

Newton's Method in Hermes

Pavel Solin

hp-FEM group, University of Nevada, Reno
Academy of Sciences, Prague, Czech Republic
<http://hpfem.math.unr.edu>

November 2008

This is to illustrate how Hermes can solve general nonlinear PDE problems via the Newton's method. Consider a problem of the form

$$-\nabla \cdot (a(u)\nabla u) = f(u), \quad u = 0 \text{ on } \partial\Omega. \quad (1)$$

The corresponding discrete problem has the form

$$\int_{\Omega} a(u)\nabla u \cdot \nabla v_i \, d\mathbf{x} = \int_{\Omega} f(u)v_i \, d\mathbf{x} \quad \text{for all } i = 1, 2, \dots, N, \quad (2)$$

where

$$u(\mathbf{Y}) = \sum_{j=1}^N y_j v_j. \quad (3)$$

Here $\mathbf{Y} = (y_1, y_2, \dots, y_N)^T$ is the vector of unknown coefficients. Equation (2) can be written in the compact form

$$\mathbf{F}(\mathbf{Y}) = \mathbf{0}, \quad (4)$$

where $\mathbf{F} = (F_1, F_2, \dots, F_N)^T$ with

$$F_i(\mathbf{Y}) = \int_{\Omega} a(u)\nabla u \cdot \nabla v_i - f(u)v_i \, d\mathbf{x}. \quad (5)$$

The Jacobi matrix $\mathbf{J}(\mathbf{Y}) = D\mathbf{F}/D\mathbf{Y}$, which is all we need for the Newton's method, has the same sparsity structure as the standard stiffness matrix. On the position ij , it has the value

$$J_{ij}(\mathbf{Y}) = \frac{\partial F_i}{\partial y_j} = \int_{\Omega} \left[\frac{\partial a}{\partial u} \frac{\partial u}{\partial y_j} \nabla u + a(u) \frac{\partial \nabla u}{\partial y_j} \right] \cdot \nabla v_i - \frac{\partial f}{\partial u} \frac{\partial u}{\partial y_j} v_i \, d\mathbf{x}. \quad (6)$$

Since

$$\frac{\partial u}{\partial y_j} = v_j \quad \text{and} \quad \frac{\partial \nabla u}{\partial y_j} = \nabla v_j,$$

equation (6) becomes

$$J_{ij}(\mathbf{Y}) = \int_{\Omega} \left[\frac{\partial a}{\partial u}(u)v_j \nabla u + a(u)\nabla v_j \right] \cdot \nabla v_i - \frac{\partial f}{\partial u}(u)v_j v_i \, d\mathbf{x}. \quad (7)$$

Let's assume that the Jacobi matrix has been assembled. The Newton's method is written formally as follows,

$$\mathbf{Y}_{n+1} = \mathbf{Y}_n - \mathbf{J}^{-1}(\mathbf{Y}_n)\mathbf{F}(\mathbf{Y}_n),$$

but a more practical formula is

$$\mathbf{J}(\mathbf{Y}_n)\delta\mathbf{Y}_{n+1} = -\mathbf{F}(\mathbf{Y}_n). \quad (8)$$

This is a system of linear algebraic equations that needs to be solved in every iteration. The Newton's method will stop when $\mathbf{F}(\mathbf{Y}_{n+1})$ is sufficiently close to the zero vector.

Implementation in Hermes

The current way we define linear and bilinear forms in Hermes is perfectly sufficient to define the function \mathbf{F} as well as its Jacobi matrix $\mathbf{J} = D\mathbf{F}/D\mathbf{Y}$, respectively. Both the linear and bilinear forms will need the global solution \mathbf{Y}_n as an extra input parameter (we did this with previous-time-step solutions in time-dependent problems). Right now, Hermes does the following (taken from Example 03):

```
// assemble the stiffness matrix and solve the system
Solution sln;
sys.assemble();
sys.solve(1, &sln);
```

For the Newton's method, there will be something like

```
// assemble the stiffness matrix and solve the system
Solution sln;
sys.assemble_newton();
sys.solve(1, &sln);
```

Here, the method `sys.assemble_newton()` calls `sys.assemble()` internally to get the stiffness matrix $\mathbf{J}(\mathbf{Y}_n)$ and the right-hand side $\mathbf{F}(\mathbf{Y}_n)$. Then, it changes the right-hand side to $\mathbf{J}(\mathbf{Y}_n)\mathbf{Y}_n - \mathbf{F}(\mathbf{Y}_n)$. After calling `sys.solve()`, the user will have \mathbf{Y}_{n+1} . That's all!

For the stopping criterion, we should allow the user to calculate the residuum $\mathbf{F}(\mathbf{Y}_{n+1})$ (which will be a vector of N real numbers). This will be analogous to the assembling procedure, but the stiffness matrix will not be created, just the right-hand side.

Note on the linear case

In the linear case we have

$$\mathbf{F}(\mathbf{Y}) = \mathbf{J}(\mathbf{Y})\mathbf{Y} - \mathbf{b},$$

where $\mathbf{S} = \mathbf{J}(\mathbf{Y})$ is a constant stiffness matrix and \mathbf{b} a load vector. Equation (8) yields

$$\mathbf{S}\mathbf{Y}_{n+1} = \mathbf{J}(\mathbf{Y}_n)\mathbf{Y}_n - \mathbf{J}(\mathbf{Y}_n)\mathbf{Y}_n + \mathbf{b} = \mathbf{b}.$$

Therefore, the Newton's method will converge in one iteration.