Division of Solid Mechanics

ISRN LUTFD2/TFHF--04/5106--SE (1-83)

ON THE INTERNAL ARCHITECTURE OF BONE BASED ON A CONTINUUM MODEL

Master's Dissertation by

Zoltán Habony

Supervisors
Ingrid Svensson, Div. of Solid Mechanics
Magnus Tägil, Dep. of Orthopaedics, Lund University Hospital
Ian McCarthy, Biomechanics Laboratory, Dep. of Orthopaedics, Lund University Hospital

Acknowledgement

This master's dissertation was carried out at the Division of Solid Mechanics at Lund University, Sweden under the supervision of Dr. Ingrid Svensson, Division of Solid Mechanics, Dr. Magnus Tägil, Department of Orthopaedics and Dr. Ian McCarthy, Biomechanics Laboratory, Department of Orthopaedics.

I would like to thank Ingrid for priceless help and encouragement during the work and Magnus and Ian for great help in the area of medicine. I would also like to thank Magnus Harrysson and Professor Matti Ristinmaa for their patient help.

Abstract

This master's dissertation is done in cooperation with the Division of Solid Mechanics at Lund University and the Orthopaedic Department at Lund University Hospital. The thesis takes over where another master's thesis written by Magnus Harrysson in 2002 at the Division of Solid Mechanics ends.

The idea of bone remodeling goes back to the 19:th century when Julius Wolff suggested that the trabecular directions coincide with the principal stress trajectories. Many theories have since then been proposed. The motivation for such a work is to understand the macro structure of bone when changes in the loading environment have occurred. For instance, when an implant is put in surgically to replace the femoral head after a fracture, the load distribution near the interface between the implant and the bone is changed dramatically. Changes like these trigger the remodeling process.

In this thesis an orthotropic linear hyper elastic material model is used to cover the behavior of bone. In the first part, changes in the orientation of the material's privileged frame, i.e. the directions of orthotropy, were investigated. The direction belonging to the largest stiffness is assumed to be the preferred trabecular direction. It was assumed that this frame always remains right handed orthonormal. It is shown that the material's privileged frame is re-oriented so that the material direction with the largest stiffness is parallel with the direction of the largest absolute principal strain.

In the second part an evolution law for the material stiffness properties is proposed. A stimulus based on the strain tensor is defined and an algorithm based on an implicit Euler method is used to update the material properties. In this part it is shown that the elastic properties are updated in a realistic manner.

Contents

Chapter 1

Introduction

1.1 Background

Understanding the behavior of the macro structure of bone has been the goal of researchers worldwide for centuries. In 1892 Wolff suggested that the trabecular directions coincide with the principal stress trajectories [18]. This became the famous Wolff's law and to this day it influences all major work done in the field. In a recent study Harrysson [7] investigated the ability of bone adaptation considering the bone as isotropic. He showed that the elastic modulus is updated to fit the current load situation. Bone is however a highly anisotropic material. Like for example wood it is stiffer in some directions than others. This work focuses on extending Harrysson's work including orthotropy in the ability of adaptation.

1.2 Objectives

The objectives for this thesis is first to make a literary study of existing bone adaptation models using an orthotropic material model, and secondly, based on the studies formulate and implement an algorithm that updates both the material directions and the elastic properties. Last, an evaluation of the results of the implementation is performed.

Chapter 2

Macroscopic description of bone and mechanical properties

2.1 Macroscopic description of bone

Bone is a highly complex structure that consists of an organic component (cells and the matrix) and an inorganic or mineral component which gives the tissue its hardness. The skeleton has several functions. They are to support the body against external forces, protect the vital internal organs and act as a lever system to transfer forces. Bone is also responsible to form blood cells and to store calcium. Bone tissue is constantly undergoing reconstruction during a lifetime. It dissolves (resorption) locally and new tissue forms (deposition) in its place. One stimulating factor for bone remodeling is the mechanical loading exerted on the bone by the muscles. As ageing progresses, some people suffer the effects of osteoporosis. This is also a form of remodeling where the bone structure becomes more orientated and less able to carry loads in unusual directions [8]. Another factor may be of chemical nature. The macroscopic behavior, i.e. things visible to the naked eye, of bone is built up from microscopic parts but this description focuses only on the macro level of bone.

The bones in the adult skeleton have two main structural components: the cortical and the cancellous or trabecular bone. Cortical or compact bone, Figure 2.1 is the solid, dense material comprising the external surfaces of long bones. It is strong, solid and resistant to bending.

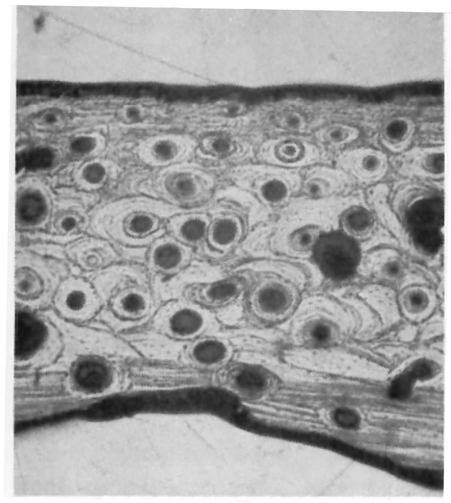

Figure 2.1 The structure of cortical bone.

Cancellous or spongy bone, Figure 2.2 consists of thin bony spicules called trabeculae. This type of bone can be found at the ends of long bones, the epiphysis. These trabeculae have been observed to orient themselves in the direction of the forces applied. Cortical or compact bone can be found in the middle part of long bones, the diaphysis.

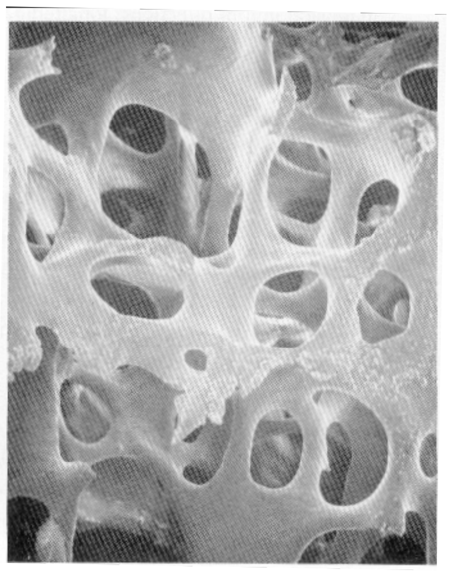

Figure 2.2 The trabecular pattern observable in cancellous bone.

Irregular spaces filled with soft tissue occur between the trabeculae, reducing the weight of the bone and giving it shock absorbing properties. For more details see for instance Nigg et al. [10].

2.2 Mechanical properties of bone

The structure of a skeletal bone is adapted to the load it is supposed to carry. When the mechanical properties of bone are to be examined using for example machine testing [17] or ultra sound [1], different results are obtained depending on both the orientation of the specimen and the measuring technique. In this thesis a linear elastic orthotropic material model is used and some measured properties can be seen in Table 2.1 for cortical bone.

Table 2.1 Some mechanical properties of cortical bone

Bone Type	Human Tibia	Human Femur	Human Femur
Method of Measurement	Mechanical Testing	Ultrasound	Ultrasound
Reference	Knets [17]	Van Buskirk et al. [17]	Ashman et al. [1]
E_1 (Gpa)	6.91	13.0	12.0
E_2 (Gpa)	8.51	14.4	13.4
E_3 (Gpa)	18.4	21.5	20.0
G_{12} (Gpa)	2.41	4.74	4.53
G_{13} (Gpa)	3.56	5.85	5.61
G_{23} (Gpa)	4.91	6.56	6.23
ν_{12}	0.488	0.37	0.376
ν_{13}	0.119	0.24	0.222
ν_{23}	0.142	0.22	0.235
ν_{21}	0.622	0.42	0.422
ν_{31}	0.315	0.40	0.371
ν_{32}	0.307	0.33	0.350

In the following calculations the values form Ashman et al. in the last column have been used. Later it will be clear that the choice of values is not critical for the calculations. It will be assumed that the mechanical properties are the same for cancellous as for cortical bone. This assumption may seem quite rude but literature studies show that variations of the elastic properties are very large due to difference in tissue, anatomical location and individual. For example Zysset et al. [19] tested the isotropic elastic modulus of trabecular bone in human femur using nanoindentation. His studies show that the elastic moduli range from $6.9 \pm 4.3\,\text{GPa}$ for a 74 year old female to $15.9 \pm 5.1\,\text{GPa}$ for a 69 year old female.

Chapter 3

The constitutive law

The goal of this chapter is to derive the invariant formulation of orthotropic elasticity. Invariant means that the formulation takes the same shape irrespective of the coordinate system chosen. A fully anisotropic material has no planes of symmetry and needs a total of 21 different material parameters. This would off course be the ideal way of describing bone because it can describe any material but it would increase the complexity to an unnecessary level. If three different planes of symmetry are chosen the orthotropic formulation can be derived. This reduces the independent material parameters to nine. It will later be clear that three of these parameters will be considered constant and the other six will be updated. The orthotropic formulation can be further simplified. By assuming that every plane in the material is a symmetry plane the number of independent material parameters is reduced to two. This is called isotropy and it is the simplest way to describe linear elasticity. However the structure of bone clearly shows that isotropy is not suitable to describe its elastic behavior. The orthotropic formulation was chosen because predictions of properties in different directions can be made. To derive the constitutive law governing the relations between stresses and strains in a hyper elastic orthotropic material our starting point will be the irreducible representation of the problem

$$\begin{aligned}\mathbf{T} = \alpha_1\mathbf{M}_1 + \alpha_2\mathbf{M}_2 + \alpha_3\mathbf{M}_3 + \alpha_4(\mathbf{M}_1\mathbf{E} + \mathbf{E}\mathbf{M}_1) + \alpha_5(\mathbf{M}_2\mathbf{E} + \mathbf{E}\mathbf{M}_2) + \\ + \alpha_6(\mathbf{M}_3\mathbf{E} + \mathbf{E}\mathbf{M}_3) + \alpha_7\mathbf{E}^2\end{aligned} \tag{3.1}$$

$$\alpha_i = \alpha_i\left(tr\mathbf{M}_1\mathbf{E}, tr\mathbf{M}_2\mathbf{E}, tr\mathbf{M}_3\mathbf{E}, tr\mathbf{M}_1\mathbf{E}^2, tr\mathbf{M}_2\mathbf{E}^2, tr\mathbf{M}_3\mathbf{E}^2, tr\mathbf{E}^3\right)$$

where the response $\mathbf{T}$ is the stress tensor, $\mathbf{E}$ is the strain tensor and $\mathbf{M}_1, \mathbf{M}_2, \mathbf{M}_3$ are the structural tensors

$$\mathbf{M}_1 = \mathbf{v}_1 \otimes \mathbf{v}_1 \quad ; \quad \mathbf{M}_2 = \mathbf{v}_2 \otimes \mathbf{v}_2 \quad ; \quad \mathbf{M}_3 = \mathbf{v}_3 \otimes \mathbf{v}_3 \tag{3.2}$$

where $(\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3)$ are the privileged directions of orthotropy. The operation $\otimes$ is called dyad and is interpreted in index notation as

$$M_{ij} = v_i v_j$$

At this stage the representation (3.1) describes any material response since no restrictions have been made. In our case however the material is assumed to be linear elastic, eliminating the quadratic α_7 term. In this derivation we use the following derivatives

$$\begin{aligned}
&\frac{\partial}{\partial \mathbf{E}}(tr\mathbf{E}) = \mathbf{I} \quad ; \quad \frac{\partial}{\partial \mathbf{E}}\left(tr\mathbf{E}^2\right) = 2\mathbf{E} \quad ; \quad \frac{\partial}{\partial \mathbf{E}}\left(tr\mathbf{E}^3\right) = 3\mathbf{E}^2 \\
&\frac{\partial}{\partial \mathbf{E}}(tr\mathbf{M}_i\mathbf{E}) = \mathbf{M}_i \quad ; \quad \frac{\partial}{\partial \mathbf{E}}\left(tr\mathbf{M}_i\mathbf{E}^2\right) = \mathbf{M}_i\mathbf{E} + \mathbf{E}\mathbf{M}_i
\end{aligned} \tag{3.3}$$

The linear restriction of the representation (3.1) is given by

$$\begin{aligned}
\mathbf{T} = {} & \alpha_1\mathbf{M}_1 + \alpha_2\mathbf{M}_2 + \alpha_3\mathbf{M}_3 + \alpha_4(\mathbf{M}_1\mathbf{E} + \mathbf{E}\mathbf{M}_1) + \alpha_5(\mathbf{M}_2\mathbf{E} + \mathbf{E}\mathbf{M}_2) + \\
& + \alpha_6(\mathbf{M}_3\mathbf{E} + \mathbf{E}\mathbf{M}_3)
\end{aligned}$$

$$\begin{aligned}
\alpha_1 &= a_1 + b_1 tr\mathbf{M}_1\mathbf{E} + c_1 tr\mathbf{M}_2\mathbf{E} + d_1 tr\mathbf{M}_3\mathbf{E} \\
\alpha_2 &= a_2 + b_2 tr\mathbf{M}_1\mathbf{E} + c_2 tr\mathbf{M}_2\mathbf{E} + d_2 tr\mathbf{M}_3\mathbf{E} \\
\alpha_3 &= a_3 + b_3 tr\mathbf{M}_1\mathbf{E} + c_3 tr\mathbf{M}_2\mathbf{E} + d_3 tr\mathbf{M}_3\mathbf{E} \\
\alpha_4 &= a_4 \qquad \alpha_5 = a_5 \qquad \alpha_6 = a_6
\end{aligned} \tag{3.4}$$

where the higher order terms of $\mathbf{E}$ have been ignored. The linear elastic response depends now on 15 independent material constants. When the material is undistorted i.e. $\mathbf{E} = \mathbf{0}$ we expect the response $\mathbf{T}$ also to be $\mathbf{0}$. This gives us

$$a_1 = a_2 = a_3 = 0 \tag{3.5}$$

The number of independent material constants is now reduced to 12. To further reduce the representation we assume the material to be hyper elastic i.e. we recognize that the response is the derivative of a scalar valued strain energy W

$$\mathbf{T} = \frac{\partial W}{\partial \mathbf{E}} \tag{3.6}$$

The representation of the strain energy function is given by

$$W = W\left(tr\mathbf{M}_1\mathbf{E}, tr\mathbf{M}_2\mathbf{E}, tr\mathbf{M}_3\mathbf{E}, tr\mathbf{M}_1\mathbf{E}^2, tr\mathbf{M}_2\mathbf{E}^2, tr\mathbf{M}_3\mathbf{E}^2, tr\mathbf{E}^3\right) \tag{3.7}$$

Inserting (3.7) into (3.6) and making use of the expressions given by (3.3) we obtain

$$\begin{aligned}
\mathbf{T} = {} & \frac{\partial W}{\partial tr\mathbf{M}_1\mathbf{E}}\mathbf{M}_1 + \frac{\partial W}{\partial tr\mathbf{M}_2\mathbf{E}}\mathbf{M}_2 + \frac{\partial W}{\partial tr\mathbf{M}_3\mathbf{E}}\mathbf{M}_3 + \frac{\partial W}{\partial tr\mathbf{M}_1\mathbf{E}^2}(\mathbf{M}_1\mathbf{E} + \mathbf{E}\mathbf{M}_1) + \\
& + \frac{\partial W}{\partial tr\mathbf{M}_2\mathbf{E}^2}(\mathbf{M}_2\mathbf{E} + \mathbf{E}\mathbf{M}_2) + \frac{\partial W}{\partial tr\mathbf{M}_3\mathbf{E}^2}(\mathbf{M}_3\mathbf{E} + \mathbf{E}\mathbf{M}_3) + 3\frac{\partial W}{\partial tr\mathbf{E}^3}\mathbf{E}^2
\end{aligned} \tag{3.8}$$

Comparing (3.8) with the linear form (3.4) and identifying terms gives

$$\begin{cases} \dfrac{\partial W}{\partial tr\mathbf{M}_1\mathbf{E}} = b_1 tr\mathbf{M}_1\mathbf{E} + c_1 tr\mathbf{M}_2\mathbf{E} + d_1 tr\mathbf{M}_3\mathbf{E} \\ \dfrac{\partial W}{\partial tr\mathbf{M}_2\mathbf{E}} = b_2 tr\mathbf{M}_1\mathbf{E} + c_2 tr\mathbf{M}_2\mathbf{E} + d_2 tr\mathbf{M}_3\mathbf{E} \\ \dfrac{\partial W}{\partial tr\mathbf{M}_3\mathbf{E}} = b_3 tr\mathbf{M}_1\mathbf{E} + c_3 tr\mathbf{M}_2\mathbf{E} + d_3 tr\mathbf{M}_3\mathbf{E} \\ \dfrac{\partial W}{\partial tr\mathbf{M}_1\mathbf{E}^2} = a_4 \\ \dfrac{\partial W}{\partial tr\mathbf{M}_2\mathbf{E}^2} = a_5 \\ \dfrac{\partial W}{\partial tr\mathbf{M}_3\mathbf{E}^2} = a_6 \\ \dfrac{\partial W}{\partial tr\mathbf{E}^3} = 0 \end{cases} \tag{3.9}$$

The derivatives (3.9) are subject to the following integrability conditions

$$\begin{cases} \dfrac{\partial}{\partial tr\mathbf{M}_1\mathbf{E}}\left(\dfrac{\partial W}{\partial tr\mathbf{M}_2\mathbf{E}}\right) = \dfrac{\partial}{\partial tr\mathbf{M}_2\mathbf{E}}\left(\dfrac{\partial W}{\partial tr\mathbf{M}_1\mathbf{E}}\right) \\ \dfrac{\partial}{\partial tr\mathbf{M}_1\mathbf{E}}\left(\dfrac{\partial W}{\partial tr\mathbf{M}_3\mathbf{E}}\right) = \dfrac{\partial}{\partial tr\mathbf{M}_3\mathbf{E}}\left(\dfrac{\partial W}{\partial tr\mathbf{M}_1\mathbf{E}}\right) \\ \dfrac{\partial}{\partial tr\mathbf{M}_2\mathbf{E}}\left(\dfrac{\partial W}{\partial tr\mathbf{M}_3\mathbf{E}}\right) = \dfrac{\partial}{\partial tr\mathbf{M}_3\mathbf{E}}\left(\dfrac{\partial W}{\partial tr\mathbf{M}_2\mathbf{E}}\right) \end{cases} \tag{3.10}$$

Carrying out the derivatives (3.10) gives the following relations

$$c_1 = b_2 \qquad b_3 = d_1 \qquad d_2 = c_3 \tag{3.11}$$

From (3.11) it follows that the linear hyper elastic behavior of an orthotropic material depends on nine independent material constants. Finally, (3.8) with (3.9), reduced with (3.11) giving the invariant formulation of an orthotropic linear hyper elastic material

$$\begin{aligned} \mathbf{T} = &\left(b_1 tr\mathbf{M}_1\mathbf{E} + b_2 tr\mathbf{M}_2\mathbf{E} + d_1 tr\mathbf{M}_3\mathbf{E}\right)\mathbf{M}_1 + \\ &+ \left(b_2 tr\mathbf{M}_1\mathbf{E} + c_2 tr\mathbf{M}_2\mathbf{E} + c_3 tr\mathbf{M}_3\mathbf{E}\right)\mathbf{M}_2 + \\ &+ \left(d_1 tr\mathbf{M}_1\mathbf{E} + c_3 tr\mathbf{M}_2\mathbf{E} + d_3 tr\mathbf{M}_3\mathbf{E}\right)\mathbf{M}_3 + \\ &+ a_4\left(\mathbf{M}_1\mathbf{E} + \mathbf{E}\mathbf{M}_1\right) + a_5\left(\mathbf{M}_2\mathbf{E} + \mathbf{E}\mathbf{M}_2\right) + a_6\left(\mathbf{M}_3\mathbf{E} + \mathbf{E}\mathbf{M}_3\right) \end{aligned} \tag{3.12}$$

Integration of the three first partial derivates in (3.9) with (3.11) gives the expression for the strain energy

$$\begin{aligned} W = &\frac{1}{2} b_1 tr^2 \mathbf{M}_1\mathbf{E} + \frac{1}{2} c_2 tr^2 \mathbf{M}_2\mathbf{E} + \frac{1}{2} d_3 tr^2 \mathbf{M}_3\mathbf{E} + c_3 tr\mathbf{M}_2\mathbf{E} tr\mathbf{M}_3\mathbf{E} + \\ &+ d_1 tr\mathbf{M}_3\mathbf{E} tr\mathbf{M}_1\mathbf{E} + b_2 tr\mathbf{M}_1\mathbf{E} tr\mathbf{M}_2\mathbf{E} + a_4 tr\mathbf{M}_1\mathbf{E}^2 + a_5 tr\mathbf{M}_2\mathbf{E}^2 + a_6 tr\mathbf{M}_3\mathbf{E}^2 \end{aligned} \tag{3.13}$$

The classical formulation of an orthotropic linear hyper elastic material in its orthonormal privileged frame ($\mathbf{v}_1$, $\mathbf{v}_2$, $\mathbf{v}_3$) is given by

$$\mathbf{T} = \mathbf{DE} \quad ; \quad \mathbf{D} = \begin{bmatrix} a & f & e & 0 & 0 & 0 \\ f & b & d & 0 & 0 & 0 \\ e & d & c & 0 & 0 & 0 \\ 0 & 0 & 0 & g & 0 & 0 \\ 0 & 0 & 0 & 0 & h & 0 \\ 0 & 0 & 0 & 0 & 0 & j \end{bmatrix} \tag{3.14}$$

where **D** is the fourth order elastic stiffness tensor and the parameters *a, b, c, d, e, f, g, h, j* are nine independent material constants. The symmetry property of the elastic stiffness tensor can be shown using the first law of thermodynamics. The transition from the invariant formulation (3.12) to the classical formulation (3.14) is given by

$$\begin{aligned} &b_1 = a + 2g - 2h - 2j \quad ; \quad c_2 = b + 2h - 2j - 2g \quad ; \\ &d_3 = c + 2j - 2g - 2h \quad ; \quad a_4 = h + j - g \quad ; \quad a_5 = j + g - h \quad ; \\ &a_6 = g + h - j \quad ; \quad c_3 = d \quad ; \quad d_1 = e \quad ; \quad b_2 = f \end{aligned} \tag{3.15}$$

This gives us an alternate invariant form of orthotropic linear hyper elasticity

$$\begin{aligned} \mathbf{T} = &[(a + 2g - 2h - 2j) tr\mathbf{M}_1\mathbf{E} + f\, tr\mathbf{M}_2\mathbf{E} + e\, tr\mathbf{M}_3\mathbf{E}]\mathbf{M}_1 + \\ &+ [f\, tr\mathbf{M}_1\mathbf{E} + (b + 2h - 2j - 2g) tr\mathbf{M}_2\mathbf{E} + d\, tr\mathbf{M}_3\mathbf{E}]\mathbf{M}_2 + \\ &+ [e\, tr\mathbf{M}_1\mathbf{E} + d\, tr\mathbf{M}_2\mathbf{E} + (c + 2j - 2g - 2h) tr\mathbf{M}_3\mathbf{E}]\mathbf{M}_3 + \\ &+ (h + j - g)(\mathbf{M}_1\mathbf{E} + \mathbf{E}\mathbf{M}_1) + (j + g - h)(\mathbf{M}_2\mathbf{E} + \mathbf{E}\mathbf{M}_2) + \\ &+ (g + h - j)(\mathbf{M}_3\mathbf{E} + \mathbf{E}\mathbf{M}_3) \end{aligned} \tag{3.16}$$

This formulation is very useful because with $\mathbf{M}_i$ it contains the privileged directions of the material and it allows us to redefine them at any time. For more detail see Boehler [2] and Ottosen et al [12]. The elastic stiffness tensor given in (3.14) can be derived from the stress tensor (3.12) as

$$\begin{aligned} D_{ijkl} = \frac{\partial T_{ij}}{\partial E_{kl}} = &\, b_1 M^1_{ij} M^1_{kl} + c_2 M^2_{ij} M^2_{kl} + d_3 M^3_{ij} M^3_{kl} + b_2 \left(M^1_{ij} M^2_{kl} + M^2_{ij} M^1_{kl} \right) + \\ &+ d_1 \left(M^1_{ij} M^3_{kl} + M^3_{ij} M^1_{kl} \right) + c_3 \left(M^2_{ij} M^3_{kl} + M^3_{ij} M^2_{kl} \right) + \\ &+ a_4 \left(M^1_{ik} \delta_{jl} + \delta_{ik} M^1_{jl} \right) + a_5 \left(M^2_{ik} \delta_{jl} + \delta_{ik} M^2_{jl} \right) + a_6 \left(M^3_{ik} \delta_{jl} + \delta_{ik} M^3_{jl} \right) \end{aligned} \tag{3.17}$$

where the numerical index for the structural tensors is placed in the upper right hand corner. Formulation (3.17) is given in index notation to clearly illustrate that it is a fourth order tensor. Also (3.17) is given in the global orthonormal frame $\left(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\right)$ where

$$\mathbf{x}_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \quad ; \quad \mathbf{x}_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \quad ; \quad \mathbf{x}_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{3.18}$$

As we assume hyper elasticity the elastic stiffness tensor have the following symmetry properties

$$D_{ijkl} = D_{jikl} \quad ; \quad D_{ijkl} = D_{ijlk} \quad ; \quad D_{ijkl} = D_{klij} \tag{3.19}$$

The last condition in (3.19) is only for hyper elasticity. More explicitly with (3.19), (3.17) is given by

$$\mathbf{D} = \begin{bmatrix} D_{1111} & D_{1122} & D_{1133} & D_{1112} & D_{1113} & D_{1123} \\ D_{2211} & D_{2222} & D_{2233} & D_{2212} & D_{2213} & D_{2223} \\ D_{3311} & D_{3322} & D_{3333} & D_{3312} & D_{3313} & D_{3323} \\ D_{1211} & D_{1222} & D_{1233} & D_{1212} & D_{1213} & D_{1223} \\ D_{1311} & D_{1322} & D_{1333} & D_{1312} & D_{1313} & D_{1323} \\ D_{2311} & D_{2322} & D_{2333} & D_{2312} & D_{2313} & D_{2323} \end{bmatrix} \tag{3.20}$$

Chapter 4

FE-formulation of 3D elasticity and numerical treatment

Many problems are too complex to be solved analytically and therefore the finite element method offers a numerical approximate solution for systems of differential equations.

