Course 2
16.532.031 Computational Electromagnetics

Prerequisites: undergraduate course on Engineering Electromagnetics or equivalent

The required textbook is PART III of the book *Electromagnetic Waves, Materials and Computation with MATLAB*, written by the instructor, to be published (August15, 2011) by CRC Press, Taylor and Francis Group.

The text book uses the math. software MATLAB in coding the algorithms developed. There are many examples of the MATLAB code in the text for writing the main programs. There are five function MATLAB programs written by the instructor, whose soft copies will be supplied by the instructor to the registrants to facilitate doing the homework and a project. In case you are not familiar with MATLAB, the last reference is a 'easy to learn', MATLAB reference. You do not have to use the MATLAB in the first 3 weeks. The homework is given so that you can do all of it it by hand computation.

Homework Problems are in the text book at the end and are listed as PX.x, X being the chapter number and x is the serial number of the problem.. The course outline given below lists the syllabus for the whole course. The third column gives the relevant Sections of the hard copy for the topics listed in Column 2. The last column gives the homework assignment.

Course Outline

16.532. Computational Electromagnetics Spring 2011

Required Text Book . *Electromagnetic Waves, Materials, and Computation with MATLAB* , by Professor D. K. Kalluri published in August 2011 by CRC Press, Taylor and Francis Group.

Wk#	Date	Topic	Section	Homework
1	01/24/2011	Formulation of E.M. Problems as Differential and Integral Equations, Finite Difference Methods, Method of Weighted Residuals	15.1 - 15.3.2	P15.2 - P15.5
2	01/31/2011	Moment Method, Finite Element Method and shape functions	15.3.3 - 15.3.5	P15.8 – P15.10
3	02/07/2011	Two Dimensional Problems: Finite Difference Method, Finite Element Method (FEM)	16.1 - 16.2	P16.1
4	02/14/2011	FEM for Poisson's and Laplace's Equations	16.2.1 - 16.3	P16.5
5	02/21/2011	FEM for Homogeneous Waveguide	16.4	P16.4

6	02/28/2011	First Examination		
7	03/072011	FEM: second order node-based, Vector finite element techniques and edge-based shape functions	16.4.1 - 16.4.3	P16.6, P16.7
8	03/14/2011	Application of vector FEM to Homogeneous Waveguide: Characteristic impedance of Transmission Lines	16.4.4 - 16.5	P16.9,P16.10
9	03/21/2011	Moment Method, Capacitance of parallel plates taking into account fringing	16.6	P16.12
10	03/28/2011	FEM: Inhomogeneous waveguide problem; 3D-triangular elements	17.1 - 17.3.1, 17.5	Project 1
11	04/04/2011	Survey of FEM Case studies and Appendices of Chapter 16	Chapter 18, 16A - E	Project 2
12	04/11/2011	Finite Difference Time-Domain (FDTD) Method	19.1 – 19.5	P19.3, P19.4
13	04/18/2011	Continuation of FDTD and Survey of FDTD Case study	19.7 – 19.10, Chapter 20	Work on Projects
14	04/25/2011	Final Examination		
15	05/02/2011	Submit Projects		

Numerical Grades

There are a total of 100 points possible in the course, distributed as follows.

5 Quizes-openbook	10%
Project 1	15%
Project 2	15%
First Exam-openbook	30%
Final Exam-openbook	30%

Problems suitable for Short Quizes
P15.1, P16.2, P16.11, P16.13, P17.1

Problems suitable for Exams.
P15.6, P15.7, P16.3, P16.8, P16.14, P17.2, P19.1, P19.2

Reference Books
The text book is sufficient for the course. For completeness some references are given below.

[1] Harrington, R. F., *Field Computation by Moment Methods*, Macmillan, New York, 1968.

[2] Jin Jianming, *The Finite Element Method in Electromagnetics*, Second Edition, Wiley, New York, 2002.

[3] Volakis, J. L., Chatterjee, A., and Kempel, L. C., *Finite Element Method for Electromagnetics*, IEEE Press, New York, 1998.

[4] Taflove, A., *Computational Electrodynamics, The Finite_Difference Time_Domain, Method*, Artech House, 1995.

[5] Sadiku, N. O. M., *Numerical Techniques in Electromagnetics*, CRC Press, 1992.

