Numerical integration of a rate type constitutive equation for incompressible isotropic elastic Neo-Hookean material behaviour

WFW 89.010

Numerical Integration of a Rate Type Constitutive Equation for Incompressible Isotropic Elastic Neo–Hookean Material Behaviour

Peter van Hoogstraten

August 1989

Numerical Integration of a Rate Constitutive Equation for Incompressible Isotropic Elastic Neo–Hookean Material Behaviour

Abstract

The Neo–Hookean equation is a well–known constitutive equation for isotropic elastic incompressible material behaviour. In this report a rate form of the Neo–Hookean equation is derived by introducing the so–called Truesdell rate. An integration algorithm is proposed for integrating the rate type equation. Also an alternative calculation method is presented.

Purpose of these algorithms is the implementation of Neo–Hookean material behaviour in the Finite Element package SEPRAN: It can be shown that there is a strong analogy between the proposed forms of the Neo–Hookean equation and the constitutive equation for an incompressible Newtonian flow, which is available in SEPRAN. In future the Neo–Hookean equation will be applied to the modelling of the rubber coating on pressure rollers in capstan drives of magnetic recorders.

The results of the algorithms strongly depend on the value of a certain interpolation parameter. By choosing a correct value, the algorithms will be 'incrementally objective', i.e. rigid body rotations will not cause any stresses in the incremental formulation, and more accurate.

Several applications of the algorithms are presented, showing that they are reliable and robust.

Keywords: *Rubbers, Neo–Hookean material, constitutive equations, numerical integration*

Table of Contents

Appendices

Appendix A	:	*Evaluation of Tensors* $(\boldsymbol{F}_{n+1} \cdot \boldsymbol{F}^{-1}_{n+\alpha})$ *and* $\boldsymbol{D}_{n+\alpha}$
Appendix B	:	*Evaluation of Tensor* $\delta\boldsymbol{H}_{n+\alpha}$ *for Small Deformations*
Appendix C	:	*Notes on the Objectivity of the integration algorithm*
Appendix D	:	*PC–Matlab Program File for Calculations on the Objectivity of the Integration Algorithm*
Appendix E	:	*PC–Matlab Program Files of Applications of the Integration Algorithm*
Appendix F	:	*PC–Matlab Program Files of Applications of the Alternative Algorithm*

Literature

1. Durieu,J. and Petit,M.
A 2–d Finite Element Solution of the Steady Rolling Contact Problem
in the Capstan/Tape/Roller Mechanism of Magnetic Recorders
Technical Note N146 Philips Research Laboratory Brussels

2. van Hoogstraten,P.A.A.
Calculations on a Capstan Pressure–Roller Mechanism Using the
Computer Program CTR
Nat.Lab. Technical Note 118/88

3. MARC Manuals, volume A,B,C revision K.3
MARC Analysis Research Corporation

4. Schreurs,P.J.G.
Numerical Simulation of Forming Processes
(PhD Thesis)

5. Pinsky,P.M., Ortiz,M. and Pister,K.S.
Numerical Integration of Rate Constitutive Equations
in Finite Deformations Analysis
Comp.Meth.Appl.Mech.Eng. 40 (1983) 137–158

6. PC–Matlab User's Guide
The MathWorks Inc.

CHAPTER 1: INTRODUCTION

Tape transport is one of the main aspects of audio and video recording. All consumer recorders and most of the professional recorders are equiped with a capstan pressure roller drive to perform this function. In figure 1.1 such a mechanism is represented.

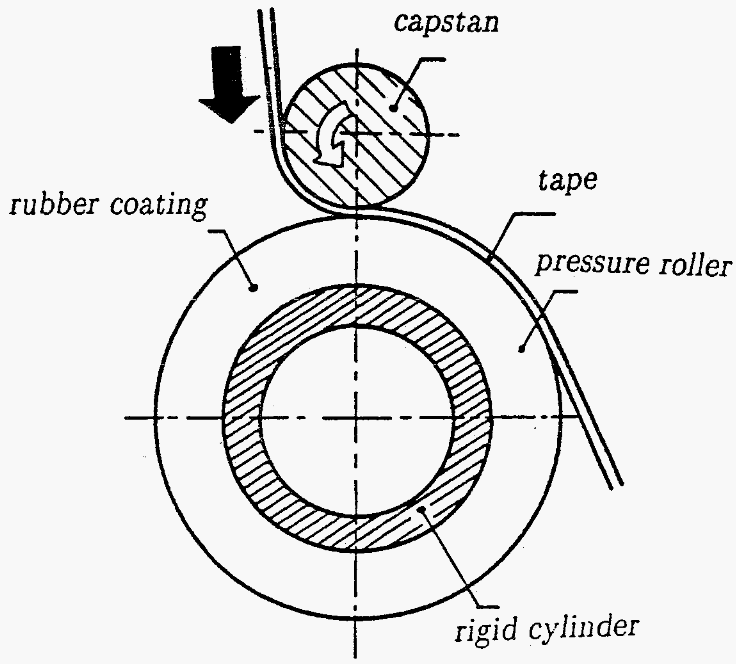


fig.1.1: the capstan pressure roller drive

A rigid metal cylinder covered with a rubber coating is pressed against a rigid drive shaft (capstan) by an elastic spring. Due to the frictional forces in the contact zones the tape is transported.

Several phenomena occuring in the capstan drive still do not have a reliable explanation. One of these incomprehended phenomena is the self–adjusting property of the pressure roller.

To clarify the behaviour of the capstan drive a research project has been started with the aim to create a three–dimensional numerical model of the tape transport mechanism and to verify its results with experiments.

In the recent past a two–dimensional finite element model of the capstan drive has been implemented in the computer program CTR [1]. Using CTR an analysis has been carried out to investigate the sensitivity of the results for variations in the model parameters [2]. One of the main conclusions drawn from this analysis was the observation of a remarkable and physically inexplicable sensitivity for the value of Poisson's ratio of the rubber coating; In CTR this coating is modelled as a linear elastic isotropic medium whose material parameters are Young's modulus and Poisson's ratio. Rubber is incompressible or nearly incompressible hence its Poisson's ratio is close to a half. It is well known that values of Poisson's ratio near a half may cause problems of numerical nature often referred to as locking phenomena. These phenomena form a sound explanation for the observed sensitivity.

In our three–dimensional model of the capstan drive we wish to avoid this sensitivity. That is why we will use a constitutive equation for the rubber material which includes its incompressibility. In chapter 2 of this report a general equation for incompressible isotropic elastic material behaviour is presented. The simplest constitutive equation which can be deduced from this general form is the Neo–Hookean constitutive equation. This equation contains only one material parameter. The Mooney–Rivlin equation contains two parameters and higher order equations even more. Comparable test calculations performed using the MARC finite element package [3] on the frictionless indentation of an elastic layer by a rigid cylinder showed no differences in the resulting stresses for the Neo–Hookean equation and the results for higher order equations such as the Mooney–Rivlin equation. This indicates that application of the Neo–Hookean constitutive equation is admissible.

In general three dimensional finite element models consume large amounts of computing time. Modelling the capstan drive, computing time can be saved when applying an Eulerian approach instead of an Lagrangian approach: Only a small part of the pressure roller contains stresses and deformations unequal to zero. By using an Eulerian approach only the interesting part of the pressure roller has to be taken into account while in a Lagrangian approach the total pressure roller has to be modelled. Using a Lagrangian approach, mesh refinements are ineffectual. In an Eulerian approach mesh refinement can decrease the necessary computing time without loss of numerical efficiency.

In an Eulerian approach the Neo–Hookean constitutive equation has to be adapted. The subject of chapter 3 of this report is the derivation and the numerical integration of a rate constitutive equation for the Neo–Hookean material behaviour. The proposed algorithm calculates stresses when an estimation of the deformation tensor is given. The objectivity of the obtained integration algorithm is discussed. In chapter 4 several applications of the algorithm will be presented.

In chapter 5 an alternative algorithm for computing the stresses is given. Here the calculation method is based on an estimation of the velocity gradient. This algorithm is illustrated by several examples in chapter 6.

CHAPTER 2: THE NEO–HOOKEAN CONSTITUTIVE EQUATION

2.1 Kinematics

In this paragraph, we will look at some basic definitions regarding the kinematics of a deforming body. In the Euclidean space a three dimensional body B will be considered. Each material point P of this body can be identified by a column $\underset{\sim}{\xi}$ of three material coordinates. This column may be considered as a label attached to point P.

Fixed in the Euclidean space an origin O is chosen. The current position vector of a point, measured with respect to this origin, is denoted by $\vec{x}(\underset{\sim}{\xi},t)$, being a function of label $\underset{\sim}{\xi}$ of the point considered and of time t. We will assume that $\vec{x}(\underset{\sim}{\xi},t)$ is continuous and differentiable with respect to both $\underset{\sim}{\xi}$ and t. The complete set of position vectors of all material points of body B at time t is called the configuration of B at time t, denoted by G(t):

$$G(t) = \{ \vec{x} = \vec{X}(\underset{\sim}{\xi},t) \mid \underset{\sim}{\xi} \in B \} \tag{2.1.1}$$

The deformation of a continuum will be described with respect to its configuration at a chosen time t_0. This configuration $G(t_0)$ is called reference configuration.

Let $d\vec{x}_0$ be the distance vector between two neighboring material points P and Q of body B in the reference configuration $G(t_0)$. Vector $d\vec{x}$ is the distance vector between these two material points in the current configuration G(t).

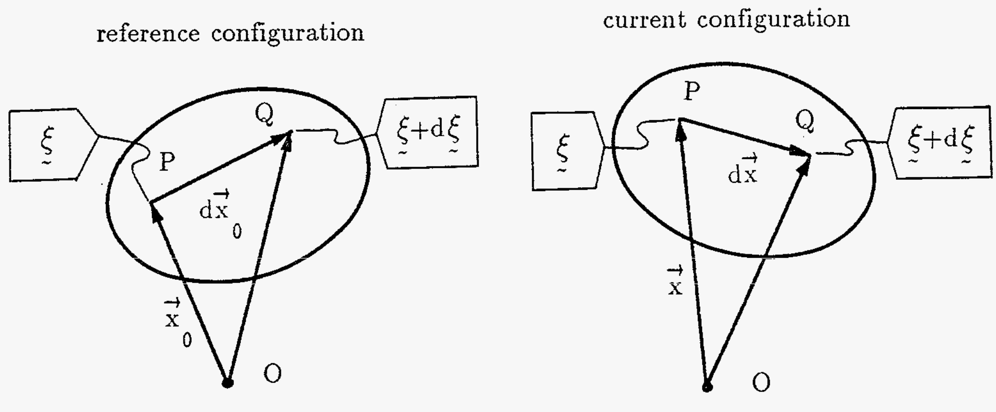


fig.2.1: body B in reference and current configuration

The deformation tensor **F** characterizes the deformation of body B in the current configuration at a material point with label $\underset{\sim}{\xi}$ with respect to the reference configuration. Tensor **F** is defined by:

$$\mathbf{F} = (\vec{\nabla}_0 \vec{x})^c \quad ; \quad d\vec{x} = \mathbf{F} \cdot d\vec{x}_0 \tag{2.1.2}$$

Here $\vec{\nabla}_0$ is the gradient operator with respect to the reference configuration. This operator is related to the gradient operator $\vec{\nabla}$ with respect to the current current configuration by:

$$\vec{\nabla} = \mathbf{F}^{-c} \cdot \vec{\nabla}_0 \tag{2.1.3}$$

Regarding an infinitesimal small part of B in the surroundings of the material point with label $\underset{\sim}{\xi}$ it can be shown that the determinant of deformation tensor **F** is equal to the ratio of the current volume of this part to its reference volume. This can be proven using the formal definition of the determinant of a tensor.

$$J = \det(\mathbf{F}) = \frac{dV}{dV_0} = \text{volume change factor} \tag{2.1.4}$$

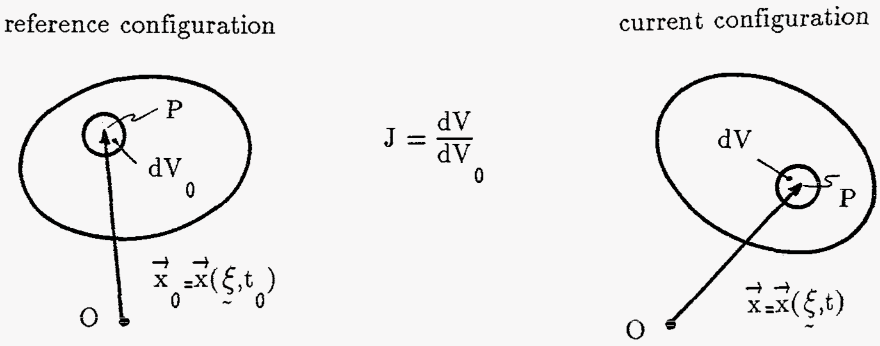


fig.2.2: volume change in point P

For isochoric deformations the volume change factor will always be equal to 1, as for incompressible media. While physically a volume change factor less than or equal to zero is impossible, the determinant of **F** will always be greater than zero. Hence tensor **F** is regular.

The deformation tensor **F** can be decomposed into an orthogonal rotation tensor **R** and a positive definite symmetric stretch tensor **U**:

$$\mathbf{F} = \mathbf{R}\cdot\mathbf{U} \begin{cases} \mathbf{R}\cdot\mathbf{R}^c = \mathbf{I} \quad ; \det(\mathbf{R})=1 \\ \mathbf{U} = \mathbf{U}^c \end{cases} \tag{2.1.5}$$

Tensor **R** describes the rigid–body rotation of the body while **U** describes the real deformation. This decomposition is called the right polar decomposition of **F**.

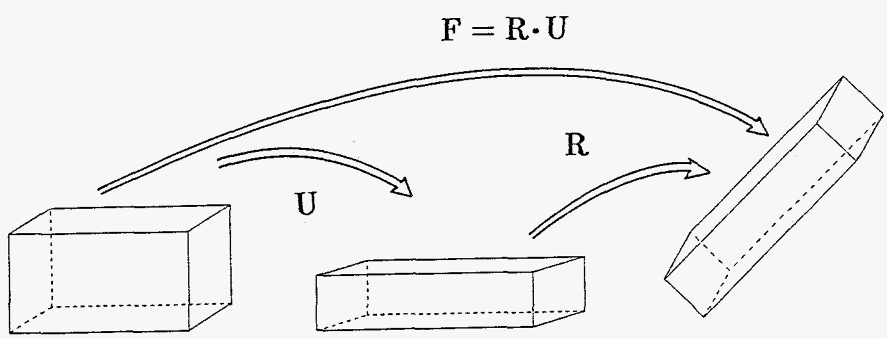


fig.2.3: right polar decomposition of F

Based on the deformation tensor several strain tensors can be defined. For geometrical nonlinear deformations the Green–Lagrange strain tensor is often used:

$$\mathbf{E} = \frac{1}{2}\left(\mathbf{F}^c \cdot \mathbf{F} - \mathbf{I} \right) \tag{2.1.6}$$

In the sequel we will frequently use the right and left Cauchy–Green strain tensors, which are defined in the following manner:

$$\mathbf{C} = \mathbf{F}^c \cdot \mathbf{F} = \mathbf{U}^2 \qquad \text{: right Cauchy–Green tensor} \tag{2.1.7}$$

$$\mathbf{B} = \mathbf{F} \cdot \mathbf{F}^c = \mathbf{R} \cdot \mathbf{U}^2 \cdot \mathbf{R}^c \qquad \text{: left Cauchy–Green tensor} \tag{2.1.8}$$

Let $\vec{a}(\underset{\sim}{\xi},t)$ be a vector field specifying some property of body B. The material derivative of $\vec{a}$ in a point with label $\underset{\sim}{\xi}$ is equal to the rate of change of $\vec{a}(\underset{\sim}{\xi},t)$ with respect to time, and is given by:

$$\dot{\vec{a}}(\underset{\sim}{\xi},t) = \lim_{\Delta t \to 0} \frac{\vec{a}(\underset{\sim}{\xi},t+\Delta t) - \vec{a}(\underset{\sim}{\xi},t)}{\Delta t} \tag{2.1.9}$$

The velocity $\vec{v}(\underset{\sim}{\xi},t)$ of a material point is defined as the material derivative of position vector $\vec{x}(\underset{\sim}{\xi},t)$ of this material point:

$$\vec{v}(\underset{\sim}{\xi},t) = \dot{\vec{x}}(\underset{\sim}{\xi},t) = \lim_{\Delta t \to 0} \frac{\vec{x}(\underset{\sim}{\xi},t+\Delta t) - \vec{x}(\underset{\sim}{\xi},t)}{\Delta t} \tag{2.1.10}$$

Using definition (2.1.2) the following equation can be deduced:

$$\dot{d\vec{x}} = \dot{\mathbf{F}} \cdot \mathbf{F}^{-1} \cdot d\vec{x} = \mathbf{L} \cdot d\vec{x} \tag{2.1.11}$$

Tensor **L** is equal to the velocity gradient $(\vec{\nabla}\vec{v})^c$ where $\vec{\nabla}$ denotes the gradient with respect to the current configuration. The symmetric part of **L** is the rate of strain tensor (**D**) while the skew–symmetric part of **L** is called the spin tensor or rate of rotation tensor (**Ω**).

$$\mathbf{D} = \frac{1}{2}\,(\mathbf{L} + \mathbf{L}^c) = \frac{1}{2}\,[\ (\vec{\nabla}\vec{v})^c + \vec{\nabla}\vec{v}\ ] \qquad ;\ \mathbf{D} = \mathbf{D}^c \tag{2.1.12}$$

$$\mathbf{\Omega} = \frac{1}{2}\,(\mathbf{L} - \mathbf{L}^c) = \frac{1}{2}\,[\ (\vec{\nabla}\vec{v})^c - \vec{\nabla}\vec{v}\ ] \qquad ;\ \mathbf{\Omega} = -\mathbf{\Omega}^c \tag{2.1.13}$$

2.2 Balance laws

An isothermal deformation process in a non–polar continuum has to satisfy three local balance laws:

(1)	Balance of mass:	$J\rho = \rho_0$	(2.2.1)
(2)	Balance of momentum:	$\vec{\nabla}\cdot\sigma^c + \rho\vec{b} = \rho\dot{\vec{v}}$	(2.2.2)
(3)	Balance of moment of momentum	$\sigma^c = \sigma$	(2.2.3)

where J is the volume change factor, ρ and ρ_0 are the specific masses in respectively the current and the reference configuration, $\vec{\nabla}$ is the gradient operator with respect to the current configuration, σ is the Cauchy stress tensor, $\vec{b}$ is the specific load vector and $\dot{\vec{v}}$ is the material derivative of the velocity vector, called the acceleration vector.

These three local balance laws result in a set of seven equations. The law of balance of mass leads to a single equation, while the other two laws both lead to three independent equations. The number of unknown variables is equal to thirteen; density ρ is a scalar, position vector $\vec{x}$ is a vector and the Cauchy stress tensor is a second order tensor with nine components. When the law of moment of momentum is taken into account, the number of equations reduces to four while the number of unknowns equals ten. To create a solvable set of equations six extra equations have to be formulated. These equations are called constitutive relations and will be the subject of the next paragraph.

2.3 A constitutive equation for homogeneous incompressible isotropic elastic material

In the previous paragraph it was stated that six constitutive relations have to be formulated for a complete description of an isothermal deformation process. The unknowns to be solved are the density $\rho(\underset{\sim}{\xi},t)$, the position vector $\vec{x}(\underset{\sim}{\xi},t)$ and the symmetric Cauchy stress tensor $\boldsymbol{\sigma}(\underset{\sim}{\xi},t)$.

Based on general constitutive principles it can be shown that for an elastic medium the Cauchy stress tensor may be interpreted as a function of the deformation tensor:

$$\boldsymbol{\sigma}(\underset{\sim}{\xi},t) = \mathbf{N}\{\ \mathbf{F}(\underset{\sim}{\xi},t),\} \tag{2.3.1}$$

Due to the symmetry of tensor $\boldsymbol{\sigma}$ relation (2.3.1) effectively consists of only six equations.

It is evident that the stress state in a continuum may not change when subjecting the continuum to a rigid–body translation or rotation. This fact is a manner of formulating the so–called principle of objectivity. This principle must hold for all constitutive equations. When using equation (2.3.1) a restriction must be formulated to obey this principle. Tensor function $\mathbf{N}$ must satisfy the following equation

$$\mathbf{N}\{\mathbf{Q}\cdot\mathbf{F}\} = \mathbf{Q}\cdot\mathbf{N}\{\mathbf{F}\}\cdot\mathbf{Q}^{c} \tag{2.3.2}$$

where tensor $\mathbf{Q}$ is an arbitrary rigid–body rotation tensor.

For isotropic elastic material behaviour the following constitutive equation can be derived based on the internal energy in the continuum.

$$\sigma = 2\rho \left[\frac{\partial \phi}{\partial J_3} J_3 I + \left[\frac{\partial \phi}{\partial J_1} + \frac{\partial \phi}{\partial J_2} J_1 \right] \mathbf{B} - \left[\frac{\partial \phi}{\partial J_2} \right] \mathbf{B}^2 \right] \tag{2.3.3}$$

Tensor **B** is the earlier mentioned left Cauchy–Green strain tensor. The scalar ϕ is the specific internal energy in the continuum. For isotropic materials it can be proven that ϕ may only depend on the three invariants of the left Cauchy–Green strain tensor:

$$\phi = \phi\ (J_1, J_2, J_3) \tag{2.3.4}$$

$$\text{with} \quad J_1 = \mathrm{tr}(\mathbf{B})$$

$$J_2 = \tfrac{1}{2}\left[\ (\mathrm{tr}\ \mathbf{B})^2 - \mathrm{tr}(\mathbf{B}^2)\ \right]$$

$$J_3 = \det(\mathbf{B})$$

It can easily be shown that equation (2.3.3) obeys the principle of objectivity.

