D I P L O M A R B E I T

M A S T E R ' S T H E S I S

Viskoelastizität von Knochen: Auswertung von Experimenten und Mehrskalen-Modellierung

Viscoelasticity of bone: Review of experiments and multiscale modeling

ausgeführt zum Zwecke der Erlangung des akademischen Grades eines Diplom-Ingenieurs

unter der Anleitung von

Ao.Univ.-Prof. Dipl.-Ing. Dr. techn. **Christian Hellmich**
Institut für Mechanik der Werkstoffe und Strukturen
Fakultät für Bauingenieurwesen
Technische Universität Wien

eingereicht an der Technischen Universität Wien
Fakultät für Bauingenieurwesen

von

Lukas Eberhardsteiner

Matr.Nr.: 0425851
Premreinergasse 24/6
A - 1130 Wien

Wien, im Juni 2010

Danksagung

Diese Diplomarbeit stellt das Ende meines Bauingenieurwesen-Studiums an der Technischen Universität Wien dar. An dieser Stelle möchte ich mich bei all jenen bedanken, die mich bei der Absolvierung dieses Studiums unterstützt und zur Erstellung dieser Arbeit beigetragen haben.

Mein größter Dank gilt meiner Familie, ohne deren Unterstützung ein Studium in dieser Form sicherlich nicht möglich gewesen wäre und die mich über all die Jahre in vielerlei Hinsicht unterstützt hat.

Besonders herzlicher Dank gebührt Herrn Ao. Univ.-Prof. Dipl.-Ing- Dr. techn. Christian Hellmich für die hervorragende Betreuung bei der Erstellung dieser Arbeit. Trotz meist sehr vollem Terminkalender stand seine Tür für mich nicht nur für Fragen zur Diplomarbeit immer offen. In einigen, oft längeren Gesprächen über meine berufliche Zukunft bekam ich so manchen Ratschlag, der mir bei der Entscheidungsfindung sehr geholfen hat.

Großer Dank gilt auch Herrn Dipl.-Ing. Dr. techn. Stefan Scheiner, der mit seiner Arbeit über viskoelastisches Verhalten von Beton eine wichtige Grundlage für das Entstehen dieser Diplomarbeit gelegt hat.

Weiters möchte ich auch Herrn Dipl.-Ing. Dr. techn. Andreas Fritsch danken, der mich mit der Durchführung der einen oder anderen Berechnung unterstützte und sich für die Beantwortung von Fragen immer Zeit nahm.

Zu guter Letzt möchte ich mich noch bei meinen Studienkollegen bedanken, die mich vor allem am schwierigen Beginn meines Studiums unterstützt haben und mir immer mit Rat und Tat zur Seite standen.

Wien, im Juni 2010 Lukas Eberhardsteiner

Kurzfassung

Es scheint Sinn zu machen, den Knochenbestandteil "Kollagen" für das viskoelastische Verhalten von Knochen verantwortlich zu machen. Allerdings zeigen neuere Forschungen, dass mit dem Kriechen einhergehendes Brechen von "Collagen-Crosslinks" erst dann auftritt, wenn die mit dünnen Wasserlagen umgebenen Hydroxyapatit-Kristalle schon längst in irreversible Gleitprozesse involviert sind. Daher ist es reizvoller, die Kriechdehnungen von Knochen diesen Gleitungsprozessen von Mineralkristallen auf geordneten Wasserlagen im extrafibrillären Raum zuzuordnen. Um dies mathematisch zu erfassen, wird in der vorliegenden Arbeit ein Mehrskalenmodell für Knochen entwickelt. Im Rahmen eines 6-Schritt-Verfahrens werden die Kriecheigenschaften hydrierter Kristalloberflächen durch ein Burgers-Modell erfasst. Dieser Ansatz gibt nicht nur experimentelle Daten zufriedenstellend wieder, er erlaubt auch das Studium des Einflusses verschiedener Mineralgehalte und vaskulärer Porositäten auf das makroskopische Kriechverhalten von Knochenmaterialien.

Abstract

It seems somehow natural to consider collagen to be responsible for the viscoelastic behavior of bone. According to latest research, however, the corresponding failure of crosslinks between collagen molecules is probable to take place when the hydrated hydroxyapatite crystals are already sliding one upon another, which makes it more appealing to assign creep strains to gliding events of hydroxyapatite needles on intercrystalline water layers in the extrafibrillar space. To be able to mathematically describe this viscoelastic behavior, a multiscale model for bone material is developed. Within a six-step homogenization scheme, the Burgers model was used to characterize the long-term viscoelastic properties of layered water between hydroxyapatite crystal needles. This approach does not only fit experimental results, but it also allows us to study the influence of varying mineral contents and vascular porosities on the overall creep behavior of bone materials.

Contents

Chapter 1

Introduction

1.1 Motivation

In order to understand the correlation between stresses and strains in bone material, not only the elastic or plastic, but also the viscoelastic behavior has to be examined. Especially, the phenomenon of relaxation, which has the ability to reduce stress concentrations occurring at micro cracks, bone fractures or fixations of implants, for example, is worth to take a look at. In addition, viscoelastic behavior has to be considered in the development of new biomaterials, in order to avoid non-physiological stresses at the interface between natural bone and implant.

There is no general agreement on the microstructural origin of bone creep. Since microstructural mechanisms like cement line motions cannot explain the observed amounts of creep [58], various ultrastructural creep mechanisms have been discussed [53], [21], [64], [51]. Such ultrastructural creep mechanisms include: (i) dislocation generation or cracking of hydroxyapatite mineral crystals, (ii) shear between collagen fibrils and subsequent failure of the collagen crosslinks, or (iii) separation (debonding) of hydroxyapatite crystals and the collagen "matrix". Bowman et al. [13] exclude creep of hydroxyapatite crystals because their melting temperature of 1600 centigrades [38] would suggest no creep below 800 centigrades [63]. Instead, Bowman et al. prefer to explain creep through some collageneous mechanism, probably related to crosslink failure. In a recent multiscale investigation, however, Fritsch et al. [25] have shown that, far below the load level related to crosslink failure, inelastic sliding-type mechanisms are probable to occur between the hydroxyapatite crystal interfaces, which are covered by layered water [8], [78]. Most remarkably, this idea is consistent with a multitude of biophysical, biochemical, and biomechanical test data.

This motivates us to assign creep strains to gliding events of hydroxyapatite needles on these intercrystalline water layers in the extrafibrillar space. In order to describe this vis-

coelastic behavior, a multiscale model of bone material is introduced herein, proposing six steps of homogenization. Within a representative material volume of extrafibrillar porous polycrystal, the Burgers model characterizes the long-term creep properties of hydrated hydroxyapatite needles.

1.2 Previous work

As to our knowledge, this is the first time viscoelasticity of bone is examined in the context of a multiscale model. Accordingly, the theoretical background of this work concerns viscoelastic behavior in general and a multiscale model for bone material in particular. By way of example, some publications are reviewed in the following:

The book "Viscoelastic Solids" by Lakes [43] can be regarded as standard textbook on viscoelasticity of solid materials. Relaxation and creep behavior in general, and their mutual correlation, are explained in a straightforward manner, followed by examples of creep behavior of different materials.

Scheiner and Hellmich [68] developed a viscoelastic multiscale model, considering aging effects in early-age concrete. Especially, the transformation into the Laplace-Carson-domain, as well as the back-transformation to the time domain, were shown in detail, and proved as a good basis for writing this theses.

A two-scale model for bone material was proposed by Katz [40], resolving the osteons in cortical bone into differently oriented lamellar bone layers. From a conceptual viewpoint, Lees and Davidson [46] went much further down, discussing mineral-collagen interactions as the origin of bone elasticity. The combination of both of these traditions has led the way to multiscale micromechanics for bone elasticity and strength, as driven forward by Hellmich et al. since the early 2000's [24], [25], [33], [29]. These authors presented models which are capable of predicting bone elasticity and strength characteristics from tissue-specific mineral and collagen contents and from intercrystalline, intermolecular, lacunar and vascular porosities, on the basis of "universal" elastic and strength properties of hydroxyapatite, water, and collagen.

The mathematical background of continuum micromechanics is provided by Zaoui [79], [80], the fundamentals concerning the Laplace-Carson-transformation and the back-transformation into the time domain are given by Donolato [19], Gurtin et al. [28], Gaver [26], as well as Valkó and Abate [2], [74], respectively.

The experimental data used to validate our model are provided by Iyo et al. [39] and Sasaki et al. [67]. In the first publication, three-point bending tests were performed on beam-like specimens cut in the longitudinal or transverse material direction of bone, in order to examine the anisotropic viscoelastic properties of bone material. The second manuscript deals with an approach for describing the relaxation Young's and shear moduli, as analytical function of time.

1.3 Scope

This theses is divided into 6 chapters. After this introducing chapter, Chapter 2 contains a short literature review. The experiments performed in [39] are examined, followed by a discussion on potential pitfalls concerning the derivation of Young's modulus from quasi-static tests. Values for Young's modulus derived in this matter are often lower than the actual Young's moduli, as determined by eigenfrequency tests, by ultrasonic tests, or predicted by well-validated microelastic models.

In Chapter 3, the multiscale model is described. At first, the fundamentals of continuum micromechanics and homogenization are explained, followed by subsections introducing the RVEs extrafibrillar space, wet collagen, mineralized collagen fibril, extracellular matrix, extravascular matrix and bone microstructure. In addition to the homogenization of the extrafibrillar space, viscoelastic behavior in general, including the Burgers model, and the Laplace-Carson-transformation are discussed in Subsection 3.1, while the back-transformation into the time domain is dealt with in Subsection 3.6.

In Chapter 4, the parameters of the Burgers model are determined and validated using experimental data from [39] and [67]. In addition, another approach to describe viscoelastic behavior using the Nabarro model is made in Subsection 4.4.

Chapter 5 deals with the discussion of elastic and viscoelastic properties considering varying mineral content, vascular porosity and water content.

Finally, Chapter 6 concludes this work.

1.4 Nomenclature

The quantities and indices used in this work are listed in Tables 1.1 and 1.2.

Table 1.1: Quantities used in this thesis

Physical quantity		Unit of measurement	
Symbol	Name	Symbol	Name
$\mathbb{C}$	stiffness tensor	GPa	Gigapascal
d	characteristic inhomogeneity length	m	meter
D	averaged standardized relative difference	-	-
$\bar{e}$	mean relative error	%	percent
e_s	standard deviation of relative error	-	-
E_0	initial modulus of elasticity	GPa	Gigapascal
f	volume fraction	%	percent
F	relaxation force	N	Newton
F_0	initial relaxation force	N	Newton
h	height of beam-like specimen	mm	millimeter
I	moment of inertia of beam-like specimen	mm^4	millimeter to the fourth power
$\mathbb{I}$	fourth-order unity tensor	-	-
$\mathbb{I}_{dev}$	deviatoric part of fourth-order unity tensor	-	-
$\mathbb{I}_{vol}$	volumetric part of fourth-order unity tensor	-	-
$\mathbb{J}$	creep tensor	1/GPa	1/Gigapascal
k	bulk modulus	GPa	Gigapascal
l	length of beam-like specimen	mm	millimeter
ℓ	characteristic length of RVE	m	meter
$\mathcal{L}$	characteristic length of structure	m	meter
p	complex variable in Laplace-Carson-domain	-	-
$\mathbb{P}$	Hill tensor	-	-
$\mathbb{R}$	relaxation tensor	GPa	Gigapascal
$\mathbb{S}$	Eshelby tensor	-	-
t	time	s	second
w	deflection	m	meter
α_{HA}	Nabarro parameter	GPa^{-1}	1/Gigapascal
ε	strain	-	-
$\eta_{HA,KV}$	Burgers parameter	GPa s	Gigapascal second
$\eta_{HA,M}$	Burgers parameter	GPa s	Gigapascal second
μ	shear modulus	GPa	Gigapascal
$\mu_{HA,KV}$	Burgers parameter	GPa	Gigapascal
ν	Poisson's ratio	-	-
ρ	density	kg/m^3	kilogram per cubic meter
σ	stress	GPa	Gigapascal
τ_{HA}	Nabarro parameter	s	second

Table 1.2: Indices used in this thesis

Symbol	Name	Symbol	Name
col	collagen	lac	lacunae
$cort$	cortical	mod	model
cyl	cylindrical	MT	Mori-Tanaka scheme
el	elastic	pl	plastic
ef	extrafibrillar space	SCS	Self-consistent scheme
exp	experimental	sph	spherical
$exvas$	extravascular	$ultra$	ultrastructure
fib	fibrillar	vas	vascular
HA	hydroxyapatite	wet	wet
ic	intercrystalline	$wetcol$	wet collagen
im	intermolecular	$*$	quantity in Laplace-Carson domain

Chapter 2

Experimental viscoelasticity of bone

2.1 Experimental determination of bone's elasticity and viscoelasticity - challenges

Derivation of elastic properties of bone from quasi-static tests is a delicate task. Accordingly, the values reported e.g. for the longitudinal direction of bovine femoral bone (defined in Fig. 2.1) are largely varying: Some researchers [39], [14], [50] report values between 10 and 14 GPa, based on not further specified static bending tests. Other researchers [50], [15] report much higher values, lying between 21 and 26 GPa, based on three-point bending tests according to ASTM standards [5]. The latter values agree very well with ultrasonic test [48], [45], [47], [6], [7], [52], which are known to reliably deliver bone's elastic properties, for two reasons: (i) the strain and stress amplitudes are very small, so that no inelastic (e.g. plastic) deformations are encountered; and (ii) the strain rates are well below 1, so that an elastic stiffness increase due to dynamic effects can be excluded.

As compared to elastic tests, viscoelastic tests have been performed much more rarely. Classical relaxation tests are by definition quasi-static, and virtually all related documents, including that of Iyo et al. [39], on which we base the following developments, contain elastic values, which seem much too low. Correspondingly, the aforementioned relaxation tests need to be carefully re-evaluated. Fortunately, this can be done in a straightforward manner, since the tests themselves often do not directly refer to elasticity components, but rather to forces and displacements.

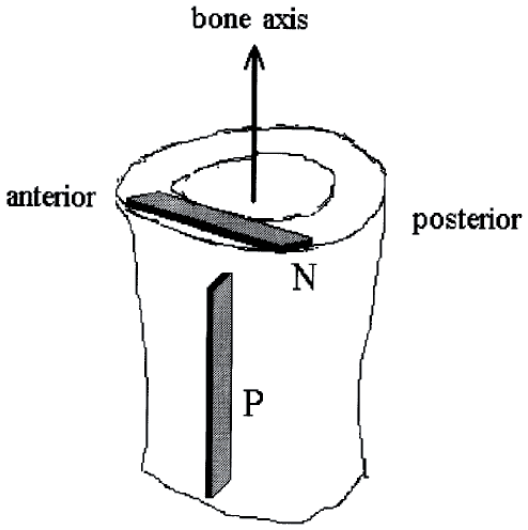

Figure 2.1: Definition of longitudinal (P) and transverse (N) direction of bone [39]

2.2 Three-point bending tests of Iyo et al. [39]: purely viscoelastic interpretation of the authors

As mentioned before, the studies of Iyo et al. [39] were taken as a reference for the current investigations. In this paper, three-point bending tests on bone samples, which were obtained from 36-month-old bovine femur were performed in order to determine the relaxation Young's modulus. The dimensions of the specimens can be seen in Fig. 2.2.

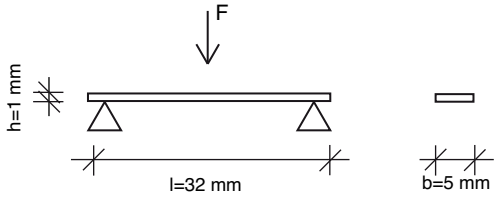

Figure 2.2: Dimensions of the specimens tested in [39]

The relaxation forces were measured by a force sensor, while the deflection was held constant. Based on these experimental data and on the *assumption of linear viscoelasticity of the entire viscoelastic beam structure*, Iyo et al. report the results of computations concerning the maximum strains at the bottom of the specimen at the midpoint of the beam and the relaxation Young's modulus R^{exp} as function of time,

$$R^{exp}(t) = E_0\{A_1 exp[-(t/\tau_1)^{\beta}] + (1 - A_1)exp[-(t/\tau_2)^{\gamma}]\}, \tag{2.1}$$

(see Table 2.1 for $E_0, A_1, \tau_1, \tau_2, \beta$ and γ, and Fig. 2.3 for R^{exp}) [39]. In [39], a yield strain of 0.6 % was assumed, a value much higher than the calculated maximum strain of 0.23 %. This was the argument of Iyo et al. for using elasticity theory, which - as we will see in the following - deserves a second thought.

Table 2.1: Parameters for $R^{exp}(t)$ [39]

direction	E_0 [GPa]	A_1	τ_1 [s]	τ_2 [$\times 10^6$ s]	β	γ
longitudinal	14.2	0.08	49	9.3	0.28	0.35
transverse	11.6	0.11	50	6.4	0.26	0.37

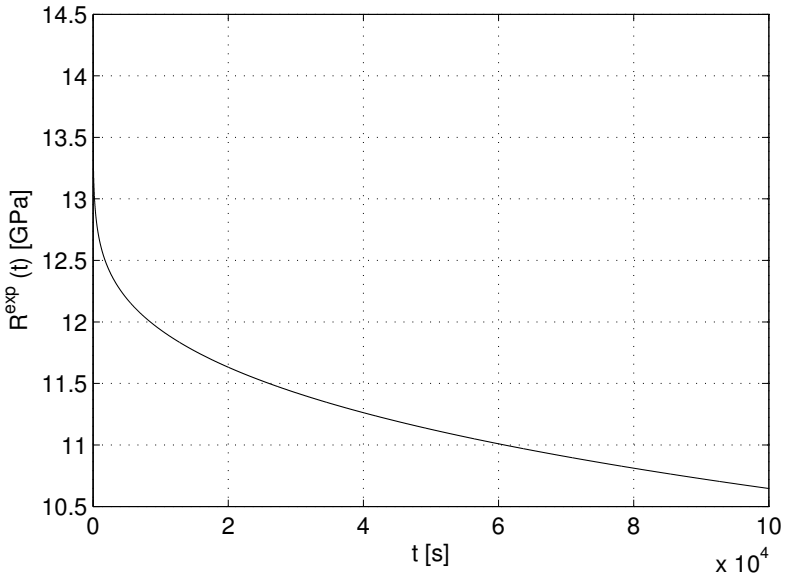

Figure 2.3: Time-dependent relaxation Young's modulus $R^{exp}(t)$ in longitudinal direction, as reported in Iyo et al. [39]

2.3 Three-point bending tests of Iyo et al. [39]: probability of plastic strains at the bottom of the beam center

The elastic limit of 0.6% strain as considered by Iyo et al. [39] typically refers to a situation, where the stress becomes an almost non-increasing function of strains, i.e. soon thereafter a behavior close of "ideal plasticity" has often been reported [62]. However, a recent multiscale investigation of Fritsch et al. [25] reveals that the elastic limit is reached probably much earlier: This study considers pure elasticity of the overall material until the contact strength between the hydroxyapatite crystals within bone is reached. This contact strength was determined from tests on hydroxyapatite polycrystals [23]. Beyond this point, which typically relates to an overall strain of already 0.045% (in an uniaxial stress test), the hydroxyapatite crystals start to slide on each other, and more and more (micro-) stresses are transfered to the collagen molecule assemblies, until the latter reach their strength (at a strain much closer to the 0.6% reported by Iyo et al.).

Since the microelastoplastic model of Fritsch et al. predicts stress-strain curves as functions of the material's mineral, collagen, and water content, we take a close look at the situation

of bovine tibia (the composition of which is known from tests reported in [48]), which is comparable to that with bovine femur. Namely, bovine femur and tibia are characterized by rather similar load transfer mechanisms. Therefore, the physiological structure and chemical composition, respectively, are more or less the same.

The yield strain according to Fritsch et al. [25] is one order of magnitude lower than the value of 0.23%, calculated by Iyo et al.This indicates that there are most probably plastic deformations at the bottom of the center of the tested beam-like specimen.

2.4 Three-point bending tests of Iyo et al. [39]: probability of plastic strains at the supports of the beam

Another reason for low values of the modulus of elasticity could be attributed to the missing consideration of plastic deformations at the supports.

Examination of the load-deflection curve in Fig. 2.4, obtained by Currey et al. [17], who is a representative of the first group of authors mentioned before, indicates that, although the gradient is not constant at the beginning of the respective curve, modulus of elasticity is calculated based on the linear elastic theory.

