SOLUSI PERSAMAAN YUKAWA DI DAERAH SEDERHANA MENGGUNAKAN METODE GALERKIN DALAM MATLAB

K. DAHLAN[1], A. D. GARNADI[2,5], M. ILYAS[2], E. H. NUGRAHANI[2], Y. S. PUTRA[3], E. YULIANY[4], L. YULIAWATI[5]

Abstrak

Tulisan ini, merupakan sebuah tutorial bagaimana mengimplementasikan metode Galerkin untuk menyelesaikan persamaan Yukawa. Persamaan ini, misalnya digunakan untuk memodelkan perambatan air dalam keadaan tak jenuh. Misalnya untuk memperoleh informasi bentuk pembasahan akibat adanya sumber air jenuh, persamaan Yukawa perlu diselesaikan secara numerik. Persamaan gelombang skalar untuk background homogen digunakan untuk memperkenalkan FEM. Untuk lebih sederhananya, digunakan elemen segitiga orde pertama. Makalah ini menunjukkan bagaimana pengetahuan tentang metode elemen hingga (FEM) dapat dipelajari dalam waktu singkat dengan menggunakan MATLAB. Hal ini menunjukkan bagaimana pengetahuan yang diperoleh dapat diperluas untuk masalah bentuk serupa lainnya.

1 PENDAHULUAN

Ketersediaan daya komputasi yang besar dan murah menggunakan komputer desktop atau laptop menjadikan solusi numerik dari permasalahan elektromagnetik menjadi hal yang layak, bahkan bagi mahasiswa sarjana sekalipun. Di kalangan pendidik telah diambil dua pendekatan: menggunakan perangkat lunak yang tersedia secara komersial [1] (yang mungkin menjadi pilihan yang mahal), atau desain antarmuka pengguna dan kode simulasi ([2, 3]) berdasarkan paket matematis terprogram. Kedua pendekatan ini bukan merupakan obyek dari tulisan ini. Tujuan dari tulisan ini adalah memperkenalkan metode momen melalui MATLAB dan menyelesaikan permasalahan elektromagnetik. MATLAB telah digunakan di seluruh dunia dalam pengajaran banyak mata kuliah rekayasa, misalnya, pemrosesan sinyal dan teknik kontrol. Ini tidak akan mudah bagi para pengajar dalam bidang elektromagnetik untuk mengharapkan siswa untuk memiliki pengetahuan dan akses menggunakan MATLAB. Metode Elemen Hingga (FEM) adalah teknik yang relatif mapan dalam elektromagnetik dan masih merupakan area penelitian yang cukup aktif. Hal ini didukung oleh beberapa buku teks dan

[1]Departemen Fisika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Jalan Meranti Kampus IPB Dramaga Bogor, 16680.
[2]Departemen Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Jalan Meranti Kampus IPB Dramaga Bogor, 16680.
[3]Departemen Fisika, Universitas Tanjung Pura, Pontianak.
[4]Departemen Matematika, Universitas Tanjung Pura, Pontianak.
[5]Research Cluster for Dynamics and Complex System, Fakultas Matematika dan Ilmu Pengetahuan Alam IPB.

monograf yang cukup baik tersedia. Tetapi, bahan tersebut hanya cocok bagi para peneliti atau untuk perkuliahan khusus. Selain itu, buku teks tersebut sangat menekankan dasar matematika yang ketat dari berbagai formulasi berbagai FEM. Sebuah pendekatan praktis untuk penurunan dari FEM dan pelaksanaannya diberikan pada pustaka [6]. Pada tulisan ini diberikan uraian FEM yang singkat dan mudah dipahami melalui MATLAB, mengingat MATLAB menyediakan fasilitas untuk operasi matriks dan alokasi memori yang dinamis. Materi dalam tulisan ini dapat dibahas dalam kuliah selama dua jam. Mahasiswa mungkin diberikan kesempatan selama dua minggu untuk menyerap materi dan mengerjakan tugas yang serupa dengan contoh yang diberikan dalam tulisan ini. Perlu dicatat bahwa setidaknya satu paket perangkat lunak FEM komersial, yaitu FEMLAB dari COMSOL Inc, [7] memiliki versi sebagai add-on untuk MATLAB. Paket ini tidak cocok untuk mengajar pemrograman FEM. Namun, seperti yang digambarkan dalam rujukan [8], mahasiswa dapat menggunakan software ini untuk menguji formulasi variasional yang diperoleh oleh mereka, mengingat perangkat lunak menerima bentuk formulasi variasional sebagai input. Dengan demikian, [8] dan tulisan ini saling melengkapi jika FEMLAB tersedia.

2 PERSAMAAN MODIFIED HELMHOLTZ DAN FORMULASI FEM

Persamaan modified Helmholtz, atau dikenal sebagai persamaan Yukawa, muncul akibat dari teori potensial Newton. Yukawa menjelaskan bahwa teori potensial Newton akan mengakibatkan energi potensial memenuhi persamaan Yukawa [14]. Persamaan ini juga analog dengan persamaan yang muncul pada persamaan air tak jenuh dan media porous [10, 12, 11]. Di balik analogi tersebut, persamaan ini juga memberikan usaha pengembangan dan pemahaman baru terhadap ilmu hidrologi. Persamaan ini akan ditinjau lebih jauh lagi dengan pendekatan FEM pada artikel ini.

Untuk mempelajari penggunaan FEM melalui MATLAB, pada pembahasan lebih lanjut digunakan persamaan Yukawa berikut

$$-\nabla^2 u - ku = 0, \tag{1}$$

dengan $k < 0$, $k = -l^2$ dan u sebagai variabel medan listrik atau medan magnet.