[6] Pratap, R., *Getting Started with MATLAB 5, A Quick Introduction for Scientists and Engineers*, Oxford University Press, Oxford, 1999.

Undergraduate Books
Applied Electromagnetics by Ulaby
Electromagnetic Waves by U. S. Inan & A. S. Inan

Electronic copy of the 5 function programs

% GLANT.m

```
function [S,T] = GLANT(Nn,Ne,n1L,n2L,n3L,xn,yn);
% Global Assembly two dimensional node based
% triangular elements
for e = 1:Ne;
   n(1,e) = n1L(e);
```

```
    n(2,e) = n2L(e);
    n(3,e) = n3L(e);
end

% Initialization
S = zeros(Nn,Nn);
T = zeros(Nn,Nn);
% Loop through all elements
for e = 1: Ne;
    % coordinates of the element nodes
    for i = 1:3;
        x(i) = xn(n(i,e));
        y(i) = yn(n(i,e));
    end
    % compute the element matrix entries
    b(1) = y(2) - y(3);
    b(2) = y(3) - y(1);
    b(3) = y(1) - y(2);
    c(1) = x(3) - x(2);
    c(2) = x(1) - x(3);
    c(3) = x(2) - x(1);
    Area = 0.5 * abs ( b(2)*c(3) - b(3) *c(2));
    % Compute the elemnt matrix entries
    for i = 1:3;
```

```
      for j = 1:3;
         Se(i,j) = (0.25/Area)*(b(i)*b(j) + c(i) *c(j));
         if i ==j
            Te(i,j) = Area/6;
         else
            Te(i,j) = Area/12;
         end
 % Assemble the Element matrices into Global FEM System
         S(n(i,e),n(j,e)) = S(n(i,e),n(j,e))+ Se(i,j);
         T(n(i,e),n(j,e)) = T(n(i,e),n(j,e))+ Te(i,j);
      end
   end
end
```

% PGLANT2.m

```
function [S,T,g] =
PGLANT2(Nn,Ne,n1L,n2L,n3L,Rho,Epr,xn,yn);
% Solution of Poisson's Equation
% Laplacian of V = -Rho/(epsilon0*epr)
% Rho is the array of values of volume charge density in
each element
% Epr is the array of values of dielectric constant in each
element
% Global Assembly two dimensional node based
```

```
% triangular elements
for e = 1:Ne;
   n(1,e) = n1L(e);
   n(2,e) = n2L(e);
   n(3,e) = n3L(e);
end

% Initialization
S = zeros(Nn,Nn);
T = zeros(Nn,Nn);
g = zeros(Nn);
% Loop through all elements
for e = 1: Ne;
   % coordinates of the element nodes
   for i = 1:3;
      x(i) = xn(n(i,e));
      y(i) = yn(n(i,e));
   end
   % compute the element matrix entries
   b(1) = y(2) - y(3);
   b(2) = y(3) - y(1);
   b(3) = y(1) - y(2);
   c(1) = x(3) - x(2);
   c(2) = x(1) - x(3);
```

```
   c(3) = x(2) - x(1);
   Area = 0.5 * abs ( b(2)*c(3) - b(3) *c(2));
   % Compute the elemnt matrix entries
   ge = Rho(e)*Area/3;
   for i = 1:3;
          g(n(i,e)) = g(n(i,e)) + ge;
      for j = 1:3;
         Se(i,j) = (0.25/Area)*(b(i)*b(j) + c(i)
*c(j))*Epr(e)*8.854*10^(-12);
         if i ==j
            Te(i,j) = Area/6;
         else
            Te(i,j) = Area/12;
         end
 % Assemble the Element matrices into Global FEM System
         S(n(i,e),n(j,e)) = S(n(i,e),n(j,e))+ Se(i,j);
         T(n(i,e),n(j,e)) = T(n(i,e),n(j,e))+ Te(i,j);
      end
   end
end
```

