ภาคผนวก ก

รูปของสมการในวิธีไฟไนต์อีลีเมนต์ที่ใช้สนามแม่เหล็กตามขวาง 2 องค์ประกอบ

นิพจน์แปรผันของสมการคลื่นในวิธีไฟไนต์อีลีเมนต์ที่ใช้สนามแม่เหล็กตามขวาง 2 องค์ประกอบ คือ

$$\tilde{F}(H,k_0)=\{H\}^T\left([S]-\left(k_0/\beta\right)^2[\tilde{M}]\right)\{H\}$$

เมื่อ

$$\{H\}=\begin{bmatrix}\{H_x\}\\ \{H_y\}\\ \{H_z\}\end{bmatrix}$$

เมตริก $[S]$ หาได้จากสมการต่อไปนี้

$$[S]=\sum_e\iint_e[T]^*[\varepsilon_r]^{-1}[T]^T\,dxdy$$

เมื่อ

$$[T]=\begin{bmatrix}\{0\} & -j\beta\{N\} & \dfrac{-\partial\{N\}}{\partial x}\\ j\beta\{N\} & \{0\} & \dfrac{\partial\{N\}}{\partial x}\\ \dfrac{j\partial\{N\}}{\partial y} & \dfrac{-j\partial\{N\}}{\partial x} & \{0\}\end{bmatrix}$$

กำหนดให้องค์ประกอบของเมตริก $[\varepsilon_r]^{-1}$ คือ

$$[\varepsilon_r]^{-1}=\begin{bmatrix}\rho_{xx} & \rho_{xy} & \rho_{xz}\\ \rho_{yx} & \rho_{yy} & \rho_{yz}\\ \rho_{zx} & \rho_{zy} & \rho_{zz}\end{bmatrix}$$

และกำหนดให้เมตริกย่อยของ $[S]$ คือ

$$[S]=\begin{bmatrix}[S_{xx}] & [S_{xy}] & [S_{xz}]\\ [S_{yx}] & [S_{yy}] & [S_{yz}]\\ [S_{zx}] & [S_{zy}] & [S_{zz}]\end{bmatrix}$$

โดยที่เมตริกย่อย $[S_{xx}],[S_{xy}],...,[S_{zz}]$ ในแต่ละอีลีเมนต์มีอันดับเป็น 3x3 โดยจะหาได้จาก

$$[S_{xx}]=\sum_e\iint_e\left(\beta^2\rho_{yy}\{N\}\{N\}^T-j\beta\rho_{yz}\{N\}\{N\}_y^T+j\beta\rho_{zy}\{N\}_y\{N\}^T+\rho_{zz}\{N\}_y\{N\}_y^T\right)dxdy$$

$$[S_{xy}]=\sum_e\iint_e\left(-\beta^2\rho_{yx}\{N\}\{N\}^T+j\beta\rho_{yz}\{N\}\{N\}_x^T-j\beta\rho_{zx}\{N\}_y\{N\}^T-\rho_{zz}\{N\}_y\{N\}_x^T\right)dxdy$$

$$[S_{xz}]=\sum_e\iint_e\left(-\beta\rho_{yx}\{N\}\{N\}_y^T+\beta\rho_{yy}\{N\}\{N\}_x^T-j\rho_{zx}\{N\}_y\{N\}_y^T+j\rho_{zy}\{N\}_y\{N\}_x^T\right)dxdy$$

$$[S_{yx}]=\sum_e\iint_e\left(-\beta^2\rho_{xy}\{N\}\{N\}^T+j\beta\rho_{xz}\{N\}\{N\}_y^T-j\beta\rho_{zy}\{N\}_x\{N\}^T-\rho_{zz}\{N\}_x\{N\}_y^T\right)dxdy$$

$$[S_{yy}]=\sum_e\iint_e\left(\beta^2\rho_{xx}\{N\}\{N\}^T-j\beta\rho_{xz}\{N\}\{N\}_x^T-j\beta\rho_{zx}\{N\}_x\{N\}^T+\rho_{zz}\{N\}_x\{N\}_x^T\right)dxdy$$