The Cauchy stress tensor can be devided into a deviatoric part σ^d and a hydrostatic part σ^h. These tensors are related to respectively the change of form and the change of volume of the deforming body. When assuming the material behaviour to be incompressible volume change is absent and an arbitrary hydrostatic pressure can be applied to the body without causing any deformation. So a part of the Cauchy stress tensor will be determined not by the deformation but by the boundary conditions. The internal energy in the body is no longer dependent on the third invariant of **B** :

$$J_3 = \det(\mathbf{B}) = \det^2(\mathbf{F}) = J^2 = 1 \tag{2.3.5}$$

$$\phi = \phi(J_1, J_2) \tag{2.3.6}$$

The assumption of incompressibility results in the following equations for the hydrostatic and deviatoric part of $\boldsymbol{\sigma}$.

$$\boldsymbol{\sigma} = \boldsymbol{\sigma}^h + \boldsymbol{\sigma}^d \quad \begin{cases} \boldsymbol{\sigma}^h = -p\mathbf{I} \\ \boldsymbol{\sigma}^d = \boldsymbol{\sigma} - \frac{1}{3}(\mathrm{tr}\ \boldsymbol{\sigma})\mathbf{I} \end{cases} \tag{2.3.7}$$

$$\boldsymbol{\sigma}^d = 2\rho \left[\left[\frac{\partial\phi}{\partial J_1} + \frac{\partial\phi}{\partial J_2} J_1\right]\left[\mathbf{B} - \frac{1}{3}(\mathrm{tr}\ \mathbf{B})\ \mathbf{I}\right] - \left[\frac{\partial\phi}{\partial J_2}\right]\left[\mathbf{B}^2 - \frac{1}{3}\mathrm{tr}(\mathbf{B}^2)\ \mathbf{I}\right] \right]$$

So for incompressible isotropic elastic behaviour the subsequent constitutive equation can be formulated using the definition of the invariants J_1 and J_2.

$$\boldsymbol{\sigma} = -p\mathbf{I} + 2\rho\left[\left[\frac{\partial\phi}{\partial J_1} + \frac{\partial\phi}{\partial J_2} J_1\right]\mathbf{B} - \left[\frac{\partial\phi}{\partial J_2}\right]\mathbf{B}^2 - \left[\frac{1}{3}\frac{\partial\phi}{\partial J_1} J_1 + \frac{2}{3}\frac{\partial\phi}{\partial J_2} J_2\right]\mathbf{I} \right] \tag{2.3.8}$$

Scalar p is the hydrostatic pressure on the body under consideration. This pressure is determined by the boundary conditions applied to the body. Using an equation of the form (2.3.8) the incompressibility constraint (2.3.5) has to be satisfied.

In the past several functions $\phi(J_1,J_2)$ have been proposed. The most common functions can be deduced from the next class of polynomials:

$$\phi = \phi(J_1,J_2) = \sum_{i=0}^{m}\sum_{j=0}^{n} C_{ij}(J_1-3)^i(J_2-3)^j \ ; \ C_{00} = 0 \tag{2.3.9}$$

For all members of this class it holds that, when no deformation occurs, the internal energy will be equal to zero.

The Neo–Hookean constitutive equation is the simplest example of (2.3.9):

$$\phi = C_{10}\,(J_1 - 3) \tag{2.3.10}$$

The resulting constitutive equation contains only one material constant:

$$\sigma = -p\mathbf{I} + 2C\left\{\mathbf{B} - \frac{1}{3}\,\mathrm{tr}(\mathbf{B})\mathbf{I}\right\} \tag{2.3.11}$$

where C is a material parameter equal to the product of density ρ and parameter C_{10}.

CHAPTER 3: THE NEO–HOOKEAN CONSTITUTIVE EQUATION IN AN EULERIAN APPROACH: NUMERICAL INTEGRATION OF THE NEO–HOOKEAN RATE CONSTITUTIVE EQUATION

3.1 The Neo–Hookean rate constitutive equation

In the previous chapter the Neo–Hookean constitutive equation has been derived:

$$\sigma = -p\mathbf{I} + 2C\left\{\mathbf{B} - \frac{1}{3}\,\mathrm{tr}(\mathbf{B})\mathbf{I}\right\} \tag{3.1.1}$$

with:

σ : the Cauchy stress tensor

p : the hydrostatic pressure

C : a material constant

$\mathbf{B}$: the left Cauchy–Green strain tensor

An incompressibility constraint must be satisfied when using this equation:

$$\text{incompressibility: } J = \det(\mathbf{F}) = \frac{dV}{dV_0} = 1 \tag{3.1.2}$$

In this paragraph a rate form of the Neo–Hookean constitutive equation will be presented, which will show to be very usefull when implementing the capstan drive model in the F.E.M. package SEPRAN.

The first step to generate the rate form is the rearranging of equation (3.1.2)

$$\boldsymbol{\sigma} = -\left[\mathrm{p} + 2\mathrm{C}\left\{ \tfrac{1}{3}\,\mathrm{tr}(\mathbf{B}) - 1 \right\} \right] \mathbf{I} + 2\mathrm{C}\left[\mathbf{B} - \mathbf{I} \right] \tag{3.1.3}$$

$$= -\ \mathrm{p}^* \,\mathbf{I} + \boldsymbol{\tau}$$

with:

$$\mathrm{p}^* = \mathrm{p} + 2\mathrm{C}\left\{ \tfrac{1}{3}\,\mathrm{tr}(\mathbf{B}) - 1 \right\}$$

$$\boldsymbol{\tau} = 2\mathrm{C}\left[\mathbf{B} - \mathbf{I} \right]$$

Again p* can be interpreted as a (pseudo) hydrostatic pressure, partially determined by the applied boundary conditions.

Next a time derivative of $\boldsymbol{\tau}$ will be introduced. Because a constitutive equation has to satisfy the principle of objectivity this time derivative must be objective. It can easily be shown that the material derivative of $\boldsymbol{\tau}$ is not objective. In literature several proposals for objective time derivatives can be found. In our case the Truesdell time derivative is extremely usefull. Its definition is:

$$\overset{\nabla}{\mathbf{A}}_{\mathrm{tr}} = \dot{\mathbf{A}} - (\,\boldsymbol{\Omega}+\mathbf{D}\,)\cdot\mathbf{A} - \mathbf{A}\cdot(\,\boldsymbol{\Omega}+\mathbf{D}\,)^{\mathrm{c}} \tag{3.1.4}$$

$\mathbf{A}$ is an arbitrary tensor

$\dot{\mathbf{A}}$ is the material derivative of $\mathbf{A}$

$\boldsymbol{\Omega}$ is the rate of rotation tensor

$\mathbf{D}$ is the rate of deformation tensor

Using this definition it can be shown that the Truesdell rate of the left Cauchy–Green strain tensor $\mathbf{B}$ equals the null tensor.

It can also be shown that the Truesdell rate of the unity tensor equals minus two times the rate of deformation tensor:

$$\overset{\nabla}{\mathbf{B}}_{tr} = \mathbf{O} \tag{3.1.5}$$

$$\overset{\nabla}{\mathbf{I}}_{tr} = -2\,\mathbf{D} \tag{3.1.6}$$

So for the Truesdell rate of tensor $\boldsymbol{\tau}$ the relation (3.1.7) holds:

$$\overset{\nabla}{\boldsymbol{\tau}}_{tr} = 2C \left[\overset{\nabla}{\mathbf{B}}_{tr} - \overset{\nabla}{\mathbf{I}}_{tr} \right] = 4C\,\mathbf{D} \tag{3.1.7}$$

Summarizing, for the Neo–Hookean constitutive equation the following objective rate form holds:

$$\boldsymbol{\sigma} = -\,p^*\,\mathbf{I} + \boldsymbol{\tau}$$

$$\overset{\nabla}{\boldsymbol{\tau}}_{tr} = \dot{\boldsymbol{\tau}} - (\,\boldsymbol{\Omega}+\mathbf{D}\,)\cdot\boldsymbol{\tau} - \boldsymbol{\tau}\cdot(\,\boldsymbol{\Omega}+\mathbf{D}\,)^c = 4C\,\mathbf{D} \tag{3.1.8}$$

The incompressibility constraint (3.1.2) that also has to be satisfied can be rewritten as:

$$J = \det(\mathbf{F}) = 1 \quad \Longrightarrow \quad \frac{\dot{J}}{J} = \mathrm{tr}(\mathbf{D}) = \vec{\nabla}\cdot\vec{v} = 0 \tag{3.1.9}$$

The latter equation is also known as the continuity equation.

3.2 Numerical integration of the Neo–Hookean rate constitutive equation

Using the rate constitutive equation as proposed in the previous paragraph, an integration procedure for the Truesdell–derivative of tensor τ will have to be formulated. In this paragraph such a numerical integration procedure is explained. This procedure is based on the work of Pinsky et al.[5] and uses an estimation of the deformation tensor.

Introducing tensor **T** as in formula (3.2.1) a relation between the material derivative of **T** and the Truesdell derivative of τ can be deduced:

$$\mathbf{T} = \mathbf{F}^{-1} \cdot \tau \cdot \mathbf{F}^{-c} \tag{3.2.1}$$

$$\Rightarrow \quad \dot{\mathbf{T}} = \mathbf{F}^{-1} \cdot \overset{\nabla}{\tau}_{tr} \cdot \mathbf{F}^{-c} \quad ; \quad \overset{\nabla}{\tau}_{tr} = 4\mathrm{C}\,\mathbf{D} \tag{3.2.2}$$

Tensor **T** will be discretized at a discrete number of points in time (1,2,..,n,n+1,..,N). The time interval between two subsequent configurations n and n+1 is denoted as h_n.

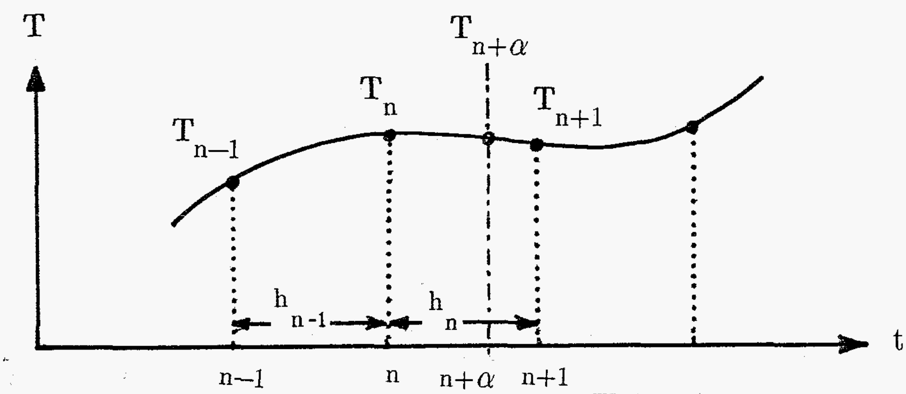


*fig.3.1: schematic representation of time discretization of tensor **T***

Assume tensor **T** to be known at time n. $\dot{\mathbf{T}}_{n+\alpha}$ denotes an estimation of the material derivative of **T** at a certain time in the interval (n,n+1). The value of parameter α is in between the limits 0 and 1. Tensor **T** at time n+1 now can be approximated by:

$$\mathbf{T}_{n+1} \simeq \mathbf{T}_n + h_n\, \dot{\mathbf{T}}_{n+\alpha} \tag{3.2.3}$$

In this paragraph an updated approach will be used. So the known configuration at time n will be regarded as a reference configuration. The substitution of formula (3.2.2) and definition (3.2.1) in equation (3.2.3) results in the following expression for τ_{n+1}:

$$\tau_{n+1} \simeq \mathbf{F}_{n+1} \cdot \tau_n \cdot \mathbf{F}^c_{n+1} + 4C\, h_n\, \mathbf{F}_{n+1} \cdot \mathbf{F}^{-1}_{n+\alpha} \cdot \mathbf{D}_{n+\alpha} \cdot \mathbf{F}^{-c}_{n+\alpha} \cdot \mathbf{F}^c_{n+1} \tag{3.2.4}$$

Applying an updated approach the deformation tensor in configuration n equals the unity tensor. In the configurations at time (n+α) and (n+1) the deformation tensors are equal to the gradient with respect to the reference configuration of the current displacement field.

$$\begin{array}{lll} \text{configuration n:} & \mathbf{F}_n & = (\vec{\nabla}_n \vec{x}_n)^c = \mathbf{I} \\ \text{configuration n+}\alpha\text{:} & \mathbf{F}_{n+\alpha} & = (\vec{\nabla}_n \vec{x}_{n+\alpha})^c \\ \text{configuration n+1:} & \mathbf{F}_{n+1} & = (\vec{\nabla}_n \vec{x}_{n+1})^c \end{array} \tag{3.2.5}$$

An essential aspect of the numerical integration algorithm is the linear interpolation of the displacement field:

$$\vec{x}(\xi,n+\alpha) = (1-\alpha)\, \vec{x}(\xi,n) + \alpha\, \vec{x}(\xi,n+1) \tag{3.2.6}$$

So for the approximation of $\mathbf{F}_{n+\alpha}$ the following equation holds:

$$\mathbf{F}_{n+\alpha} = (\vec{\nabla}_n \vec{x}_{n+\alpha})^c = \left[\vec{\nabla}_n \left[(1-\alpha)\, \vec{x}_n + \alpha\, \vec{x}_{n+1} \right] \right]^c$$

$$= (1-\alpha)\, \mathbf{I} + \alpha\, \mathbf{F}_{n+1} \tag{3.2.7}$$

By choosing a linear interpolation of the displacement field, it is possible to formulate an approximation for tensor τ_{n+1} as a function of tensor $\mathbf{F}_{n+1}$ and interpolation parameter α.

In appendix A the validity of approximations for tensors $(\mathbf{F}_{n+1} \cdot \mathbf{F}^{-1}_{n+\alpha})$ and $\mathbf{D}_{n+\alpha}$ is proved using equations (3.2.5) and (3.2.7). It is shown that:

$$\mathbf{F}_{n+1} \cdot \mathbf{F}^{-1}_{n+\alpha} = \left[(1-\alpha)\, \mathbf{F}^{-1}_{n+1} + \alpha\, \mathbf{I} \right]^{-1} \tag{3.2.8}$$

$$\mathbf{D}_{n+\alpha} = \frac{1}{2} \left[\mathbf{L}_{n+\alpha} + \mathbf{L}^{c}_{n+\alpha} \right]$$

$$\mathbf{L}_{n+\alpha} = \frac{1}{h_n} \left[[\mathbf{F}_{n+1} - \mathbf{I}] \cdot [(1-\alpha)\, \mathbf{I} + \alpha\, \mathbf{F}_{n+1}]^{-1} \right] \tag{3.2.9}$$

Equation (3.2.4) can be rewritten as:

$$\tau_{n+1} \simeq 4C\, h_n\, \mathbf{D}_{n+1} + \mathbf{H}_{n+1} + \delta\mathbf{H}_{n+\alpha} \tag{3.2.10}$$

with: $\mathbf{H}_{n+1} = \mathbf{F}_{n+1} \cdot \tau_n \cdot \mathbf{F}^{c}_{n+1}$

and: $\delta\mathbf{H}_{n+\alpha} = 4C\, h_n \left\{ \mathbf{F}_{n+1} \cdot \mathbf{F}^{-1}_{n+\alpha} \cdot \mathbf{D}_{n+\alpha} \cdot \mathbf{F}^{-c}_{n+\alpha} \cdot \mathbf{F}^{c}_{n+1} - \mathbf{D}_{n+1} \right\}$

Together with expressions (3.2.8) and (3.2.9) this formula forms the algorithm for integrating tensor τ. In appendix B it is shown that for small deformations (when $\mathbf{F}_{n+1} = \mathbf{I} + \epsilon_{n+1}$ with $\|\epsilon_{n+1}\| << 1$) tensor $\delta\mathbf{H}_{n+\alpha}$ tends to the null tensor for all values of α.

3.3 Incremental objectivity of the integration algorithm according to Pinsky c.s.

Pinsky [5] investigated the objectivity of integration algorithms such as the algorithm presented in the previous paragraph.

According to Pinsky this algorithm is 'incrementally' objective if and only if the following conditions are satisfied:

condition A If deformation gradient $\mathbf{F}_{n+1}$ is a symmetric stretch tensor $\mathbf{U}$ then the rate of rotation tensor $\mathbf{\Omega}_{n+\alpha}$ must be equal to the null tensor.

In reverse: if $\mathbf{\Omega}_{n+\alpha}$ equals the null tensor then $\mathbf{F}_{n+1}$ must be a symmetric stretch tensor $\mathbf{U}$.

condition B If deformation gradient $\mathbf{F}_{n+1}$ is an orthogonal rigid body rotation tensor $\mathbf{R}$ the rate of strain tensor $\mathbf{D}_{n+\alpha}$ must be equal to the null tensor.

In reverse: if $\mathbf{D}_{n+\alpha}$ equals the null tensor then $\mathbf{F}_{n+1}$ must be an orthogonal rigid body rotation tensor.

Pinsky's definition of incremental objectivity does not agree with the formal, commonly accepted definition of objectivity as presented in paragraph 2.3. Pinsky merely checks the algorithm for two properties which objective algorithms must possess. In paragraph 3.4 the formal definition of objectivity will be applied on the integration algorithm.

It can be shown that condition A is satisfied for all values $0 \leq \alpha \leq 1$.

Condition B turns out to be satisfied only for $\alpha=\frac{1}{2}$. This last statement can be proven as follows:

proof: Assume $\mathbf{F}_{n+1}$ is an arbitrary rigid body rotation tensor $\mathbf{R}$.

So: $\mathbf{F}_{n+1} = \mathbf{R}$ with $\mathbf{R}^{-1} = \mathbf{R}^{c}$

if $\mathbf{D}_{n+\alpha} = \frac{1}{2}\left[\mathbf{L}_{n+\alpha} + \mathbf{L}^{c}_{n+\alpha}\right]$ has to equal the null tensor then:

$$\mathbf{L}_{n+\alpha} = -\,\mathbf{L}^{c}_{n+\alpha} \quad \text{for all tensors } \mathbf{R}$$

For $\mathbf{L}_{n+\alpha}$ holds (see 3.2.9):

$$\mathbf{L}_{n+\alpha} = \frac{1}{h_n}\left[\,[\mathbf{R}-\mathbf{I}] \cdot [(1-\alpha)\,\mathbf{I} + \alpha\,\mathbf{R}]^{-1}\right] \tag{3.3.1}$$

Using $\mathbf{R}^{-1}= \mathbf{R}^{c}$ and expression (3.3.1) for $-\mathbf{L}^{c}_{n+\alpha}$ can be found:

$$\begin{aligned}
\left[-\mathbf{L}^{c}_{n+\alpha}\right] &= \frac{1}{h_n}\left[\,[(1-\alpha)\,\mathbf{I} + \alpha\,\mathbf{R}]^{-c} \cdot [\mathbf{I}-\mathbf{R}]^{c}\right] \\
&= \frac{1}{h_n}\left[\,[(1-\alpha)\,\mathbf{I} + \alpha\,\mathbf{R}^{c}]^{-1} - [(1-\alpha)\,\mathbf{I} + \alpha\,\mathbf{R}^{c}]^{-1} \cdot \mathbf{R}^{-1}\right] \\
&= \frac{1}{h_n}\left[\,[\{(1-\alpha)\,\mathbf{R} + \alpha\,\mathbf{I}\}\cdot\mathbf{R}^{c}]^{-1} - [(1-\alpha)\,\mathbf{R} + \alpha\,\mathbf{I}]^{-1}\right] \\
&= \frac{1}{h_n}\left[\,[\mathbf{R}-\mathbf{I}] \cdot [(1-\alpha)\,\mathbf{R} + \alpha\,\mathbf{I}]^{-1}\right]
\end{aligned} \tag{3.3.2}$$

Tensor $-\mathbf{L}^{c}_{n+\alpha}$ has to equal $\mathbf{L}_{n+\alpha}$ for all rigid body rotation tensors. Comparing equation (3.3.1) with equation (3.3.2) this can only be true if $(1-\alpha)$ equals α. So α has to be equal to $\frac{1}{2}$.

■

Based on the fact that both condition A and B are satisfied when choosing $\alpha=\frac{1}{2}$ Pinsky concludes that the integration algorithm is incrementally objective for this value of α.

3.4 Objectivity of the integration algorithm

In the previous paragraph the definition of incremental objectivity has been introduced. It turned out that the integration algorithm of paragraph 3.2 is incrementally objective if and only if interpolation parameter α is chosen equal to $\frac{1}{2}$.

In this paragraph it will be investigated if the proposed algorithm is also objective in the formal sense for $\alpha=\frac{1}{2}$. The formal definition of objectivity has been introduced in paragraph 2.3: A tensor $\mathbf{N}(\mathbf{F})$, where $\mathbf{F}$ is the deformation gradient, is formally objective if:

$$\mathbf{N}(\mathbf{Q}\cdot\mathbf{F}) = \mathbf{Q}\cdot\mathbf{N}(\mathbf{F})\cdot\mathbf{Q}^{c} \qquad (2.3.2)$$

for all (extra) rigid body rotation tensors $\mathbf{Q}$.

