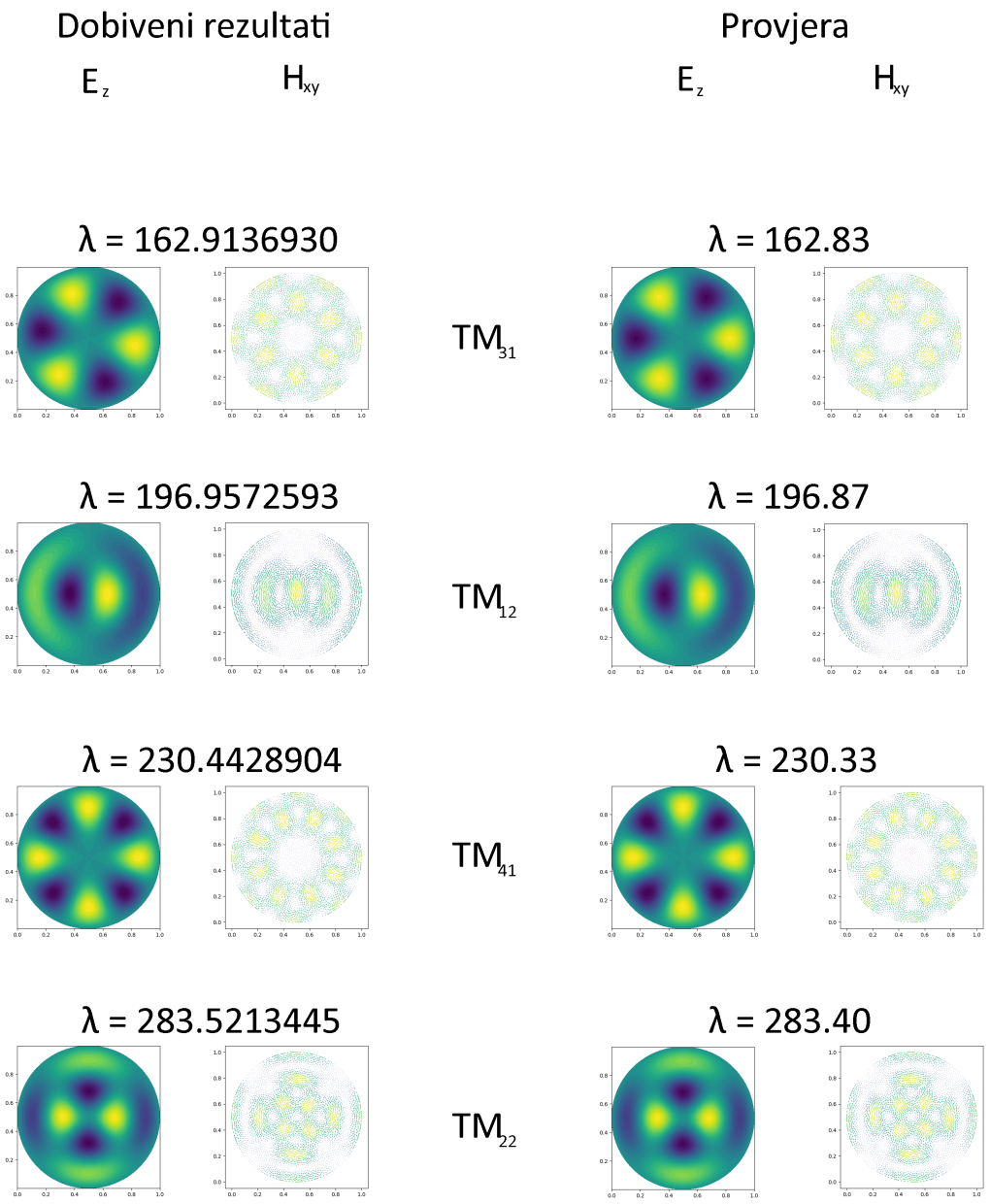
Slika 4.7: Raspodjela longitudinalne komponente magnetskog i transverzalne komponente električnog polja nekih TE modova cilindričnog valovoda.



