

TEACHING FINITE ELEMENTS: AN ALTERNATIVE APPROACH USING MODELICA

Wolfgang Wiechert¹, Marc Treude¹

¹University of Siegen, Dept. of Simulation,
Paul-Bonatz-Str. 9-11, 57068 Siegen, Germany

wolfgang.wiechert@uni-siegen.de

Abstract

The Finite Element Method (FEM) is one of the most important simulation methods in engineering and natural sciences. However, it rather seldom is an integral part of a simulation lecture. Likewise, books on the FEM typically avoid the term “simulation”. This strange distinction between “general” simulation methods, on the one hand, and Finite Elements (FEs), on the other hand, has historical, cultural, didactical and methodological roots.

Interestingly, modular or object oriented approaches to complex system simulation have many ideas in common with the classical FEM. This gives rise to an alternative teaching approach which seamlessly integrates the basics of FEs into an advanced simulation lecture on continuous time simulation. The advantage of this concept is the exploitation of commonalities between FEs and modular modeling to speed up the learning process.

The modular approach aims at a quick but completely transparent implementation of first FE simulations by using Modelica. Having understood the basic computational machinery the students will then be highly motivated to learn about the theoretical foundations of the FEM. The concept was tested with encouraging results in the Siegen simulation courses.

Since Modelica, as a general purpose system, is not really a good FE tool, the classical trisection into preprocessor, numerical solver and postprocessor has been implemented by combining Modelica with Matlab. The preprocessor allows drawing simple 2D meshes from which a Modelica file is automatically generated. The postprocessor takes up the Modelica output file and visualizes the simulation results on a 2D mesh. All intermediate steps are completely transparent to the students. Making the experience that all steps of a basic FE simulation can be practically managed on an elementary level frees the method from several theoretical and formal burdens.

Keywords: Finite Elements, Modelica, modular modeling, simulation education

Presenting Author’s biography

WOLFGANG WIECHERT studied mathematics and computer science at the University of Bonn and obtained his PhD in 1991. From 1991 to 1996, he worked at the Jülich Research Center. 1996, he became a professor for simulation at the Institute of Systems Engineering at the University of Siegen. His major research fields are the development of new methods and tools for modeling and simulation in materials sciences, production processes, micro fluidic devices, systems biology, and multi sensorics.



1 Introduction

Doubtlessly, the Finite Element Method (FEM) is one of the most important simulation methods in the engineering and natural sciences. For engineers it is an integral part of many university curricula [1]. Mathematicians simply consider it as the most important numerical method to solve partial differential equations [2].

Despite of its importance, FEs are not taught in the context of typical simulation lectures. Likewise, many FE text books don't even use the term "simulation". One reason might be the historical origin of the method as a tool to solve continuum mechanical problems; another one is the fact that time dependent problems usually do not play an important role in an introductory FEM lecture. Most important, traditional FE lectures take a full semester without leaving any space for other topics.

This is one motivation for the developments described in the present contribution: FEs should be seamlessly integrated into an advanced simulation lecture. "Seamless" here means that several concepts which are also part of a typical lecture on continuous time simulation should be efficiently "reused" for introducing the FEM. This significantly speeds up the learning process because a lot of concepts are already familiar to the students.

The second motivation to try an alternative approach to FEs is that the original idea of the FEM – i.e. a modular decomposition of spatial continua – is obscured in many modern text books by a lot of formal and technical details. Since the beginner is usually not able to distinguish between the core ideas and purely technical or formal aspects, the whole method makes an unnecessarily complicated impression. Particularly, it is not clear for the student, how the theoretical and formal constructs introduced in the first parts of a typical FEM lecture are finally transformed into a practical numerical method. In many cases the students will see their first running program at a rather late stage of the lecture.

Consequently, a major goal of the alternative approach presented here is to find the shortest way to the first nontrivial FE program. This does not mean that there is a way to understand the FEM in depth without the underlying mathematical theory. For this reason, some details and mathematical precision have to be sacrificed, for the "hands on" approach. However, once the students got their first program running, the students are highly motivated to learn about the mathematical details.