4.1 FE-formulation of 3D elasticity

The objective is to find the finite element formulation by first starting with the strong formulation, finding the week formulation and finally getting the finite element formulation. In the present case the problem is to solve the differential equations of equilibrium. Expressed in the stress tensor **σ** and the body force vector **b**, these are given by

$$\tilde{\nabla}^T \boldsymbol{\sigma} + \mathbf{b} = \mathbf{0} \tag{4.1}$$

where

$$\tilde{\nabla}^T = \begin{bmatrix} \frac{\partial}{\partial x} & 0 & 0 & \frac{\partial}{\partial y} & \frac{\partial}{\partial z} & 0 \\ 0 & \frac{\partial}{\partial y} & 0 & \frac{\partial}{\partial x} & 0 & \frac{\partial}{\partial z} \\ 0 & 0 & \frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial x} & \frac{\partial}{\partial y} \end{bmatrix}; \boldsymbol{\sigma} = \begin{bmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{zz} \\ \sigma_{xy} \\ \sigma_{xz} \\ \sigma_{yz} \end{bmatrix}; \mathbf{b} = \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} \tag{4.2}$$

Carrying out the multiplications of (4.1) gives the three equations of equilibrium, one in each direction, i.e.

$$\begin{aligned}
&\frac{\partial\sigma_{xx}}{\partial x}+\frac{\partial\sigma_{xy}}{\partial y}+\frac{\partial\sigma_{xz}}{\partial z}+b_x=0\\
&\frac{\partial\sigma_{yx}}{\partial x}+\frac{\partial\sigma_{yy}}{\partial y}+\frac{\partial\sigma_{yz}}{\partial z}+b_y=0\\
&\frac{\partial\sigma_{zx}}{\partial x}+\frac{\partial\sigma_{zy}}{\partial y}+\frac{\partial\sigma_{zz}}{\partial z}+b_z=0
\end{aligned} \tag{4.3}$$

On the boundary of the body the traction vector **t** acts

$$\mathbf{t}=\begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \tag{4.4}$$

The traction vector must fulfill the boundary condition

$$\begin{aligned}
t_x &= \sigma_{xx}n_x+\sigma_{xy}n_y+\sigma_{xz}n_z\\
t_y &= \sigma_{yx}n_x+\sigma_{yy}n_y+\sigma_{yz}n_z\\
t_z &= \sigma_{zx}n_x+\sigma_{zy}n_y+\sigma_{zz}n_z
\end{aligned} \tag{4.5}$$

Observe that on the exterior surface of the body (4.5) expresses a relation between the forces on the external surface and the stress tensor. To find the week formulation of the problem it is needed to define an arbitrary vector **v,**

$$\mathbf{v}=\begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} \tag{4.6}$$

By using (4.2) we have

$$\tilde{\nabla}\mathbf{v}=\begin{bmatrix} \frac{\partial v_x}{\partial x} \\ \frac{\partial v_y}{\partial y} \\ \frac{\partial v_z}{\partial z} \\ \frac{\partial v_x}{\partial y}+\frac{\partial v_y}{\partial x} \\ \frac{\partial v_x}{\partial z}+\frac{\partial v_z}{\partial x} \\ \frac{\partial v_y}{\partial z}+\frac{\partial v_z}{\partial y} \end{bmatrix} \tag{4.7}$$

Transposing (4.5) and multiplying with $\boldsymbol{\sigma}$ gives

$$
\begin{aligned}
\left(\tilde{\nabla}\mathbf{v}\right)^T \boldsymbol{\sigma} = {} & \frac{\partial v_x}{\partial x}\sigma_{xx} + \frac{\partial v_y}{\partial y}\sigma_{yy} + \frac{\partial v_z}{\partial z}\sigma_{zz} + \left(\frac{\partial v_x}{\partial y} + \frac{\partial v_y}{\partial x}\right)\sigma_{xy} + \\
& + \left(\frac{\partial v_x}{\partial z} + \frac{\partial v_z}{\partial x}\right)\sigma_{xz} + \left(\frac{\partial v_y}{\partial z} + \frac{\partial v_z}{\partial y}\right)\sigma_{yz}
\end{aligned}
\tag{4.8}
$$

Now, multiply the first equation of (4.3), i.e. the equation in the x-direction by the arbitrary function v_x and integrate over the volume

$$
\int_V v_x \frac{\partial \sigma_{xx}}{\partial x} dV + \int_V v_x \frac{\partial \sigma_{xy}}{\partial y} dV + \int_V v_x \frac{\partial \sigma_{xz}}{\partial z} dV + \int_V v_x b_x dV = 0
$$

Integration by parts using the Green-Gauss theorem yields

$$
\begin{aligned}
& \int_S v_x \sigma_{xx} n_x dS - \int_V \frac{\partial v_x}{\partial x}\sigma_{xx} dV + \int_S v_x \sigma_{xy} n_y dS - \int_V \frac{\partial v_x}{\partial y}\sigma_{xy} dV + \\
& \quad + \int_S v_x \sigma_{xz} n_z dS - \int_V \frac{\partial v_x}{\partial z}\sigma_{xz} dV + \int_V v_x b_x dV = 0
\end{aligned}
$$

With (4.5) this expression reduces to

$$
\int_S v_x t_x dS - \int_V \left(\frac{\partial v_x}{\partial x}\sigma_{xx} + \frac{\partial v_x}{\partial y}\sigma_{xy} + \frac{\partial v_x}{\partial z}\sigma_{xz}\right) dV + \int_V v_x b_x dV = 0 \tag{4.9}
$$

Similarly if we multiply the second and third equation of (4.3) by the arbitrary functions v_y and v_z respectively we get

$$
\int_S v_y t_y dS - \int_V \left(\frac{\partial v_y}{\partial x}\sigma_{yx} + \frac{\partial v_y}{\partial y}\sigma_{yy} + \frac{\partial v_y}{\partial z}\sigma_{yz}\right) dV + \int_V v_y b_y dV = 0 \tag{4.10}
$$

$$
\int_S v_z t_z dS - \int_V \left(\frac{\partial v_z}{\partial x}\sigma_{zx} + \frac{\partial v_z}{\partial y}\sigma_{zy} + \frac{\partial v_z}{\partial z}\sigma_{zz}\right) dV + \int_V v_z b_z dV = 0 \tag{4.11}
$$

Addition of (4.9)-(4.11) yields

$$
\begin{aligned}
& \int_S \left(v_x t_x + v_y t_y + v_z t_z\right) dS + \int_V \left(v_x b_x + v_y b_y + v_z b_z\right) dV - \\
& - \int_V \left[\frac{\partial v_x}{\partial x}\sigma_{xx} + \frac{\partial v_y}{\partial y}\sigma_{yy} + \frac{\partial v_z}{\partial z}\sigma_{zz} + \left(\frac{\partial v_x}{\partial y} + \frac{\partial v_y}{\partial x}\right)\sigma_{xy} + \right. \\
& \qquad \left. + \left(\frac{\partial v_x}{\partial z} + \frac{\partial v_z}{\partial x}\right)\sigma_{xz} + \left(\frac{\partial v_y}{\partial z} + \frac{\partial v_z}{\partial x}\right)\sigma_{yz} \right] dV = 0
\end{aligned}
$$

Use of (4.8) gives us

$$\int_V \left(\tilde{\nabla}\mathbf{v}\right)^T \boldsymbol{\sigma}\, dV = \int_S \mathbf{v}^T \mathbf{t}\, dS + \int_V \mathbf{v}^T \mathbf{b}\, dV \tag{4.12}$$

This is the week form of the differential equations of equilibrium (4.1). At this point no restrictions have been made regarding the constitutive relation. Also note that the weight vector *v* is completely arbitrary. To reach the fe-formulation, the displacement vector **u** will be approximated by

$$\mathbf{u} = \mathbf{N}\mathbf{a} \tag{4.13}$$

where **N** are the form functions and **a** are the nodal displacements. Using the Galerkin method for the choice of weight function we have

$$\mathbf{v} = \mathbf{N}\mathbf{c} \tag{4.14}$$

Since **v** is arbitrary the **c**-matrix must be arbitrary. It now follows that

$$\tilde{\nabla}\mathbf{v} = \mathbf{B}\mathbf{c} \quad ; \quad \mathbf{B} = \tilde{\nabla}\mathbf{N} \tag{4.15}$$

Using (4.14) and (4.15) in the week form (4.12) and noting that the **c**-matrix is independent of the coordinates, we get

$$\mathbf{c}^T \left(\int_V \mathbf{B}^T \boldsymbol{\sigma} dV - \int_S \mathbf{N}^T \mathbf{t} dS - \int_V \mathbf{N}^T \mathbf{b} dV \right) = 0$$

Using that the **c**-matrix is arbitrary we conclude that

$$\int_V \mathbf{B}^T \boldsymbol{\sigma} dV = \int_S \mathbf{N}^T \mathbf{t} dS + \int_V \mathbf{N}^T \mathbf{b} dV \tag{4.16}$$

This formulation holds for any constitutive model since no relation between the stresses and the strains have been introduced. Using a linear elastic constitutive relation as

$$\boldsymbol{\sigma} = \mathbf{D}\boldsymbol{\varepsilon} \tag{4.17}$$

where the strains can be written as

$$\boldsymbol{\varepsilon} = \mathbf{B}\mathbf{a} \tag{4.18}$$

the fe-formulation becomes

$$\left(\int_V \mathbf{B}^T \mathbf{D}\mathbf{B} dV \right) \mathbf{a} = \int_{S_h} \mathbf{N}^T \mathbf{h}\, dS + \int_{S_g} \mathbf{N}^T \mathbf{t}\, dS \tag{4.19}$$

where the following conditions have been applied

$$\begin{cases} \mathbf{t} = \mathbf{h} & \text{on } S_h \\ \mathbf{u} = \mathbf{g} & \text{on } S_g \\ \mathbf{b} = \mathbf{0} \end{cases} \tag{4.20}$$

The first condition in (4.20) is the natural boundary condition where the traction is prescribed on the boundary S_h and the second condition is the essential boundary condition where the displacements are prescribed on the boundary S_g. Together S_h and S_g comprises the total boundary of the body. The third condition is an approximation that the body forces are small in comparison to the other forces. In a more compact manner (4.19) becomes

$$\mathbf{Ka} = \mathbf{f} \tag{4.21}$$

where

$$\mathbf{K} = \int_V \mathbf{B}^T \mathbf{DB}\, dV \quad ; \quad \mathbf{f} = \int_{S_h} \mathbf{N}^T \mathbf{h}\, dS + \int_{S_g} \mathbf{N}^T \mathbf{t}\, dS \tag{4.22}$$

For more detail see Ottosen et al. [11].

4.2 Newton’s method

The finite element method for nonlinear problems leads to the solution of nonlinear equations. It will later be clear that the problem at hand is in fact nonlinear. The choice of solution method here is based on convergence speed. It can be shown that Newton’s method is locally quadratic convergent. For one-dimensional problems Figure 4.1 illustrates Newton’s method. The problem is to find the solution to $f(x) = 0$. By guessing a solution point x_1 the tangent to the curve at the corresponding point can be extrapolated to obtain the next estimate x_2. This procedure can then be repeated so the next estimate x_3 is obtained. When the error between the real solution and the estimate is small enough convergence is achieved.

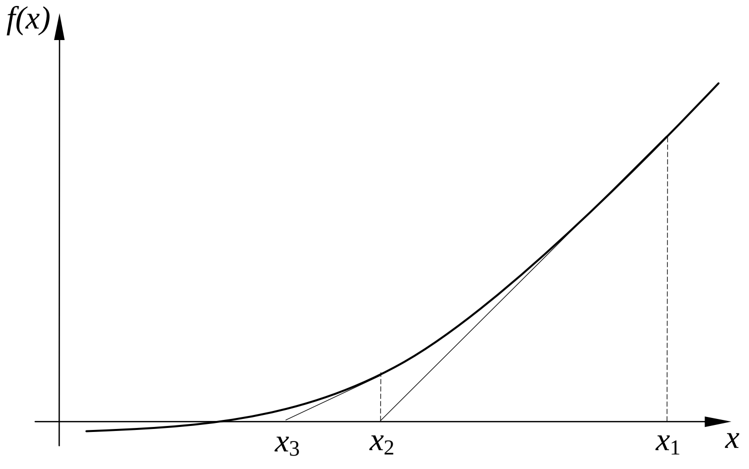

Figure 4.1 Newton’s method for one-dimensional problems.

The idea behind Newton’s method is to linearize the function about a point. This can be achieved by a Taylor expansion about the point and ignoring higher order terms. In the present case the nonlinear function is given by

$$\mathbf{F}(\mathbf{a}) = \mathbf{0} \tag{4.21}$$

where

$$\mathbf{F}(\mathbf{a}) = \int_V \mathbf{B}^T \boldsymbol{\sigma}\, dV - \mathbf{f} \tag{4.22}$$

The external forces **f** are chosen by us and the stresses **σ** depend on the displacements **a**. Now assume that the solution $\mathbf{a}^{\mathrm{i}}$ are known. A Taylor expansion of $\mathbf{F}(\mathbf{a})$ about $\mathbf{a}^{\mathrm{i}}$ gives

$$\mathbf{F}(\mathbf{a}^{i+1}) = \mathbf{F}(\mathbf{a}^{i}) + \left(\frac{\partial \mathbf{F}}{\partial \mathbf{a}}\right)^{i} (\mathbf{a}^{i+1} - \mathbf{a}^{i}) \tag{4.23}$$

where higher order terms have been ignored. This is the linearized approximation to $\mathbf{F}(\mathbf{a}^{i+1})$ and so it is the tangent to the solution at point $\mathbf{a}^{\mathrm{i}}$. We require that $\mathbf{F}(\mathbf{a}^{i+1}) = \mathbf{0}$ and (4.23) becomes

$$\mathbf{0} = \mathbf{F}(\mathbf{a}^{i}) + \left(\frac{\partial \mathbf{F}}{\partial \mathbf{a}}\right)^{i} (\mathbf{a}^{i+1} - \mathbf{a}^{i}) \tag{4.24}$$

Since the external load **f** is fix (4.22) gives

$$\frac{\partial \mathbf{F}}{\partial \mathbf{a}} = \int_V \mathbf{B}^T \frac{d\boldsymbol{\sigma}}{d\mathbf{a}} dV \tag{4.25}$$

The incremental constitutive relation $\dot{\boldsymbol{\sigma}} = \mathbf{D}^{ATS} \dot{\boldsymbol{\varepsilon}}$ gives us

$$\frac{d\boldsymbol{\sigma}}{d\mathbf{a}} = \mathbf{D}^{ATS} \mathbf{B} \tag{4.26}$$

This incremental constitutive relation, suitable for numerical treatment is derived in Chapter 6. Insertion of (4.26) into (4.25) yields

$$\frac{\partial \mathbf{F}}{\partial \mathbf{a}} = \mathbf{K}^{ATS} \quad ; \quad \mathbf{K}^{ATS} = \int_V \mathbf{B}^T \mathbf{D}^{ATS} \mathbf{B}\, dV \tag{4.28}$$

where $\mathbf{K}^{ATS}$ is the algorithmic tangential stiffness matrix of the body. This gives us

$$\left(\mathbf{K}^{ATS}\right)^{i} (\mathbf{a}^{i+1} + \mathbf{a}^{i}) = -\mathbf{F}(\mathbf{a}^{i}) \tag{4.29}$$

which is the sought Newton method for solution of nonlinear multi-dimensional equations. For more detail see Ottosen et al. [13].

Chapter 5

The evolution law for the material directions

5.1 The evolution law

The aim of this chapter is to establish a satisfactory evolution law for the change of the privileged directions of the material, i.e. the directions of orthotropy towards the principal strain directions. This is done since according to Cowin [4], biological tissue can only sense strain and not stress. The first task is to establish which quantities the rate of change depends on. Perhaps the most obvious thing is the difference between the privileged directions of the material and the principal strain directions, i.e. the eigenvectors of the strain tensor. When the difference is large enough, the adaptation process may begin. The second stimulus is the size of the strain. When the strain reaches a threshold it triggers the adaptation process. Several quantities exist when the "size" of the strain is to be evaluated and we will return to this later. This leads to the simple evolution law

$$\dot{\mathbf{v}}_i = \dot{h}_i \mathbf{n}_i \quad ; \quad i = 1, 2 \tag{5.1}$$

where $\dot{\mathbf{v}}_i$ is the rate of change of the privileged direction *i*, $\dot{h}_i$ is the rate of change of a scalar weight function h_i and $\mathbf{n}_i$ is the *i*:th principal strain direction. Integrating (5.1) between time t^k and t^{k+1} gives

$$\mathbf{v}_i^{k+1} - \mathbf{v}_i^k = \dot{h}_i^k \Delta t \mathbf{n} \quad ; \quad \Delta t = t^{k+1} - t^k \tag{5.2}$$

This evolution law is later to be formulated as a numerical law but first we need to evaluate the scalar weight function *h*.

5.2 The weight function *h*

The scalar function *h* determines the rate of change in every step *k*. It is a function of firstly, the degree of orthogonality between the principal directions of strain and the privileged directions of the material, secondly, the trace of the principal strain tensor squared and last the squared trace of the principal strain tensor. The degree of orthogonality is calculated as a simple scalar product. The trace of the principal strain tensor squared and the squared trace of the principal strain tensor are used because they are positive invariants. The general formulation of the time derivative of the weight function for the first principal direction looks like

$$\dot{h}_1 = \dot{h}_1\left(\mathbf{v}_i^T\mathbf{n}_i, tr\tilde{\mathbf{E}}^2, tr^2\tilde{\mathbf{E}}\right) \tag{5.3}$$

where $\tilde{\mathbf{E}}$ is the strain tensor in the principal frame. The aim is to derive a function that produces numerically acceptable values. For simplicity, the time derivative of the weight function was chosen as

$$\dot{h}_1 = \mathbf{v}_1^T\mathbf{n}_1 \frac{tr^2\tilde{\mathbf{E}}}{tr\tilde{\mathbf{E}}^2} \tag{5.4}$$

Through the introduction of the division on the right hand side in equation (5.4) it is only valid for non zero loads. This is acceptable however since the goal is to simulate the behavior of a loaded bone sample. Integration of equation (5.4) gives

$$h_1 = \Delta t\left(\mathbf{v}_1^T\mathbf{n}_1\right)\frac{tr^2\tilde{\mathbf{E}}}{tr\tilde{\mathbf{E}}^2} \tag{5.5}$$

This formulation leaves determining the sign of h_1 up to the scalar product $\mathbf{v}_1^T\mathbf{n}_1$. Comparing (5.4) and (5.5) with (5.2) we see that

$$h_i = \dot{h}_i\Delta t \tag{5.6}$$

To ensure that the two directions first calculated in every step remain orthogonal the second weight function h_2 is subject to the following restriction

$$h_2 = -h_1\frac{\mathbf{v}_2^T\mathbf{n}_1}{\mathbf{v}_1^T\mathbf{n}_2} \tag{5.7}$$

5.3 The numerical evolution law

The numerical evolution law is given by (5.2) with a supplement.

$$\begin{cases} \mathbf{v}_i^{k+1} = \mathbf{v}_i^k + h_i^k \mathbf{n}_i \\ \left|\mathbf{v}_i^{k+1}\right| = 1 \end{cases} \quad ; \quad i = 1,2 \tag{5.8}$$

The second equation of (5.8) is a simple normalization of the new direction calculated. This is done because we are only interested in the directions of orthotropy. Figure 5.1 clearly illustrates equation (5.8).

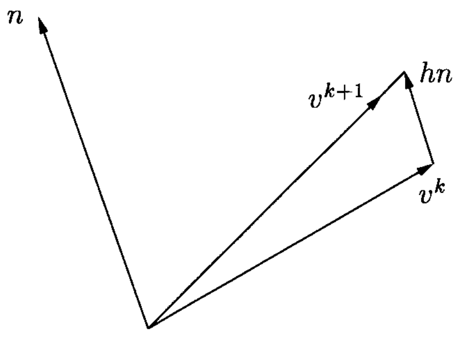

Figure 5.1 The procedure of finding the new direction in each step.

Since $\mathbf{v}_1^T \mathbf{n}_1$ can be either positive or negative we conclude that

$$\left(\mathbf{v}_1^T \mathbf{n}_1\right)^k = \pm 1 \quad ; \quad k \rightarrow \infty \tag{5.9}$$

One consequence of this choice is that we assume that bone reacts similar to tensile and compressive strain meaning that we never rotate one privileged axis more than 90 degrees, thus allowing the privileged direction to point in the opposite direction of its corresponding principal direction. The third privileged direction is calculated by a vector product of the two first to ensure that the new privileged coordinate system is orthonormal right-handed. With this scheme we can follow the change of the privileged directions of the material.