Figure 2.4: Load-deflection curve by Currey et al. [17]

From Fig. 2.4, two observations can be made: (i) low gradient at the beginning of the curve, and (ii) too low overall stiffness during the loading phase.

The first phenomenon is caused by the fact that at low force levels surface roughness prevents full contact between support and specimen, which yields unrealistic large local deformations at the supports.

In the second case, concentrated load application, caused by the use of cylindrical or sharp-edged supports, is unavoidable and leads to stress peaks in the specimen and to plastic deformations, consequently. Because the deflection is overestimated, the gradient in the load-deflection curve is too low (second slope in Fig. 2.4) and the material seems to be weaker than it really is.

In this case the modulus of elasticity could only be determined from an unloading curve.

2.5 Three-point bending tests by Iyo et al. [39]: force relaxation-characteristics

Since the formally elastic relaxation moduli reported by Iyo et al. [39] are most probably afflicted with errors due to negligence of actually plastic deformations, there is high interest in getting access to the quantities which were actually measured by Iyo et al.: the relaxing forces on the three-point bending test, while holding the beam deformation constant.

The aforementioned access is gained through using linear (visco-) elastic beam theory (like Iyo et al. did), but now for back-calculating the (actually measured) forces from the reported (but actually computed) relaxation moduli, according to

$$F^{exp}(t) = \frac{4R^{exp}(t)\varepsilon I}{l \times (h/2)}. \tag{2.2}$$

Eq. (2.2) relates to the statically determinate structure of a simply supported beam of length l, of moment of inertia I ($I = bh^3/12$, with cross sectional width b and height h), and with maximum longitudinal strain at the beam center, $\varepsilon = 0.23\%$. The corresponding force relaxation curve is depicted in Fig. 2.5.

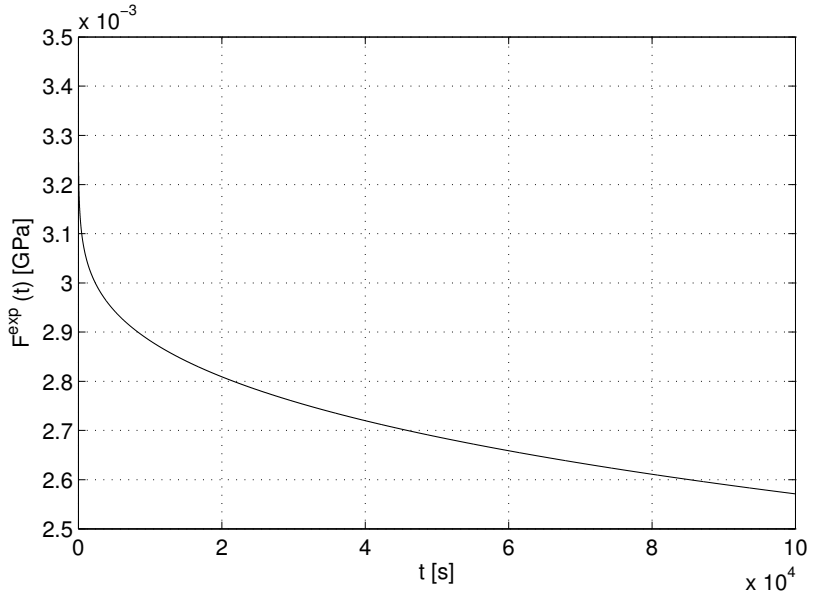

Figure 2.5: Relaxation force evolution $F^{exp}(t)$, measured by Iyo et al. [39], but not directly documented. Therefore, $F^{exp}(t)$ is back-calculated from Iyo et al.'s "formally viscoelastic relaxation modulus", according to Eq. (2.2)

Chapter 3

Multiscale viscoelasticity of bone

Continuum micromechanics [68], [35], [36], [79], [80], [72] is based on the idea that a material is a micro-heterogenous body filling a macro-homogeneous representative volume element (RVE) with characteristic length $\ell, \ell \gg d$, where d stands for the characteristic length of the inhomogeneities within the RVE (see Fig. 3.1), and $\ell \ll \mathcal{L}$, with $\mathcal{L}$ as the characteristic lengths of geometry or loading of a structure built up by the material defined on the RVE. Quasi-homogenous subdomains, so called material phases [79], [80], [72], with known physical properties, e.g. volume fractions or elastic/viscoelastic properties, are reasonably chosen, as to describe the complicated microstructure within an RVE as simply as possible but not simpler than that.

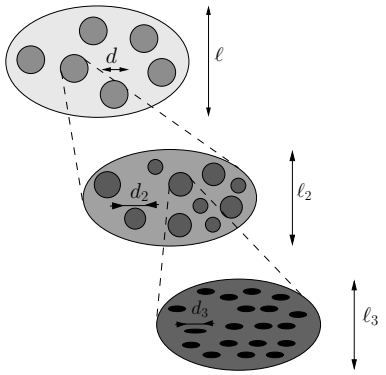

Figure 3.1: Multistep homogenization [24]

The elastic/viscoelastic behavior of this inhomogeneities within the RVE (material phases), as well as the volume fractions, their characteristic shapes and their interactions, are used to estimate the correlation between homogeneous deformations acting on the boundary of the RVE and resulting (average) stresses, in other words the homogenized mechanical behavior of the overall material.

If a single phase has a heterogeneous microstructure itself, RVEs can be introduced within this phase in order to estimate its mechanical behavior. These RVEs have dimensions, which fulfill $\ell_2 \leq d$, and imply again smaller inhomogeneities with characteristic length $d_2 \ll \ell_2$, and so on (see Fig. 3.1).

This approach leads to a multistep homogenization scheme. In the following subsections, the single steps of such a scheme for bone material (see Fig. 3.2) are discussed.

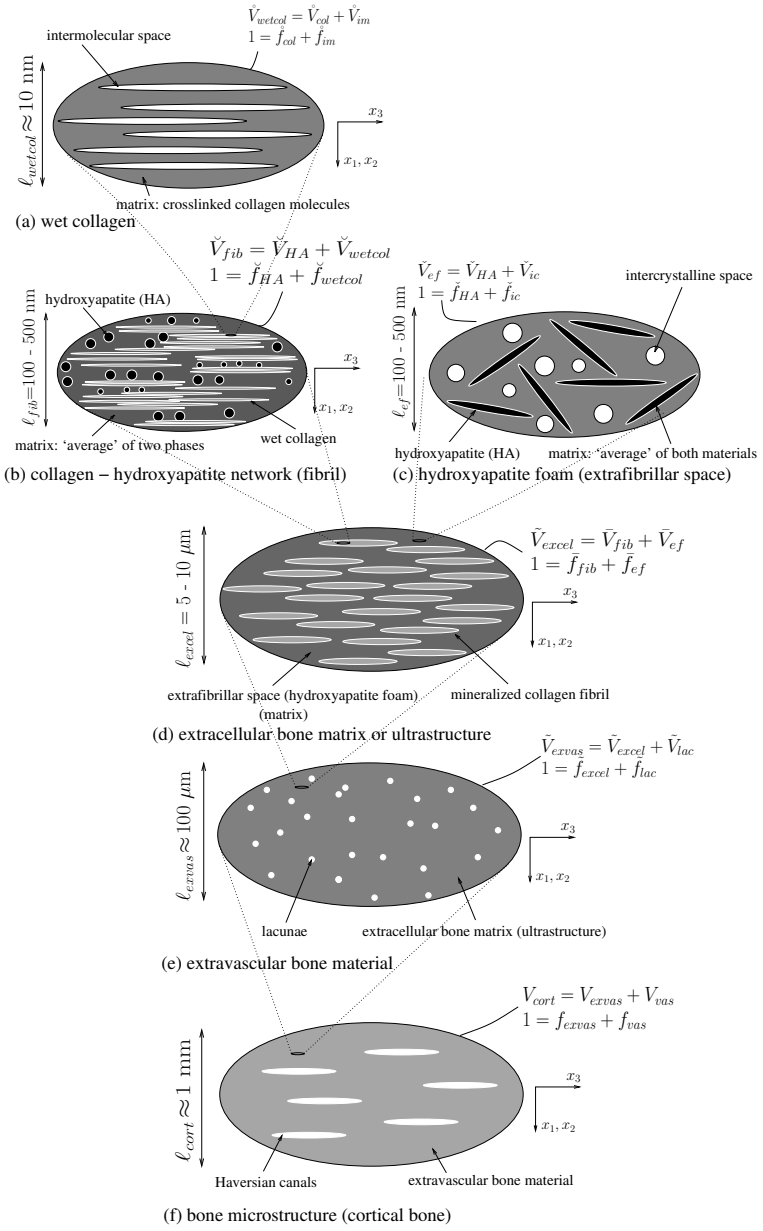

Figure 3.2: Micromechanical representation of bone material by means of a six-step homogenization scheme [25]

3.1 Homogenization of extrafibrillar space (hydroxyapatite foam)

An RVE $\check{V}_{ef}$ of hydroxyapatite foam [see Fig. 3.2 (c)] has a characteristic length between 100 and 500 nm. It is built up by hydroxyapatite needles, being oriented in all directions (labelled by suffix "HA"), and spherical, water-filled pores, called intercrystalline space (labelled by suffix "ic").

In order to describe the behavior of the viscously sliding thin water layers between the needle-shaped hydroxyapatite crystals, the latter are assigned viscoelastic material behavior. Accordingly, we here discuss the behavior of viscoelastic phases (with index r) and of homogenized materials built up from such phases.

The Riemann-Stieltjes integrals [28], [10], [66]

$$\boldsymbol{\sigma}_r(t) = \int_{-\infty}^{t} \mathbb{r}_r(t-\tau) : \dot{\boldsymbol{\varepsilon}}_r(\tau) d\tau = [\mathbb{r}_r * \boldsymbol{\varepsilon}_r](t) \tag{3.1}$$

and

$$\boldsymbol{\varepsilon}_r(t) = \int_{-\infty}^{t} \mathbb{j}_r(t-\tau) : \dot{\boldsymbol{\sigma}}_r(\tau) d\tau = [\mathbb{j}_r * \boldsymbol{\sigma}_r](t) \tag{3.2}$$

describe the relationships between microscopic stresses and strains and strains and stresses in a viscoelastic material, respectively. $\mathbb{r}_r(t-\tau)$ in Eq. (3.1) is the fourth-order tensorial relaxation function of phase r and $\mathbb{j}_r(t-\tau)$ in Eq. (3.2) stands for the fourth-order tensorial creep function of phase r. $\dot{\boldsymbol{\varepsilon}}$ and $\dot{\boldsymbol{\sigma}}$ refer to the temporal derivatives of the strain and stress tensors, respectively, of phase r. τ is the time instant, when strain or stress rates, $\dot{\boldsymbol{\varepsilon}}_r$ or $\dot{\boldsymbol{\sigma}}_r$, were applied onto phase r, and $*$ stands for the Stieltjes convolution operator.

While instantaneous elasticity is described by $\mathbb{r}_r(t=\tau)$, $\mathbb{r}_r(t>\tau)$ is related to viscoelastic behavior. If $\partial \mathbb{r}_r / \partial t = \mathbb{0}$ ($\mathbb{r}_r$ =constant), pure elasticity is described and $\mathbb{r}_r = \mathbb{c}_r$, where $\mathbb{c}_r$ stands for the fourth-order elastic stiffness tensor of phase r.

The relation between the creep and relaxation function can be described through [69], [42]

$$\int_{-\infty}^{t} \mathbb{r}_r(t-\tau) : \mathbb{j}_r(\tau) d\tau = \int_{-\infty}^{t} \mathbb{j}_r(t-\tau) : \mathbb{r}_r(\tau) d\tau = t\mathbb{I}. \tag{3.3}$$

The key mathematical tool for making multiscale viscoelasticity tractable is the the Laplace-Carson (LC) transformation [68],[19],

$$f^*(p) = \mathcal{C}\{f(t)\} = p\hat{f}(p) = p\int_0^\infty f(t)e^{-pt}dt, \tag{3.4}$$

where $f^*(p)$ represents the LC-transform of the time-dependent function $f(t)$, p is the complex variable in the LC-domain, and $\hat{f}(p)$ the Laplace transform of $f(t)$. In fact, when applying Eq. (3.4) to the relaxation and creep functions [Eqs. (3.1) and (3.2)], the latter are transformed into algebraic constitutive equations of the formally elastic type [28]

$$\boldsymbol{\sigma}_r^*(p) = \mathbb{r}_r^*(p) : \boldsymbol{\varepsilon}_r^*(p) \tag{3.5}$$

and

$$\boldsymbol{\varepsilon}_r^*(p) = \mathbb{j}_r^*(p) : \boldsymbol{\sigma}_r^*(p). \tag{3.6}$$

Accordingly, the viscoelastic behavior of the hydroxyapatite needles can be expressed by

$$\mathbb{r}_{HA}^*(p) = [\mathbb{j}_{HA}^*(p)]^{-1} = 3k_{HA}^*(p)\mathbb{I}_{vol} + 2\mu_{HA}^*(p)\mathbb{I}_{dev}. \tag{3.7}$$

In Eq. (3.7), $k_{HA}^*(p)$ and $\mu_{HA}^*(p)$ represent the LC-transformed bulk and shear relaxation moduli of the hydroxyapatite needles; $\mathbb{I}_{vol}$ is the volumetric part of the fourth-order unity tensor, with components $I_{vol,ijkl} = 1/3\delta_{ij}\delta_{kl}$, $\mathbb{I}_{dev}$ represents the deviatoric part of the fourth-order unity tensor $\mathbb{I}$ ($\mathbb{I}_{dev} = \mathbb{I} - \mathbb{I}_{vol}$).

In this paper, the Burgers model is used to describe the creep behavior of the thin water layers between the hydroxyapatite needles [as depicted in Fig. 3.3(a)], by means of the LC-transformed shear relaxation modulus [73], [75]

$$\mu_{HA}^*(p) = \left(\frac{1}{\mu_{HA,inst}} + \frac{1}{\mu_{HA,KV} + p\eta_{HA,KV}} + \frac{1}{p\eta_{HA,M}}\right)^{-1}, \tag{3.8}$$

where $\mu_{HA,inst} = \mu_{HA}$ is the instantaneous elastic shear modulus, according to Table 3.1, and $\mu_{HA,KV}$, $\eta_{HA,KV}$ and $\eta_{HA,M}$ are the Burgers parameters. Three regimes typical for creep curves can also be observed for the Burgers model [see Fig. 3.3 (b)]: (i) an instantaneous elastic response, (ii) a decaying creep response, and finally (iii) a constant-rate creep response.

Since we envision sliding along water layers rather than consolidation of the latter as the governing phenomenon, $k_{HA}^*(p)$ is constant and equal to the elastic bulk modulus k_{HA} (see Table 3.1).

A homogenized relaxation function of a material in general can be formulated [80], considering matrix-inclusion problems described by Eshelby [20] and Laws [44]:

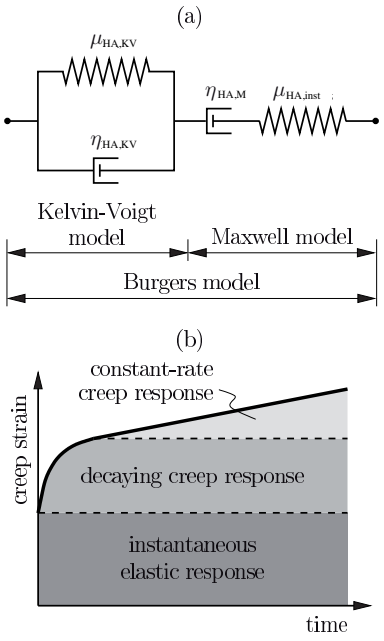

Figure 3.3: (a) Viscolelastic behavior based on the Burgers model; (b) Burgers model-related typical regimes of creep strains [68]

$$\mathbb{R}^{*,est} = \sum_r f_r \mathbb{r}_r^* : \left[\mathbb{I} + \mathbb{P}_r^{*,0} : \left(\mathbb{r}_r^* - \mathbb{R}^{*,0}\right)\right]^{-1} : \left\{\sum_s f_s \left[\mathbb{I} + \mathbb{P}_s^{*,0} : \left(\mathbb{r}_s^* - \mathbb{R}^{*,0}\right)\right]^{-1}\right\}^{-1}, \quad (3.9)$$

with f_r and $\mathbb{c}_r$ as the volume fraction and viscoelastic stiffness of phase r, and $\mathbb{I}$ as the fourth-order unity tensor with the components $I_{ijkl} = (1/2) \times (\delta_{ik}\delta_{jl} + \delta_{il}\delta_{jk})$, δ_{ij} is the Kronecker delta ($\delta_{ij} = 1$ for $i = j$ and $\delta_{ij} = 0$ for $i \neq j$). Both sums are taken over all phases appearing within the heterogeneous RVE. The characteristic shape of phase r is considered through the fourth-order Hill tensor $\mathbb{P}_r^{*,0}$, which depends on the relaxation function $\mathbb{R}^{*,0}$ of the surrounding phase r. There are two options for $\mathbb{R}^{*,0}$ to describe the interaction between the phases: (i) the self-consistend scheme [35], [34] with $\mathbb{R}^{*,0} = \mathbb{R}^{*,est}$, represents a dispersed arrangement of the phases, typical for polycrystals, and (ii) the Mori-Tanaka scheme [55], [76], with $\mathbb{R}^{*,0}$ coinciding with one of the phase stiffnesses, considers a contiguous matrix with inclusions, typical for composite materials.

Considering the relaxation function in Eq. (3.7), the homogenization of an RVE of extrafibrillar space can be carried out on the basis of a p-related, continuos sequence of formally elastic problems of the type of Eq. (3.9), delivering homogenized relaxation tensors $\mathbb{R}^{est}(p)$. Because of the disordered arrangement of the hydroxyapatite needles [25], a self-consistent scheme is used to determine the relaxation function of the foam, $\mathbb{R}_{ef}^{*,SCS}(p)$. Therefore, Eq. (3.9) is specified for $r \in [HA, ic]$, $\mathbb{r}_{HA}^* = 3k_{HA}\mathbb{I}_{vol} + 2\mu_{HA}^*(p)\mathbb{I}_{dev}$, $\mathbb{c}_{ic} = \mathbb{r}_{ic}^* = 3k_{H_2O}\mathbb{I}_{vol}$ [see Table 3.1 for k_{HA} and k_{H_2O}, and Eq. (3.8) for $\mu_{HA}^*(p)$], for $\mathbb{R}^{*,0} = \mathbb{R}_{ef}^{*,SCS}(p)$, as well as for $\mathbb{P}_{HA}^{*,0} = \mathbb{P}_{cyl}^{*,ef}(p)$ and $\mathbb{P}_{ic}^{*,0} = \mathbb{P}_{sph}^{*,ef}(p)$; resulting in

$$
\begin{aligned}
&\mathbb{R}_{ef}^{*,SCS}(p) = \\
&\left\{ \check{f}_{HA} \mathbb{r}_{HA}^{*} : \left\langle \left[\mathbb{I} + \mathbb{P}_{cyl}^{*,ef}(p) : \left(\mathbb{r}_{HA}^{*} - \mathbb{R}_{ef}^{*,SCS}(p) \right) \right]^{-1} \right\rangle \right. \\
&\left. + \check{f}_{ic} \mathbb{c}_{ic} : \left[\mathbb{I} + \mathbb{P}_{sph}^{*,ef}(p) : \left(\mathbb{c}_{ic} - \mathbb{R}_{ef}^{*,SCS}(p) \right) \right]^{-1} \right\} : \\
&\left\{ \check{f}_{HA} \left\langle \left[\mathbb{I} + \mathbb{P}_{cyl}^{*,ef}(p) : \left(\mathbb{r}_{HA}^{*} - \mathbb{R}_{ef}^{*,SCS}(p) \right) \right]^{-1} \right\rangle \right. \\
&\left. + \check{f}_{ic} \left[\mathbb{I} + \mathbb{P}_{sph}^{*,ef}(p) : \left(\mathbb{c}_{ic} - \mathbb{R}_{ef}^{*,SCS}(p) \right) \right]^{-1} \right\}^{-1}
\end{aligned}
\tag{3.10}
$$

with the angular average

$$
\begin{aligned}
&\left\langle \left[\mathbb{I} + \mathbb{P}_{cyl}^{*,ef}(p) : \left(\mathbb{r}_{HA}^{*} - \mathbb{R}_{ef}^{*,SCS}(p) \right) \right]^{-1} \right\rangle \\
&= \int_{\varphi=0}^{2\pi} \int_{\vartheta=0}^{\pi} \left[\mathbb{I} + \mathbb{P}_{cyl}^{*,ef}(p)(\vartheta,\varphi) : \left(\mathbb{r}_{HA}^{*} - \mathbb{R}_{ef}^{*,SCS}(p) \right) \right]^{-1} \frac{\sin\vartheta d\vartheta d\varphi}{4\pi}.
\end{aligned}
\tag{3.11}
$$

Table 3.1: Phase stiffness values

Phase	Bulk modulus k [GPa]	Shear modulus μ [GPa]	Experimental source
hydroxyapatite	$k_{HA} = 82.6$	$\mu_{HA} = 44.9$	[41]
water containing non-collageneous organics or osteocytes	$k_{H_2O} = 2.3$	$\mu_{H_2O} = 0$	
	c_{ijkl} [GPa]	c_{ijkl} [GPa]	
Collagen	$r_{col,3333} = 17.9$ $r_{col,1111} = 11.7$	$r_{col,1133} = 7.1$ $r_{col,1122} = 5.1$ $r_{col,1313} = 3.3$	[18]

The integrals in Eq. (3.11) can be solved very efficiently with the help of Stroud's integration formula [71], [61]

$$
\begin{aligned}
&\int_{\varphi=0}^{2\pi} \int_{\vartheta=0}^{\pi} \left[\mathbb{I} + \mathbb{P}_{cyl}^{*,ef}(p)(\vartheta,\varphi) : \left(\mathbb{r}_{HA}^{*} - \mathbb{R}_{ef}^{*,SCS}(p) \right) \right]^{-1} \frac{\sin\vartheta d\vartheta d\varphi}{4\pi} \\
&= \sum_{j=0}^{n} \omega(\vartheta_j,\varphi_j) \left[\mathbb{I} + \mathbb{P}_{cyl}^{*,ef}(p)(\vartheta_j,\varphi_j) : \left(\mathbb{r}_{HA}^{*} - \mathbb{R}_{ef}^{*,SCS}(p) \right) \right]^{-1}.
\end{aligned}
\tag{3.12}
$$

The scalar weight $\omega(\vartheta_j, \varphi_j)$ and the orientations ϑ_j and φ_j are defined in Table 3.2.

