Computational Materials Modeling FHLN05
Computer lab 2013

Non-linear finite element problems

Motivation

In the basic Finite Element (FE) course, the analysis is restricted to materials where the relationship between stresses and strains is linear. For most real applications, this is not the case and nonlinear and inelastic effects enters the formulation. In this assignment you will write a FE program that is capable of dealing with such effects. The structure of a non-linear finite element program will be examined. The understanding of the structure of a non-linear finite element program is absolutely vital since it is virtually the same irrespectively of the cause of the non-linear behavior. One of the most extensively used algorithms for solving the global equilibrium equations in FE-analysis is the Newton-Raphson algorithm, which possesses a good convergence rate. The main purpose of this assignment is to gain understanding of the different aspects involved when establishing a non-linear FE program. It is emphasized that exactly the same Newton-Raphson algorithm can be used in the next assignment.

General instructions

Two assignments will be carried out during the course: one non-linear elastic problem and one elasto-plastic problem. Since a similar program structure will be used in the elasto-plastic problem as in the elastic problem it is of great importance that the first problem is solved properly before you start solving the second.

This first assignment should be done during a computer session and needs to be approved before leaving. You are not required to hand in a formal report but in order to pass the course it is required that you complete the task and get an approval from your teaching assistant. It is recommended that you work in groups of two but it is allowed to work alone.

Problem description

A toy for dogs is constructed from a nonlinear elastic, rubberlike material and has the geometry shown in Fig. 1. The geometry is given in meters and the toy has a thickness of 3 cm. The dogs pull at different ends of the toy, causing it to elongate. The nonlinear

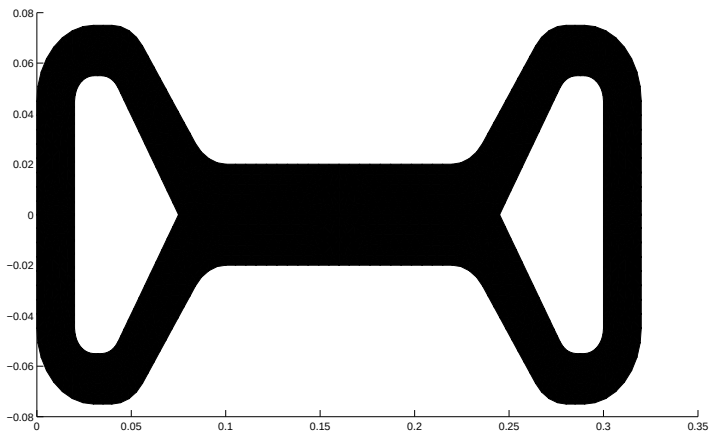


Figure 1: Geometry of dog toy.

elastic material is assumed to be isotropic and the response independent of the third strain invariant, i.e. the stress strain relations are given by

$$\sigma_{kk} = 3K\epsilon_{kk} \qquad s_{ij} = 2Ge_{ij}$$

where σ_{ij} and ϵ_{ij} are the stress and strain tensors, respectively. Moreover, $s_{ij} = \sigma_{ij} - 1/3\delta_{ij}\sigma_{kk}$ and $e_{ij} = \epsilon_{ij} - 1/3\delta_{ij}\epsilon_{kk}$ are the deviatoric stress and strain tensors, respectively.

From experimental tests it is concluded that in the small-strain regime, the shear modulus of the material varies with the strains according to the relationship

$$G = A \exp\left(B\tilde{J}_2\right),$$

where the invariant $\tilde{J}_2$ is given by $\tilde{J}_2 = 1/2e_{ij}\, e_{ji}$. K and G are the bulk and shear modulus respectively. It can be assumed that the bulk modulus $K = \frac{E}{3(1-2\nu)}$ remains constant. The initial elastic modulus of the material is $E = 10$ MPa and the Poisson's ratio is set to $\nu = 0.45$ as the material shows high resistance to volume changes. The parameter B is taken as $B = 1 \cdot 10^5$ while the value of A needs to be determined.

Instructions

The task is to perform a calculation of the nonlinear response of the given structure. To solve the problem the CALFEM-toolbox should be used. In CALFEM, certain general FE-routines are already established and the task is to establish the extra routines needed to solve the nonlinear elastic boundary value problem.