Chapter 6

The evolution law for the material properties

6.1 The evolution law

In this part an updating algorithm is defined for the material properties. The Young's moduli and the shear moduli were chosen for the updating purpose while the Poisson's ratios were kept constant. What happens when bone is loaded is that the thickness of the trabeculae increases where it is needed and decreased in other areas. The choice of varying the Young's moduli and the shear moduli was made because changes in thickness for the trabeculae are intuitively coupled to changes in the elastic properties. The numerical values were taken from Ashman et al. [1]. The first task is to introduce some kind of evolution law for the material properties. From Miller et al. [9] this is defined as a trilinear relation

$$\dot{E} = \begin{cases} c_2\left(\psi - \psi_0\left(1+s\right)\right), & \psi > \psi_0\left(1+s\right) \\ 0, & \psi_0\left(1-s\right) < \psi < \psi_0\left(1+s\right) \\ c_2\left(\psi - \psi_0\left(1-s\right)\right), & \psi < \psi_0\left(1-s\right) \end{cases} \tag{6.1}$$

where $\dot{E}$ is the rate of change of any one of the six elastic properties, c_2 is a positive scalar constant, ψ is the stimulus for its corresponding elastic property, ψ_0 is a reference stimulus and *s* is half the width of the "lazy" zone. The "lazy" zone is the part where no change in the elastic property occurs. A graphical interpretation of equation (6.1) may be seen in Figure 6.1.

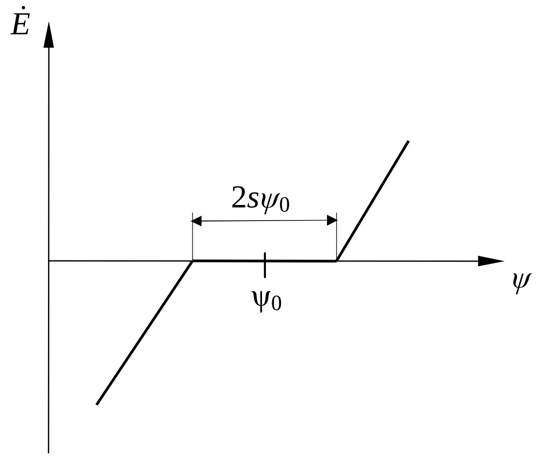

Figure 6.1 The graphical interpretation of the evolution law, equation (6.1).

Mathematically the change from the “lazy” zone to any of the two other zones is a discontinuity and it may later cause problems. With this in mind *s* is assumed to be zero giving the evolution law

$$\dot{E} = c_2(\psi - \psi_0) \tag{6.2}$$

The next task is to define the stimulus. In the former part of this thesis the changes in the material directions were based on the strain tensor. It is appropriate to use a strain based stimulus also for the evolution of the material properties in accordance with the procedure for the updating of the directions. The definition of this stimulus incorporates the strain tensor, the elastic modulus and the material direction as

$$\psi = c_1 \frac{1}{E}\left|\varepsilon_{ij} M_{ij}\right| = c_1 \frac{1}{E}\left|tr\, \boldsymbol{\varepsilon}\mathbf{M}\right| \tag{6.3}$$

where c_1 is a positive scalar constant, *E* is any one of the six elastic moduli, $\boldsymbol{\varepsilon}$ is the strain tensor and **M** is the structural tensor corresponding to the elastic modulus. The absolute value of the trace is used because of the ability to react similarly to compressive and tensile strains which has been mentioned earlier. Since there are six elastic properties to update there should be six stimuli, one for each property. It is therefore necessary to define three structural tensors in addition to the three defined earlier.

$$\begin{cases} \mathbf{M}_{12} = \mathbf{v}_1 \otimes \mathbf{v}_2 + \mathbf{v}_2 \otimes \mathbf{v}_1 \\ \mathbf{M}_{13} = \mathbf{v}_1 \otimes \mathbf{v}_3 + \mathbf{v}_3 \otimes \mathbf{v}_1 \\ \mathbf{M}_{23} = \mathbf{v}_2 \otimes \mathbf{v}_3 + \mathbf{v}_3 \otimes \mathbf{v}_2 \end{cases} \tag{6.4}$$

These structural tensors are used in the process of updating the three shear moduli. With these things clear we proceed in order to find an expression for the elastic moduli in the next step based on the last equilibrium state. Equation (6.2) is integrated between the time *t* and Δt giving

$$\int_{t}^{t+\Delta t} \dot{E}\, dt = \int_{E^n}^{E^{n+1}} dE = \int_{t}^{t+\Delta t} c_2 \left(\psi - \psi_0\right) dt \tag{6.5}$$

(6.3) inserted into (6.5) and the use of an implicit Euler method, i.e. evaluating the integral on the right hand side in (6.5) in the next equilibrium state *n+1* gives an ordinary equation of the second order

$$\left(E^{n+1}\right)^2 + E^{n+1}\left(c_2 \Delta t \psi_0 - E^n\right) - c_1 c_2 \Delta t \left|\varepsilon_{ij} M_{ij}\right|^{n+1} = 0 \tag{6.6}$$

where E^{n+1} is the sought quantity and index n represents the last equilibrium state. Introducing the two quantities A and B as

$$\begin{aligned} A &= c_2 \Delta t \psi_0 - E^n \\ B &= c_1 c_2 \Delta t \left|\varepsilon_{ij} M_{ij}\right|^{n+1} \end{aligned} \tag{6.7}$$

the solution for (6.6) is

$$E^{n+1} = -\frac{A}{2} \pm \sqrt{\frac{A^2}{4} + B} \tag{6.8}$$

In (6.8) the solution that comes from the minus sign is not considered since it gives a non realistic solution to the problem.

6.2 The algorithmic tangential stiffness

Now that the elastic moduli in the next step are obtained the next objective is to find the algorithmic tangential stiffness matrix. Consider first the constitutive equation given by (3.14) written in index notation

$$\sigma_{ij} = D_{ijkl} \varepsilon_{kl} \tag{6.9}$$

where the constitutive matrix is given by (3.17). The time derivative of (6.9) is given by

$$\dot{\sigma}_{ij} = \dot{D}_{ijkl} \varepsilon_{kl} + D_{ijkl} \dot{\varepsilon}_{kl} \tag{6.10}$$

In order to get an expression for $\dot{\sigma}_{ij}$ suitable for numerical integration, the relation is written using an algorithmic tangential stiffness matrix

$$\dot{\sigma}_{ij} = D_{ijkl}^{ATS} \dot{\varepsilon}_{kl} \tag{6.11}$$

To transform (6.10) to (6.11) the term $\dot{D}_{ijkl}$ has to be investigated. The constitutive equation is a function of the six elastic moduli and the three structural tensors. In this part the structural tensors are not considered as time dependent so the time derivative of the constitutive matrix is given by

$$\dot{D}_{ijkl} = \frac{\partial D_{ijkl}}{\partial t} = \\ \frac{\partial D_{ijkl}}{\partial E_1}\frac{\partial E_1}{\partial t} + \frac{\partial D_{ijkl}}{\partial E_2}\frac{\partial E_2}{\partial t} + \frac{\partial D_{ijkl}}{\partial E_3}\frac{\partial E_3}{\partial t} + \frac{\partial D_{ijkl}}{\partial G_{12}}\frac{\partial G_{12}}{\partial t} + \frac{\partial D_{ijkl}}{\partial G_{13}}\frac{\partial G_{13}}{\partial t} + \frac{\partial D_{ijkl}}{\partial G_{23}}\frac{\partial G_{23}}{\partial t} \tag{6.12}$$

Here both $\frac{\partial D_{ijkl}}{\partial E}$ and $\frac{\partial E}{\partial t}$ needs further development. The first of the two terms is obtained by taking the derivative of (3.17) with the elastic modulus

$$\frac{\partial D_{ijkl}}{\partial E} = \frac{\partial b_1}{\partial E} M^1_{ij} M^1_{kl} + \frac{\partial c_2}{\partial E} M^2_{ij} M^2_{kl} + \frac{\partial d_3}{\partial E} M^3_{ij} M^3_{kl} + \frac{\partial b_2}{\partial E}\left(M^1_{ij} M^2_{kl} + M^2_{ij} M^1_{kl}\right) + \\ + \frac{\partial d_1}{\partial E}\left(M^1_{ij} M^3_{kl} + M^3_{ij} M^1_{kl}\right) + \frac{\partial c_3}{\partial E}\left(M^2_{ij} M^3_{kl} + M^3_{ij} M^2_{kl}\right) + \\ + \frac{\partial a_4}{\partial E}\left(M^1_{ik}\delta_{jl} + \delta_{ik} M^1_{jl}\right) + \frac{\partial a_5}{\partial E}\left(M^2_{ik}\delta_{jl} + \delta_{ik} M^2_{jl}\right) + \frac{\partial a_6}{\partial E}\left(M^3_{ik}\delta_{jl} + \delta_{ik} M^3_{jl}\right) \tag{6.13}$$

The elastic constants used by Boehler [2], $b_1, c_2, d_3, b_2, d_1, c_3, a_4, a_5$ and a_6 are functions of the constants found in the constitutive matrix written in its privileged orthonormal frame a, b, c, d, e, f, g, h and j according to (3.15). These are functions of the elastic moduli, the shear moduli and the Poisson's ratios as follows

$$\begin{cases} a = \dfrac{E_1 E_2 \left(\nu_{32}^2 E_2 - E_3\right)}{\nu_{21}^2 E_1 E_3 + 2\nu_{21}\nu_{31}\nu_{32} E_1 E_2 + \nu_{32}^2 E_2^2 - E_2 E_3 + \nu_{31}^2 E_1 E_2} \\[2ex] b = \dfrac{E_2^2 \left(\nu_{31}^2 E_1 - E_3\right)}{\nu_{21}^2 E_1 E_3 + 2\nu_{21}\nu_{31}\nu_{32} E_1 E_2 + \nu_{32}^2 E_2^2 - E_2 E_3 + \nu_{31}^2 E_1 E_2} \\[2ex] c = \dfrac{E_3^2 \left(\nu_{21}^2 E_1 - E_2\right)}{\nu_{21}^2 E_1 E_3 + 2\nu_{21}\nu_{31}\nu_{32} E_1 E_2 + \nu_{32}^2 E_2^2 - E_2 E_3 + \nu_{31}^2 E_1 E_2} \\[2ex] d = \dfrac{E_2 E_3 \left(\nu_{31}\nu_{21} E_1 + \nu_{32} E_2\right)}{\nu_{21}^2 E_1 E_3 + 2\nu_{21}\nu_{31}\nu_{32} E_1 E_2 + \nu_{32}^2 E_2^2 - E_2 E_3 + \nu_{31}^2 E_1 E_2} \\[2ex] e = \dfrac{E_1 E_2 E_3 \left(\nu_{31} + \nu_{21}\nu_{32}\right)}{\nu_{21}^2 E_1 E_3 + 2\nu_{21}\nu_{31}\nu_{32} E_1 E_2 + \nu_{32}^2 E_2^2 - E_2 E_3 + \nu_{31}^2 E_1 E_2} \\[2ex] f = \dfrac{E_1 E_2 \left(\nu_{31}\nu_{32} E_2 + \nu_{21} E_3\right)}{\nu_{21}^2 E_1 E_3 + 2\nu_{21}\nu_{31}\nu_{32} E_1 E_2 + \nu_{32}^2 E_2^2 - E_2 E_3 + \nu_{31}^2 E_1 E_2} \end{cases} \tag{6.14}$$

and

$$\begin{cases} g = G_{12} \\ h = G_{13} \\ j = G_{23} \end{cases}$$

The second of the two terms in (6.12), i.e. $\frac{\partial E}{\partial t}$ is obtained by taking the time derivative of the newly calculated expression for E^{n+1}, see equation (6.8)

$$\frac{\partial E}{\partial t} = \frac{\partial}{\partial t}\left(\sqrt{\frac{A^2}{4}+B}\right) = \frac{c_1 c_2 \Delta t}{2\sqrt{\frac{A^2}{4}+B}}\left(\dot{\varepsilon}_{ij} M_{ij}\right)^{n+1} sign\left(\varepsilon_{ij} M_{ij}\right)^{n+1} \tag{6.15}$$

Here the fact that the structural tensors are not regarded as time dependent was used once again. Introducing the function F as

$$F = \sqrt{\frac{A^2}{4}+B}$$

equation (6.15) is reduced to

$$\frac{\partial E}{\partial t} = \frac{c_1 c_2 \Delta t}{2F}\left(\dot{\varepsilon}_{ij} M_{ij}\right)^{n+1} sign\left(\varepsilon_{ij} M_{ij}\right) \tag{6.16}$$

Equation (6.10) in combination with equations (6.12) and (6.16) gives the algorithmic tangential stiffness

$$\begin{aligned} D_{ijkl}^{ATS} = \frac{c_1 c_2 \Delta t}{2}\Bigg(& \frac{\partial D_{ijkl}}{\partial E_1}\frac{M_{op}^{1}}{F_1} sign\left(\dot{\varepsilon}_{rs} M_{rs}^{1}\right)^{n+1} + \frac{\partial D_{ijkl}}{\partial E_2}\frac{M_{op}^{2}}{F_2} sign\left(\dot{\varepsilon}_{rs} M_{rs}^{2}\right)^{n+1} + \\ & + \frac{\partial D_{ijkl}}{\partial E_3}\frac{M_{op}^{3}}{F_3} sign\left(\dot{\varepsilon}_{rs} M_{rs}^{3}\right)^{n+1} + \frac{\partial D_{ijkl}}{\partial G_{12}}\frac{M_{op}^{12}}{F_{12}} sign\left(\dot{\varepsilon}_{rs} M_{rs}^{12}\right)^{n+1} + \\ & + \frac{\partial D_{ijkl}}{\partial G_{13}}\frac{M_{op}^{13}}{F_{13}} sign\left(\dot{\varepsilon}_{rs} M_{rs}^{13}\right)^{n+1} + \frac{\partial D_{ijkl}}{\partial G_{23}}\frac{M_{op}^{23}}{F_{23}} sign\left(\dot{\varepsilon}_{rs} M_{rs}^{23}\right)^{n+1}\Bigg)\varepsilon_{op} + D_{ijkl}^{n+1} \end{aligned} \tag{6.17}$$

The updating algorithm now only needs the reference stimulus ψ_0 which is calculated according to

$$\psi_0 = \frac{1}{V_{tot}}\sum_{i=1}^{nel} V_i \psi_i \tag{6.18}$$

where V_{tot}, V_i are the total volume of the geometry and the volume of element *i* respectively.

6.3 Thermodynamic restrictions on the elastic properties

These restrictions are based on the fact that the work done on an elastic material has to be strictly positive [6]. The strain energy is defined as

$$U = \frac{1}{2}\varepsilon_{ij}\sigma_{ij} = \frac{1}{2}\varepsilon_{ij}D_{ijkl}\varepsilon_{kl} \tag{6.19}$$

For this quantity to be positive for all values of $\varepsilon_{ij} \neq \mathbf{0}$ requires the constitutive matrix to be positive definite. The conditions for D_{ijkl} to be positive definite implies that all of the minor determinants formed from D_{ijkl}, which along their diagonal contain the diagonal elements of D_{ijkl}, have to be positive. Applying these conditions gives

$$\begin{cases} E_1, E_2, E_3, G_{12}, G_{13}, G_{23} > 0 \\ (1-\nu_{23}\nu_{32}), (1-\nu_{13}\nu_{31}), (1-\nu_{12}\nu_{21}) > 0 \\ 1-\nu_{12}\nu_{21}-\nu_{23}\nu_{32}-\nu_{13}\nu_{31}-2\nu_{21}\nu_{32}\nu_{13} > 0 \end{cases} \tag{6.20}$$

Using the condition of symmetry of the compliances, i.e. the symmetry of the compliance matrix

$$\frac{\nu_{ij}}{E_i} = \frac{\nu_{ji}}{E_j} \tag{6.21}$$

in which the summation convention is not employed, the following inequalities are found that must be satisfied by the elastic constants

$$\begin{aligned} |\nu_{21}| &< \left(\frac{E_2}{E_1}\right)^{1/2}; \quad |\nu_{12}| < \left(\frac{E_1}{E_2}\right)^{1/2} \\ |\nu_{32}| &< \left(\frac{E_3}{E_2}\right)^{1/2}; \quad |\nu_{23}| < \left(\frac{E_2}{E_3}\right)^{1/2} \\ |\nu_{13}| &< \left(\frac{E_1}{E_3}\right)^{1/2}; \quad |\nu_{31}| < \left(\frac{E_3}{E_1}\right)^{1/2} \end{aligned} \tag{6.22}$$

It is also found that

$$\nu_{21}\nu_{32}\nu_{13} < \frac{1-\nu_{21}^2\left(\frac{E_1}{E_2}\right)-\nu_{32}^2\left(\frac{E_2}{E_3}\right)-\nu_{13}^2\left(\frac{E_3}{E_1}\right)}{2} < \frac{1}{2} \tag{6.23}$$

These inequalities can be adopted as restrictions on the Poisson's ratios. However in this case the Poisson's ratios are assumed to be fixed and these inequalities work as restrictions on the elastic moduli.

Chapter 7

The computations

The whole adaptation process is built up in two parts. The first part describes the change of the material directions and the second part covers the adaptation of the elastic moduli. The calculations were performed in Matlab using the CALFEM [20] toolbox on two different geometries. In the first geometry a total of 60 8-node isoparametric brick elements were used and an integration rule of two was adopted giving eight integration points in every element. To be able to compare the results from the computations to real life experiments and for the sake of simplicity, the first geometry chosen was a small cylinder. The dimensions of the cylinder were taken from the chamber of Tägil et al. [16], Figure 7.1. In his experiments Tägil introduced these chambers into the tibia of Sprague-Dawley rats. Tissue was allowed to grow into the chamber for three weeks before the mechanical loading was started. Thereafter a cyclic load was applied which was estimated to produce a compressive hydrostatic stress of 2 MPa. After seven weeks of loading the chambers were harvested and all contained newly formed bone. It was assumed that the ingrowth holes do not interfere with the calculations. They are at the bottom of the chamber while the tissue subject to this investigation is further up in the chamber, thus not in contact with the ingrowth holes.

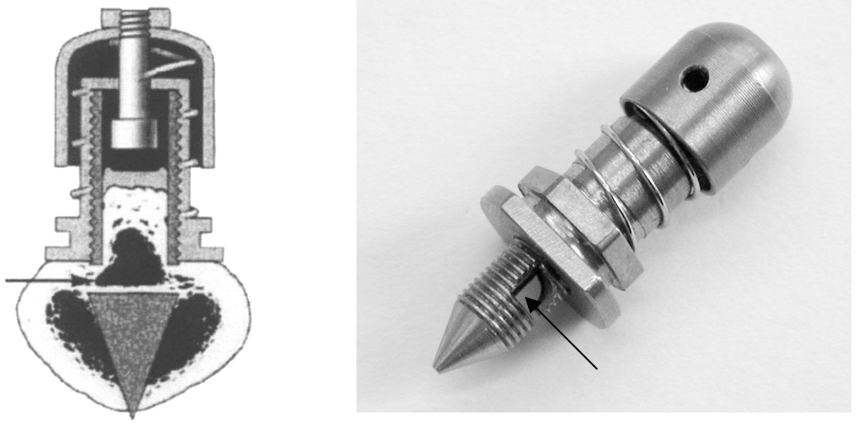

Figure 7.1 The chamber with inner diameter of 2 mm. The left figure shows a schematic cross section when the chamber is attached to the rat tibia. The right figure is a photography of the chamber. The ingrowth holes are marked with arrows.

According to symmetry, one fourth of the whole cylinder was modeled giving a quite simple geometry which can be seen in Figure 7.2.

The second geometry adopted was a symmetry plane of a long hollow cylinder seen in Figure 7.2 since the problem is axisymmetric. Here a total of 60 4-node isoparametric axisymmetric elements were used and an integration rule of one was adopted giving a single integration point in each element. The dimensions of the imaginary cylinder in Figure 7.2 were taken from Harrysson [7]. Irregular spacing between the elements in the radial direction was used because the variation of the properties with the radius was of interest.

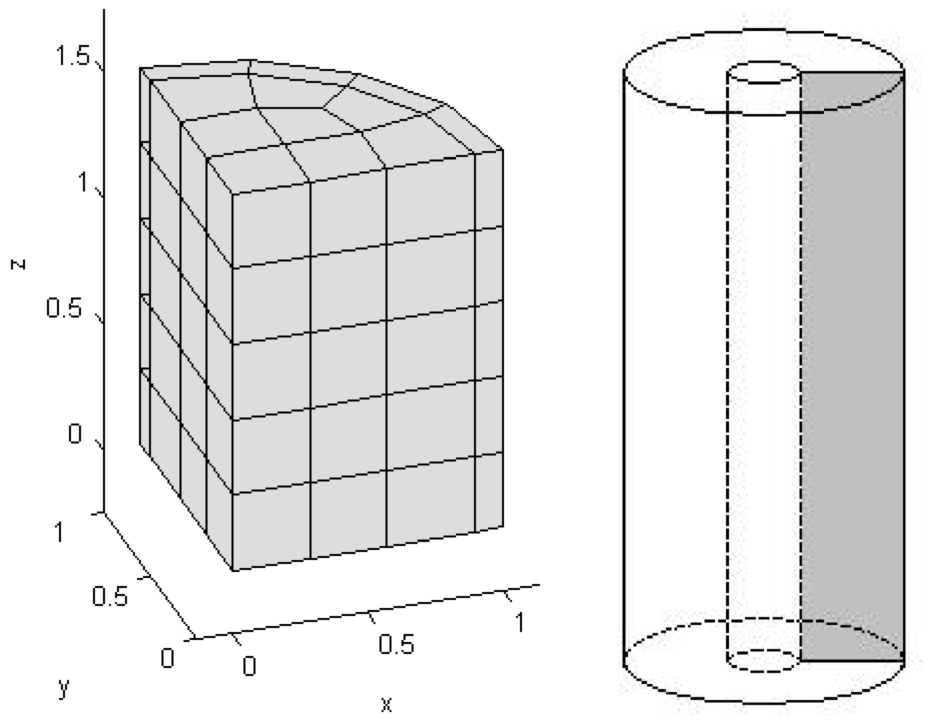

Figure 7.2 The two geometries used in the computations. The inner diameter, outer diameter and length respectively of the cylinder are 10, 34 and 120. All dimensions in mm.

For the 3D geometry, the following degrees of freedom (dofs) were locked, i.e. given a boundary condition of zero displacement: all dofs on the bottom, the dofs normal to the two symmetry planes and all but the vertical dof on the curved side. Initially each element was given a randomly oriented orthonormal frame. This was done to mimic the unordered macro structure of a never previously loaded sample of bone tissue. The evenly distributed normal load of 2 MPa was applied to the top of the geometry and kept constant throughout the calculations. This load gives rise to shear forces acting on the surface of the cylinder due to friction between the bone and the metal chamber. For simplicity it was assumed that the shear load applied to the cylinder side is a constant fraction of the normal load applied on the top. This assumption is rather crude considering that the material is assumed to be orthotropic, thus giving that the normal load on the side of the cylinder due to the top load is a function of the orientation of the privileged frame and the orientation of the element. Further, the friction coefficient between bone and metal varies with the orientation of the bone in contact with the metal giving variations in the shear forces [15]. It can be mentioned that the metal chamber in contact with the bone is assumed to be undeformable as a consequence of the boundary conditions applied. This is in reality not the case although the metal chamber is much stiffer than the bone it deforms nonetheless giving changes in the normal forces acting on the sides of the cylinder. These changes give variations in the friction force. However, to limit the extent of this thesis the assumption of a constant fraction is used.