Table 3.2: Scalar weights $\omega(\vartheta_j, \varphi_j)$ and orientations ϑ_j and φ_j for Stroud's integration [61]

j	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$\sin(\vartheta_j)\cos(\varphi_j)$	$+r$	$+r$	$-r$	$-r$	$+t$	$+t$	$-t$	$-t$	$+s$	$+s$	$-s$	$-s$	1	0	0
$\sin(\vartheta_j)\sin(\varphi_j)$	$+s$	$-s$	$+s$	$-s$	$+r$	$-r$	$+r$	$-r$	$+t$	$-t$	$+t$	$-t$	0	1	0
$\cos(\vartheta_j)$	$+t$	$+t$	$+t$	$+t$	$+s$	$+s$	$+s$	$+s$	$+r$	$+r$	$+r$	$+r$	0	0	1
$\omega(\vartheta_j, \varphi_j)$	$\frac{1}{15}$	$\frac{1}{15}$	$\frac{1}{15}$	$\frac{1}{15}$	$\frac{1}{15}$	$\frac{1}{15}$	$\frac{1}{15}$	$\frac{1}{15}$	$\frac{1}{15}$	$\frac{1}{15}$	$\frac{1}{15}$	$\frac{1}{15}$	$\frac{1}{15}$	$\frac{1}{15}$	$\frac{1}{15}$

$$\text{with} \quad r = 1/2, \quad s = (\sqrt{5}+1)/4 \quad \text{and} \quad t = (\sqrt{5}-1)/4$$

The volume fractions $\check{f}_{HA}$ and $\check{f}_{ic}$ fulfill $\check{f}_{HA} + \check{f}_{ic} = 1$. For the determination of $\check{f}_{HA}$ and $\check{f}_{ic}$ and all volume fractions in the following subsections, we refer to [24].

Since the overall hydroxyapatite foam is isotropic, the Hill tensor, $\mathbb{P}^{ef}_{sph}(p)$ for a spherical inclusion in an isotropic material of stiffness $\mathbb{R}^{*,SCS}_{ef}(p)$ reads as [79], [20]

$$\mathbb{P}^{*,ef}_{sph}(p) = \mathbb{S}^{*,Esh,ef}_{sph}(p) : \mathbb{R}^{*,SCS,-1}_{ef}(p)\,, \tag{3.13}$$

$$\mathbb{S}^{*,Esh,ef}_{sph}(p) = \alpha^{*,SCS}_{ef}(p)\,\mathbb{I}_{vol} + \beta^{*,SCS}_{ef}(p)\,\mathbb{I}_{dev} \tag{3.14}$$

with

$$\begin{aligned} \alpha^{*,SCS}_{ef}(p) &= \frac{3\,k^*_{ef}(p)}{3\,k^*_{ef}(p) + 4\,\mu^*_{ef}(p)} \\ \beta^{*,SCS}_{ef}(p) &= \frac{6\,[k^*_{ef}(p) + 2\,\mu^*_{ef}(p)]}{5\,[3\,k^*_{ef}(p) + 4\,\mu^*_{ef}(p)]}. \end{aligned} \tag{3.15}$$

The Hill tensor, $\mathbb{P}^{*,ef}_{cyl}(p)$, for cylindrical inclusions in an isotropic material can be obtained from

$$\mathbb{P}^{*,ef}_{cyl}(p) = \mathbb{S}^{*,Esh,ef}_{cyl}(p) : \mathbb{R}^{*,SCS,-1}_{ef}(p). \tag{3.16}$$

The Eshelby tensor, $\mathbb{S}^{*,Esh,ef}_{cyl}(p)$, is standardly defined in a local coordinate system attached to individual solid needles (as depicted in Fig. 3.4). The corresponding non-zero components read as [23], [20]

$$
\begin{aligned}
S^{*,esh}_{cyl,1111}(p) = S^{*,esh}_{cyl,2222}(p) &= \frac{5-4\nu^{*,SCS}_{ef}(p)}{8[1-\nu^{*,SCS}_{ef}(p)]} \\
S^{*,esh}_{cyl,1122}(p) = S^{*,esh}_{cyl,2211}(p) &= \frac{-1+4\nu^{*,SCS}_{ef}(p)}{8[1-\nu^{*,SCS}_{ef}(p)]} \\
S^{*,esh}_{cyl,1133}(p) = S^{*,esh}_{cyl,2233}(p) &= \frac{\nu^{*,SCS}_{ef}(p)}{2[1-\nu^{*,SCS}_{ef}(p)]} \\
S^{*,esh}_{cyl,2323}(p) = S^{*,esh}_{cyl,3232}(p) &= \frac{1}{4} \\
S^{*,esh}_{cyl,3223}(p) = S^{*,esh}_{cyl,2332}(p) &= \frac{1}{4} \\
S^{*,esh}_{cyl,3131}(p) = S^{*,esh}_{cyl,1313}(p) &= \frac{1}{4} \\
S^{*,esh}_{cyl,1331}(p) = S^{*,esh}_{cyl,3113}(p) &= \frac{1}{4} \\
S^{*,esh}_{cyl,1212}(p) = S^{*,esh}_{cyl,2121}(p) &= \frac{3-4\nu^{*,SCS}_{ef}(p)}{8[1-\nu^{*,SCS}_{ef}(p)]} \\
S^{*,esh}_{cyl,2112}(p) = S^{*,esh}_{cyl,1221}(p) &= \frac{3-4\nu^{*,SCS}_{ef}(p)}{8[1-\nu^{*,SCS}_{ef}(p)]},
\end{aligned}
\tag{3.17}
$$

with the foam's Poisson's ratio reading as

$$
\nu^{*,SCS}_{ef}(p) = \frac{3k^{*}_{ef}(p) - 2\mu^{*}_{ef}(p)}{6k^{*}_{ef}(p) + 2\mu^{*}_{ef}(p)}. \tag{3.18}
$$

To homogenize over the RVE, the Hill tensors, related to differently oriented hydroxyapatite needles are transformed into one, single base frame, using [57]

$$
\mathbb{P}^{*,ef}_{cyl}(p,\varphi,\vartheta) = Q(\varphi,\vartheta)\mathbb{P}^{*,ef}_{cyl}(p)Q^{t}(\varphi,\vartheta) \tag{3.19}
$$

with

$$
Q(\varphi,\vartheta) = \begin{bmatrix}
q^2_{11} & q^2_{12} & q^2_{13} & \frac{2}{\eta}q_{12}q_{13} & \frac{2}{\eta}q_{13}q_{11} & \frac{2}{\eta}q_{11}q_{12} \\
q^2_{21} & q^2_{22} & q^2_{23} & \frac{2}{\eta}q_{22}q_{23} & \frac{2}{\eta}q_{23}q_{21} & \frac{2}{\eta}q_{21}q_{22} \\
q^2_{31} & q^2_{32} & q^2_{33} & \frac{2}{\eta}q_{32}q_{33} & \frac{2}{\eta}q_{33}q_{31} & \frac{2}{\eta}q_{31}q_{32} \\
\eta q_{21}q_{31} & \eta q_{22}q_{32} & \eta q_{23}q_{33} & q_{23}q_{32}+q_{33}q_{22} & q_{21}q_{33}+q_{31}q_{23} & q_{22}q_{31}+q_{32}q_{21} \\
\eta q_{31}q_{11} & \eta q_{32}q_{12} & \eta q_{33}q_{13} & q_{33}q_{12}+q_{13}q_{32} & q_{31}q_{13}+q_{11}q_{33} & q_{32}q_{11}+q_{12}q_{31} \\
\eta q_{11}q_{21} & \eta q_{12}q_{22} & \eta q_{13}q_{23} & q_{13}q_{22}+q_{23}q_{12} & q_{11}q_{23}+q_{21}q_{13} & q_{12}q_{21}+q_{22}q_{11}
\end{bmatrix}
\tag{3.20}
$$

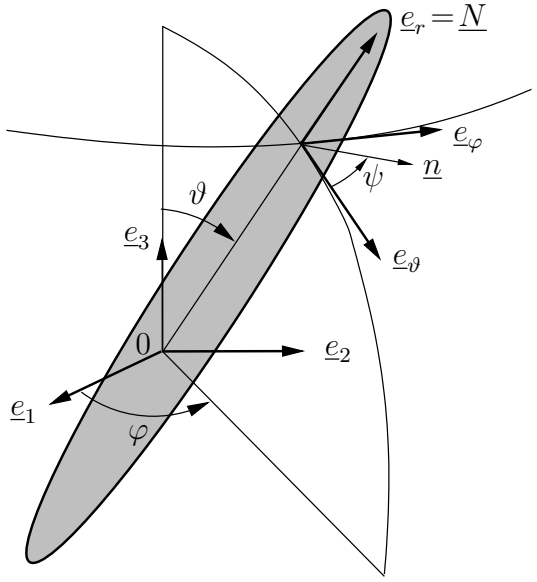

Figure 3.4: Cylindrical HA-inclusion [25]

and

$$q_{ij} = [\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3], \qquad i = 1 \ldots 3, j = 1 \ldots 3, \tag{3.21}$$

$$\begin{aligned} \mathbf{e}_1 &= \begin{bmatrix} \cos\varphi\cos\vartheta \\ \sin\varphi\cos\vartheta \\ -\sin\varphi \end{bmatrix} \\ \mathbf{e}_2 &= \begin{bmatrix} -\sin\varphi \\ \cos\varphi \\ 0 \end{bmatrix} \\ \mathbf{e}_3 &= \begin{bmatrix} \cos\varphi\sin\vartheta \\ \sin\varphi\sin\vartheta \\ \cos\varphi \end{bmatrix}. \end{aligned} \tag{3.22}$$

3.2 Homogenization of wet collagen

An RVE $\mathring{V}_{wetcol}$ of wet collagen is about $\ell_{wetcol} = 10$ nm long (Fig. 3.2(a), [37], [54]). It consists of a contiguous matrix of crosslinked molecular collagen (labeled by suffix "col"), penetrated by cylindrical intermolecular pores (labeled as "im"). These inclusions are filled with water containing mechanically insignificant amounts of non-collageneous organic matter. Because of this morphology, the use of a Mori-Tanaka scheme is considered for the estimation of the stiffness $\mathbb{R}^{*,MT}_{wetcol}$ of wet collagen, using Eq. (3.9), with $r \in [col, im]$, $\mathbb{c}_{im} = \mathbb{r}^*_{im} = 3k_{H_2O}\mathbb{I}_{vol}$, $\mathbb{R}^{*,0} = \mathbb{c}_{col} = \mathbb{r}^*_{col}$ according to Table 3.1, and $\mathbb{P}^{*,0}_{im} = \mathbb{P}^{col}_{cyl}$; resulting in

$$\begin{aligned}\mathbb{R}^{*,MT}_{wetcol} &= \left\{(1-\mathring{f}_{im})\mathbb{c}_{col} + \mathring{f}_{im}\mathbb{c}_{im} : \left[\mathbb{I} + \mathbb{P}^{col}_{cyl} : (\mathbb{c}_{im} - \mathbb{c}_{col})\right]^{-1}\right\} : \\ &\quad \left\{(1-\mathring{f}_{im})\mathbb{I} + \mathring{f}_{im}\left[\mathbb{I} + \mathbb{P}^{col}_{cyl} : (\mathbb{r}^{*}_{im} - \mathbb{c}_{col})\right]^{-1}\right\}^{-1}. \end{aligned} \tag{3.23}$$

The material is transversely isotropic. The fourth-order Hill tensor $\mathbb{P}^{col}_{cyl}$ is used for cylindrical inclusions in a transversely isotropic matrix with stiffness $\mathbb{r}^{*}_{col}$ and the plane of isotropy perpendicular to the long axis of the cylinder. $\mathbb{P}^{col}_{cyl}$ is symmetric, the non-zero components read as [24]

$$\begin{aligned} P^{col}_{cyl,1111} &= P^{col}_{cyl,2222} = 1/8\,(5\,r^{*}_{col,1111} - 3\,r^{*}_{col,1122})/r^{*}_{col,1111}/\mathcal{D}_2\,, \\ P^{col}_{cyl,1122} &= P^{col}_{cyl,2211} = -1/8\,(r^{*}_{col,1111} + r^{*}_{col,1122})/r^{*}_{col,1111}/\mathcal{D}_2\,, \\ P^{col}_{cyl,2323} &= P^{col}_{cyl,1313} = 1/(8\,r^{*}_{col,2323})\,, \\ P^{col}_{cyl,1212} &= 1/8\,(3\,r^{*}_{col,1111} - r^{*}_{col,1122})/r^{*}_{col,1111}/\mathcal{D}_2\,, \end{aligned} \tag{3.24}$$

whereby

$$\mathcal{D}_2 = r^{*}_{col,1111} - r^{*}_{col,1122} \tag{3.25}$$

The volume fractions $\mathring{f}_{col}$ and $\mathring{f}_{im}$ fulfill $f_{col} + f_{im} = 1$.

3.3 Homogenization of mineralized collagen fibril

An RVE $\breve{V}_{fib}$ of fibrillar material [as depicted in Fig. 3.2 (b)] has a characteristic length of ℓ_{fib} between 100 and 500 nm [18], [54]. It is built up by hydroxyapatite crystal agglomerations ("HA") and microfibrils consisting of wet collagen ("wetcol"; see Subsection 3.2), interpenetrating each other. This suggests the use of a self-consistent scheme with spherical inclusions for the mineral phase, and cylindrical inclusions for the wet collagen. Hence, Eq. (3.9) is applied to calculate the stiffness of mineralized collagen fibril, $\mathbb{R}^{*,SCSII}_{fib}$, with $r \in [HA, wetcol]$, for $\mathbb{c}_{HA} = \mathbb{r}^{*}_{HA} = 3k_{HA}\mathbb{I}_{vol} + 2\mu_{HA}\mathbb{I}_{dev}$, k_{HA} and μ_{HA} according to Table 3.1, for $\mathbb{r}^{*}_{wetcol} = \mathbb{R}^{*,MT}_{wetcol}$ [see Eq. (3.23)] and for $\mathbb{R}^{*,0} = \mathbb{R}^{*,SCSII}_{fib}$, as well as for $\mathbb{P}^{*,0}_{HA} = \mathbb{P}^{fib}_{sph}$ and $\mathbb{P}^{*,0}_{HA} = \mathbb{P}^{fib}_{cyl}$; yielding

$$\begin{aligned} &\mathbb{R}^{*,SCSII}_{fib} = \\ &\left\{\breve{f}_{wetcol}\,\mathbb{R}^{*,MT}_{wetcol} : \left[\mathbb{I} + \mathbb{P}^{fib}_{cyl} : \left(\mathbb{R}^{*,MT}_{wetcol} - \mathbb{R}^{*,SCSII}_{fib}\right)\right]^{-1}\right. \\ &\left. + \breve{f}_{HA}\,\mathbb{c}_{HA} : \left[\mathbb{I} + \mathbb{P}^{fib}_{sph} : \left(\mathbb{r}^{*}_{HA} - \mathbb{R}^{*,SCSII}_{fib}\right)\right]^{-1}\right\} : \\ &\left\{\breve{f}_{wetcol}\left[\mathbb{I} + \mathbb{P}^{fib}_{cyl} : \left(\mathbb{R}^{*,MT}_{wetcol} - \mathbb{R}^{*,SCSII}_{fib}\right)\right]^{-1}\right. \\ &\left. + \breve{f}_{HA}\left[\mathbb{I} + \mathbb{P}^{fib}_{sph} : \left(\mathbb{c}_{HA} - \mathbb{R}^{*,SCSII}_{fib}\right)\right]^{-1}\right\}^{-1} \end{aligned} \tag{3.26}$$

Mineralized collagen fibrils are transversely isotropic. The non-zero components of $\mathbb{P}_{sph}^{fib}$ can be determined by using $R_{fib,ijkl}^{*,SCSII}$ instead of R_{ijkl}^{0} in Eqs. (A.1)-(A.7), see appendix A. By replacing $r_{col,ijkl}^{*}$ by $R_{fib,ijkl}^{*,SCSII}$ in Eq. (3.24), the non-zero components of $\mathbb{P}_{cyl}^{fib}$ can be ascertained. The volume fractions $\breve{f}_{HA}$ and $\breve{f}_{wetcol}$ fulfill $\breve{f}_{HA} + \breve{f}_{wetcol} = 1$. See [29] for implicit solution of Eq. (3.26).

3.4 Homogenization of extracellular bone matrix

An RVE $\bar{V}_{ultra}$ of extracellular bone matrix, or bone ultrastructure, (see Fig. 3.2) has a characteristic length ℓ_{ultra} between 5 and 10 μm. The cylindrical mineralized fibrils (labelled by suffix "fib"; see Subsections 3.3) are surrounded by a contiguous matrix of the viscoelastic hydroxyapatite foam (labelled by suffix "ef"; see Subsections 3.1). This morphology motivates the use of a Mori-Tanaka scheme to estimate the relaxation function of the extracellular bone matrix, $\mathbb{R}_{ultra}^{*,MTII}(p)$, considering Eq. (3.9), with $r \in [fib, ef]$, for $\mathbb{c}_{fib} = \mathbb{r}_{fib}^{*} = \mathbb{R}_{fib}^{*,SCSII}$ from Eq. (3.26), for $\mathbb{R}^{*,0} = \mathbb{r}_{ef}^{*} = \mathbb{R}_{ef}^{*,SCS}(p)$ from Eq. (3.10), as well as for $\mathbb{P}_{fib}^{*,0} = \mathbb{P}_{cyl}^{*,ef}(p)$; resulting in

$$\begin{aligned}\mathbb{R}_{ultra}^{*,MTII}(p) \;=\; & \left\{(1-\bar{f}_{fib})\mathbb{R}_{ef}^{*,SCS}(p) + \bar{f}_{fib}\mathbb{R}_{fib}^{*,SCSII} : \left[\mathbb{I} + \mathbb{P}_{*,cyl}^{ef}(p) : \left(\mathbb{R}_{fib}^{*,SCSII} - \mathbb{R}_{ef}^{*,SCS}(p)\right)\right]^{-1}\right\} : \\ & \left\{(1-\bar{f}_{fib})\mathbb{I} + \bar{f}_{fib}\left[\mathbb{I} + \mathbb{P}_{*,cyl}^{ef}(p) : \left(\mathbb{R}_{fib}^{*,SCSII} - \mathbb{R}_{ef}^{*,SCS}(p)\right)\right]^{-1}\right\}^{-1}. \end{aligned} \tag{3.27}$$

The ultrastructure material is transversely isotropic. The volume fractions $\bar{f}_{fib}$ and $\bar{f}_{ef}$ satisfy $\bar{f}_{fib} + \bar{f}_{ef} = 1$, the non-zero components of $\mathbb{P}_{cyl}^{*,ef}(p)$ follow from replacing $r_{col,ijkl}$ by $R_{ef,ijkl}^{*,SCS}(p)$ in Eq. (3.24).