The geometry for the problem is given in the file `datalabb_2013_geom.mat`, available on the course homepage. Note that due to symmetry, only one fourth of the structure needs to be modelled. The mesh is shown in Fig. 2. All quantities are given in SI units.

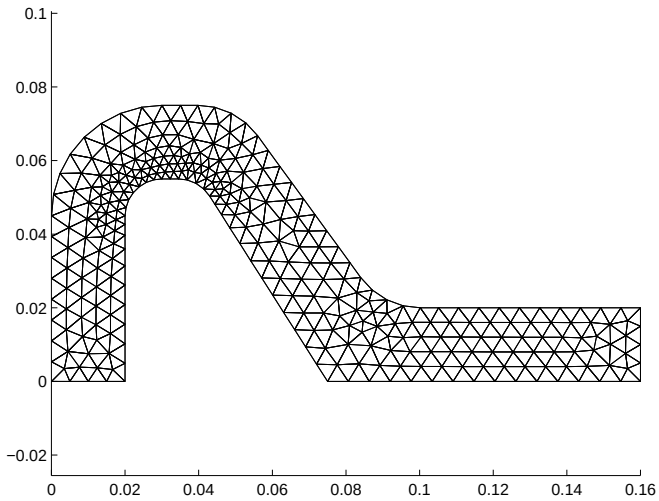


Figure 2: Dog toy geometry.

The file contains, in addition to the geometry files, boundary conditions and an external load increment vector. Table A.1 in appendix A describes the different files given by `datalabb_2013_geom.mat`.

Work out the following steps;

a) Derive expressions for the stress σ_{ij} and the tangent operator D^t_{ijkl}. Some hints are found in appendix A.

b) Compare your expression for the tangent with the linear expression in eq. (4.89) in the course book. The tangents should coincide for zero strain. What is the value of A? Use this value for the parameter.

c) Write element routines `my_stress.m` and `my_tangent.m` that calculate the stress and tangent. See appended manuals for the structure of the functions.

d) Construct a FE-program using a force controlled Newton-Raphson algorithm to solve the global equilibrium equation.

e) Plot the force-displacement curve for force controlled loading. Load the structure so that the final applied load is approximately 8 N.

f) Plot the force-displacement curve during a load cycle (loading an unloading) for displacement controlled loading. Load the structure up to 3 mm and back.

The toy is assumed to be loaded well into the nonlinear region. Simple constant strain triangle elements should be used in the finite element calculations. The calculations should be done for plane strain condition for simplicity.

For the global equilibrium loop the Newton-Raphson method should be used. An appropriate amount of Newton-Raphson iterations is around 2-6 in each loadstep. The stress and tangent stiffness routines should be contained in separate Matlab functions so that they can easily be reused later on.

Although you do not need to write a formal report, it is required that you complete all the tasks above and get an approval from your teaching assistant before leaving. After step a) is completed it is recommended that you consult the tutor before you proceed to program the element routines.

Preparations

Before the lab you should;

- Read through the lab instructions
- Read pages 423-445 in the course book
- *On only a few pages* present the derivations of the stress and tangential stiffness tensors as well as the value of the parameter A. Present each component of the tangent stiffness tensor in the following format;

$$D^t_{ijkl} = \begin{bmatrix} D_{1111} & D_{1122} & D_{1112} \\ D_{2211} & D_{2222} & D_{2212} \\ D_{1211} & D_{1222} & D_{1212} \end{bmatrix}$$

The derivations should be approved by the lab tutor *before* the computer session.

Implementation

After starting Matlab, begin with adding CALFEM into your Matlab library using the `editpath` command.

Write the two functions `my_stress.m` and `my_tangent.m`. All Matlab functions begin with the following structure;

```
function output = function name (input)
```

See the manuals for the specific inputs and outputs to use for the the two routines.

Check your element routines by loading the file `checkroutines.mat`. This file contain an example of a strain tensor `eps` and the corresponding stress `sigma` and tangential stiffness matrix `Dt`. Use the strain tensor together with the correct material parameters as input to your routines and check that they agree with the stress and stiffness matrices supplied by the file.