Applying this definition on the integration algorithm, the following statement must hold when the algorithm is objective:

$$\tau_{n+1} = \mathbf{N}(\mathbf{F}_{n+1}) \;\Rightarrow\; \mathbf{N}(\mathbf{Q}\cdot\mathbf{F}_{n+1}) = \mathbf{Q}\cdot\mathbf{N}(\mathbf{F}_{n+1})\cdot\mathbf{Q}^{c} \;\; \forall\, \mathbf{Q} \qquad (3.4.1)$$

This expression has been worked out for $\alpha=\frac{1}{2}$ in appendix C. In this appendix it is proven that equation (3.4.1) is valid if $\mathbf{F}_{n+1}$ is chosen equal to an orthogonal rigid body rotation tensor $\mathbf{R}$. Strong indications can been found that (3.4.1) is invalid for symmetric stretch tensors, so for arbitrary deformation tensors $\mathbf{F}_{n+1}$ formal objectivity is improbable. This assertion can be verified by formulating a numerical example, which denies the equality in (3.4.1). Ending this chapter, such an example will be presented. From this example it can be concluded that the algorithm, though incrementally objective, is not formally objective for $\alpha=\frac{1}{2}$.

For $\alpha=\frac{1}{2}$ the integration algorithm reads: (see appendix C)

$$\tau_{n+1} = N(\mathbf{F}_{n+1}) \tag{3.4.1}$$

$$\begin{aligned} N(\mathbf{F}_{n+1}) = \; & \mathbf{F}_{n+1} \cdot \tau_n \cdot \mathbf{F}^c_{n+1} + \\ & + \; 16C \Big[\; \Big\{ (\mathrm{I}+\mathbf{F}^{-1}_{n+1})^{-1} \cdot (\mathrm{I}+\mathbf{F}^{-1}_{n+1})^{-1} \cdot (\mathrm{I}-\mathbf{F}^{-1}_{n+1}) \cdot (\mathrm{I}+\mathbf{F}^{-c}_{n+1})^{-1} \Big\} + \\ & \qquad \Big\{ (\mathrm{I}+\mathbf{F}^{-1}_{n+1})^{-1} \cdot (\mathrm{I}+\mathbf{F}^{-1}_{n+1})^{-1} \cdot (\mathrm{I}-\mathbf{F}^{-1}_{n+1}) \cdot (\mathrm{I}+\mathbf{F}^{-c}_{n+1})^{-1} \Big\}^c \Big] \end{aligned}$$

$N(\mathbf{Q}\cdot\mathbf{F}_{n+1})$

As is shown in appendix C substitution of $(\mathbf{Q}\cdot\mathbf{F}_{n+1})$ for $\mathbf{F}_{n+1}$ results in the following expression:

$$N(\mathbf{Q}\cdot\mathbf{F}_{n+1}) = \mathbf{Q}\cdot\mathbf{F}_{n+1}\cdot\tau_n\cdot\mathbf{F}_{n+1}\cdot\mathbf{Q}^c + 16C\,\mathbf{Q}\cdot[\mathrm{V}+\mathrm{V}^c]\cdot\mathbf{Q}^c \tag{3.4.2}$$

$$\text{with:} \quad \mathrm{V} = \Big\{ (\mathbf{Q}+\mathbf{F}^{-1}_{n+1})^{-1} \cdot (\mathrm{I}+\mathbf{F}^{-1}_{n+1}\cdot\mathbf{Q}^c)^{-1} \cdot (\mathrm{I}-\mathbf{F}^{-1}_{n+1}\cdot\mathbf{Q}^c) \cdot (\mathbf{Q}^c+\mathbf{F}^{-c}_{n+1})^{-1} \Big\}$$

$\mathbf{Q} \cdot N(\mathbf{F}_{n+1}) \cdot \mathbf{Q}^c$

Pre–multiplying (3.4.1) with tensor $\mathbf{Q}$ and post–multiplying this equation with $\mathbf{Q}^c$ gives:

$$\mathbf{Q}\cdot N(\mathbf{F}_{n+1})\cdot\mathbf{Q}^c = \mathbf{Q}\cdot\mathbf{F}_{n+1}\cdot\tau_n\cdot\mathbf{F}_{n+1}\cdot\mathbf{Q}^c + 16C\,\mathbf{Q}\cdot[\mathrm{W}+\mathrm{W}^c]\cdot\mathbf{Q}^c \tag{3.4.3}$$

$$\text{with:} \quad \mathrm{W} = \Big\{ (\mathrm{I}+\mathbf{F}^{-1}_{n+1})^{-1} \cdot (\mathrm{I}+\mathbf{F}^{-1}_{n+1})^{-1} \cdot (\mathrm{I}-\mathbf{F}^{-1}_{n+1}) \cdot (\mathrm{I}+\mathbf{F}^{-c}_{n+1})^{-1} \Big\}$$

Objectivity requires the right hand side of equation (3.4.2) to be equal to the right hand side of (3.4.3), so tensor $(\mathrm{V}+\mathrm{V}^c)$ has to equal tensor $(\mathrm{W}+\mathrm{W}^c)$ for all rigid body tensors $\mathbf{Q}$. From (3.4.2) and (3.4.3) there is little reason to assume this to be true.

The easiest way to proof the absence of objectivity is by means of a numerical example:

Consider a two–dimensional rectangular body, deforming from configuration (n) into configuration (n+1). The deformation consists of an isovolumetric elongation of the rectangle with elongation factor λ, followed by a rigid body rotation over φ radians.

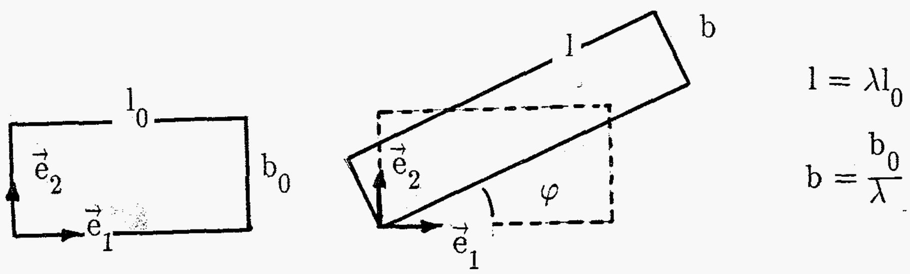


fig.3.2: elongation and rigid body rotation of a rectangular body

According to its polar decomposition the deformation matrix $\underline{F}_{n+1}$ with respect to the Cartesian reference system $[\vec{e}_1\ \vec{e}_2]$ can be written as:

$$\underline{F}_{n+1} = \underline{R}_{n+1} \cdot \underline{U}_{n+1} \tag{3.4.4}$$

$$\underline{U}_{n+1} = \text{stretch matrix} \qquad = \begin{bmatrix} \lambda & 0 \\ 0 & \frac{1}{\lambda} \end{bmatrix}$$

$$\underline{R}_{n+1} = \text{rotation matrix} \qquad = \begin{bmatrix} \cos(\varphi) & -\sin(\varphi) \\ \sin(\varphi) & \cos(\varphi) \end{bmatrix}$$

Elongation factor λ is chosen equal to 1.01 ($\equiv$ 1% elongation). Angle φ is chosen equal to $\frac{\pi}{36}$ rad. ($\equiv$ 5 degrees). In configuration (n) the stress matrix $\underline{\tau}_n$ is assumed to be equal to:

$$\underline{\tau}_n = \begin{bmatrix} 1.5 & 0 \\ 0 & 0.5 \end{bmatrix}$$

The matrices $\underline{\tau}^{A}_{n+1} = \underline{N}(\underline{Q}\cdot\underline{F}_{n+1})$ and $\underline{\tau}^{B}_{n+1} = \underline{Q}\cdot\underline{N}(\underline{F}_{n+1})\cdot\underline{Q}^{c}$ will be evaluated for several extra rigid body rotation tensors $\underline{Q}$:

$$\underline{Q} = \begin{bmatrix} \cos(\psi) & -\sin(\psi) \\ \sin(\psi) & \cos(\psi) \end{bmatrix}$$

where ψ varies between 0 and $\frac{\pi}{2}$ radians.

An easy–to–use program to perform the necessary calculations is PC–Matlab [6]. In appendix D a PC–Matlab program file is included, which calculates $\underline{\tau}^{A}_{n+1}$ and $\underline{\tau}^{B}_{n+1}$ for several values of ψ. In the next series of figures the resulting components of these matrices with respect to the fixed reference system $[\vec{e}_1\ \vec{e}_2]$ are plotted against the extra rigid body rotation angle ψ.

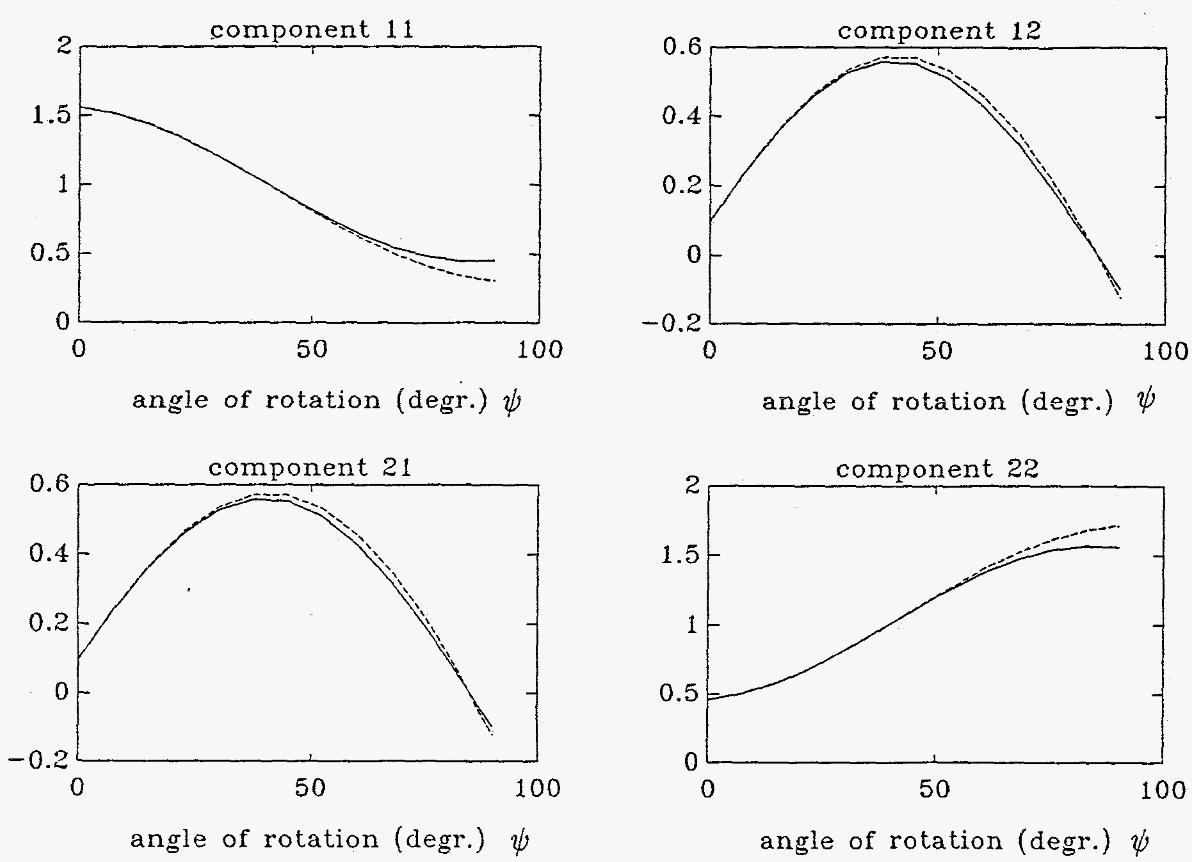


fig.3.3: components of $\underline{\tau}^{A}_{n+1}$ *and* $\underline{\tau}^{B}_{n+1}$

The dashed lines represent components of $\underline{\tau}^{A}_{n+1}$ while the solid lines represent the components of $\underline{\tau}^{B}_{n+1}$. Clearly the dashed and solid lines do not agree. The differences are too large to descend from numerical inaccuracies, so it can safely be concluded that $\underline{N}(\underline{Q}\cdot\underline{F}_{n+1})$ and $\underline{Q}\cdot\underline{N}(\underline{F}_{n+1})\cdot\underline{Q}^{c}$ are not equal and that the proposed algorithm can't be objective for $\alpha=\frac{1}{2}$.

CHAPTER 4: APPLICATIONS OF THE INTEGRATION ALGORITHM

4.1 Introduction

In this chapter some two–dimensional applications of the integration algorithm will be presented.

The first example considers a rigid body rotation. In the second example an isovolumetric elongation will be investigated. A more complex deformation pattern is the subject of the last example, where the deformation consists of a combination of simple shear, isovolumetric elongation and rigid body rotation. The integration algorithm is based on the Neo–Hookean constitutive equation for incompressible elastic materials. Hence the three examples all deal with isovolumetric deformations. Furthermore only the (pseudo–)deviatoric part $\boldsymbol{\tau}$ of the Cauchy stress tensor $\boldsymbol{\sigma}$ is regarded. The hydrostatic part of $\boldsymbol{\sigma}$ will be determined by the boundary conditions.

The numerical simulations have been performed on a PC using PC–Matlab [6]. Program files of these simulations are included in appendix E. In all calculations both the value of interpolation parameter α and the number of increments in which the end configuration is realized are varied.

4.2 Rigid Body Rotation

In the $(\vec{e}_1,\vec{e}_2)$–plane a rigid body is rotated over angle $\varphi=\frac{2\pi}{3}$ rad. as is sketched in figure 4.1. In the reference configuration stress tensor $\boldsymbol{\tau}$ is assumed to equal the null tensor. Using the algorithm of chapter 3, tensor $\boldsymbol{\tau}$ in the end configuration will be calculated.

Two reference systems are introduced; the global reference system $[\vec{e}_1,\vec{e}_2]$ is fixed in space, while the local reference system $[\vec{e}^*_1,\vec{e}^*_2]$ is attached to the rigid body.

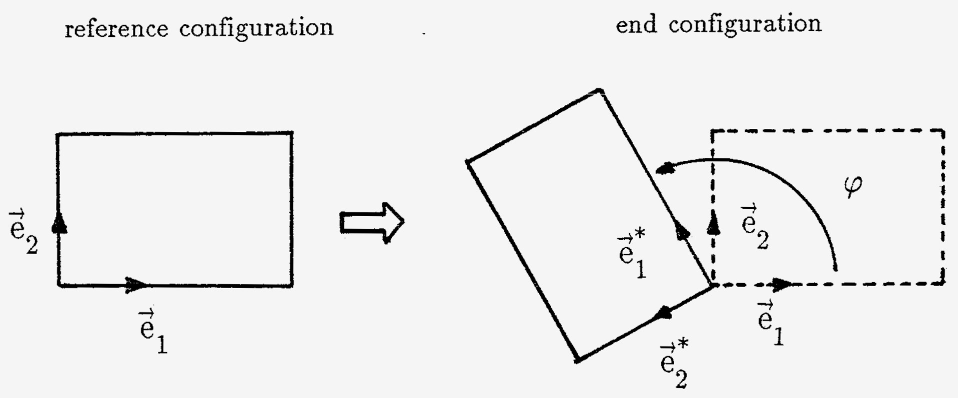


fig.4.1: Local and global reference systems

The total deformation tensor $\mathbf{F}_N$ equals:

$$\mathbf{F}_N = \cos(\varphi)\vec{e}_1\vec{e}_1 - \sin(\varphi)\vec{e}_1\vec{e}_2 + \sin(\varphi)\vec{e}_2\vec{e}_1 + \cos(\varphi)\vec{e}_2\vec{e}_2 \tag{4.2.1}$$

$\underline{F}_N$ is the matrix representation of $\mathbf{F}_N$ with respect to the global reference system:

$$\underline{F}_N = \begin{bmatrix} \cos(\varphi) & -\sin(\varphi) \\ \sin(\varphi) & \cos(\varphi) \end{bmatrix} \tag{4.2.2}$$

According to the definition in paragraph 3.1 the matrix representation of τ_N in the end configuration with respect to the global reference system equals:

$$\underline{\tau}_N = 2\,C\,[\,\underline{F}_N \cdot \underline{F}_N^T - \underline{I}\,] \tag{4.2.3}$$

where $\underline{F}_N^T$ is the transposed of $\underline{F}_N$.

Substitution of (4.2.2) in (4.2.3) easily shows that (of course) $\underline{\tau}_N$ has to equal the null matrix. The matrix representation of τ_N with respect to the local reference system $[\vec{e}_1^*, \vec{e}_2^*]$ is denoted by $\underline{\tau}_N^*$. This matrix, which can be interpreted as the matrix $\underline{\tau}_N$ neutralized for rigid body rotations, also equals the null matrix.

The components of $\underline{\tau}_N^*$ are calculated with the proposed integration algorithm implemented in PC–Matlab. The total deformation is divided into N increments. In each increment the same deformation matrix with respect to the local reference system is used (updated approach). This deformation matrix is:

$$\underline{F}_{incr} = \begin{bmatrix} \cos(\frac{\varphi}{N}) & -\sin(\frac{\varphi}{N}) \\ \sin(\frac{\varphi}{N}) & \cos(\frac{\varphi}{N}) \end{bmatrix} \tag{4.2.4}$$

Five different values for the interpolation parameter α and six different values for the number of increments are evaluated. The results are plotted in figure 4.2.

From this figure, it can be concluded that, even using 480 increments, only for $\alpha=\frac{1}{2}$ the components $(\underline{\tau}_N^*)_{11}$ and $(\underline{\tau}_N^*)_{22}$ equal to zero. This agrees with the conclusion of paragraph 3.1, stating that the integration algorithm can only be incrementally objective, thus insensitive for rigid body rotations, for $\alpha=\frac{1}{2}$.

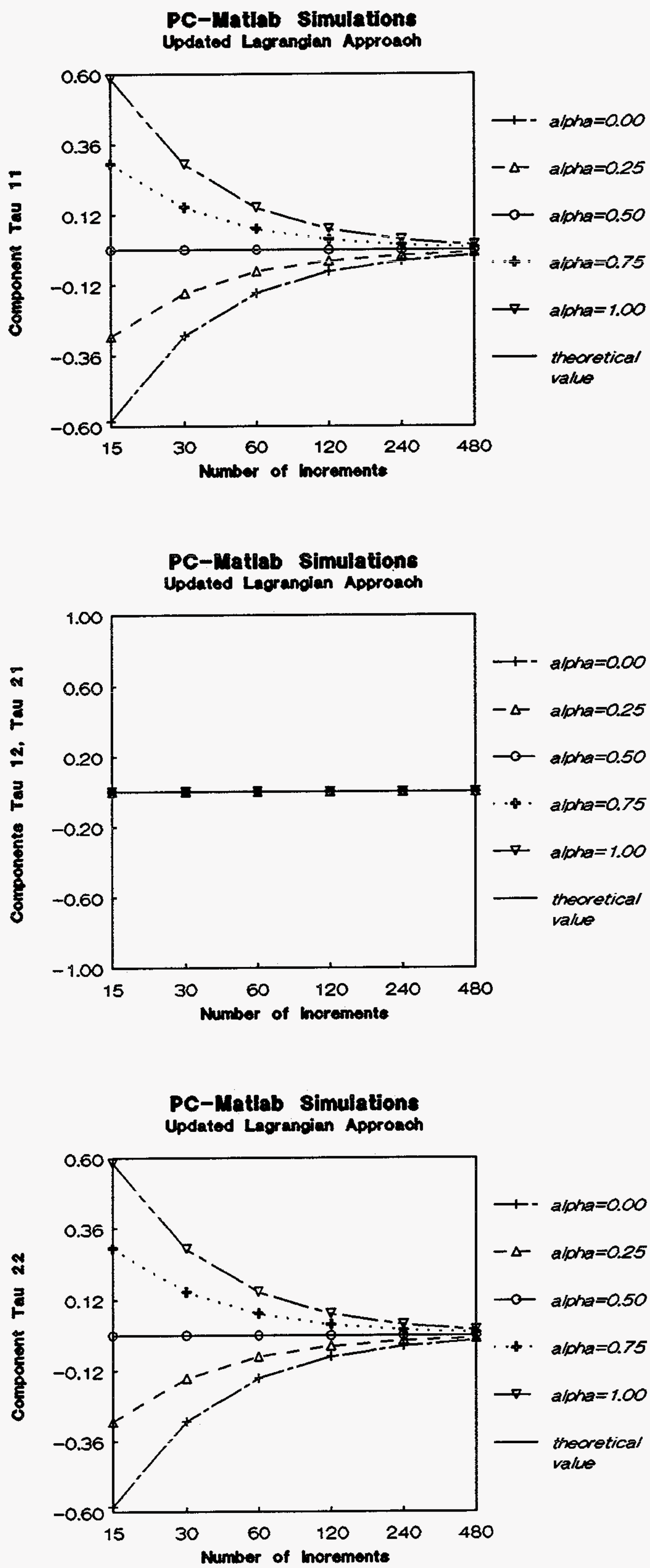


fig.4.2: results for the rigid body rotation

4.3 Isovolumetric Elongation

The body depicted in figure 4.1 now is elongated along the $\vec{e}_1$–axis. In the $\vec{e}_1$–direction the elongation factor is $\lambda = 1.3$. The deformation is isovolumetric, so in the $\vec{e}_2$–direction the elongation factor equals $\frac{1}{\lambda} = \frac{1}{1.3}$.

Again tensor τ in the reference configuration is assumed to equal the null tensor in the reference configuration. The material parameter C is chosen equal to 1.0. No rigid body rotation occurs so the local reference system $[\vec{e}_1^{\,*},\vec{e}_2^{\,*}]$ coincides with the global reference system $[\vec{e}_1,\vec{e}_2]$.