3.5 Homogenization of extravascular bone material

An RVE $\tilde{V}_{exvas}$ of extravascular bone material [see Fig. 3.2 (e)] is about $\ell_{exvas} = 100$ μm long. It is built up by a contiguous matrix of extracellular bone material (suffix "ultra"; see Subsection 3.4), with embedded spherical pores, called lacunae (suffix "lac"), which are filled with water including osteocytes. Due to unsealed conditions, the water in the pores is pressed out during testing. Therefore, the stiffness of the lacunae is vanishing. Eq. (3.9) is applied to calculate the relaxation function of extravascular bone material, $\mathbb{R}_{exvas}^{*,MTIII}(p)$, using a Mori-Tanaka scheme: for $r \in [ultra, lac]$, for $\mathbb{c}_{lac} = \mathbb{r}_{lac}^{*} = \mathbb{0}$, for $\mathbb{R}^{*,0} = \mathbb{r}_{ultra}^{*} = \mathbb{R}_{ultra}^{*,MTII}(p)$ from Eq. (3.27), and for $\mathbb{P}_{lac}^{*,0} = \mathbb{P}_{sph}^{*,ultra}(p)$; resulting in

$$\mathbb{R}^{*,MTIII}_{exvas}(p) = \left\{ \tilde{f}_{ultra}\mathbb{R}^{*,MTII}_{ultra}(p) + \tilde{f}_{lac}\,\mathbb{c}_{lac} : \left[\mathbb{I} + \mathbb{P}^{*,ultra}_{sph}(p) : \left(\mathbb{c}_{lac} - \mathbb{R}^{*,MTII}_{ultra}(p)\right)\right]^{-1}\right\} : \left\{ \tilde{f}_{ultra}\,\mathbb{I} + \tilde{f}_{lac}\left[\mathbb{I} + \mathbb{P}^{*,ultra}_{sph}(p) : \left(\mathbb{c}_{lac} - \mathbb{R}^{*,MTII}_{ultra}(p)\right)\right]^{-1}\right\}^{-1}. \quad (3.28)$$

The extravascular bone material is transversely isotropic as well. The volume fractions $\tilde{f}_{ultra}$ and $\tilde{f}_{lac}$ fulfill $\tilde{f}_{ultra} + \tilde{f}_{lac} = 1$; the non-zero components of $\mathbb{P}^{*,ultra}_{sph}(p)$ can obtained from substitution of R^0_{ijkl} by $R^{*,MTII}_{ultra,ijkl}(p)$ in Eqs. (A.1)-(A.7), see appendix A.

3.6 Homogenization of cortical bone material

An RVE V_{cort} of cortical bone material has a characteristic length of about 1 mm. A matrix of extravascular bone material (labelled "exvas"; see Subsection 3.5) is penetrated by cylindrical water-filled pores, called Haversian canals or vascular pores (labelled "vas"). As in the lacunar pores, the water is pressed out in the Haversian canals, as well. Hence, the stiffness also vanishes. The morphology motivates the use of a Mori-Tanaka scheme for homogenization. Applying Eq. (3.9) with $r \in [exvas, vas]$, $\mathbb{c}_{vas} = \mathbb{r}^*_{vas} = \mathbb{0}$, $\mathbb{R}^{*,0} = \mathbb{r}^*_{exvas} = \mathbb{R}^{*,MTIII}_{exvas}(p)$ from Eq. (3.28), as well as $\mathbb{P}^{*,0}_{vas} = \mathbb{P}^{*,exvas}_{cyl}(p)$, leads to

$$\mathbb{R}^{*,MTIV}_{cort}(p) = \left\{ f_{exvas}\mathbb{R}^{*,MTIII}_{exvas}(p) + f_{vas}\,\mathbb{c}_{vas} : \left[\mathbb{I} + \mathbb{P}^{*,exvas}_{cyl}(p) : \left(\mathbb{c}_{vas} - \mathbb{R}^{*,MTIII}_{exvas}(p)\right)\right]^{-1}\right\} : \left\{ f_{exvas}\,\mathbb{I} + f_{lac}\left[\mathbb{I} + \mathbb{P}^{*,exvas}_{cyl}(p) : \left(\mathbb{c}_{vas} - \mathbb{R}^{*,MTIII}_{exvas}(p)\right)\right]^{-1}\right\}^{-1}. \quad (3.29)$$

The volume fractions f_{exvas} and f_{vas} fulfill $f_{exvas} + f_{vas} = 1$. The cortical bone material is transversely isotropic, the Hill tensor, $\mathbb{P}^{*,exvas}_{cyl}(p)$ follows from substitution of $r_{col,ijkl}$ by $R^{*,MTIII}_{exvas,ijkl}(p)$ in Eq. (3.24).

To be able to give the predicted relaxation functions of cortical bone from Eq. (3.29) in a physically relevant format, we have to back-transform the LC-transformed material function $\mathbb{R}^{*,MTIV}_{cort}(p)$ into the time domain. The use of a numerical algorithm is obligatory, since the self-consistent scheme, used for homogenization of the extrafibrillar space and the mineralized collagen fibrils in Subsections 3.1 and 3.3 is implicit, and therefore, $\mathbb{R}^{*,MTIV}_{cort}(p)$ is only pointwisely available for any value of the complex variable p. To fulfill this requirement, we use the numerical Gaver-Wynn-Rho (GWR) algorithm presented by Abate and Valkó [2], Valkó and Abate [74] and Scheiner and Hellmich [68].

GWR algorithm requires an increasing calculation precision of the LC-transforms with increasing desired accuracy of the inversion of the transformation, but only to a limit, given by the computing precision of the LC-transformed homogenized material functions [68]. This is achieved by using a multiprecision computational environment for determination of the LC-transforms, as provided by the software tool *Maple* [1].

The GWR algorithm is based on the famous Post-Widder formula [77], which constitutes an analytical expression whose limit ($k \to \infty$) tends to the inverse of the LC-transformed function $f^*(p)$, i.e., to the time-dependent function $f(t)$

$$f_k(t) = \frac{(-1)^k}{k!} p^{k+1} \frac{d^k}{dp^k} \hat{f}(p) \to f(t) \text{ for } k \to \infty \tag{3.30}$$

where $\hat{f}(p)$ is the Laplace transform, being related to LC transform by $f^*(p) = p\hat{f}(p)$; see Eq. (3.4). As a consequence of difficulties with the calculation of the higher-order derivatives, Gaver [26] came up with the so-called Gaver functionals, a discrete form of Eq. (3.30), reading as

$$f_k(t) = \frac{(-1)^k \alpha (2k)!}{tk!(k-1)!} \Delta^k [\hat{f}(k\alpha/t)]. \tag{3.31}$$

Δ represents the forward difference operator $\Delta f(nx) = f[(n+1)x] - f(nx)$, α is $\ln(2)$ and p was replaced by $p = k/t$. Expanding the difference operator results in [3]

$$f_k(t) = \frac{\alpha (2k)!}{tk!(k-1)!} \sum_{j=0}^{k} (-1)^j \frac{k! \hat{f}[(k+j)\alpha/t}{j!(k-j)!}. \tag{3.32}$$

The use of an acceleration algorithm leads to a faster convergence of Eq. (3.32). According to [2] and [74], the Wynn-Rho algorithm,

$$f(t) \approx f(t, M) = \rho_M^{(0)} \tag{3.33}$$

yields the best result, on the basis of the recursive algorithm

$$\rho_n^{(k)} = \begin{cases} 0 & \text{for } k = -1 \\ f_n(t) & \text{for } k = 0 \\ \rho_{k-2}^{(n+1)} + \frac{k}{\rho_{k-1}^{(n+1)} - \rho_{k-1}^{(n)}} & \text{for } k \geq 1 \end{cases} \tag{3.34}$$

where $n \geq 0$ and k running through $k = -1, 0, ..., M - n$, for each n out of $n = M, M - 1, ..., 0$.

The precision requirement, according to [2] and [74], reading as

$$\text{number of precision decimal digits } \geq 2.1M \tag{3.35}$$

determines the computing precision for a chosen M.

In Chapter 4 and 5, $M = 4$ delivers the appropriate approximations.

3.7 Elastic properties of the elementary constituents of bone

The phase stiffness values, listed in Table 3.1, are the same for all tissues dealt with in this manuscript. These properties were achieved by the following experiments [25]:

- The isotropic elastic properties of hydroxyapatite were gained from tests with an ultrasonic interferometer coupled with a solid media pressure apparatus [41], [27]. Hydroxyapatite powder was examined because of largely disordered arrangement of minerals [49],[22],[60],[31].
- Since there are no direct measurements of the elastic properties of molecular collagen, the stiffness values of dry rat tendon, which consists almost exclusively of collagen, were used instead. The five independent elastic constants, needed to build up the stiffness matrix of a transversely isotropic material, were achieved using Brillouin light scattering [18].
- The elastic behavior of water with mechanically insignificant amounts of non-collagenous organic matter is described by the standard bulk modulus of water.

Chapter 4

Identification of creep properties

4.1 Determination of initial plastic strains

With the help of the multiscale micromechanics model introduced in Chapter 3, the initial plastic strains described in Subsections 2.3 and 2.4 can be determined.

Using the volume fractions of bovine tibia according to Table 4.1, available in [24] and [48], the microelastoplastic model presented in Fritsch et al. [25] is able to predict the macroscopic stress-strain diagram, depicted in Fig. 4.1. Accordingly, the yield stress is around 0.049%, which coincides with the strain when the contact strength between the hydroxyapatite crystals is reached (mentioned in Subsection 2.3) and is significantly lower than the value of 0.23%, calculated by Iyo et al [39]. This confirms the assumption that there are plastic deformations at the bottom of the center of the specimen.

Table 4.1: Volume fractions for bovine tibia [24] [48]

ρ_{wet}	$\bar{f}_{col}$	$\bar{f}_{HA}$	$\bar{f}_{fib}$	$\check{f}_{HA}$	$\breve{f}_{HA}$	$\mathring{f}_{col}$	f_{vas}	f_{lac}
1.99	0.29	0.46	0.51	0.65	0.28	9.41	0.05	0.02

In order to validate this assumption, the elastoplastic stress and strain distributions in the cross-section directly under the applied load were ascertained. Therefore, the load level has to be determined and, for that purpose, the relaxation forces were back-calculated from the initial Young's modulus given in [39], using elasticity theory: From the maximum bending moment of

$$M_0 = \frac{F_0 l}{4}, \tag{4.1}$$

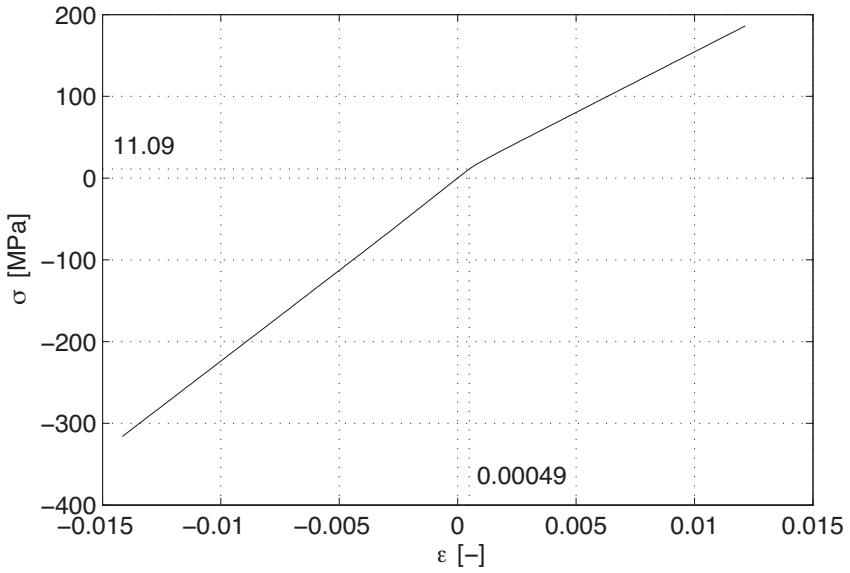

Figure 4.1: Macroscopic stress-strain diagram for bovine tibia

the longitudinal normal stress component

$$\sigma_0(z = h/2) = \frac{M_0 z}{I} = E_0 \varepsilon(z = h/2) \tag{4.2}$$

with z as the distance between the neutral axis of the stress distribution and the bottom of the specimen, the maximum longitudinal strain $\varepsilon(z = h/2) = 0.0023$, and the moment of inertia

$$I = \frac{bh^3}{12}, \tag{4.3}$$

can be calculated.

From Eqs. (4.1) and (4.2), the relaxation force can be derived as

$$F_0 = \frac{4E_0\varepsilon(z = h/2)I}{l(h/2)}. \tag{4.4}$$

With the help of the stress resultants

$$N_0 = \int_A \sigma_0(z) dA = 0, \tag{4.5}$$

$$M_0 = \int_A \sigma_0(z) z dA \tag{4.6}$$

and the yield stress and yield strain from the stress-strain diagram (Fig. 4.1), the elastoplastic strain and stress distributions over the considered cross-section were ascertained (solid lines in Fig. 4.2).

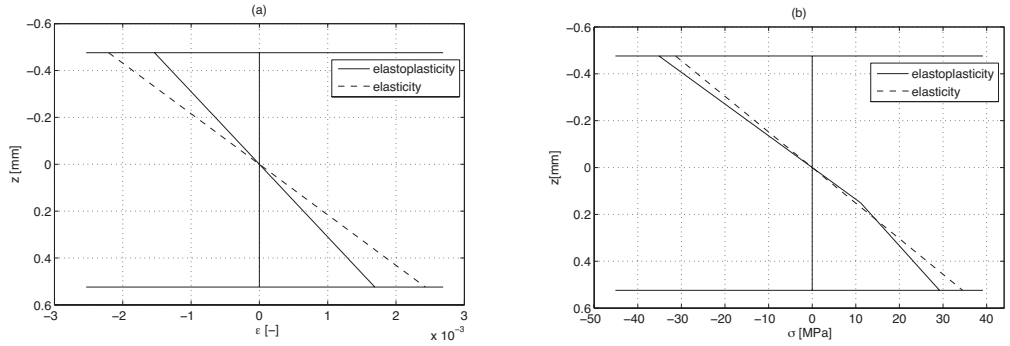

Figure 4.2: Strain (a) and stress (b) distribution in the cross-section under the applied load for F_0

The dashed lines in Fig. 4.2 refer to calculations based on elasticity theory, as considered by Iyo et al. [39].

Although the elastic and the elastoplastic distributions are determined at the same load level, the strains in the first case are larger. This indicates that Young's moduli for bone material based on elasticity theory are lower than values calculated considering elastoplastic theory.

According to Subsection 2.4 plastic deformations at the supports are very probable. To validate this assumption, the extent of the deflection w can be calculated by means of

$$w = \frac{F_0 l^3}{48 E_0 I}, \tag{4.7}$$

where F_0 is the initial force and E_0 denotes the the initial Young's modulus.

Using $E_0^{exp} = 14.2$ GPa and $\mathrm{E}_0^{mod} = 22.89$ GPa, respectively, Eq. (4.7) provides the so called experimental deflection w^{exp} and the elastic deflection w^{el}. The plastic deflection is then obtained as

$$w^{pl} = w^{exp} - w^{el}. \tag{4.8}$$

w^{pl} is about 38 % of w^{exp}, which is a clear indicator for the occurrence of plastic deformations at the supports.

Another evidence for this phenomenon is the fact, that modulus of elasticity in transverse direction is more or less the same in the experiments by Iyo et al. [39] and in our calculations. The respective values are 11.6 GPa and 12.65 GPa, whereas for the longitudinal direction Young's moduli of 14.2 and 22.89 GPa were obtained. This phenomenon could be explained by the arrangement of the collagen fibrils, which are aligned in longitudinal

direction in the extracellular bone matrix, and the plastic behavior of the hydroxyapatite foam between the fibrils. In three-point bending tests on samples cut parallel to the bone axis, the influence of the hydroxyapatite foam is much larger than in tests on transverse samples. This illustrates, why plastic deformations and therefore, the difference between experimental results and model estimates of longitudinal samples are larger compared to the ones of transversely loaded specimens.

4.2 Determination of the Burgers parameters $\mu_{HA,KV}$, $\eta_{HA,KV}$ and $\eta_{HA,M}$

To determine the Burgers parameters $\mu_{HA,KV}$, $\eta_{HA,KV}$ and $\eta_{HA,M}$, data from experiments performed by Iyo et. al [39] was used (see Subsection 2.2).

As described in detail in Chapter 2, differences between E_0^{exp} and E_0^{mod} exist. Therefore, relaxation forces were used to determine $\mu_{HA,KV}$, $\eta_{HA,KV}$ and $\eta_{HA,M}$. These are unaffected by the mentioned divergences.

The experimental relaxation forces have already been determined in Subsection 2.5.

The time-dependent relaxation Young's modulus $R^{mod}(t)$ in longitudinal direction was calculated with the viscoelastic model presented in Chapter 3. With the help of the initial distribution of elastic strains in the cross-section under the applied load (Fig. 4.3) and the constitutive equation [28], [10], [66]

$$\boldsymbol{\sigma}(t,z) = \int_{-\infty}^{t} \mathbb{R}^{mod}(t-\tau) : \dot{\boldsymbol{\varepsilon}}(\tau,z)d\tau, \tag{4.9}$$

with z as the distance between the neutral axis of the stress distribution and the bottom of the specimen, the stress distributions at different times were calculated. Because viscoelasticity is examined, only elastic strains were used for relaxation (see Fig. 4.3).