$$[S_{yz}]=\sum_e\iint_e\left(\beta\rho_{xx}\{N\}\{N\}_y^T-\beta\rho_{xy}\{N\}\{N\}_x^T+j\rho_{zx}\{N\}_x\{N\}_y^T-j\rho_{zy}\{N\}_x\{N\}_x^T\right)dxdy$$

$$[S_{zx}]=\sum_e\iint_e\left(-\beta\rho_{xy}\{N\}_y\{N\}^T+j\rho_{xz}\{N\}_y\{N\}_y^T+\beta\rho_{yy}\{N\}_x\{N\}^T-j\rho_{yz}\{N\}_x\{N\}_y^T\right)dxdy$$

$$[S_{zy}]=\sum_e\iint_e\left(\beta\rho_{xx}\{N\}_y\{N\}^T-j\rho_{xz}\{N\}_y\{N\}_x^T-\beta\rho_{yx}\{N\}_x\{N\}^T+j\rho_{yz}\{N\}_x\{N\}_x^T\right)dxdy$$

$$[S_{zz}]=\sum_e\iint_e\left(\rho_{xx}\{N\}_y\{N\}_y^T-\rho_{xy}\{N\}_y\{N\}_x^T-\rho_{yx}\{N\}_x\{N\}_y^T+\rho_{yy}\{N\}_x\{N\}_x^T\right)dxdy$$

ส่วนเมตริก $\left[\tilde{M}\right]$ สามารถหาได้จากสมการ

$$\left[\tilde{M}\right]=\sum_e\iint_e\mu_r[N]^*[N]^T\,d\bar{x}d\bar{y}$$

เมื่อ

$$[N]=\begin{bmatrix}\{N\} & \{0\} & \{0\}\\ \{0\} & \{N\} & \{0\}\\ \{0\} & \{0\} & j\{N\}\end{bmatrix}$$

$$\bar{x}=\beta x$$
$$\bar{y}=\beta y$$

องค์ประกอบของเมตริก $\left[\tilde{M}\right]$ คือ

$$\left[\tilde{M}\right]=\begin{bmatrix}\begin{bmatrix}\tilde{M}_{xx}\\ \tilde{M}_{yx}\\ \tilde{M}_{zx}\end{bmatrix} & \begin{bmatrix}\tilde{M}_{xy}\\ \tilde{M}_{yy}\\ \tilde{M}_{zy}\end{bmatrix} & \begin{bmatrix}\tilde{M}_{xz}\\ \tilde{M}_{yz}\\ \tilde{M}_{zz}\end{bmatrix}\end{bmatrix}$$

โดยที่เมตริกย่อย $\left[\tilde{M}_{xx}\right],\left[\tilde{M}_{xy}\right],\ldots,\left[\tilde{M}_{zz}\right]$ ในแต่ละอีลีเมนต์มีอันดับเป็น 3x3 สามารถหาได้จากสมการต่อไปนี้

$$\left[\tilde{M}_{xx}\right]=\left[\tilde{M}_{yy}\right]=\left[\tilde{M}_{zz}\right]=\sum_{e}\iint_{e}\mu_r\{N\}\{N\}^T\,d\bar{x}d\bar{y}$$

$$\left[\tilde{M}_{xy}\right]=\left[\tilde{M}_{yx}\right]=[0]$$

$$\left[\tilde{M}_{xz}\right]=\left[\tilde{M}_{zx}\right]=[0]$$

$$\left[\tilde{M}_{yz}\right]=\left[\tilde{M}_{zy}\right]=[0]$$

โดยที่

$$\{N\}_x \equiv \partial\{N\}/\partial x$$

$$\{N\}_y \equiv \partial\{N\}/\partial y$$

เมตริกไดเวอร์เจนซ์ของความหนาแน่นฟลักซ์แม่เหล็กคือ

$$[D_z]\{H_z\}=[D_t]\{H_t\}$$

เมื่อ

$$[D_z] = \sum_e \iint_e \mu\{N\}\{N\}^T \, d\bar{x}d\bar{y}$$

$$[D_t] = -\sum_e \iint_e \left[\mu\{N\}\partial\{N\}^T / \partial\bar{x} \quad \mu\{N\}\partial\{N\}^T / \partial\bar{y} \right] d\bar{x}d\bar{y}$$