FEM memecahkan persamaan (1) dengan meminimumkan fungsional bentuk lemah yang diberikan oleh:

$$F(u,u) = \int_\Omega (\nabla u \cdot \nabla u + kuu)dx. \tag{2}$$

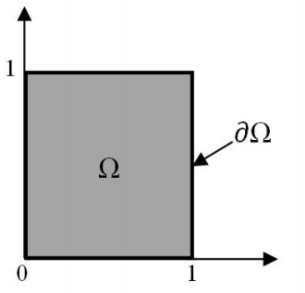
Kasus 1

Perhatikan persamaan Yukawa dengan syarat batas Dirichlet berikut

$$\begin{aligned} -\nabla^2 u - ku &= 0 \quad \text{di } \Omega \\ u &= g \quad \text{di } \partial\Omega \end{aligned} \tag{3}$$

di mana k dan g adalah sembarang fungsi yang diketahui. Hal ini dapat diilustrasikan untuk gelombang datar yang melewati medium kedua yang merupakan penghantar sempurna.

Misalkan $\Omega = [0, 1] \times [0, 1]$ dan syarat batas $\partial\Omega$ yang diilustrasikan pada Gambar 1.



Gambar 1 Daerah Ω

Kasus 2

Hal lain yang mungkin jika gelombang datar tersebut mengalami perubahan setelah melewati medium kedua. Perhatikan persamaan Yukawa dengan syarat batas Dirichlet dan Neumann berikut

$$\begin{aligned} -\nabla \cdot \nabla u - ku &= 0 \quad \text{di } \Omega \\ u &= p \quad \text{di } L_1 \\ n \cdot \nabla u &= q \quad \text{di } L_2 \end{aligned} \tag{4}$$

di mana k, p dan q adalah fungsi-fungsi yang diketahui. Misalkan $\Omega = [0, 1] \times [0, 1]$ dengan syarat batas Dirichlet di L_1 dan syarat batas Neumann di L_2.

3 HAMPIRAN GALERKIN

Untuk menyelesaikan kasus-kasus yang sudah diuraikan di atas secara numerik digunakan FEM dengan hampiran Galerkin. Simulasi untuk kedua kasus ini akan diuraikan dengan menggunakan MATLAB.

Kasus 1

Misal $v \in H^1$ di mana H^1 adalah ruang Hilbert $H^1 = \{v : \|v\| + \|\nabla v\| < \infty\}$, sehingga bentuk lemah kasus 1 adalah

$$\int_{\Omega} (\nabla u \cdot \nabla v - kuv) dx = 0. \tag{5}$$

Misal $V_h \in H^1$ adalah ruang dari semua fungsi *piecewise* linear yang kontinu dalam sebuah partisi $K = \{T\}$ dari S menjadi segitiga berukuran $|T|$.Himpunan hat function didefinisikan $\varphi_{i=1}^{N}$ di mana N menyatakan banyaknya node dalam triangulasi T yang merupakan basis untuk V_h. Pendekatan elemen hingga (*finite element approximation*) dari bentuk lemah (*weak form*) persamaan (3) adalah $u_h \in V_h$ sedemikian sehingga

$$\int_{\Omega} (\nabla u_h \cdot \nabla v - k u_h v) dx = 0. \tag{6}$$

Karena $u_h \in V_h$ maka u_h dapat ditulis

$$u_h = \sum_{j=1}^{N} z_j \varphi_j.$$

Oleh karena itu persamaan (6) dapat ditulis

$$\sum_{j=1}^{N} z_j \left(\int_{\Omega} \nabla \varphi_j \cdot \nabla \varphi_i \, dx - \int_{\Omega} k \varphi_j \varphi_i \, dx \right) = 0. \tag{7}$$

Dari persamaan (7) diperoleh sistem persamaan linear $Az = b$, di mana elemen-elemen sistem persamaan linear tersebut sebagai berikut

$$A_{i,j} = \int_{\Omega} \nabla \varphi_j \cdot \nabla \varphi_i \, dx - \int_{\Omega} k \varphi_j \varphi_i \, dx \tag{8}$$

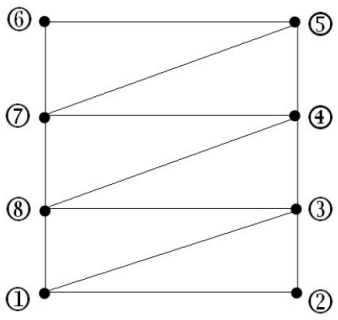
$$b_i = 0 \tag{9}$$

$$z = (z_1, z_2, ..., z_N)^T. \tag{10}$$

Pada persamaan di atas j bergerak sepanjang node dan i bergerak sepanjang node dengan nilai u diketahui. Nilai u pada $\partial\Omega$ yang diketahui dinotasikan dalam vektor z_e dan nilai u selainnya dinotasikan dalam z_n,

$$(A_e | A_n) \begin{pmatrix} z_e \\ z_n \end{pmatrix} = A_e z_e + A_n z_n = b,$$

matriks A_e dihitung pada syarat batas. Sehingga solusi akhir z_n dapat dihitung dengan sistem persamaan $A_n z_n = b - A_e z_e$ di mana A_n sudah diketahui. Misalkan daerah Ω dibagi ke dalam enam ($N = 6$) segitiga yang diperlihatkan pada gambar berikut ini.