There are no strains before the beginning of the experiment, therefore, the Heaviside-function H is used:

$$\boldsymbol{\varepsilon}(t,z) = H(t)\bar{\boldsymbol{\varepsilon}}(t,z), \tag{4.10}$$

where $\bar{\boldsymbol{\varepsilon}}(t,z)$ is the time-dependent function of strains for $-\infty \leq t \leq +\infty$. The time derivative of Eq. (4.10) is

$$\frac{d\boldsymbol{\varepsilon}(t,z)}{dt} = \delta(t)\bar{\boldsymbol{\varepsilon}}(t,z) + H(t)\frac{d\bar{\boldsymbol{\varepsilon}}(t,z)}{dt}. \tag{4.11}$$

During a relaxation test, strains are constant:

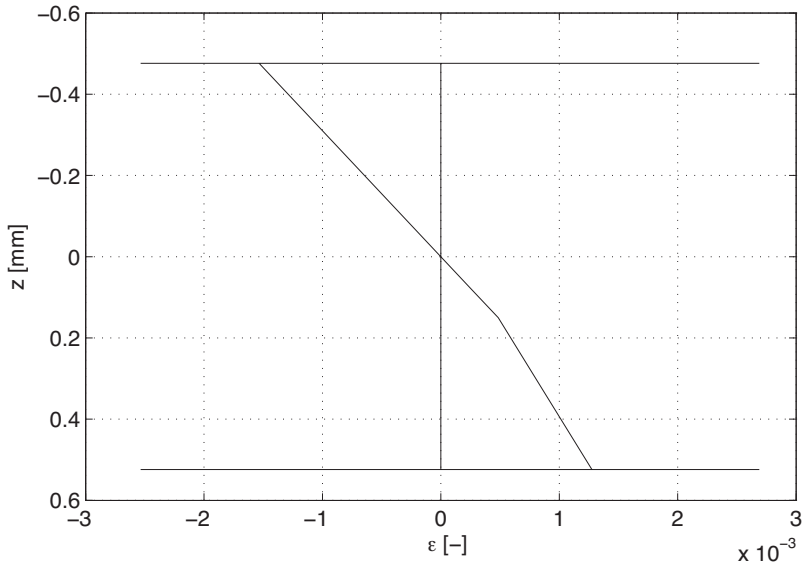

Figure 4.3: Initial distribution of elastic strains

$$\frac{d\bar{\boldsymbol{\varepsilon}}(t,z)}{dt} = 0, \tag{4.12}$$

so, Eq. (4.11) can be simplified to

$$\frac{d\boldsymbol{\varepsilon}(t,z)}{dt} = \delta(t)\bar{\boldsymbol{\varepsilon}}(t,z). \tag{4.13}$$

Eq. (4.13) can be applied to Eq. (4.9)

$$\boldsymbol{\sigma}(t,z) = \int_{-\infty}^{t} \mathbb{R}^{mod}(t-\tau) : \delta(\tau)\bar{\boldsymbol{\varepsilon}}(\tau,z)d\tau. \tag{4.14}$$

Because of the characteristics of the Dirac-δ-function, Eq. (4.14) can be simplified to

$$\boldsymbol{\sigma}(t,z) = \mathbb{R}^{mod}(t) : \bar{\boldsymbol{\varepsilon}}(t=0,z). \tag{4.15}$$

$\bar{\boldsymbol{\varepsilon}}(t=0,z)$ denotes the strain distribution at $t=0$ (see Fig. 4.3) and can be determined by means of

$$\bar{\boldsymbol{\varepsilon}}(t=0,z) = \boldsymbol{\varepsilon}^{ve}(t=0,z) = \boldsymbol{\varepsilon}(t=0,z) - \boldsymbol{\varepsilon}^{pl}(t=0,z). \tag{4.16}$$

Using the stress $\boldsymbol{\sigma}(t,z)$, calculated in Eq. (4.15), and

$$M(t) = \int_A \boldsymbol{\sigma}(t,z) z dA, \tag{4.17}$$

the relaxation force can be predicted, by means of

$$F^{mod}(t) = \frac{4M(t)}{l}. \tag{4.18}$$

In order to minimize the difference between $F^{mod}(t)$ and $F^{exp}(t)$, $F^{mod}(t)$ was determined repeatedly with varying Burgers parameters $\mu_{HA,KV}$, $\eta_{HA,KV}$ and $\eta_{HA,M}$ and then compared to $F^{exp}(t)$. This calibration of the Burgers model was done using the averaged standardized relative difference D, by means of

$$D = \frac{1}{N+1} \sum \sqrt{\left(\frac{F^{mod}(t) - F^{exp}(t)}{F^{exp}(t)} \right)^2}, \quad D \to 0, \tag{4.19}$$

with N as the number of time intervals observed. In order to have only one calibration parameter, D was used instead of the mean and standard deviation of the relative error [see Eq. (4.22)]. The "two-membered evolution strategy" [70], [32] was introduced to perform this calibration automatically. The idea behind this strategy follows Darwin's evolution strategy, i.e. a set of Burgers parameters $(\mu_{HA,KV}, \eta_{HA,KV}, \eta_{HA,M})_{parent}$, referred to as "parent"-values, is slightly varied, considering

$$\begin{aligned} (\mu_{HA,KV}, \eta_{HA,KV}, \eta_{HA,M})_{child} \;=\; & (\mu_{HA,KV}, \eta_{HA,KV}, \eta_{HA,M})_{parent} + \\ & (\mathcal{N}\sigma; \eta_{HA,KV}/\mu_{HA,KV}\mathcal{N}\sigma; \eta_{HA,M}/\mu_{HA,KV}\mathcal{N}\sigma) \end{aligned} \tag{4.20}$$

to determine the "child"-values, $(\mu_{HA,KV}, \eta_{HA,KV}, \eta_{HA,M})_{child}$. $\mathcal{N}$ stands for a standardized normally distributed random number, σ is a "scattering factor". If $F^{mod}_{child}(t)$, referring to the "child"-set of Burgers parameters, fits better to $F^{exp}(t)$ than $F^{mod}_{parent}(t)$, refering to the "parents", i.e.

$$D[(\mu_{HA,KV}, \eta_{HA,KV}, \eta_{HA,M})_{child}] < D[(\mu_{HA,KV}, \eta_{HA,KV}, \eta_{HA,M})_{parent}], \tag{4.21}$$

the "child"-parameters change into the new "parent"-values and are further varied. If Eq. (4.21) is not fulfilled, the "child"-values are given up and the "parent"-set is used for a next variation.

The scattering factor σ determines the influence of the random number $\mathcal{N}$ for creating new "children". σ is altered regarding the ratio of the number of cases, for which Eq. (4.21) is true, to the total number of changes, following the "1/5 rule". If this ratio is lower than 1/5, the scattering factor is reduced, otherwise it is enlarged.

If $D[(\mu_{HA,KV}, \eta_{HA,KV}, \eta_{HA,M})_{parent}]$ is lower than a required value, the "mutation" is stopped and the final set of Burgers parameters $\mu_{HA,KV}$, $\eta_{HA,KV}$ and $\eta_{HA,M}$ has been found.

Using this evolution-strategy, the Burgers parameters listed in Table 4.2 were found:

Table 4.2: Burgers parameters

$\mu_{HA,KV}$	179.5262 GPa
$\eta_{HA,KV}$	1.3430×10^5 GPa s
$\eta_{HA,M}$	1.2842×10^7 GPa s

Mean and standard deviation of the relative error between predicted values and experimental results were used as statistical instruments to evaluate the quality of the results:

$$\begin{aligned} \bar{e} &= \frac{1}{n}\sum e_i = \frac{1}{n}\sum \frac{F^{mod}(t) - F^{exp}(t)}{F^{exp}(t)} \\ e_s &= \left[\frac{1}{n-1}\sum_i (e_i - \bar{e})^2\right]^{1/2} . \end{aligned} \tag{4.22}$$

The difference between $F^{mod}(t)$ and $F^{exp}(t)$ is $+1.83 \pm 0.05\%$ (see Fig. 4.4).

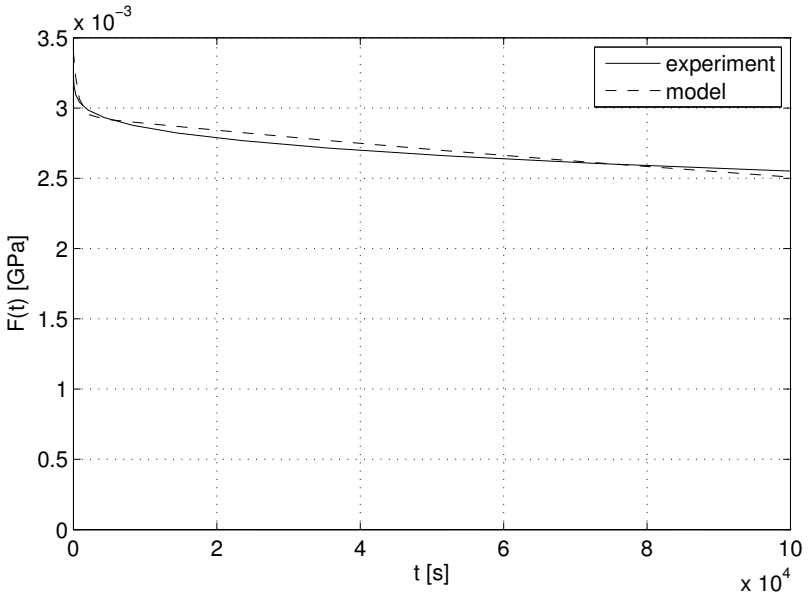

Figure 4.4: Comparison between experimental data and predicted results for a three-point bending test on longitudinal specimens performed by Iyo et al. [39]

4.3 Experimental Validation

To be sure the Burgers parameters determined in Subsection 4.2 (see Table 4.2) are not found by chance, but are valid for several types of experimental examinations, the predictions of our multiscale viscoelastic model and its parameters were compared to the results of different other experimental tests: (i) cantilever bending tests carried out on longitudinal samples made of bovine femur by Sasaki et al. [67], and (ii) three-point bending tests performed on transverse specimens made of bovine femur by Iyo et al. [39].

At first, the cantilever bending tests were examined. The experimental relaxation force and the initial distribution of the elastic strains in the cross-section under the load were determined for the experimental data given in [67] in the same way as described in the beginning of the previous subsection, using Eq. (2.2). With the help of Eqs. (4.15)-(4.18), the predicted relaxation forces were found. The divergence between predictions and experiments is $+2.52 \pm 0.09\%$ (see Fig. 4.5).

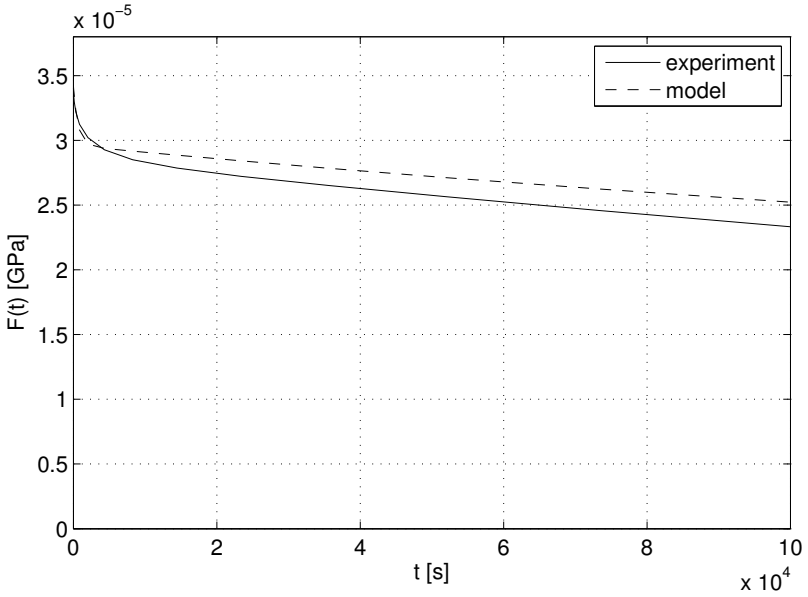

Figure 4.5: Comparison between experimental data and predicted results for a cantilever bending test on longitudinal specimens performed by Sasaki et al. [67]

In contrast to the examinations on longitudinal specimens discussed before, the initial, elastoplastic distribution of strains can not be determined for the transverse samples tested by Iyo et al. [39]. Since there is no axis of symmetry, the distribution of the hydroxyapatite needles is very complicated. Hence, the model presented in [25] is not able to deliver the needed macroscopic stress-strain diagram, but is capable to calculate the elastic limits (yield stress and yield strain) in transverse direction. Therefore, the limit cases of ideal elasticity and ideal plasticity were examined (see distribution of strains and stresses in Fig. 4.6).

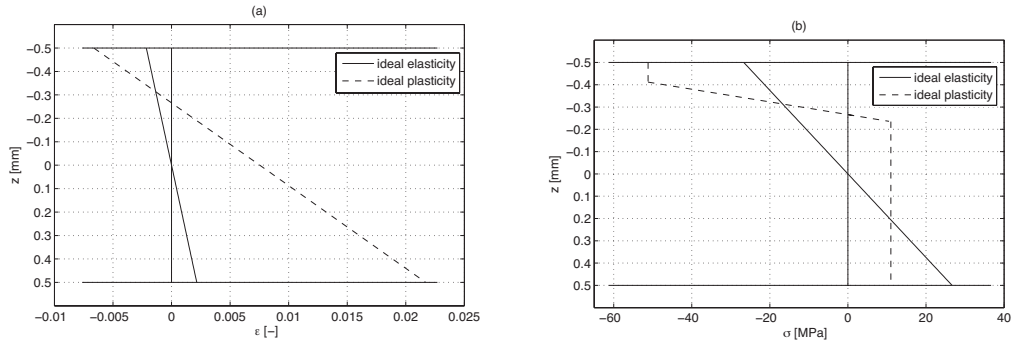

Figure 4.6: Distribution of (a) strains and (b) stress for limit cases of ideal elasticity and ideal plasticity in the cross-section under the applied load of transverse specimen

In Fig. 4.7, the experimental results and the predicted relaxation forces for the limit cases are shown. The experimental curve is located in between the limitation curves.

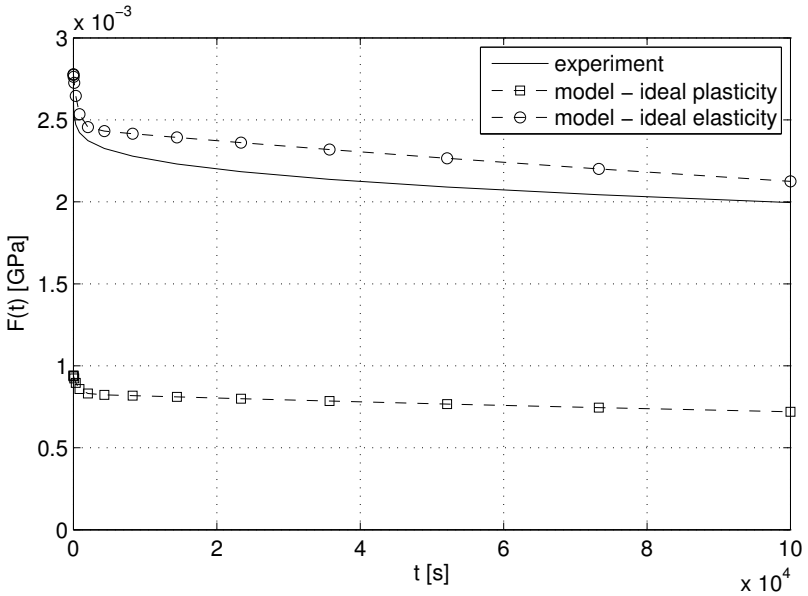

Figure 4.7: Comparison between experimental data and predicted results for a three-point bending test on transverse specimens performed by Iyo et al. [39]; upper and lower bounds considering ideal elastic theory and ideal plastic theory

The relative errors are $+6.48 \pm 0.03\%$ for the elastic and $-63.96 \pm 0.003\%$ for the ideal plastic calculations, implying that the effect of plasticity is very low.

4.4 Use of the Nabarro model instead of the Burgers model

Besides the Burgers model used in Eq. (3.8), another approach for $\mu^*_{HA}(p)$ was made using the model of Nabarro, who found out that under particular circumstances creep behavior is proportional to the logarithm of time $\ln(t/\tau)$ and, for thermally activated processes, dependent on the temperature [56], [59]. Therefore, he introduced two models to describe these phenomena: (i) the work-hardening model, and (ii) the exhaustion model. The time constant τ is used to determine, which model is applied.

Because creep of bone is neither a thermally activated, nor a temperature-dependent process, the general form of the model given in [59] is used:

$$J_{hyd}(t) = \frac{1}{\mu_{HA,inst}} + \alpha_{HA} \times \ln\left(1 + \frac{t}{\tau_{HA}}\right), \tag{4.23}$$

yielding the shear relaxation modulus in LC-domain by means of

$$\mu^*_{HA}(p) = \left[\frac{1}{\mu_{HA,inst}} + \alpha_{HA} e^{p\tau_{HA}} E_1(p\tau_{HA})\right]^{-1} \tag{4.24}$$

with

$$E_1(x) = \int_1^\infty \frac{e^{-tx}}{t} dt, \qquad \forall x > 0 \tag{4.25}$$

and $\mu_{HA,inst} = \mu_{HA}$, according to Table 3.1.

The parameters of the Nabarro model, α_{HA} and τ_{HA} (see Table 4.3), are determined in the same way as the Burgers parameters $\mu_{HA,KV}$, $\eta_{HA,KV}$ and $\eta_{HA,M}$ in Subsection 4.2.

Table 4.3: Nabarro parameters

α_{HA}	$1.6865 \times 10^{-}3$ GPa^{-1}
τ_{HA}	162.2881 s

The difference between the predicted values and the experimental data of the three-point bending tests on longitudinal specimens made of bovine femur performed by Iyo et al. [39] (see Subsection 2.2) is $+2.16 \pm 0.04\%$ (see Fig. 4.8).

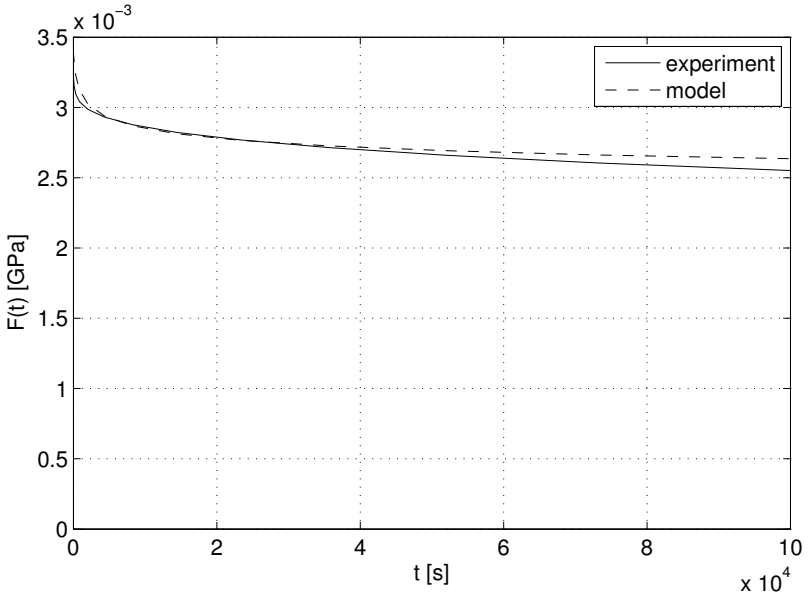

Figure 4.8: Comparison between experimental data and predicted results for a three-point bending test on longitudinal specimens performed by Iyo et al. [39] using the Nabarro model

Since the mean relative error for the Nabarro model is higher than for the Burgers model, the Burgers model is used for further investigations.

Chapter 5

Model-predicted composition-dependent creep of bone

Due to biodiversity, there is a great variation in bone microstructure. Because of the significant influence on viscoelastic behavior of bone materials, especially vascular porosity and mineral content in the extravascular matrix are interesting.

Considering mammals, vascular porosity in healthy cortical bone lies between 2% and 8%, and could increase up to 27% in the case of osteoporosis [25], [12].

The chemical composition within the extravascular matrix depends only on the organ, the individual and the species, but do not vary in space and time [25], [30], [9], [65], [4], [11]. The extravascular mineral content $\bar{f}_{HA}$ reaches from 30%, typical for deer antler, up to 70% in equine metacarpus [25], while the collagen content $\bar{f}_{col}$ can be calculated by means of [25], [24]:

$$\mathcal{F}_{\bar{f}_{col}}(\rho_{ultra,wet}) = \frac{0.9}{\rho_{H_2O} - \rho_{org}} \times \left\{ \mathcal{F}_{\bar{f}_{HA}}(\rho_{ultra,wet}) \times [\rho_{HA} - \rho_{H_2O}] - \rho_{ultra,wet} + \rho_{H_2O} \right\}, \tag{5.1}$$

with

$$\mathcal{F}_{\bar{f}_{HA}} = \mathcal{A} \times \rho_{ultra,wet} + \mathcal{B}, \tag{5.2}$$

$\mathcal{A} = 0.59$ ml/g and $\mathcal{B} = -0.75$, and $\rho_{H_2O} = 1.00$ g/cm^3, $\rho_{org} \approx \rho_{col} = 1.41$ g/cm^3 and $\rho_{HA} = 3.00$ g/cm^3, $\rho_{ultra,wet}$ can be back-calcualted from Eq. (5.2).

In the following subsection, the viscoelastic behavior of bone materials with vascular porosities of 2%, 10%, 20% and 30%, and extravascular mineral contents of 32%, 39%, 43%, 49% and 55% are examined.

5.1 Influence of varying extravascular mineral content and vascular porosity on the elastic and viscoelastic behavior

In Fig. 5.1, the effect of increasing extravascular mineral content $\bar{f}_{HA}$, as well as of increasing vascular porosity f_{vas} on modulus of elasticity in longitudinal [Fig. 5.1 (a)] and transverse direction [Fig. 5.1 (b)], respectively, can be seen. Enlarging $\bar{f}_{HA}$, leads to a growth of Young's modulus, whereas raising f_{vas} results in decreasing, implying that the mineral is responsible for the stiffness of bone material, which is consistent with the findings of Currey [16].