$$\{H_t\} = \begin{bmatrix} \{H_x\} \\ \{H_y\} \end{bmatrix}$$

ภาคผนวก ข

Integral Theorems

ทฤษฎีของแกเดียน

$$\iiint_V \nabla f dV = \oiint_S \hat{n} f dS$$

ทฤษฎีไดเวอร์เจน

$$\iiint_V \nabla \cdot f dV = \oiint_S \hat{n} \cdot f dS$$

ทฤษฎีเคริล

$$\iiint_V \nabla \times f dV = \oiint_S \hat{n} \times f dS$$

ทฤษฎี Croos-gradient

$$\iint_S \hat{n} \times \nabla f dS = \oint_C \hat{l} f dl$$

ทฤษฎีของสโตค

$$\iint_S \hat{n} \cdot \nabla f dS = \oint_C \hat{l} \cdot f dl$$

ทฤษฎี Croos-del-croos

$$\iint_S (\hat{n} \times \nabla) \times f dS = \oint_C \hat{l} \times f dl$$

ทฤษฎีสเกลาข้อที่ 1 ของกรีน

$$\iiint_V [a\nabla \cdot (u\nabla b) + u(\nabla a)\cdot(\nabla b)]dV = \oiint_S au\frac{\partial b}{\partial n}dS$$

ทฤษฎีสเกลาข้อที่ 2 ของกรีน

$$\iiint_V [a\nabla \cdot (u\nabla b) - b\nabla \cdot (u\nabla a)]dV = \oiint_S u\left(a\frac{\partial b}{\partial n} - b\frac{\partial a}{\partial n}\right)dS$$

ทฤษฎีเวกเตอร์ข้อที่ 1 ของกรีน

$$\iiint_V [u(\nabla \times a)\cdot(\nabla \times b) - a\cdot(\nabla \times u\nabla \times b)]dV = \oiint_S u(a \times \nabla \times b)\cdot \hat{n}dS$$

ทฤษฎีเวกเตอร์ข้อที่ 2 ของกรีน

$$\iiint_V [b\cdot(\nabla \times u\nabla \times a) - a\cdot(\nabla \times u\nabla \times b)]dV = \oiint_S u(a \times \nabla \times b - b \times \nabla \times a)\cdot \hat{n}dS$$

ภาคผนวก ค

อินทิกรัลของฟังก์ชันรูปร่าง

อินทิกรัลของฟังก์ชันรูปร่างในอีลีเมนต์สามเหลี่ยมสามารถหาได้จากสมการต่อไปนี้ (Kardestuncer , 1988 , Sivester และ ferrari , 1990)

$$I^e(l,m,n) = \iint_e (L_1)^l (L_2)^m (L_3)^n \, dxdy$$

$$= \frac{l!\,m!\,n!}{(l+m+n)!} 2A$$

เมื่อ (L_1, L_2, L_3) คือฟังก์ชันเชิงเส้นที่สามารถหาได้จากสมการ

$$\begin{bmatrix} L_1 \\ L_2 \\ L_3 \end{bmatrix} = \frac{1}{2A} \begin{bmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \end{bmatrix} \begin{bmatrix} 1 \\ x \\ y \end{bmatrix}$$