% GLANT2T.m

```
function [S,T] =
GLAN2T(Nn,Ne,n1L,n2L,n3L,n4L,n5L,n6L,xn,yn);
% Global Assembly two dimensional node based
% triangular elements of second order
% written by D. K. Kalluri
for e = 1:Ne;
   n(1,e) = n1L(e);
   n(2,e) = n2L(e);
   n(3,e) = n3L(e);
   n(4,e) = n4L(e);
   n(5,e) = n5L(e);
   n(6,e) = n6L(e);
end
Q1 =  (1/6)*[0,0,0,0,0,0;0,8,-8,0,0,0;0,-8,8,0,0,0;
   0,0,0,3,-4,1;0,0,0,-4,8,-4;0,0,0,1,-4,3];
Q2 = (1/6)*[3,0,-4,0,0,1;0,8,0,0,-8,0;-4,0,8,0,0,-4;
   0,0,0,0,0,0;0,-8,0,0,8,0;1,0,-4,0,0,3];
Q3 = (1/6)*[3,-4,0,1,0,0;-4,8,0,-4,0,0;0,0,8,0,-8,0;
   1,-4,0,3,0,0;0,0,-8,0,8,0;0,0,0,0,0,0];
Tek = (1/180)*[6,0,0,-1,-4,-1;0,32,16,0,16,-4;0,16,32,-
4,16,0;
   -1,0,-4,6,0,-1;-4,16,16,0,32,0;-1,-4,0,-1,0,6];
% Initialization
S = zeros(Nn,Nn);
```

```
T = zeros(Nn,Nn);
% Loop through all elements
for e = 1: Ne;
   % coordinates of the element nodes
   x(1) = xn(n(1,e));
   x(2) = xn(n(4,e));
   x(3) = xn(n(6,e));
   y(1) = yn(n(1,e));
   y(2) = yn(n(4,e));
   y(3) = yn(n(6,e));

% compute the element matrix entries
   b(1) = y(2) - y(3);
   b(2) = y(3) - y(1);
   b(3) = y(1) - y(2);
   c(1) = x(3) - x(2);
   c(2) = x(1) - x(3);
   c(3) = x(2) - x(1);
   Area = 0.5 * abs ( b(2)*c(3) - b(3) *c(2));
   COTTH1 = -(0.5/Area)*(b(2)*b(3) + c(2)*c(3));
   COTTH2 = -(0.5/Area)*(b(3)*b(1) + c(3)*c(1));
   COTTH3 = -(0.5/Area)*(b(1)*b(2) + c(1)*c(2));
   Te = Area*Tek;
   Se = COTTH1*Q1 + COTTH2*Q2 + COTTH3*Q3;
```

```
    % Compute the elemnt matrix entries
    for i = 1:6;
        for j = 1:6;
            % Assemble the Element matrices into Global FEM
System
            S(n(i,e),n(j,e)) = S(n(i,e),n(j,e))+ Se(i,j);
            T(n(i,e),n(j,e)) = T(n(i,e),n(j,e))+ Te(i,j);
        end
    end
end
```

```
% GLAET.M
function [E,F] = glaet(Neg, Nn, Ne, n1L, n2L, n3L, n1EL,
n2EL, n3EL, xn, yn)
for e = 1:Ne
    n(1,e) = n1L(e);
    n(2,e) = n2L(e);
    n(3,e) = n3L(e);
    ne(1,e) = n1EL(e);
    ne(2,e) = n2EL(e);
    ne(3,e) = n3EL(e);
end

E=zeros(Neg, Neg);
```

```
F=zeros(Neg, Neg);


for e = 1:Ne
   for i = 1:3;
      x(i) = xn(n(i,e));
      y(i) = yn(n(i,e));
   end


   b(1)=y(2)-y(3);
   b(2)=y(3)-y(1);
   b(3)=y(1)-y(2);
   c(1)=x(3)-x(2);
   c(2)=x(1)-x(3);
   c(3)=x(2)-x(1);


   Area=0.5*abs(b(2)*c(3)-b(3)*c(2));
   l(1)=sqrt(b(3)*b(3)+c(3)*c(3));
   l(2)=sqrt(b(1)*b(1)+c(1)*c(1));
   l(3)=sqrt(b(2)*b(2)+c(2)*c(2));


   for i=1:3
      for j = 1:3
          ff(i,j)=b(i)*b(j)+c(i)*c(j);
     end
```

```
end

G(1,1)=2*(ff(2,2)-ff(1,2)+ff(1,1));
G(2,2)=2*(ff(3,3)-ff(2,3)+ff(2,2));
G(3,3)=2*(ff(1,1)-ff(3,1)+ff(3,3));
G(2,1)=G(1,2);
G(1,3)=ff(2,1)-2*ff(2,3)-ff(1,1)+ff(1,3);
G(3,1)=G(1,3);
G(2,3)=ff(3,1)-ff(3,3)-2*ff(2,1)+ff(2,3);
G(3,2)=G(2,3);

for i=1:3
   for j=1:3
       Ee(i,j)=(1/Area)*(l(i)*l(j));
       Fe(i,j)=(1/48)*(1/Area)*l(i)*l(j)*G(i,j);
       if(ne(i,e)<0)
       Ee(i,j)=-Ee(i,j);
       Fe(i,j)=-Fe(i,j);
   else;
   end;

   if(ne(j,e)<0);
       Ee(i,j)=-Ee(i,j);
       Fe(i,j)=-Fe(i,j);
```

```
      else;
      end;

      ane(i,e)=abs(ne(i,e));
      ane(j,e)=abs(ne(j,e));
      E(ane(i,e),ane(j,e))=E(ane(i,e),ane(j,e))+Ee(i,j);
      F(ane(i,e),ane(j,e))=F(ane(i,e),ane(j,e))+Fe(i,j);
    end

  end
end
```

