A Quick Intro to Femlab

Roger Thelwell

thelwell@math.colostate.edu

Sept 16, 2003

1 Introduction

Femlab is a front-end for Matlab, and allow multi-physics PDEs to be solved using finite element methods. Femlab can be controlled through a GUI and/or a script.m file.

```
version =
     name: 'FEMLAB 2.3'
     major: 0
     build: 145
     rcs: '$Name:  $'
     date: '$Date: 2002/06/10 19:39:20 $'
```

There are newer versions available for down-load.

2 Getting started

The best way to start learning Femlab is through the manuals (User's Guide , Reference, and the Model Library) - available in both print and electronic format. Femlab 's help has been installed on Reynolds and can reached by through the help button in the Femlab control window. You'll have access to htm and pdf files. There's also online help:

```
http://www.femlab.com/
```

In addition, the demos are a great resource. You can load and run these, view all setting and modify them.

You'll need to spend some time playing with Femlab - I've been at it for several weeks now. It's powerful, but sometimes hard to control. There are many options to set. These notes are just a few comments that I would have found helpful. I'll focus on the GUI application, which makes it pretty easy to build models. I haven't yet learned much about scripting, but I think that scripts are really the best way to manage experiments.

2.1 GUI

The GUI is a good place to start with Femlab . You might want to step through some of the examples in the User's Guide. You'll pick up quite a bit by doing so. You can call the GUI from the Matlab prompt

```
matlab>> femlab
```

Two windows should appear - the `Femlab` window and the `Model Navigator` window. New models can be started by clicking the `New` tab inside the `Model Navigator` window, selecting the dimension of the problem and a general format to use. These formats are:

- `Geometry only`
- `Physics modes`
- `Weak modes`
- `Classical PDEs`
- `Applications`

Double clicking allows you to enter each and make more specific choices. The `Geometry only` can be used to draw the domain. This domain can then be saved as a geometry file, to be reloaded at any time. This is useful when a complex geometry might be used again.

The `Physics modes` section includes a wide range of common PDEs. The coefficients are identified by their usual physical terminology. This makes it easy to apply these models to engineering applications. It's more difficult to incorporate additional features to these prebuilt forms.

The `PDE mode` section lets you build in either a coefficient form

$$d_a \frac{\partial u}{\partial t} + \nabla \cdot (-c\nabla u - \alpha u + \gamma) + \beta \cdot \nabla u + au = f$$

or the general form,

$$d_a \frac{\partial u}{\partial t} + \nabla \cdot \Gamma = F.$$

The general form is more flexible than the coefficient form, but the boundary terms are usually more difficult to set properly. Also be aware that sometimes you'll need to enter scalars, and other times matrices. The display format isn't the best. The `View as Coefficients` can be helpful to see the structure.

The `Model Library` tab allow access to a variety of pre-coded models. The wide range can be used as a foundation for modifications. Multi-physics demos can be found in this section as well.

The `Multiphysics` tab is used to create multi-physics models. Once the model and its variables are chosen, the are moved to the application side by double clicking or by clicking on the >> arrow button. The default variable names will be independent, but be careful if you choose your own. Each model has reserved local variable names (which you can check

at a later stage, or in the documentation) so be careful with any changes. Don't worry too much, though, as it's also possible to change these names at a later stage.

> Suggestion: Build your new model using a existing Femlab demo model. It's easy to add and modify Femlab models. This includes changing the geometry, equation coefficients, multi-physics modes, boundary and initial conditions, mesh and solvers. The default options are in general a good place to start.

2.2 Starting a New Model

We'll step though to following process to build a model:

1. Choose a demo or model
2. Draw or load geometry
3. Define the boundary setting
4. Define the Subdomain
5. Verify that the mesh is OK
6. Solve
7. Post mode to visualize the solution
8. Modify?

Be careful when changing geometry, and modifying multi-physics. Defaults are reset in mesh setting and solver tolerances when these changes are made.

2.2.1 Choosing the Model

From the `Model Navigator` window, click the `New` tab, and select the model and variables, then click OK. The default elements work well for the physics mode problems, but might need to be modified for the general/weak modes. You can open the `Model Navigator` window any time from the Femlab window by Ctrl-n or by clicking `File` and then New.

The geometry of the model needs to be established first, but once this is done the order of the others (boundary and sub-domain) really doesn't matter. I'll just step through them in the the order that they're found on the menu.

