AdFEM manual. November 8, 2000. Originally written by S. Larsson.

AdFEM is an adaptive finite element MATLAB program written by K. Eriksson for approximate solution of two-point boundary value problems of the form: find $u = u(x)$ such that, with $D = d/dx$,

$$-D(d(x)Du) + c(x)Du + a(x)u = f(x), \quad \text{for } x_{\min} < x < x_{\max},$$

and such that

$$\begin{aligned} -dDu + ku = g \quad &(\text{Neumann/Robin}), \quad \text{or} \quad u = g \quad (\text{Dirichlet}) \quad \text{at } x = x_{\min}, \\ dDu + ku = g \quad &(\text{Neumann/Robin}), \quad \text{or} \quad u = g \quad (\text{Dirichlet}) \quad \text{at } x = x_{\max}. \end{aligned}$$

Here $d(x)$ is the diffusion coefficient, $c(x)$ represents convection, $a(x)$ may be interpreted as an absorption intensity, and $f(x)$ is a source or a force acting on the system.

The boundary conditions may be of Dirichlet type where the value of u is prescribed, or of Neumann/Robin type where a combination of the normal flux $-dDu$ and u is prescribed. In a heat conduction problem, where u is the temperature and $-dDu$ is the heat flux, this corresponds to prescribing the temperature at the boundary, prescribing the heat flux through the boundary (with $k = 0$), or relating the heat flux through the boundary to the temperature jump across the boundary in which case g/k represents the exterior temperature. Of course, the values of k and g need not be the same at $x = x_{\min}$ and $x = x_{\max}$.

The discretization method is the standard finite element method based on piecewise linear trial and test functions (with a stabilizing "stream-line diffusion modification" if the problem is convection dominated).

The error may be controled in the energy norm, L_2-norm, or maximum norm on a user specified tolerance level.

The structure of the adaptive algorithm is as follows. Once all the data of the problem have been given, the program predicts (in the subroutine `pred`) a suitable initial (usually uniform) mesh, and computes the corresponding approximate solution. This solution is then tested (in the subroutine `test`) using a so called "a posteriori" error estimate, where the error is estimated (in the given norm) in terms of the residual of the approximate solution. If the error is below the given tolerance, then the current approximate solution is presented as "tested and accepted" and the algorithm stops. Otherwise, a new (corrected) mesh is constructed (in the subroutine `corr`) using the information obtained in the previous test, and the solution and test procedure is repeated. If we are lucky the new approximate solution is OK. Otherwise, the adaptive procedure is repeated until the tolerance is met. The current meshsize, solution, and residual are plotted during the adaptive process. A certain stability factor which is used in the error estimate is also displayed. The size of this factor reflects the "difficulty" of the given problem.

To run the program, first give the command `matlab` to start MATLAB. Then start AdFEM by typing `adfem` at the MATLAB prompt, that is, `>> adfem` (here `>>` is the MATLAB prompt and should not be typed). Input data by answering the questions and await the solution! You will be prompted for the coefficients $d(x)$, $c(x)$, $a(x)$, $f(x)$, and you type them in as MATLAB expressions. For example, AdFEM says `d(x) =` and you type `2*exp(-x^2)`.

If you want to rerun AdFEM with slightly different data, then you can take a short cut by entering the new data directly at the MATLAB prompt and then type the command `>> solve`. For example, if you want to rerun with a smaller tolerance you type `>> tol=0.001; solve` and AdFEM will solve the given problem with the new tolerance. Other data that you may want to reset in this way are:

`xmin, xmax, diffusion, convection, absorption, force, errnorm,`
`exactflag, uexact, bctype, k, g.`

Here `bctype`, `k` and `g` are 2-vectors (1×2 matrices in MATLAB) defining the boundary condition type (`0` for Dirichlet and `1` for Neumann/Robin) and the corresponding values of k and g at $x = x_{\min}$ and $x = x_{\max}$, see the example below. `xmin` and `xmax` define the computational domain. The other data variables are text strings and have to be given within single quotes, for example, `>> force='1/x'`. The error norm is prescribed by the statement `>> errnorm='en'` (for energy), `='l2'` (for L_2), or `='ma'` (for the maximum norm). Finally, you have the option to check the efficiency and reliability of the error control by providing the exact solution `uexact`. The program will then compute the true error `exacterror` for comparison with the estimated error `error`. Set `>> exactflag='y'` (for yes) if you want to use this option.

You may also enter the data and run AdFEM from the MATLAB prompt as the following example shows. The example solves the boundary value problem:

$$
\begin{aligned}
-\left(\frac{1}{1+x}u'(x)\right)' &= x \quad \text{for } 0 < x < 1, \\
\frac{1}{1+x}u'(x) = 0 \quad \text{at } x = 0, &\quad u(x) = 1 \quad \text{at } x = 1,
\end{aligned}
$$

with exact solution $u(x) = \frac{31}{24} - \frac{1}{6}x^3 - \frac{1}{8}x^4$. To run this example you can type the following:

```
>> xmin=0
>> xmax=1
>> diffusion='1/(1+x)'
>> convection='0'
>> absorption='0'
>> force='x'
>> errnorm='ma'
>> exactflag='y'
>> uexact='31/24-(x^3)/6-(x^4)/8'
>> bctype=[1 0]
>> k=[0 0]
>> g=[0 1]
>> tol=0.01
>> run=0
>> solve
```

Note: It is better to start by typing `>> adfem` and enter the data at the AdFEM prompt; this example is only supposed to illustrate how to reset the variables before you rerun your computation using the command `>> solve`.

Warning: Text strings must be surrounded by single quotes, for example `'1/(1+x)'`, when you enter data in this way, but not when you are prompted for data by AdFEM.

Enjoy! `/Niklas`