2 Finite Elements and modular models

From a naive viewpoint the basic idea of FEs is quite simple (cf. Fig. 1): A continuous spatial domain is discretized as a system of interacting FEs. For each single element the physical laws governing the local

behavior of the system are approximated using a discrete set of variables. The result is some kind of linearized constitutive law for the single elements. The chosen variables are associated to certain interface points at the boundary of the elements. Then, the coupling relations between neighbored elements and the boundary conditions of the system have to be specified. This, finally, yields an equation system having as many unknowns as there are equations (Fig. 1) [3].

Interestingly, this is exactly the approach taken in modern modular or object oriented simulation languages like Modelica [4] or VHDL-AMS [5] to model complex systems with spatially concentrated parameters. From now on it is assumed that the fundamentals of modular modeling and (exemplarily) the Modelica language have already been introduced and practiced in an advanced simulation lecture. Then it is an obvious idea to develop the FEM as a special case of a general modular approach.

The new concept has been tested in a one semester advanced lecture on "Multidisciplinary Modeling and Simulation" which is primarily concerned with object oriented modeling concepts and the computational mechanisms underlying the Modelica language. If the theory of the FEM is omitted, it just takes a few hours to understand the FE machinery at the end of the lecture.

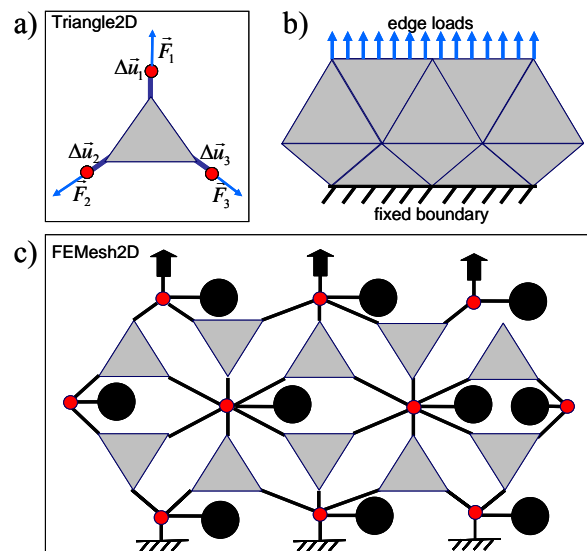


Fig. 1: a) Displacements and forces as interface variables of an elastic triangle element. b) Simple triangulated elastic body with loads and bearings. c) FE network approximating the elastic continuum.

3 Advantages of the new approach

Using the Modelica approach, the following technical or formal concepts can be avoided in an early stage of the FEM introduction:

- The mathematical theory for locally approximating the solution of partial differential equations on FEs can be shifted to the second part of a lecture.

Instead the approximation result is first introduced by examples. Nevertheless, the students can understand the behavior of the approximation by doing empirical explorations on single elements in the computer lab (cf. Fig. 4). Clearly, this empirical understanding should be theoretically founded later on.

- The central but rather technical concept of test function spaces and their basis functions need not be introduced from the beginning on because FEs are first introduced as approximated physical components with their interface points and constitutive potential-flow laws. However, some problems arise when spatially distributed properties like mass density, or non constant boundary conditions have to be mapped to the discrete state variables of the elements (cf. Fig. 5). Again, there is an intuitive empirical work around available.
- The algorithmic problem of assembling the elements to the whole system model by integrating the local equations into one large equation system is simply not present in the new approach. The reason is that this assembly mechanism, being universal for any (linear or nonlinear) modular method, should already be a central part of a Modelica course. Consequently, the students are already familiar with the description of global behavior from local equations. This understanding usually takes a lot of time in a classical FEM lecture.
- In the new approach time dependency is included from the beginning because the students in an advanced simulation lecture will be familiar with a dynamical systems approach. Clearly, the static case turns out as a special situation.

It should be pointed out that the alternative approach is not a “light” version of a classical FEM lecture because it really goes to the details of the computational machinery instead of just demonstrating the usage of a commercial FE tool. The difference to the classical teaching concept is rather a changed arrangement of the traditional contents: The mathematically involved parts of the theory are shifted to the second part of the lecture without sacrificing too much terminological and methodological precision.

The following sections now concentrate on the technical details of the alternative approach and the teaching tools developed for this purpose.

