Numerical Solution of PDE: Comsol Multiphysics

This is an introduction to get you acquainted with Comsol Multiphysics for numerical solutuon of PDE by finite elements. The program has many facilities and we will use only a small fraction. The documentation is available (PDF and HTML) under the help button, **Help**, rightmost on the menu bar (top). The simplest is *Quick Start*, but even that is quite comprehensive. The GUI-functions are hopefully intuitive.

Example 1, The Poisson equation on an ellipse

Start comsol. In the first screen **ModelNavigator** we choose 2D och `ApplicationModes>ComsolMultiphysics>PDEModes>ClassicalPDEs>Poisson's Equation`.

We see that the element type is **Lagrange P2**-triangles. The solution is approximated with second degree polynomials, one over each triangle, a globally C^0 function. The elements can be changed later, if you wish. But the name of the solution (here **u**) must be changed here.

The next screen shows the work plane (2D …) and one usually works from left to right in the menu bar:
1. Create geometry (**Draw**)
2. Define the PDE (**Physics**)
3. Set boundary conditions (**Physics**)
4. Make the mesh (**Mesh**)
5. Solve (**Solve**)
6. Plot (**Postprocessing**)

Under **File** we find operations like **open, save, save as**, etc. **Options** gives facilities like defining constants and expressions. It is practical to have names for important model parameters like heat transfer coefficients, specific heat, etc.

Geometry

The geometry is built in a drawing program, from elementary bodies (2D Rectangles and ellipses), or surfaces bounded by Bezier-curves. The bodies are put together by set operations (union, intersection, difference). We make an ellipse, centered at the origin and half axes 1 and 0.2. Note that the objects are snapped to the grid, change the grid under **Options>GridSettings** if needed. Our domain is only the single ellipse so we move on to
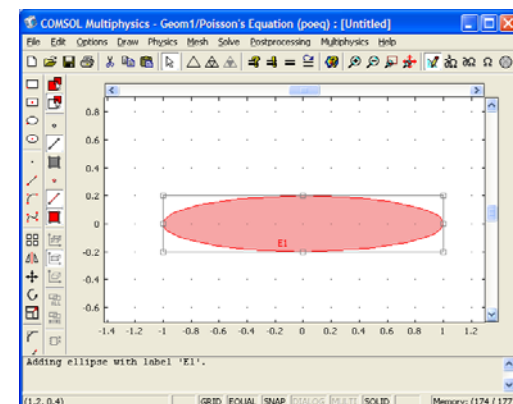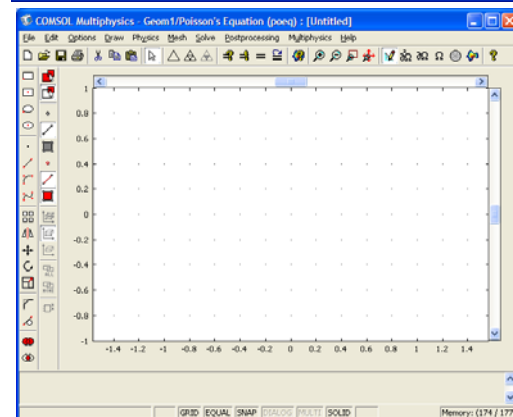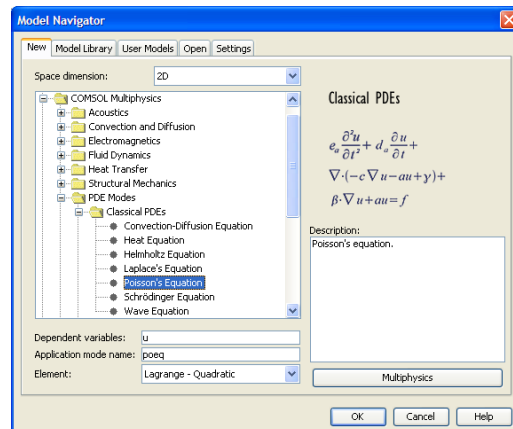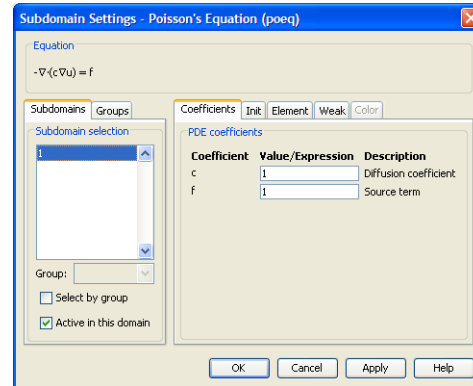
Define PDE