The total deformation tensor is given by:

$$\mathbf{F}_N = \lambda\, \vec{e}_1\vec{e}_1 + \frac{1}{\lambda}\, \vec{e}_2\vec{e}_2 \tag{4.3.1}$$

So the matrix representation of τ_N in the end configuration (with respect to the local or the global reference system) can be written as:

$$\underline{\tau}_N = 2C[\, \underline{F}_N \cdot \underline{F}_N^T - \underline{I}\,] = 2C\begin{bmatrix} \lambda^2-1 & 0 \\ 0 & \frac{1}{\lambda^2}-1 \end{bmatrix} = \begin{bmatrix} 1.3800 & 0 \\ 0 & -0.8166 \end{bmatrix} \tag{4.3.2}$$

This matrix is approximated using the integration algorithm. The total deformation is split in N increments. It can easily be seen that the incremental deformation matrix has to equal:

$$\underline{F}_{incr} = \begin{bmatrix} \sqrt[N]{\lambda} & 0 \\ 0 & \sqrt[N]{1/\lambda} \end{bmatrix} \qquad \text{so: } (\underline{F}_{incr})^N = \underline{F}_N \tag{4.3.3}$$

The components of $\underline{\tau}_N$, resulting from the numerical simulations are presented in figure 4.3.

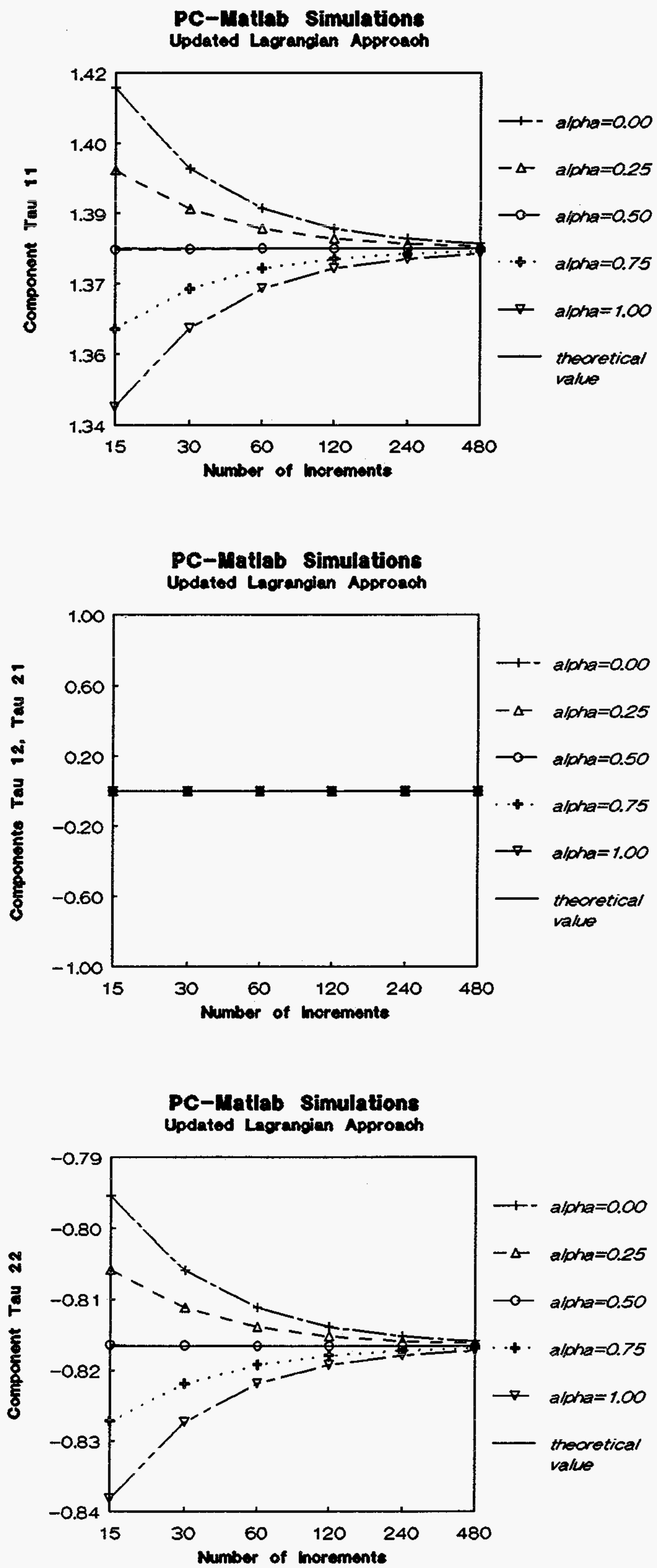


fig.4.3: results for the isovolumetric elongation

There is a remarkable resemblance with the results of the rigid body rotation simulation. Again for $\alpha=\frac{1}{2}$ the results are far more better than for the other values of α. But now, because of the absence of rigid body rotation, this can't be explained by the incremental objectivity of the algorithm. For $\alpha=\frac{1}{2}$ the integration algorithm is not only incrementally objective, but it also turns out to be far more accurate. (Using $\alpha=0$ / $\alpha=1$ or using $\alpha=\frac{1}{2}$ could be compared with integrating an arbitrary function respectively using rectangular integration or using trapezoidal integration as depicted in figure 4.4)

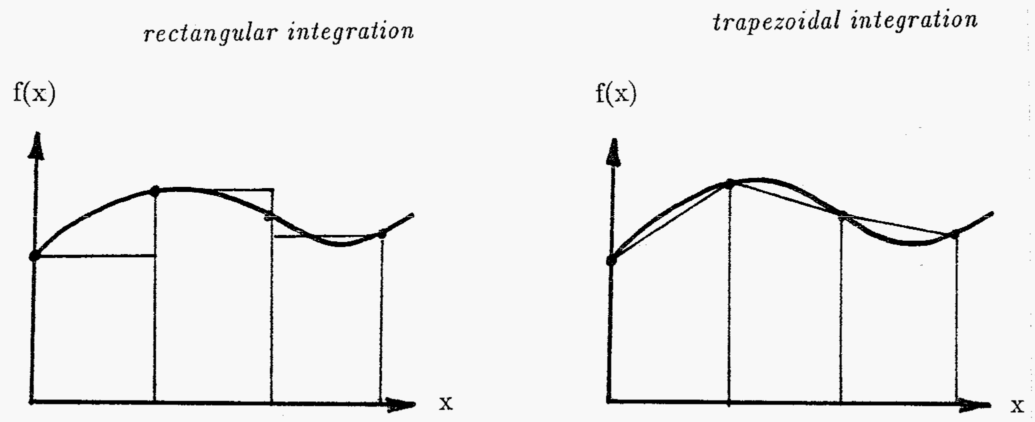


fig.4.4: rectangular and trapezoidal integration

4.4 Simple Shear, Isovolumetric Elongation and Rigid Body Rotation

In this example, the end configuration of the deforming body is reached after:

(1) simple shear over φ=60 degrees

(2) isovolumetric elongation with λ=1.3

(3) rigid body rotation over ψ=120 degrees

In the reference system stress tensor τ is the null tensor. Material parameter C is chosen equal to 1.0. The total deformation process is presented in figure 4.5

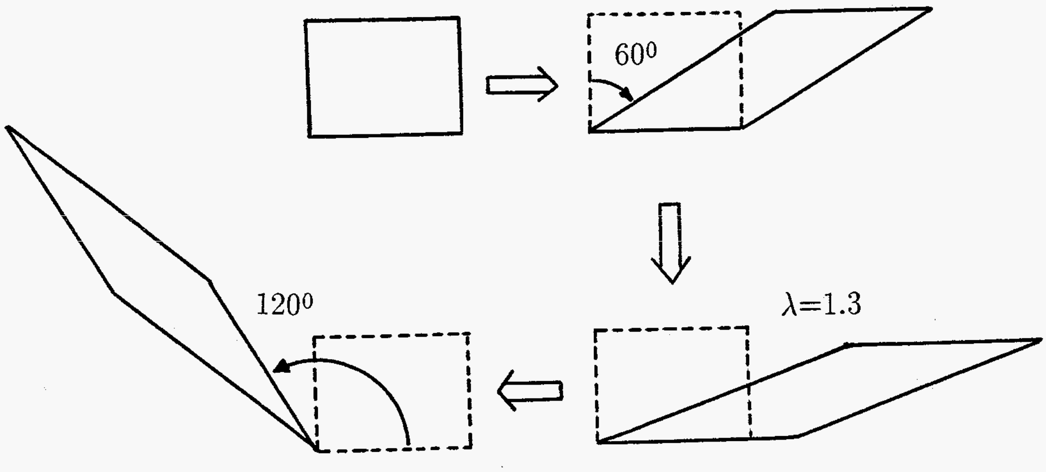


fig.4.5: Realisation of the end configuration

The deformation matrix $\underline{F}_N$ with respect to $[\vec{e}_1,\vec{e}_2]$, transforming the reference configuration into the end configuration is given by:

$$\underline{F}_N = \underline{F}_3 \cdot \underline{F}_2 \cdot \underline{F}_1 \tag{4.4.1}$$

where $\underline{F}_1 = \begin{bmatrix} 1 & \tan(\varphi) \\ 0 & 1 \end{bmatrix}$; simple shear over $\varphi = \frac{\pi}{3}$ rad.

$\underline{F}_2 = \begin{bmatrix} \lambda & 0 \\ 0 & 1/\lambda \end{bmatrix}$; isovolumetric elong. with λ=1.3

$\underline{F}_3 = \begin{bmatrix} \cos(\psi) & -\sin(\psi) \\ \sin(\psi) & \cos(\psi) \end{bmatrix}$; rigid body rotation, $\psi = \frac{2\pi}{3}$ rad.

The matrix representation of τ_N with respect to the global reference system is calculated using:

$$\underline{\tau}_N = 2C[\,\underline{F}_N \cdot \underline{F}_N^T - \underline{I}\,] = \begin{bmatrix} 5.2676 & -7.0739 \\ -7.0739 & 5.4359 \end{bmatrix} \tag{4.4.2}$$

Neutralizing $\underline{\tau}_N$ for rigid body rotations, $\underline{\tau}_N^*$ is calculated by:

$$\underline{\tau}_N^* = \underline{F}_3^T \cdot \underline{\tau}_N \cdot \underline{F}_3 = \begin{bmatrix} 11.5200 & 3.4641 \\ 3.4641 & -0.8166 \end{bmatrix} \tag{4.4.3}$$

The components of $\underline{\tau}^*_N$ have been approximated using the integration algorithm implemented in the program file MIXED.M (see appendix E). In each increment the same incremental deformation matrix is used. This matrix is not trivial: It can be proven that application of the following incremental deformation matrix (in the updated approach) will result in the correct end configuration.

$$\underline{F}_{incr} = \begin{bmatrix} \cos(\frac{\psi}{N}) & -\sin(\frac{\psi}{N}) \\ \sin(\frac{\psi}{N}) & \cos(\frac{\psi}{N}) \end{bmatrix} \begin{bmatrix} \sqrt[N]{\lambda} & 0 \\ 0 & \sqrt[N]{1/\lambda} \end{bmatrix} \begin{bmatrix} 1 & \frac{\lambda\tan(\varphi)}{\beta_N} \\ 0 & 1 \end{bmatrix} \tag{4.4.4}$$

with: N = number of increments

β_N calculated by:
$$\beta_i = \sqrt[N]{\lambda}\left\{\beta_{i-1} + \lambda^{(\frac{1-i}{N})}\right\}$$
$$\beta_1 = \sqrt[N]{\lambda}$$

The resulting stress components with respect to the local reference system are plotted in figure 4.6.

Again for $\alpha=\frac{1}{2}$ the best results are obtained. From paragraph 4.2 and 4.3 we know that this is due to the incremental objectivity and the higher accuracy of the integration algorithm.

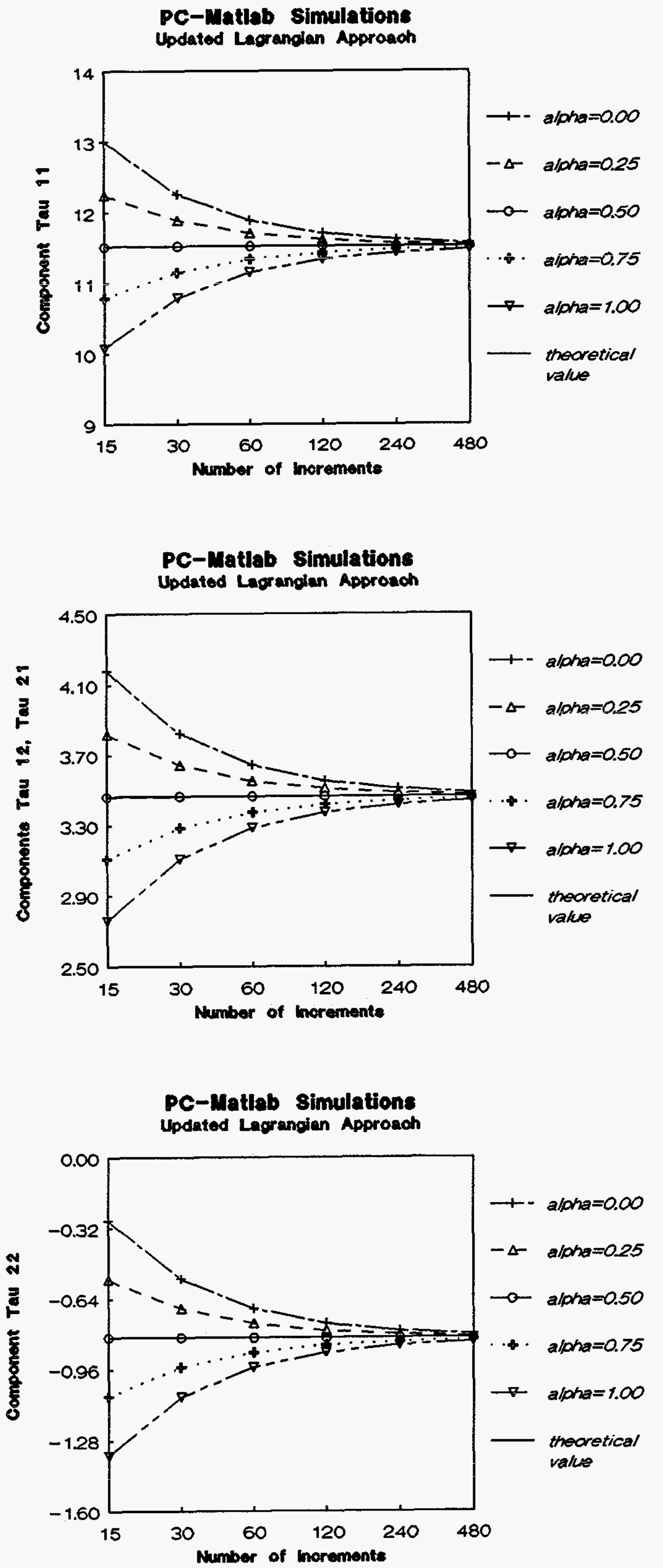


fig.4.6: *results for simple shear, isovolumetric elongation and rigid body rotation*

CHAPTER 5: THE NEO–HOOKEAN CONSTITUTIVE EQUATION IN AN EULERIAN APPROACH: AN ALTERNATIVE CALCULATION METHOD

5.1 Introduction

In chapter 3 an algorithm for integrating stress tensor $\boldsymbol{\tau}$ has been proposed. This algorithm is based on the definition of the so–called Truesdell rate of $\boldsymbol{\tau}$ and uses an estimation of the deformation tensor $\mathbf{F}_{n+1}$.

In this chapter an algorithm which uses an estimation for the velocity gradient $\mathbf{L}_{n+1}$ will be presented. Like the algorithm in chapter 3 this algorithm can also be 'incrementally objective' if a correct value for the interpolation parameter is chosen.

The formal definitions of the deformation gradient and the velocity gradient are:

$$\mathbf{F}_{n+1} = (\vec{\nabla}_n \vec{x}_{n+1})^c \tag{5.1.1}$$

$$\mathbf{L}_{n+1} = (\vec{\nabla}_{n+1} \vec{v}_{n+1})^c \tag{5.1.2}$$

As can be seen from these definitions the deformation tensor is defined with respect to a reference configuration (configuration n). In an Eulerian approach it is desirable to avoid quantities defined with respect to a reference configuration as much as possible. Then only current values have to be taken into account and no difficult trace–back procedures have to be applied. That is why the algorithm in this chapter is preferred above the algorithm in chapter 3.

5.2 Derivation of the algorithm

As in chapter 3 an updated formulation will be used when deriving the algorithm. Three different configurations will be considered:

the starting configuration	at $t=t_0$
the reference configuration	at $t=t_n$
the current configuration	at $t=t_{n+1}$

Deformation tensor $\tilde{\mathbf{F}}_n$ describes the *total* deformation from t_0 to t_n

Deformation tensor $\tilde{\mathbf{F}}_{n+1}$ describes the *total* deformation from t_0 to t_{n+1}

Deformation tensor $\mathbf{F}_{n+1}$ describes the *incremental* deformation from t_n to t_{n+1}

so:

$$\tilde{\mathbf{F}}_{n+1} = \mathbf{F}_{n+1} \cdot \tilde{\mathbf{F}}_n \tag{5.2.1}$$

The velocity gradient $\mathbf{L}_{n+1} = (\vec{\nabla}_{n+1}\vec{v}_{n+1})^c$ is assumed to be known.

According to definition (2.1.11) for an arbitrary velocity gradient $\mathbf{L}$ equation (5.2.2) must hold:

$$\mathbf{L} = \dot{\mathbf{F}} \cdot \mathbf{F}^{-1} \tag{5.2.2}$$

so:

$$\mathbf{L} \cdot \mathbf{F} = \dot{\mathbf{F}} \tag{5.2.3}$$

Integrating equation (5.2.3) from $t=t_n$ to $t=t_{n+1}$ results in:

$$\int_{t_n}^{t_{n+1}} \mathbf{L} \cdot \mathbf{F} \, dt = \int_{t_n}^{t_{n+1}} \dot{\mathbf{F}} \, dt \tag{5.2.4}$$

For the right–hand side it can easily be found that:

$$\int_{t_n}^{t_{n+1}} \dot{\mathbf{F}} \, dt = \tilde{\mathbf{F}}_{n+1} - \tilde{\mathbf{F}}_n = \mathbf{F}_{n+1} \cdot \tilde{\mathbf{F}}_n - \tilde{\mathbf{F}}_n = (\mathbf{F}_{n+1} - \mathbf{I}) \cdot \tilde{\mathbf{F}}_n \tag{5.2.5}$$

It can be shown that there must be a value of α $(0 \leq \alpha \leq 1)$ for which the left–hand side of (5.2.4) equals:

$$\int_{t_n}^{t_{n+1}} \mathbf{L} \cdot \mathbf{F}\, dt = \int_{t_n}^{t_{n+1}} \mathbf{L}\, dt \cdot \tilde{\mathbf{F}}_{n+\alpha} \qquad (0 \leq \alpha \leq 1) \tag{5.2.6}$$

As in chapter 3 tensor $\tilde{\mathbf{F}}_{n+\alpha}$ can be approximated by:

$$\tilde{\mathbf{F}}_{n+\alpha} \approx (1-\alpha)\, \tilde{\mathbf{F}}_n + \alpha\, \tilde{\mathbf{F}}_{n+1} = [(1-\alpha)\, \mathbf{I} + \alpha\, \mathbf{F}_{n+1}] \cdot \tilde{\mathbf{F}}_n \tag{5.2.7}$$

This approximation is based on a linear interpolation of the position vector $\vec{x}(t)$, which is allowed for small time steps Δt.

For small time steps the integral of **L** can be estimated by:

$$\int_{t_n}^{t_{n+1}} \mathbf{L}\, dt \approx \mathbf{L}_{n+1}\, \{t_{n+1} - t_n\} = \mathbf{L}_{n+1} \Delta t \tag{5.2.8}$$

Substitution of eq. (5.2.5), (5.2.6), (5.2.7) and (5.2.8) in equation (5.2.4) gives:

$$\mathbf{L}_{n+1} \Delta t \cdot \{(1-\alpha)\, \mathbf{I} + \alpha\, \mathbf{F}_{n+1}\} \cdot \tilde{\mathbf{F}}_n = (\mathbf{F}_{n+1} - \mathbf{I}) \cdot \tilde{\mathbf{F}}_n \tag{5.2.9}$$

This expression can be rearranged:

$$\mathbf{F}_{n+1} = (\mathbf{I} - \alpha\, \mathbf{L}_{n+1} \Delta t)^{-1} \cdot (\mathbf{I} + (1-\alpha)\, \mathbf{L}_{n+1} \Delta t) \tag{5.2.10}$$

Formula (5.2.10) approximates the incremental deformation tensor $\mathbf{F}_{n+1}$ when $\mathbf{L}_{n+1}$ is given.

Stress tensor τ is defined as:

$$\tau = 2\ \mathrm{C}\ \{\mathbf{F}\cdot\mathbf{F}^{c} - \mathbf{I}\} \tag{5.2.11}$$

So in configuration t_n holds:

$$\tau_n = 2\ \mathrm{C}\ \{\tilde{\mathbf{F}}_n\cdot\tilde{\mathbf{F}}_n^{c} - \mathbf{I}\} \tag{5.2.12}$$

and in configuration t_{n+1}:

$$\begin{aligned}
\tau_{n+1} &= 2\ \mathrm{C}\ \{\tilde{\mathbf{F}}_{n+1}\cdot\tilde{\mathbf{F}}_{n+1}^{c} - \mathbf{I}\} \\
&= 2\ \mathrm{C}\ \{\mathbf{F}_{n+1}\cdot\tilde{\mathbf{F}}_n\cdot\tilde{\mathbf{F}}_n^{c}\cdot\mathbf{F}_{n+1}^{c} - \mathbf{I}\} \\
&= \mathbf{F}_{n+1}\cdot\ [2\ \mathrm{C}\ \{\tilde{\mathbf{F}}_n\cdot\tilde{\mathbf{F}}_n^{c} - \mathbf{I}\}]\ \cdot\mathbf{F}_{n+1}^{c} + 2\ \mathrm{C}\ (\mathbf{F}_{n+1}\cdot\mathbf{F}_{n+1}^{c} - \mathbf{I}) \\
&= \mathbf{F}_{n+1}\cdot\ \tau_n\cdot\ \mathbf{F}_{n+1}^{c} + 2\ \mathrm{C}\ (\mathbf{F}_{n+1}\cdot\mathbf{F}_{n+1}^{c} - \mathbf{I})
\end{aligned} \tag{5.2.13}$$

Now tensor τ_{n+1} can be calculated when tensor τ_n is known.