The finite element mesh for the dog toy `datalabb_2013_geom.mat` is available at the course homepage `www.solid.lth.se`. The file can be loaded into the Matlab workspace by typing e.g. `load datalabb_2013_geom.mat`. You can view the geometry by using the CALFEM routine `eldraw2`;

```
eldraw2(ex,ey,[1 2 0],1:nbr_elem)
```

Main program

When writing the main program use the structure for the Newton-Raphson algorithm sketched in Table 1 and follow the instructions given for the steps `a)-l)`.

```
% a) define material parameters and b) initiate quantities
% c) calc. initial tangential stiffness matrix and stiffness matrix

% load-loop
for n=1:nbr_step

    % d) apply load
    % e) calc. residual

    % iteration-loop
    while res>TOL

        % f) solve the linearised residual

        for j=1:nbr_elem

            % g) update the strains
            % h) calc. stresses and tangential stiffness matrix
            % i) calc. element internal forces and stiffness matrix
            % j) assembly element matrices to global matrices

        end

        % k) update the residual
        % l) set boundary conditions to zero at appropriate places

    end
end
```

Table 1: Newton-Raphson algorithm

Description of steps

a) Define your material parameters and an element property vector `ep` that is needed for some of the CALFEM routines.

   `ep = [ptype t]` where;

   `ptype =` analyse type (ptype=2 for plane strain) and

   `t =` thickness of the specimen.

   Additional quantities, like the size of the tolerance required for the Newton-Rapshon loop and number of load steps should also be set here.

b) Initiate the following quantities;

```
a = zeros(nbr_dof,1);           %Displacement vector
K = zeros(nbr_dof);             %Stiffness matrix
f = zeros(nbr_dof,1);           %External force vector
f_int = zeros(nbr_dof,1);       %Internal force vector
eps = zeros(nbr_elem,3);        %Element strain matrix
```

c) Calculate an initial elastic tangential stiffness matrix and stiffness matrix using the CALFEM routines `hooke` and `plante` respectively. See hint for assembling the stiffness matrix in point `j)`.

d) For each load step, apply the incremental load. In the case of force-controlled loading, add a incremental external force `df` to the total external force `f` in each load step. In case of displacement-controlled loading the external force is zero and the load is controlled by the boundary conditions given in `bc_sd`. Use this vector when solving your equations but don't forget to reset the second column afterwards, you only want to make one displacement in each load step.

e) Calculate the residual used for checking convergence of the iterations.

f) Solve the linearised residual to get the incremental displacements using the CALFEM routine `solveq`.

g) Update the strains using the CALFEM routine `plants`.

h) Calculate stresses and tangential stiffness matrix using your own routines `my_stress` and `my_tangent`.

i) Calculate internal forces and stiffness matrix for each element using the CALFEM routines `plantf` and `plante` respectively. Don't forget to reset the global matrices in each iteration before assembling them again, otherwise you will just continue adding numbers in an old matrix/vector.

`j)` Assembly element matrices to global matrices using CALFEMs `assem` routine or add the following lines inside your element-loop;

```
indx = edof(j,2:end);
K(indx,indx)=K(indx,indx)+Ke; f_int(indx)=f_int(indx)+fe';
```

`k)` Update residual.

`l)` Set boundary conditions to zero at appropriate places.

Finally for plotting the results the CALFEM routine `eldisp2` is useful.

Happy programming!

Appendix A

A.1 Variables in the supplied geometry files

Table A.1: Variables in the supplied geometry file.

Variable	Description	Size
`bc_ld`	Boundary cond. during force-controlled loading	
`bc_sd`	Boundary cond. during disp.-controlled loading	
`coord`	Coordinates of elements	[`nbr_node`×2]
`dof`	Degrees of freedom	
`edof`	Element topology matrix	[`nbr_elem`×7]
`ex`	Element x-coordinates	[`nbr_elem`×3]
`ey`	Element y-coordinates	[`nbr_elem`×3]
`df`	External force increment vector	[`nbr_dof`×1]
`nbr_dof`	Number of degrees of freedom	
`nbr_elem`	Number of elements	
`nbr_node`	Number of nodes	
`plot_dof`	Degree of freedom used for plotting	