Gambar 2 Ilustrasi *mesh* (enam elemen segitiga) dalam Ω

Misalkan koordinat dari setiap node diberikan pada tabel di bawah ini

Tabel 1
Nomor-nomor koordinat sebagai node

Node	1	2	3	4	5	6	7	8
x	0	1	1	1	1	0	0	0
y	0	0	$\frac{1}{3}$	$\frac{2}{3}$	1	1	$\frac{2}{3}$	$\frac{2}{3}$

MATLAB mampu membaca data dari file yang diberikan dalam format ascii dengan file .dat, sehingga dalam MATLAB kita dapat menyimpan data koordinat untuk setiap node dengan pemrograman sebagai berikut.

```
node1 = [1 0 0;
    2 1 0;
    3 1 1/3;
    4 1 2/3;
    5 1 1;
    6 0 1;
    7 0 2/3;
    8 0 1/3];
dlmwrite('koordinat1.dat', node1,' ')
type koordinat1.dat
```

Pada Tabel 2 diberikan elemen-elemen yang berpadanan dengan node-node pada Tabel 1.

Tabel 2
Nomor-nomor elemen segitiga

Elemen	1	2	3	4	5	6
Node 1	7	7	8	8	1	1
Node 2	5	4	4	3	3	2
Node 3	6	5	7	4	8	3

Dengan menggunakan MATLAB kita dapat menyimpan data elemen-elemen di atas dengan pemrograman berikut.

```
segitiga1=[1 7 5 6;
    2 7 4 5;
    3 8 4 7;
    4 8 3 4;
    5 1 3 8;
    6 1 2 3];
dlmwrite('elemen1.dat', segitiga1,' ')
type elemen1.dat
```

Syarat batas untuk Gambar 1 diperlihatkan pada tabel berikut.

Tabel 3
Nomor-nomor edge dengan syarat batas

Edge	1	2	3	4	5	6	7	8
Node 1	1	2	3	4	5	6	7	8
Node 2	2	3	4	5	6	7	8	1

```
deltaomega=[1 1 2;
    2 2 3;
    3 3 4;
    4 4 5;
    5 5 6;
    6 6 7;
    7 7 8;
    8 8 1];
dlmwrite('deltaomega.dat', deltaomega,' ')
type deltaomega.dat
```

Menyusun matriks *stiffness*

Perhatikan bahwa untuk sebuah elemen segitiga T misalkan (x_1, y_1), (x_2, y_2) dan (x_3, y_3) adalah titik-titik verteks dan φ_1, φ_2, dan φ_3 adalah fungsi basis yang berkorespondensi di K, yaitu

$$\varphi_i(x_j, y_j) = \delta_{ij}, \qquad i, j = 1, 2, 3.$$

Oleh karena itu, diperoleh

$$\varphi_i(x,y) = \frac{\det \begin{pmatrix} 1 & x & y \\ 1 & x_{i+1} & y_{i+1} \\ 1 & x_{i+2} & y_{i+2} \end{pmatrix}}{\det \begin{pmatrix} 1 & x_i & y_i \\ 1 & x_{i+1} & y_{i+1} \\ 1 & x_{i+2} & y_{i+2} \end{pmatrix}}, \tag{11}$$

dengan

$$\nabla\varphi_i(x,y) = \frac{1}{2|T|} \begin{pmatrix} y_{i+1} - y_{i+2} \\ x_{i+2} - x_{i+1} \end{pmatrix}. \tag{12}$$

Indeks yang digunakan pada persamaan di atas di dalam modulo 3, sehingga diperoleh

$$2|T| = \det \begin{pmatrix} x_2 - x_1 & x_3 - x_1 \\ y_2 - y_1 & y_3 - y_1 \end{pmatrix}.$$

Matriks *stiffness* yang diperoleh adalah

$$\begin{aligned} M_{ij} &= \int_T \nabla\varphi_i (\nabla\varphi_j)^T dx \\ &= \frac{|T|}{(2|T|)^2} \begin{pmatrix} y_{i+1} - y_{i+2} & x_{i+2} - x_{i+1} \end{pmatrix} \begin{pmatrix} y_{j+1} - y_{j+2} \\ x_{j+2} - x_{j+1} \end{pmatrix}. \end{aligned}$$

dengan indeks di dalam modulo 3. Bentuk sederhana persamaan di atas adalah

$$M = \frac{|T|}{2} G G^T,$$

di mana

$$G = \begin{pmatrix} 1 & 1 & 1 \\ x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \end{pmatrix}^{-1} \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

Pemrograman dalam MATLAB untuk menyusun matriks *stiffness* dapat kita tuliskan sebagai berikut.

```
function M = stima(vertices)
d = size(vertices,2);
G = [ones(1,d+1);vertices'] \ [zeros(1,d);eye(d)];
M = det([ones(1,d+1);vertices']) * G * G' / prod(1:d);
end
```

Menyusun matriks massa

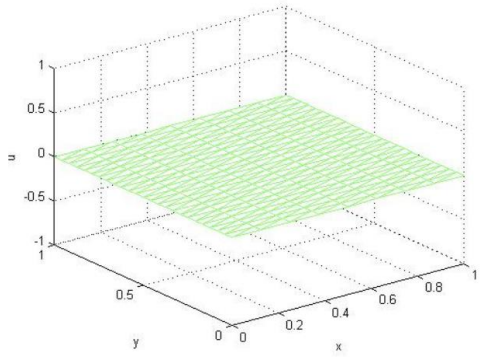
Perhatikan bahwa

$$B_{ij} = \sum_{T\in\Omega} \int_T \varphi_i \varphi_j k dx.$$

Dengan cara yang hampir sama dalam menyusun matriks *stiffness*, diperoleh matriks massa

$$\int_T \varphi_i \varphi_j k dx = \frac{1}{24} \det \begin{pmatrix} x_2 - x_1 & x_3 - x_1 \\ y_2 - y_1 & y_3 - y_1 \end{pmatrix} \begin{pmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{pmatrix} k(x_s, y_s)$$

di mana (x_s, ys) adalah titik tengah pada segitiga T. Untuk k = 0, dengan menggunakan MATLAB diperoleh solusi dalam bentuk grafik berikut ini.