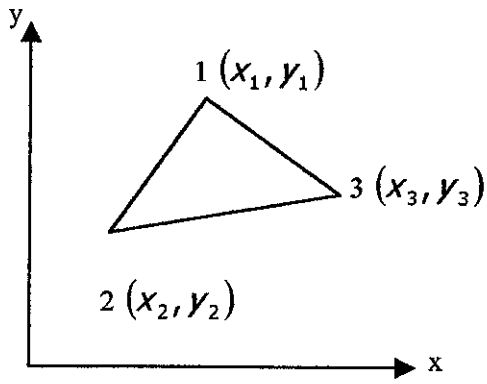
เมื่อ

$$a_k = x_l y_m - x_m y_l$$
$$b_k = y_l - y_m$$
$$c_k = x_m - x_l$$

โดยที่ (k,l,m) เรียงในลักษณะมอดุโล 3 ,$(x_1, y_1), (x_2, y_2)$ และ (x_3, y_3) คือพิกัดของมุม 1, 2 และ 3 ของอีลีเมนต์สามเหลี่ยม ตามลำดับ ดังแสดงในรูป ฉ.1 , A คือพื้นที่ของอีลีเมนต์รูปสามเหลี่ยมซึ่งหาได้จากสมการ

$$A = \frac{1}{2} \begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{bmatrix}$$

โดยที่ [ ] คือตัวกำหนด



รูป ฉ.1 อีลีเมนต์รูปสามเหลี่ยม และพิกัดโนด

สำหรับอีลีเมนต์อันดับที่หนึ่ง ดังแสดงในรูป ฉ.1 ฟังก์ชันรูปร่าง $\{N\}$ คือ

$$\{N\} = [L_1 \quad L_2 \quad L_3]^T$$

ผลอินทิกรัลของฟังก์ชันรูปร่างสำหรับแต่ละอีลีเมนต์ในวิธีไฟไนต์อีลีเมนต์ที่ใช้สนามแม่เหล็ก 3 องค์ประกอบ วิธีไฟไนต์อีลีเมนต์ที่ใช้สนามไฟฟ้า 3 องค์ประกอบ วิธีไฟไนต์อีลีเมนต์ที่ใช้แม่เหล็กสนามตามขวาง 2 องค์ประกอบ และวิธีไฟไนต์อีลีเมนต์ที่ใช้สนามไฟฟ้าตามขวาง 2 องค์ประกอบ และวิธีไฟไนต์อีลีเมนต์ที่ใช้สนามไฟฟ้าในแนวสัมผัสและในแนวแกน มีดังนี้

$$\left[\iint_e \{N\}\{N\}^T dxdy\right]_{ij} = \begin{cases} \frac{A}{6}, i = j \\ \frac{A}{12}, i \neq j \end{cases}$$

