FreeFem++: A High Level MultiPhysics Finite Element Software

By Olivier Pironneau

The finite element method (FEM) was invented shortly after computers as a natural framework for solid mechanics; the success of NASTRAN—a finite element analysis program—in the 1960s is well known. Variational methods were then popular among mathematicians in the analysis of partial differential equations (PDEs), which rendered FEM easily adaptable to other domains of physics, such as thermodynamics, fluid mechanics, and electromagnetism. However, the preparation of data at the time was a nightmare, and still is for many 3D applications. Thus, teaching FEM for PDEs was a challenge: one could easily lose sight of the essence on translation and graphics methods.

Niklaus Wirth and the Finite Element Method

Dominique Bernardi and I wrote FreeFem++ in the 1990s at the University of Paris VI to ease the teaching and prototyping of PDEs. Upon Fédéric Hecht’s introduction of the advanced automatic mesh generator, I realized the power of that module and asked him to partake in the FreeFem++ project. We prioritized the user interface’s ability to define the PDE and hide the numerical methods. We aimed for the highest possible level, nearest to the mathematical statements. For instance, a Stokes system in

\[
-\Delta u + \nabla p = 0, \quad \nabla \cdot u = 0, \quad u \text{ given on the boundary},
\]

discretized with the Hood-Taylor element and with regularization of the pressure constraint, would be defined and solved by the Freefem++ code for the Navier-Stokes equations in 3D.

SOFTWARE AND PROGRAMMING

See FreeFem++ on page 8

Figure 2. Freefem++ code for the Navier-Stokes equations in 3D.

The advantages of C++ to Freefem++

As the years passed, more and more instructions were added to FreeFem++, which now has a considerable subset of C++ instructions in its syntax (useful to open and close files, manipulate matrices, etc.). Actually, the name FreeFem++ comes from the fact that the software was entirely rewritten in C++ in 2002 and again in 2010. It makes extensive use of operator overloading and templates, something known as generic programming. Consequently, including complex numbers and adding 3D problems—vector problems like Stokes’—was easy, since the syntax is the same as for the heat equation.

FreeFem++ is a collaborative open-source effort like its cousin fenics-project, including complex numbers and adding 3D problems—vector problems like Stokes’—was easy, since the syntax is the same as for the heat equation.

SOFTWARE AND PROGRAMMING

Figure 2. Freefem++ code for the Navier-Stokes equations in 3D.
Strengths and Weaknesses

By comparison to its commercial competitor COMSOL, FreeFem++ is primitive for graphics but interfaced with ParaView and others. Nonetheless, it does things that COMSOL cannot do. Both run on Mac OS, Windows, and Linux in integrated environments. FreeFem++ runs on smartphones thanks to Antoine Le Hyaric. However, we haven’t found much use for this extension yet.

FreeFem++ has the best open-source module for mesh adaptivity. One can also easily interpolate functions on different meshes, simplifying domain decomposition methods, for example. It has a robust implementation of the characteristic Galerkin method for convection terms, and a large library of triangular finite elements—P0, P1, RT, HCT, Edge, mixed—adapted to discontinuous Galerkin methods (DG) and a posteriori mesh refinements. The one thing FreeFem++ does not handle well is conservation laws with shocks. We have not yet been able to write a fast Riemann solver for a general hyperbolic system of any size; one may solve such PDEs with DG but doing so requires considerable expertise, which is not in the spirit of FreeFem++.

Many C++/Fortran90 toolboxes are potentially just as powerful, such as Feel++5, deal.II,6 and Elmer,7 among others.8 But in general, these require that users have a high level of programming ability. Field-specific languages are coming; the meta language scala9 makes such a promise.

FreeFem++ has at least 3,000 active users. Ph.D. students worldwide seem to love it; in France, almost all courses on numerical analysis for PDEs use FreeFem++. But it is also increasingly popular as a research tool because execution time is comparable with handwritten C++ codes, and development time is so much faster. For instance, Figure 4a shows the iso-pressure surfaces of blood flow in an aorta using a mesh constructed from an MRI. A simulation of Maxwell equations in 3D (see Figure 4b), entirely written with FreeFem++, won the French supercomputing Atos-Fourier prize10 because it scales perfectly on a parallel machine up to 10,000 cores. The next challenge is hiding parallel computing instructions from the user and automatically employing the computer’s resource. Frédéric Nataf and Pierre Jolivet are working hard at it, with Hecht’s help, of course!

References


Olivier Pironneau is emeritus professor at the Université Pierre et Marie Curie in Paris, France. He is also a member of the French Académie des Sciences and co-author of eight books, including Computational Methods for Option Pricing (SIAM, 2005). Visit his website11 for more information.

Randall J. LeVeque (rjl@uw.edu) of the University of Washington, Seattle, is the editor of the Software and Programming column.

Figure 3. Results of Figures 1 and 2. 3a. Stokes flow: velocity vectors and pressure lines. 3b. Navier-Stokes flow: iso-pressure surfaces. Image credit: Olivier Pironneau.


Figure 5. Blood flow reconstruction in the brain by simulation of Maxwell equations. Top: Target perfusivity of the brain generated by high-resolution imaging. Bottom: Computer-reconstructed image from noisy microwave measurements. The arrows indicate the region of stroke. Image credit: Frédéric Nataf.