Gambar 3 Solusi kasus 1 untuk k = 0

Solusi analitik pada kasus 1 untuk k = 0 adalah u = 0 mempunyai kesimpulan yang sama dengan solusi numerik yang ditampilkan pada Gambar 3.

```
function k = k0(x,y);
k = 0;
end
function SyaratBatas = delta0(x)
SyaratBatas = zeros(size(x,1),1);
End
```

Pemrograman dengan MATLAB untuk kasus 1 sebagai berikut.

```
load koordinat1.dat; koordinat1(:,1)=[]; %koordinat-
koordinat
load elemen1.dat; elemen1(:,1)=[]; %elemen hingga
berbentuk segitiga
load deltaomega.dat; deltaomega(:,1)=[];
neumann = [];

% Mesh Refinement
mesh = getmesh(koordinat1,elemen1,deltaomega,neumann);
for k = 1:3
     mesh = uniformrefine(mesh);
end
koordinat = mesh.node;
elemen = mesh.elem;
```

```
deltaomega = mesh.Dirichlet;

FreeNodes=setdiff(1:size(koordinat,1),unique(deltaomega
));
A = sparse(size(koordinat,1),size(koordinat,1));
b = sparse(size(koordinat,1),1);
% 1. menyusun matriks A1
% dengan memanggil fungsi stima
for i = 1:size(elemen,1)
A(elemen(i,:),elemen(i,:)) = A(elemen(i,:),elemen(i,:))
     + stima(koordinat(elemen(i,:),:));
end
% 2. menyusun matriks A2
for i = 1:size(elemen,1)
A(elemen(i,:),elemen(i,:))= A(elemen(i,:),elemen(i,:))-
det([ones(1,3);koordinat(elemen(i,:),:)'])*(diag(ones(3
,1))+ ones(3))*k0(sum(koordinat(elemen(i,:),:))/3)/24;
end
u = sparse(size(koordinat,1),1);
u(unique(deltaomega)) =
delta0(koordinat(unique(deltaomega),:));
b = b - A*u;
% 3. Computation of the solution
u(FreeNodes) = A(FreeNodes,FreeNodes) \ b(FreeNodes);
% 4. Graphic representation
trimesh(elemen,koordinat(:,1),koordinat(:,2),full(u));
xlabel('x');ylabel('y');zlabel('u');
```

Kasus 2

Bentuk lemah untuk kasus 2 adalah

$$\int_{\Omega}(\nabla u \cdot \nabla v - kuv)dx = \int_{L_2} qvds. \tag{13}$$

Pendekatan elemen hingga (*finite element approximation*) dari bentuk lemah persamaan (13) adalah $u_h \in V_h$ sedemikian sehingga

$$\int_{\Omega}(\nabla u_h \cdot \nabla v - ku_h v)dx = \int_{L_2} qvds. \tag{14}$$

Dengan cara yang sama seperti yang sudah diuraikan pada kasus 1, terdapat sistem persamaan linear dengan elemen-elemen berikut.

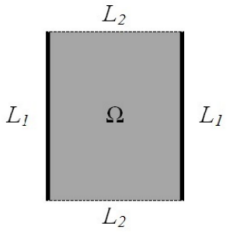
$$\begin{aligned} A_{ij} &= \left(\int_{\Omega}\nabla\varphi_i \cdot \nabla\varphi_j dx - \int_{\Omega}\varphi_i\varphi_j dx\right) \\ b_i &= \int_{L_2}\varphi_i qds \\ z &= (z_1, z_2, ..., z_N)^T. \end{aligned}$$

Dengan menggunakan cara yang serupa pada kasus 1, matriks A_e dihitung pada syarat batas Dirichlet. Sehingga solusi akhir z_n dapat dihitung dengan sistem persamaan

$$A_n z_n = b - A_e z_e.$$

Beberapa simulasi untuk kasus 2 akan dibahas di bawah ini.

a. Misal Ω dan syarat batas diberikan pada gambar berikut



Gambar 4 Syarat batas untuk kasus 2a

Jika Ω dibagi ke dalam enam segitiga seperti pada gambar 2 dengan syarat batas seperti pada Gambar 4, maka dalam MATLAB kita dapat menyimpan data syarat batas dengan pemrograman berikut.

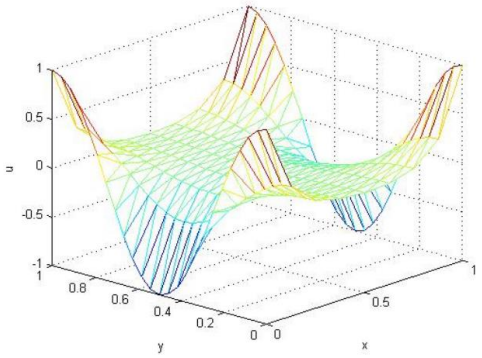
```
neumann2a=[1 1 2;
2 5 6];
dlmwrite('neumann2a.dat',neumann2a,' ')
type neumann2a.dat
dirichlet2a=[1 2 3;
2 3 4;
3 4 5;
4 6 7;
5 7 8;
6 8 1];
dlmwrite('dirichlet2a.dat', dirichlet2a,' ')
type dirichlet2a.dat
```

Misalkan dipilih $k = -(2\pi)\,2$, $p = \cos(2\pi y)$ dan $q = 0$.

```
function k = k1(x,y);
k = -((2*pi))^2;
end
function S = g(x)
S = zeros(size(x,1),1);
end

function a=q1(x);
```

```
a=0;
end

function BatasDirichletKasus2a = ud1(x)
BatasDirichletKasus2a = zeros(size(x,1),1);
I=find(x(:,1)==0|x(:,1)==1);
BatasDirichletKasus2a (I) = cos(2*pi*x(:,2));
end
```

Dengan menggunakan MATLAB diperoleh solusi pada Gambar 5.