For the axisymmetric geometry, the boundary condition of zero displacement was assigned to the four rightmost dofs (two horizontal and two vertical) on the bottom. The loads applied simulate the bone being loaded by a femoral head implant. A normal stress of 3.3 kPa was applied to the whole side facing the hole in the cylinder, c.f. Harrysson [7]. A shear force of 600 N was assumed to be evenly distributed on the inside of the cylinder representing the body weight of a person.

In this work bone is modeled as a continuum and the trabecular structure visible to the eye is ignored. One restriction when using the continuum approach is the size of the elements. According to Cowin et al. [5] the minimum length of an element should be at least several trabecular widths. The mean width of a trabecula was taken from measurements to be 0.1 mm. Comparing the first geometry to this characteristic length we see that we are well within the range for all elements except for those that are on the outer surface in contact with the chamber wall. The shortest side on these elements is 0.1 mm. All element sides in the second geometry are well within the limitations.

The goal of the first part of the computations is to se that the privileged frame of the material is coincident with the principal strain frame. However we may end up with a result that says that some or all privileged axes assumed the exact opposite direction as its corresponding principal strain axes. This is in total agreement of the assumption that bone reacts similarly to tensile and compressive strain. Once the privileged frame is reoriented the computations go to part two. The goal of the second part is to update the elastic constants in each element. The convergence of the first part is measured as the norm of the difference between the global stiffness matrices in two consecutive computational steps. In the second part the convergence in each step is measured as the norm of the difference between the external and internal forces, i.e. out of balance forces.

The 3D geometry is subjected to both the adaptation of the directions and the adaptation of the elastic properties. The axisymmetric geometry however is only subjected to the elastic properties adaptation procedure. In this case the directions in each element was prescribed so that the 1-direction lies in radial or horizontal direction, the 2-direction lies into the element plane and the 3-direction lies in the axial or vertical direction.

Chapter 8

Results and discussion

The results for both geometries are presented in this chapter. All the results would take up too much space so some interesting results have been picked out.

8.1 Results for the 3D geometry

This geometry was subjected to both the adaptation of the material privileged directions and the adaptation of the elastic moduli. The initial configuration for the privileged directions of orthotropy can bee seen in Figure 8.1a and the results for the updating of the directions can be seen in Figure 8.1b. Note that only the direction belonging to the largest stiffness is plotted in each element in both Figures in 8.1.

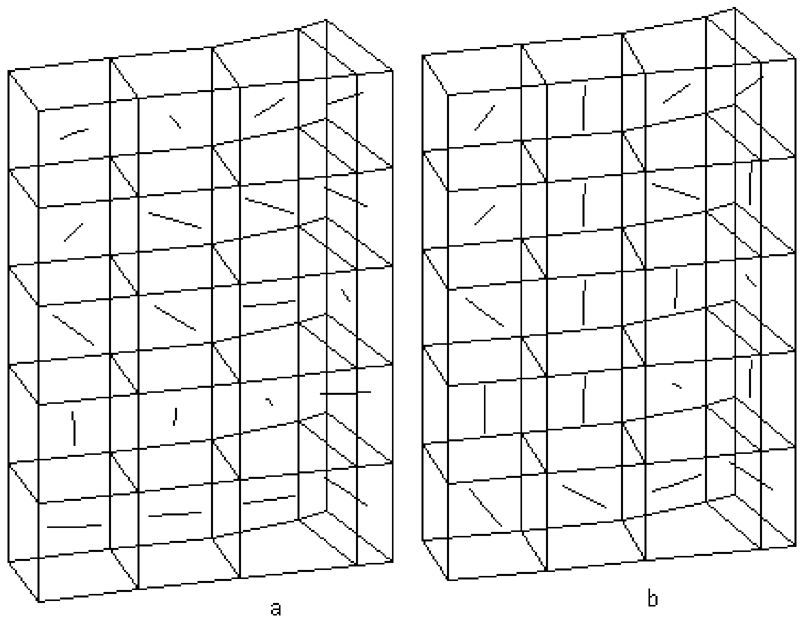

Figure 8.1 Part a shows the initial state and part b shows the final state of the directions of the largest stiffness in each element.

These directions in each element represent the preferred trabecular direction. It is difficult to present these results in a space efficient way so only one plane of elements is shown. This plane is the plane closest to the x-axis in Figure 7.2. One can clearly see that the directions of the largest stiffness are reoriented. Since the configuration of the initial privileged directions of orthotropy is randomly chosen at the beginning, Figure 8.1a varies between runs and it will influence the final orientation of the privileged directions and later the evolution of the elastic properties.

Figure 8.2 shows the element numbering system. The whole geometry contains 60 elements with one plane containing 12 elements. This local numbering system shown in Figure 8.2 displays the bottom plane. The adjacent plane above contains the elements 13 – 24 and so on. In Figure 8.3 the updating process for the elastic properties are shown for element 15. Figure 8.4 shows the updating process for the elastic moduli for element 49.

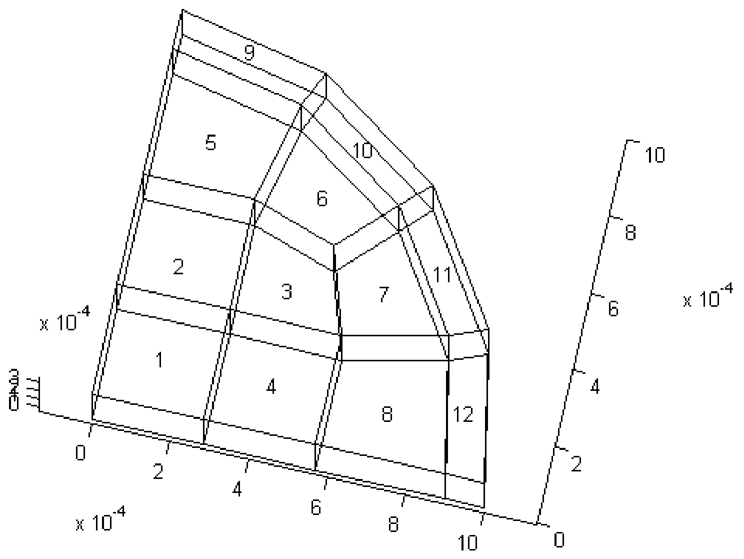

Figure 8.2 The element numbering system used in the geometry seen in Figure 7.2. Only the bottom plane is shown. The next plane above contains the next 12 elements.

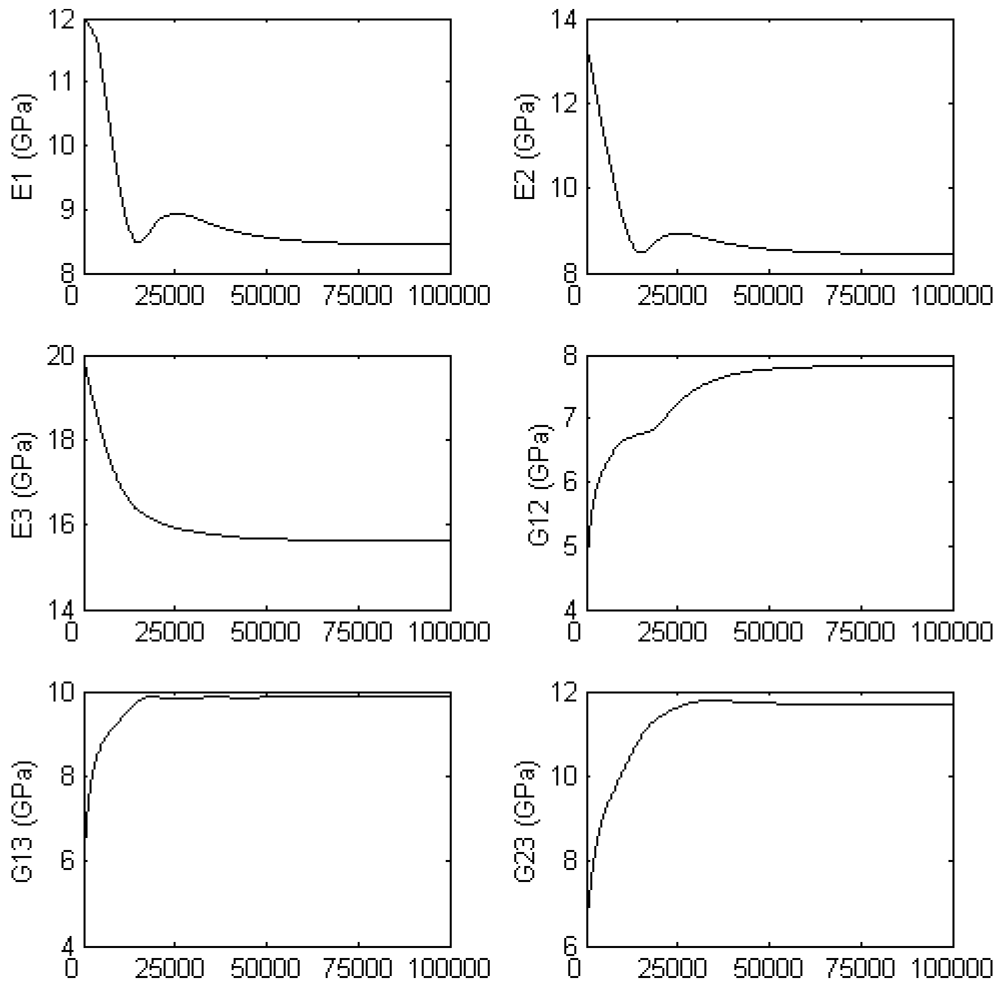

Figure 8.3 Results of the updating algorithm for the elastic moduli in element 15. The horizontal axis shows number of load steps.

Every element shows a different type of adaptation process. The local maximum for example seen for element 15, E_1 is not necessarily seen for other elements.

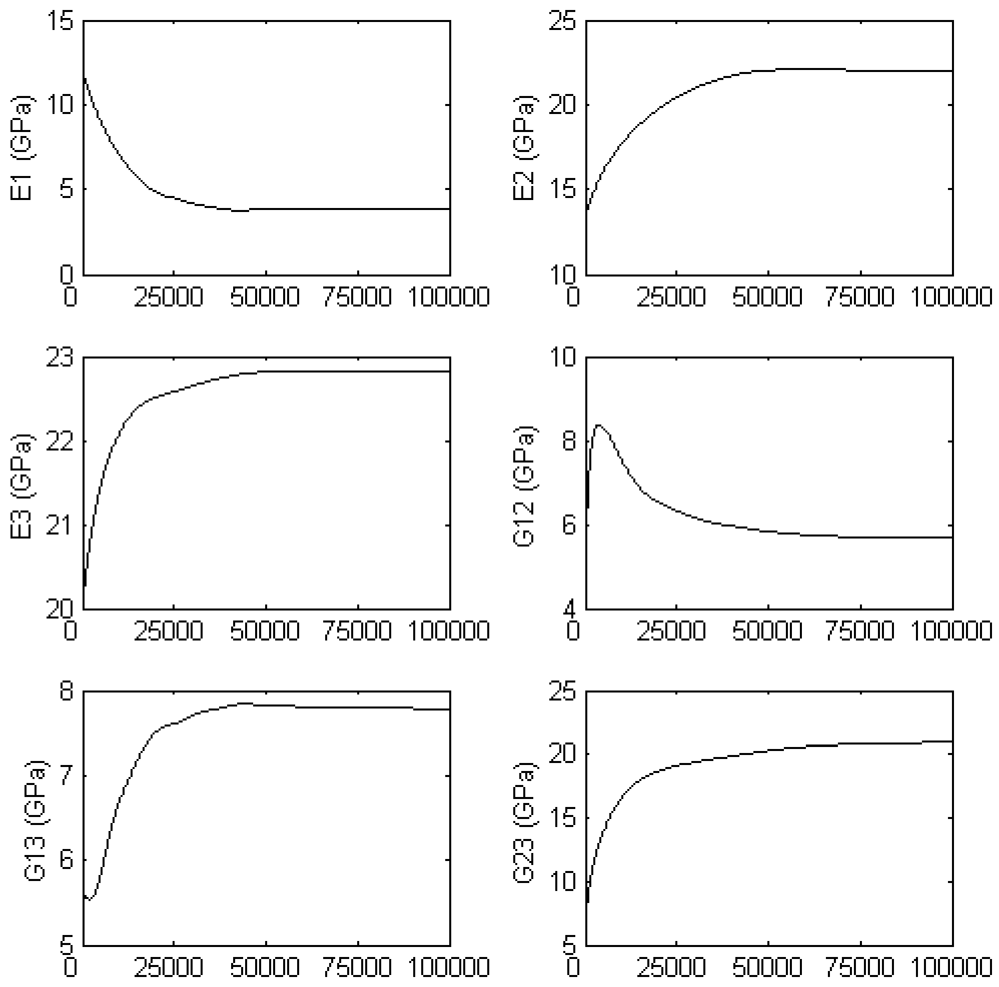

Figure 8.4 Results of the updating algorithm for the elastic moduli in element 49. The horizontal axis shows number of load steps.

Figure 8.5 shows the results for element 39. Note the two peaks for E_1 and E_2. These are due to the lack of a lazy zone (6.2). Inclusion of a lazy zone would probably eliminate these kinds of peaks to a large extent.

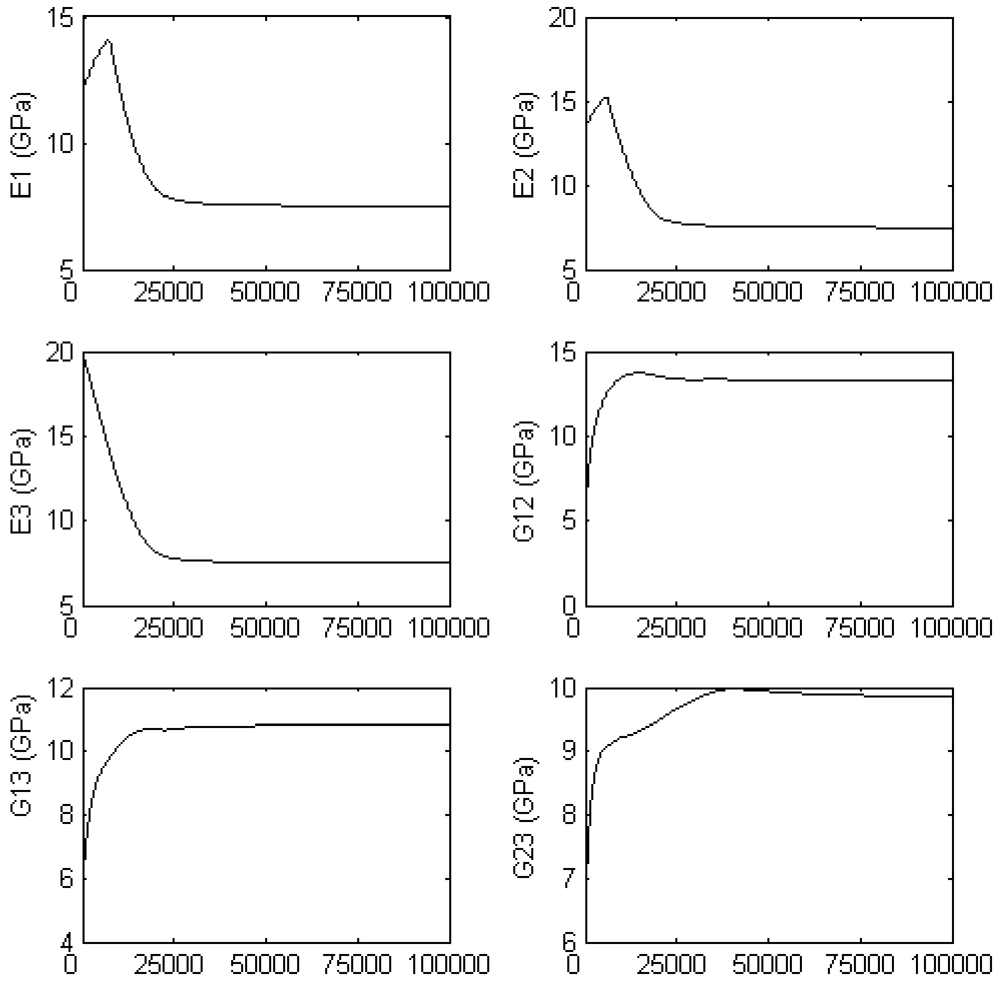

Figure 8.5 Results of the updating algorithm for the elastic moduli in element 39. The horizontal axis shows number of load steps.

In Tägil’s experiments bone existed in the chamber after they had been harvested. This is also shown in these calculations. To examine the existence of bone and the character of bone formation in the present model the hypothesis of Claes et al. [3] was applied. In his work Claes uses several finite element models that represent different stages in the healing process of a fracture. The results are compared with animal experiments and conclusions are drawn regarding the formation of different tissue types. The results from each element in the present calculation are plotted in Figure 8.6. Figure 8.7 shows the figure from Claes et al. [3].

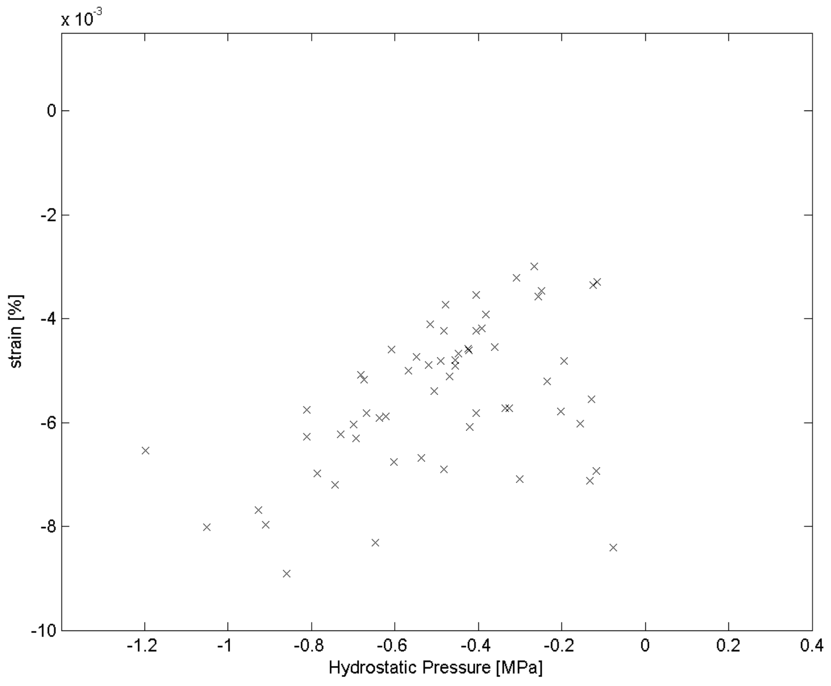

Figure 8.6 The hypothesis of Claes et al. implemented on the 3D geometry.

In Figure 8.7 area A, intramembranous ossification means that bone is formed directly from soft connective tissue. Bones such as the frontal and the parietal bones of the skull grow through intramembranous ossification. Area B, endochondral means that the bone growth is preceded by cartilage. The majority of bones in the skeleton grow through the process of endochondral ossification, c.f. [10].

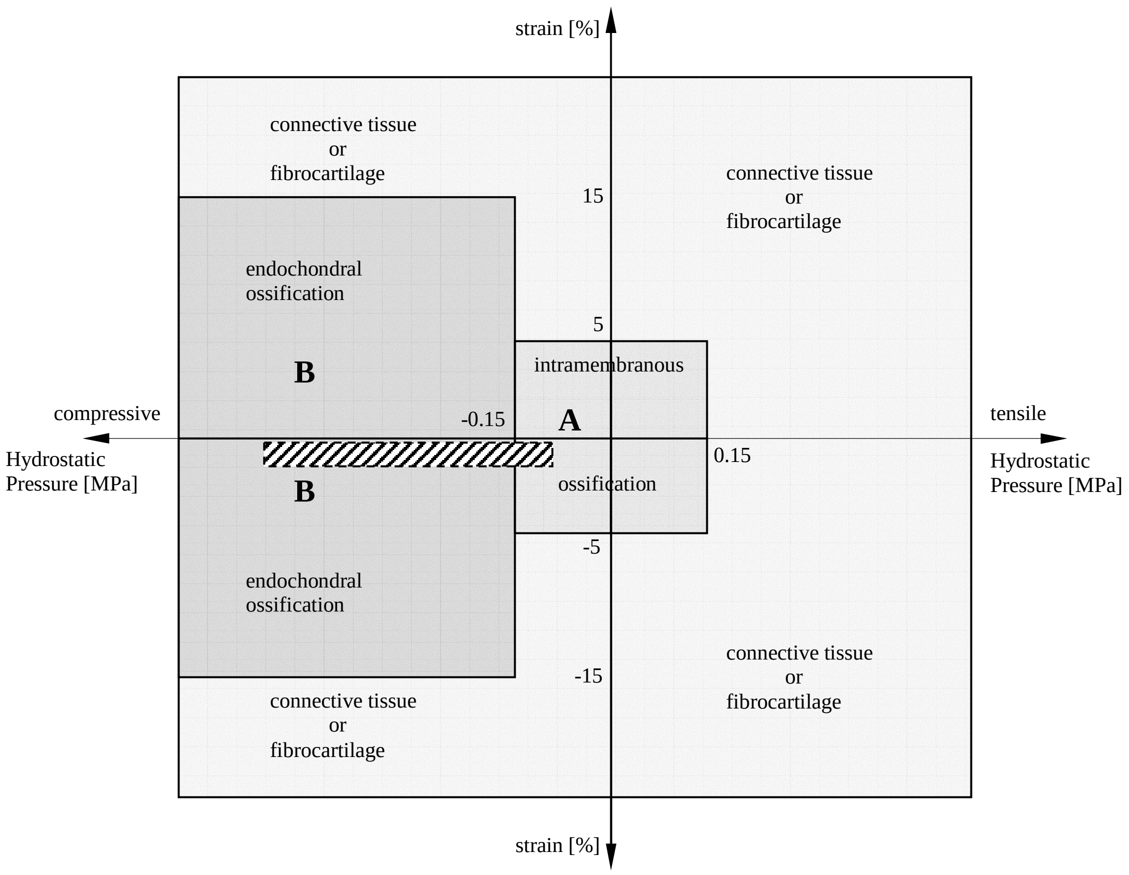
strain [%]
connective tissue or fibrocartilage
connective tissue or fibrocartilage
15
endochondral ossification
5
B
intramembranous
compressive
-0.15
A
tensile
Hydrostatic Pressure [MPa]
0.15
Hydrostatic Pressure [MPa]
B
ossification
-5
endochondral ossification
connective tissue or fibrocartilage
-15
connective tissue or fibrocartilage
strain [%]

Figure 8.7 The hypothesis of Claes et al.. The elements from the problem fall into the hatched area.

In Figure 8.6 the strain on the vertical axis is the normal strain in the direction of loading. A comparison of Figures 8.6 and 8.7 clearly shows that every element is within region A or B, i.e. the region for bone formation. This area is also schematically shown in Figure 8.7. This means that the presence of bone is justified.

8.2 Results for the axisymmetric geometry

This geometry which also contains 60 elements was only subjected to the adaptation of the elastic properties. The load situation analyzed here was to represent the presence of a medullary pin in the femur. The privileged directions for each element was prescribed and was assumed not to change throughout the adaptation process as described in chapter 7. The mesh contains six elements along the radial direction and ten elements along the axial direction resulting in a total of 60 elements. The numbering of the elements starts at the lower left hand corner with element one and ends in the upper right hand corner with element 60. On every plane the element number increases to the right. Figure 8.8 shows the adaptation process for element 11 and Figure 8.9 shows the adaptation process for element 40.