`Physics>SubdomainSettings` (there is
boundary too, in a minute…)
The PDE is in the Equation pane,
$$-\nabla \cdot (c\nabla u) = f$$
and we need to define the coefficient *c* and
the source term *f*. One can give functions of
x,y, u, ux, uy and, if time-dependent, *t*. There
is only one subdomain, the ellipse, so click it
and accept a *c* = *f* = 1 by OK.

`Physics>BoundarySettings`
Each ellipse gives four boundaries. Choose
them all by `ctrl-a`; They can be selected
also by clicking (shift-clicking, ctrl-clicking)
on the work pane. The default boundary
condition is apparently Dirichlet: *h u* = *r*
with *h* = 1 and *r* = 0 and we change to
*r* = sin(10*x*) for some variation, OK.
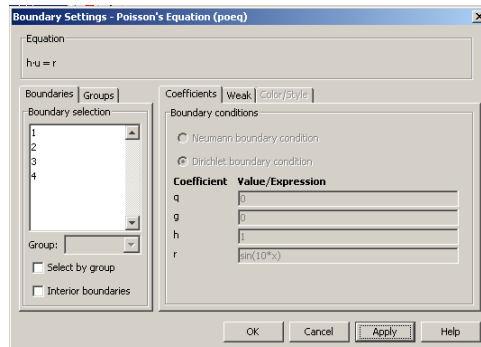
Element mesh

The triangle icon gives a mesh. It looks
good, OK. The mesh can be controlled in
some detail by parameters under
`Mesh>MeshParameters` .
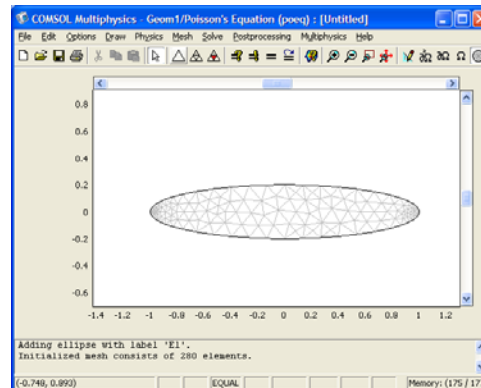
Solution

The generated equations are solved when
you click the = icon. Scroll down to
`Solve>SolverParameters` and check the
possibilities offered, Our choice of *c, f* and
boundary conditions makes the problem
linear so the default selection `stationary
linear` is correct, OK. Details that can be
very important for large (and non-linear)
problems, like choice of solver for linear
systems, are controllable here. It says
`Unsymmetric` about the matrix – whereas it
is in fact symmetric – but never mind, OK,
and solve by =.
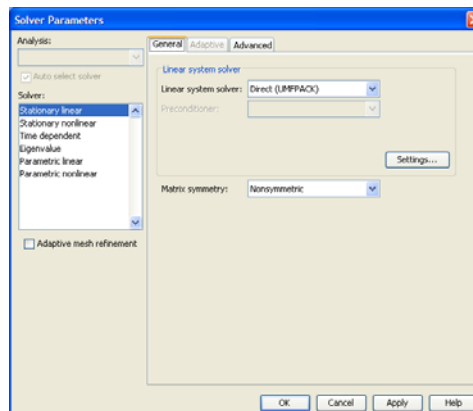
After a fraction of a second, the system with 613 unknowns has been constructed and solved and painted as colored iso-surfaces on the work pane, and the log-line (bottom of screen) says

```
Number of degrees of freedom
solved for: 613
Solution time: 0.093 s
```
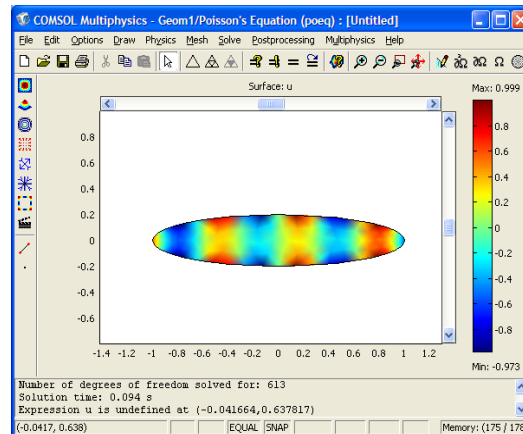