Resuming:

Alternative algorithm for calculating stress tensor τ

a) Tensor τ_n is assumed to be known

b) For tensor $\mathbf{L}_{n+1}$ an approximation must be available

c) Then the incremental deformation tensor $\mathbf{F}_{n+1}$ can be calculated using:

$$\mathbf{F}_{n+1} = (\mathbf{I} - \alpha\ \mathbf{L}_{n+1}\Delta t)^{-1}\cdot(\mathbf{I} + (1-\alpha)\ \mathbf{L}_{n+1}\Delta t)$$

d) Interpolation parameter α is to chosen from $(0 \leq \alpha \leq 1)$

e) Finally stress tensor τ_{n+1} can be calculated:

$$\tau_{n+1} = \mathbf{F}_{n+1}\cdot\ \tau_n\cdot\ \mathbf{F}_{n+1}^{c} + 2\ \mathrm{C}\ (\mathbf{F}_{n+1}\cdot\mathbf{F}_{n+1}^{c} - \mathbf{I})$$

5.3 Incremental objectivity of the algorithm

The proposed algorithm is incrementally objective if and only if two conditions are satisfied:

condition A

1. If the rate of rotation tensor $\mathbf{\Omega}_{n+1} = \frac{1}{2}\{\mathbf{L}_{n+1} - \mathbf{L}^c_{n+1}\}$ equals to the null tensor the deformation tensor $\mathbf{F}_{n+1}$ must be a symmetric stretch tensor.
2. In reverse: if $\mathbf{F}_{n+1}$ is symmetric then the rate of rotation tensor $\mathbf{\Omega}_{n+1}$ must be equal to the null tensor.

condition B

1. If the rate of deformation tensor $\mathbf{D}_{n+1} = \frac{1}{2}\{\mathbf{L}_{n+1} + \mathbf{L}^c_{n+1}\}$ equals to the null tensor the deformation tensor $\mathbf{F}_{n+1}$ must be an orthogonal rigid body rotation tensor.
2. In reverse: if $\mathbf{F}_{n+1}$ is orthogonal then the rate of deformation tensor $\mathbf{D}_{n+1}$ must be equal to the null tensor.

In this paragraph it will be proven that the algorithm is incrementally objective if and only if the interpolation parameter α is chosen equal to a half, just as the algorithm in the preceeding chapters.

Proof:

condition A.1

If $\boldsymbol{\Omega}_{n+1} = \frac{1}{2}\{\mathbf{L}_{n+1} - \mathbf{L}^c_{n+1}\}$ equals to the null tensor then $\mathbf{L}_{n+1}$ equals $\mathbf{L}^c_{n+1}$.
According to paragraph 5.2 the incremental deformation tensor $\mathbf{F}_{n+1}$ is calculated by:

$$\mathbf{F}_{n+1} = (\mathbf{I} - \alpha\, \mathbf{L}_{n+1}\Delta t)^{-1} \cdot (\mathbf{I} + (1-\alpha)\, \mathbf{L}_{n+1}\Delta t) \tag{5.3.1}$$

The conjugate incremental deformation tensor equals:

$$\begin{aligned} \mathbf{F}^c_{n+1} &= (\mathbf{I} + (1-\alpha)\, \mathbf{L}_{n+1}\Delta t))^c \cdot (\mathbf{I} - \alpha\, \mathbf{L}_{n+1}\Delta t)^{-c} \\ &= (\mathbf{I} + (1-\alpha)\, \mathbf{L}^c_{n+1}\Delta t)) \cdot (\mathbf{I} - \alpha\, \mathbf{L}^c_{n+1}\Delta t)^{-1} \end{aligned} \tag{5.3.2}$$

and if $\mathbf{L}_{n+1} = \mathbf{L}^c_{n+1}$:

$$\begin{aligned} \mathbf{F}^c_{n+1} &= (\mathbf{I} + (1-\alpha)\, \mathbf{L}_{n+1}\Delta t)) \cdot (\mathbf{I} - \alpha\, \mathbf{L}_{n+1}\Delta t)^{-1} \\ &= (\mathbf{I} - \alpha\, \mathbf{L}_{n+1}\Delta t)^{-1} + (1-\alpha)\, \mathbf{L}_{n+1}\Delta t \cdot (\mathbf{I} - \alpha\, \mathbf{L}_{n+1}\Delta t)^{-1} \end{aligned} \tag{5.3.3}$$

For all tensors $\mathbf{A}$, for which $(\mathbf{I} + \mathbf{A})$ is regular, holds:

$$\mathbf{A} \cdot (\mathbf{I} + \mathbf{A})^{-1} = (\mathbf{I} + \mathbf{A})^{-1} \cdot \mathbf{A} \tag{5.3.4}$$

So:

$$(1-\alpha)\, \mathbf{L}_{n+1}\Delta t \cdot (\mathbf{I} - \alpha\, \mathbf{L}_{n+1}\Delta t)^{-1} = (\mathbf{I} - \alpha\, \mathbf{L}_{n+1}\Delta t)^{-1} \cdot (1-\alpha)\, \mathbf{L}_{n+1}\Delta t \tag{5.3.5}$$

Substitution of (5.3.5) in (5.3.3) gives:

$$\begin{aligned} \mathbf{F}^c_{n+1} &= (\mathbf{I} - \alpha\, \mathbf{L}_{n+1}\Delta t)^{-1} + (\mathbf{I} - \alpha\, \mathbf{L}_{n+1}\Delta t)^{-1} \cdot (1-\alpha)\, \mathbf{L}_{n+1}\Delta t \\ &= (\mathbf{I} - \alpha\, \mathbf{L}_{n+1}\Delta t)^{-1} \cdot (\mathbf{I} + (1-\alpha)\, \mathbf{L}_{n+1}\Delta t\ ) \end{aligned} \tag{5.3.6}$$

The comparison of equation (5.3.6) with equation (5.3.1) shows that tensor $\mathbf{F}^c_{n+1}$ equals $\mathbf{F}_{n+1}$ for all values of α if $\mathbf{L}_{n+1} = \mathbf{L}^c_{n+1}$.

condition A.2

From equation (5.2.10) the following expression for $\mathbf{L}_{n+1}$ can be deduced:

$$\mathbf{L}_{n+1} = \frac{1}{\Delta t}(\mathbf{F}_{n+1} - \mathbf{I}) \cdot \{ (1-\alpha)\,\mathbf{I} + \alpha\,\mathbf{F}_{n+1} \}^{-1} \tag{5.3.7}$$

For $\mathbf{L}^c_{n+1}$ holds:

$$\begin{aligned} \mathbf{L}^c_{n+1} &= \frac{1}{\Delta t}\{ (1-\alpha)\,\mathbf{I} + \alpha\,\mathbf{F}_{n+1} \}^{-c} \cdot (\mathbf{F}_{n+1} - \mathbf{I})^c \\ &= \frac{1}{\Delta t}\{ (1-\alpha)\,\mathbf{I} + \alpha\,\mathbf{F}^c_{n+1} \}^{-1} \cdot (\mathbf{F}^c_{n+1} - \mathbf{I}) \end{aligned} \tag{5.3.8}$$

If $\mathbf{F}_{n+1}$ is symmetric then:

$$\mathbf{L}^c_{n+1} = \frac{1}{\Delta t}\{ (1-\alpha)\,\mathbf{I} + \alpha\,\mathbf{F}_{n+1} \}^{-1} \cdot (\mathbf{F}_{n+1} - \mathbf{I}) \tag{5.3.9}$$

Again after using equation (5.3.4) for $\mathbf{L}^c_{n+1}$ can be found:

$$\mathbf{L}^c_{n+1} = \frac{1}{\Delta t}(\mathbf{F}_{n+1} - \mathbf{I}) \cdot \{ (1-\alpha)\,\mathbf{I} + \alpha\,\mathbf{F}_{n+1} \}^{-1} \tag{5.3.10}$$

So if $\mathbf{F}_{n+1}$ equals $\mathbf{F}^c_{n+1}$ tensor $\mathbf{L}_{n+1}$ will be equal to $\mathbf{L}^c_{n+1}$ for all values of α and the rate of rotation tensor $\mathbf{\Omega}_{n+1}$ will be equal to the null tensor.

<u>condition B.1</u>

If $\mathbf{D}_{n+1} = \frac{1}{2}\{\mathbf{L}_{n+1}+\mathbf{L}^c_{n+1}\}$ equals to the null tensor then $\mathbf{L}_{n+1}$ equals $(-\mathbf{L}^c_{n+1})$.
For the conjugate deformation tensor $\mathbf{F}_{n+1}$ was found (eq.(5.3.2)):

$$\mathbf{F}^c_{n+1} = (\mathbf{I} + (1-\alpha)\,\mathbf{L}^c_{n+1}\Delta t) \cdot (\mathbf{I} - \alpha\,\mathbf{L}^c_{n+1}\Delta t)^{-1} \tag{5.3.11}$$

If $\mathbf{L}_{n+1} = -\mathbf{L}^c_{n+1}$ then:

$$\mathbf{F}^c_{n+1} = (\mathbf{I} + (\alpha-1)\,\mathbf{L}_{n+1}\Delta t) \cdot (\mathbf{I} + \alpha\,\mathbf{L}_{n+1}\Delta t)^{-1} \tag{5.3.12}$$

and for $\mathbf{F}_{n+1}\cdot\mathbf{F}^c_{n+1}$ holds:

$$\begin{aligned} \mathbf{F}_{n+1}\cdot\mathbf{F}^c_{n+1} &= [\mathbf{I}-\alpha\mathbf{L}_{n+1}\Delta t]^{-1}\cdot[\mathbf{I}+(1-\alpha)\mathbf{L}_{n+1}\Delta t]\cdot[\mathbf{I}+(\alpha-1)\mathbf{L}_{n+1}\Delta t]\cdot[\mathbf{I}+\alpha\mathbf{L}_{n+1}\Delta t]^{-1} \\ &= [\mathbf{I} - \alpha\,\mathbf{L}_{n+1}\Delta t]^{-1}\cdot[\mathbf{I} + \alpha\,\mathbf{L}_{n+1}\Delta t]^{-1} \\ &\quad - \{(1-\alpha)\Delta t\}^2\,[\mathbf{I} - \alpha\,\mathbf{L}_{n+1}\Delta t]^{-1}\cdot\,\mathbf{L}^2_{n+1}\cdot\,[\mathbf{I} + \alpha\,\mathbf{L}_{n+1}\Delta t]^{-1} \end{aligned} \tag{5.3.13}$$

After multiple use of the equality $\mathbf{A}\cdot(\mathbf{A}+\mathbf{B})^{-1}\cdot\mathbf{B} = (\mathbf{A}^{-1}+\mathbf{B}^{-1})^{-1}$ equation (5.3.13) can be rewritten:

$$\mathbf{F}_{n+1}\cdot\mathbf{F}^c_{n+1} = [\mathbf{I} - \{(1-\alpha)\,\mathbf{L}_{n+1}\Delta t\}^2] \cdot [\mathbf{I} - \{\alpha\,\mathbf{L}_{n+1}\Delta t\}^2]^{-1} \tag{5.3.14}$$

Incremental objectivity requires $\mathbf{F}_{n+1}\cdot\mathbf{F}^c_{n+1}$ to be equal to the unity tensor; if tensor $\mathbf{L}_{n+1}$ is skew–symmetric then tensor $\mathbf{F}_{n+1}$ must be orthogonal. From equation (5.3.14) it can be deduced that $[\mathbf{I} - \{(1-\alpha)\,\mathbf{L}_{n+1}\Delta t\}^2]$ has to equal $[\mathbf{I} - \{\alpha\,\mathbf{L}_{n+1}\Delta t\}^2]$ to obtain incremental objectivity. This is only possible when $(1-\alpha)^2$ equals α^2, and this can only be true when α is chosen equal to $\frac{1}{2}$.

<u>condition B.2</u>

If $\mathbf{F}_{n+1}$ is orthogonal then $\mathbf{F}_{n+1} \cdot \mathbf{F}^c_{n+1}$ equals to the unity tensor $\mathbf{I}$, so:

$$\mathbf{F}^{-1}_{n+1} = \mathbf{F}^c_{n+1} \tag{5.3.15}$$

For $\mathbf{L}_{n+1}$ was found (equation 5.3.7):

$$\mathbf{L}_{n+1} = \frac{1}{\Delta t}(\mathbf{F}_{n+1} - \mathbf{I}) \cdot \{ (1-\alpha)\,\mathbf{I} + \alpha\,\mathbf{F}_{n+1} \}^{-1} \tag{5.3.16}$$

and:

$$\mathbf{L}^c_{n+1} = \frac{1}{\Delta t}\{ (1-\alpha)\,\mathbf{I} + \alpha\,\mathbf{F}^c_{n+1} \}^{-1} \cdot (\mathbf{F}^c_{n+1} - \mathbf{I}) \tag{5.3.17}$$

For orthogonal deformation tensors $\mathbf{F}_{n+1}$ equation (5.3.17) transforms into:

$$\begin{aligned} \mathbf{L}^c_{n+1} &= \frac{1}{\Delta t}\{ (1-\alpha)\,\mathbf{I} + \alpha\,\mathbf{F}^{-1}_{n+1} \}^{-1} \cdot (\mathbf{F}^{-1}_{n+1} - \mathbf{I}) \\ &= \frac{1}{\Delta t}[\, \{(1-\alpha)\,\mathbf{I} + \alpha\,\mathbf{F}^{-1}_{n+1} \}^{-1} \cdot \mathbf{F}^{-1}_{n+1} - \{(1-\alpha)\,\mathbf{I} + \alpha\,\mathbf{F}^{-1}_{n+1} \}^{-1})\,] \end{aligned} \tag{5.3.18}$$

Both tensor products in the last expression can be rewritten using the equality $\mathbf{A} \cdot (\mathbf{A}+\mathbf{B})^{-1} \cdot \mathbf{B} = (\mathbf{A}^{-1} + \mathbf{B}^{-1})^{-1}$. This results in:

$$\begin{aligned} \mathbf{L}^c_{n+1} &= \frac{1}{\Delta t}[\, \{(1-\alpha)\mathbf{F}_{n+1} + \alpha\mathbf{I}\}^{-1} - \mathbf{F}_{n+1} \cdot \{(1-\alpha)\mathbf{F}_{n+1} + \alpha\mathbf{I}\}^{-1}] \\ &= \frac{1}{\Delta t}(\mathbf{I} - \mathbf{F}_{n+1}) \cdot \{(1-\alpha)\mathbf{F}_{n+1} + \alpha\mathbf{I}\}^{-1} \\ &= -\frac{1}{\Delta t}(\mathbf{F}_{n+1} - \mathbf{I}) \cdot \{(1-\alpha)\mathbf{F}_{n+1} + \alpha\mathbf{I}\}^{-1} \end{aligned} \tag{5.3.19}$$

For an orthogonal deformation tensor $\mathbf{F}_{n+1}$ incremental objectivity demands the rate of deformation tensor $\mathbf{D}_{n+1} = \frac{1}{2}\{\mathbf{L}_{n+1} + \mathbf{L}^c_{n+1}\}$ to equal the null tensor, so $\mathbf{L}_{n+1}$ has to equal $(-\mathbf{L}^c_{n+1})$. Comparing eq.(5.3.16) with (5.3.19) the following equality must hold:

$$\frac{1}{\Delta t}(\mathbf{F}_{n+1} - \mathbf{I}) \cdot \{(1-\alpha)\mathbf{I} + \alpha\mathbf{F}_{n+1} \}^{-1} = -[-\frac{1}{\Delta t}(\mathbf{F}_{n+1} - \mathbf{I}) \cdot \{(1-\alpha)\mathbf{F}_{n+1} + \alpha\mathbf{I}\}^{-1}] \tag{5.3.20}$$

This equality holds when:

$$\{ (1-\alpha)\,\mathbf{I} + \alpha\,\mathbf{F}_{n+1} \}^{-1} = \{ (1-\alpha)\,\mathbf{F}_{n+1} + \alpha\,\mathbf{I}\}^{-1} \tag{5.3.21}$$

So:

$$\{ (1-\alpha)\,\mathbf{I} + \alpha\,\mathbf{F}_{n+1} \} = \{ (1-\alpha)\,\mathbf{F}_{n+1} + \alpha\,\mathbf{I}\}$$

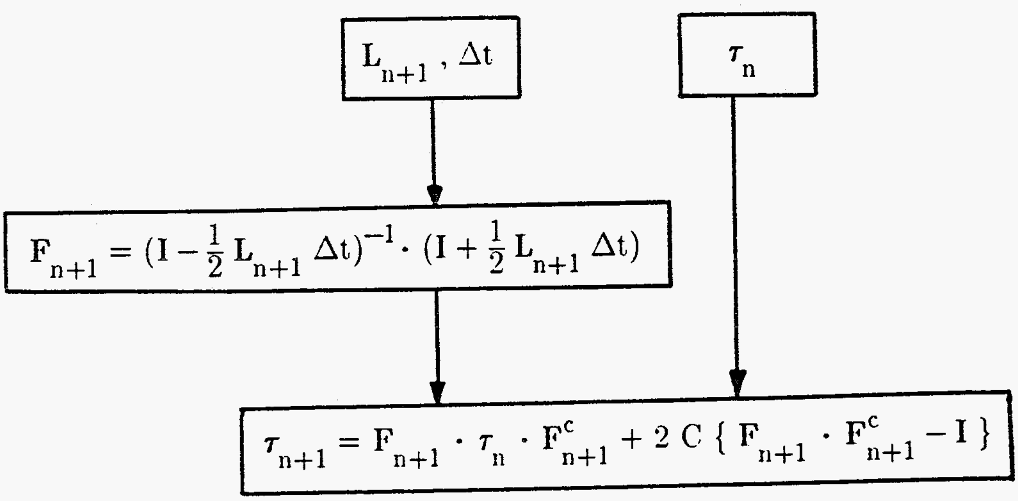
$$\Leftrightarrow \qquad (1-2\alpha)\,\mathbf{I} + (1-2\alpha)\,\mathbf{F}_{n+1} = \mathbf{O} \tag{5.3.22}$$

Equation (5.3.22) must be valid for all orthogonal tensors $\mathbf{F}_{n+1}$. This is only possible when $(1-2\alpha)$ equals to zero. So interpolation parameter α has to be chosen equal to a half. ■

Conclusion

Condition A is satisfied for all values of $\alpha \in [0,1]$. Condition B is satisfied only when α is chosen equal to $\frac{1}{2}$. So the proposed algorithm is incrementally objective if and only if the interpolation parameter α is equal to $\frac{1}{2}$.

For α equal to a half the algorithm can be presented in the following diagram:



CHAPTER 6: APPLICATIONS OF THE ALTERNATIVE ALGORITHM

6.1 Introduction

Two examples of applications of the alternative algorithm will be presented; a rigid body rotation and an isovolumetric elongation.

Both examples were also treated in chapter 4, which contained examples of the integration algorithm according to Pinsky c.s. All calculations in that chapter were performed in an incremental way and during these calculations the deformation tensor was kept constant in all increments.

The calculations performed in this chapter will be carried out with a velocity gradient which is constant in time because of the nature of the algorithm. The end configurations of both examples correspond with the end configurations of the comparable calculations in chapter 4.

The third example in chapter 4 contained a mixed deformation pattern, consisting of simple shear, isovolumetric elongation and rigid body rotation. The end configuration was reached with a constant incremental deformation tensor. It was not possible to calculate a theoretical velocity gradient, constant in time, which resulted in the same end configuration. That is why this example is omitted in this chapter.

Beside the two examples presented in this chapter some calculations were performed on simple shear with a constant velocity gradient. It turned out that this simple shear deformation is a quite trivial example; for all values of the interpolation parameter and the number of increments the same, correct, results were found. So in this chapter only a rigid body rotation and an isovolumetric elongation will be treated.

6.2 Rigid Body Rotation

A two–dimensional rigid body is rotated over $\phi = \frac{2}{3}\pi$ rad. The end configuration is reached after T = 5 s. The initial stress tensor is chosen equal to the null tensor and the material parameter C is equal to 1. Theoretically the deformation tensor $\mathbf{F}(t)$ equals:

$$\begin{aligned}\mathbf{F}(t) &= \cos(\varphi(t))\,\vec{e}_1\vec{e}_1 - \sin(\varphi(t))\,\vec{e}_1\vec{e}_2 + \\ &+ \sin(\varphi(t))\,\vec{e}_2\vec{e}_1 + \cos(\varphi(t))\,\vec{e}_2\vec{e}_2 \qquad (6.2.1)\end{aligned}$$

At t=0 rotation angle $\varphi(t)$ equals to zero. At t=T(=5 s.) $\varphi(t)$ has to equal $\phi(=\frac{2}{3}\pi$ rad.). For the velocity gradient the following equality is valid:

$$\mathbf{L}(t) = \dot{\mathbf{F}}(t) \cdot \mathbf{F}^{-1} = \dot{\varphi}(t)\ \{\vec{e}_2\vec{e}_1 - \vec{e}_1\vec{e}_2\} \qquad (6.2.2)$$

As expected the velocity gradient is skew–symmetric resulting in a rate of defromation tensor equal to the null tensor.