4 Modular modeling with Modelica

FEs can be best introduced with examples from continuum mechanics or heat conduction. In both cases the arising equations can be understood quite intuitively. Moreover, the one dimensional case (i.e. Hooke's or Fourier's law) is familiar even to students with a cursory knowledge in mechanics or heat conduction. For this reason any Modelica course will in-

clude examples for mass-spring-damper systems and networks of heat capacities-heat conductors in the thermal case. These components are also included in the standard Modelica libraries [4]. Although both examples are treated in the lecture we restrict to the mechanical case in the following.

To recall the basic concepts of modular physical modeling using Modelica it is shown here by example how 2D mass-spring systems with different boundary conditions can be modeled. Fig. 2a shows the interface variables required to describe the components. Clearly, the system behavior can be made more realistic by introducing additional dampers which are omitted here for brevity.

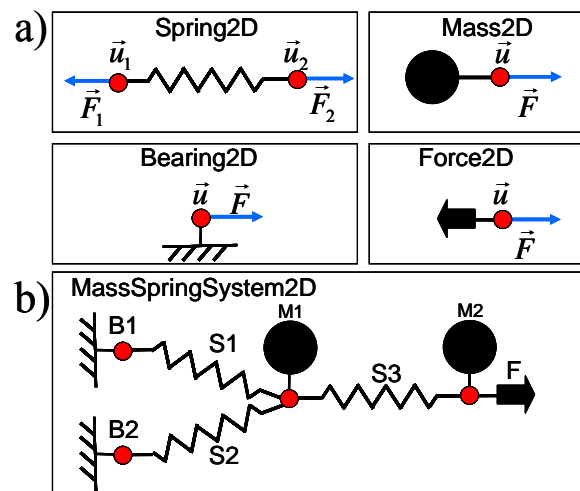


Fig. 2: a) Basic modules for modeling mass-spring systems with their interface variables (positions and forces) b) Structure of a simple mass spring system.

The constitutive laws of these components are given by (\vec{u}_i position vectors, \vec{F}_i force vectors, L relaxed spring length, c spring constant):

$$\text{Mass: } m \cdot \ddot{\vec{u}} = \vec{F}$$

$$\text{Spring: } \vec{F}_1 = -\vec{F}_2 = c \cdot (L - \|\vec{u}_{12}\|) \cdot \vec{u}_{12} / \|\vec{u}_{12}\|$$

$$\text{with } \vec{u}_{12} = \vec{u}_1 - \vec{u}_2$$

$$\text{Bearing: } \vec{u} = \text{const.}$$

$$\text{Force: } \vec{F} = \text{const.} \quad (1)$$

The laws relate the potential variables \vec{u}_i of the components with their flow variables \vec{F}_i . A Modelica code fragment describing the spring is given by:

```
Connector MechNode2D
  flow Real Fx, Fy ;
  Real ux, uy ;
end MechNode2D ;

model Spring2D
  MechNode2D n1, n2 ;
  Parameter Real c=1, L=1 ;
```

equation

```
L12 = sqrt ((n1.ux-n2.ux)^2+(n1.uy-n2.uy)^2);
f = c * (L-L12);
n1.Fx=f * (n1.ux-n2.ux) / L12;  n2.Fx=-n1.Fx;
n1.Fy=f * (n1.uy-n2.uy) / L12;  n2.Fy=-n1.Fy;
end Spring2D;
```

Building up a system from its components now means to integrate multiple copies of these modules with the respective coupling equations that arise from connecting the components. E.g. the coupling structure of the mass spring system shown in Fig. 2b is given by (parameters omitted):

```
model MassSpringSystem2D
  Spring2D S1,S2,S3;
  Mass2D M2,M2;
  Bearing B1,B2;
  Force2D F;
equation
  connect (B1.n, S1.n1);
  connect (B2.n, S2.n1);
  connect (S1.n2, S2.n2);
  connect (S1.n2,M1.n);
  ...
end MassSpringSystem2D
```

5 Transition to Finite Elements

One key idea of the classical FEM now is to linearize the system around a reference state. In the mechanical case this reference state is given by the relaxed (i.e. force free) system configuration. Clearly, this approximation is only valid if the displacements remain small when the boundary constraints and the gravitational forces are applied.

The students in an advanced simulation course should be familiar with linearization concepts. In the FEM method the linearization is done module wise before the system assembly takes place. After linearization the potential variables are no more given by the positions of the coupling points but by their displacements from their reference configuration. Displacements are henceforth indicated with a Δ symbol in front of the corresponding variable name.