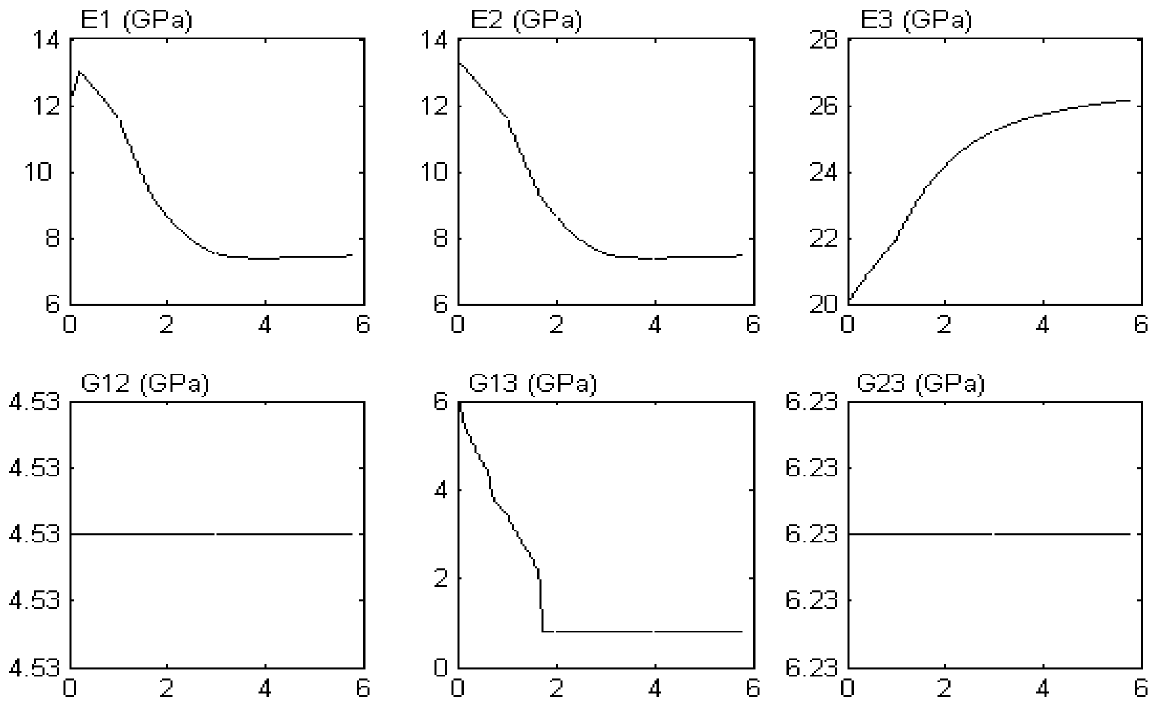

Figure 8.8 The adaptation of the elastic moduli for element 11. The horizontal axis shows number of steps*10^5.

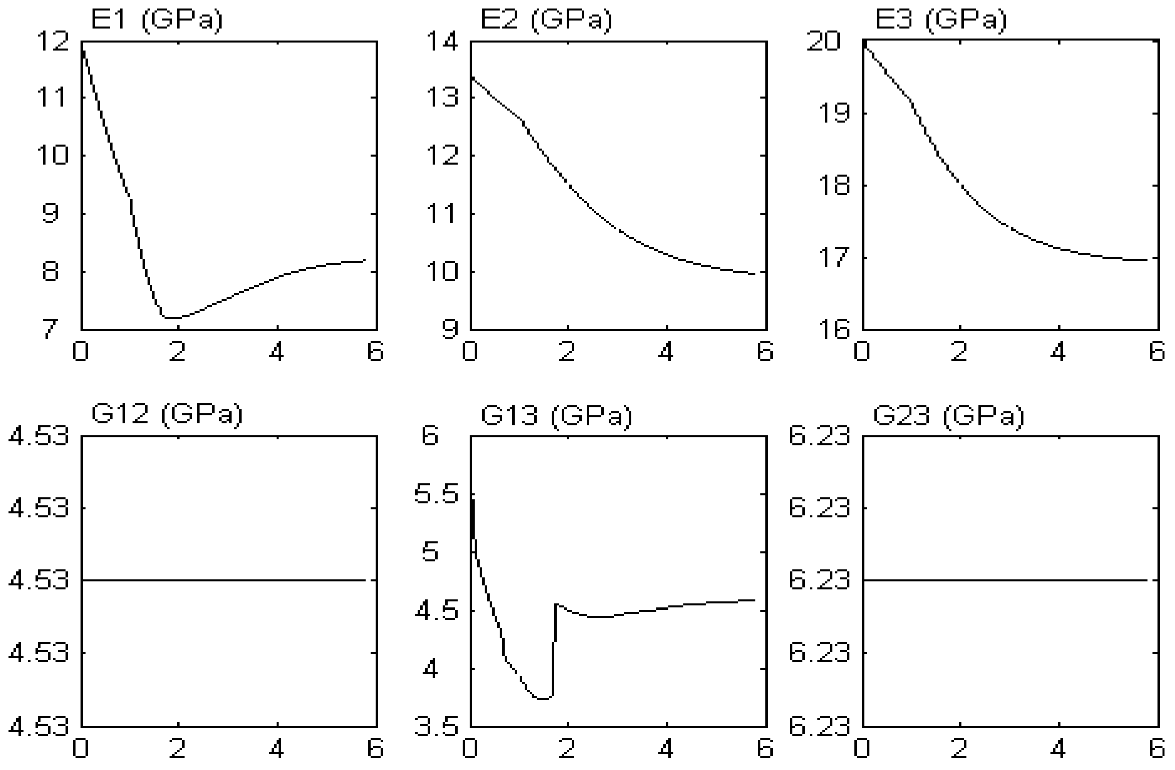

Figure 8.9 The adaptation of the elastic moduli for element 40. The horizontal axis shows number of steps*10^5.

The two elements show quite different results. This is due to the variation of position between the elements. G_{12} and G_{23} show no change in the whole process for both elements. This phenomenon is not seen for all elements though. The difference seen in E_3 for the two elements can be explained through their relative position. Element 11 lies much lower than element 40, thus absorbing a larger part of the shear force acting on the inner boundary indicating a greater need for load bearing capacity in the axial direction. Their relative position also influences the degree of orthogonality. Element 11 displays a much larger difference between the maximum and minimum elastic modulus than element 40.

Further the implementation of the hypothesis of Claes et al. is shown in Figure 8.10 and Figure 8.11 shows the hypothesis of Claes et al. schematically.

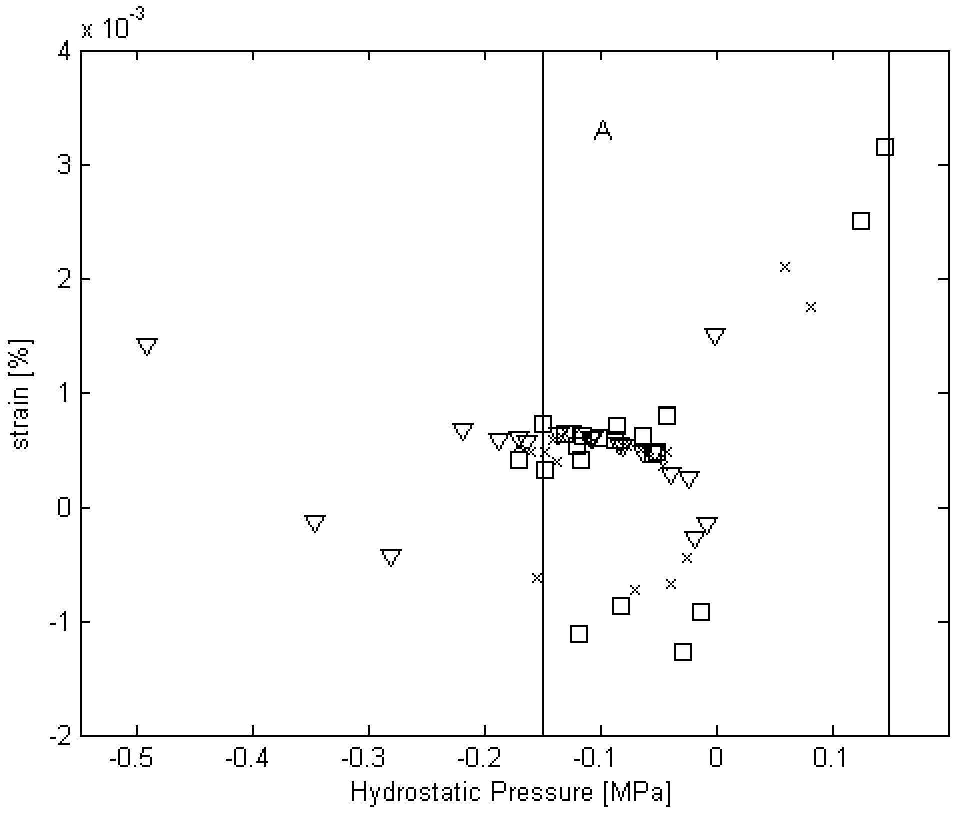

Figure 8.10 The hypothesis of Claes et al. implemented on the axisymmetric geometry. Elements from the two columns closest to the axis of symmetry are marked with squares. Elements from the two columns farthest from the axis of symmetry are marked with triangles. Elements from the two middle columns are marked with x. The two vertical lines mark the boundaries for the area A in Figure 8.10.

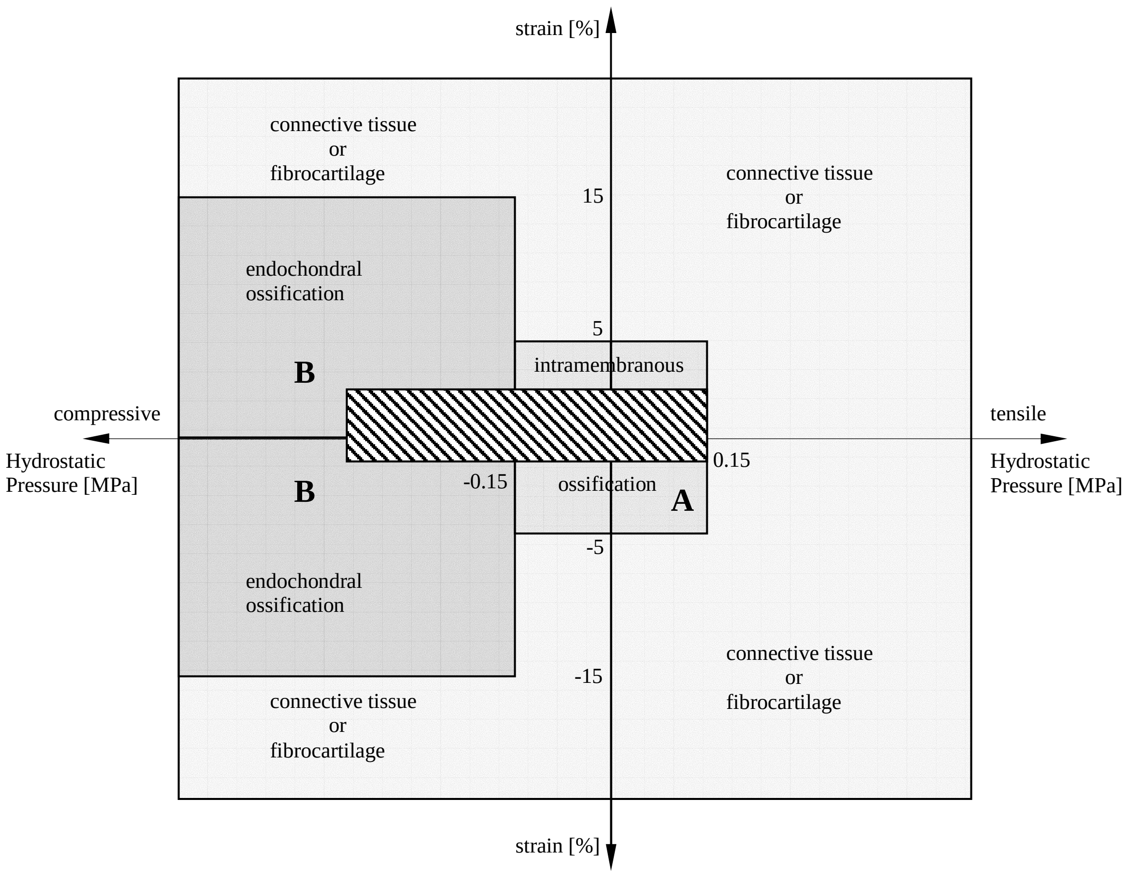
strain [%]
connective tissue or fibrocartilage
connective tissue or fibrocartilage
15
endochondral ossification
5
B
intramembranous
compressive
tensile
Hydrostatic Pressure [MPa]
Hydrostatic Pressure [MPa]
0.15
-0.15
B
ossification
A
-5
endochondral ossification
-15
connective tissue or fibrocartilage
connective tissue or fibrocartilage
strain [%]

Figure 8.11 The hypothesis of Claes et al.. The elements from the problem fall into the hatched area.

As seen in Figure 8.10, the elements in the axisymmetric geometry show much larger deviation than the elements in the 3D geometry. This is due to difference in load character, geometry and the boundary conditions. The hydrostatic pressure for all elements in the 3D application was negative while it for a few elements in the axisymmetric case was positive. Also the strains differ noticeably between the two analyzed cases. From Figure 8.10 it can be seen that elements from the two closest columns to the axis of symmetry tend to lie farthest to the right in the figure. Further elements from the two farthest columns to the axis of symmetry tend to lie more to the left in the figure than other elements. Elements from the two middle columns tend to lie between the other two column pairs.

Chapter 9

Future work

Throughout this work bone has been considered a continuum and only internal remodeling was investigated. No surface remodeling was assumed to take place. The two geometries were chosen to be quite simple.

Additional investigations have to be made both in the numerical and in the medical field to fully understand the nature of bone adaptation. The next step in the line of continuing this work would be to find an algorithmic tangential stiffness for the whole problem (directions of orthotropy and the elastic properties). Furthermore, in this work the load applied is held constant throughout the entire process but in real life this is seldom the case where dynamic forces appear alongside with static forces. To further increase the realism one should formulate some other stimulus that takes into account the dynamic loading as well as the strains. With this new stimulus, simulations could be performed where a periodic load is applied and predictions could be made of the macro structure in trabecular bone. Off course a new geometry in the shape of a real life bone will have to be defined so that the predictions made will be credible. Also this work deals solely with internal remodeling of bone. A more complete model would consider surface remodeling as well. Additional medical experiments will also have to be made so that more accurate numerical solutions can be made. As a last remark other types of constitutive models, such as poroelasticity can be investigated. This allows for other things to be accounted for such as fluid flow within the bone c.f. Papathanasopoulou et al [14].

Bibliography

[1] Ashman R. B., Cowin S. C., Van Buskirk W. C., Rice J. C. “A continuous wave technique for the measurement of the elastic properties of cortical bone” *Journal of Biomechanics* 17, 349-361, 1984

[2] Boehler J. P. Applications of Tensor Functions in Solid Mechanics, Springer Verlag Wien-New York, 1987

[3] Claes L. E., Heigele C. A. ”Magnitudes of local stress and strain along bony surfaces predict the course and type of fracture healing” *Journal of Biomechanics*, 32, 255-266, 1999

[4] Cowin S. C. “Mechanical Modeling of the Stress Adaptation Process in Bone” *Calcified Tissue International* 36, S98-S103, 1984

[5] Cowin S. C., Sadegh A. M., Luo G. M. “An Evolutionary Wolff’s Law for Trabecular Architecture” *Journal of Biomechanical Engineering* 114, 1992

[6] Cowin S. C., Van Buskirk W. C. ”Thermodynamic restrictions on the elastic constants of bone” *Journal of Biomechanics* 19, 85-87, 1986

[7] Harrysson M., “Adaptive bone remodelling” Master’s Dissertation, Division of Solid Mechanics, Lund, 2002

[8] Homminga J. “Towards a Rational Definition of Osteoporosis” Doctoral Dissertation, Technische Universitet Eindhoven, 2003

[9] Miller Z., Fuchs M. B., Arcan M. “Trabecular bone adaptation with an orthotropic material model” *Journal of Biomechanics* 35, 247-256, 2002

[10] Nigg B. M., Herzog W. “Biomechanics of the Musculo-Skeletal System” John Wiley & Sons, 1999

[11] Ottosen N. S., Petersson H. “Introduction to the Finite Element Method” Prentice Hall, 1992

[12] Ottosen N. S., Ristinmaa M. “The Mechanics of Constitutive Modelling”, volume 1, Division of Solid Mechanics, Lund, 1999

[13] Ottosen N. S., Ristinmaa M. “The Mechanics of Constitutive Modelling”, volume 2, Division of Solid Mechanics, Lund, 1999

[14] Papathanasopoulou V. A., Fotiadis D. I., Foutsitzi G., Massalas C. V. “A poroelastic bone model for internal remodeling” *International Journal of Engineering Science* 40, 511-530, 2002

[15] Shirazi-Adl A., Dammak M., Paiement G. “Experimental determination of friction characteristics at the trabecular bone/porous-coated metal interface in cementless implants” *Journal of Biomedical Materials Research* 27, 167-175, 1993

[16] Tägil, M., Aspenberg, P. “Cartilage Induction by Controlled Mechanical Stimulation *In Vivo*” *Journal of Orthopaedic Research* 17, 200-204, 1999

[17] Van Buskirk W. C., Ashman R. B. ”The Elastic Moduli of Bone” Mechanical Properties of Bone ASME, AMD 45, 1981

[18] Wolff J. “The Law of Bone Remodelling” Springer Verlag Berlin, 1986

[19] Zysset P. K., Guo X. E., Hoffler C. E., Moore K. E., Goldstein S. A. ”Elastic modulus and hardness of cortical and trabecular bone lamellae measured by nanoindentation in the human femur” *Journal of Biomechanics* 32, 1005-1012, 1999

[20] “CALFEM A finite element toolbox to MATLAB Version 3.3” Department of Mechanics and Materials, Lund, 1999

