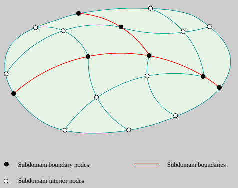
Chapter: 1

MEK4560
The Finite Element Method in Solid Mechanics II

(January 25, 2008)

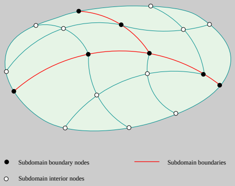
Torgeir Rusten

(E-post:torgeiru@math.uio.no)



Content

Department of Mathematics University of Oslo



Mek 4560
Torgeir Rusten

Contents

Page 2 of 14

Go Back

Close

Quit

1. Introduction to MEK4560

This course is a continuation of *MEK4550, The Finite Element Method in Solid Mechanics I*. The purpose of this course is twofold, give a practical introduction to the us of a commercial finite element program and to introduce some theory and methods relevant for structural analysis.

The Finite Element programs used in the course Ansys and Comsol. The choice of software to use in the homework assignment is up to you.

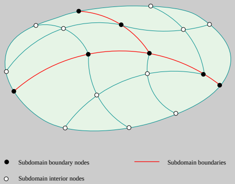
On the theoretical side, we derive element methods for structural analysis, in particular, an alternative formulation of the beam element, and plate and shell elements. Some modification of the element formulations relevant to structural analysis will be introduced. In addition to static analysis we will introduce dynamic analysis and linearized buckling. Linear adaptive analysis and some nonlinear methods will be discussed briefly.

The software packages available for the exercises are

- ANSYS
- *COMSOL Multiphysics*

the choice is yours.

Department of Mathematics
University of Oslo



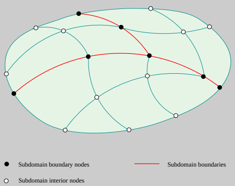
Contents
◁◁ ▷▷
◁ ▷
Page 3 of 14
Go Back
Close
Quit

ANSYS 10.0 is a “classical” system for doing structural analysis using the Finite Element Methods. It is a major task to master, and this course will only give an introduction. This should give you some insight in mechanics and the use of Finite Element methods.

COMSOL Multiphysics takes on a different approach in the design of the software. It is a general purpose Finite Element system, with a number of partial differential equations. It also has a Structural analysis module.

The following analysis are routine in a Finite Element analysis of construction:

- linear static analysis (can time dependent effects be neglected?)
- buckling analysis, $\rightarrow$ eigenvalue analysis, (stability?)
- modal analysis, $\rightarrow$ eigenvalue analysis, (eigenfrequencies or resonance?)
- modal superposition, $\rightarrow$ dynamic analysis, (reduced dynamic system?)
- nonlinear analysis, (large deformations, plasticity, contact analysis, tension only?)
  - static analysis,
  - dynamic analysis,
- restart analysis



1.1. Minimum Potential energy

We now recall a mathematical model of a solid body. A deformation of a body moves a point (x, y, z) to the point $(\phi_x(x, y, x), \phi_y(x, y, z), \phi_z(x, y, z)$. The displacement is

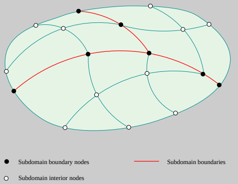
$$\begin{aligned}\mathbf{u}(x, y, z) &= (u(x, y, z), v(x, y, z), w(x, y, z)) \\ &= (\phi_x(x, y, z), \phi_y(x, y, z), \phi_z(x, y, z)) - (x, y, z)\end{aligned} \tag{1.1}$$

The strains are

$$\epsilon(\mathbf{u}) = \begin{pmatrix} \varepsilon_x(\mathbf{u}) \\ \varepsilon_y(\mathbf{u}) \\ \varepsilon_z(\mathbf{u}) \\ \gamma_{xy}(\mathbf{u}) \\ \gamma_{yz}(\mathbf{u}) \\ \gamma_{zx}(\mathbf{u}) \end{pmatrix} = \begin{pmatrix} \partial_x u \\ \partial_y v \\ \partial_z w \\ \partial_y u + \partial_x v \\ \partial_z v + \partial_y w \\ \partial_z u + \partial_x w \end{pmatrix} \quad \text{or} \quad \epsilon(\mathbf{u}) = \begin{pmatrix} \epsilon_x(\mathbf{u}) \\ \epsilon_y(\mathbf{u}) \\ \gamma_{xy}(\mathbf{u}) \end{pmatrix} = \begin{pmatrix} \partial_x u \\ \partial_y v \\ \partial_y u + \partial_x v \end{pmatrix} \tag{1.2}$$

the latter for two dimensional bodies. Furthermore, the stress strain relations for a homogen, homogen isotropic material is

$$\sigma = E\varepsilon(\mathbf{u}). \tag{1.3}$$



Here

$$\sigma = \begin{pmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{zz} \\ \tau_{xy} \\ \tau_{yz} \\ \tau_{zx} \end{pmatrix} \tag{1.4}$$

and

$$E = C \begin{pmatrix} 1-\nu & \nu & \nu & 0 & 0 & 0 \\ \nu & 1-\nu & \nu & 0 & 0 & 0 \\ \nu & \nu & 1-\nu & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2}(1-2\nu) & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2}(1-2\nu) & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{2}(1-2\nu) \end{pmatrix} \tag{1.5}$$

The constant

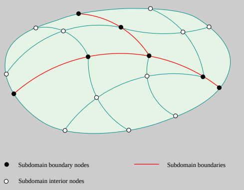
$$C = \frac{E}{(1+\nu)(1-2\nu)} \tag{1.6}$$

E is the elastic modulus and ν is Poisson's ratio.