$$\left[\iint_e \{N\}_x \{N\}_x^T dxdy\right]_{ij} = \frac{1}{4A} b_i b_j$$

$$\left[\iint_e \{N\}_x \{N\}_y^T dxdy\right]_{ij} = \frac{1}{4A} b_i c_j$$

$$\left[\iint_e \{N\}_y \{N\}_y^T dxdy\right]_{ij} = \frac{1}{4A} c_i c_j$$

$$\left[\iint_e \{N\}\{N\}_x^T dxdy\right]_{ij} = \frac{b_j}{6}$$

$$\left[\iint_e \{N\}\{N\}_y^T dxdy\right]_{ij} = \frac{c_j}{6}$$

$$\iint_e \{N\}_y \{N\}_x^T dxdy = \left[\iint_e \{N\}_y \{N\}_y dxdy\right]^T$$

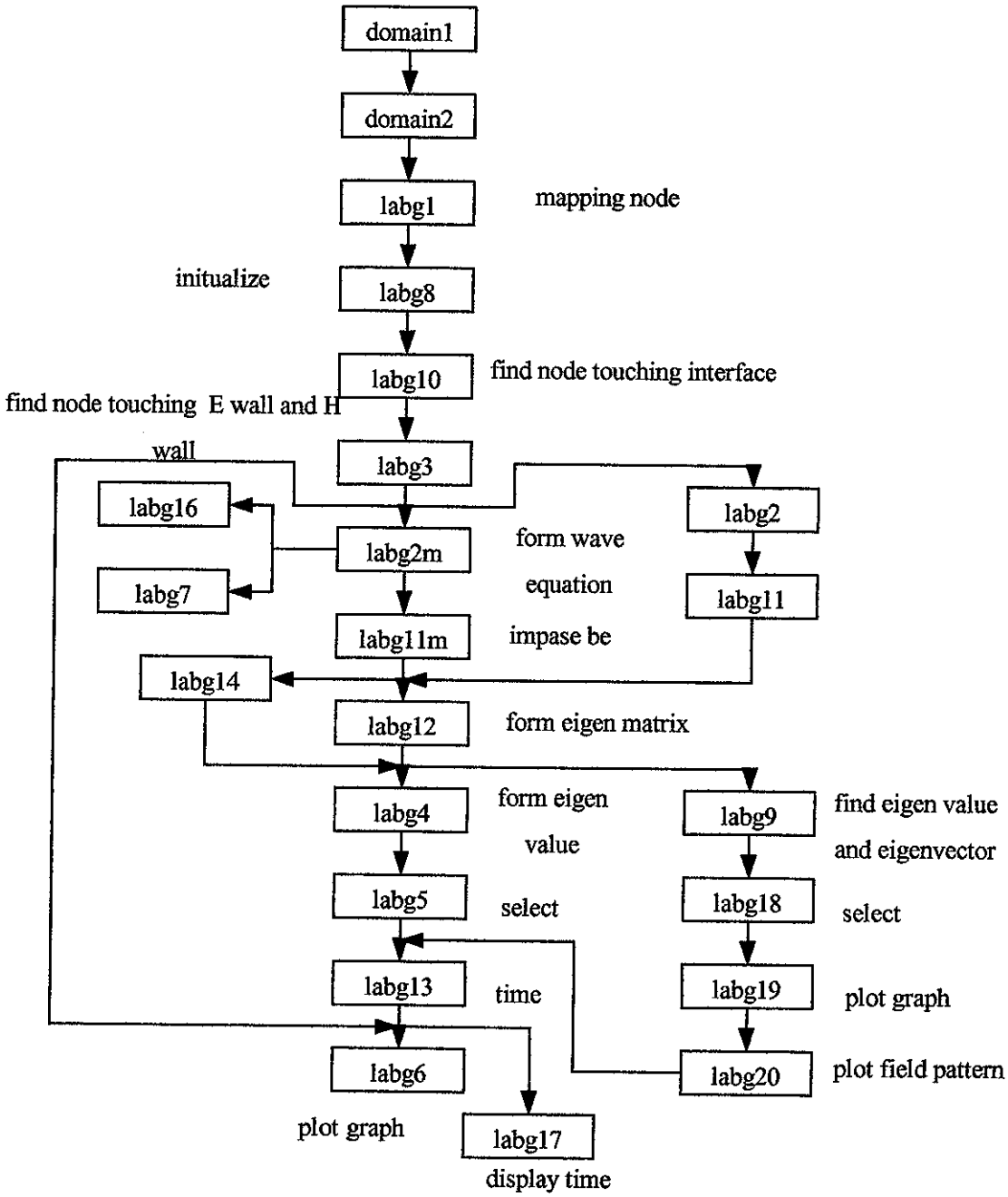
$$\iint_e \{N\}_x \{N\}^T dxdy = \left[\iint_e \{N\}\{N\}_x^T dxdy\right]^T$$

$$\iint_e \{N\}_y \{N\}^T dxdy = \left[\iint_e \{N\}\{N\}_y^T dxdy\right]^T$$

เมื่อ $i = 1,2,3$ และ $j = 1,2,3$

ภาคผนวก ง

แผนผังของโปรแกรมที่ใช้ในการวิเคราะห์ท่อนำคลื่นแบบไอโซทรอปิกและท่อนำแสงแบบแอนไอโซทรอปิกไฟฟ้า



จากแผนผังโปรแกรมสามารถอธิบายได้ดังนี้ นำค่าที่ได้มาทำการ mapping node และ initualize ε, μ จากนั้นหาโนดที่สัมผัสกัน หาโนดที่สัมผัสกันในส่วนที่เป็น E wall ทำให้อยู่ในรูปสมการคลื่น ทำเป็น eigen metrix และ eigen value จากนั้นทำการ plot graph และ plot field pattern