Convergence study
To see how the solution converges with refinement of the mesh, we compute u in the center. Choose
Postprocessing>DataDisplay>Subdomain
where we give the coordinates for the center – (0,0) and see

```
Value: 0.020201, Expression: u, Position: (0,0)
```
in the log-line.
Make a regular refinement (each triangle split into four) by the four triangle icon, solve again, and print u(center) again:
```
        Number of degrees of freedom solved for: 2345
        Solution time: 0.188 s
        Value: 0.019069, Expression: u, Position: (0,0)
```
and again, and again …, ... try up to, say, a million degrees of freedom when the solver runs out of memory.
Convergence:
```
u(0,0)  diff.
 1201
   69  -1132
  231    162
  234      3
  231     -3
```
The order of convergence should be 2 for regular solutions in energy norm and 3 in max-norm. The exponent is hard to see from the data, but the convergence is rapid. The convergence is *not* as regular as you have seen for e,g, the trapezoidal rule for quadrature, because the mesh is so irregular.
Now try piecewise linear basis functions instead. **Lagrange P1** elements are chosen under **Physics>SubdomainSettings>Element**

Repeat the experiment: Slower convergence.

```
#dof      time      u(0.0)     diff.
   613    0.094    0.019103
  2345    0.156    0.019115   12
  9169    0.547    0.019205   90
 36257    2.172    0.019224   19
144193   11.204    0.019230    6
```

Exempel 2 – Poisson's equation on a four-leaf clover

Start a new model by choosing `File>New,`Choose PDE like above: we want to solve the same equation but on a four-leaf clover.

Geometri

Create a circle. Copy it with **ctrl-c** (like in Windows), paste with **ctrl-v**, give the desired translation in the dialog box, paste twice more with suitable translations. Create the union of the four disks in **Draw>CreateCompositeObject, e.g.** "**select all**" (union is the default-operation, as you see in the formula window), OK. The new object was named **CO1** (Composite Object 1).

Define PDE

`Physics>SubdomainSettings`

Again, only one subdomain, the unionen; we could have kept all the interior boundary curves from the four circles, but we chose (unconsciously) (default!) to remove them, so they are now gone. Select the only domain and accept $c = f = 1$ by OK.

Boundary conditions

`Physics>BoundarySettings`

Many boundaries, 8 in all; ewach circle gives two and the two inner were erased. Select all ctrl-a; The default condition is - obviously - Dirichlet: $r = \sin(10x)$ like above, OK.

Element Mesh

The Triangle-icon provides an initial mesh which looks good so OK.

Solution

Solve by = and after a second or so they color picture appears. We carry out the convergence study like above – possibly you have different coordinates, no matter.

```
#dof     tid      u(0,0)      diff.
  5057  0.422    -0.253078
 20033  1.469    -0.253353   275
 79745  7.218    -0.253463   110
318209 39.953    -0.253507    44
```

Order of convergende somewhere between 1 and 2, DIFFERENT from the single ellipse. Why? Could it be that the solution is not as regular (smooth) as is required ? The sharp intruding corners are suspects. Plot |gradu| and it will show,

`PlotParameters>Surface>SurfaceData`, select **|gradu|**, OK

maybe even more clearly in a 3D-graph**,**

`PlotParameters>Surface>HeightData,` select **|gradu|**, click the check box **HeightData**, OK.

One can show (by separation of variables) that the singularity is of the type $r^{\frac{\alpha}{\pi}}$ when the exterior angle is α (here $\pi/2$). That is enough to reduce the rate of convergence noticeably.