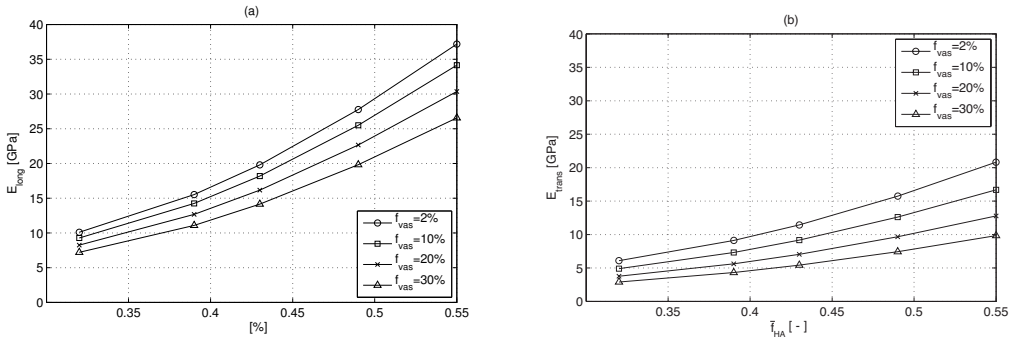

Figure 5.1: Model-predicted modulus of elasticity in (a) longitudinal, and (b) transverse direction as function of $\bar{f}_{HA}$ for different f_{vas}

The variation of $\bar{f}_{HA}$ and f_{vas} affects the relaxation modulus $1/J_{3333}$ in the same way [see Figs. 5.2 (a) and (b)].

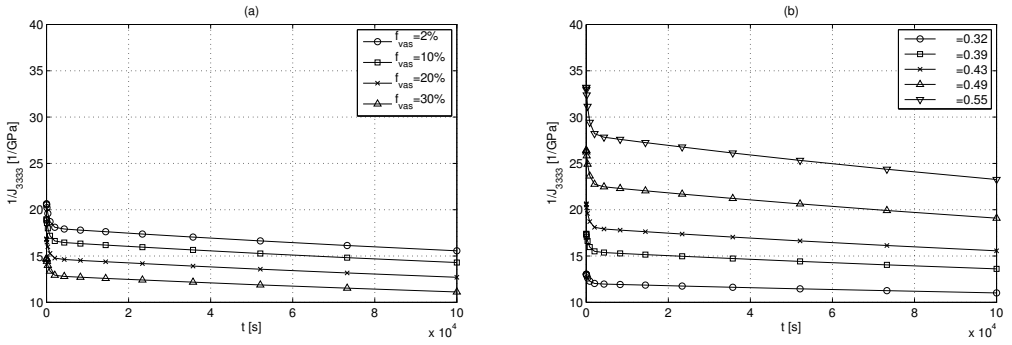

Figure 5.2: (a) Model-predicted relaxation modulus $1/J_{3333}$ for $\bar{f}_{HA} = 0.43$, (b) Model-predicted relaxation modulus $1/J_{3333}$ for $f_{vas} = 2\%$

Fig. 5.3 shows the influence of varying f_{vas} and $\bar{f}_{HA}$ on the five independent components of the creep tensor $\mathbb{J}$.

The isoline plots seen in Figs. 5.4-5.8 show the five independent components of the creep tensor $\mathbb{J}$ as functions of time, for different mineral content and different vascular porosities. In addition to the effects discussed before, two further phenomena can be observed: (i) an increase in $\bar{f}_{HA}$ leads to a decrease in the creep compliances in both directions (J_{1111} and J_{3333}), and (ii) this decrease is more pronounced in the transverse direction than in longitudinal direction (compare J_{1111} and J_{3333} in Figs. 5.4 and 5.5).

The first effect can be explained by the distribution of load to the hydroxyapatite needles. If the mineral content grows, e.g. if there are more needles, the load applied to one single needle reduces. Hence, the thin water layers between the single needles creep less, and therefore the whole material also creeps less. This phenomenon is direction-independent.

The second effect is related to the explanations made in Subsection 4.1 concerning the plasticity of the hydroxyapatite foam. In the transverse direction, the minerals carry much more of an overall macroscopic load than they do in longitudinal direction, where the collagens contribute more. Therefore, higher mineral content expresses itself in a stiffness gain (and creep loss) which is larger in transverse and smaller in longitudinal direction. Higher contribution in load carrying may lead to earlier (and more) plastification as detailed in Subsection 4.1.

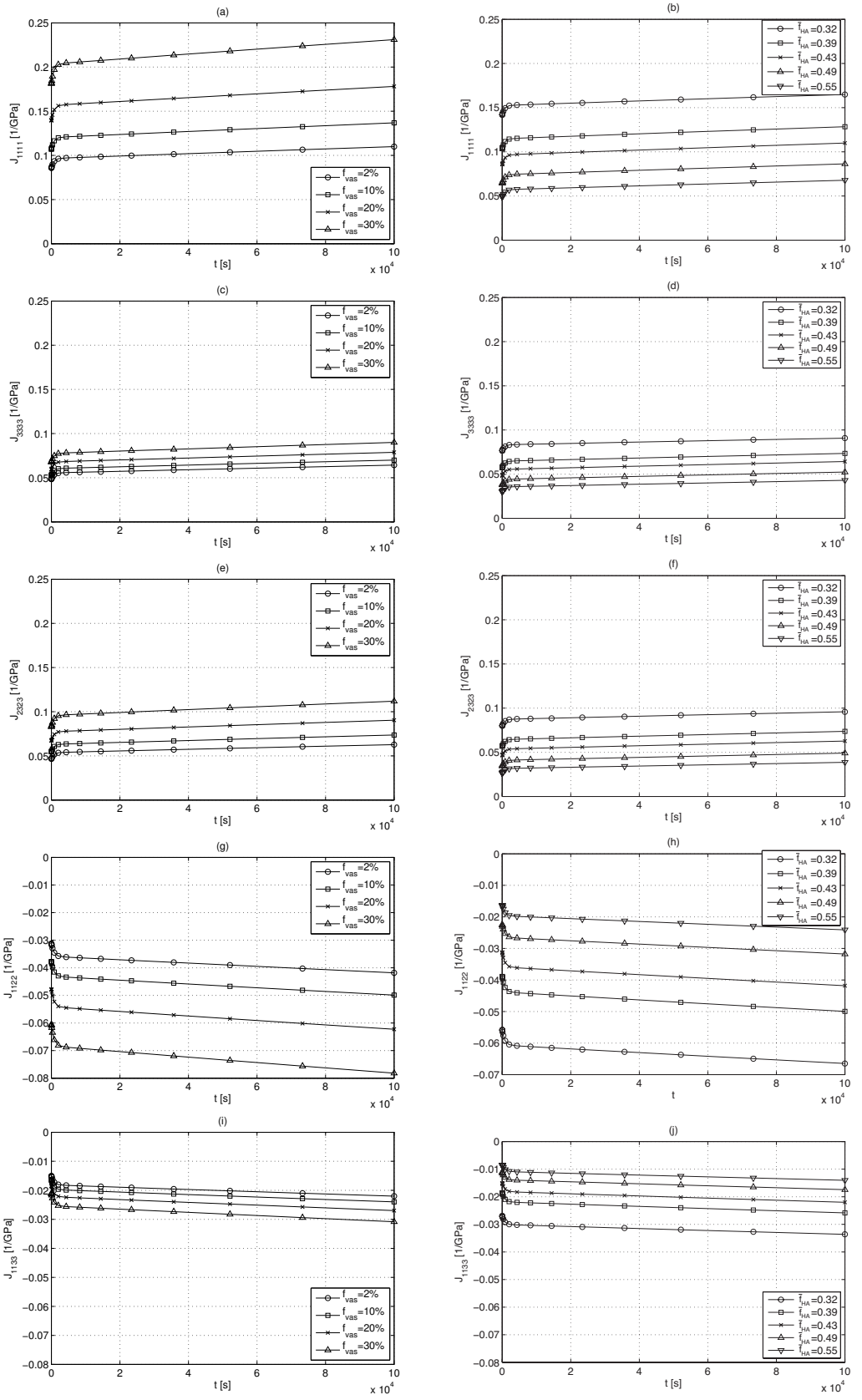

Figure 5.3: Plots for (a) J_{1111} for $\bar{f}_{HA} = 0.43$, (b) J_{1111} for $f_{vas} = 2\%$, (c) J_{3333} for $\bar{f}_{HA} = 0.43$, (d) J_{3333} for $f_{vas} = 2\%$, (e) J_{2323} for $\bar{f}_{HA} = 0.43$, (f) J_{2323} for $f_{vas} = 2\%$, (g) J_{1122} for $\bar{f}_{HA} = 0.43$, (h) J_{1122} for $f_{vas} = 2\%$, (i) J_{1133} for $\bar{f}_{HA} = 0.43$, (j) J_{1133} for $f_{vas} = 2\%$

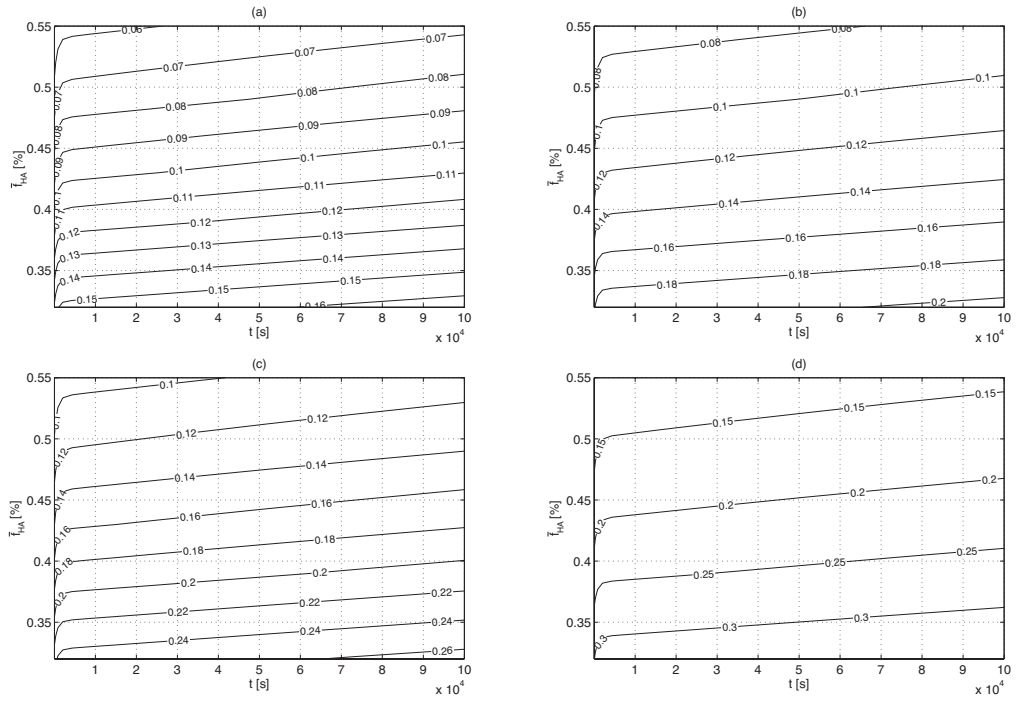

Figure 5.4: Isoline plot for (a) J_{1111} for $f_{vas} = 2\%$, (b) J_{1111} for $f_{vas} = 10\%$, (c) J_{1111} for $f_{vas} = 20\%$, (d) J_{1111} for $f_{vas} = 30\%$

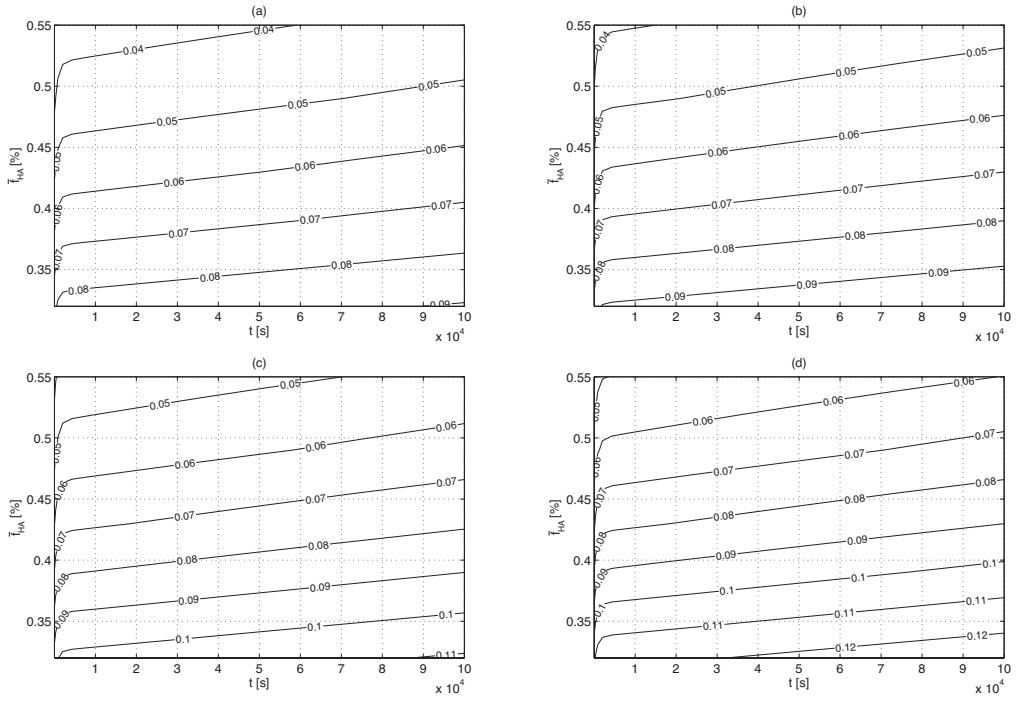

Figure 5.5: Isoline plot for (a) J_{3333} for $f_{vas} = 2\%$, (b) J_{3333} for $f_{vas} = 10\%$, (c) J_{3333} for $f_{vas} = 20\%$, (d) J_{3333} for $f_{vas} = 30\%$

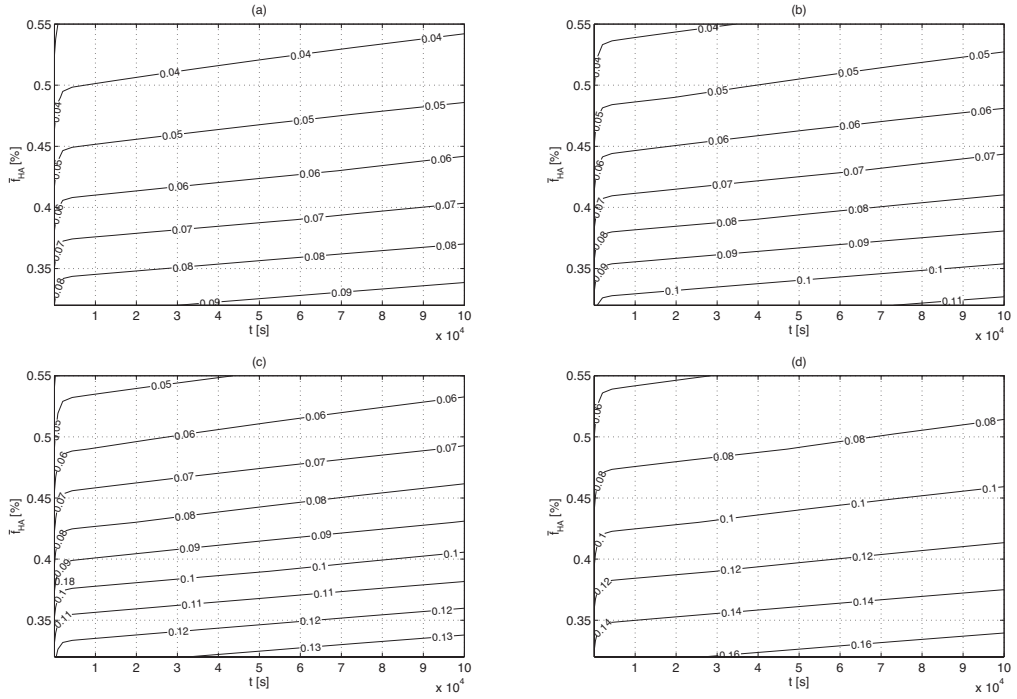

Figure 5.6: Isoline plot for (a) J_{2323} for $f_{vas} = 2\%$, (b) J_{2323} for $f_{vas} = 10\%$, (c) J_{2323} for $f_{vas} = 20\%$, (d) J_{2323} for $f_{vas} = 30\%$

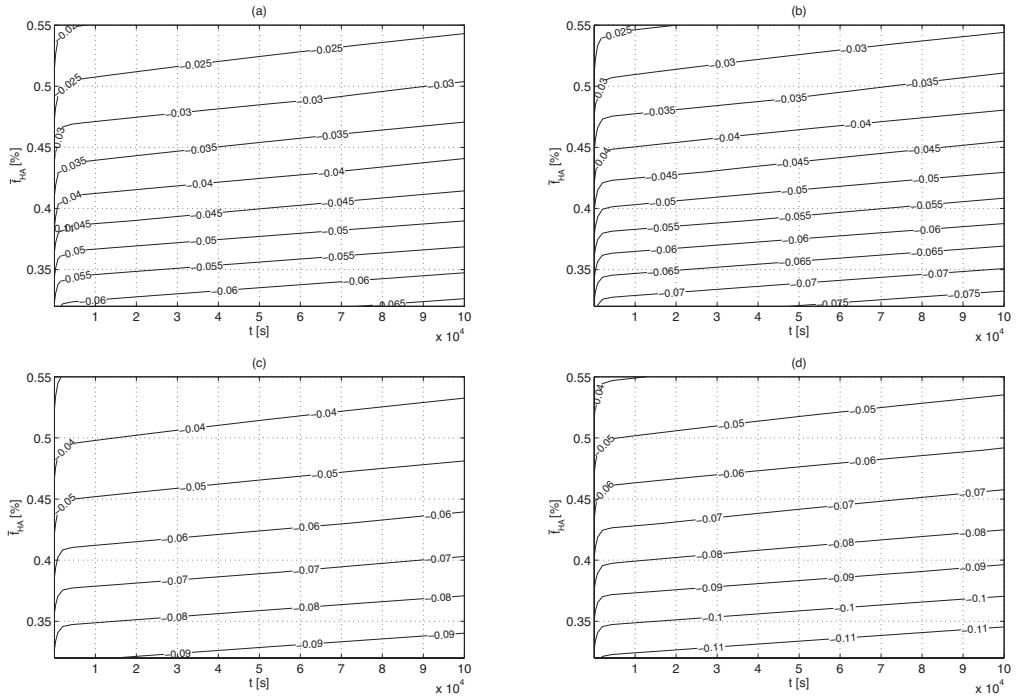

Figure 5.7: Isoline plot for (a) J_{1122} for $f_{vas} = 2\%$, (b) J_{1122} for $f_{vas} = 10\%$, (c) J_{1122} for $f_{vas} = 20\%$, (d) J_{1122} for $f_{vas} = 30\%$

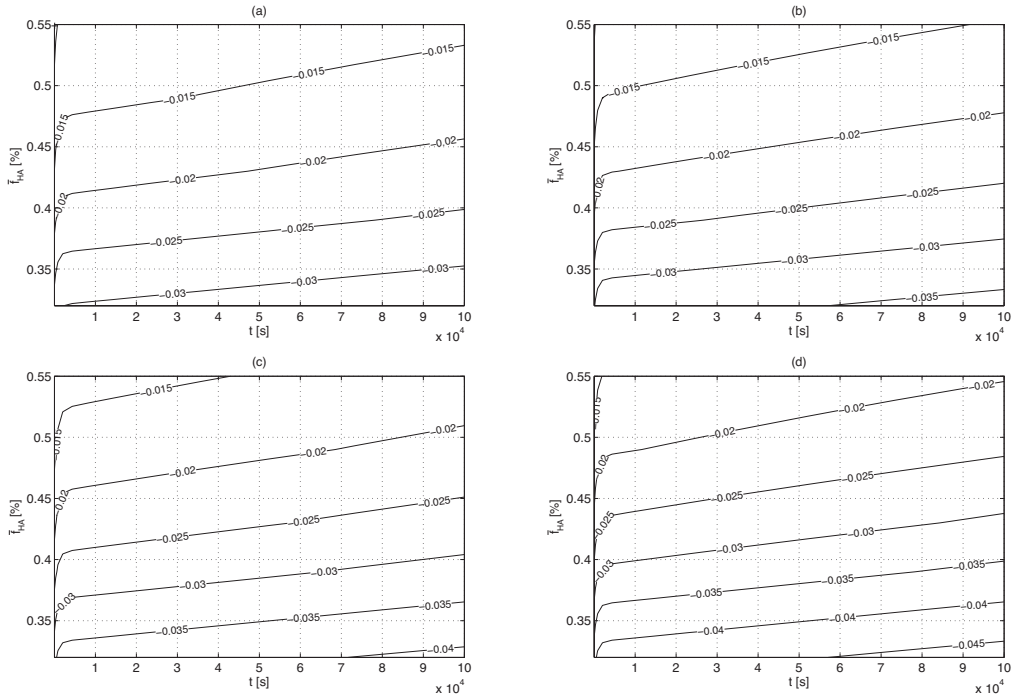

Figure 5.8: Isoline plot for (a) J_{1133} for $f_{vas} = 2\%$, (b) J_{1133} for $f_{vas} = 10\%$, (c) J_{1133} for $f_{vas} = 20\%$, (d) J_{1133} for $f_{vas} = 30\%$

5.2 Influence of varying water content on the viscoelastic behavior

As mentioned before, creep of bone material is understood as gliding of hydroxyapatite needles on intercrystalline water layers in the extrafibrillar space.