The incremental velocity gradient $\mathbf{L}_{n+1}(t)$ is calculated by equation (5.2.8):

$$\begin{aligned}\mathbf{L}_{n+1}(t) &= \frac{1}{\Delta t}\int_{t_n}^{t+\Delta t} \mathbf{L}(\tau)d\tau = \\ &= \frac{1}{\Delta t}\left[\ \varphi(t)\ \{\vec{e}_2\vec{e}_1 - \vec{e}_1\vec{e}_2\}\ \right]\Big|_t^{t+\Delta t} \\ &= \frac{\varphi(t+\Delta t)\ -\ \varphi(t)}{\Delta t}\ \{\vec{e}_2\vec{e}_1 - \vec{e}_1\vec{e}_2\} \qquad (6.2.3)\end{aligned}$$

The incremental velocity gradient is constant in time when $\varphi(t)$ is chosen equal to:

$$\varphi(t) = \phi\ t/T \qquad (6.2.4)$$

Then for $\mathbf{L}_{n+1}$ holds:

$$\mathbf{L}_{n+1} = \phi/T\ \{\vec{e}_2\vec{e}_1 - \vec{e}_1\vec{e}_2\} \qquad (6.2.5)$$

No stresses may be caused by the rigid body rotation. In appendix F the PC–Matlab file AROTAT.M is included, containing the alternative algorithm applied to the rigid body rotation. As in chapter 4 the number of increments and the value of the interpolation parameter are varied. The results are presented in figure 6.1. The scales in this figure fully agree with the scales in figure 4.2.

There is no significant difference between the results obtained with the integration algorithm and the results obtained with the alternative algorithm. For α equal to a half again the best results are found. Even using 480 increments stresses unequal to zero were found for α unequal to a half. This is due to the incremental objectivity which can only be obtained when α is chosen equal to a half.

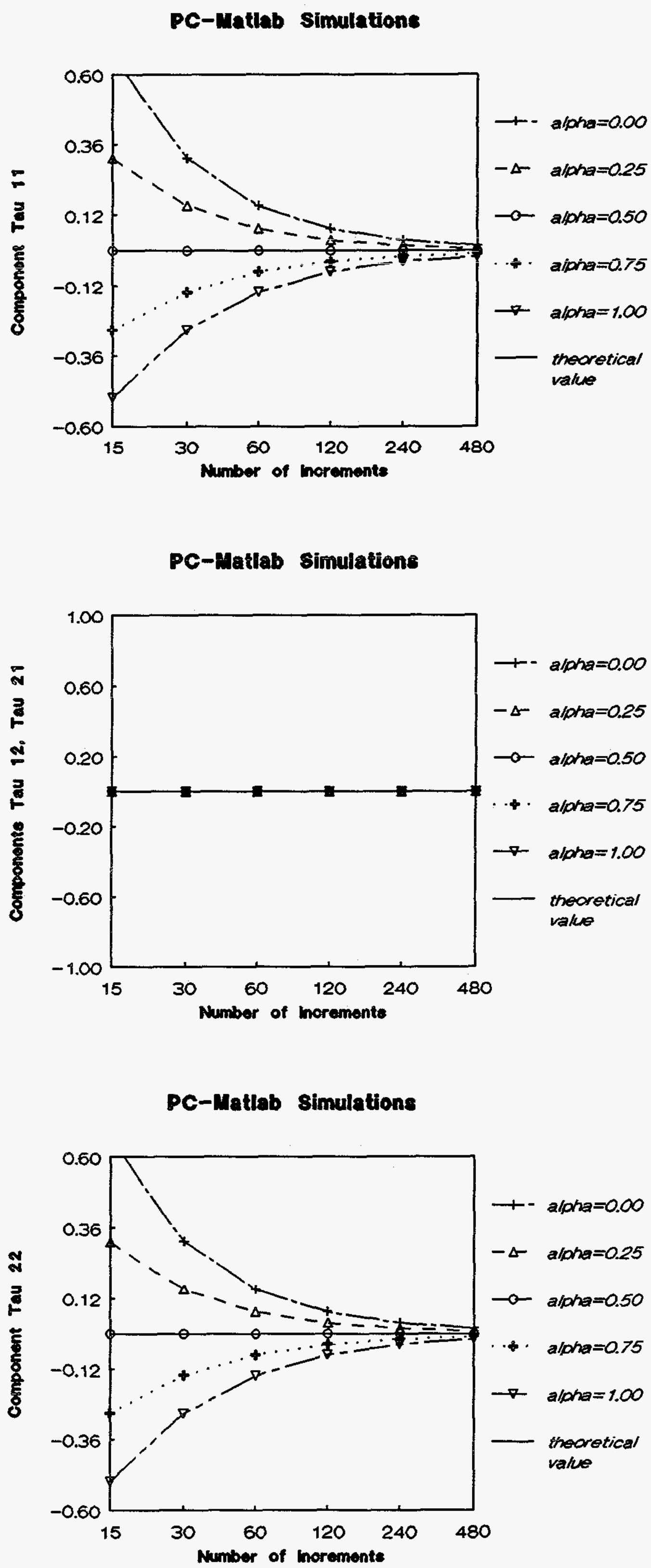


fig.6.1: results of the rigid body rotation

6.3 Isovolumetric Elongation

A body is isovolumetrically elongated along the $\vec{e}_1$–axis. In this $\vec{e}_1$–direction the total elongation factor Λ is 1.3 . The end configuration is reached after T = 5 s. There are no initial stresses and the material parameter is chosen equal to 1.0.

The deformation tensor equals:

$$\mathbf{F}(t) = \lambda(t)\, \vec{e}_1\vec{e}_1 + \frac{1}{\lambda(t)}\, \vec{e}_2\vec{e}_2 \tag{6.3.1}$$

where $\lambda(t)$ is the time dependent elongation factor, equal to 1 at t=0 and equal to Λ=1.3 at t=T(=5 s.).

For the velocity gradient $\mathbf{L}(t)$ holds:

$$\mathbf{L}(t) = \dot{\mathbf{F}}(t) \cdot \mathbf{F}^{-1} = \frac{\dot{\lambda}(t)}{\lambda(t)} \{\vec{e}_1\vec{e}_1 - \vec{e}_2\vec{e}_2\} \tag{6.3.2}$$

The incremental velocity gradient $\mathbf{L}_{n+1}(t)$ is calculated by:

$$\begin{aligned} \mathbf{L}_{n+1}(t) &= \frac{1}{\Delta t} \int_{t_n}^{t+\Delta t} \mathbf{L}(\tau) d\tau = \\ &= \frac{1}{\Delta t} \left[\ln(\lambda(t))\, \{\vec{e}_1\vec{e}_1 - \vec{e}_2\vec{e}_2\} \right] \Big|_{t}^{t+\Delta t} \\ &= \frac{1}{\Delta t} \ln\left[\frac{\lambda(t+\Delta t)}{\lambda(t)}\right] \{\vec{e}_1\vec{e}_1 - \vec{e}_2\vec{e}_2\} \end{aligned} \tag{6.3.3}$$

Tensor $\mathbf{L}_{n+1}$ is constant in time when the time–dependent elongation factor $\lambda(t)$ is chosen equal to:

$$\lambda(t) = \Lambda^{(t/T)} \tag{6.3.4}$$

Then:

$$\begin{aligned} \mathbf{L}_{n+1}(t) &= \frac{1}{\Delta t} \ln\left[\Lambda^{(t/T)}\right] \{\vec{e}_1\vec{e}_1 - \vec{e}_2\vec{e}_2\} \\ &= \frac{1}{T} (\ln(\Lambda))\, \{\vec{e}_1\vec{e}_1 - \vec{e}_2\vec{e}_2\} \end{aligned} \tag{6.3.5}$$

This velocity gradient is programmed in the file ASTRET.M, included in appendix F. Both the interpolation parameter α and the number of increments N are varied. In chapter 4 it was found that in the end configuration the stress matrix $\underline{\tau}$ theoretically has to be equal to:

$$\underline{\tau}_{\mathrm{T}} = \begin{bmatrix} 1.3800 & 0 \\ 0 & -0.8166 \end{bmatrix} \tag{6.3.6}$$

The results of the calculation performed with the alternative algorithm are presented in figure 6.2. Again the scales in this figure agree with the scales used in figure 4.3.

The results obtained with the alternative algorithm are better than the results of the "Pinsky"–algorithm; in figure 6.2 the absolute errors in the stress components τ_{11} and τ_{22} are less than half of the errors in figure 4.3 (for α unequal to a half). Just as in the previous paragraph for $\alpha = \frac{1}{2}$ the best results are found. As in chapter 4 this the due to the accuracy of the algorithm, being higher for α equal to a half.

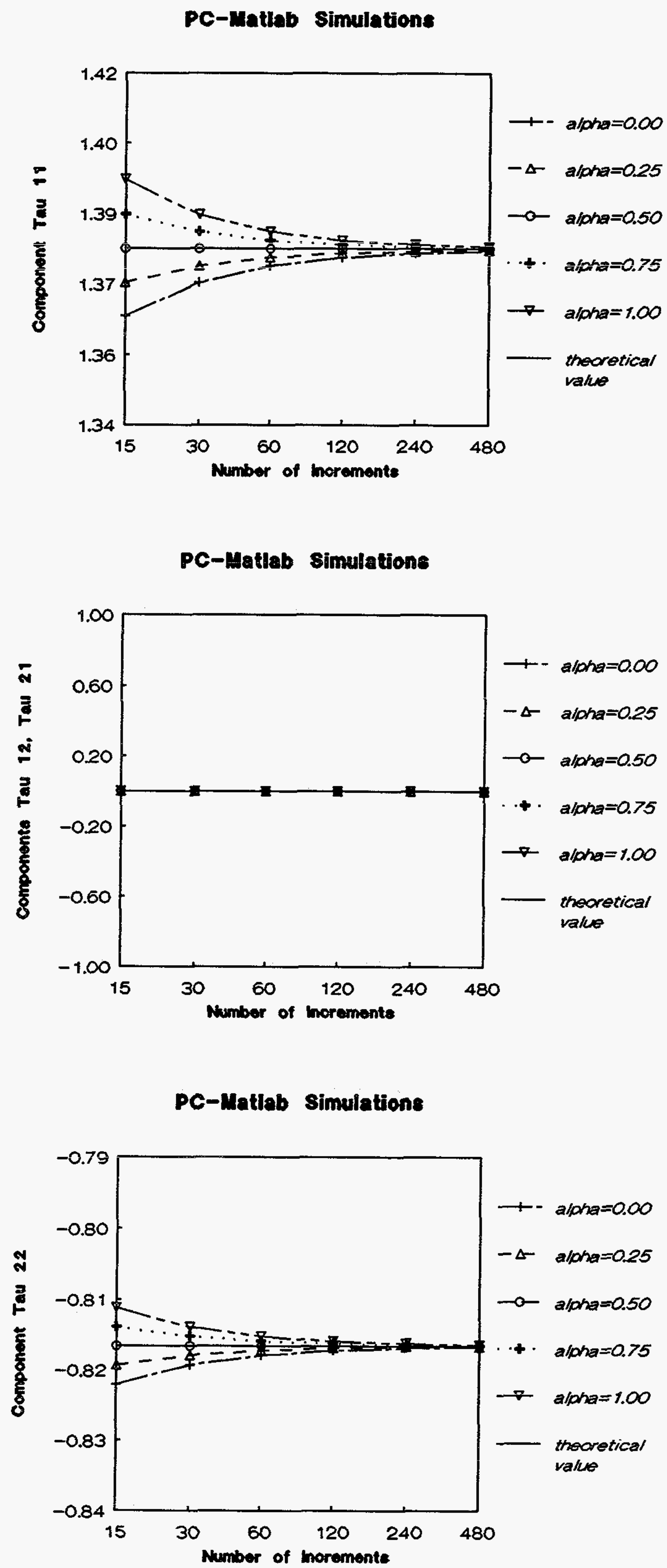


fig.6.2: results of the isovolumetric elongation

Conclusions

Both proposed algorithms for calculating stress tensor τ of the (modified) Neo–Hookean constitutive equation showed correct results for several applications. The value of the interpolation parameter α turned out to be of great importance. Extreme improvements of the results have been found when choosing this parameter equal to a half.

For $\alpha=\frac{1}{2}$ it can be proven that both algorithms are incrementally objective, i.e. insensitive for rigid body rotations. Even if no rigid body occurs we have found the algorithms to show best results for $\alpha=\frac{1}{2}$. The algorithms are not only incrementally objective for this value of α, but are also more accurate. So it is strongly recommended to use $\alpha=\frac{1}{2}$.

The definition of incremental objectivity does not agree with the widely accepted formal definition of objectivity; the integration algorithms, though incrementally objective for $\alpha=\frac{1}{2}$, are not objective in the general sense for this value of the interpolation parameter.

P.A.A. van Hoogstraten

Eindhoven, May 1989

Philips Research Laboratories

Nederlandse Philips Bedrijven B.V.

Eindhoven University of Technology

Department of Mechanical Engineering

APPENDIX A: EVALUATION OF TENSORS $(\mathbf{F}_{n+1} \cdot \mathbf{F}^{-1}_{n+\alpha})$ AND $\mathbf{D}_{n+\alpha}$

A.1: Approximation of $(\mathbf{F}_{n+1} \cdot \mathbf{F}^{-1}_{n+\alpha})$

For tensor $(\mathbf{F}_{n+1} \cdot \mathbf{F}^{-1}_{n+\alpha})$ the following expressions can be deduced using the linear interpolation of the deformation tensor (3.2.7):

$$\mathbf{F}_{n+\alpha} = (1-\alpha)\,\mathbf{I} + \alpha\,\mathbf{F}_{n+1} \tag{3.2.7}$$

$$\begin{aligned} \Rightarrow \mathbf{F}_{n+1} \cdot \mathbf{F}^{-1}_{n+\alpha} &= [\,\mathbf{F}_{n+\alpha} \cdot \mathbf{F}^{-1}_{n+1}\,]^{-1} \\ &= [\,\{(1-\alpha)\,\mathbf{I} + \alpha\,\mathbf{F}_{n+1}\} \cdot \mathbf{F}^{-1}_{n+1}\,]^{-1} \\ &= [\,(1-\alpha)\,\mathbf{F}^{-1}_{n+1} + \alpha\,\mathbf{I}\,]^{-1} \end{aligned} \tag{A.1.1}$$

Formula (A.1.1) can easily be transformed in:

$$\mathbf{F}_{n+1} \cdot \mathbf{F}^{-1}_{n+\alpha} = \mathbf{F}_{n+1} \cdot [\,(1-\alpha)\,\mathbf{I} + \alpha\,\mathbf{F}_{n+1}\,]^{-1} \tag{A.1.2}$$

$$= [\,(1-\alpha)\,\mathbf{I} + \alpha\,\mathbf{F}_{n+1}\,]^{-1} \cdot \mathbf{F}_{n+1} \tag{A.1.3}$$

using:

$$\mathbf{A} \cdot (\mathbf{A}+\mathbf{B})^{-1} \cdot \mathbf{B} = \mathbf{B} \cdot (\mathbf{A}+\mathbf{B})^{-1} \cdot \mathbf{A} = (\mathbf{A}^{-1}+\mathbf{B}^{-1})^{-1} \tag{A.1.4}$$

A.2: Approximation of $\mathbf{D}_{n+\alpha}$

In general the rate of strain tensor **D** is defined as:

$$\mathbf{D} = \frac{1}{2}\left(\mathbf{L} + \mathbf{L}^c \right) \quad \text{with: } \mathbf{L} = (\vec{\nabla}\vec{v})^c \tag{A.2.1}$$

To evaluate tensor $\mathbf{D}_{n+\alpha}$ we will first evaluate tensor $\mathbf{L}_{n+\alpha}$:

$$\mathbf{L}_{n+\alpha} = (\vec{\nabla}_{n+\alpha}\vec{v}_{n+\alpha})^c = (\vec{\nabla}_{n+\alpha}\dot{\vec{x}}_{n+\alpha})^c \tag{A.2.2}$$

Choosing a linear interpolation for $\vec{x}_{n+\alpha}$ the velocity field $\vec{v}_{n+\alpha}$ can be approximated by:

$$\vec{v}_{n+\alpha} = \dot{\vec{x}}_{n+\alpha} \approx \frac{\vec{x}_{n+1} - \vec{x}_n}{h_n} \tag{A.2.3}$$

So:

$$\mathbf{L}_{n+\alpha} = \frac{1}{h_n}\left[\vec{\nabla}_{n+\alpha}\vec{x}_{n+1} - \vec{\nabla}_{n+\alpha}\vec{x}_n \right]^c \tag{A.2.4}$$

First gradient $\vec{\nabla}_{n+\alpha}$ is considered. This gradient can be expressed in the gradient with respect to the reference configuration and the deformation tensor $\mathbf{F}_{n+\alpha}$:

$$\vec{\nabla}_{n+\alpha} = \mathbf{F}^{-c}_{n+\alpha} \cdot \vec{\nabla}_n \tag{A.2.5}$$

proof:

$$\mathbf{F}_{n+\alpha} = (\vec{\nabla}_n\vec{x}_{n+\alpha})^c \quad \Leftrightarrow \quad \mathbf{F}^c_{n+\alpha} = \vec{\nabla}_n\vec{x}_{n+\alpha}$$

$$\mathbf{I} = \mathbf{F}^{-c}_{n+\alpha} \cdot \mathbf{F}^c_{n+\alpha} = \mathbf{F}^{-c}_{n+\alpha} \cdot \vec{\nabla}_n\vec{x}_{n+\alpha}$$

but also: $\mathbf{I} = \mathbf{I}^c = \vec{\nabla}_{n+\alpha}\vec{x}_{n+\alpha}$

so: $\vec{\nabla}_{n+\alpha} = \mathbf{F}^{-c}_{n+\alpha} \cdot \vec{\nabla}_n$ ∎

Applying (A.2.5) to equation (A.2.4) and using the definition of the deformation tensor $\mathbf{F}_{n+1}$ results in:

$$\mathbf{L}_{n+\alpha} = \frac{1}{h_n}\left\{ \mathbf{F}_{n+\alpha}^{-c} \cdot [\vec{\nabla}_n \vec{x}_{n+1} - \vec{\nabla}_n \vec{x}_n]\right\}^c$$
$$= \frac{1}{h_n}\left\{ [\, \mathbf{F}_{n+1} - \mathbf{I}\,] \cdot \mathbf{F}_{n+\alpha}^{-1} \right\} \tag{A.2.6}$$

Since tensor $\mathbf{F}_{n+\alpha}$ is linearly interpolated (see equation (3.2.6)) the approximation of tensor $\mathbf{L}_{n+\alpha}$ can be written as:

$$\mathbf{L}_{n+\alpha} = \frac{1}{h_n}\left\{ [\mathbf{F}_{n+1} - \mathbf{I}] \cdot [(1-\alpha)\,\mathbf{I} + \alpha\,\mathbf{F}_{n+1}]^{-1} \right\} \tag{A.2.7}$$

And the resulting approximation for $\mathbf{D}_{n+\alpha}$ is: (A.2.8)

$$\mathbf{D}_{n+\alpha} = \frac{1}{2h_n}\left\{[\mathbf{F}_{n+1} - \mathbf{I}] \cdot [(1-\alpha)\,\mathbf{I} + \alpha\,\mathbf{F}_{n+1}]^{-1} + [(1-\alpha)\,\mathbf{I} + \alpha\,\mathbf{F}_{n+1}]^{-c} \cdot [\mathbf{F}_{n+1} - \mathbf{I}]^c\right\}$$

Appendix A.4

APPENDIX B: EVALUATION OF TENSOR $\delta\mathbf{H}_{n+\alpha}$ FOR SMALL DEFORMATIONS

Tensor $\delta\mathbf{H}_{n+\alpha}$ is defined as:

$$\delta\mathbf{H}_{n+\alpha} = 4C\,h_n \left\{ \mathbf{F}_{n+1} \cdot \mathbf{F}^{-1}_{n+\alpha} \cdot \mathbf{D}_{n+\alpha} \cdot \mathbf{F}^{-c}_{n+\alpha} \cdot \mathbf{F}^{c}_{n+1} - \mathbf{D}_{n+1} \right\}$$

In this appendix $\delta\mathbf{H}_{n+\alpha}$ will be evaluated for small deformations. A deformation is called small when no large rotations or strains occur. Then its deformation tensor can be written as:

$$\mathbf{F}_{n+1} = \mathbf{I} + \boldsymbol{\epsilon}_{n+1} \quad \text{with } \|\boldsymbol{\epsilon}_{n+1}\| << 1 \tag{B.1}$$

For $\mathbf{F}_{n+\alpha}$ holds:

$$\mathbf{F}_{n+\alpha} = \mathbf{I} + \boldsymbol{\epsilon}_{n+\alpha} = \mathbf{I} + \alpha\,\boldsymbol{\epsilon}_{n+1} \tag{B.2}$$

The inverse tensors $\mathbf{F}^{-1}_{n+1}$ and $\mathbf{F}^{-1}_{n+\alpha}$ can be approximated using a Taylor series:

$$\mathbf{F}^{-1}_{n+1} = (\mathbf{I} + \boldsymbol{\epsilon}_{n+1})^{-1} \approx \mathbf{I} - \boldsymbol{\epsilon}_{n+1} \tag{B.3}$$

$$\mathbf{F}^{-1}_{n+\alpha} = (\mathbf{I} + \boldsymbol{\epsilon}_{n+\alpha})^{-1} \approx \mathbf{I} - \alpha\,\boldsymbol{\epsilon}_{n+1} \tag{B.4}$$