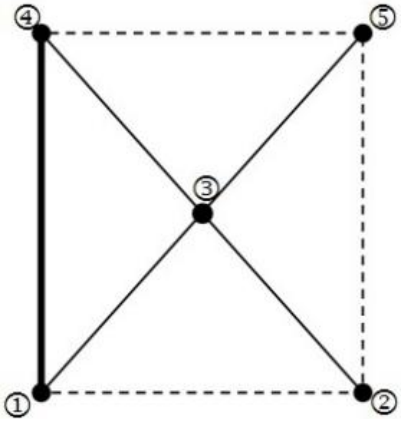
Gambar 5 Solusi kasus 2a

Pemrograman dengan MATLAB sebagai berikut.

```
% inisialisasi
load koordinat2a.dat; koordinat2a(:,1)=[]; %
koordinat-koordinat
load elemen2a.dat; elemen2a(:,1)=[]; % elemen hingga
berbentuk segitiga
load neumann2a.dat; neumann2a(:,1)=[];
load dirichlet2a.dat; dirichlet2a(:,1)=[];

%Mesh Refinement
mesh =
getmesh(koordinat2a,elemen2a,dirichlet2a,neumann2a);
for k = 1:3
   mesh = uniformrefine(mesh);
end

koordinat = mesh.node;
elemen = mesh.elem;
neumann = mesh.Neumann;
dirichlet = mesh.Dirichlet;

FreeNodes=setdiff(1:size(koordinat,1),unique(dirichl
```

```
et));
A = sparse(size(koordinat,1),size(koordinat,1));
b = sparse(size(koordinat,1),1);
% 1. menyusun matriks A1
% dengan memanggil fungsi stima_1
for i = 1:size(elemen,1)
   A(elemen(i,:),elemen(i,:)) =
        A(elemen(i,:),elemen(i,:)) ...
        + stima(koordinat(elemen(i,:),:));
end
% 2. menyusun matriks A2
for i = 1:size(elemen,1)
   A(elemen(i,:),elemen(i,:)) =
     A(elemen(i,:),elemen(i,:))-...
     det([ones(1,3);koordinat(elemen(i,:),:)'])*(diag
     (ones(3,1))+ones(3))*k1(sum(koordinat(elemen(i,:
     ),:))/3)/24;
end
% 3. volume forces
for i = 1:size(elemen,1)
   b(elemen(i,:)) = b(elemen(i,:))+...
      det([ones(1,3);koordinat(elemen(i,:),:)'])*...
      g(sum(koordinat(elemen(i,:))/3))/6;
end
% 4. Neumann condition
for i = 1:size(neumann,1)
   b(neumann(i,:))=b(neumann(i,:))
      +norm(koordinat(neumann(i,1),:)-
      koordinat(neumann(i,2),:))
      *q1(sum(koordinat(neumann(i,:),:))/2)/2;
end
% 5. Dirichlet conditions
u = sparse(size(koordinat,1),1);
u(unique(dirichlet)) =
ud1(koordinat(unique(dirichlet),:));
b = b - A*u;
% 6. Computation of the solution
u(FreeNodes) = A(FreeNodes,FreeNodes) \
b(FreeNodes);
% 7. Graphic representation
trimesh(elemen,koordinat(:,1),koordinat(:,2),full(u)
);
xlabel('x');ylabel('y');zlabel('u');
```

b. Misalkan daerah Ω dibagi ke dalam empat (N = 4) segitiga yang diperlihatkan pada gambar berikut ini



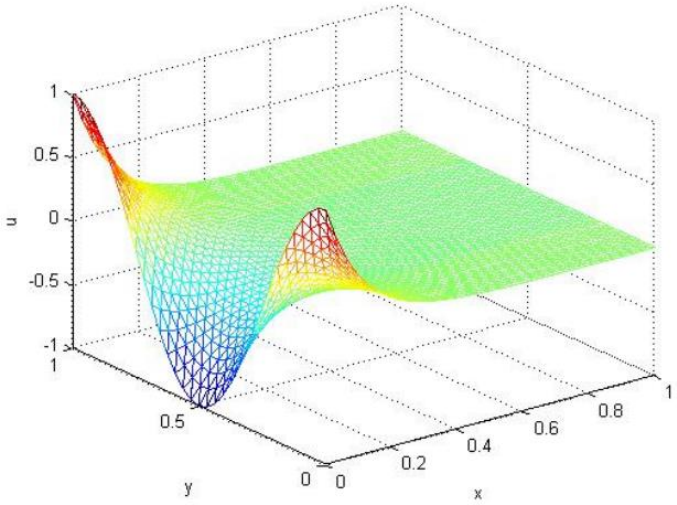
Gambar 6 Daerah Ω dibagi ke dalam empat segitiga

Dengan nomor-nomor node, elemen segitiga dan edge syarat batas pada tabel berikut:

Tabel 4
Nomor-nomor node, elemen, dan *edge* syarat batas

Nomor	Node (Koordinat)	Elemen Segitiga	Syarat Batas Neumann	Syarat batas Dirichlet
1.	(0,0)	1-2-3	1-2	4-1
2.	(1,0)	1-3-4	2-5	
3.	(1/2,1/2)	3-2-5	5-4	
4.	(0,1)	3-5-4		
5.	(1,1)			

Misalkan $k = -(2\pi)\ 2$, $p = cos(2\pi y)$ dan $q = 0$, dengan menggunakan MATLAB, diperoleh solusi dalam bentuk grafik berikut ini:



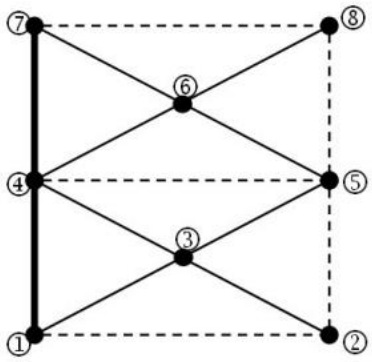
Gambar 7 Solusi Kasus 2b

Pemrograman dengan MATLAB sebagai berikut.

```
load koordinat2b.dat; koordinat2b(:,1)=[]; %
koordinat-koordinat
load elemen2b.dat; elemen2b(:,1)=[]; % elemen hingga
berbentuk segitiga
load neumann2b.dat; neumann2b(:,1)=[];
load dirichlet2b.dat; dirichlet2b(:,1)=[];

%Mesh Refinement
mesh =
getmesh(koordinat2b,elemen2b,dirichlet2b,neumann2b);
for k = 1:5
   mesh = uniformrefine(mesh);
end

koordinat = mesh.node;
elemen = mesh.elem;
neumann = mesh.Neumann;
dirichlet = mesh.Dirichlet;

FreeNodes=setdiff(1:size(koordinat,1),unique(dirichl
et));
A = sparse(size(koordinat,1),size(koordinat,1));
b = sparse(size(koordinat,1),1);
% 1. menyusun matriks A1
```

```
% dengan memanggil fungsi stima_1
for i = 1:size(elemen,1)
   A(elemen(i,:),elemen(i,:)) =
      A(elemen(i,:),elemen(i,:)) ...
      + stima(koordinat(elemen(i,:),:));
end
% 2. menyusun matriks A2
for i = 1:size(elemen,1)
   A(elemen(i,:),elemen(i,:)) =
      A(elemen(i,:),elemen(i,:))-...
      det([ones(1,3);koordinat(elemen(i,:),:)'])*...
      (diag(ones(3,1))+
      ones(3))*k1(sum(koordinat(elemen(i,:),:))/3)/24
      ;
end
% 3. volume forces
for i = 1:size(elemen,1)
   b(elemen(i,:)) = b(elemen(i,:))+...
     det([ones(1,3);koordinat(elemen(i,:),:)'])*...
     g(sum(koordinat(elemen(i,:))/3))/6;
end
% 4. Neumann condition
for i = 1:size(neumann,1)
   b(neumann(i,:))=b(neumann(i,:)) +...
     norm(koordinat(neumann(i,1),:)- ...
     koordinat(neumann(i,2),:)) *...
     q1(sum(koordinat(neumann(i,:),:))/2)/2;
end
% 5. Dirichlet conditions
u = sparse(size(koordinat,1),1);
u(unique(dirichlet)) =
ud1(koordinat(unique(dirichlet),:));
b = b - A*u;
% 6. Computation of the solution
u(FreeNodes) = A(FreeNodes,FreeNodes) \
b(FreeNodes);
% 7. Graphic representation
trimesh(elemen,koordinat(:,1),koordinat(:,2),full(u)
);
xlabel('x');ylabel('y');zlabel('u');
```

c. Misalkan daerah Ω dibagi ke dalam delapan (N = 8) segitiga yang diperlihatkan pada gambar berikut ini



Gambar 8 Daerah Ω dibagi ke dalam delapan segitiga

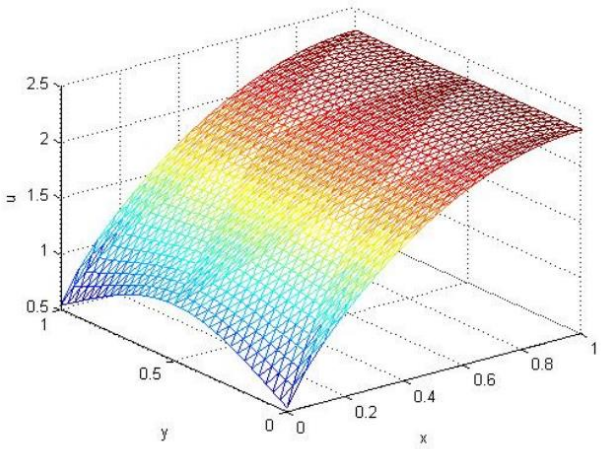
Dengan nomor-nomor node, elemen segitiga dan edge syarat batas pada Tabel 5.

Tabel 5
Nomor-nomor node, elemen, dan *edge* syarat batas

Nomor	Node (Koordinat)	Elemen Segitiga	Syarat Batas Neumann	Syarat batas Dirichlet
1.	(0,0)	1-2-3	1-2	7-4
2.	(1,0)	1-3-4	2-5	4-1
3.	(1/2,1/4)	3-2-5	5-8	
4.	(0,1/2)	4-3-5	8-7	
5.	(1,1/2)	4-5-6		
6.	(1/2,3/4)	4-6-7		
7.	(0,1)	6-5-8		
8.	(1,1)	7-6-8		

```
function k = k3(x)
if x(:,2)<=0.5
     k = 1.5;
else if x(:,2)>0.5
          k = 1.4;
     end
end
end
function DirichletBoundaryValueCase2d = ud3(x)
DirichletBoundaryValueCase2d = zeros(size(x,1),1);
I=find(x(:,1)==0);
DirichletBoundaryValueCase2d(I) = exp((-(x(I,2)-
0.5).^2)./0.4);
end
```

Dengan menggunakan MATLAB, diperoleh solusi dalam bentuk grafik berikut ini:



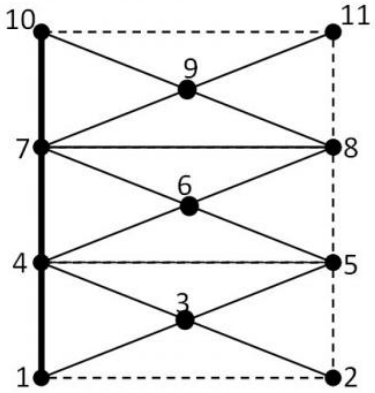
Gambar 9 Solusi Kasus 2c

Pemrograman dengan MATLAB sebagai berikut.

```
load koordinat2c.dat; koordinat2c(:,1)=[]; %
koordinat-koordinat
load elemen2c.dat; elemen2c(:,1)=[]; % elemen
hingga berbentuk segitiga
load neumann2c.dat; neumann2c(:,1)=[];
load dirichlet2c.dat; dirichlet2c(:,1)=[];

%Mesh Refinement
mesh =
getmesh(koordinat2c,elemen2c,dirichlet2c,neumann2c
);
for k = 1:4
     mesh = uniformrefine(mesh);
end

koordinat = mesh.node;
elemen = mesh.elem;
neumann = mesh.Neumann;
dirichlet = mesh.Dirichlet;

FreeNodes=setdiff(1:size(koordinat,1),unique(diric
hlet));
A = sparse(size(koordinat,1),size(koordinat,1));
b = sparse(size(koordinat,1),1);
% 1. menyusun matriks A1
```

```
% dengan memanggil fungsi stima_1
for i = 1:size(elemen,1)
     A(elemen(i,:),elemen(i,:)) =
        A(elemen(i,:),elemen(i,:)) ...
        + stima(koordinat(elemen(i,:),:));
end
% 2. menyusun matriks A2
for i = 1:size(elemen,1)
     A(elemen(i,:),elemen(i,:)) =
        A(elemen(i,:),elemen(i,:))-...
        det([ones(1,3);koordinat(elemen(i,:),:)'])
        *...
        (diag(ones(3,1))+
        ones(3))*k3(sum(koordinat(elemen(i,:),:))/
        3)/24;
end
% 3. volume forces
for i = 1:size(elemen,1)
     b(elemen(i,:)) = b(elemen(i,:))+...
        det([ones(1,3);koordinat(elemen(i,:),:)'])
        *...
        g(sum(koordinat(elemen(i,:))/3))/6;
end
% 4. Neumann condition
for i = 1:size(neumann,1)
     b(neumann(i,:))=b(neumann(i,:)) +...
        norm(koordinat(neumann(i,1),:)- ...
        koordinat(neumann(i,2),:)) *...
        q1(sum(koordinat(neumann(i,:),:))/2)/2;
end
% 5. Dirichlet conditions
u = sparse(size(koordinat,1),1);
u(unique(dirichlet)) =
ud3(koordinat(unique(dirichlet),:));
b = b - A*u;
% 6. Computation of the solution
u(FreeNodes) = A(FreeNodes,FreeNodes) \
b(FreeNodes);
% 7. Graphic representation
trimesh(elemen,koordinat(:,1),koordinat(:,2),full(
u));
xlabel('x');ylabel('y');zlabel('u');
```

d. Misalkan daerah **Ω** dibagi ke dalam duabelas (N = 12) segitiga yang diperlihatkan pada Gambar 10.



Gambar 10 Daerah **Ω** dibagi ke dalam duabelas segitiga

Sedangkan nomor-nomor node, elemen segitiga dan node dengan syarat batas pada Tabel 6. Misalkan

$$k = \begin{cases} 1.4; & y > \frac{2}{3} \text{ atau } y < \frac{1}{3} \\ 1.5; & \frac{1}{3} \le y \le \frac{2}{3}. \end{cases}$$

Tabel 6
Nomor-nomor node, elemen, dan *edge* syarat batas

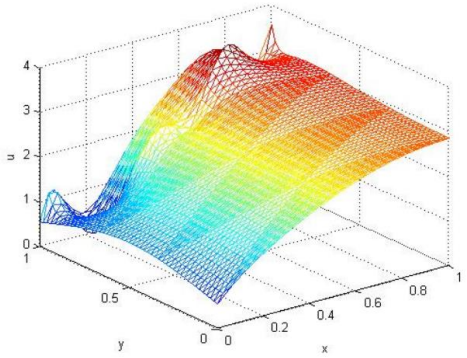
Nomor	Node (Koordinat)	Elemen Segitiga	Syarat Batas Neumann	Syarat batas Dirichlet
1.	(0,0)	1-2-3	1-2	10-7
2.	(1,0)	1-3-4	2-5	7-4
3.	(1/2,1/6)	3-2-5	5-8	4-1
4.	(0,1/3)	3-5-4	8-11	
5.	(1,1/3)	4-5-6	11-10	
6.	(1/2,1/2)	4-6-7		
7.	(0,2/3)	6-5-8		
8.	(1,2/3)	6-8-7		
9.	(1/2,5/6)	7-8-9		
10.	(0,1)	7-9-10		
11.	(1,1)	9-8-11		
12.		9-10-11		

sedangkan p dan q sama seperti pada kasus 2*c*.

```
function k = k4(x)
if (x(:,2)<1/3)||(x(:,2)>2/3)
      k = 1.4;
```

```
else
     k = 1.5;
end
end
```

Dengan menggunakan MATLAB, diperoleh solusi dalam bentuk grafik berikut ini:



Gambar 11 Solusi Grafik Kasus 2d

Pemrograman dengan MATLAB sebagai berikut.

```
load koordinat2d.dat; koordinat2d(:,1)=[]; %
koordinat-koordinat
load elemen2d.dat; elemen2d(:,1)=[]; % elemen
hingga berbentuk segitiga
load neumann2d.dat; neumann2d(:,1)=[];
load dirichlet2d.dat; dirichlet2d(:,1)=[];

%Mesh Refinement
mesh =
getmesh(koordinat2d,elemen2d,dirichlet2d,neumann2d
);
for k = 1:4
     mesh = uniformrefine(mesh);
end

koordinat = mesh.node;
elemen = mesh.elem;
neumann = mesh.Neumann;
dirichlet = mesh.Dirichlet;

% for k = 1:5
% [Kantennr,elemen] =
```

```
GeneriereKantennr(elemen,koordinat);
% VK = (1:full(max(max(Kantennr))))' +
size(koordinat,1);
% [koordinat,elemen,dirichlet,neumann] = ...
%
Verfeinerung(koordinat,elemen,dirichlet,neumann,Ka
ntennr,VK);
% end

FreeNodes=setdiff(1:size(koordinat,1),unique(diric
hlet));
A = sparse(size(koordinat,1),size(koordinat,1));
b = sparse(size(koordinat,1),1);
% 1. menyusun matriks A1
% dengan memanggil fungsi stima_1
for i = 1:size(elemen,1)
     A(elemen(i,:),elemen(i,:)) =
        A(elemen(i,:),elemen(i,:)) ...
        + stima(koordinat(elemen(i,:),:));
end
% 2. menyusun matriks A2
for i = 1:size(elemen,1)
     A(elemen(i,:),elemen(i,:)) =
        A(elemen(i,:),elemen(i,:))-...
        det([ones(1,3);koordinat(elemen(i,:),:)'])
        *...
        (diag(ones(3,1))+
        ones(3))*k4(sum(koordinat(elemen(i,:),:))/
        3)/24;
end
% 3. volume forces
for i = 1:size(elemen,1)
     b(elemen(i,:)) = b(elemen(i,:))+...
        det([ones(1,3);koordinat(elemen(i,:),:)'])
        *...
        g(sum(koordinat(elemen(i,:))/3))/6;
end
% 4. Neumann condition
for i = 1:size(neumann,1)
     b(neumann(i,:))=b(neumann(i,:)) +...
        norm(koordinat(neumann(i,1),:)- ...
        koordinat(neumann(i,2),:)) *...
        q1(sum(koordinat(neumann(i,:),:))/2)/2;
end
% 5. Dirichlet conditions
```

```
u = sparse(size(koordinat,1),1);
u(unique(dirichlet)) =
ud3(koordinat(unique(dirichlet),:));
b = b - A*u;
% 6. Computation of the solution
u(FreeNodes) = A(FreeNodes,FreeNodes) \
b(FreeNodes);
% 7. Graphic representation
trimesh(elemen,koordinat(:,1),koordinat(:,2),full(
u));
xlabel('x');ylabel('y');zlabel('u');
```

4 SIMPULAN

Kode MATLAB berhubungan langsung dengan operasi FEM. Ditunjukkan bagaimana FEM diperkenalkan kepada mahasiswa hanya menggunakan beberapa baris kode MATLAB. Kedua contoh yang diberikan dapat dijalankan pada PC standar. Hal ini juga menunjukkan bagaimana konsep-konsep yang diperoleh dapat dengan mudah diperluas ke masalah lain yang diberikan bersifat 2D. Namun, teknik pemrograman yang sama dapat digunakan untuk masalah yang lebih kompleks, seperti masalah 3D yang misalnya disajikan oleh Silvester dan Ferrari [5]. Lembaga pendidikan dengan sumber daya keuangan yang terbatas dapat mengambil manfaat dari metodologi yang diberikan, karena hanya memerlukan MATLAB, tidak ada add-ons (bahkan pembangkit mesh sekalipun). Materi yang disajikan dalam makalah ini mungkin juga berguna untuk para pemula penelitian yang belum mengenal FEM.

DAFTAR PUSTAKA

[1] Lu J, Thiel DV. 2000. Computational and visual electromagnetics using an integrated programming language for undergraduate engineering students. *IEEE Trans, Magnetics*. 36: 1000-1003.
[2] Selleri S. 2003. A MATLAB exp erimental framework for electromagnetic education. *IEEE Antennas and Propagat, Mag*. 45: 85-90.
[3] Selleri S. 2003. A MATLAB application programmer interface for educational electromagnetics in Antennas and Propagat. *So c, Symp*.
[4] MATLAB, The Math Works Inc., http://www.mathworks.com.
[5] Silvester PP, Ferrari RL. 1990. *Finite Elements for Electrical Engineers*. Cambridge: Cambridge University Press.
[6] Hoole SRH. 1989. *Computer-Aided Design of Electromagnetic Devices*. New York: Elsevier.

[7] Foster K. 2004. Retaking the field an old computational favourite is overhauled. *IEEE Spectrum*. 54-55.
[8] Tio LY, Gibson AAP, Dillon BM, Davis LE. 2004. Weak form finite element formulation for Helmholtz equation. *Int. J. Elect. Enging. Educ*. 41(19).
[9] Martinez MJ, McTigue DF. 1991. The steady distribution of moisture beneath a two-dimensional surface source. *Water resources research* (*Wiley Online Library*). 27: 1193-1206.
[10] Natalini B and Popov V. 2007. Boundary element formulation for flow in unsaturated porous media. *Mecanica Computational*. 26: 1158-1173.
[11] Philip JR. 1989. Multidimensional steady infiltration to a water table. *Water resources research* (*Wiley Online Library*). 25: 109-116.
[12] Pullan AJ. 1990. The quasilinear approximation for unsaturated porous media flow. *Water resources research* (*Wiley Online Library*). 26: 1219-1234.
[13] Bragg LR, Dettman JW. 1995. Function theories for the Yukawa and Helmholtz equations. *Journal of Mathematics*.
[14] Duffin RJ. 1971. Yukawan potential theory. *Journal of Mathematical Analysis and Applications* (*Elsevier*). 35: 105-130.