A.2 Hints

The tangential stiffness tensor is found by differentiating the stress tensor with respect to the strain tensor according to

$$D^t_{ijkl} = \frac{\partial \sigma_{ij}}{\partial \varepsilon_{kl}}$$

In performing the calculations the derivative of the second strain invariant might be useful

$$\frac{\partial \tilde{J}_2}{\partial \varepsilon_{ij}} = e_{ij}$$

Finally, when differentiating symmetric tensors – as σ_{ij} and ε_{ij} – with respect to themselves ,symmetry is preserved by differentiating according to

$$\frac{\partial \varepsilon_{ij}}{\partial \varepsilon_{kl}} = \frac{1}{2}\left(\delta_{il}\delta_{jk} + \delta_{ik}\delta_{jl}\right) \quad \text{and} \quad \frac{\partial \sigma_{ij}}{\partial \sigma_{kl}} = \frac{1}{2}\left(\delta_{il}\delta_{jk} + \delta_{ik}\delta_{jl}\right)$$

where δ_{ij} is the Kronecker delta.

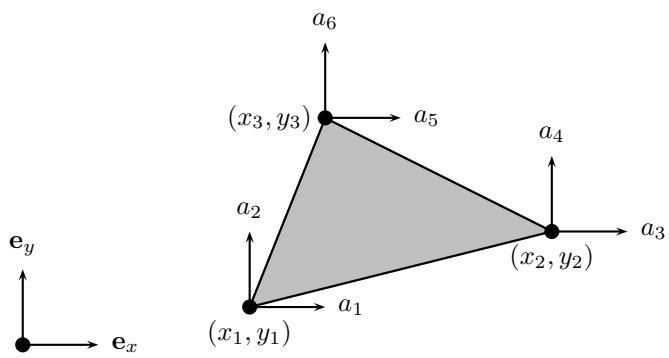
A.3 Manuals

Function:

my_stress

Purpose:

Compute the stress in a triangular 3 node element in plane strain.



Syntax:

sigma=my_stress(eps,mp)

Description:

my_stress provides the stresses σ_{11}, σ_{22}, σ_{33} and σ_{12} for a triangular 3 node element. The element strains ε_{11}, ε_{22} and γ_{12} for are supplied by eps;

$$\text{eps} = \begin{bmatrix} \varepsilon_{11} \\ \varepsilon_{22} \\ \gamma_{12} \end{bmatrix}$$

The material parameters are defined in a vector mp. Calculate and send out all four stress components in the following format;

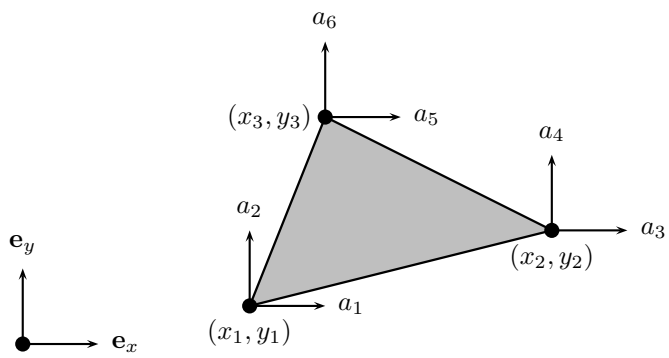
$$\text{sigma} = \begin{bmatrix} \sigma_{11} \\ \sigma_{22} \\ \sigma_{33} \\ \sigma_{12} \end{bmatrix}$$

Function:

my_tangent

Purpose:

Compute the material tangent matrix D for a triangular 3 node element.



Syntax:

[D]=my_tangent(eps,mp)

Description:

my_tangent computes the material tangent $\mathbf{D}$ for a triangular 3 node element. Inputs to the function are the strains eps and material parameters as defined in `my_stress`.

Store the following components of the tangential stiffness matrix;

$$\mathsf{D} = \mathbf{D}^t = \begin{bmatrix} D^t_{1111} & D^t_{1122} & D^t_{1112} \\ D^t_{2211} & D^t_{2222} & D^t_{2212} \\ D^t_{1211} & D^t_{1222} & D^t_{1212} \end{bmatrix}$$