Sasaki et al. [67] carried out cantilever bending tests on specimens with different water content: (i) samples soaked in water but drained, which means that the water in the lacunar and vascular pores could escape, and therefore, do not participate in load transfer, as well as (ii) samples with a water content of $\phi = 0.14$.

Specimens of the first group exhibits much more creep behavior than ones of the second group (see Fig. 5.9).

With the help of the multiscale model presented in Chapter 3, the water content of the soaked specimens was determined to $\phi = 0.37$. This means that there is less water in the extrafibriallar space within bone material with $\phi = 0.14$, which leads to less creep behavior.

These results underline our understanding of creep of bone.

Examining the influence of the water content on the level of the extrafibrillar space on the viscoelastic behavior is beyond the scope of the multiscale model in its current form and this manuscript. This could be the subject of further investigations.

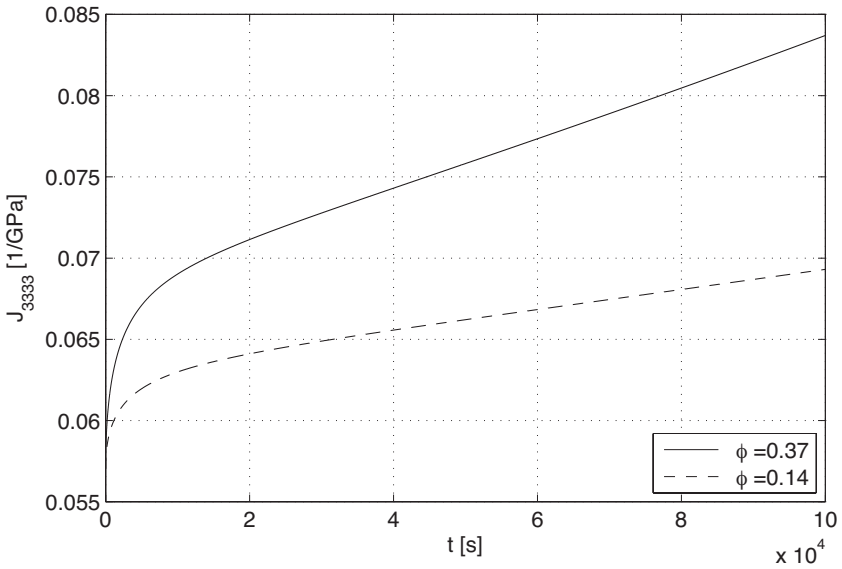

Figure 5.9: Comparison between experimental data for a cantilever bending test on longitudinal specimens with $\phi = 0.37$ and $\phi = 0.14$ performed by Sasaki et al. [67]

Chapter 6

Conclusion

A first approach of predicting long-term creep and relaxation functions for bone material considering a multiscale micromechanics model has been proposed in this manuscript. As to the knowledge of the authors, it was the first time that the intercrystalline water layers between hydroxyapatite needles appearing in the extrafibrillar space were regarded as responsible for the viscoelastic behavior of cortical bone.

The Burgers model turned out to describe the creep behavior of bone very well. In a first step the parameters $\mu_{HA,KV}$, $\eta_{HA,KV}$ and $\eta_{HA,M}$ were determined using an evolution strategy. To validate this approach, the predicted values were compared to the results of different experimental tests. The fact that the mean values of the relative error between experimental results and model estimates are less than 3% for several tests, is a clear indicator for the significance of this approach.

Taking a multiscale model into account, allowed to study the effects of varying extravascular mineral contents and vascular porosities on viscoelastic behavior of bone material. As expected, enlarging the mineral content resulted in an increase, enlarging the vascular porosity in a decrease of the relaxation modulus. The creep compliance reduced as the mineral content increased, which could be explained by the distribution of the applied load on the single hydroxyapatite needles. A unequally fast decrease of the creep compliance in longitudinal and transverse direction could be attributed to the contribution of collagens to load transfer.

This work should help to better understand, how bone works mechanically. Especially, the ability of relaxation behavior to reduce stress concentrations, which occur at micro cracks, bone fractures or fixations of implants, for example, is of interest, in order to optimize, e.g., the treatment after bone fractures or fixation techniques.

Because of lack of long-term experimental data, only bovine femur was examined in this work. To investigate other bones of others species could be the task of further research. In

addition, the influence of the water content in the extrafibrillar space on the viscoelastic behavior of bone microstructure could be studied to provide another evidence for gliding events of hydroxyapatite needles on intercrystaline water layers to be responsible for creep of bone.

Appendix A

Hill tensor $\mathbb{P}^0_{sph}$ for spherical inclusions in a transversely isotropic matrix

The non-zero components of the Hill tensor $\mathbb{P}^0_{sph}$ for spherical inclusions in a transversely isotropic matrix are give as [29]:

$$\begin{aligned}P^0_{sph,1111} = \frac{1}{16}\int_{-1}^{1} &-(-5R^0_{1111}x^4R^0_{3333} - 3R^0_{1122}x^2R^0_{3333} - 3R^0_{1122}x^4R^0_{2323}\\ &+3R^0_{1122}x^4R^0_{3333} + 5R^0_{1111}x^4R^0_{2323} - 10R^0_{1111}R^0_{2323}x^2 + 2x^4R^{0,2}_{1133}\\ &+8R^0_{2323}x^4R^0_{3333} - 6R^{0,2}_{2323}x^4 + 4R^0_{2323}x^4R^0_{1133} + 6R^0_{1122}R^0_{2323}x^2\\ &+5R^0_{1111}R^0_{2323} + 5R^0_{1111}x^2R^0_{3333} - 4R^0_{2323}x^2R^0_{1133} + 6R^{0,2}_{2323}x^2\\ &-2x^2R^{0,2}_{1133} - 3R^0_{1122}R^0_{2323})(-1+x^2)/\mathcal{D}_1 dx\end{aligned} \tag{A.1}$$

$$\begin{aligned}P^0_{sph,1122} = P^0_{sph,2211} = \frac{1}{16}\int_{-1}^{1} &(R^0_{1111}R^0_{2323} - 2R^0_{1111}R^0_{2323}x^2 + R^0_{1111}x^2R^0_{3333}\\ &+R^0_{1122}R^0_{2323} - 2R^0_{1122}R^0_{2323}x^2 + R^0_{1122}x^2R^0_{3333} + R^0_{1111}x^4R^0_{2323} - R^0_{1111}x^4R^0_{3333}\\ &+R^0_{1122}x^4R^0_{2323} - R^0_{1122}x^4R^0_{3333} - 2R^{0,2}_{2323}x^2 + 2R^{0,2}_{2323}x^4 - 4R^0_{2323}x^2R^0_{1133}\\ &+4R^0_{2323}x^4R^0_{1133} - 2x^2R^{0,2}_{1133} + 2x^4R^{0,2}_{1133})(-1+x^2)/\mathcal{D}_1 dx\end{aligned} \tag{A.2}$$

$$P^0_{sph,1133} = P^0_{sph,3311} = \frac{1}{4}\int_{-1}^{1}(-1+x^2)x^2(R^0_{2323} + R^0_{1133})/\mathcal{D}_2 dx \tag{A.3}$$

$$P^0_{sph,2323} = \frac{1}{16}\int_{-1}^{1}(4R^0_{1111}R^0_{2323}x^2 - 8R^0_{2323}x^4R^0_{1133} - 2x^4R^{0,2}_{1133} - R^0_{1122}x^4R^0_{3333}$$
$$-8R^0_{1111}x^4R^0_{2323} + 3R^0_{1111}x^4R^0_{3333} + 4R^0_{1111}x^4R^0_{1133} - 4R^0_{1122}x^4R^0_{1133}$$
$$+2R^0_{1122}x^6R^0_{1133} - 2R^0_{1111}x^6R^0_{1133} + R^0_{1122}x^6R^0_{1111} - 3R^0_{1122}x^4R^0_{1111}$$
$$+3R^0_{1122}R^0_{1111}x^2 - 2R^0_{1111}x^2R^0_{1133} + 2R^0_{1122}x^2R^0_{1133} + 8x^6R^0_{2323}R^0_{1133}$$
$$-3x^6R^0_{1111}R^0_{3333} + 4x^6R^0_{2323}R^0_{3333} + 4R^0_{1111}x^6R^0_{2323} + R^0_{1122}x^6R^0_{3333} + 3R^{0,2}_{1111}x^4$$
$$-R^{0,2}_{1111}x^6 + 2R^{0,2}_{1133}x^6 - 3R^{0,2}_{1111}x^2 + R^{0,2}_{1111} - R^0_{1122}C^0_{1111})/\mathcal{D}_1 dx \tag{A.4}$$

$$P^0_{sph,3333} = \frac{1}{2}\int_{-1}^{1} x^2(x^2R^0_{2323} - R^0_{1111}x^2 + R^0_{1111})/\mathcal{D}_2 dx \tag{A.5}$$

whereby

$$\mathcal{D}_1 = -2R^{0,2}_{1111}x^4R^0_{3333} + 2R^{0,2}_{2323}x^6R^0_{3333} - 4R^0_{1111}R^{0,2}_{2323}x^4 - 3R^{0,2}_{1111}R^0_{2323}x^2 + R^{0,2}_{1111}x^2R^0_{3333} +$$
$$2R^0_{1111}R^{0,2}_{2323}x^2 - 2R^0_{2323}x^4R^{0,2}_{1133} - R^0_{1111}R^{0,2}_{1133}x^6 + 2R^0_{1111}R^{0,2}_{1133}x^4 + 4R^{0,2}_{2323}x^6R^0_{1133}$$
$$-2R^0_{1122}R^{0,2}_{1133}x^4 + 2R^0_{2323}x^6R^{0,2}_{1133} + 3R^{0,2}_{1111}x^4R^0_{2323} + R^0_{1122}R^{0,2}_{1133}x^6 - R^{0,2}_{1111}x^6R^0_{2323}$$
$$+2R^0_{1111}x^6R^{0,2}_{2323} + R^{0,2}_{1111}x^6R^0_{3333} - R^0_{1111}R^{0,2}_{1133}x^2 - 4R^{0,2}_{2323}x^4R^0_{1133} + R^0_{1122}R^{0,2}_{1133}x^2$$
$$+R^{0,2}_{1111}R^0_{2323} - R^0_{1122}R^0_{1111}R^0_{2323} - R^0_{1122}x^6R^0_{1111}R^0_{3333} + 4R^0_{1111}x^4R^0_{2323}R^0_{1133} - 2R^0_{1111}x^2R^0_{2323}R^0_{1133}$$
$$-4R^0_{1122}x^4R^0_{2323}R^0_{1133} + 2R^0_{1122}x^2R^0_{2323}R^0_{1133} + 2R^0_{1122}x^6R^0_{2323}R^0_{1133} - 2R^0_{1111}x^6R^0_{2323}R^0_{1133}$$
$$-3R^0_{1111}x^6R^0_{2323}R^0_{3333} + 2R^0_{1122}R^0_{1111}x^4R^0_{3333} - R^0_{1122}R^0_{2323}x^4R^0_{3333} - 3R^0_{1122}R^0_{1111}x^4R^0_{2323}$$
$$-R^0_{1122}R^0_{1111}x^2R^0_{3333} + 3R^0_{1122}R^0_{1111}R^0_{2323}x^2 + 3R^0_{1111}R^0_{2323}x^4R^0_{3333} + R^0_{1122}x^6R^0_{1111}R^0_{2323}$$
$$+R^0_{1122}x^6R^0_{2323}R^0_{3333} \tag{A.6}$$

and

$$\mathcal{D}_2 = 2R^0_{2323}x^4R^0_{1133} + R^0_{2323}x^4R^0_{3333} + R^0_{1111}x^4R^0_{2323} - 2R^0_{2323}x^2R^0_{1133} - 2R^0_{1111}R^0_{2323}x^2$$
$$+R^0_{1111}R^0_{2323} + x^4R^{0,2}_{1133} - R^0_{1111}x^4R^0_{3333} - x^2R^{0,2}_{1133} + R^0_{1111}x^2R^0_{3333} \tag{A.7}$$

Appendix B

Program code

In this chapter, the source codes of the programs developed with the software tool *Maple* [1] and used in the framework of this thesis are presented.