โปรแกรมที่ใช้ในการวิเคราะห์ท่อนำคลื่นแบบไอโซทรอปิกและท่อนำแสงแบบแอนไอโซทรอปิกไฟฟ้า

```
%************
%main program
%************
runmod  %call for runmode

if rmfg==1
 clc
 fprintf('\n\n')
 disp(' ************************************************************')
 disp(' *                                                          *')
 disp(' *    Vectorial Finite-element Method Without               *')
 disp(' *    Any Spurious Solutions for Electrically               *')
 disp(' *    and Magnetically Anisotropic Waveguiding Problems     *')
 disp(' *    Using transverse Magnetic-Fields Component.           *')
 disp(' *                                                          *')
 disp(' ************************************************************')
end

if rmfg==1
 while 1
 fprintf('\n\n')
 disp(' Do you wish to run program')
 disp('  1. yes')
```

```
 disp('  2. no')
 temp=input(' input your selection >> ');
 if (temp==2)|(temp==1),break,end
end
else
 temp=1;  %data for background mode
end

if temp==2,break,end

clear all
save dhdg.mat
save infor.mat

format short

runmod
tic
domain1
timed1=toc;

if (ehtnfg==1),break,end

if disfg==2  %non-dispersive media
 tic
 domain2  %call for permeability and permittivity
 timed2=toc;
end

if disfg==2
```

```
 if rmfg==1
  while 1
   fprintf('\n\n')
   disp(' select your eigen value')
   disp('  1. k0 is eigen value')
   disp('  2. beta is eigen value')
   k0fg=input(' input your selection >> ');
   if (k0fg==1)|(k0fg==2),break,end
  end
 else
  k0fg=1;  %data for background mode
 end
else
 k0fg=2;  %media is dispersive
end

if (mhtnfg==1)&(k0fg==2),break,end

if rmfg==1
 while 1
  fprintf('\n\n')
  k0mn=input(' input the minimum value for k0 axis >> ');
  k0mx=input(' input the maximum value for k0 axis >> ');
  if (k0mn>=0)&(k0mn<k0mx),break,end
 end
else
 k0mn=1;
 k0mx=9;
end
```

```
if disfg==2
 if rmfg==1
  while 1
   fprintf('\n\n')
   disp(' select your range for beta/k0 axis')
   disp('  1. automatic adapt range')
   disp('  2. custom adapt range')
   lifg=input('input your selection >> ');
   if (lifg==1)|(lifg==2),break,end
  end
 else
  lifg=1;  %data for background mode
 end
else
 lifg=2;  %data for dispersive media
end

if lifg==2
 while 1
  nefmn=[0,0];
  nefmx=[0,0];
  if rmfg==1
   fprintf('\n\n')
   nefmn(1,2)=input(' input minimum value for beta/k0 axis >> ');
   nefmx(1,2)=input(' input maximum value for beta/k0 axis >> ');
  else
   nefmn(1,2)=0;  %data for background mode
   nefmx(1,2)=9;
  end
 temp=(nefmx(1,2)>nefmn(1,2));
```

```
 if (temp==1)
  clear temp;
  timelt=0;
  save dsltg.mat nefmn nefmx k0mx k0mn
  break
 end
end
else
 limit
end

if cwfg==2  %rectangular cross-section waveguide
 if rmfg==1
  fprintf('\n\n')
  disp('  A                B')
  disp(' ----------------------')
  disp(' |                  |')
  disp(' |                  |')
  disp(' |   waveguide      |')
  disp(' |                  |')
  disp(' |                  |')
  disp(' ----------------------')
  disp('  D                C')
  fprintf('\n')
  disp(' Impose boundary condition')
  disp(' Enter  1.for H wall')
  disp('        2.for E wall')
  disp('        3.for not inpose anything')
  fprintf('\n')
```

```
while 1
 abside =input(' AB side >> ');
 if abside==1|abside==2|abside==3,break,end