Appendix A

Matlab code

The Matlab code is presented below with the geometry files omitted.

```
clear all; close all;
format long
load c:\Ort13D  %Load geometry
load c:\Dmatrix %D-matrix, Ashman et al. 1984

V=rnddir(nel); %randomly assign privileged directions in each
               %element
Vin=V;
K=zeros(ndof);
Kold=zeros(ndof);
f=zeros(ndof,1);
fint=zeros(ndof,1);
a=zeros(ndof,1);

sigma=-2e6; %normal load in top (Pa)
tau=-sigma/8; %shear load on side (Pa)
nstep=100000;
dt=1/10;
Emod=zeros(6,nel,nstep);
Emod(1,:,1)=12e9; Emod(2,:,1)=13.4e9; Emod(3,:,1)=20e9;
Emod(4,:,1)=4.53e9; Emod(5,:,1)=5.61e9; Emod(6,:,1)=6.23e9;
tol1=1;

Vol=0;
Vole=zeros(nel,1);

%-----------------Assign global load vector-------------------
for j=1:nel
    switch j
        case {49,50,51,52,53,54,55,56} %on top
            fe=normalforce(Ex(j,:),Ey(j,:),Ez(j,:),sigma);
                                                     %normal force
        case {9,10,11,12,21,22,23,24,33,34,…
                35,36,45,46,47,48,57,58,59,60} %on side
            fe=shearforce(Ex(j,:),Ey(j,:),Ez(j,:),tau);
                                                    %shear force
        otherwise
            fe=zeros(24,1);
        end
        f=insert(Edof(j,:),f,fe');
        Del(:,:,j)=Dnew(D,V(:,:,j));
        D2(:,:,j)=D;
        Vole(j)=elvol(Ex(j,:),Ey(j,:),Ez(j,:));
        Vol=Vol+Vole(j);
end
f(bc(:,1))=0;
%--------------------------------------------------------------

i=0;
while norm(K-Kold)>tol1 | i==0
    i=i+1;
    [i norm(K-Kold)]
    Kold=K;
    a=zeros(ndof,1);
    Ed=zeros(nel,24);
    K=zeros(ndof);
    for j=1:nel
        Ke=zeros(24);
        Ke=soli8e(Ex(j,:),Ey(j,:),Ez(j,:),ep,Del(:,:,j));
        K=assem(Edof(j,:),K,Ke);
    end
    a=solveq(K,f,bc);
    Ed=extract(Edof,a);
    for j=1:nel
        et=zeros(ep^3,6);
        [es,et,eci]=soli8s(Ex(j,:),Ey(j,:),Ez(j,:),ep,…
                            Del(:,:,j),Ed(j,:));
        [V(:,:,j),N]=changedir(V(:,:,j),et,dt);
                            %calculate new directions
        Del(:,:,j)=Dnew(D,V(:,:,j));
    end
end
a=zeros(ndof,1);
tol2=0.0000001;
for n=1:nstep
    i=0;
    while norm(f-fint)>tol2 | i==0
        i=i+1;
        K=zeros(ndof);
        for j=1:nel
            if n==1 & i==1
                Ke=soli8e(Ex(j,:),Ey(j,:),Ez(j,:),ep,…
                            Del(:,:,j));
            else
                Ke=soli8e(Ex(j,:),Ey(j,:),Ez(j,:),ep,…
                            Dt(:,:,j));
            end
            K=assem(Edof(j,:),K,Ke);
        end
        da=solveq(K,(f-fint),bc);
        a=a+da;

%-------------------Reference stimulus-----------------------
        if n==1 & i==1
            psi0=PSI0(Ex,Ey,Ez,ep,Edof,da,V,…
                    Emod(:,:,1),Vol,Vole);
        end
%------------------------------------------------------------
        [Dt,Emodnew,fint]=DATS(Ex,Ey,Ez,Edof,bc,ep,dt,…
                            V,Emod(:,:,n),a,psi0);
        [n*1e10 i*1e10 norm(f-fint)*1e10 Emodnew(1,1)]
    end
    Emod(:,:,n+1)=Emodnew;
end
```

```
function [V]=rnddir(nel);
% [V]=rnddir(nel)
%
%Random directions generator
%
%V   - direction matrix, (3*3*nel)
%nel - number of elements, (1*1)

V=zeros(3,3,nel);
for i=1:nel
    V(:,1,i)=randn(3,1);
    V(:,1,i)=V(:,1,i)./norm(V(:,1,i));
    x=randn;
    y=randn;
    V(:,2,i)=[x;y;-(V(1,1,i)*x+V(2,1,i)*y)/V(3,1,i)];
    V(:,2,i)=V(:,2,i)./norm(V(:,2,i));
    V(:,3,i)=cross(V(:,1,i),V(:,2,i));
end
```

```
function [fenew]=normalforce(ex,ey,ez,sigma);
% [fenew]=normalforce(ex,ey,ez,sigma)
%
%Function for calculating the equivalent nodal forces for an
%evenly distributed normal load on the top (zeta=1) of an
%8-node isoparametric brick element
%
%fenew    - new local force vector, (24*1)
```

```
%ex,ey,ez - local coordinate matrices, (1*8)
%sigma    - load per area unit (Pa), (1*1)

fenew=zeros(24,1);
fenew(15)=sigma/12*(2*(ex(5)*ey(6)-ex(5)*ey(8)-ex(6)*ey(5)+…
           ex(8)*ey(5))+ex(6)*ey(7)+ex(6)*ey(8)-ex(7)*ey(6)+…
           ex(7)*ey(8)-ex(8)*ey(6)-ex(8)*ey(7));
fenew(18)=sigma/12*(2*(ex(5)*ey(6)-ex(6)*ey(5)+ex(6)*ey(7)-…
           ex(7)*ey(6))-ex(5)*ey(7)-ex(5)*ey(8)+ex(7)*ey(5)+…
           ex(7)*ey(8)+ex(8)*ey(5)-ex(8)*ey(7));
fenew(21)=sigma/12*(2*(ex(6)*ey(7)-ex(7)*ey(6)+ex(7)*ey(8)-…
           ex(8)*ey(7))+ex(5)*ey(6)-ex(5)*ey(8)-ex(6)*ey(5)-…
           ex(6)*ey(8)+ex(8)*ey(5)+ex(8)*ey(6));
fenew(24)=sigma/12*(2*(-ex(5)*ey(8)+ex(7)*ey(8)+ex(8)*ey(5)-…
           ex(8)*ey(7))+ex(5)*ey(6)+ex(5)*ey(7)-ex(6)*ey(5)+…
           ex(6)*ey(7)-ex(7)*ey(5)-ex(7)*ey(6));
```

```
function [fenew]=shearforce(ex,ey,ez,tau);
% [fenew]=shearforce(ex,ey,ez,tau)
%
%Function for calculating the equivalent nodal forces for an
%evenly distributed shear load on the side (eta=1) of an
%8-node isoparametric brick element
%
%fenew    - new local force vector, (24*1)
%ex,ey,ez - local coordinate matrices, (1*8)
%tau      - load per area unit (Pa), (1*1)

fenew=zeros(24,1);
index=[9 12 21 24];

%----------------------Gauss points-------------------
xi  =1/sqrt(3)*[-1 1 1 -1];
zeta=1/sqrt(3)*[-1 -1 1 1];
eta=1;
%-------------------------------------------------------

%----------------Form functions--------------------
N(:,1)=1/8*((1+xi).*(1+eta).*(1-zeta))'; %N3
N(:,2)=1/8*((1-xi).*(1+eta).*(1-zeta))'; %N4
N(:,3)=1/8*((1+xi).*(1+eta).*(1+zeta))'; %N7
N(:,4)=1/8*((1-xi).*(1+eta).*(1+zeta))'; %N8
%------------------------------------------------------

a1=1/4*[(1-zeta).*ex(3)+(-1+zeta).*ex(4)+(1+zeta).*ex(7)+…
        (-1-zeta).*ex(8);…
        (1-zeta).*ey(3)+(-1+zeta).*ey(4)+(1+zeta).*ey(7)+…
        (-1-zeta).*ey(8);…
        (1-zeta).*ez(3)+(-1+zeta).*ez(4)+(1+zeta).*ez(7)+…
        (-1-zeta).*ez(8)];
```

```
b1=1/4*[(-1-xi).*ex(3)+(-1+xi).*ex(4)+(1+xi).*ex(7)+…
        (1-xi).*ex(8);
        (-1-xi).*ey(3)+(-1+xi).*ey(4)+(1+xi).*ey(7)+…
        (1-xi).*ey(8);
        (-1-xi).*ez(3)+(-1+xi).*ez(4)+(1+xi).*ez(7)+…
        (1-xi).*ez(8)];

for i=1:4
    for j=1:4
        fenew(index(i))=fenew(index(i))+…
                   tau*N(i,j)*norm(cross(a1(:,j),b1(:,j)));
    end
end
```

```
function vol=elvol(ex,ey,ez)
% vol=elvol(ex,ey,ez)
%
%Function for calculating the element volume for
%an element with even height
%
%vol      - the volume for an element, (1*1)
%ex,ey,ez - local coordinate matrices, (1*8)

a=[ex(2)-ex(1);ey(2)-ey(1);0];
b=[ex(4)-ex(1);ey(4)-ey(1);0];
c=[ex(2)-ex(3);ey(2)-ey(3);0];
d=[ex(4)-ex(3);ey(4)-ey(3);0];
vol=1/2*(ez(5)-ez(1))*(abs(a(1)*b(2)-a(2)*b(1))+…
        abs(c(1)*d(2)-c(2)*d(1)));
```

```
function [Vnew,N]=changedir(Vold,et,dt);
% [Vnew]=changedir(Vold,et);
%
%Function for calculatiing new privileged directions
%of orthotropy for one element
%
%Vnew - the new privileged directions, (3*3)
%Vold - the old privileged directions, (3*3)
%et   - element strains, (ngp^3*6)
%dt   - time step, (1*1)

tol=eps;
Et=[sum(et(:,1))/8   sum(et(:,4))/16 sum(et(:,6))/16;
    sum(et(:,4))/16 sum(et(:,2))/8   sum(et(:,5))/16;
    sum(et(:,6))/16 sum(et(:,5))/16 sum(et(:,3))/8];
[N,E]=eig(Et); %N-vectors, E-values
if (Vold(:,3)'*N(:,2))<tol
    Vnew=Vold;
```

```
else
    if abs(E(3,3))>abs(E(1,1))% v3 --> n3
        h1=dt*Vold(:,3)'*N(:,3)*trace(E)^2/trace(E^2);
        h2=-h1*(Vold(:,2)'*N(:,3))/(Vold(:,3)'*N(:,2));
        Vnew(:,3)=Vold(:,3)+h1*N(:,3);
        Vnew(:,2)=Vold(:,2)+h2*N(:,2);
    else  % v3 --> n1
        h1=dt*Vold(:,3)'*N(:,1)*trace(E)^2/trace(E^2);
        h2=-h1*(Vold(:,2)'*N(:,1))/(Vold(:,3)'*N(:,2));
        Vnew(:,3)=Vold(:,3)+h1*N(:,1);
        Vnew(:,2)=Vold(:,2)+h2*N(:,2);
    end
    Vnew(:,3)=Vnew(:,3)./norm(Vnew(:,3));
    Vnew(:,2)=Vnew(:,2)./norm(Vnew(:,2));
    Vnew(:,1)=cross(Vnew(:,2),Vnew(:,3));
end
```

```
function [Dnew]=Dnew(D,v)
% [Dnew]=Dnew(D,v)
%
%Function for calculating the new D-matrix for
%the new privileged directions for one element
%
%Dnew - the new D-matrix, (6*6)
%D    - the D-matrix written in the privileged directions,
%                                                    (6*6)
%v    - the privileged directions, (3*3)

Dnew=zeros(6);
I=eye(3);
a4=D(5,5)+D(6,6)-D(4,4);
a5=D(6,6)+D(4,4)-D(5,5);
a6=D(4,4)+D(5,5)-D(6,6);
b1=D(1,1)+2*D(4,4)-2*D(5,5)-2*D(6,6);
b2=D(1,2);
c2=D(2,2)+2*D(5,5)-2*D(6,6)-2*D(4,4);
c3=D(2,3);
d1=D(1,3);
d3=D(3,3)+2*D(6,6)-2*D(4,4)-2*D(5,5);

M1=v(:,1)*v(:,1)';
M2=v(:,2)*v(:,2)';
M3=v(:,3)*v(:,3)';

for i=1:3
    Dnew(i,i)=b1*M1(i,i)*M1(i,i)+b2*M1(i,i)*M2(i,i)+…
              d1*M1(i,i)*M3(i,i)+b2*M2(i,i)*M1(i,i)+…
              c2*M2(i,i)*M2(i,i)+c3*M2(i,i)*M3(i,i)+…
              d1*M3(i,i)*M1(i,i)+c3*M3(i,i)*M2(i,i)+…
              d3*M3(i,i)*M3(i,i)+a4*(M1(i,i)+M1(i,i))+…
```

```
                a5*(M2(i,i)+M2(i,i))+a6*(M3(i,i)+M3(i,i));
end
Dnew(1,2)=b1*M1(1,1)*M1(2,2)+b2*M1(1,1)*M2(2,2)+…
           d1*M1(1,1)*M3(2,2)+b2*M2(1,1)*M1(2,2)+…
           c2*M2(1,1)*M2(2,2)+c3*M2(1,1)*M3(2,2)+…
           d1*M3(1,1)*M1(2,2)+c3*M3(1,1)*M2(2,2)+…
           d3*M3(1,1)*M3(2,2);
Dnew(1,3)=b1*M1(1,1)*M1(3,3)+b2*M1(1,1)*M2(3,3)+…
           d1*M1(1,1)*M3(3,3)+b2*M2(1,1)*M1(3,3)+…
           c2*M2(1,1)*M2(3,3)+c3*M2(1,1)*M3(3,3)+…
           d1*M3(1,1)*M1(3,3)+c3*M3(1,1)*M2(3,3)+…
           d3*M3(1,1)*M3(3,3);
Dnew(1,4)=b1*M1(1,1)*M1(1,2)+b2*M1(1,1)*M2(1,2)+…
           d1*M1(1,1)*M3(1,2)+b2*M2(1,1)*M1(1,2)+…
           c2*M2(1,1)*M2(1,2)+c3*M2(1,1)*M3(1,2)+…
           d1*M3(1,1)*M1(1,2)+c3*M3(1,1)*M2(1,2)+…
           d3*M3(1,1)*M3(1,2)+a4*M1(2,1)+…
           a5*M2(2,1)+a6*M3(2,1);
Dnew(1,5)=b1*M1(1,1)*M1(1,3)+b2*M1(1,1)*M2(1,3)+…
           d1*M1(1,1)*M3(1,3)+b2*M2(1,1)*M1(1,3)+…
           c2*M2(1,1)*M2(1,3)+c3*M2(1,1)*M3(1,3)+…
           d1*M3(1,1)*M1(1,3)+c3*M3(1,1)*M2(1,3)+…
           d3*M3(1,1)*M3(1,3)+a4*M1(3,1)+…
           a5*M2(3,1)+a6*M3(3,1);
Dnew(1,6)=b1*M1(1,1)*M1(2,3)+b2*M1(1,1)*M2(2,3)+…
           d1*M1(1,1)*M3(2,3)+b2*M2(1,1)*M1(2,3)+…
           c2*M2(1,1)*M2(2,3)+c3*M2(1,1)*M3(2,3)+…
           d1*M3(1,1)*M1(2,3)+c3*M3(1,1)*M2(2,3)+…
           d3*M3(1,1)*M3(2,3);
Dnew(2,3)=b1*M1(2,2)*M1(3,3)+b2*M1(2,2)*M2(3,3)+…
           d1*M1(2,2)*M3(3,3)+b2*M2(2,2)*M1(3,3)+…
           c2*M2(2,2)*M2(3,3)+c3*M2(2,2)*M3(3,3)+…
           d1*M3(2,2)*M1(3,3)+c3*M3(2,2)*M2(3,3)+…
           d3*M3(2,2)*M3(3,3);
Dnew(2,4)=b1*M1(2,2)*M1(1,2)+b2*M1(2,2)*M2(1,2)+…
           d1*M1(2,2)*M3(1,2)+b2*M2(2,2)*M1(1,2)+…
           c2*M2(2,2)*M2(1,2)+c3*M2(2,2)*M3(1,2)+…
           d1*M3(2,2)*M1(1,2)+c3*M3(2,2)*M2(1,2)+…
           d3*M3(2,2)*M3(1,2)+a4*M1(2,1)+…
           a5*M2(2,1)+a6*M3(2,1);
Dnew(2,5)=b1*M1(2,2)*M1(1,3)+b2*M1(2,2)*M2(1,3)+…
           d1*M1(2,2)*M3(1,3)+b2*M2(2,2)*M1(1,3)+…
           c2*M2(2,2)*M2(1,3)+c3*M2(2,2)*M3(1,3)+…
           d1*M3(2,2)*M1(1,3)+c3*M3(2,2)*M2(1,3)+…
           d3*M3(2,2)*M3(1,3);
Dnew(2,6)=b1*M1(2,2)*M1(2,3)+b2*M1(2,2)*M2(2,3)+…
           d1*M1(2,2)*M3(2,3)+b2*M2(2,2)*M1(2,3)+…
           c2*M2(2,2)*M2(2,3)+c3*M2(2,2)*M3(2,3)+…
           d1*M3(2,2)*M1(2,3)+c3*M3(2,2)*M2(2,3)+…
           d3*M3(2,2)*M3(2,3)+a4*M1(3,2)+…
           a5*M2(3,2)+a6*M3(3,2);
```

```
Dnew(3,4)=b1*M1(3,3)*M1(1,2)+b2*M1(3,3)*M2(1,2)+…
             d1*M1(3,3)*M3(1,2)+b2*M2(3,3)*M1(1,2)+…
             c2*M2(3,3)*M2(1,2)+c3*M2(3,3)*M3(1,2)+…
             d1*M3(3,3)*M1(1,2)+c3*M3(3,3)*M2(1,2)+…
             d3*M3(3,3)*M3(1,2);
Dnew(3,5)=b1*M1(3,3)*M1(1,3)+b2*M1(3,3)*M2(1,3)+…
             d1*M1(3,3)*M3(1,3)+b2*M2(3,3)*M1(1,3)+…
             c2*M2(3,3)*M2(1,3)+c3*M2(3,3)*M3(1,3)+…
             d1*M3(3,3)*M1(1,3)+c3*M3(3,3)*M2(1,3)+…
             d3*M3(3,3)*M3(1,3)+a4*M1(3,1)+…
             a5*M2(3,1)+a6*M3(3,1);
Dnew(3,6)=b1*M1(3,3)*M1(2,3)+b2*M1(3,3)*M2(2,3)+…
             d1*M1(3,3)*M3(2,3)+b2*M2(3,3)*M1(2,3)+…
             c2*M2(3,3)*M2(2,3)+c3*M2(3,3)*M3(2,3)+…
             d1*M3(3,3)*M1(2,3)+c3*M3(3,3)*M2(2,3)+…
             d3*M3(3,3)*M3(2,3)+a4*M1(3,2)+…
             a5*M2(3,2)+a6*M3(3,2);
Dnew(4,4)=b1*M1(1,2)*M1(1,2)+b2*M1(1,2)*M2(1,2)+…
             d1*M1(1,2)*M3(1,2)+b2*M2(1,2)*M1(1,2)+…
             c2*M2(1,2)*M2(1,2)+c3*M2(1,2)*M3(1,2)+…
             d1*M3(1,2)*M1(1,2)+c3*M3(1,2)*M2(1,2)+…
             d3*M3(1,2)*M3(1,2)+a4*(M1(1,1)+M1(2,2))+…
             a5*(M2(1,1)+M2(2,2))+a6*(M3(1,1)+M3(2,2));
Dnew(4,5)=b1*M1(1,2)*M1(1,3)+b2*M1(1,2)*M2(1,3)+…
             d1*M1(1,2)*M3(1,3)+b2*M2(1,2)*M1(1,3)+…
             c2*M2(1,2)*M2(1,3)+c3*M2(1,2)*M3(1,3)+…
             d1*M3(1,2)*M1(1,3)+c3*M3(1,2)*M2(1,3)+…
             d3*M3(1,2)*M3(1,3)+a4*M1(3,2)+…
             a5*M2(3,2)+a6*M3(3,2);
Dnew(4,6)=b1*M1(1,2)*M1(2,3)+b2*M1(1,2)*M2(2,3)+…
             d1*M1(1,2)*M3(2,3)+b2*M2(1,2)*M1(2,3)+…
             c2*M2(1,2)*M2(2,3)+c3*M2(1,2)*M3(2,3)+…
             d1*M3(1,2)*M1(2,3)+c3*M3(1,2)*M2(2,3)+…
             d3*M3(1,2)*M3(2,3);
Dnew(5,5)=b1*M1(1,3)*M1(1,3)+b2*M1(1,3)*M2(1,3)+…
             d1*M1(1,3)*M3(1,3)+b2*M2(1,3)*M1(1,3)+…
             c2*M2(1,3)*M2(1,3)+c3*M2(1,3)*M3(1,3)+…
             d1*M3(1,3)*M1(1,3)+c3*M3(1,3)*M2(1,3)+…
             d3*M3(1,3)*M3(1,3)+a4*(M1(1,1)+M1(3,3))+…
             a5*(M2(1,1)+M2(3,3))+a6*(M3(1,1)+M3(3,3));
Dnew(5,6)=b1*M1(1,3)*M1(2,3)+b2*M1(1,3)*M2(2,3)+…
             d1*M1(1,3)*M3(2,3)+b2*M2(1,3)*M1(2,3)+…
             c2*M2(1,3)*M2(2,3)+c3*M2(1,3)*M3(2,3)+…
             d1*M3(1,3)*M1(2,3)+c3*M3(1,3)*M2(2,3)+…
             d3*M3(1,3)*M3(2,3)+a4*M1(1,2)+…
             a5*M2(1,2)+a6*M3(1,2);
Dnew(6,6)=b1*M1(2,3)*M1(2,3)+b2*M1(2,3)*M2(2,3)+…
             d1*M1(2,3)*M3(2,3)+b2*M2(2,3)*M1(2,3)+…
             c2*M2(2,3)*M2(2,3)+c3*M2(2,3)*M3(2,3)+…
             d1*M3(2,3)*M1(2,3)+c3*M3(2,3)*M2(2,3)+…
             d3*M3(2,3)*M3(2,3)+a4*(M1(2,2)+M1(3,3))+…
```

```
           a5*(M2(2,2)+M2(3,3))+a6*(M3(2,2)+M3(3,3));
Dnew(2,1)=Dnew(1,2);
Dnew(3,1)=Dnew(1,3);
Dnew(4,1)=Dnew(1,4);
Dnew(5,1)=Dnew(1,5);
Dnew(6,1)=Dnew(1,6);
Dnew(3,2)=Dnew(2,3);
Dnew(4,2)=Dnew(2,4);
Dnew(5,2)=Dnew(2,5);
Dnew(6,2)=Dnew(2,6);
Dnew(4,3)=Dnew(3,4);
Dnew(5,3)=Dnew(3,5);
Dnew(6,3)=Dnew(3,6);
Dnew(5,4)=Dnew(4,5);
Dnew(6,4)=Dnew(4,6);
Dnew(6,5)=Dnew(5,6);
```

```
function psi0=PSI0(Ex,Ey,Ez,ep,Edof,a,V,Emod,Vol,Vole)
% psi0=PSI0(Ex,Ey,Ez,ep,Edof,a,V,Emod,Vol,Vole)
%
%Function for calculating the reference stimulus for
%3D geometry or axisymmetric geometry
%
%psi0     - the reference stimulus, (1*6)
%Ex,Ey,Ez - the global coordinate matrices, (nel*8) or (nel*4)
%           Ez for axisymmetry is an empty matrix
%ep       - element property, (1*1) or (1*3)
%Edof     - the global topology matrix, (nel*24) or (nel*9)
%a        - nodal displacements, (ndof*1)
%V        - the privileged directions, (3*3*nel)
%Emod     - the elastic properties, (6*nel)
%Vol      - the total volume of the geometry, (1*1)
%Vole     - the volume for each element, (nel,1)

nel=size(Emod,2);
c1=1e14;
psi0=zeros(1,6);
Ed=feval('extract',Edof,a);
if length(ep)==1 %3D
    [Es,Et,Eci]=feval('soli8s',Ex,Ey,Ez,ep,eye(6),Ed);
    ngp=ep^3;
else
     [Es,Et,Eci]=feval('plani4saxi',Ex,Ey,ep,eye(6),Ed);
    ngp=ep(3)^2;
end
for j=1:nel
    M(:,:,1)=V(:,1,j)*V(:,1,j)';
    M(:,:,2)=V(:,2,j)*V(:,2,j)';
    M(:,:,3)=V(:,3,j)*V(:,3,j)';
    M(:,:,4)=V(:,1,j)*V(:,2,j)'+V(:,2,j)*V(:,1,j)';
    M(:,:,5)=V(:,1,j)*V(:,3,j)'+V(:,3,j)*V(:,1,j)';
    M(:,:,6)=V(:,2,j)*V(:,3,j)'+V(:,3,j)*V(:,2,j)';
    eps=[sum(Et(j*ngp-(ngp-1):j*ngp,1))…
        sum(Et(j*ngp-(ngp-1):j*ngp,4))/2…
        sum(Et(j*ngp-(ngp-1):j*ngp,5))/2;
        sum(Et(j*ngp-(ngp-1):j*ngp,4))/2…
        sum(Et(j*ngp-(ngp-1):j*ngp,2))…
        sum(Et(j*ngp-(ngp-1):j*ngp,6))/2;
        sum(Et(j*ngp-(ngp-1):j*ngp,5))/2…
        sum(Et(j*ngp-(ngp-1):j*ngp,6))/2…
        sum(Et(j*ngp-(ngp-1):j*ngp,3))]/ngp;
    for k=1:6
        psi0(k)=psi0(k)+Vole(j)*c1/Emod(k,j)*…
                    abs(trace(eps*M(:,:,k)));
    end
end
psi0=psi0/Vol;
```

```
function
[Dt,Emodnew,fint]=DATS(Ex,Ey,Ez,Edof,bc,ep,dt,V,Emod,a,psi0)
% [Dt,Emodnew,fint]=DATS(Ex,Ey,Ez,Edof,bc,ep,dt,V,Emod,a,psi0)
%
%Function for calculating the algorithmic tangential stiffness
%for 3D geometry or axisymmetric geometry
%
%Dt        - the algorithmic tangential stiffness, (6*6*nel)
%Emodnew   - the new elastic properties, (6*nel)
%fint      - the internal forces, (ndof*1)
%Ex,Ey,Ez - the global coordinate matrices, (nel*8) or (nel*4)
%           Ez for axisymmetry is an empty matrix
%Edof      - the global topology matrix, (nel*24) or (nel*9)
%bc        - boundary conditions
