INSA Toulouse, Applied Mathematics department
Finite Element Methods

Programming Practicals based on FEniCS

By Jerome.Monnier@insa-toulouse.fr
Other information and all files are available on the INSA Moodle page.

1 Introduction to FEniCS Library

a) Consult the short introduction and examples presented on-line (see URLs provided on the course Moodle page).

b) Install the library FEniCS 2019 on your own laptop.
NB. The provided codes are in FEniCS 2019 and nit in FEniCSx...

c) Upload and perform a few codes provided on the website of FEniCS e.g. those solving :
— the Laplace - Poisson equation,
— the time-dependent diffusion equation (heat equation).

2 The PDE-based model

2.1 The equations

The geometry is supposed to be bidimensional as $\Omega = [0, L_x] \times [0, L_y]$.
The unknown field (a scalar function) is denoted by $u(x)$.
As an illustration, u can represent a chemical specie concentration in a fluid in motion, or a fluid temperature. We denote by $\mathbf{w}(x)$ the fluid velocity. $\mathbf{w}$ is given.

The considered model is the following steady-state advection-diffusion equation :

$$-\mathrm{d}iv(\mu(u(x))\ \nabla u(x)) + \mathbf{w}(x) \cdot \nabla u(x) \ = \ f(x) \text{ in } \Omega \tag{1}$$

accompanied with B.C. (which are detailed below).

The diffusivity coefficient $\mu(u)$ depends the unknown field $u(x)$. As a consequence, the model is non linear.
We assume that : $\mu : u \in V \mapsto \mu(u) \in L^\infty(\Omega), \ \ \mu(u) > 0$. Moreover, $\mu(u)$ is supposed to be differentiable.

Remark. In the case μ is a given constant, the model is linear. In this case, we simply assume that $\mu \in L^\infty(\Omega)$, $\mu > 0$ a.e.

The elliptic PDE above is closed with the following B.C. :

$$u \text{ given on } \Gamma_{in}; \ \partial_n u = 0 \text{ on } \Gamma_{wall}; \ -\mu(u)\partial_n u = c\ u \text{ on } \Gamma_{out} \tag{2}$$

with c a given parameter, $c \geq 0$.
We have $\partial\Omega = \Gamma_{wall} \cup \Gamma_{in} \cup \Gamma_{out}$.

3 Advection - diffusion phenomena

The model is those previously detailed.

3.1 Mathematical analysis

a) Write the weak formulation of this BVP.
b) In the case μ is a given constant, derive conditions such that this BVP is well-posed.

3.2 Numerical simulations

a) Interprate the simulated phenomena.
b) Peclet number.
- Compute and plot out the local values of the dimensionless Peclet number $Pe = (\frac{L^* W^*}{\mu})$.
- Comment the obtained Peclet map.

3.3 Stabilized numerical scheme

This is a "to go further" task.
a) Implement the numerical model in FeniCS by using P_k-Lagrange FE.

To do so, you will start from the Python-Fenics code available on the course Moodle page.
b) Explain how you could, (should!), validate your computational code and demonstrate its actual order.

4 Non-linear model resolution

The diffusivity parameter $\mu(u)$ is given, depending on u as described above. It is supposed to be $C^1(\bar{\Omega})$, and strictly positive.

For example, the diffusivity expression may be set as :

$$\mu(u) = \mu_0 u^m \text{ with } m \geq 1$$

μ_0 given, strictly positive.

4.1 Build-up and compare non linear solvers

a) Write the weak formulation of this BVP.
b) (Partly) code non-linear solvers.
- Derive the equations to be solved at each iteration.
- Implement your home-developed non-linear solver based on a semi-linearization.
Of course, you will first write in details the algorithm(s) you implement.
- Implement your home-developed non-linear solver based on the Newton-Raphson method.

You will start your own code from the Python-FeniCS code(s) available online.

c) Black-box solver.
Implement the black-box non-linear solver proposed in FEniCS.
d)
- Compare the few approaches and the obtained few solutions (those with your partly home coded solvers, those using the black-box solver).
- Compare the convergence speed and CPU-time.
- Do the algorithms converge whatever the "initial point"? Compare their "robustness". Discuss.

e) To go further. Describe a strategy to measure the actual order of convergence of each approach.