% INHWGD.m

```
function
[Att,Btt,Btz,Bzz,C]=INHWGD(epr,mur,k0,Neg,Nn,Ne,...
   n1L,n2L,n3L,n1EL,n2EL,n3EL,xn,yn);
% Inhomogeneous waveguide problem, page 73-75 text
% written by D. K. Kalluri
% Edge elements for transverse fields and node based
%   elements for longitudinal fields
% Notation of text modified to remove confusion
%       Text( Equation 3.39 )           My notation
%       [St]                                Att
%       [Tt]                                Btt
```

```
%       [Sz]                                        Bzz
%       [G]                                     Btz
% Global Assembly of two dimensional edge based
% triangular elements
% Neg= number of edge elements
% Nn = number of nodes
% Ne = number of elements
% n1L = array containing global node number
%           of the first local node of e th elemnt
% n2L = array containing global node number
%           of the second local node of e th elemnt
% % n3L = array containing global node number
%           of the third local node of e th elemnt

% n1EL = array containing global edge number
%           of the first local edge of e th elemnt
% n2EL = array containing global edge number
%           of the first local edge of e th elemnt
%  n3EL = array containing global edge number
%           of the first local edge of e th elemnt
% a negative value indicates opposite directions
%   of the global edge and the local edge
for e = 1:Ne;
   n(1,e) = n1L(e);
```

```
    n(2,e) = n2L(e);
    n(3,e) = n3L(e);
    ne(1,e) = n1EL(e);
    ne(2,e) = n2EL(e);
    ne(3,e) = n3EL(e);
end
% Initialization
Att = zeros(Neg,Neg);
Btt = zeros(Neg,Neg);
Btz = zeros(Neg,Nn);
Bzz = zeros(Nn,Nn);
% Loop through all elements
for e = 1: Ne;
    % coordinates of the element nodes
    for i = 1:3;
        x(i) = xn(n(i,e));
        y(i) = yn(n(i,e));
    end
    % compute the element matrix entries
    b(1) = y(2) - y(3);
    b(2) = y(3) - y(1);
    b(3) = y(1) - y(2);
    c(1) = x(3) - x(2);
    c(2) = x(1) - x(3);
```

```
c(3) = x(2) - x(1);
Area = 0.5 * abs ( b(2)*c(3) - b(3) *c(2));
l(1) = sqrt(b(3)*b(3) + c(3)*c(3));
l(2) = sqrt(b(1)*b(1) + c(1)*c(1));
l(3) = sqrt(b(2)*b(2) + c(2)*c(2));
K(1) = l(1)/(12*Area);
K(2) = l(2)/(12*Area);
 K(3) = l(3)/(12*Area);
% Compute the elemnt matrix entries
for i = 1:3;
   for j = 1:3;
      ff(i,j) = b(i)*b(j) + c(i)*c(j);
      Se(i,j) = (0.25/Area)*(b(i)*b(j) + c(i) *c(j));