%ep        - element properties, (1*1) or (1*3)
%V         - the privileged directions, (3*3*nel)
%Emod      - the elastic properties from the last state of
%            equilibrium, (6*nel)
%a         - the nodal displacements, (ndof*1)
%psi0      - the reference stimulus, (1*6)

%------------------------Constants----------------------------
c1=1e14;
c2=3.5e6;
nel=size(Edof,1);
ndof=max(max(Edof(:,2:end)));
index=[2 3 1];
ny=[0.422 0.35 0.371 0.222]; %[ny21 ny32 ny31 ny13]
Emin=[2e6 2e6 2e6 1e6 1e6 1e6];
fint=zeros(ndof,1);
if length(ep)==1 %3D
    ngp=ep^3;
```

```
else %axi
    ngp=ep(3)^2;
end
%--------------------------------------------------------------

Ed=feval('extract',Edof,a);
if length(ep)==1 %3D
    [Es,Et,Eci]=feval('soli8s',Ex,Ey,Ez,ep,eye(6),Ed);
else %axi
    [Es,Et,Eci]=feval('plani4saxi',Ex,Ey,ep,eye(6),Ed);
end
Emodnew=Emod;
for j=1:nel
    eps=[sum(Et(j*ngp-(ngp-1):j*ngp,1))…
        sum(Et(j*ngp-(ngp-1):j*ngp,4))/2…
        sum(Et(j*ngp-(ngp-1):j*ngp,5))/2;
        sum(Et(j*ngp-(ngp-1):j*ngp,4))/2…
        sum(Et(j*ngp-(ngp-1):j*ngp,2))…
        sum(Et(j*ngp-(ngp-1):j*ngp,6))/2;
        sum(Et(j*ngp-(ngp-1):j*ngp,5))/2…
        sum(Et(j*ngp-(ngp-1):j*ngp,6))/2…
        sum(Et(j*ngp-(ngp-1):j*ngp,3))]/ngp;
    M(:,:,1)=V(:,1,j)*V(:,1,j)'; M(:,:,2)=V(:,2,j)*V(:,2,j)';
    M(:,:,3)=V(:,3,j)*V(:,3,j)';
    M(:,:,4)=V(:,1,j)*V(:,2,j)'+V(:,2,j)*V(:,1,j)';
    M(:,:,5)=V(:,1,j)*V(:,3,j)'+V(:,3,j)*V(:,1,j)';
    M(:,:,6)=V(:,2,j)*V(:,3,j)'+V(:,3,j)*V(:,2,j)';
    Emodtest=Emod(:,j);
    for k=1:6
        A=(c2*dt*psi0(k)-Emod(k,j));
        B=c1*c2*dt*abs(trace(eps*M(:,:,k)));
        psi=c1/Emod(k,j)*abs(trace(eps*M(:,:,k)));
        Etest=-A/2+sqrt(A^2/4+B);
        if psi>psi0(k)
            p(k)=1;
        else
            p(k)=-1;
        end
        if Etest<Emin(k) Etest=Emin(k); end
        if k==1|k==2
            if Etest>Emod(k+1,j)
                Etest=Emod(k+1,j);
            end
        end
        Emodtest(k)=Etest;
        if k==1|k==2|k==3
            if ny(k)>(Emodtest(index(k))/Emodtest(k))^.5
                Etest=Emod(k,j);
                Emodtest(k)=Etest;
            end
            x=(1-ny(1)^2*Emodtest(1)/Emodtest(2)-…
                ny(2)^2*Emodtest(2)/Emodtest(3)-…
                ny(4)^2*Emodtest(3)/Emodtest(1))/2;
            if ny(1)*ny(2)*ny(4)>x | x>.5
                Etest=Emod(k,j);
                Emodtest(k)=Etest;
            end
        end
        F(k)=sqrt(A^2/4+B);
    end
    Emodnew(:,j)=Emodtest;
%--------------------Derivatives----------------------------
    dDdE1=feval('dDnew',Emodnew(:,j),ny(1:3),V(:,:,j),1);
    dDdE2=feval('dDnew',Emodnew(:,j),ny(1:3),V(:,:,j),2);
    dDdE3=feval('dDnew',Emodnew(:,j),ny(1:3),V(:,:,j),3);
    dDdG12=feval('dDnew',Emodnew(:,j),ny(1:3),V(:,:,j),4);
    dDdG13=feval('dDnew',Emodnew(:,j),ny(1:3),V(:,:,j),5);
    dDdG23=feval('dDnew',Emodnew(:,j),ny(1:3),V(:,:,j),6);
%------------------------------------------------------------
    D(:,:,j)=feval('dDnew',Emodnew(:,j),ny(1:3),V(:,:,j),7);
%----------------------------DATS----------------------------
    Dt(:,:,j)=(c1*c2*dt/2*(dDdE1*abs(trace(eps*M(:,:,1)))/…
        F(1)*p(1)+dDdE2*abs(trace(eps*M(:,:,2)))/F(2)*p(2)+…
        dDdE3*abs(trace(eps*M(:,:,3)))/F(3)*p(3)+…
        dDdG12*abs(trace(eps*M(:,:,4)))/F(4)*p(4)+…
        dDdG13*abs(trace(eps*M(:,:,5)))/F(5)*p(5)+…
        dDdG23*abs(trace(eps*M(:,:,6)))/F(6)*p(6))+D(:,:,j));
%------------------------------------------------------------
    finte=zeros(1,size(Edof,2)-1);
    if length(ep)==1 %3D
        [es,et,eci]=feval('soli8s',Ex(j,:),Ey(j,:),…
                           Ez(j,:),ep,D(:,:,j),Ed(j,:));
        finte=feval('soli8f',Ex(j,:),Ey(j,:),Ez(j,:),ep,es);
    else %axi
        [es,et,eci]=feval('plani4saxi',Ex(j,:),Ey(j,:),…
                           ep,D(:,:,j),Ed(j,:));
        finte=feval('plani4faxi',Ex(j,:),Ey(j,:),ep,es);
    end
    fint=feval('insert',Edof(j,:),fint,finte);
end
fint(bc(:,1))=0;
```

```
function [dDnew]=dDnew(Emod,ny,V,q)
% [dDnew]=dDnew(Emod,ny,V,q)
%
%Function for calculating the derivative of the D-matrix
%
%dDnew - the derivtive of the D-matrix, (6*6)
%Emod  - the elastic properties, (6*1)
%ny    - the Poisson's ratios, (1*3)
%V     - the privileged directions, (3*3)
```

```
%q      - variable that states with which elastic property
%         the derivation should be preformed

M1=V(:,1)*V(:,1)';
M2=V(:,2)*V(:,2)';
M3=V(:,3)*V(:,3)';
E11=Emod(1); E22=Emod(2); E33=Emod(3); G12=Emod(4);
G13=Emod(5); G23=Emod(6);
nu32=ny(2); nu21=ny(1); nu31=ny(3);
%--------------------Boehler's constants--------------------
b1=E11*E22*(E22*nu32^2-E33)/(E11*nu21^2*E33+…
    2*E11*E22*nu21*nu31*nu32+E22^2*nu32^2-E22*E33+…
    E22*E11*nu31^2)+2*G12-2*G13-2*G23;
c2=E22^2*(E11*nu31^2-E33)/(E11*nu21^2*E33+…
    2*E11*E22*nu21*nu31*nu32+E22^2*nu32^2-E22*E33+…
    E22*E11*nu31^2)+2*G13-2*G23-2*G12;
d3=E33^2*(E11*nu21^2-E22)/(E11*nu21^2*E33+…
    2*E11*E22*nu21*nu31*nu32+E22^2*nu32^2-…
    E22*E33+E22*E11*nu31^2)+2*G23-2*G12-2*G13;
a4=G13+G23-G12;
a5=G23+G12-G13;
a6=G12+G13-G23;
c3=-E22*E33*(E11*nu31*nu21+nu32*E22)/(E11*nu21^2*E33+…
    2*E11*E22*nu21*nu31*nu32+E22^2*nu32^2-…
    E22*E33+E22*E11*nu31^2);
d1=-E11*E22*E33*(nu31+nu21*nu32)/(E11*nu21^2*E33+…
    2*E11*E22*nu21*nu31*nu32+E22^2*nu32^2-…
    E22*E33+E22*E11*nu31^2);
b2=-E11*E22*(nu21*E33+E22*nu31*nu32)/(E11*nu21^2*E33+…
    2*E11*E22*nu21*nu31*nu32+E22^2*nu32^2-…
    E22*E33+E22*E11*nu31^2);
%-----------------------------------------------------------

%-----------------------The derivatives---------------------
db1dE11=E22^2*(E22*nu32^2-E33)^2/(E11*nu21^2*E33+…
        2*E11*E22*nu21*nu31*nu32+E22^2*nu32^2-…
        E22*E33+E22*E11*nu31^2)^2;
db1dE22=E11^2*(2*E22*nu32^2*nu21^2*E33+…
        2*E22^2*nu32^3*nu21*nu31+E22^2*nu32^2*nu31^2-…
        nu21^2*E33^2)/(E11*nu21^2*E33+…
        2*E11*E22*nu21*nu31*nu32+E22^2*nu32^2-…
        E22*E33+E22*E11*nu31^2)^2;
db1dE33=-E11^2*E22^2*(2*nu21*nu31*nu32+nu31^2+nu32^2*nu21^2)/…
        (E11*nu21^2*E33+2*E11*E22*nu21*nu31*nu32+E22^2*nu32^2-…
        E22*E33+E22*E11*nu31^2)^2;
db1dG12=2;
db1dG13=-2;
db1dG23=-2;

dc2dE11=E22^2*(E22^2*nu32^2*nu31^2+nu21^2*E33^2+…
        2*E33*E22*nu21*nu31*nu32)/(E11*nu21^2*E33+…
        2*E11*E22*nu21*nu31*nu32+E22^2*nu32^2-…
        E22*E33+E22*E11*nu31^2)^2;
dc2dE22=E22*(E11*nu31^2-E33)*(2*E11*nu21^2*E33+…
        2*E11*E22*nu21*nu31*nu32-E22*E33+E22*E11*nu31^2)/…
        (E11*nu21^2*E33+2*E11*E22*nu21*nu31*nu32+E22^2*nu32^2-…
        E22*E33+E22*E11*nu31^2)^2;
dc2dE33=-E22^2*(2*E11*E22*nu21*nu31*nu32+E22^2*nu32^2+…
        E11^2*nu31^2*nu21^2)/(E11*nu21^2*E33+…
        2*E11*E22*nu21*nu31*nu32+…
        E22^2*nu32^2-E22*E33+E22*E11*nu31^2)^2;
dc2dG12=-2;
dc2dG13=2;
dc2dG23=-2;

dd3dE11=E33^2*E22^2*(2*nu21*nu31*nu32+nu31^2+nu32^2*nu21^2)/…
        (E11*nu21^2*E33+2*E11*E22*nu21*nu31*nu32+E22^2*nu32^2-…
        E22*E33+E22*E11*nu31^2)^2;
dd3dE22=-E33^2*(-E22^2*nu32^2+2*E11^2*nu21^3*nu31*nu32+…
        2*E22*nu32^2*E11*nu21^2+E11^2*nu31^2*nu21^2)/…
        (E11*nu21^2*E33+2*E11*E22*nu21*nu31*nu32+E22^2*nu32^2-…
        E22*E33+E22*E11*nu31^2)^2;
dd3dE33=E33*(E11*nu21^2-E22)*(E11*nu21^2*E33+…
        4*E11*E22*nu21*nu31*nu32+2*E22^2*nu32^2-…
        E22*E33+2*E22*E11*nu31^2)/(E11*nu21^2*E33+…
        2*E11*E22*nu21*nu31*nu32+E22^2*nu32^2-…
        E22*E33+E22*E11*nu31^2)^2;
dd3dG12=-2;
dd3dG13=-2;
dd3dG23=2;

da4dE11=0;
da4dE22=0;
da4dE33=0;
da4dG12=-1;
da4dG13=1;
da4dG23=1;

da5dE11=0;
da5dE22=0;
da5dE33=0;
da5dG12=1;
da5dG13=-1;
da5dG23=1;

da6dE11=0;
da6dE22=0;
da6dE33=0;
da6dG12=1;
da6dG13=1;
da6dG23=-1;
```

```
dc3dE11=E22^2*E33*(nu31*nu21*E22*nu32^2+E33*nu31*nu21+…
        nu32*nu21^2*E33+nu32*E22*nu31^2)/…
        (E11*nu21^2*E33+2*E11*E22*nu21*nu31*nu32+…
        E22^2*nu32^2-E22*E33+E22*E11*nu31^2)^2;
dc3dE22=-E33*(E11^2*nu31*nu21^3*E33+…
        E11*nu31*nu21*E22^2*nu32^2+2*nu32*E22*E11*nu21^2*E33-…
        nu32*E22^2*E33+nu32*E22^2*E11*nu31^2)/…
        (E11*nu21^2*E33+2*E11*E22*nu21*nu31*nu32+…
        E22^2*nu32^2-E22*E33+E22*E11*nu31^2)^2;
dc3dE33=-E22^2*(E11*nu31*nu21+nu32*E22)*…
        (2*E11*nu21*nu31*nu32+E22*nu32^2+E11*nu31^2)/…
        (E11*nu21^2*E33+2*E11*E22*nu21*nu31*nu32+…
        E22^2*nu32^2-E22*E33+E22*E11*nu31^2)^2;
dc3dG12=0;
dc3dG13=0;
dc3dG23=0;

dd1dE11=-E22^2*E33*(nu31+nu21*nu32)*(E22*nu32^2-…
        E33)/(E11*nu21^2*E33+2*E11*E22*nu21*nu31*nu32+…
        E22^2*nu32^2-E22*E33+E22*E11*nu31^2)^2;
dd1dE22=-E11*E33*(nu31+nu21*nu32)*(E11*nu21^2*E33-…
        E22^2*nu32^2)/(E11*nu21^2*E33+…
        2*E11*E22*nu21*nu31*nu32+E22^2*nu32^2-…
        E22*E33+E22*E11*nu31^2)^2;
dd1dE33=-E11*E22^2*(nu31+nu21*nu32)*(2*E11*nu21*nu31*nu32+…
        E22*nu32^2+E11*nu31^2)/(E11*nu21^2*E33+…
        2*E11*E22*nu21*nu31*nu32+…
        E22^2*nu32^2-E22*E33+E22*E11*nu31^2)^2;
dd1dG12=0;
dd1dG13=0;
dd1dG23=0;

db2dE11=-E22^2*(nu21*E33+E22*nu31*nu32)*(E22*nu32^2-…
        E33)/(E11*nu21^2*E33+2*E11*E22*nu21*nu31*nu32+…
        E22^2*nu32^2-E22*E33+E22*E11*nu31^2)^2;
db2dE22=-E11*(nu21^3*E33^2*E11+…
        2*nu21^2*E33*E11*E22*nu31*nu32-nu21*E33*E22^2*nu32^2+…
        2*E22^2*nu31^2*nu32^2*E11*nu21-E22^2*nu31*nu32*E33+…
        E22^2*nu31^3*nu32*E11)/(E11*nu21^2*E33+…
        2*E11*E22*nu21*nu31*nu32+E22^2*nu32^2-…
        E22*E33+E22*E11*nu31^2)^2;
db2dE33=-E22^2*E11*(E11*nu21^2*nu31*nu32+…
        nu21*E22*nu32^2+nu21*E11*nu31^2+E22*nu31*nu32)/…
        (E11*nu21^2*E33+2*E11*E22*nu21*nu31*nu32+…
        E22^2*nu32^2-E22*E33+E22*E11*nu31^2)^2;
db2dG12=0;
db2dG13=0;
db2dG23=0;
%-------------------------------------------------------------
db1=[db1dE11 db1dE22 db1dE33 db1dG12 db1dG13 db1dG23 b1];
dc2=[dc2dE11 dc2dE22 dc2dE33 dc2dG12 dc2dG13 dc2dG23 c2];
```

```
dd3=[dd3dE11 dd3dE22 dd3dE33 dd3dG12 dd3dG13 dd3dG23 d3];
da4=[da4dE11 da4dE22 da4dE33 da4dG12 da4dG13 da4dG23 a4];
da5=[da5dE11 da5dE22 da5dE33 da5dG12 da5dG13 da5dG23 a5];
da6=[da6dE11 da6dE22 da6dE33 da6dG12 da6dG13 da6dG23 a6];
dc3=[dc3dE11 dc3dE22 dc3dE33 dc3dG12 dc3dG13 dc3dG23 c3];
dd1=[dd1dE11 dd1dE22 dd1dE33 dd1dG12 dd1dG13 dd1dG23 d1];
db2=[db2dE11 db2dE22 db2dE33 db2dG12 db2dG13 db2dG23 b2];

for i=1:3
    dDnew(i,i)=db1(q)*M1(i,i)*M1(i,i)+db2(q)*M1(i,i)*M2(i,i)+…
        dd1(q)*M1(i,i)*M3(i,i)+db2(q)*M2(i,i)*M1(i,i)+…
        dc2(q)*M2(i,i)*M2(i,i)+dc3(q)*M2(i,i)*M3(i,i)+…
        dd1(q)*M3(i,i)*M1(i,i)+dc3(q)*M3(i,i)*M2(i,i)+…
        dd3(q)*M3(i,i)*M3(i,i)+da4(q)*(M1(i,i)+M1(i,i))+…
        da5(q)*(M2(i,i)+M2(i,i))+…
        da6(q)*(M3(i,i)+M3(i,i));
end
dDnew(1,2)=db1(q)*M1(1,1)*M1(2,2)+db2(q)*M1(1,1)*M2(2,2)+…
            dd1(q)*M1(1,1)*M3(2,2)+db2(q)*M2(1,1)*M1(2,2)+…
            dc2(q)*M2(1,1)*M2(2,2)+dc3(q)*M2(1,1)*M3(2,2)+…
            dd1(q)*M3(1,1)*M1(2,2)+dc3(q)*M3(1,1)*M2(2,2)+…
            dd3(q)*M3(1,1)*M3(2,2);
dDnew(1,3)=db1(q)*M1(1,1)*M1(3,3)+db2(q)*M1(1,1)*M2(3,3)+…
            dd1(q)*M1(1,1)*M3(3,3)+db2(q)*M2(1,1)*M1(3,3)+…
            dc2(q)*M2(1,1)*M2(3,3)+dc3(q)*M2(1,1)*M3(3,3)+…
            dd1(q)*M3(1,1)*M1(3,3)+dc3(q)*M3(1,1)*M2(3,3)+…
            dd3(q)*M3(1,1)*M3(3,3);
dDnew(1,4)=db1(q)*M1(1,1)*M1(1,2)+db2(q)*M1(1,1)*M2(1,2)+…
            dd1(q)*M1(1,1)*M3(1,2)+db2(q)*M2(1,1)*M1(1,2)+…
            dc2(q)*M2(1,1)*M2(1,2)+dc3(q)*M2(1,1)*M3(1,2)+…
            dd1(q)*M3(1,1)*M1(1,2)+dc3(q)*M3(1,1)*M2(1,2)+…
            dd3(q)*M3(1,1)*M3(1,2)+da4(q)*M1(2,1)+…
            da5(q)*M2(2,1)+da6(q)*M3(2,1);
dDnew(1,5)=db1(q)*M1(1,1)*M1(1,3)+db2(q)*M1(1,1)*M2(1,3)+…
            dd1(q)*M1(1,1)*M3(1,3)+db2(q)*M2(1,1)*M1(1,3)+…
            dc2(q)*M2(1,1)*M2(1,3)+dc3(q)*M2(1,1)*M3(1,3)+…
            dd1(q)*M3(1,1)*M1(1,3)+dc3(q)*M3(1,1)*M2(1,3)+…
            dd3(q)*M3(1,1)*M3(1,3)+da4(q)*M1(3,1)+…
            da5(q)*M2(3,1)+da6(q)*M3(3,1);
dDnew(1,6)=db1(q)*M1(1,1)*M1(2,3)+db2(q)*M1(1,1)*M2(2,3)+…
            dd1(q)*M1(1,1)*M3(2,3)+db2(q)*M2(1,1)*M1(2,3)+…
            dc2(q)*M2(1,1)*M2(2,3)+dc3(q)*M2(1,1)*M3(2,3)+…
            dd1(q)*M3(1,1)*M1(2,3)+dc3(q)*M3(1,1)*M2(2,3)+…
            dd3(q)*M3(1,1)*M3(2,3);
dDnew(2,3)=db1(q)*M1(2,2)*M1(3,3)+db2(q)*M1(2,2)*M2(3,3)+…
            dd1(q)*M1(2,2)*M3(3,3)+db2(q)*M2(2,2)*M1(3,3)+…
            dc2(q)*M2(2,2)*M2(3,3)+dc3(q)*M2(2,2)*M3(3,3)+…
            dd1(q)*M3(2,2)*M1(3,3)+dc3(q)*M3(2,2)*M2(3,3)+…
            dd3(q)*M3(2,2)*M3(3,3);
dDnew(2,4)=db1(q)*M1(2,2)*M1(1,2)+db2(q)*M1(2,2)*M2(1,2)+…
            dd1(q)*M1(2,2)*M3(1,2)+db2(q)*M2(2,2)*M1(1,2)+…
```

```
              dc2(q)*M2(2,2)*M2(1,2)+dc3(q)*M2(2,2)*M3(1,2)+…
              dd1(q)*M3(2,2)*M1(1,2)+dc3(q)*M3(2,2)*M2(1,2)+…
              dd3(q)*M3(2,2)*M3(1,2)+da4(q)*M1(2,1)+…
              da5(q)*M2(2,1)+da6(q)*M3(2,1);
dDnew(2,5)=db1(q)*M1(2,2)*M1(1,3)+db2(q)*M1(2,2)*M2(1,3)+…
              dd1(q)*M1(2,2)*M3(1,3)+db2(q)*M2(2,2)*M1(1,3)+…
              dc2(q)*M2(2,2)*M2(1,3)+dc3(q)*M2(2,2)*M3(1,3)+…
              dd1(q)*M3(2,2)*M1(1,3)+dc3(q)*M3(2,2)*M2(1,3)+…
              dd3(q)*M3(2,2)*M3(1,3);
dDnew(2,6)=db1(q)*M1(2,2)*M1(2,3)+db2(q)*M1(2,2)*M2(2,3)+…
              dd1(q)*M1(2,2)*M3(2,3)+db2(q)*M2(2,2)*M1(2,3)+…
              dc2(q)*M2(2,2)*M2(2,3)+dc3(q)*M2(2,2)*M3(2,3)+…
              dd1(q)*M3(2,2)*M1(2,3)+dc3(q)*M3(2,2)*M2(2,3)+…
              dd3(q)*M3(2,2)*M3(2,3)+da4(q)*M1(3,2)+…
              da5(q)*M2(3,2)+da6(q)*M3(3,2);
dDnew(3,4)=db1(q)*M1(3,3)*M1(1,2)+db2(q)*M1(3,3)*M2(1,2)+…
              dd1(q)*M1(3,3)*M3(1,2)+db2(q)*M2(3,3)*M1(1,2)+…
              dc2(q)*M2(3,3)*M2(1,2)+dc3(q)*M2(3,3)*M3(1,2)+…
              dd1(q)*M3(3,3)*M1(1,2)+dc3(q)*M3(3,3)*M2(1,2)+…
              dd3(q)*M3(3,3)*M3(1,2);
dDnew(3,5)=db1(q)*M1(3,3)*M1(1,3)+db2(q)*M1(3,3)*M2(1,3)+…
              dd1(q)*M1(3,3)*M3(1,3)+db2(q)*M2(3,3)*M1(1,3)+…
              dc2(q)*M2(3,3)*M2(1,3)+dc3(q)*M2(3,3)*M3(1,3)+…
              dd1(q)*M3(3,3)*M1(1,3)+dc3(q)*M3(3,3)*M2(1,3)+…
              dd3(q)*M3(3,3)*M3(1,3)+da4(q)*M1(3,1)+…
              da5(q)*M2(3,1)+da6(q)*M3(3,1);
dDnew(3,6)=db1(q)*M1(3,3)*M1(2,3)+db2(q)*M1(3,3)*M2(2,3)+…
              dd1(q)*M1(3,3)*M3(2,3)+db2(q)*M2(3,3)*M1(2,3)+…
              dc2(q)*M2(3,3)*M2(2,3)+dc3(q)*M2(3,3)*M3(2,3)+…
              dd1(q)*M3(3,3)*M1(2,3)+dc3(q)*M3(3,3)*M2(2,3)+…
              dd3(q)*M3(3,3)*M3(2,3)+da4(q)*M1(3,2)+…
              da5(q)*M2(3,2)+da6(q)*M3(3,2);
dDnew(4,4)=db1(q)*M1(1,2)*M1(1,2)+db2(q)*M1(1,2)*M2(1,2)+…
              dd1(q)*M1(1,2)*M3(1,2)+db2(q)*M2(1,2)*M1(1,2)+…
              dc2(q)*M2(1,2)*M2(1,2)+dc3(q)*M2(1,2)*M3(1,2)+…
              dd1(q)*M3(1,2)*M1(1,2)+dc3(q)*M3(1,2)*M2(1,2)+…
              dd3(q)*M3(1,2)*M3(1,2)+da4(q)*(M1(1,1)+M1(2,2))+…
              da5(q)*(M2(1,1)+M2(2,2))+da6(q)*(M3(1,1)+M3(2,2));
dDnew(4,5)=db1(q)*M1(1,2)*M1(1,3)+db2(q)*M1(1,2)*M2(1,3)+…
              dd1(q)*M1(1,2)*M3(1,3)+db2(q)*M2(1,2)*M1(1,3)+…
              dc2(q)*M2(1,2)*M2(1,3)+dc3(q)*M2(1,2)*M3(1,3)+…
              dd1(q)*M3(1,2)*M1(1,3)+dc3(q)*M3(1,2)*M2(1,3)+…
              dd3(q)*M3(1,2)*M3(1,3)+da4(q)*M1(3,2)+…
              da5(q)*M2(3,2)+da6(q)*M3(3,2);
dDnew(4,6)=db1(q)*M1(1,2)*M1(2,3)+db2(q)*M1(1,2)*M2(2,3)+…
              dd1(q)*M1(1,2)*M3(2,3)+db2(q)*M2(1,2)*M1(2,3)+…
              dc2(q)*M2(1,2)*M2(2,3)+dc3(q)*M2(1,2)*M3(2,3)+…
              dd1(q)*M3(1,2)*M1(2,3)+dc3(q)*M3(1,2)*M2(2,3)+…
              dd3(q)*M3(1,2)*M3(2,3);
dDnew(5,5)=db1(q)*M1(1,3)*M1(1,3)+db2(q)*M1(1,3)*M2(1,3)+…
              dd1(q)*M1(1,3)*M3(1,3)+db2(q)*M2(1,3)*M1(1,3)+…
```

```
           dc2(q)*M2(1,3)*M2(1,3)+dc3(q)*M2(1,3)*M3(1,3)+…
           dd1(q)*M3(1,3)*M1(1,3)+dc3(q)*M3(1,3)*M2(1,3)+…
           dd3(q)*M3(1,3)*M3(1,3)+da4(q)*(M1(1,1)+M1(3,3))+…
           da5(q)*(M2(1,1)+M2(3,3))+da6(q)*(M3(1,1)+M3(3,3));
dDnew(5,6)=db1(q)*M1(1,3)*M1(2,3)+db2(q)*M1(1,3)*M2(2,3)+…
           dd1(q)*M1(1,3)*M3(2,3)+db2(q)*M2(1,3)*M1(2,3)+…
           dc2(q)*M2(1,3)*M2(2,3)+dc3(q)*M2(1,3)*M3(2,3)+…
           dd1(q)*M3(1,3)*M1(2,3)+dc3(q)*M3(1,3)*M2(2,3)+…
           dd3(q)*M3(1,3)*M3(2,3)+da4(q)*M1(1,2)+…
           da5(q)*M2(1,2)+da6(q)*M3(1,2);
dDnew(6,6)=db1(q)*M1(2,3)*M1(2,3)+db2(q)*M1(2,3)*M2(2,3)+…
           dd1(q)*M1(2,3)*M3(2,3)+db2(q)*M2(2,3)*M1(2,3)+…
           dc2(q)*M2(2,3)*M2(2,3)+dc3(q)*M2(2,3)*M3(2,3)+…
           dd1(q)*M3(2,3)*M1(2,3)+dc3(q)*M3(2,3)*M2(2,3)+…
           dd3(q)*M3(2,3)*M3(2,3)+da4(q)*(M1(2,2)+M1(3,3))+…