2.2.2 Geometry

The first step is to enter the geometry. Geometry can be either constructed or imported (File → Import from File → Geometry file).

To draw a new geometry, a grid needs to be established. Under Options → Axis/Grid Setting, set both the axis and grid options. Press the Auto radio button to get control of the

grid. If you think you'd like to use the snap-to-grid feature, you'll generally need to enter grid values everywhere you'd to place a vertex in the drawing.

A sidebar menu should pop up, from which you can chose shapes or choose them from the pull-down menu. The right mouse button can be used to draw squares and circles, and only requires a grid mark in one dimension.

Union(+), intersection (*), difference(-) can be used to create composite objects. For example, R1-C1 removes circle C1 from Rectangle R1.

A single click on a vertex lets you move it. Double clicking on an object will open a window that lets you modify the objects properties (vertex, name, etc). Left clicking and holding lets you move the object.

I had some problems with the draw mode, particularly in 3d. The drawing would go well, but the mesh generation would fail, producing degenerate elements. The reference manual has a list of errors that can occur (pg 3-71), and some cures. The cures for my problems were vague.

Once you're finished with the geometry, you can set up boundary and subdomain. At this point, you can also save the geometry (File → Export to File → Geometry file).

2.2.3 Boundary Mode

Once you enter boundary mode (either from the menu of by pressing Ctrl-b), you can select `boundary setting` from the pull-down menu, or double clicking on the separate boundaries in the line drawing. Several boundaries can be selected by using the shift key and clicking each, or by drawing a rectangle with the left mouse button and enclosing the boundary elements and then double right clicking. Once the `Boundary Setting` window appears, the boundary values can be set. Once entered, press OK.

You won't be able set boundary values unless your in some type of model mode. The `Geometry only` selection in the `Model Navigator` is exactly what it says, and no more. If you find you can't set boundary values, this might be the reason.

2.2.4 Subdomain

Now set the subdomain values. If you'd like to use parameters here, you can go back to Options on the menu bar and select `Add/Edit constants`. Enter any parameters that you'd like to use in the subdomain section. Also under Options, you can find `Assigned variable names`, and a list of preassigned variable names. Make sure you don't have duplicates.

Click on Subdomain from the menu bar and either double click on drawing or setting in the pull-down menu. Then use constants (or parameters) and set either coefficient values (coefficient form) or functions (in general form).

2.2.5 Mesh

Enter mesh mode, and either use the refine mesh button or the mesh parameters to create a mesh that looks good. You also have the option to turn on a adaptive mesh in the solver section, so you don't have to worry too much here. The parameter section (at the bottom of the menu) allow you to set general edge size, growth factor and curvature (effectively the

degree of the polygonal approximation on curves). You should reinitialize mesh when you're done here. You also need to be careful when you change the geometry - the mesh reverts to the default parameters. The mesh statistics button is useful to verify that things are right.

2.2.6 Solve

The solve stage has the most parameters to set, and is difficult to optimize. A general process to solve a time dependent problem might be:

- Solve time independent problem (General tab). Fix any problems with geometry, grid, and initial/boundary conditions.
- Restart the dependent for a short period of time. (General and Timestepping tabs). Explore solver tolerances, and test a few different time-dependent solvers. Also look at time dependent boundary and initial conditions.
- Restart with full length time dependent problem. (Timestepping tab)

Relative tolerence needs to be scalar, and is used to control all unknown variables. Absolute tolerence allows you to single out individual variables. For flow problems, it seems important to use both.

The initialization in nonlinear problems can be important for convergence. One way to do this is by prescribing a good initial solution, which is controlled on the `Subdomain` setting window, under the `Initial` tab. Another way is to use the restart button, and do some coarse parameter continuation to find a suitable initialization.

When solving multiphysics problems, you don't need to solve all modes simultaneously. Chose the mode(s) to solve by clicking on `Solve for variables` on the pull-down menu and then selecting. This can be useful in multiphysics initialization.

Be careful if you've changed the basic model - (ie removed mulitphysics, changed geometry, added boundary elements etc), since some changes might reset parameters. It took me several days to notice changes in the time dependent solver tolerances and in mesh parameters. These only became apparent in the long time runs.

2.2.7 Post Mode

Used for graphics. Animations can be saved as mpeg, gif or jpg. All plots are controlled from the plot parameter selection. The Mpeg format movies seem to be limited to 16 frames, and the resolution is quite poor.. The support at femlab recommended that I make a sequence of tiff format plots, and then use a free app to make mpg movies. `Resol` sets resolution of tiff format.

