Laboratory Sessions : FreeFem++

Preschool CIMPA Workshop

June 24^{th} – July 4^{th} 2015

IIT Mumbai

IIT Mumbai 400 076

Abstract

The manual will guide you through the session and with some explict step by step instructions to follow during the FreeFem++ Laboratory Sessions.

The Main objective: there is a definitive objective for each session and the Laboratory Sessions will synchronise with the theory being covered in class, e.g, we start with steady state linear problems (elliptic), move to transient simulations (parabolic heat equation), implement the mixed method (with the linear Stokes problem) and end with sessions on the Navier-Stokes equation. Steady State Nonlinear problems and the Adaptive meshing algorithms will also be covered.

The manual is a rich source of useful information, the key is that there are a *huge* number of examples. We learn by *spending time* in looking carefully at these illuminating examples. The manual and some useful documents are available in *documents* folder.

Contents

1 Session 1: Steady State Problem: FEM for Elliptic PDE, e.g., Laplace and Poisson

In this session, we pick up the simplest problem in FEM; an elliptic PDE in two dimensions and also learn how the standard algorithmic steps in a FEM solution are executed with FreeFem++.

§ The complete program to solve an planar Elliptic PDE.

§ Simple exercises to map the program to the FEM algorithm.

§ A few more exercises.

We will solve,

$$\begin{aligned} -\Delta u(x,y) &= f(x,y) \ \ \forall (x,y) \in \Omega, \\ u(x,y) &= 0 \ \ \forall (x,y) \ \text{ on } \ \partial\Omega \end{aligned}$$

The statement of the weak form of the problem after discretization is,
Find $u_h(x,y)$ such that,

$$\int_{T_h} \left(\frac{\partial u_h}{\partial x}\frac{\partial v_h}{\partial x} + \frac{\partial u_h}{\partial y}\frac{\partial v_h}{\partial y} \right) = \int_{T_h} f v_h \tag{1.1}$$

where u_h and v_h belong to suitable finite dimensional subspaces of suitable Hilbert spaces.

Let us see the program in FreeFem++ for this Poisson problem on a circular domain with $f(x,y) = x \times y$

1.1 Getting Started

1.1.1 Execute the complete program in FreeFem++

The program is given below. Copy it

1. Onto your command window.
2. Save it in the Directory *"Contents/Resources/examples/chapt3" as "myfirstprogram.edp"*.
3. Execute it.

```
// defining the boundary
border C(t=0,2*pi){x=cos(t); y=sin(t);}
// the triangulated domain Th is on the left side of its boundar
mesh Th = buildmesh (C(50));
// the finite element space defined over Th is called here Vh
fespace Vh(Th,P1);
Vh u,v; // defines u and v as piecewise-P1 continuous functions
func f= x*y; // definition of a called f function
real cpu=clock(); // get the clock in second
solve Poisson(u,v,solver=LU) = // defines the PDE
int2d(Th)(dx(u)*dx(v) + dy(u)*dy(v)) // bilinear part
- int2d(Th)( f*v) // right hand side
+ on(C,u=0) ; // Dirichlet boundary condition
plot(u);
```

1.1.2 What is happening behind these innocent looking lines of code?

1. We can visualise some of these lines. Study the following lines of code.

```
bool debug = true;
border a(t=0,2*pi){x=cos(t); y=sin(t); label=1;}
border b(t=0,2*pi){x=0.3+0.3*cos(t); y=0.3*sin(t); label=1;}
plot(a(50)+b(-30),wait=debug);//plot the borders to see the intersection
//(so change the 0.8 to say 0.3 and then a mouse click
mesh Th= buildmesh(a(50)+b(-30));
```

```
plot(Th, wait=debug); // Plot Th then needs a mouse click
fespace Vh(Th,P2);
Vh f = sin(pi*x)*cos(pi*y);
plot(f,wait=debug);
Vh g=sin(pi*x+cos(pi*y));
plot(g,wait=debug);
```

2. Now *insert suitably* the lines of code required to *visually* understand the first few lines of the code of "myfirstprogram.edp". You should re-open "myfirstprogram.edp"

3. The next few lines is nothing but the weak form (given in eq(1.1) above) of this particular problem with the Dirichlet boundary condition. Our background in FEM clearly tells us what happens when the program executes this line i.e, it obtains suitably the linear form finite dimensional $Au = b$ and solves (we have chosen the LU solver amongst the many available) it for the piecewise linear u (P_1 finite elements).

4. The final line is just a visual plot of the contours of the isovalues of $u(x, y)$.

5. Open the program *"Contents/Resources/examples/ffcs/p2hat.edp"*. Now add suitable lines to visualize the function $f = x \times y$ in "myprogram.edp". Do the same if the problem is solved with P_2 finite elements. Visualize the solution $u(x, y)$ in the same way (i.e., in 3-d).