Slika 4.8: Raspodjela longitudinalne komponente magnetskog i transverzalne komponente električnog polja nekih TE modova cilindričnog valovoda.



Slika 4.9: Raspodjela longitudinalne komponente električnog i transverzalne komponente magnetskog polja nekih TM modova cilindričnog valovoda.



Slika 4.10: Raspodjela longitudinalne komponente električnog i transverzalne komponente magnetskog polja nekih TM modova cilindričnog valovoda.

5. Interferencija elektromagnetskih valova u vremenu

U ovom dijelu ćemo prikazati kako se razvojno okruženje FEniCS može iskoristiti za rješavanje sustava koji su promjenjivi u prostoru, ali i vremenu. Kao primjer ćemo uzeti dva točkasta izvora elektromagnetskih valova te pokušati simulirati nastanak interferencijskih uzoraka tih valova u prostoru i vremenu.

5.1. Vremenska diskretizacija

Kako bismo mogli simulirati kako se sustav mijenja u vremenu, moramo ga podijeliti na konačan broj koraka u vremenu, stoga vratimo se ponovno na diferencijalni oblik Maxwellovih jednadžbi u vremenskoj domeni:

$$\nabla \cdot E = \frac{\rho}{\mathcal{E}} \tag{5.1}$$

$$\nabla \cdot B = 0 \tag{5.2}$$

$$\nabla \times E = -\frac{\partial B}{\partial t} \tag{5.3}$$

$$\nabla \times B = \mu J + \mu\epsilon\frac{\partial E}{\partial t} \tag{5.4}$$

Djelovanjem operatora rotacije na (5.3) te uvrštavanjem (5.4) u dobiveni izraz, dobivamo:

$$\nabla \times \nabla \times E = -\mu\frac{\partial J}{\partial t} - \mu\epsilon\frac{\partial^2 E}{\partial t^2} \tag{5.5}$$

Korištenjem relacije (4.11) i uvrštavanjem (5.1) u dobiveni izraz dobivamo jednadžbu:

$$\frac{\nabla \rho}{\epsilon} - \triangle E = -\mu \frac{\partial J}{\partial t} - \mu\epsilon \frac{\partial^2 E}{\partial t^2} \tag{5.6}$$

Definirajmo pobudu $f(t)$ kao

$$f(t) = \frac{\nabla \rho}{\epsilon} + \mu \frac{\partial J}{\partial t} \tag{5.7}$$

te zbog činjenice da je ovo valna jednadžba, zamjenimo $\mu\epsilon = \frac{1}{c^2}$, gdje je c brzina vala. Tako dobivamo konačnu jednadžbu:

$$f(t) - \triangle E = -\frac{1}{c^2} \frac{\partial^2 E}{\partial t^2} \tag{5.8}$$

Sad trebamo provesti vremensku diskretizaciju. Neka je $\triangle t$ širina jednog vremenskog koraka. S u^{n} označimo vrijednost nekakve funkcije u u n-tom koraku. Derivacija funkcije u bi onda glasila:

$$\frac{\partial u}{\partial t} = \frac{u^{\mathrm{n}} - u^{\mathrm{n\text{-}1}}}{\triangle t} \tag{5.9}$$

Odnosno, druga derivacija bi glasila:

$$\frac{\partial^2 u}{\partial t^2} = \frac{\frac{u^{\mathrm{n}} - u^{\mathrm{n\text{-}1}}}{\triangle t} - \frac{u^{\mathrm{n\text{-}1}} - u^{\mathrm{n\text{-}2}}}{\triangle t}}{\triangle t} = \frac{u^{\mathrm{n}} - 2u^{\mathrm{n\text{-}1}} + u^{\mathrm{n\text{-}2}}}{\triangle t^2} \tag{5.10}$$

Uvrštavanjem dobivenog u (5.8) i sređivanjem, dobivamo:

$$E^{\mathrm{n}} + c^2 \triangle t^2 \triangle E^{\mathrm{n}} = 2E^{\mathrm{n\text{-}1}} + E^{\mathrm{n\text{-}2}} + c^2 \triangle t^2 f(t) \tag{5.11}$$

Vidimo da vrijednost električnog polja u trenutku n ovisi o vrijednosti električnog polja u trenutcima $n-1$ i $n-2$ te pobudi $f(t)$. Zapišemo li ovu jednadžbu u slaboj formulaciji, dobivamo

$$\int_\Omega (uv + c^2 \triangle t^2 \nabla u \nabla v) dx = \int_\Omega (2u^{\mathrm{n\text{-}1}} - u^{\mathrm{n\text{-}2}} + c^2 \triangle t^2 f(t)) dx, \tag{5.12}$$

gdje je u vrijednost električnog polja u n-tom koraku.