end

while 1
 bcside=input(' BC side >> ');
 if bcside==1|bcside==2|bcside==3,break,end
end

while 1
 cdside=input(' CD side >> ');
 if cdside==1|cdside==2|cdside==3,break,end
end

while 1
 daside=input(' DA side >> ');
 if daside==1|daside==2|daside==3,break,end
end

else %data for background mode
 abside=2;
 bcside=2;
 cdside=2;
 daside=2;
end

else %circular cross-section waveguide

if rmfg==1
```

```
 fprintf('\n\n')
 disp(' Select model for calculation')
 disp('  1.full model ')
 disp('  2.quater model ')
 fufg=input(' input your selection >> ');
end

if fufg==1
 if rmfg==1
  fprintf('\n\n')
  disp(' Select wall of waveguide ')
  disp('  1. H wall')
  disp('  2. E wall')
  wafg=input(' input your selection >> ');
 else
  wafg=2;  %data for background mode
 end

else
 if rmfg==1
  fprintf('\n\n')
  disp('A')
  disp('**')
  disp('*  * ')
  disp('*    * ')
  disp('*      *')
  disp('*       *')
  disp('*         *')
  disp('*           *')
  disp('*             * ')
```

```
disp('* * * * * * * * * ')
disp('C            B')

fprintf('\n')
disp(' Impose boundary condition')
disp(' Enter  1.for H wall')
disp('        2.for E wall')
disp('        3.for not inpose anything')
fprintf('\n')

while 1
 abside =input('  AB side >> ');
 if abside==1|abside==2|abside==3,break,end
end

while 1
 bcside=input('  BC side >> ');
 if bcside==1|bcside==2|bcside==3,break,end
end

while 1
 caside=input('  CA side >> ');
 if caside==1|caside==2|caside==3,break,end
end

else  %data for background mode
 abside=1;
 bcside=2;
 caside=2;
end
```

```
 end %full model

end %cwfg

if rmfg==1
 while 1
  fprintf('\n\n')
  disp(' select choice when finish work')
  disp('  1. quit matlab')
  disp('  2. stay at prompt')
  qfg=input('input your selection >> ');
  if (qfg==1)|(qfg==2),break,end
 end
else
 qfg=2;
end

if rmfg==1
 fprintf('\n\n')
 span=input(' input order of trial function >> ');
else
 span=1;
end

if rmfg==1
 while 1
 fprintf('\n\n')
 fprintf(' select mesh pattern. \n')
 fprintf('  1. type A \n')
```

```
 fprintf('  2. type B \n')
 mhfg=input(' input your selection >> ');
 if (mhfg==1)|(mhfg==2),break,end
end
else
 mhfg=1;  %data for background mode
end

if rmfg==1
 fprintf('\n\n')
 disp(' Do you want to plot field pattern ')
 disp('  1. yes ')
 disp('  2. no  ')
 ptnfg=input(' select >> ');
else
 ptnfg=2;
 % background mode
end

if rmfg==1
 fprintf('\n\n')
 disp(' DO you want to see used memory ')
 disp('  1. yes ')
 disp('  2. no ')
 sefg=input(' select >> ');
else
 sefg=2;
end

if rmfg==1
```

```
 fprintf('\n\n')
 inl=input(' number of interval >> ');
else
 inl=1;
end

if rmfg==1
 for loop1=1:inl
  fprintf('\n\n')
  if (k0fg==1)
   fprintf(' input min t-normalized beta of interval %g ',loop1)
   min(loop1)= input(' >> ');
   fprintf(' input max t-normalized beta of interval %g ',loop1')
   max(loop1)=input(' >> ');
  else
   fprintf(' input min t-normalized freq. of interval %g ',loop1)
   min(loop1)=input(' >> ');
   fprintf(' input max t-normalized freq. of interval %g ',loop1')
   max(loop1)=input(' >> ');
  end
  istep(loop1)=input(' input step >> ');
 end
else
 min(1)=2.81;
 max(1)=13.4;
 istep(1)=(13.4-2.81)/48;
end

temp96=0;
for loop1=1:inl
```

```
temp99(loop1)=((max(loop1)-min(loop1))/istep(loop1))+1;
temp1=temp99(loop1);