In appendix A an approximation for $\mathbf{D}_{n+\alpha}$ is derived:

$$\mathbf{D}_{n+\alpha} = \frac{1}{2}\left(\mathbf{L}_{n+\alpha} + \mathbf{L}^{c}_{n+\alpha} \right)$$

with:
$$\mathbf{L}_{n+\alpha} = \frac{1}{h_n}\left\{ [\mathbf{F}_{n+1} - \mathbf{I}] \cdot [(1-\alpha)\,\mathbf{I} + \alpha\,\mathbf{F}_{n+1}]^{-1} \right\}$$

Subtitution of (B.1) in the expression for $\mathbf{L}_{n+\alpha}$ and neglection of the higher order term $(\epsilon_{n+1} \cdot \epsilon_{n+1})$ gives:

$$
\begin{aligned}
\mathbf{L}_{n+\alpha} &\approx \frac{1}{h_n} \left\{ [(\mathbf{I} + \epsilon_{n+1}) - \mathbf{I}] \cdot [(1-\alpha)\, \mathbf{I} + \alpha\, (\mathbf{I} + \epsilon_{n+1})]^{-1} \right\} \\
&= \frac{1}{h_n} \left\{ [\, \epsilon_{n+1} \,] \cdot [\, \mathbf{I} + \alpha\, \epsilon_{n+1})]^{-1} \right\} \\
&\approx \frac{1}{h_n} \left\{ [\, \epsilon_{n+1} \,] \cdot [\, \mathbf{I} - \alpha\, \epsilon_{n+1})] \right\} \\
&= \frac{1}{h_n} \left\{ \epsilon_{n+1} - \alpha\, \epsilon_{n+1} \cdot \epsilon_{n+1} \right\} \\
&\approx \frac{1}{h_n}\, \epsilon_{n+1}
\end{aligned}
\tag{B.5}
$$

And the rate of strain tensor $\mathbf{D}_{n+\alpha}$ results in:

$$
\mathbf{D}_{n+\alpha} \approx \frac{1}{2h_n} (\, \epsilon_{n+1} + \epsilon^{c}_{n+1}) \tag{B.6}
$$

Using equation (B.1), (B.4) and (B.6) the following approximation can be made:

$$
\mathbf{F}_{n+1} \cdot \mathbf{F}^{-1}_{n+\alpha} \cdot \mathbf{D}_{n+\alpha} \cdot (\mathbf{F}_{n+1} \cdot \mathbf{F}^{-1}_{n+\alpha})^{c} \approx
$$

$$
\begin{aligned}
&\approx (\mathbf{I}+\epsilon_{n+1}) \cdot (\mathbf{I}-\alpha\epsilon_{n+1}) \cdot \frac{1}{2h_n} (\epsilon_{n+1}+\epsilon^{c}_{n+1}) \cdot \{(\mathbf{I}+\epsilon_{n+1}) \cdot (\mathbf{I}-\alpha\epsilon_{n+1})\}^{c} \\
&\approx (\mathbf{I}+\epsilon_{n+1}-\alpha\epsilon_{n+1}-\alpha\epsilon^{2}_{n+1}) \cdot \frac{1}{2h_n} (\epsilon_{n+1}+\epsilon^{c}_{n+1}) \cdot (\mathbf{I}+\epsilon^{c}_{n+1}-\alpha\epsilon^{c}_{n+1}-\alpha\epsilon^{c}_{n+1} \cdot \epsilon^{c}_{n+1}) \\
&\approx \frac{1}{2h_n} (\, \epsilon_{n+1} + \epsilon^{c}_{n+1})
\end{aligned}
\tag{B.7}
$$

The latter form is found by neglecting higher order terms of ϵ_{n+1} which is admissible if the norm of ϵ_{n+1} is very small.

Rate of strain tensor $\mathbf{D}_{n+1}$ is approximated by:

$$\mathbf{D}_{n+1} = \frac{1}{2} (\mathbf{L}_{n+1} + \mathbf{L}^c_{n+1})$$

with:

$$\begin{aligned} \mathbf{L}_{n+1} &= \frac{1}{h_n} \left\{ [\mathbf{F}_{n+1} - \mathbf{I}] \cdot \mathbf{F}^{-1}_{n+1} \right\} \\ &= \frac{1}{h_n} \left\{ \mathbf{I} - \mathbf{F}^{-1}_{n+1} \right\} \end{aligned}$$

For small deformations $\mathbf{L}_{n+1}$ equals:

$$\mathbf{L}_{n+1} \approx \frac{1}{h_n} \left\{ \mathbf{I} - (\mathbf{I} - \boldsymbol{\epsilon}_{n+1}) \right\} = \frac{1}{h_n} \boldsymbol{\epsilon}_{n+1}$$

So $\mathbf{D}_{n+1}$ is approximated by:

$$\mathbf{D}_{n+1} \approx \frac{1}{2h_n} (\boldsymbol{\epsilon}_{n+1} + \boldsymbol{\epsilon}^c_{n+1}) \tag{B.8}$$

Substitution of approximation (B.7) and (B.8) in the definition of $\delta\mathbf{H}_{n+\alpha}$ gives:

$$\delta\mathbf{H}_{n+\alpha} \approx \frac{1}{2h_n} (\boldsymbol{\epsilon}_{n+1} + \boldsymbol{\epsilon}^c_{n+1}) - \frac{1}{2h_n} (\boldsymbol{\epsilon}_{n+1} + \boldsymbol{\epsilon}^c_{n+1}) = \mathbf{O} \tag{B.9}$$

conclusion:

For small deformations tensor $\delta\mathbf{H}_{n+\alpha}$ tends to the null tensor.

Appendix B.4

APPENDIX C: NOTES ON THE OBJECTIVITY OF THE INTEGRATION ALGORITHM

The algorithm for $\alpha=\frac{1}{2}$:

For $\alpha=\frac{1}{2}$ the integration algorithm in the updated approach is:

$$\tau_{n+1} = \mathbf{F}_{n+1} \cdot \tau_n \cdot \mathbf{F}^c_{n+1} + 4C\, h_n\, \mathbf{F}_{n+1} \cdot \mathbf{F}^{-1}_{n+1/2} \cdot \mathbf{D}_{n+1/2} \cdot \mathbf{F}^{-c}_{n+1/2} \cdot \mathbf{F}^c_{n+1} \tag{C.1}$$

$$\mathbf{F}_{n+1/2} = \frac{1}{2}\left[\,\mathbf{I} + \mathbf{F}_{n+1}\,\right];\quad \mathbf{F}^{-1}_{n+1/2} = 2\left[\,\mathbf{I} + \mathbf{F}_{n+1}\,\right]^{-1} \tag{C.2}$$

$$\mathbf{D}_{n+1/2} = \frac{1}{2}\left[\,\mathbf{L}_{n+1/2} + \mathbf{L}^c_{n+1/2}\,\right] \tag{C.3}$$

$$\begin{aligned} \mathbf{L}_{n+1/2} &= \frac{1}{h_n}\left[(\mathbf{F}_{n+1}-\mathbf{I}) \cdot \mathbf{F}^{-1}_{n+1/2}\right] \\ &= \frac{1}{h_n}\left[(\mathbf{F}_{n+1}-\mathbf{I}) \cdot 2\left[\,\mathbf{I} + \mathbf{F}_{n+1}\,\right]^{-1}\right] \\ &= \frac{2}{h_n}\left[\,\mathbf{F}_{n+1} \cdot \left[\,\mathbf{I} + \mathbf{F}_{n+1}\,\right]^{-1} - \left[\,\mathbf{I} + \mathbf{F}_{n+1}\,\right]^{-1}\right] \\ &= \frac{2}{h_n}\left[(\mathbf{I} + \mathbf{F}^{-1}_{n+1})^{-1} - (\mathbf{I} + \mathbf{F}_{n+1})^{-1}\right] \end{aligned} \tag{C.4}$$

Substitution of (C.2), (C.3) and (C.4) in (C.1) results in:

$$\begin{aligned} \tau_{n+1} &= \mathbf{F}_{n+1} \cdot \tau_n \cdot \mathbf{F}^c_{n+1} + \\ &+ 16\, C\, \mathbf{F}_{n+1} \cdot (\mathbf{I}+\mathbf{F}_{n+1})^{-1} \cdot \Big[\; (\mathbf{I}+\mathbf{F}^{-1}_{n+1})^{-1} - (\mathbf{I}+\mathbf{F}_{n+1})^{-1} + \\ &\qquad (\mathbf{I}+\mathbf{F}^{-c}_{n+1})^{-1} - (\mathbf{I}+\mathbf{F}^{c}_{n+1})^{-1} \;\Big] \cdot (\mathbf{I}+\mathbf{F}^c_{n+1})^{-1} \cdot \mathbf{F}^c_{n+1} \end{aligned} \tag{C.5}$$

The products before and after the square brackets can be rewritten using:

$$\mathbf{A} \cdot (\mathbf{A}+\mathbf{B})^{-1} \cdot \mathbf{B} = (\mathbf{A}^{-1}+\mathbf{B}^{-1})^{-1} \tag{C.6}$$

for all regular tensors **A** and **B**

Thus:

$$\tau_{n+1} = \mathbf{F}_{n+1} \cdot \tau_n \cdot \mathbf{F}^c_{n+1} + \tag{C.7}$$
$$+ 16\,C\,(\mathbf{I}+\mathbf{F}^{-1}_{n+1})^{-1} \cdot \Big[(\mathbf{I}+\mathbf{F}^{-1}_{n+1})^{-1} - (\mathbf{I}+\mathbf{F}_{n+1})^{-1} +$$
$$(\mathbf{I}+\mathbf{F}^{-c}_{n+1})^{-1} - (\mathbf{I}+\mathbf{F}^{c}_{n+1})^{-1} \Big] \cdot (\mathbf{I}+\mathbf{F}^{-c}_{n+1})^{-1}$$

Application of (C.6) to the second and fourth term in between the brackets and multiple use of $(\mathbf{A} \cdot \mathbf{B})^c = \mathbf{B}^c \cdot \mathbf{A}^c$ gives:

$$\tau_{n+1} = \mathbf{F}_{n+1} \cdot \tau_n \cdot \mathbf{F}^c_{n+1} + \tag{C.8}$$
$$+ 16\,C \left\{ (\mathbf{I}+\mathbf{F}^{-1}_{n+1})^{-1} \cdot (\mathbf{I}+\mathbf{F}^{-1}_{n+1})^{-1} \cdot (\mathbf{I}-\mathbf{F}^{-1}_{n+1}) \cdot (\mathbf{I}+\mathbf{F}^{-c}_{n+1})^{-1} \right\}$$
$$+ 16\,C \left\{ (\mathbf{I}+\mathbf{F}^{-1}_{n+1})^{-1} \cdot (\mathbf{I}+\mathbf{F}^{-1}_{n+1})^{-1} \cdot (\mathbf{I}-\mathbf{F}^{-1}_{n+1}) \cdot (\mathbf{I}+\mathbf{F}^{-c}_{n+1})^{-1} \right\}^C$$

So:

$$\tau_{n+1} = \mathbf{F}_{n+1} \cdot \tau_n \cdot \mathbf{F}^c_{n+1} + 16\,C\,(\mathbf{W} + \mathbf{W}^c) \tag{C.9}$$
$$\text{with:} \quad \mathbf{W} = \left\{ (\mathbf{I}+\mathbf{F}^{-1}_{n+1})^{-1} \cdot (\mathbf{I}+\mathbf{F}^{-1}_{n+1})^{-1} \cdot (\mathbf{I}-\mathbf{F}^{-1}_{n+1}) \cdot (\mathbf{I}+\mathbf{F}^{-c}_{n+1})^{-1} \right\}$$

<u>Calculation of $\mathbf{Q} \cdot \mathbf{N}(\mathbf{F}_{n+1}) \cdot \mathbf{Q}^c$</u>

Tensor $\mathbf{Q} \cdot \mathbf{N}(\mathbf{F}_{n+1}) \cdot \mathbf{Q}^c$ equals:

$$\mathbf{Q} \cdot \mathbf{N}(\mathbf{F}_{n+1}) \cdot \mathbf{Q}^c = \mathbf{Q} \cdot \mathbf{F}_{n+1} \cdot \tau_n \cdot \mathbf{F}^c_{n+1} \cdot \mathbf{Q}^c + 16\,C\,\mathbf{Q} \cdot [\mathbf{W} + \mathbf{W}^c] \cdot \mathbf{Q}^c \tag{C.10}$$

where $\mathbf{Q}$ is an arbitrary orthogonal rigid body rotation tensor and $\mathbf{W}$ is given by (C.9)

Calculation of $\mathbf{N}(\mathbf{Q}\cdot\mathbf{F}_{n+1})$

Substitution of $(\mathbf{Q}\cdot\mathbf{F}_{n+1})$ for $\mathbf{F}_{n+1}$ in (C.9) gives:

$$\mathbf{N}(\mathbf{Q}\cdot\mathbf{F}_{n+1}) = \mathbf{Q}\cdot\mathbf{F}_{n+1}\cdot\tau_n\cdot\mathbf{F}^c_{n+1}\cdot\mathbf{Q}^c + 16\,C\,[\mathbf{X}+\mathbf{X}^c] \tag{C.11}$$

$$\text{with:}\quad \mathbf{X} = \left\{(\mathbf{I}+\mathbf{F}^{-1}_{n+1}\cdot\mathbf{Q}^c)^{-1}\cdot(\mathbf{I}+\mathbf{F}^{-1}_{n+1}\cdot\mathbf{Q}^c)^{-1}\cdot(\mathbf{I}-\mathbf{F}^{-1}_{n+1}\cdot\mathbf{Q}^c)\cdot(\mathbf{I}+\mathbf{Q}\cdot\mathbf{F}^{-c}_{n+1})^{-1}\right\}$$

$$= (\{\mathbf{Q}+\mathbf{F}^{-1}_{n+1}\}\cdot\mathbf{Q}^c)^{-1}\cdot(\mathbf{I}+\mathbf{F}^{-1}_{n+1}\cdot\mathbf{Q}^c)^{-1}\cdot(\mathbf{I}-\mathbf{F}^{-1}_{n+1}\cdot\mathbf{Q}^c)\cdot(\mathbf{Q}\cdot\{\mathbf{Q}^c+\mathbf{F}^{-c}_{n+1}\})^{-1}$$

$$= \mathbf{Q}\cdot(\mathbf{Q}+\mathbf{F}^{-1}_{n+1})^{-1}\cdot(\mathbf{I}+\mathbf{F}^{-1}_{n+1}\cdot\mathbf{Q}^c)^{-1}\cdot(\mathbf{I}-\mathbf{F}^{-1}_{n+1}\cdot\mathbf{Q}^c)\cdot(\mathbf{Q}^c+\mathbf{F}^{-c}_{n+1})^{-1}\cdot\mathbf{Q}^c$$

Thus:

$$\mathbf{N}(\mathbf{Q}\cdot\mathbf{F}_{n+1}) = \mathbf{Q}\cdot\mathbf{F}_{n+1}\cdot\tau_n\cdot\mathbf{F}^c_{n+1}\cdot\mathbf{Q}^c + 16\,C\,\mathbf{Q}\cdot[\mathbf{V}+\mathbf{V}^c]\cdot\mathbf{Q}^c \tag{C.12}$$

$$\text{with:}\quad \mathbf{V} = (\mathbf{Q}+\mathbf{F}^{-1}_{n+1})^{-1}\cdot(\mathbf{I}+\mathbf{F}^{-1}_{n+1}\cdot\mathbf{Q}^c)^{-1}\cdot(\mathbf{I}-\mathbf{F}^{-1}_{n+1}\cdot\mathbf{Q}^c)\cdot(\mathbf{Q}^c+\mathbf{F}^{-c}_{n+1})^{-1}$$

Objectivity for rigid body rotations

For all rigid body rotation tensors $\mathbf{F}_{n+1}=\mathbf{R}$ with $\mathbf{R}\cdot\mathbf{R}^c=\mathbf{I}$ the algorithm proofs to be objective:

For $\mathbf{F}_{n+1}=\mathbf{R}$ both tensors $(\mathbf{V}+\mathbf{V}^c)$ and $(\mathbf{W}+\mathbf{W}^c)$ equal the zero tensor $\mathbf{O}$, so tensor $\mathbf{N}(\mathbf{Q}\cdot\mathbf{R})$ equals tensor $\mathbf{Q}\cdot\mathbf{N}(\mathbf{R})\cdot\mathbf{Q}^c$.

proof:

$$\begin{aligned}\mathbf{W}+\mathbf{W}^c &= (\mathbf{I}+\mathbf{R}^c)^{-1}\cdot(\mathbf{I}+\mathbf{R}^c)^{-1}\cdot(\mathbf{I}-\mathbf{R}^c)\cdot(\mathbf{I}+\mathbf{R})^{-1} + (\mathbf{I}+\mathbf{R}^c)^{-1}\cdot(\mathbf{I}-\mathbf{R})\cdot(\mathbf{I}+\mathbf{R})^{-1}\cdot(\mathbf{I}+\mathbf{R})^{-1}\\
&= (\mathbf{I}+\mathbf{R}^c)^{-1}\cdot[(\mathbf{I}+\mathbf{R}^c)^{-1}-(\mathbf{I}+\mathbf{R}^c)^{-1}\cdot\mathbf{R}^c + (\mathbf{I}+\mathbf{R})^{-1}-\mathbf{R}\cdot(\mathbf{I}+\mathbf{R})^{-1}]\cdot(\mathbf{I}+\mathbf{R})^{-1}\\
&= (\mathbf{I}+\mathbf{R}^c)^{-1}\cdot[(\mathbf{I}+\mathbf{R}^c)^{-1}-(\mathbf{I}+\mathbf{R})^{-1} + (\mathbf{I}+\mathbf{R})^{-1}-(\mathbf{I}+\mathbf{R}^c)^{-1}]\cdot(\mathbf{I}+\mathbf{R})^{-1}\\
&= \mathbf{O}\end{aligned}$$

$$
\begin{aligned}
\mathbf{V}+\mathbf{V}^c &= (\mathbf{Q}+\mathbf{R}^c)^{-1}\cdot(\mathbf{I}+\mathbf{R}^c\cdot\mathbf{Q}^c)^{-1}\cdot(\mathbf{I}-\mathbf{R}^c\cdot\mathbf{Q}^c)\cdot(\mathbf{Q}^c+\mathbf{R})^{-1}+\\
&+ (\mathbf{Q}+\mathbf{R}^c)^{-1}\cdot(\mathbf{I}-\mathbf{Q}\cdot\mathbf{R})\cdot(\mathbf{I}+\mathbf{Q}\cdot\mathbf{R})^{-1}\cdot(\mathbf{Q}^c+\mathbf{R})^{-1}\\
&= (\mathbf{Q}+\mathbf{R}^c)^{-1}\cdot\Big[\ (\mathbf{I}+\mathbf{R}^c\cdot\mathbf{Q}^c)^{-1} - (\mathbf{I}+\mathbf{R}^c\cdot\mathbf{Q}^c)^{-1}\cdot\mathbf{R}^c\cdot\mathbf{Q}^c+\\
&\qquad (\mathbf{I}+\mathbf{Q}\cdot\mathbf{R})^{-1} - \mathbf{Q}\cdot\mathbf{R}\cdot(\mathbf{I}+\mathbf{Q}\cdot\mathbf{R})^{-1}\Big]\cdot(\mathbf{Q}^c+\mathbf{R})^{-1}
\end{aligned}
$$

$$
\begin{aligned}
&\mathbf{V}+\mathbf{V}^c =\\
&(\mathbf{Q}+\mathbf{R}^c)^{-1}\cdot\Big[(\mathbf{I}+\mathbf{R}^c\cdot\mathbf{Q}^c)^{-1}-(\mathbf{Q}\cdot\mathbf{R}+\mathbf{I})^{-1}+(\mathbf{I}+\mathbf{Q}\cdot\mathbf{R})^{-1}-(\mathbf{I}+\mathbf{R}^c\cdot\mathbf{Q}^c)^{-1}\Big]\cdot(\mathbf{Q}^c+\mathbf{R})^{-1}\\
&\qquad = \mathbf{O}
\end{aligned}
$$