      if i ==j
         Te(i,j) =Area/6;
      else
         Te(i,j) =Area/12;
      end
   end;
end;
G(1,1) = 2*(ff(2,2) - ff(1,2)+ ff(1,1));
G(2,2) = 2*(ff(3,3) - ff(2,3)+ ff(2,2));
G(3,3) = 2*(ff(1,1) - ff(3,1) + ff(3,3));
G(1,2) = ff(2,3) - ff(2,2) - 2*ff(1,3) +ff(1,2);
```

```
    G(2,1) = G(1,2);
    G(1,3) = ff(2,1) - 2*ff(2,3)- ff(1,1) +ff(1,3);
    G(3,1) = G(1,3);
    G(2,3) = ff(3,1) - ff(3,3) - 2*ff(2,1)+ ff(2,3);
    G(3,2) = G(2,3);
    Gtz(1,1) = -K(1)*(b(1)*(b(1)-b(2))+c(1)*(c(1) -c(2)));
    Gtz(2,2) = -K(2)*(b(2)*(b(2)-b(3))+c(2)*(c(2) -c(3)));
    Gtz(3,3) = -K(3)*(b(3)*(b(3)-b(1))+c(3)*(c(3) -c(1)));
    Gtz(1,2) =K(1)*(b(2)*(b(2)-b(1))+c(2)*(c(2) -c(1)));
    Gtz(2,3) =K(2)*(b(3)*(b(3)-b(2))+c(3)*(c(3) -c(2)));
     Gtz(3,1) =K(3)*(b(1)*(b(1)-b(3))+c(1)*(c(1) -c(3)));
    Gtz(1,3) = -K(1)*(-y(1)*b(2)-y(2)*b(1)-y(3)*b(3)...
                  +x(1)*c(2)+x(2)*c(1)+x(3)*c(3));
    Gtz(2,1) = -K(2)*(-y(2)*b(3)-y(3)*b(2)-y(1)*b(1)...
                  +x(2)*c(3)+x(3)*c(2)+x(1)*c(1));
     Gtz(3,2) = -K(3)*(-y(3)*b(1)-y(1)*b(3)-y(2)*b(2)...
                  +x(3)*c(1)+x(1)*c(3)+x(2)*c(2));
     for i = 1:3;
       for j = 1:3;
          Ee(i,j) = (1/Area)*(l(i)*l(j));
          Fe(i,j) = (1/48)*(1/Area)*l(i)*l(j)*G(i,j);
             % Assemble the Element matrices into Global FEM
System
           if (ne(i,e) < 0);
```

```
          Ee(i,j) = - Ee(i,j);
          Fe(i,j) = -Fe(i,j);
          Gtz(i,j) = -Gtz(i,j);
          else;
          end;
        if (ne(j,e) < 0);
          Ee(i,j) = - Ee(i,j);
          Fe(i,j) = -Fe(i,j);
          else;
          end;
          Atte(i,j) = (1/mur(e))*Ee(i,j) -
k0*k0*epr(e)*Fe(i,j);
          Btte(i,j) = (1/mur(e))*Fe(i,j);
          Btze(i,j) = (1/mur(e))*Gtz(i,j);
          Bzze(i,j) = (1/mur(e))*Se(i,j) -
k0*k0*epr(e)*Te(i,j);
          % for storing purposes take absolute value of
          % ne(i,e) and ne(j,e) wich may have negative
values
          ane(i,e) = abs(ne(i,e));
          ane(j,e) = abs(ne(j,e));
          Att(ane(i,e),ane(j,e)) = Att(ane(i,e),ane(j,e))+
Atte(i,j);
```

```
          Btt(ane(i,e),ane(j,e)) = Btt(ane(i,e),ane(j,e))+
Btte(i,j);
          Btz(ane(i,e),n(j,e)) = Btz(ane(i,e),n(j,e))+
Btze(i,j);
          Bzz(n(i,e),n(j,e)) = Bzz(n(i,e),n(j,e))+
Bzze(i,j);
       end
    end
end
C1 = Btt - Btz*inv(Bzz)*Btz';
C2 = inv(C1);
C = C2*Att;
```