5.2. Simulacija interferencije

Sada možemo porvesti programsku implementaciju ovog problema. Kao i uvijek započinjemo s

```
from dolfin import *
```

Zatim definiramo koliko želimo da simulacija traje i na koliko vremenskih koraka taj period želimo podijeliti, a time i duljinu trajanja svakog koraka, te brzinu elektromagnetskog vala:

```
T = 100.0
num_steps = 1000
dt = T/num_steps
c = 1
```

Duljinu simulacije smo postavili na 100 sekundi, s 1000 koraka te smo zbog jednostavnosti postavili da je brzina svjetlosti $c = 1$.

Nakon toga definiramo funkcijski prostor, pravokutnu mrežu i rubne uvjete:

```
mesh = RectangleMesh(Point(-20, -40), Point(20, 40), 160, 160)
V = FunctionSpace(mesh, 'Lagrange', 1)

def boundary(x, on_boundary):
    return on_boundary

bc = DirichletBC(V, 0, boundary)
```

Vidjeli smo da trenutna vrijednost električnog polja, odnosno vrijednost električnog polja u koraku n ovisi o vrijednosti u koracima $n - 1$ i $n - 2$, stoga da bismo mogli izračunati vrijednost električnog polja na početku simulacije, moramo imati početne uvjete u^0 i u^1. Želimo da sustav na početku bude u stacionarnom stanju pa postavimo da su početni uvjeti jednaki nula:

```
u_0 = Constant(0)
u_1 = Constant(0)
u_n1 = interpolate(u_0, V)
u_n2 = interpolate(u_1, V)
```

Sad definirajmo probnu i testnu funkciju, postavimo slabu formulaciju te sastavimo sustav jednadžbi koje treba riješiti. Pritom, zasad zanemarujemo funkciju pobude $f(t)$ te ćemo ju naknadno dodati kao u slučaju rješavanja Poissonove jednadžbe za točkasti naboj:

```
u = TrialFunction(V)
v = TestFunction(V)

a = u*v*dx + c*c*dt*dt*inner(grad(u), grad(v))*dx
L = (2*u_n1 - u_n2)*v*dx

A, b = assemble_system(a, L, bc)
```

Prije nego li započnemo računanje, moramo definirati gdje želimo spremati rezultate. Dobivene vrijednosti za svaki korak u simulaciji spremati ćemo u datoteku $interference.pvd$ koju možemo kasnije otvoriti u programu poput $Paraview$ koji omogućuje prikaz niza slika u slijedu u obliku videa:

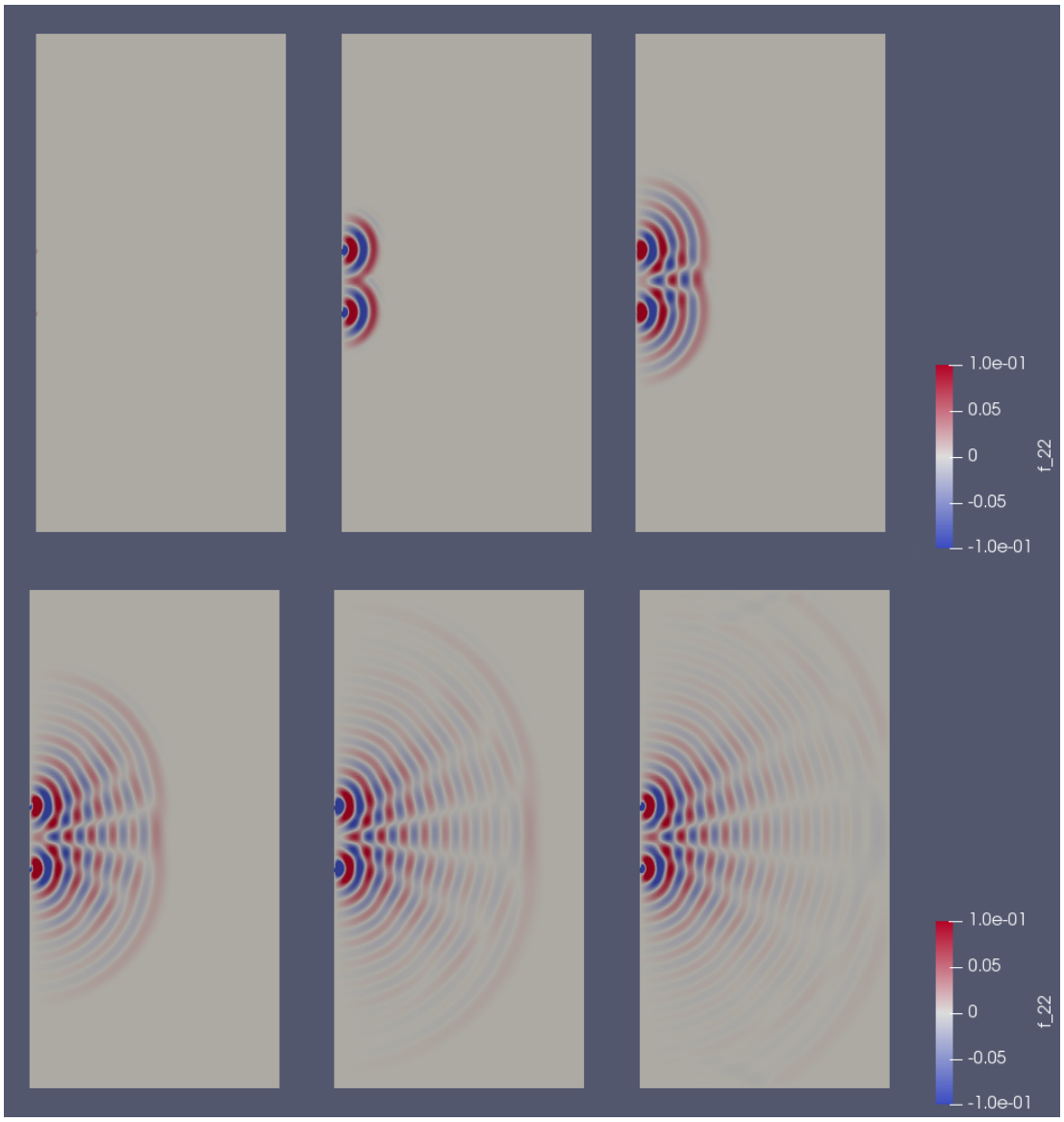
```
vtkfile = File('interference.pvd')
```

Sad pokrećemo simulaciju i istovremeno ubacujemo točkaste izvore u sustav:

```
u = Function(V)
t = 0
for n in range(num_steps):
    t += dt

    A, b = assemble_system(a, L, bc)
    delta1 = PointSource(V, Point(-20, 5), sin(t))
    delta2 = PointSource(V, Point(-20, -5), sin(t))
    delta1.apply(b)
    delta2.apply(b)

    solve(A, u.vector(), b)

    vtkfile << (u,t)
    u_n2.assign(u_n1)
    u_n1.assign(u)
```

Ovim blokom koda rješavamo sustav jednadžbu za svaki vremenski korak. U svakom koraku sastavljamo sustav jednadžbi iznova s obzirom da se sa svakim korakom sustav mijenja jer se mijenjaju vrijednosti u^{n-1} i u^{n-2}.

Također u svakom koraku implementiramo točkaste izvore sinusoidalnog oblika nazvane $delta1$ i $delta2$ koji su razmaknuti za 10 jedinica jedan od drugog. nakon toga rješavamo sustav, spremamo rješenje za taj korak u datoteku koju smo odredili te mijenjamo vrijednosti u^{n-1} i u^{n-2}.
Nakon izvršavanja programa i pokretanja datoteke u programu $Paraview$ dobivamo simulaciju prikazanu na slikama.