2.2.8 Femlab Hot-Keys

Ctrl-n	Open New model
Ctrl-s	Save model (.mat file)
Ctrl-f	Export fem to Matlab
Ctrl-b	Boundary mode
Ctrl-i	Initialize mesh
Ctrl-e	Solve
Ctrl-t	Restart
Ctrl-p	Plots (post mode)
Ctrl-w	Exit

2.3 Scripts

Femlab can also be run from `Matlab6` using a script. If you don't mind keeping Femlab running and doing everything with the GUI, you don't need to use the command language. If you'd like to run series of experiments that require you to test various aspects of the model, you might want to learn some of the basics of the command language. This language lets you call all of Femlab's commands from Matlab's command line. It also lets you run experiments from scripts and in the backround. You can actually use Femlab to create scripts automatically (which you can them modify), or they can be hand coded.

Saving your work in a .m file is a compact way to keep track of what you've done, as a list of Femlab 's command language commands. It's like a log file of your keystrokes. The commands in .m files that you capture is just serial list, so you might need to save a sequence of these files to see changes.
I used the following, a combination of GUI and the commmand language, to build experiments:

1. With Femlab 's GUI, set up the problem and make some short experiments.

2. In Femlab , use CTRL-F to export the fem struc to Matlab. (it will show up in Matlab as a structure called *fem*)

3. In Matlab, save the fem structure by using Matlab's save command

   (ie `matlab>>` save template *fem*).

4. Use Femlab 's GUI to simulate the motions of experiments, and then save these key commands in a .m file. Use the save as choice under File on the main menu bar to save a .m file.

   (ie save *sample* writes sample.m).

5. Using a text editor, write a .m script (ie *exper.m*) that loads the template file and performs the experiment. You can get the commands by looking at the saved *sample.m* files, and lift (and modify) the commands as needed. You can also look through the reference manual and pre-coded models. You might want to include a Matlab save

command in *exper.m* to save to results of the script. You can also use Matlab tic/toc to compare the run times of solvers and other options.

6. Run your script. I use *screen* to run the experiments in the backround like this:

```
reynolds> screen
reynolds> unsetenv DISPLAY
reynolds> matlab6
   matlab>> exper
   matlab>> save results
```

   screen makes it easy to check on progress, but breaking the script run prevents you from seeing current progress.

7. In `Matlab`, load your results and use the post commands to view.
   (ie `matlab>>` load results
   `matlab>>` postplot(fem))

2.3.1 Examples

A few examples of Femlab commands that might be useful in scripts:

```
% Save current fem structure for restart purposes
fem0=fem;
```

Set up the mesh again:

```
% Extend the mesh
fem.xmesh=meshextend(fem);
```

Map current solution to to the new mesh (Navier Stokes):

```
% Map initial solution to current xmesh
fem0.sol.u(:,1:end-1)=[];
umap={'p',1,1,1,1;'u',1,1,1,1;'v',1,1,1,1};
init=asseminit(fem,'init', {fem0,umap});
```

I'll use `init` from above to initialize the time dependent solve. See the reference manual pg 5-134 for all of `femtime` options - there are many.

```
% Solve dynamic problem
fem.sol = femtime(fem,'tlist',[0:.1:1],...
       'ode','fldaspk',...
       'atol',{'u' 1e-3 'v' 1e-3 'p' inf},...
       'stop' , 'on',...
       'init',init,...
       'timeind','D',...
       'report','on');
% stop on returns partial solution if timestepping fails
% timeind D, mass matrix is constant (for speed)
```

An example of post plotting. Here I plot the velocity as a colormap, pressure as a contour plot with colobar and then add flowlines. `Postplot` is worth reading about in the reference manual.

```
% Plot solution
postplot(fem,'tridata','sqrt(u^2+v^2)',...
             'contdata','p',...
             'contbar','on',...
             'flowdata',{ 'u' 'v'},...
             'title', 'Surface: Velocity (U) Contour: pressure (p)
Flow: [x velocity (u),y velocity (v)]  ',...
             'axisequal','on');
```

The stiffness matrix can be assembled and explored.

```
% View stiffness matrix
[K,L,M,N]=assemble(fem);
spy(K);
% K = Stiffness matrix , L = residual vector
% M = constraint vector , N = constraint matrix
```