For example the nonlinear spring model from Eq. (1) is linearized in the following way:

$$\vec{F}_1 = -\vec{F}_2 \approx -\underbrace{c/\|\vec{u}_{12}\|^2}_{\mathbf{S}} \cdot \vec{u}_{12} \cdot \vec{u}_{12}^T \cdot (\Delta\vec{u}_1 - \Delta\vec{u}_2) \quad (2)$$

The arising constant matrix \mathbf{S} is known as the element stiffness matrix. The students can convince themselves that the linearized version well approximates the original spring model as long as the displacements are small.

Replacing the original nonlinear model by its linear approximation (henceforth called a rod), it is immediately possible to modify the Modelica library by changing positions to displacements and springs to rods.

As an exercise classical networks of rods and masses can be immediately built up and simulated by the students (Fig. 3). These are already the first running FE models.

The most remarkable difference to the original nonlinear model is, that the former interface variables (i.e. positions) now become additional parameters for describing the relaxed configuration. Consequently, all these coordinates must now be explicitly given in the network configuration (cf. Fig. 3):

```
model MassRodSystem2D
  Spring2D S1(u1x,u1y,u4x,u4y),
           S2(u2x,u2y,u4x,u4y),
           S3(u2x,u2y,u5x,u5y),
  ...
end MassRodSystem2D
```

So far, the new approach is still similar to a typical FE introduction for engineers. The first essential difference now lies in the assembly of the system equations. Clearly, since all linearized constitutive laws as well as the coupling equations are linear, the resulting overall equation system is linear too. However, it is not necessary to explain in detail how the system stiffness matrix can be compiled from the element stiffness matrices because this is automatically performed by Modelica. This avoids a lot of technical details.

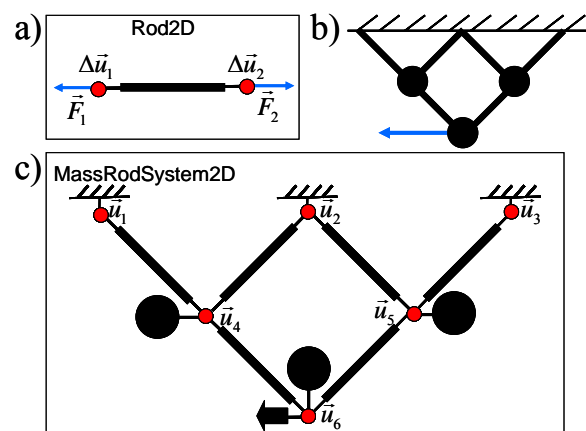


Fig. 3: a) Interface variables of a linearized spring element (rod). b) Simple rod network. c) FE approximation with coordinates of the relaxed configuration.

Another fundamental difference between a classical FEM introduction and the Modelica approach is that the coupling equations are not explicitly generated in the FEM. Instead, the system stiffness matrix is directly compiled from the element stiffness matrices. However, since a Modelica compiler will automatically remove all trivial coupling equations, the final result will be the same.

6 Triangle elements

As long as the elements have no spatial extension as in the examples of springs and heat conductors it will take just take an hour of lecture time to establish the

first FE model. This becomes more difficult when spatially extended elements like triangles in 2D or tetrahedrons in 3D are considered which are essential for the discretization of continuous media. In the following it is sufficient to illustrate the 2D triangle case. It should be immediately clear to the students, that in principle, a triangle discretization should work like the example shown in Fig. 1.

The problem now is to derive the stiffness matrix of a triangle element which should be a 6x6 matrix. However, the nonlinear behavior of the original system in space is no more described by an explicit equation like in Eq. (1) but by a partial differential equation. For this reason it is not straight forward to derive a linearized potential-flow law from the PDE model in analogy to the transition from Eq. (1) to Eq. (2). If the underlying formal machinery is presented in a lecture the students are confronted with the quite abstract formalisms of weak PDE solutions, test function spaces, basic approximation theory or even Sobolev spaces [2].