Slika 5.1: Raspodjela električnog polja u različitim trenutcima u vremenu.

Očito je vidljiv nastanak interferencijskih maksimuma i minimuma koje očekujemo u ovakvom sustavu.

6. Zaključak

Metoda konačnih elemenata je iznimno koristan postupak pri rješavanju parcijalnih diferencijalnih jednadžbi te smo prikazali njenu matematičku pozadinu, ali i primjenu u FEniCS razvojnom okruženju na primjerima iz elektromagnetizma. Pokazali smo točnost dobivenih rezultata pri računanju svojstvenih vrijednosti elektromagnetskih modova u pravokutnom i cilindričnom valovodu te sposobnost opisivanja sustava koji se mijenjaju u vremenu na primjeru interferencije elektromagnetskih valova stvorenih dvama točkastim izvorima.

Ovi primjeri su geometrijski jednostavni te se daju riješiti i analitčki, a izabrani su upravo zato da pokažemo mogućnosti primjene FEM-a u FEniCS-u, no rezultati koje smo dobili pokazuju da se ovaj alat može primjeniti i biti itekako koristan za rješavanje kompliciranijih sustava koje inače ne bismo mogli analitički riješiti.

FEniCS se pokazao primjenjiv na različitim tipovima zadataka u elektromagnetizmu, ali svoju primjenu može pronaći i u drugim sustavima koji opisuju fizikalne pojave te je stoga vrlo moćan alat koji je svim inženjerima na raspolaganju.

LITERATURA

Introduction to Rectangular Waveguides... URL http://kilyos.ee.bilkent.edu.tr/~microwave/programs/magnetic/rect/info.htm.

Circular Waveguide, 2003. URL https://www.nhn.ou.edu/~johnson/Education/Juniorlab/Microwave/CylindricalWaveguide.pdf.

Douglas N. Arnold. *Differential complexes and numerical stability*. 2003. URL http://www-users.math.umn.edu/~arnold/papers/icm2002.pdf.

Hans Petter Langtangen i Anders Logg. *Solving PDEs in Python – The FEniCS Tutorial Volume I*, 2017. URL https://fenicsproject.org/pub/tutorial/pdf/fenics-tutorial-vol1.pdf.

Mats G. Larson i Fredrik Bengzon. *The Finite Element Method: Theory, Implementation, and Practice*. 2010. URL http://matematicas.unex.es/~coco/Modelizacion/FEMLarson_Bengzon.pdf.

Implementacija metode konačnih elemenata u razvojnom okruženju FEniCS

Sažetak

Metoda konačnih elemenata je numerička metoda za rješavanje parcijalnih diferencijalnih jednadžbi s varijabilnim koeficijentima na nesimetričnim geometrijama. Odabir numeričke metode ovisi o primjenama, vrsti jednadžbe te željenim svojstvima numeričke metode. U radu je opisana teorijska pozadina slabe formulacije eliptičke diferencijalne jednadžbe te objašnjen prostor konačnih elemenata za njenu diskretizaciju. Primijenjena je metoda na različitim rubnim uvjetima i desnim stranama jednadžbe. Opisane su osnovne metode diskretizacije konačnim razlikama u vremenu te primijenjena prostorno vremenska diskretizacija na jednadžbi provođenja. Implementirane su navedene metode u razvojnom okruženju FEniCS.

Ključne riječi: Metoda konačnih elemenata, FEniCS, elektromagnetizam

Finite element method implementation using FEniCS computing platform

Abstract

Finite element method is a numerical method for solving partial differential equations with variable coefficients on asymmetrical geometries. The choice of the numerical method depends on its applications, types of equations and wanted properties of the numerical method. The thesis describes the theoretical background of the weak formulation of an elliptical differential equation and explains the space of finite elements for its discretization. The method is applied on different boundary conditions and right-hand sides of equations. Basic methods of discretizations with finite differences in time are explained and time-space discretization is applied to the conducting equation. The mentioned methods are implemented in the FEniCS platform.

Keywords: Finite element method, FEniCS, electromagnetism