temp99(loop1)=round(temp1);
%number of input data in each interval

temp96=temp96+temp99(loop1);
 %total number of input interval

if loop1==1
 temp95(loop1)=0;
else
 temp95(loop1)=temp95(loop1-1)+temp99(loop1-1);
 %offset of each interval
 temp1=temp95(loop1);
 temp95(loop1)=round(temp1);
end
clear temp1;
end

if cwfg==2
 save infor.mat k0fg lifg abside bcside cdside daside qfg span...
            inl min max istep dofg disfg temp95 temp96 ...
            temp99 mhfg mhtnfg ehtnfg cwfg ptnfg sefg

else
 if fufg==1
 save infor.mat k0fg lifg qfg span ...
            inl min max istep dofg disfg temp95 temp96 ...
            temp99 mhfg mhtnfg ehtnfg cwfg ptnfg sefg fufg wafg
```

```
 else
  save infor.mat k0fg lifg abside bcside caside qfg span ...
              inl min max istep dofg disfg temp95 temp96 ...
              temp99 mhfg mhtnfg ehtnfg cwfg ptnfg sefg
 end
 end

 %**************************
 %initaulize calculating time
 %**************************
 time4=0;
 time5=0;
 time9=0;
 time18=0;
 time11=0;
 time12=0;
 time14=0;

 if cwfg==2  %rectangular waveguide
  temp1=0;
  time8=0;
  time10=0;
  time3=0;
  time2=0;
 else  %circular cross-section waveguide
  time21=0;
  time28=0;
  time23=0;
  time22=0;
```

```
end

flag99=0;

if cwfg==2  %rectangular waveguide

 if (span==1)
  labg1  %mapping
  if disfg==2  %non-dispersive
   labg8  %initualize permeability and permittivity to each element
   labg10  %find node touching interface
  end
  labg3  %find node touching H wall and E wall of waveguide
 else
  labga1  %mapping (second order)
  if disfg==2  %non-dispersive
   labg8  %initualize permeability and permittivity to each element
   labga10  %find node touching interface
  end
  labga3  %find node touching H wall and E wall of waveguide
 end

else  %circular cross-section waveguide

 if fufg==1
  labg41
 else
  labg21  %mapping
 end
```

```
if disfg==2
 if fufg==1
  labg48
 else
  labg28  %initualize permeability and epsilon
 end
end

if fufg==1
 labg43
else
 labg23  %find node touching H wall and E wall
end

end

for loop98=1:inl
loop99=1;

while (loop99<=temp99(loop98))
 temp98=(loop99-1)*istep(loop98)+min(loop98);
 acomp=(temp95(loop98)+loop99)*100/temp96;

 if cwfg==2  %rectangular waveguide

 if (span==1)
  if disfg==1  %dispersive
   tic
   domain2
   timed2=toc;
```

```
 labg8  %initualize permeability and permittivity to each element
 labg10  %find node touching interface
end
if k0fg==1
labg2m  %form wave and divergence matrix
else
 labg2
end

else
 if disfg==1  %dispersive
 tic
 domain2
 timed2=toc;
 labg8  %initualize permeability and permittivity to each element
 labga10  %find node touching interface
end
 labga2  %second order
end

else  %circular cross-section waveguide

if disfg==1
 tic
 domain2
 timed2=toc

if fufg==1
 labg48
else
```

```
  labg28
  end
 end

 if k0fg==1
  if fufg==1
   labg42m
  else
   labg22m
  end
 else
  labg22
 end

end %cwfg==2

if k0fg==1
 labg11m  %impose B.C. to each matrix
 labg12  %form eigen  matrix where k0 is eigenvalue
else
 labg11
 labg14
end

if ptnfg==1  %plot field pattern
 labg9
 labg18
 labg19
 labg20
else
```

```
 labg4
 labg5
end

if flag99==0
 labg13
 flag99=1;
end

 loop99=loop99+1;
end

end

clear
save finish.mat
load infor.mat

if (qfg==1)
 quit
end
```