The potential energy function for linearized elasticity is

$$\Pi(\mathbf{u}) = \frac{1}{2}\int_V \epsilon(\mathbf{u})^T E \epsilon(\mathbf{u})\, dV - \int_V \mathbf{u}^T F\, dV - \int_{S_t} \mathbf{u}^T \Phi\, dS \tag{1.7}$$

Department of Mathematics
University of Oslo



Mek 4560
Torgeir Rusten

Contents
Page 6 of 14
Go Back
Close
Quit

where E is a matrix of material coefficients defining the stress-strain relations, F are body forces and Φ are traction forces. In linearized elasticity the task is to minimize the potential energy functional subject to suitable boundary conditions on the displacement.

In the finite element method the body V is subdivided into elements with local basis functions. This gives a minimization problem over a vector of degrees of freedom, or nodal values, and this is equivalent to solving a linear system

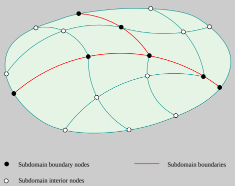
$$KU = R \tag{1.8}$$

where K is the stiffness matrix, R is composed of the given body forces, surface tractions and displacement boundary conditions. Moreover, U is the vector of degrees of freedom. Given the nodal values in U a vector function of displacements can be contructed, hopefully a good approximation to the displacement $\mathbf{u}$.

Having obtained the linear system, the following questions arise:

- Can it be solved?
- Is the computed displacement function a good approximation of $\mathbf{u}$?
- Can it be used to analyze a real structure?

We'll discuss these topics throughout the course.

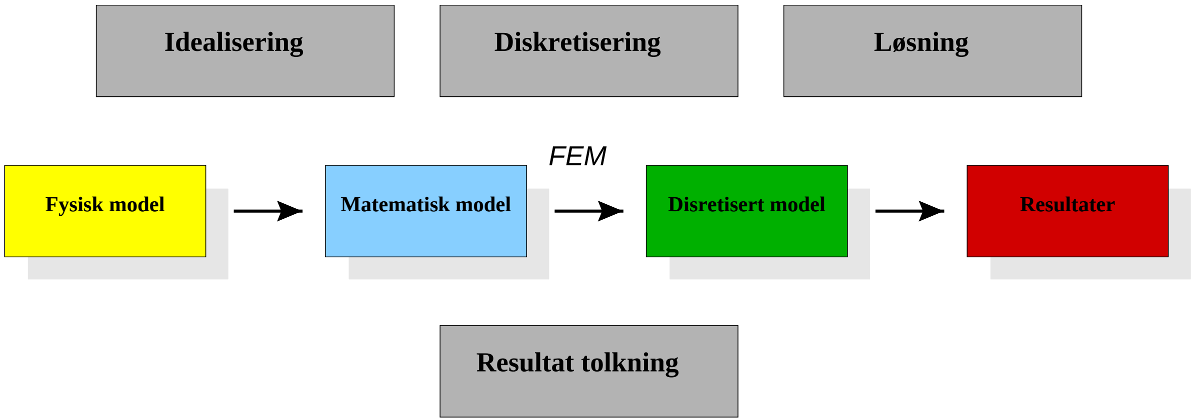


Contents
Page 7 of 14
Go Back
Close
Quit

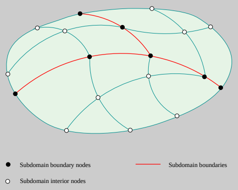
In the anlysis process a number of choices must be made, and the quality of the analyses depend on the choices:

1. Physical system.
2. Mathematical model.
3. Numerical methods.
4. Interpretation of results.

This is illustrated in the figure below. The first item is beyond the scope of this course.



Department of Mathematics University of Oslo



In order to obtain good quality and reliable analysis results the follwoing is important:

- good insight in the physics
- good knowledge of the Finite Element Method
- know the Finite Elements well
- understand how a Finite Element program works
- sceptical evaluation of results.

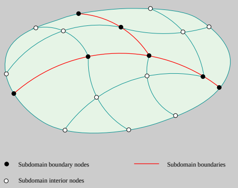
1.2. The analysis process

Classical Finite Element systems cosist of the following parts

1. Preprocessing, i.e. Geomery, mesh, material data, loads.
2. Solution. Linear system, eigenvalues and vectors, time stepping.
3. Postprocessing: Presentation of results, also derived results.