However, this is not really necessary for a first encounter with FEs. To achieve a practical understanding, the details of the approximation can be (temporarily) omitted and the resulting linearized model is presented from scratch. As an example the approximate model to describe the elastic deformation of the triangle is studied. The potential variables are given by the displacements at the three edges of the triangle and the flows by resulting forces (Fig. 1a). The element stiffness matrix \mathbf{S} in the case of plain strain is known to be (A – triangle area, d – element thickness, μ Poisson number, E – Young module) [3]:

$$\mathbf{S} = d \cdot \mathbf{V} \cdot \mathbf{B}^T \cdot \mathbf{D} \cdot \mathbf{B} \quad \text{with}$$

$$\mathbf{B} = \frac{1}{2A} \begin{pmatrix} y_{23} & 0 & y_{31} & 0 & y_{12} & 0 \\ 0 & x_{32} & 0 & x_{13} & 0 & x_{21} \\ x_{32} & y_{23} & x_{13} & y_{31} & x_{21} & y_{12} \end{pmatrix}$$

$$\mathbf{D} = \frac{E}{(1+\nu)(1-2\nu)} \begin{pmatrix} 1-\nu & \nu & 0 \\ \nu & 1-\nu & 0 \\ 0 & 0 & (1-2\nu)/2 \end{pmatrix}$$

$$A = [x_{21}y_{31} + y_{21}x_{31}] / 2$$

$$x_{ij} = x_i - x_j \quad y_{ij} = y_i - y_j \quad (3)$$

It is a 6x6 matrix relating 3x2 (x,y)-displacements of the triangle vertices to 3x2 (x,y)-vertex forces.

Although these equations look very complicated without knowing their derivation, it is very easy to convince the students that the approximated potential-flow relation shows the expected behavior. For this reason a small Matlab program has been written for interactive drawing and visualization of potential flow relationships (Fig. 4). By this empirical study the students become convinced that there is nothing mysterious about the stiffness matrix.

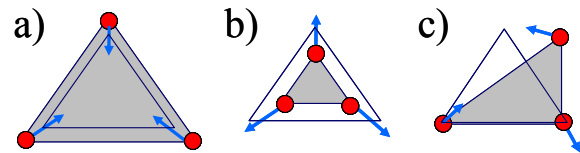


Fig. 4: Interactive exploration of the potential flow relationship: a) Expanding, b) compressing, c) shearing a triangle. The underlying relaxed triangle is shown by its contour in each case.

7 Coupling triangles

Having understood the function of single triangle elements it is straight forward to couple triangle elements for the modeling of more complex geometries (Fig. 1c). Nothing new has to be explained here if the spring example is already understood. Likewise, the introduction of boundary conditions is quite simple if only point loads are considered.

In the dynamic case masses have to be coupled to the vertex points in the triangle mesh. They can be computed by attributing a part of the triangle to the next neighbored edge point (Fig. 5a). However, it should be noticed that this intuitive approach fails if the material has non constant mass density.

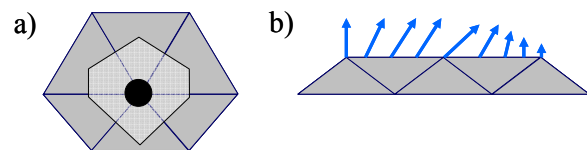


Fig. 5: a) Attributing masses to the interface nodes. b) Attributing non constant loads to interface nodes.

Likewise, if a boundary condition changes continuously a deeper knowledge of the test function formalism would be necessary to understand how they are mapped to the triangle vertices.

Taking this way the students will be able to build up a complete FE model with very little effort. However, after doing this in Modelica with the rod elements there is nothing new to be learned for triangles. For this reason a ready to use Modelica library for 2D continuum mechanics is supplied. This library can be studied by the students to see how it is implemented. The whole unit on continuum mechanics then takes only two more hours of lecture time.

8 Preprocessing and postprocessing

As long as there are only a few triangles in a system it is possible to specify all geometric parameters of a FE model manually. For a higher number of triangles it is a tedious procedure to correctly supply all the interface coordinates. Clearly, this task is automated by commercial FE tools. However, this part of automatic model generation cannot be straight forwardly implemented in Modelica.

For this reason a separate Matlab tool has been implemented that allows the drawing of simple FE

meshes in a graphical way (Fig. 6). The Matlab tool then automatically generates the corresponding Modelica model with all necessary geometric parameters filled in. The students can check with an example that this generation process is straightforward and produces correct results.