1.1.3 Some quick really simple variations

1. Change the domain to an ellipse.

2. Half the mesh size.

3. Try a piecewise quadratic approximation.

1.2 Moving ahead

- Load the file "membrane.edp" from "examples/chap3". Execute it. *What is the boundary condition in this problem*
- Load the file "membranerror.edp" from "examples/chap3". Execute it. *What is the boundary condition in this problem?*.

1.3 Some Illustrative Domains and meshings

Visualise the following illustrative code snippets related to domains and meshing.

```
real x0=1.2,x1=1.8;
real y0=0,y1=1;
int n=5,m=20;
//meshing with different flags. check pg-104 of manual for details
for (int i=0;i<5;++i)
{
int[int] labs=[11,12,13,14];
mesh Th=square(3,3,flags=i,label=labs,region=10);
plot(Th,wait=1,cmm=" square flags = "+i );
}
```

1.4 Some Illustrative Visualizations

1.4.1 Holes

```
border a(t=0,2*pi){ x=cos(t); y=sin(t);label=1;}
border b(t=0,2*pi){ x=0.3+0.3*cos(t); y=0.3*sin(t);label=2;}
plot(a(50)+b(+30)) ;//to see a plot of the border mesh
mesh Thwithouthole= buildmesh(a(50)+b(+30));
mesh Thwithhole= buildmesh(a(50)+b(-30));
plot(Thwithouthole,wait=1,ps="Thwithouthole.eps");
plot(Thwithhole,wait=1,ps="Thwithhole.eps");
/*ps="fileName" is used to generate a postscript file with identification shown on the figure.*/
```

1.4.2 rectangular boundary

```
real x0=1.2,x1=1.8;
real y0=0,y1=1;
int n=5,m=20;
//defining a rectangle with dimensions 5*20 with specified coordinates
mesh Th=square(n,m,[x0+(x1-x0)*x,y0+(y1-y0)*y]);
plot(Th,cmm="a");
//cmm will add the comment to the fig
```

1.4.3 Multiple boundaries

```
int upper = 1;
int others = 2;
int inner = 3;
border C01(t=0,1){x=0;y=-1+t;label=upper;}
border C02(t=0,1){x=1.5-1.5*t;y=-1;label=upper;}
border C03(t=0,1){x=1.5;y=-t;label=upper;}
border C04(t=0,1){x=1+0.5*t;y=0;label=others;}
border C05(t=0,1){x=0.5+0.5*t;y=0;label=others;}
border C06(t=0,1){x=0.5*t;y=0;label=others;}
border C11(t=0,1){x=0.5;y=-0.5*t;label=inner;}
border C12(t=0,1){x=0.5+0.5*t;y=-0.5;label=inner;}
border C13(t=0,1){x = 1;y = -0.5+0.5*t;label = inner;}
int n = 10;
plot(C01(-n)+C02(-n)+C03(-n)+C04(-n)+C05(-n)+C06(-n)+
C11(n)+C12(n)+C13(n), wait=true);
mesh Th = buildmesh(C01(-n)+C02(-n)+C03(-n)+C04(-n)+C05(-n)+C06(-n)+
C11(n)+C12(n)+C13(n));
plot(Th, wait=true);
cout << "Part 1 has region number " << Th(0.75, -0.25).region << endl;
cout << "Part 2 has redion number " << Th(0.25, -0.25).region << endl;
```

1.5 Homework

Essentially there is a lot of useful information in the manual and also in the huge set of *illuminating illustrative* examples. So, please check these key features which will play a ubiquitous role

1.5.1 Finite elements available

Check sections *6.1-6.6*in the manual which is list of finite elements available.

1.5.2 Extracting the matrices and other relevant data important for you

FreeFem++ has two important applications of the keyword *varf*. See section *6.12* for detailed explanations and search this word throughout the manual for several examples.

As a first example you can check by executing the file "test1.edp" from "examples/chap3".

1.5.3 Visualization

See Chapter *7* of the manual.

As a first example you must check a nice demo file "options.edp" from "examples/ffcs".

1.5.4 Parameters affecting solve and problem

See section *6.9* for all the details.

1.5.5 The func operator

The "func" operator is fairly intuitive. See also "func.edp" from "examples/tutorial"

1.5.6 The *convect* Operator

The "convect" will play a central role in the algorithms for *transient* simulation of Stokes and Navier-Stokes. Look it up in both the manual and the examples. *Now you now how to navigate through the wonderful manual and stock of great examples provided by the architects of FreeFem++ !!!! :) :)*