Below we brifely consider these steps, and relate them to the potential energy. Note that some programs may to two or all three steps in an integrated fashion.

Department of Mathematics
University of Oslo



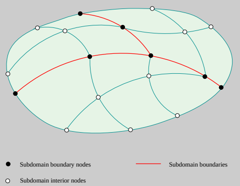
Mek 4560
Torgeir Rusten

Preprocessing: Lets consider the transition from a physical model to a mathematical model and then to a discrete Finite Element model:

- geomatric model, V and S_t
- material data, E
- weak form, 3D solid, plane stress, plate, shell, beam, truss
- boundary condtions:
  - kinematic conditions (Dirichlet, essential ...),
  - loads (Neumann, natural ...).

In order to run an analysis the mathematical model, i.e. finite element discretization, must be communicated to the analysis program. Note:

- unexpected results from the analysis software are a result of user errors



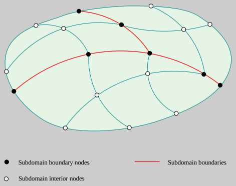
Processor: In addition to the finite element discretization one has to decide on the type of analysis, depending on the relevant physics, and communicate it to the Finite Element program. The analysis may cosist of solution of a linear system, an eigenvalue problem, direct time integraiton, an iterative solution of a nonlinear problem, etc.

$$\mathit{problem}\begin{cases}\mathit{static} & \begin{cases}\mathit{linear}\\ \mathit{nonlinear}\end{cases}\\ \mathit{dynamic}\end{cases}$$

$$\mathit{dynamic}\begin{cases}\mathit{time\ domain} & \begin{cases}\mathit{modal\ analysis}\\ \mathit{direct\ time\ integration}\end{cases}\\ \mathit{frequency\ domain}\end{cases}$$

$$\mathit{nonlinear}\begin{cases}\mathit{geometry}\\ \mathit{material}\\ \mathit{boundary\ condtions} & \begin{cases}\mathit{kinematic}\\ \mathit{loads}\end{cases}\end{cases}$$

Department of Mathematics University of Oslo



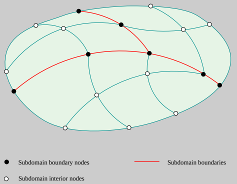
Mek 4560
Torgeir Rusten

Postprocesing: After the solution phase relevant quantites must be accesible to the user. Some examples of relevatn quantities are:

- displacements, velocities, accelerations
- stress, strain,
- reaction forces,
- errors,
- $\vdots$

Can relevant questions be answered? Is the construction acceptable? Is it good design?

Department of Mathematics University of Oslo



Mek 4560
Torgeir Rusten

1.3. Complexity of numerical algorithms

Here we briefly comment on the complexity of algorithms for the solution of linear systems.

A dense matrix K of dimension n by n, occupies n^2 floating point numbers. If double precision is used there is 8 bytes pr. number. A direct method for the solution of linear systems factor the matrix into a product of an upper and a lower triangular part. The number of floating point operations are proportional to $n^3/3$.

If we refine the mesh, e.g. by dividing a brick into 8 the number of unknowns are multiplied by 8, the amount of memory is multiplied by $8^2 = 64$ and the amount of floating point operations are multiplied by $8^3 = 512$!

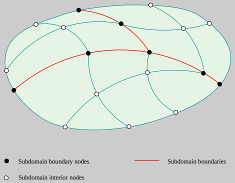
If $n = 10^6$, not uncommon in finite element computations, the amount of memory to hold a full matrix is

$$8(10^6)^2 = 8000GB \tag{1.9}$$

A fast pc today can compute 10^10 floating point operations pr. second, i.e. 10 Gflops. The number of float point operations required to do the factorization for a dense matrix is $(10^6)^3/3 = 10^18/3$. The computation time is approximately one year! (In practice is will be much more due to memory management.

Clearly, we have to to better. Note that linear systems arising from finite element systems are symmetric and sparse. Each node in a mesh is connected to a few nodes, hence the linear system is sparse. (Why?). Thus the number of nonzero elements in a matrix row is Cn for a constant C, where C can range from 5 to a few hundred. With $C = 60$ the amount of storage is

Department of Mathematics
University of Oslo



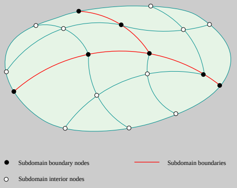
Mek 4560
Torgeir Rusten

$8\ 60\ 10^6 = 480$ Mb Thus the storage of a sparse matrix is not a problem on modern computers.

The factorization algorithm introduce fill in in the matrix, i.e. some zero elements becomes nonzero during the factorization. The amount of nonzeros introduced depend on the numbering of the degrees of freedom. With clever reordering methods the amount of floating point operations in a factorization are $Cn \log n$, for some C independent of n. With n as above $\log n$ is almost 14 thus $C \log n = 1000$ is realistic. This means around 6 GB of memory.

Note that even if computers are fast today efficient algorithms are equally important in order to do simulations on computers.

Department of Mathematics
University of Oslo



Mek 4560
Torgeir Rusten