■

APPENDIX D: PC–MATLAB PROGRAM FILE FOR CALCULATIONS ON THE OBJECTIVITY OF THE INTEGRATION ALGORITHM

FILE: OBJECTIVE.M

```
echo off;
%
%
%   Calculations on the objectivity of the integration algorithm
%   ==========================================================
%
%
%
%   Fn1        = F(n+1) ; deformation tensor in configuration (n+1)
%   Rn1        = rigid body rotation tensor          ( <— polar decomposition of Fn1)
%   Un1        = stretch tensor                      ( <— polar decomposition of Fn1)
%
%
%   Tn         = stress tensor in configuration (n)
%   Tn1        = stress tensor in configuration (n+1)
%
%   alpha      = interpolation parameter
%
%   C          = material parameter
%
%   Q          = extra rigid body rotation tensor
%
%   Fna        = F(n+alpha) ; deformation tensor in configuration (n+alpha)
%   Dna        = rate of strain tensor in configuration (n+alpha)
%
%------------------------------------------------------------------------
%
%   interpolation parameter
%
      alpha = 0.5;
%
%   material parameter
%
      C = 1.0;
%
%   unity tensor
%
      I = eye(2);
%
%
%   stress tensor in configuration n
%
      Tn = [          1.5       0
                      0         0.5 ];
```

```
%   deformation tensor in configuration (n+1)
%
        phi             = pi/36;
        labda           = 1.01;
%
%           -- rotation tensor
%
                Rn1 = [  cos(phi)     -sin(phi)
                         sin(phi)      cos(phi) ];
%
%           -- elongation tensor
%
                Un1 = [  labda      0
                           0      1/labda ];
%
%           -- deformation tensor
%
                Fn1 = Rn1*Un1;
%
%
%
%   (1) CALCULATION OF Q*[Tn+1 (Fn+1)]* Q'
%   ________________________________________
%
%       (1a) CALCULATION OF Tn+1 (Fn+1)
%       ________________________________
%
%       calculation of F(n+alpha)
%
                Fna  = (1-alpha)*I + alpha*Fn1;
%
%       calculation of L(n+alpha)*hn  ; hn = dummy parameter = step size
%
                Lna  = (Fn1 - I)*inv(Fna);
%
%       calculation of D(n+alpha)*hn
%
                Dna  = 0.5*(Lna+Lna');
%
%       calculation of Tn+1(Fn+1)
%
                Tn1  = Fn1*Tn*Fn1'+ 4*C*Fn1*inv(Fna)*Dna*(inv(Fna))'*Fn1';
%
%
%       (1b) CALCULATION OF Q*[Tn+1 (Fn+1)]*Q'
%       _______________________________________
%
%       * extra rigid body rotation tensor Q:
%             rotation over 0....90 degrees
%             calculation for 13 values of angle of rotation:
%             angle = (i-1)*7.5 degrees  {i=1,..,13}
%       * all results stored in matrix 'resQTQc'
%
%
```

```
%
    for i = 1:13;
%
        psi = (i–1)*pi/24;
%
        Q = [            cos(psi)      –sin(psi)
                         sin(psi)       cos(psi) ];
%
        QTQc = Q*Tn1*Q';
%
        resQTQc((2*i–1):(2*i),1:2) = QTQc;
%
    end;
%
%
%
%  (2) CALCULATION OF [Tn+1 (Q*Fn+1)]
%  ________________________________________
%
%       * extra rigid body rotation tensor Q:
%             rotation over 0....90 degrees
%             calculation for 13 values of angle of rotation:
%             angle = (i–1)*7.5 degrees  {i=1,..,13}
%       * all results stored in matrix 'resTQF'
%
    for i = 1:13;
%
        psi = (i–1)*pi/24;
%
        Q = [            cos(psi)      –sin(psi)
                         sin(psi)       cos(psi) ];
%
%       calculation of new deformation tensor
%
        QFn1   = Q*Fn1;
%
%       calculation of new F(n+alpha)
%
        Fna  = (1–alpha)*I + alpha*QFn1;
%
%       calculation of new L(n+alpha)*hn
%
        Lna  = (QFn1 – I)*inv(Fna);
%
%       calculation of new D(n+alpha)*hn
%
        Dna  = 0.5*(Lna+Lna');
%
%       calculation of Tn+1(Q*(Fn+1))
%
      Tn1  = QFn1*Tn*QFn1'+ 4*C*QFn1*inv(Fna)*Dna*(inv(Fna))'*QFn1';
%
      resTQF((2*i–1):(2*i),1:2) = Tn1;
%
    end;
%
```

```
%
%   Make plots
%
for i = 1:13;
      a1(i)=resQTQc((2*i-1),1);
      a2(i)=resQTQc((2*i-1),2);
      a3(i)=resQTQc((2*i),1);
      a4(i)=resQTQc((2*i),2);
      b1(i)=resTQF((2*i-1),1);
      b2(i)=resTQF((2*i-1),2);
      b3(i)=resTQF((2*i),1);
      b4(i)=resTQF((2*i),2);
end;
%
clg
%
x=1:13;
x=(x-1)*7.5;
%
subplot(221);
      plot(x,a1,x,b1)
      xlabel('angle of rotation (degr.)')
      title('component 11')
subplot(222);
      plot(x,a2,x,b2)
      xlabel('angle of rotation (degr.)')
      title('component 12')
subplot(223);
      plot(x,a3,x,b3)
      xlabel('angle of rotation (degr.)')
      title('component 21')
subplot(224);
      plot(x,a4,x,b4)
      xlabel('angle of rotation (degr.)')
      title('component 22')
pause
```

APPENDIX E: PC–MATLAB PROGRAM FILES OF APPLICATIONS OF THE INTEGRATION ALGORITHM

FILE: ROTATE.M

```
%
%   Example 1:          numerical integration according to Pinsky c.s.
%                       applied to a rigid body rotation
%   ____________________________________________________________________________
%
%   constitutive equation: rate form of Neo–Hookean equation
%
%                                *
%                   sigma =     p I + tau
%
%                            *
%                   with:   p       = (pseudo) hydrostatic pressure, determined by
%                                     boundary conditions
%                           sigma   = Cauchy stress tensor
%                           I       = Unity tensor
%                           tau     = 2C*(B–I)
%                           C       = material parameter
%                           B       = left Cauchy–Green strain tensor
%
%                   for tau holds:
%
%                   Truesdell rate of tau       = 4C*D
%                                   with  D     = rate of strain tensor
%
%   Numerical integration of tau
%   ________________________________
%
%   varied parameters:
%   – interpolation parameter alpha (alpha = 0.5 –> incrementally objective)
%   – number of increments
%
%     –> N              resp.:  15,   30,   60,  120,  240, 480
%     –> alpha          resp.: 0.00, 0.25, 0.50, 0.75, 1.00
%
%
%   deformation pattern:
%
%     in N increments a rigid body rotation over 120 degrees is reached
%     total rigid body rotation angle = 2*pi/3 radians, so the
%     incremental rigid body rotation angle = 2*pi/(3*N) radians
%
```

```
echo off;
%
%
            PHI  = 2*pi/3 ;
%
%
            C       = 1.0;
%
            T0      = [  0      0
                         0      0   ];
%
            I       = eye(2);
%
%
            aa      = [ 0.00 0.25 0.50 0.75 1.00 ];
            NN      = [  15   30   60   120  240  480];
%
%
            for iii=1:5;
                alpha=aa(iii);
%
                for ii=1:6;
                    N   = NN(ii);
%
%                   Calculation of incremental deformation tensor
%                   ______________________________________________
%
                    phi       = PHI/N;
%
                    Fn1       = [  cos(phi)        -sin(phi)
                                   sin(phi)        cos(phi)       ];
%
%                   Calculation of several matrices
%                   _______________________________
%
                    Fna  = (1-alpha)*I + alpha*Fn1;
%
                    Pna  = Fn1*inv(Fna);
%
                    Lna  = Pna - inv(Fna);
%
                    Dna  = 0.5*(Lna+Lna');
%
                    Tn   = T0;
%
%                   Calculation of tau(n+1)
%                   _______________________
%
                    for i=1:N;
%
                        Tn1=Fn1*Tn*Fn1'+ 4*C*Pna*Dna*Pna';
%
%                       Neutralization of Tn1 for rigid body rotations
                        (Updated approach !!!)
                        Tn =Fn'*Tn1*Fn;
%
                    end;
```

```
%
%
%          output
%          ------
%
                fprintf('--------------------------------------------')
                fprintf('\nRigid Body Rotation      ')
                fprintf('\n--------------------------------------------')
                fprintf('\nalpha                  = %5.2f    ',alpha);
                fprintf('\nnumber of increments = %3.0f \n',i)     ;
                fprintf('--------------------------------------------\n')
                    Tn
         end;
         fprintf('\n')
      end;
```

FILE: STRETCH.M

```
%
%    Example 2:          numerical integration according to Pinsky c.s.
%                        applied to an isovolumetric uniform elongation
%    ____________________________________________________________________
%
%    constitutive equation: rate form of Neo—Hookean equation
%
%                                 *
%                  sigma =     p I + tau
%
%                             *
%                  with:     p        = (pseudo) hydrostatic pressure, determined by
%                                       boundary conditions
%                            sigma    = Cauchy stress tensor
%                            I        = Unity tensor
%                            tau      = 2C*(B—I)
%                            C        = material parameter
%                            B        = left Cauchy—Green strain tensor
%
%                  for tau holds:
%
%                  Truesdell rate of tau          = 4C*D
%                                   with  D       = rate of strain tensor
%
%    Numerical integration of tau
%    ________________________________
%
%    varied parameters:
%    — interpolation parameter alpha (alpha = 0.5 —> incrementally objective)
%    — number of increments
%
%      —> N            resp.:  15,   30,   60,  120,  240, 480
%      —> alpha        resp.: 0.00, 0.25, 0.50, 0.75, 1.00
%
%
%    deformation pattern:
%
%      in N increments a total elongation of 30% is realized;
%      total elongation factor LABDA = 1.3, so the
%      incremental elongation factor labda = (1.3)^(1/N)
%
%
%
%
echo off;
```

```
%
%
          LABDA = 1.30   ;
%
%
          C     = 1.0;
%
          T0    = [  0     0
                     0     0   ];
%
          I     = eye(2);
%
%
          aa    = [ 0.00 0.25 0.50 0.75 1.00 ];
          NN    = [  15   30   60   120  240  480];
%
%
          for iii=1:5;
                alpha=aa(iii);
%
                for ii=1:6;
                      N   = NN(ii);
%
%                     Calculation of incremental deformation tensor
%                     ______________________________________________
%
                      labda    = LABDA^(1/N);
%
                      Fn1      = [   labda       0
                                       0       1/labda    ];
%
%                     Calculation of several matrices
%                     _______________________________
%
                      Fna  = (1-alpha)*I + alpha*Fn1;
%
                      Pna  = Fn1*inv(Fna);
%
                      Lna  = Pna - inv(Fna);
%
                      Dna  = 0.5*(Lna+Lna');
%
                      Tn   = T0;
%
%                     Calculation of tau(n+1)
%                     _______________________
%
                      for i=1:N;
%
                           Tn1=Fn1*Tn*Fn1'+ 4*C*Pna*Dna*Pna';
%
                           Tn =Tn1;
%
                      end;
```

```
%
%
%        output
%        ------
%
            fprintf('----------------------------------------')
            fprintf('\nIsovolumetric elongation')
            fprintf('\n----------------------------------------')
            fprintf('\nalpha                  = %5.2f   ',alpha);
            fprintf('\nnumber of increments = %3.0f \n',i)    ;
            fprintf('----------------------------------------\n')
               Tn
      end;
      fprintf('\n')
   end;
```

FILE: MIXED.M

```
%
%       Example 3:      numerical integration according to Pinsky c.s.
%                       applied to a uniform deformation consisting of:
%                           – simple shear, followed by an
%                           – isovolumetric elongation and a
%                           – rigid body rotation
%       ____________________________________________________________
%
%
%     constitutive equation: rate form of Neo–Hookean equation
%
%                                *
%                 sigma =       p I + tau
%
%                             *
%                 with:     p        = (pseudo) hydrostatic pressure, determined by
%                                      boundary conditions
%                           sigma    = Cauchy stress tensor
%                           I        = Unity tensor
%                           tau      = 2C*(B–I)
%                           C        = material parameter
%                           B        = left Cauchy–Green strain tensor
%
%                 for tau holds:
%
%                 Truesdell rate of tau            = 4C*D
%                                  with  D         = rate of strain tensor
%
%     Numerical integration of tau
%     ____________________________
%
%     varied parameters:
%     – interpolation parameter alpha (alpha = 0.5 –> incrementally objective)
%     – number of increments
%
%       –> N              resp.:   15,   30,   60,  120,  240, 480
%       –> alpha          resp.: 0.00, 0.25, 0.50, 0.75, 1.00
%
%
%     deformation pattern:              in N increments the following uniform
%                                       deformation is realized
%
%         * simple shear over 60 degrees, followed by
%         * uniform isovolumetric elongation, elongation factor = 1.30
%         * rigid body rotation over 120 degrees
%
%     parameters:
%                       PHI1    = total shear angle
%                       fphi1   = incremental shear factor
%                       PHI2    = total rigid body rotation angle
%                       phi2    = incremental rigid body rotation angle
%                       LABDA = total elongation factor
%                       labda   = incremental elongation factor
%
```

```
%
%
echo off;

%
                    PHI2   = 2*pi/3   ;
                    PHI1   = pi/3   ;
                    LABDA = 1.30   ;
                    TPHI1  = tan(PHI1);
%
%

                    C        = 1.0;
%
                    T0       = [  0      0
                                  0      0  ];
%
                    I        = eye(2);
%
%

                    aa       = [ 0.00 0.25 0.50 0.75 1.00 ];
                    NN       = [  15   30   60   120  240  480];
%
%

                    for iii=1:5;
                         alpha=aa(iii);
%
                         for ii=1:6;
                              N   = NN(ii);
%
%
%                             Calculation of incremental deformation tensor
%                             ________________________________________________

                              phi2      = PHI2/N      ;
                              labda     = LABDA^(1/N);
                              beta      = labda;
                              for j=1:(N-1);
                                   beta = labda*beta + labda*((labda)^(-j));
                              end;
%
                              tphi1     = LABDA*TPHI1/beta;
%
                              Rn1      = [   0      1
                                             1      0  ];
%
                              Un1      = [   0       1
                                             1      tphi1     ];
%
                              Rn2      = [  cos(phi2)       -sin(phi2)
                                            sin(phi2)        cos(phi2)          ];
%
                              Un2      = [  labda     0
                                              0       1/labda     ];
%
                              Fn1      = Rn2*Un2*Rn1*Un1;
%
%
```

```
%
%                         Calculation of several matrices
%                         ________________________________
%
                          Fna  = (1-alpha)*I + alpha*Fn1;
%
                          Pna  = Fn1*inv(Fna);
%
                          Lna  = Pna - inv(Fna);
%
                          Dna  = 0.5*(Lna+Lna');
%
                          Tn   = T0;
%
%                         Calculation of tau(n+1)
%                         _______________________
%
                          for i=1:N;
%
                             Tn1=Fn1*Tn*Fn1'+ 4*C*Pna*Dna*Pna';
%
%                            Neutralization of Tn1 for rigid body rotation
%                            (Updated approach !!)
%
                             Tn =Rn2'*Tn1*Rn2;
%
                          end;
%
%      output
%      ______
%
%
         fprintf('————————————————————')
         fprintf('\nSimple Shear, Elongation and')
         fprintf('\nRigid Body Rotation ')
         fprintf('\n————————————————————')
         fprintf('\nalpha                = %5.2f   ',alpha);
         fprintf('\nnumber of increments = %3.0f \n',i)    ;
         fprintf('————————————————————\n')
             Tn
       end;
       fprintf('\n')
   end;
```

Appendix E.10

APPENDIX F: PC–MATLAB PROGRAM FILES OF APPLICATIONS OF THE ALTERNATIVE ALGORITHM

FILE: AROTAT.M

```
%
%     Example 1:          numerical integration using the alternative
%                         integration algorithm,
%                         applied to a rigid body rotation
%_____________________________________________________________
%
%     constitutive equation: rate form of Neo–Hookean equation
%
%                                   *
%                sigma     = – p  I + tau
%
%                  *
%     with:      p         = (pseudo) hydrostatic pressure, determined by
%                            boundary conditions
%                sigma     = Cauchy stress tensor
%                I         = Unit tensor
%                tau       = 2C*(B–I)
%                C         = material parameter
%                B         = left Cauchy–Green strain tensor
%
%     calculation of tau(n+1), with tau(n) and L(n+1) given
%
%     varied parameters:
%     – interpolation parameter alpha (alpha = 0.5 –> incrementally objective)
%     – number of increments
%
%      –> N               resp.:  15,   30,   60,  120,  240, 480
%      –> alpha           resp.: 0.00, 0.25, 0.50, 0.75, 1.00
%
%
%     deformation:             in N increments a rigid body rotation over
%                              120 degrees is realized
%
%
%     parameters:
%                         PHI     = total rigid body rotation angle
%                         T       = total time in which the end configuration
%                                   is reached
%                         dt      = time step = T/N
%                         N       = number of increments
%                         alpha   = interpolation parameter
%                         C       = material parameter
%
echo off;
%
```

```
                      PHI   = 2*pi/3   ;
                      T     = 5.0      ;
                      C     = 1.0      ;
%
                      T0    = [  0      0
                                 0      0 ];
%
                      I     = eye(2);
%
%
                      aa    = [ 0.00 0.25 0.50 0.75 1.00 ];
                      NN    = [  15   30   60   120  240  480];
%
                      fprintf('\nPC Matlab calculation of tensor Tau ')
                      fprintf('\n————————————————————————————————')
                      fprintf('\n')
                      fprintf('\ntotal elapsed time       :  %5.2f s.     ',T);
                      fprintf('\ntotal angle of rotation :  %5.2f rad.   ',PHI);
                      fprintf('\nmaterial parameter       :  %5.2f N/mm2 ',C);
                      fprintf('\n\n\n\n')
%
%
          for iii=1:5;
            alpha=aa(iii);
%
            for ii=1:6;
              N    = NN(ii);
%
%             Calculation of incremental L matrix
%
%
%
              L  = [  0          –PHI/T
                      PHI/T       0     ];
%
%             Approximation of the incremental deformation tensor
%
              dt     = T/N;
%
              Fn1    = (inv(I – alpha*dt*L))*(I+(1–alpha)*dt*L);
%
              Tn     = T0;
%
%             Calculation of tau(n+1)
%
              for i=1:N;
                Tn1=Fn1*Tn*Fn1'+ 2*C*(Fn1*Fn1'–I);
%
%               Neutralization of Tn1 for rigid body rotation
%               (Updated approach !!)
%
                Rn    = [  cos(PHI*dt/T)        –sin(PHI*dt/T)
                           sin(PHI*dt/T)         cos(PHI*dt/T)  ];
%
                Tn = Rn'*Tn1*Rn;
              end;
```

```
        fprintf('————————————————————————')
        fprintf('\nRigid Body Rotation ')
        fprintf('\nalternative algorithm')
        fprintf('\n————————————————————————')
        fprintf('\nalpha                = %5.2f   ',alpha);
        fprintf('\nnumber of increments = %3.0f \n',i)    ;
        fprintf('————————————————————————\n')
        deter = det(Fn1);
        fprintf('\n———> determinant of Fn1 equals: %7.5f' , deter)
        trac = trace(L);
        fprintf('\n———> trace      of L   equals: %7.5f' , trac)
        Tn
    end;
fprintf('\n')
end;
```

FILE: ASTRET.M

```
%
%    Example 2:          numerical integration using the alternative
%                        integration algorithm,
%                        applied to a isovolumetric elongation
%_____________________________________________________________
%
%    constitutive equation: rate form of Neo-Hookean equation
%
%                                      *
%                    sigma    =  -  p  I + tau
%
%                      *
%    with:           p        = (pseudo) hydrostatic pressure, determined by
%                               boundary conditions
%                    sigma    = Cauchy stress tensor
%                    I        = Unity tensor
%                    tau      = 2C*(B-I)
%                    C        = material parameter
%                    B        = left Cauchy-Green strain tensor
%
%    calculation of tau(n+1), with tau(n) and L(n+1) given
%
%    varied parameters:
%    - interpolation parameter alpha (alpha = 0.5 -> incrementally objective)
%    - number of increments
%
%     ->        N        resp.:  15,   30,   60,  120,  240, 480
%     ->        alpha    resp.: 0.00, 0.25, 0.50, 0.75, 1.00
%
%
%       deformation:           in N increments a isovolumetric elongation with
%                              elongation factor LABDA is reached
%
%       parameters:
%                        LABDA = total elongation factor
%                        T     = total time in which the end configuration
%                                is reached
%                        dt    = time step = T/N
%                        N     = number of increments
%                        alpha = interpolation parameter
%                        C     = material parameter
%
echo off;
%
                         LABDA = 1.3     ;
                         T     = 5.0     ;
                         C     = 1.0     ;
%
                         T0    = [  0      0
                                    0      0 ];
%
               I   = eye(2);
```

```
%
                    aa    = [ 0.00 0.25 0.50 0.75 1.00 ];
                    NN    = [  15    30    60    120   240   480];
%
                    fprintf('\nPC Matlab calculation of tensor Tau ')
                    fprintf('\n-----------------------------------------------')
                    fprintf('\n')
                    fprintf('\ntotal elapsed time        :  %5.2f s.     ',T);
                    fprintf('\ntotal elongation factor :  %5.2f [-]    ',LABDA);
                    fprintf('\nmaterial parameter        :  %5.2f N/mm2 ',C);
                    fprintf('\n\n\n\n')
%
%
      for iii=1:5;
         alpha=aa(iii);
%
         for ii=1:6;
            N    = NN(ii);
%
%           Calculation of incremental L matrix
%
            L  =  (1/T)*log(LABDA)*        [  1       0
                                              0      -1 ];
%
%           Approximation of the incremental deformation tensor
%
            dt      =  T/N;
%
            Fn1   =  (inv(I - alpha*dt*L))*(I+(1-alpha)*dt*L);
%
            Tn      =  T0;
%
%           Calculation of tau(n+1)
%
            for i=1:N;
%
               Tn1=Fn1*Tn*Fn1'+ 2*C*(Fn1*Fn1'-I);
%
               Tn = Tn1;
%
            end;
%
            fprintf('-----------------------------------')
            fprintf('\nIsovolumetric Elongation ')
            fprintf('\nalternative algorithm')
            fprintf('\n-----------------------------------')
            fprintf('\nalpha                    = %5.2f    ',alpha);
            fprintf('\nnumber of increments = %3.0f \n',i)     ;
            fprintf('-----------------------------------\n')
            deter = det(Fn1);
            fprintf('\n-------> determinant of Fn1 equals: %7.5f' , deter)
            trac = trace(L);
            fprintf('\n-------> trace        of L    equals: %7.5f' , trac)
            Tn
         end;
         fprintf('\n')
      end;
```

Appendix F.6