The same problem occurs when model assembly and system simulation has been performed. Because Modelica has no built in concepts to represent spatial data the simulation result is a collection of time courses of all state variables of the system. It makes little sense to visualize these results by time course plots as Modelica does.

For this reason a post processing tool has been implemented in Matlab which takes the Modelica output and performs a spatial visualization in geometrical space (Fig. 7). Again, this is a purely technical transformation which is completely transparent to the students.

9 Conclusions

Summarizing, a complete FE tool for heat conduction and elastic deformation in 2D has been implemented together with the corresponding pre- and post-processing tools. More details can be found in [6]. All materials are available and can be downloaded from www.simtec.mb.uni-siegen.de.

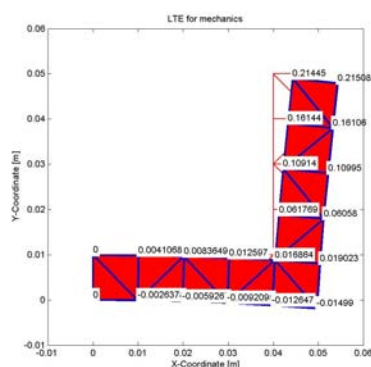


Fig. 6: MATLAB preprocessing tool for model geometry specification

It must be emphasized that these tools are designed for teaching and are not intended as a substitution for a professional FE tool. Using these tools it is possible to establish a short introduction to FEs based on a previous knowledge of modular modeling with Modelica. More details and more theory can be added later on. Experiences from the lecture given at the University of Siegen are quite encouraging.

As an outlook, even advanced concepts of the FEM can be explained within the Modelica framework. This holds e.g. for more sophisticated elements like hexagonal elements or higher order elements. Even non linear elements or elements with additional memory variables (e.g. for plastic materials) can be introduced.

Also the Finite Volume Method (FVM) can be introduced in complete analogy to the approach taken here. The important difference then is that the interface points of the triangles now lie on the edge center points. Masses or heat capacities can be handled more easily in the FVM because they directly correspond to the triangle domain. Another aspect is automatic grid generation, which is rather a topic of computer geometry than physical modeling.

On the other hand it turns out very quickly that Modelica is not the tool to do FEs if meshes with more than a few elements are considered. In this case the performance rapidly drops down because many FE specific simplification or preconditioning methods for the arising equation systems are not available in an all purpose tool. Nevertheless, it is very easy to explain this difference between Modelica and a specialized FE tool to the students. In the Siegen lecture, the Comsol FE tool [7] is finally demonstrated to the students how a specialized tool looks like.

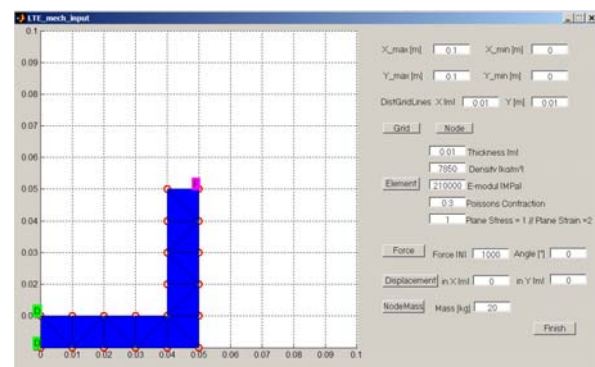


Fig. 7: MATLAB post-processing tool for simulation result visualization.

10 References

- [1] R.K. Livesley. Finite Elements: An Introduction for Engineers. Cambridge Univ. Press 1983.
- [2] S. Larsson, V. Thomee. Partial Differential Equations with Numerical Methods. Springer. 2005.
- [3] P.I. Kattan. MATLAB Guide to Finite Elements. Springer Verlag. 2002.
- [4] P. Fritzson. Principles of Object-Oriented Modeling and Simulation with Modelica. IEEE Press. 2004.
- [5] P.J. Ashenden and G.D. Peterson, The System Designers Guide to VHDL-AMS. Elsevier. 2002.
- [6] M. Treude. Realisierung Finiter Elemente in Modelica. Studienarbeit. Univ. of Siegen. 2002.
- [7] W.B.J. Zimmerman. Process Modelling and Simulation with Finite Element Methods. World Scientific. 2004.