           da5(q)*(M2(2,2)+M2(3,3))+da6(q)*(M3(2,2)+M3(3,3));
dDnew(2,1)=dDnew(1,2);
dDnew(3,1)=dDnew(1,3);
dDnew(4,1)=dDnew(1,4);
dDnew(5,1)=dDnew(1,5);
dDnew(6,1)=dDnew(1,6);
dDnew(3,2)=dDnew(2,3);
dDnew(4,2)=dDnew(2,4);
dDnew(5,2)=dDnew(2,5);
dDnew(6,2)=dDnew(2,6);
dDnew(4,3)=dDnew(3,4);
dDnew(5,3)=dDnew(3,5);
dDnew(6,3)=dDnew(3,6);
dDnew(5,4)=dDnew(4,5);
dDnew(6,4)=dDnew(4,6);
dDnew(6,5)=dDnew(5,6);
```

```
function [Ke,fe]=plani4eaxi(ex,ey,ep,D,eq)
% Ke=plani4eaxi(ex,ey,ep,D)
% [Ke,fe]=plani4e(ex,ey,ep,D,eq)
%-------------------------------------------------------------
% PURPOSE
%  Calculate the stiffness matrix for a 4 node isoparametric
%  element in plane strain,plane stress or axisymmetry.
%
% INPUT:  ex = [x1 x2 x3 x4]  element coordinates
%         ey = [y1 y2 y3 y4]
%
%         ep =[ptype t ir]    element property
%                               ptype: analysis type
%                               ir: integration rule
%                               t : thickness
%
%         D                   constitutive matrix
%
```

```
%         eq = [bx; by]       bx: body force in x direction
%                             by: body force in y direction
%
% OUTPUT: Ke : element stiffness matrix (8 x 8)
%         fe : equivalent nodal forces (8 x 1)
%--------------------------------------------------------------

% LAST MODIFIED: M Ristinmaa  1995-10-25
% Copyright (c)  Division of Structural Mechanics and
%                Department of Solid Mechanics.
%                Lund Institute of Technology
%--------------------------------------------------------------
  ptype=ep(1); t=ep(2);  ir=ep(3);  ngp=ir*ir;
  if nargin==4   b=zeros(2,1);  else  b=eq; end

%--------- gauss points ---------------------------------------
  if ir==1
    g1=0.0; w1=2.0;
    gp=[ g1 g1 ];  w=[ w1 w1 ];
  elseif ir==2
    g1=0.577350269189626; w1=1;
    gp(:,1)=[-g1; g1;-g1; g1];  gp(:,2)=[-g1;-g1; g1; g1];
    w(:,1)=[ w1; w1; w1; w1];   w(:,2)=[ w1; w1; w1; w1];
  elseif ir==3
    g1=0.774596669241483; g2=0.;
    w1=0.555555555555555; w2=0.888888888888888;
    gp(:,1)=[-g1;-g2; g1;-g1; g2; g1;-g1; g2; g1];
    gp(:,2)=[-g1;-g1;-g1; g2; g2; g2; g1; g1; g1];
    w(:,1)=[ w1; w2; w1; w1; w2; w1; w1; w2; w1];
    w(:,2)=[ w1; w1; w1; w2; w2; w2; w1; w1; w1];
  else
    disp('Used number of integration points not implemented');
    return
  end
  wp=w(:,1).*w(:,2);
  xsi=gp(:,1);  eta=gp(:,2);  r2=ngp*2;

%--------- shape functions ------------------------------------
  N(:,1)=(1-xsi).*(1-eta)/4;  N(:,2)=(1+xsi).*(1-eta)/4;
  N(:,3)=(1+xsi).*(1+eta)/4;  N(:,4)=(1-xsi).*(1+eta)/4;

  dNr(1:2:r2,1)=-(1-eta)/4;     dNr(1:2:r2,2)= (1-eta)/4;
  dNr(1:2:r2,3)= (1+eta)/4;     dNr(1:2:r2,4)=-(1+eta)/4;
  dNr(2:2:r2+1,1)=-(1-xsi)/4;   dNr(2:2:r2+1,2)=-(1+xsi)/4;
  dNr(2:2:r2+1,3)= (1+xsi)/4;   dNr(2:2:r2+1,4)= (1-xsi)/4;


  Ke=zeros(8,8);
  fe=zeros(8,1);
  JT=dNr*[ex;ey]';
```

```
%--------- plane stress --------------------------------------
if ptype==1

    colD=size(D,2);
    if colD>3
        Cm=inv(D);
        Dm=inv(Cm([1 2 4],[1 2 4]));
    else
        Dm=D;
    end

    for i=1:ngp
        indx=[ 2*i-1; 2*i ];
        detJ=det(JT(indx,:));
        if detJ<10*eps
            disp('Jacobideterminant equal or less than zero!')
        end
        JTinv=inv(JT(indx,:));
        dNx=JTinv*dNr(indx,:);

        B(1,1:2:8-1)=dNx(1,:);
        B(2,2:2:8)  =dNx(2,:);
        B(3,1:2:8-1)=dNx(2,:);
        B(3,2:2:8)  =dNx(1,:);

        N2(1,1:2:8-1)=N(i,:);
        N2(2,2:2:8)  =N(i,:);

        Ke=Ke+B'*Dm*B*detJ*wp(i)*t;
        fe=fe+N2'*b*detJ*wp(i)*t;
  end
%--------- plane strain -----------------------------------
elseif ptype==2

    colD=size(D,2);
    if colD>3
        Dm=D([1 2 4],[1 2 4]);
    else
        Dm=D;
    end

    for i=1:ngp
        indx=[ 2*i-1; 2*i ];
        detJ=det(JT(indx,:));
        if detJ<10*eps
            disp('Jacobideterminant equal or less than zero!')
        end
        JTinv=inv(JT(indx,:));
        dNx=JTinv*dNr(indx,:);

        B(1,1:2:8-1)=dNx(1,:);
        B(2,2:2:8)  =dNx(2,:);
        B(3,1:2:8-1)=dNx(2,:);
        B(3,2:2:8)  =dNx(1,:);

        N2(1,1:2:8-1)=N(i,:);
        N2(2,2:2:8)  =N(i,:);

        Ke=Ke+B'*Dm*B*detJ*wp(i)*t;
        fe=fe+N2'*b*detJ*wp(i)*t;
    end
%--------------------Axisymmetry------------------------------
elseif ptype==3

    Dm=D([1:4],[1:4]);

     for i=1:ngp
         indx=[ 2*i-1; 2*i ];
         detJ=det(JT(indx,:));
         if detJ<10*eps
             disp('Jacobideterminant equal or less than zero!')
         end
         JTinv=inv(JT(indx,:));
         dNx=JTinv*dNr(indx,:);
         xbar1=N(i,:)*ex';

         B(1,1:2:8-1)=dNx(1,:);
         B(2,2:2:8)  =dNx(2,:);
         B(3,1:2:8-1)=N(i,:)/xbar1;
         B(4,1:2:8)=dNx(2,:);
         B(4,2:2:8)=dNx(1,:);

         Ke=Ke+B'*Dm*B*detJ*wp(i)*xbar1*2*pi;
     end

else
   error('Error ! Check first argument, ptype=1,2 or 3
allowed')
   return
end
%--------------------------end--------------------------------
```

```
function [es,et,eci]=plani4saxi(ex,ey,ep,D,ed)
% [es,et,eci]=plani4saxi(ex,ey,ep,D,ed)
%-------------------------------------------------------------
% PURPOSE
%  Calculate element normal and shear stress for a 4 node
%  isoparametric element in plane strain,plane stress or
%  axisymmetry
%
% INPUT:   ex = [x1 x2 x3 x4]  element coordinates
%          ey = [y1 y2 y3 y4]
%
%          ep = [ptype t ir ]  ptype: analysis type
%                              ir: integration rule
%                              t : thickness It: integration
rule
%
%          D                   constitutive matrix
%
%          ed = [u1 u2 ..u8;   element displacement vector
%               .........]     one row for each element
%
% OUTPUT: es = [ sigx sigy [sigz] tauxy    element stress
matrix
%                 ......              ]    one row for each
element
%         et = [ epsx epsy [epsz] gamxy    element strain
matrix
%                 ......              ]    one row for each
element
%-------------------------------------------------------------

% LAST MODIFIED: M Ristinmaa  1995-10-25
% Copyright (c)  Division of Structural Mechanics and
%                Department of Solid Mechanics.
%                Lund Institute of Technology
%-------------------------------------------------------------
  ptype=ep(1); t=ep(2);  ir=ep(3);  ngp=ir*ir;

%--------- gauss points --------------------------------------
if ir==1
    g1=0.0; w1=2.0;
    gp=[ g1 g1 ];  w=[ w1 w1 ];
elseif ir==2
    g1=0.577350269189626; w1=1;
    gp(:,1)=[-g1; g1;-g1; g1];  gp(:,2)=[-g1;-g1; g1; g1];
    w(:,1)=[ w1; w1; w1; w1];   w(:,2)=[ w1; w1; w1; w1];
elseif ir==3
    g1=0.774596669241483; g2=0.;
    w1=0.555555555555555; w2=0.888888888888888;
    gp(:,1)=[-g1;-g2; g1;-g1; g2; g1;-g1; g2; g1];
    gp(:,2)=[-g1;-g1;-g1; g2; g2; g2; g1; g1; g1];
    w(:,1)=[ w1; w2; w1; w1; w2; w1; w1; w2; w1];
    w(:,2)=[ w1; w1; w1; w2; w2; w2; w1; w1; w1];
else
    disp('Used number of integration points not implemented');
    return
end
wp=w(:,1).*w(:,2);
xsi=gp(:,1);  eta=gp(:,2);  r2=ngp*2;
```

```
%--------- shape functions -----------------------------------
  N(:,1)=(1-xsi).*(1-eta)/4;  N(:,2)=(1+xsi).*(1-eta)/4;
  N(:,3)=(1+xsi).*(1+eta)/4;  N(:,4)=(1-xsi).*(1+eta)/4;

  dNr(1:2:r2,1)=-(1-eta)/4;     dNr(1:2:r2,2)= (1-eta)/4;
  dNr(1:2:r2,3)= (1+eta)/4;     dNr(1:2:r2,4)=-(1+eta)/4;
  dNr(2:2:r2+1,1)=-(1-xsi)/4;   dNr(2:2:r2+1,2)=-(1+xsi)/4;
  dNr(2:2:r2+1,3)= (1+xsi)/4;   dNr(2:2:r2+1,4)= (1-xsi)/4;

%--------- plane stress --------------------------------------
if ptype==1

    rowed=size(ed,1);
    rowex=size(ex,1);
    colD =size(D ,2);

    if colD>3
        Cm=inv(D);
        Dm=inv(Cm([1 2 4],[1 2 4]));
    else
        Dm=D;
    end

    if rowex==11 incie=0; else incie=1; end

    es=[]; et=[]; eci=[]; ie=1;
    for ied=1:rowed
        eci=[eci N*[ex(ie,:);ey(ie,:)]'];
        JT=dNr*[ex(ie,:);ey(ie,:)]';

        for i=1:ngp
            indx=[ 2*i-1; 2*i ];
            detJ=det(JT(indx,:));
            if detJ<10*eps
                disp('Jacobideterminant equal or less…
                            than zero!')
            end
            JTinv=inv(JT(indx,:));
            dNx=JTinv*dNr(indx,:);

            B(1,1:2:8-1)=dNx(1,:);
            B(2,2:2:8)  =dNx(2,:);
            B(3,1:2:8-1)=dNx(2,:);
            B(3,2:2:8)  =dNx(1,:);

            ee=B*ed(ied,:)';
            if colD>3
                ss=zeros(colD,1);
                ss([1 2 4])=Dm*ee;
                ee=Cm*ss;
            else
```

```
                ss=Dm*ee;
            end

            et=[et; ee'];
            es=[es; ss'];
        end

        ie=ie+incie;
    end

%--------- plane strain --------------------------------------
elseif ptype==2

    rowed=size(ed,1);
    rowex=size(ex,1);
    colD =size(D ,2);

    if rowex==11 incie=0; else incie=1; end

    es=[]; et=[]; eci=[]; ie=1; ee=zeros(colD,1);
    for ied=1:rowed
        eci=[eci N*[ex(ie,:);ey(ie,:)]'];
        JT=dNr*[ex(ie,:);ey(ie,:)]';

        for i=1:ngp
            indx=[ 2*i-1; 2*i ];
            detJ=det(JT(indx,:));
            if detJ<10*eps
                disp('Jacobideterminant equal or less…
                            than zero!')
            end
            JTinv=inv(JT(indx,:));
            dNx=JTinv*dNr(indx,:);

            B(1,1:2:8-1)=dNx(1,:);
            B(2,2:2:8)  =dNx(2,:);
            B(3,1:2:8-1)=dNx(2,:);
            B(3,2:2:8)  =dNx(1,:);

            e=B*ed(ied,:)';
            if colD>3 ee([1 2 4])=e; else ee=e; end

            et=[et; ee'];
            es=[es; (D*ee)'];
        end

        ie=ie+incie;
    end

%-----------Axisymmetry----------------------------------------
elseif ptype==3
```

```
    rowed=size(ed,1);
    rowex=size(ex,1);
    colD =size(D ,2);

    if rowex==11 incie=0; else incie=1; end

    es=[]; et=[]; eci=[]; ie=1; ee=zeros(colD,1);
    Dm=D([1:4],[1:4]);
    for ied=1:rowed
        eci=[eci N*[ex(ie,:);ey(ie,:)]'];
        JT=dNr*[ex(ie,:);ey(ie,:)]';

        for i=1:ngp
            indx=[ 2*i-1; 2*i ];
            detJ=det(JT(indx,:));
            if detJ<10*eps
                disp('Jacobideterminant equal or less…
                           than zero!')
            end
            JTinv=inv(JT(indx,:));
            dNx=JTinv*dNr(indx,:);

            xbar1=N(i,:)*ex(ied,:)';

            B(1,1:2:8-1)=dNx(1,:);
            B(2,2:2:8)  =dNx(2,:);
            B(3,1:2:8-1)=N(i,:)/xbar1;
            B(4,1:2:8)=dNx(2,:);
            B(4,2:2:8)=dNx(1,:);

            e=B*ed(ied,:)';
            ee([1:4])=e;
            et=[et; ee'];
            es=[es; (D*ee)'];
        end
    end
%--------------------------------------------------------------


else
   error('Error ! Check first argument, ptype=1,2 or 3
allowed')
   return
end
%--------------------------end--------------------------------
```

```
function [ef]=plani4faxi(ex,ey,ep,es)
% ef=plani4faxi(ex,ey,ep,es)
%--------------------------------------------------------------
```

```
% PURPOSE
%  Calculate the element force vector corresponding to the
%  stresses in a 4 node isoparametric element.
%
% INPUT:  ex = [x1 x2 x3 x4]        element coordinates
%         ey = [y1 y2 y3 y4]
%
%         ep = [ptype t ir]         ptype: analysis type
%                                   ir: integration rule
%                                   t : thickness
%
%         es = [ sigx sigy [sigz] tauxy  element stress matrix
%                  ......              ]  one row for each
integration point
%
% OUTPUT: fe = [f1 f2 ...f8]';      internal force vector
%--------------------------------------------------------------

% LAST MODIFIED: M Ristinmaa  1995-10-25
% Copyright (c)  Division of Structural Mechanics and
%                Department of Solid Mechanics.
%                Lund Institute of Technology
%--------------------------------------------------------------
  ptype=ep(1); t=ep(2); ir=ep(3); ngp=ir*ir;

%--------- gauss points --------------------------------------
if ir==1
    g1=0.0; w1=2.0;
    gp=[ g1 g1 ];  w=[ w1 w1 ];
elseif ir==2
    g1=0.577350269189626; w1=1;
    gp(:,1)=[-g1; g1;-g1; g1];  gp(:,2)=[-g1;-g1; g1; g1];
    w(:,1)=[ w1; w1; w1; w1];   w(:,2)=[ w1; w1; w1; w1];
elseif ir==3
    g1=0.774596669241483; g2=0.;
    w1=0.555555555555555; w2=0.888888888888888;
    gp(:,1)=[-g1;-g2; g1;-g1; g2; g1;-g1; g2; g1];
    gp(:,2)=[-g1;-g1;-g1; g2; g2; g2; g1; g1; g1];
    w(:,1)=[ w1; w2; w1; w1; w2; w1; w1; w2; w1];
    w(:,2)=[ w1; w1; w1; w2; w2; w2; w1; w1; w1];
else
    disp('Used number of integration points not implemented');
    return
end
wp=w(:,1).*w(:,2);
xsi=gp(:,1);  eta=gp(:,2);  r2=ngp*2;

%--------- shape functions -----------------------------------
  N(:,1)=(1-xsi).*(1-eta)/4;  N(:,2)=(1+xsi).*(1-eta)/4;
  N(:,3)=(1+xsi).*(1+eta)/4;  N(:,4)=(1-xsi).*(1+eta)/4;
  dNr(1:2:r2,1)=-(1-eta)/4;     dNr(1:2:r2,2)= (1-eta)/4;
  dNr(1:2:r2,3)= (1+eta)/4;     dNr(1:2:r2,4)=-(1+eta)/4;
  dNr(2:2:r2+1,1)=-(1-xsi)/4;   dNr(2:2:r2+1,2)=-(1+xsi)/4;
  dNr(2:2:r2+1,3)= (1+xsi)/4;   dNr(2:2:r2+1,4)= (1-xsi)/4;

%--------- plane stress --------------------------------------
if ptype==1

    [rowes,colD]=size(es);
    rowex=size(ex,1);

    if rowex==1 incie=0; else incie=1; end

    ef=[]; ir=0; ie=1;
    for ied=1:rowes/ngp
        JT=dNr*[ex(ie,:);ey(ie,:)]';

        fint=zeros(8,1);
        for i=1:ngp
            ir=ir+1;
            indx=[ 2*i-1; 2*i ];
            detJ=det(JT(indx,:));
            if detJ<10*eps
                disp('Jacobideterminant equal or less…
                            than zero!')
            end
            JTinv=inv(JT(indx,:));
            dNx=JTinv*dNr(indx,:);

            B(1,1:2:8-1)=dNx(1,:);
            B(2,2:2:8)  =dNx(2,:);
            B(3,1:2:8-1)=dNx(2,:);
            B(3,2:2:8)  =dNx(1,:);

            if colD>3 stress=es(ir,[1 2 4]); else…
                stress=es(ir,:); end
            fint=fint+B'*stress'*wp(i)*detJ*t;
        end

        ef=[ef; fint'];
        ie=ie+incie;
    end

%--------- plane strain --------------------------------------
elseif ptype==2

    [rowes,colD]=size(es);
    rowex=size(ex,1);

    if rowex==1 incie=0; else incie=1; end
    ef=[]; ir=0; ie=1;
    res=size(es,1);
    for ied=1:rowes/ngp
        JT=dNr*[ex(ie,:);ey(ie,:)]';

        fint=zeros(8,1);
        for i=1:ngp
            ir=ir+1;
            indx=[ 2*i-1; 2*i ];
            detJ=det(JT(indx,:));
            if detJ<10*eps
                disp('Jacobideterminant equal or less…
                            than zero!')
            end
            JTinv=inv(JT(indx,:));
            dNx=JTinv*dNr(indx,:);

            B(1,1:2:8-1)=dNx(1,:);
            B(2,2:2:8)  =dNx(2,:);
            B(3,1:2:8-1)=dNx(2,:);
            B(3,2:2:8)  =dNx(1,:);

            if colD>3 stress=es(ir,[1 2 4]); else…
                stress=es(ir,:); end
            fint=fint+B'*stress'*wp(i)*detJ*t;
        end

        ef=[ef; fint'];
        ie=ie+incie;
    end

%-----------Axisymmetry---------------------------------------
elseif ptype==3
    [rowes,colD]=size(es);
    rowex=size(ex,1);

    if rowex==1 incie=0; else incie=1; end

    ef=[]; ir=0; ie=1;
    res=size(es,1);
    for ied=1:rowes/ngp
        JT=dNr*[ex(ie,:);ey(ie,:)]';

        fint=zeros(8,1);
        for i=1:ngp
            ir=ir+1;
            indx=[ 2*i-1; 2*i ];
            detJ=det(JT(indx,:));
            if detJ<10*eps
                disp('Jacobideterminant equal or less…
                            than zero!')
            end
            JTinv=inv(JT(indx,:));
            dNx=JTinv*dNr(indx,:);

            xbar1=N(i,:)*ex(ied,:)';

            B(1,1:2:8-1)=dNx(1,:);
            B(2,2:2:8)  =dNx(2,:);
            B(3,1:2:8-1)=N(i,:)/xbar1;
            B(4,1:2:8)=dNx(2,:);
            B(4,2:2:8)=dNx(1,:);
            if colD>3 stress=es(ir,[1:4]); else…
                   stress=es(ir,:); end
            fint=fint+B'*stress'*wp(i)*detJ*xbar1*2*pi;
        end

        ef=[ef; fint'];
        ie=ie+incie;
    end
%--------------------------------------------------------------

else
   error('Error ! Check first argument, ptype=1,2 or 3
allowed')
   return
end
%--------------------------end--------------------------------
```