B.1 Elastic calculations

This section contains the program for the elastic calculations used in Section 5.1:

```
restart;with(plots):with(linalg):with(LinearAlgebra):with(MTM):

#definition of calculation parameters
#-------------------------------------

kapproxmax:=2:                              #approximation order of gaver-wynn-rho algorithm for laplace inversion
Digits:=round(kapproxmax*2.1)+8:            #calculation precision
tmax:=100000:                               #time instant (in seconds) at end of observed viscoelastic response)
numberofintervals:=1:                       #number of intervals
tolerance:=1*10^(-(evalf(round(kapproxmax*2.1))+1));


#definition of material properties (in GPa, GPa.s and mm)
#--------------------------------------------------

ccol3333:=17.9:       ccol1111:=11.7:
ccol1133:=7.1:        ccol1122:=5.1:      ccol1313:=3.3:
muhainst:=44.9:       kha:=82.6:
muwater:=0:           kwater:=2.3:


#determination of volume fractions
#---------------------------------

flac:=0.020:
fvas:=0.300:
rhomu:=1.99:
fmupor:=fvas+flac:
```

```
barfha:=0.55:

A:=0.59:
B:=-0.75:
rhocol:=1.41:
rhoha:=3:
rhowater:=1:

rhoult:=(barfha-B)/A:
barfcol:=0.9*((1/(rhowater-rhocol))*(barfha*(rhoha-rhowater)-rhoult+rhowater)):
rhomu:=rhoult*(1-fmupor)+rhowater*fmupor:

barfcol:=0.29:

tildeflac:=0.021:
ds:=-0.2000*rhoult+1.6580:
b:=1.47:
Dcol:=64:
vcol:=335.6:
vfib:=b*ds*5*Dcol:
barffib:=barfcol*vfib/vcol:
barfef:=1-barffib:

phihaef:=(1-barffib)/(1-barfcol):
checkfha:=phihaef*barfha/barfef:
checkfic:=1-checkfha:

brevefha:=barfha*(1-phihaef)/barffib:
brevefwetcol:=1-brevefha:

circlefcol:=barfcol/brevefwetcol:
circlefuw:=1-circlefcol:

fim:=1-circlefcol;
circlefcol:=circlefcol;
fwetcol:=brevefwetcol;
fha:=checkfha;
ffib:=barffib;
flac:=flac;
fvas:=fvas;

wc:=(fim*fwetcol*ffib*(1-flac)*(1-fvas))+((1-fha)*(1-ffib)*(1-flac)*(1-fvas));


#definition of volumetric and deviatoric matrix
#-----------------------------------------------

k:=1/3:
K:=Matrix([[k,k,k,0,0,0],[k,k,k,0,0,0],[k,k,k,0,0,0],[0,0,0,0,0,0],[0,0,0,0,0,0],[0,0,0,0,0,0]]):
J:=IdentityMatrix(6,6)-K:
I6:=IdentityMatrix(6,6):


#defintion of Q, theta & phi for rotation
#----------------------------------------

e1:=Vector([cos(phi)*cos(theta),sin(phi)*cos(theta),-sin(theta)]):
e2:=Vector([-sin(phi),cos(phi),0]):
e3:=Vector([cos(phi)*sin(theta),sin(phi)*sin(theta),cos(theta)]):

q11:=e1[1]:
q21:=e1[2]:
q31:=e1[3]:
q12:=e2[1]:
q22:=e2[2]:
q32:=e2[3]:
q13:=e3[1]:
q23:=e3[2]:
q33:=e3[3]:
```

```
a:=2/sqrt(2):
n:=sqrt(2):

Q:=Matrix(6):
Q[1,1]:=q11^2:
Q[1,2]:=q12^2:
Q[1,3]:=q13^2:
Q[1,4]:=a*q12*q13:
Q[1,5]:=a*q13*q11:
Q[1,6]:=a*q11*q12:
Q[2,1]:=q21^2:
Q[2,2]:=q22^2:
Q[2,3]:=q23^2:
Q[2,4]:=a*q22*q23:
Q[2,5]:=a*q23*q21:
Q[2,6]:=a*q21*q22:
Q[3,1]:=q31^2:
Q[3,2]:=q32^2:
Q[3,3]:=q33^2:
Q[3,4]:=a*q32*q33:
Q[3,5]:=a*q33*q31:
Q[3,6]:=a*q31*q32:
Q[4,1]:=n*q21*q31:
Q[4,2]:=n*q22*q32:
Q[4,3]:=n*q23*q33:
Q[4,4]:=q23*q32+q33*q22:
Q[4,5]:=q21*q33+q31*q23:
Q[4,6]:=q22*q31+q32*q21:
Q[5,1]:=n*q31*q11:
Q[5,2]:=n*q32*q12:
Q[5,3]:=n*q33*q13:
Q[5,4]:=q33*q12+q13*q32:
Q[5,5]:=q31*q13+q11*q33:
Q[5,6]:=q32*q11+q12*q31:
Q[6,1]:=n*q11*q21:
Q[6,2]:=n*q12*q22:
Q[6,3]:=n*q13*q23:
Q[6,4]:=q13*q22+q23*q12:
Q[6,5]:=q11*q23+q21*q13:
Q[6,6]:=q12*q21+q22*q11:

r:=1/2:
s:=(sqrt(5)+1)/4:
t:=(sqrt(5)-1)/4:

theta1:=evalf(arccos(t)):
phi1:=evalf(arccos(r/sin(theta1))):
evalf(s-sin(theta1)*sin(phi1)):
theta2:=evalf(arccos(t)):
phi2:=evalf(arcsin(-s/sin(theta2))):
evalf(r-sin(theta2)*cos(phi2)):
theta3:=evalf(arccos(t)):
phi3:=evalf(arccos(-r/sin(theta3))):

evalf(s-sin(theta3)*sin(phi3)):
theta4:=evalf(arccos(t)):
phi4:=evalf(Pi-arcsin(-s/sin(theta4))):

evalf(-r-sin(theta4)*cos(phi4)):
theta5:=evalf(arccos(s)):
phi5:=evalf(arccos(t/sin(theta5))):

evalf(r-sin(theta5)*sin(phi5)):
theta6:=evalf(arccos(s)):
phi6:=evalf(arcsin(-r/sin(theta6))):

evalf(t-sin(theta6)*cos(phi6)):
theta7:=evalf(arccos(s)):
```

```
phi7:=evalf(arccos(-t/sin(theta7))):

evalf(r-sin(theta7)*sin(phi7)):
theta8:=evalf(arccos(s)):
phi8:=evalf(Pi-arcsin(-r/sin(theta8))):

evalf(-t-sin(theta8)*cos(phi8)):
theta9:=evalf(arccos(r)):
phi9:=evalf(arccos(s/sin(theta9))):
evalf(t-sin(theta9)*sin(phi9)):
theta10:=evalf(arccos(r)):
phi10:=evalf(arcsin(-t/sin(theta10))):

evalf(s-sin(theta10)*cos(phi10)):
theta11:=evalf(arccos(r)):
phi11:=evalf(arccos(-s/sin(theta11))):

evalf(t-sin(theta11)*sin(phi11)):
theta12:=evalf(arccos(r)):
phi12:=evalf(Pi-arcsin(-t/sin(theta12))):

evalf(-s-sin(theta12)*cos(phi12)):
theta13:=evalf(Pi/2):
phi13:=0:

theta14:=evalf(Pi/2):
phi14:=evalf(Pi/2):

theta15:=0:
phi15:=0:

Q1:=Matrix(6):
Q2:=Matrix(6):
Q3:=Matrix(6):
Q4:=Matrix(6):
Q5:=Matrix(6):
Q6:=Matrix(6):
Q7:=Matrix(6):
Q8:=Matrix(6):
Q9:=Matrix(6):
Q10:=Matrix(6):
Q11:=Matrix(6):
Q12:=Matrix(6):
Q13:=Matrix(6):
Q14:=Matrix(6):
Q15:=Matrix(6):

for i from 1 to 6 do
for j from 1 to 6 do
   Q1[i,j]:=eval(Q[i,j],[theta=theta1,phi=phi1]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q2[i,j]:=eval(Q[i,j],[theta=theta2,phi=phi2]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q3[i,j]:=eval(Q[i,j],[theta=theta3,phi=phi3]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q4[i,j]:=eval(Q[i,j],[theta=theta4,phi=phi4]):
end do:
```

```
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q5[i,j]:=eval(Q[i,j],[theta=theta5,phi=phi5]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q6[i,j]:=eval(Q[i,j],[theta=theta6,phi=phi6]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q7[i,j]:=eval(Q[i,j],[theta=theta7,phi=phi7]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q8[i,j]:=eval(Q[i,j],[theta=theta8,phi=phi8]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q9[i,j]:=eval(Q[i,j],[theta=theta9,phi=phi9]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q10[i,j]:=eval(Q[i,j],[theta=theta10,phi=phi10]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q11[i,j]:=eval(Q[i,j],[theta=theta11,phi=phi11]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q12[i,j]:=eval(Q[i,j],[theta=theta12,phi=phi12]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q13[i,j]:=eval(Q[i,j],[theta=theta13,phi=phi13]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q14[i,j]:=eval(Q[i,j],[theta=theta14,phi=phi14]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q15[i,j]:=eval(Q[i,j],[theta=theta15,phi=phi15]):
end do:
end do:
```

```
#calculation of creep compliance
#-------------------------------

#wet collagen
#-------------
Cwater:=3*kwater*K+2*muwater*J:
Cim:=Cwater:

Ccol:=Matrix(6):
Ccol[1,1]:=ccol1111:
Ccol[2,2]:=Ccol[1,1]:
Ccol[3,3]:=ccol3333:
Ccol[1,2]:=ccol1122:
Ccol[2,1]:=Ccol[1,2]:
Ccol[1,3]:=ccol1133:
Ccol[3,1]:=Ccol[1,3]:
Ccol[3,2]:=Ccol[3,1]:
Ccol[2,3]:=Ccol[3,2]:
Ccol[4,4]:=2*ccol1313:
Ccol[5,5]:=Ccol[4,4]:
Ccol[6,6]:=2*0.5*(Ccol[1,1]-Ccol[1,2]):

Pcylcol:=Matrix(6):
P1111:=1/8*(5*Ccol[1,1]-3*Ccol[1,2])/Ccol[1,1]/(Ccol[1,1]-Ccol[1,2]):
P1122:=-1/8*(Ccol[1,1]+Ccol[1,2])/Ccol[1,1]/(Ccol[1,1]-Ccol[1,2]):
P2222:=P1111:
P2323:=1/8*1/(0.5*Ccol[4,4]):
P1313:=P2323:
P1212:=1/8*(3*Ccol[1,1]-Ccol[1,2])/Ccol[1,1]/(Ccol[1,1]-Ccol[1,2]):
Pcylcol[1,1]:=P1111:
Pcylcol[2,2]:=P2222:
Pcylcol[1,2]:=P1122:
Pcylcol[2,1]:=Pcylcol[1,2]:
Pcylcol[4,4]:=2*P2323:
Pcylcol[5,5]:=2*P1313:
Pcylcol[6,6]:=2*P1212:

help_im:=inv(I6+Pcylcol.(Cim-Ccol)):
Cestwetcol:=((1-fim)*Ccol+fim*Cim.help_im).inv((1-fim)*I6+fim*help_im):

#fibril
#------
Cha:=3*kha*K+2*muhainst*J:

Cestfibnew:=Cestwetcol:
Cestfib:=(1+tolerance*10)*Cestfibnew:

while abs(Cestfib[1,1]-Cestfibnew[1,1])>tolerance do
   Cestfib:=Cestfibnew:

   Pcylfib:=Matrix(6):
   P1111:=1/8*(5*Cestfib[1,1]-3*Cestfib[1,2])/Cestfib[1,1]/(Cestfib[1,1]-Cestfib[1,2]):
   P1122:=-1/8*(Cestfib[1,1]+Cestfib[1,2])/Cestfib[1,1]/(Cestfib[1,1]-Cestfib[1,2]):
   P2222:=P1111:
   P2323:=1/8*1/(0.5*Cestfib[4,4]):
   P1313:=P2323:
   P1212:=1/8*(3*Cestfib[1,1]-Cestfib[1,2])/Cestfib[1,1]/(Cestfib[1,1]-Cestfib[1,2]):
   Pcylfib[1,1]:=P1111:
   Pcylfib[2,2]:=P2222:
   Pcylfib[1,2]:=P1122:
   Pcylfib[2,1]:=Pcylfib[1,2]:
   Pcylfib[4,4]:=2*P2323:
   Pcylfib[5,5]:=2*P1313:
   Pcylfib[6,6]:=2*P1212:

   C1111:=Cestfib[1,1]:
   C1122:=Cestfib[1,2]:
   C1133:=Cestfib[1,3]:
   C3333:=Cestfib[3,3]:
```

```
C2323:=Cestfib[4,4]/2:

Psphfib:=Matrix(6):

D1:=(-2)*C1111^2*x^4*C3333+2*C2323^2*x^6*C3333-4*C1111*C2323^2*x^4-
       3*C1111^2*C2323*x^2+C1111^2*x^2*C3333+
       2*C1111*C2323^2*x^2-2*C2323*x^4*C1133^2-C1111*C1133^2*x^6+
       2*C1111*C1133^2*x^4+4*C2323^2*x^6*C1133-
       2*C1122*C1133^2*x^4+2*C2323*x^6*C1133^2+3*C1111^2*x^4*C2323+
         C1122*C1133^2*x^6-C1111^2*x^6*C2323+
       2*C1111*x^6*C2323^2+C1111^2*x^6*C3333-C1111*C1133^2*x^2-
       4*C2323^2*x^4*C1133+C1122*C1133^2*x^2+
         C1111^2*C2323-C1122*C1111*C2323-C1122*x^6*C1111*C3333+
       4*C1111*x^4*C2323*C1133-2*C1111*x^2*C2323*C1133-
       4*C1122*x^4*C2323*C1133+2*C1122*x^2*C2323*C1133+
       2*C1122*x^6*C2323*C1133-2*C1111*x^6*C2323*C1133-
       3*C1111*x^6*C2323*C3333+2*C1122*C1111*x^4*C3333-
         C1122*C2323*x^4*C3333-3*C1122*C1111*x^4*C2323-
         C1122*C1111*x^2*C3333+3*C1122*C1111*C2323*x^2+
       3*C1111*C2323*x^4*C3333+C1122*x^6*C1111*C2323+
         C1122*x^6*C2323*C3333:

D2:=2*C2323*x^4*C1133+C2323*x^4*C3333+C1111*x^4*C2323-
    2*C2323*x^2*C1133-2*C1111*C2323*x^2+
      C1111*C2323+x^4*C1133^2-C1111*x^4*C3333-
      x^2*C1133^2+C1111*x^2*C3333:

P1111int:=-((-5)*C1111*x^4*C3333-3*C1122*x^2*C3333-
               3*C1122*x^4*C2323+
               3*C1122*x^4*C3333+5*C1111*x^4*C2323-
              10*C1111*C2323*x^2+2*x^4*C1133^2+
               8*C2323*x^4*C3333-6*C2323^2*x^4+
               4*C2323*x^4*C1133+6*C1122*C2323*x^2+
               5*C1111*C2323+5*C1111*x^2*C3333-
               4*C2323*x^2*C1133+6*C2323^2*x^2-
               2*x^2*C1133^2-3*C1122*C2323)*(-1+x^2)/D1:

P1122int:=(C1111*C2323-2*C1111*C2323*x^2+
           C1111*x^2*C3333+C1122*C2323-
         2*C1122*C2323*x^2+C1122*x^2*C3333+
           C1111*x^4*C2323-C1111*x^4*C3333+
           C1122*x^4*C2323-C1122*x^4*C3333-2*C2323^2*x^2+
         2*C2323^2*x^4-4*C2323*x^2*C1133+
         4*C2323*x^4*C1133-2*x^2*C1133^2+
         2*x^4*C1133^2)*(-1+x^2)/D1:

P1133int:=(-1+x^2)*x^2*(C2323+C1133)/D2:

P2323int:=(4*C1111*C2323*x^2-8*C2323*x^4*C1133-
            2*x^4*C1133^2-C1122*x^4*C3333-
            8*C1111*x^4*C2323+3*C1111*x^4*C3333+
            4*C1111*x^4*C1133-4*C1122*x^4*C1133+
            2*C1122*x^6*C1133-2*C1111*x^6*C1133+
              C1122*x^6*C1111-3*C1122*x^4*C1111+
            3*C1122*C1111*x^2-2*C1111*x^2*C1133+
            2*C1122*x^2*C1133+8*x^6*C2323*C1133-
            3*x^6*C1111*C3333+4*x^6*C2323*C3333+
            4*C1111*x^6*C2323+C1122*x^6*C3333+
            3*C1111^2*x^4-C1111^2*x^6+
            2*C1133^2*x^6-3*C1111^2*x^2+
              C1111^2-C1122*C1111)/D1:

P3333int:=x^2*(x^2*C2323-C1111*x^2+
           C1111)/D2:

P1111:=evalf(1/16*int(P1111int,x=-1..1)):
P1122:=evalf(1/16*int(P1122int,x=-1..1)):
P1133:=evalf(1/4*int(P1133int,x=-1..1)):
P2323:=evalf(1/16*int(P2323int,x=-1..1)):
```

```
   P3333:=evalf(1/2*int(P3333int,x=-1..1)):

   Psphfib[1,1]:=P1111:
   Psphfib[2,2]:=Psphfib[1,1]:
   Psphfib[3,3]:=P3333:
   Psphfib[1,2]:=P1122:
   Psphfib[1,3]:=P1133:
   Psphfib[2,3]:=Psphfib[1,3]:
   Psphfib[2,1]:=Psphfib[1,2]:
   Psphfib[3,1]:=Psphfib[1,3]:
   Psphfib[3,2]:=Psphfib[2,3]:
   Psphfib[4,4]:=2*P2323:
   Psphfib[5,5]:=Psphfib[4,4]:
   Psphfib[6,6]:=2*0.5*(P1111-P1122):
   help_wetcol:=inv(I6+Pcylfib.(Cestwetcol-Cestfib)):
   help_ha:=inv(I6+Psphfib.(Cha-Cestfib)):
   Cestfibnew:=(fwetcol*Cestwetcol.help_wetcol+(1-fwetcol)*Cha.help_ha).
               inv(fwetcol*help_wetcol+(1-fwetcol)*help_ha):

end do:
Cestfib:=Cestfibnew:

#extrafibrillar space
#--------------------
Cic:=Cwater:
Cestefnew:=Cha:
Cestef:=(1+tolerance*10)*Cestefnew:

while abs(Cestef[1,1]-Cestefnew[1,1])>tolerance do
            Cestef:=Cestefnew:
            Dest:=inv(Cestef):
            muestef:=1/(2*Dest[4,4]):
            Eestef:=1/Dest[1,1]:
            kestef:=(muestef*Eestef)/(3*(3*muestef-Eestef)):

            alphaef:=3*kestef/(3*kestef+4*muestef):
            betaef:=6*(kestef+2*muestef)/(5*(3*kestef+4*muestef)):
            Ssphef:=alphaef*K+betaef*J:
            Psphef:=Ssphef.inv(Cestef):

            nup:=(3*kestef-2*muestef)/(6*kestef+2*muestef):
            Scylef:=Matrix(6):
            Scylef[1,1]:=(5-4*nup)/(8*(1-nup)):
            Scylef[2,2]:=Scylef[1,1]:
            Scylef[1,2]:=(-1+4*nup)/(8*(1-nup)):
            Scylef[2,1]:=Scylef[1,2]:
            Scylef[1,3]:=nup/(2*(1-nup)):
            Scylef[2,3]:=Scylef[1,3]:
            Scylef[4,4]:=2*(1/4):
            Scylef[5,5]:=Scylef[4,4]:
            Scylef[6,6]:=2*((3-4*nup)/(8*(1-nup))):
            Pcyl_LOCAL:=Scylef.inv(Cestef):

            Pcyl1:=Q1.Pcyl_LOCAL.transpose(Q1):
            Pcyl2:=Q2.Pcyl_LOCAL.transpose(Q2):
            Pcyl3:=Q3.Pcyl_LOCAL.transpose(Q3):
            Pcyl4:=Q4.Pcyl_LOCAL.transpose(Q4):
            Pcyl5:=Q5.Pcyl_LOCAL.transpose(Q5):
            Pcyl6:=Q6.Pcyl_LOCAL.transpose(Q6):
            Pcyl7:=Q7.Pcyl_LOCAL.transpose(Q7):
            Pcyl8:=Q8.Pcyl_LOCAL.transpose(Q8):
            Pcyl9:=Q9.Pcyl_LOCAL.transpose(Q9):
            Pcyl10:=Q10.Pcyl_LOCAL.transpose(Q10):
            Pcyl11:=Q11.Pcyl_LOCAL.transpose(Q11):
            Pcyl12:=Q12.Pcyl_LOCAL.transpose(Q12):
            Pcyl13:=Q13.Pcyl_LOCAL.transpose(Q13):
            Pcyl14:=Q14.Pcyl_LOCAL.transpose(Q14):
            Pcyl15:=Q15.Pcyl_LOCAL.transpose(Q15):
```

```
            help_ha:=1/15*inv(I6+Pcyl1.(Cha-Cestef))+1/15*inv(I6+Pcyl2.(Cha-Cestef))+1/15*inv(I6+Pcyl3.(Cha-Cestef))+
            1/15*inv(I6+Pcyl4.(Cha-Cestef))+1/15*inv(I6+Pcyl5.(Cha-Cestef))+1/15*inv(I6+Pcyl6.(Cha-Cestef))+
            1/15*inv(I6+Pcyl7.(Cha-Cestef))+1/15*inv(I6+Pcyl8.(Cha-Cestef))+1/15*inv(I6+Pcyl9.(Cha-Cestef))+
            1/15*inv(I6+Pcyl10.(Cha-Cestef))+1/15*inv(I6+Pcyl11.(Cha-Cestef))+1/15*inv(I6+Pcyl12.(Cha-Cestef))+
            1/15*inv(I6+Pcyl13.(Cha-Cestef))+1/15*inv(I6+Pcyl14.(Cha-Cestef))+1/15*inv(I6+Pcyl15.(Cha-Cestef)):

            help_pores:=inv(I6+Psphef.(Cic-Cestef)):
            Cestefnew:=(fha*Cha.help_ha+(1-fha)*Cic.help_pores).inv(fha*help_ha+(1-fha)*help_pores):

        end do:

Cestef:=Cestefnew:

#ultrastructure
#--------------
Pcylef:=Matrix(6):
P1111:=1/8*(5*Cestef[1,1]-3*Cestef[1,2])/Cestef[1,1]/(Cestef[1,1]-Cestef[1,2]):
P1122:=-1/8*(Cestef[1,1]+Cestef[1,2])/Cestef[1,1]/(Cestef[1,1]-Cestef[1,2]):
P2222:=P1111:
P2323:=1/8*1/(0.5*Cestef[4,4]):
P1313:=P2323:
P1212:=1/8*(3*Cestef[1,1]-Cestef[1,2])/Cestef[1,1]/(Cestef[1,1]-Cestef[1,2]):
Pcylef[1,1]:=P1111:
Pcylef[2,2]:=P2222:
Pcylef[1,2]:=P1122:
Pcylef[2,1]:=Pcylef[1,2]:
Pcylef[4,4]:=2*P2323:
Pcylef[5,5]:=2*P1313:
Pcylef[6,6]:=2*P1212:

help_fib:=inv(I6+Pcylef.(Cestfib-Cestef)):
Cestult:=((1-ffib)*Cestef+ffib*Cestfib.help_fib).inv((1-ffib)*I6+ffib*help_fib):

#extravascular bone material
#---------------------------
Clac:=Matrix(6):

C1111:=Cestult[1,1]:
C1122:=Cestult[1,2]:
C1133:=Cestult[1,3]:
C3333:=Cestult[3,3]:
C2323:=Cestult[4,4]/2:

Psphult:=Matrix(6):

D1:=(-2)*C1111^2*x^4*C3333+2*C2323^2*x^6*C3333-4*C1111*C2323^2*x^4-
       3*C1111^2*C2323*x^2+C1111^2*x^2*C3333+
       2*C1111*C2323^2*x^2-2*C2323*x^4*C1133^2-C1111*C1133^2*x^6+
       2*C1111*C1133^2*x^4+4*C2323^2*x^6*C1133-
       2*C1122*C1133^2*x^4+2*C2323*x^6*C1133^2+3*C1111^2*x^4*C2323+
         C1122*C1133^2*x^6-C1111^2*x^6*C2323+
       2*C1111*x^6*C2323^2+C1111^2*x^6*C3333-C1111*C1133^2*x^2-
       4*C2323^2*x^4*C1133+C1122*C1133^2*x^2+
         C1111^2*C2323-C1122*C1111*C2323-C1122*x^6*C1111*C3333+
       4*C1111*x^4*C2323*C1133-2*C1111*x^2*C2323*C1133-
       4*C1122*x^4*C2323*C1133+2*C1122*x^2*C2323*C1133+
       2*C1122*x^6*C2323*C1133-2*C1111*x^6*C2323*C1133-
       3*C1111*x^6*C2323*C3333+2*C1122*C1111*x^4*C3333-
         C1122*C2323*x^4*C3333-3*C1122*C1111*x^4*C2323-
         C1122*C1111*x^2*C3333+3*C1122*C1111*C2323*x^2+
       3*C1111*C2323*x^4*C3333+C1122*x^6*C1111*C2323+
         C1122*x^6*C2323*C3333:

D2:=2*C2323*x^4*C1133+C2323*x^4*C3333+C1111*x^4*C2323-
    2*C2323*x^2*C1133-2*C1111*C2323*x^2+
      C1111*C2323+x^4*C1133^2-C1111*x^4*C3333-
      x^2*C1133^2+C1111*x^2*C3333:

P1111int:=-((-5)*C1111*x^4*C3333-3*C1122*x^2*C3333-
```

```
               3*C1122*x^4*C2323+
               3*C1122*x^4*C3333+5*C1111*x^4*C2323-
              10*C1111*C2323*x^2+2*x^4*C1133^2+
               8*C2323*x^4*C3333-6*C2323^2*x^4+
               4*C2323*x^4*C1133+6*C1122*C2323*x^2+
               5*C1111*C2323+5*C1111*x^2*C3333-
               4*C2323*x^2*C1133+6*C2323^2*x^2-
               2*x^2*C1133^2-3*C1122*C2323)*(-1+x^2)/D1:

P1122int:=(C1111*C2323-2*C1111*C2323*x^2+
           C1111*x^2*C3333+C1122*C2323-
         2*C1122*C2323*x^2+C1122*x^2*C3333+
           C1111*x^4*C2323-C1111*x^4*C3333+
           C1122*x^4*C2323-C1122*x^4*C3333-2*C2323^2*x^2+
          2*C2323^2*x^4-4*C2323*x^2*C1133+
          4*C2323*x^4*C1133-2*x^2*C1133^2+
         2*x^4*C1133^2)*(-1+x^2)/D1:

P1133int:=(-1+x^2)*x^2*(C2323+C1133)/D2:

P2323int:=(4*C1111*C2323*x^2-8*C2323*x^4*C1133-
           2*x^4*C1133^2-C1122*x^4*C3333-
           8*C1111*x^4*C2323+3*C1111*x^4*C3333+
           4*C1111*x^4*C1133-4*C1122*x^4*C1133+
           2*C1122*x^6*C1133-2*C1111*x^6*C1133+
             C1122*x^6*C1111-3*C1122*x^4*C1111+
           3*C1122*C1111*x^2-2*C1111*x^2*C1133+
           2*C1122*x^2*C1133+8*x^6*C2323*C1133-
           3*x^6*C1111*C3333+4*x^6*C2323*C3333+
           4*C1111*x^6*C2323+C1122*x^6*C3333+
           3*C1111^2*x^4-C1111^2*x^6+
           2*C1133^2*x^6-3*C1111^2*x^2+
             C1111^2-C1122*C1111)/D1:

P3333int:=x^2*(x^2*C2323-C1111*x^2+
          C1111)/D2:

P1111:=evalf(1/16*int(P1111int,x=-1..1)):
P1122:=evalf(1/16*int(P1122int,x=-1..1)):
P1133:=evalf(1/4*int(P1133int,x=-1..1)):
P2323:=evalf(1/16*int(P2323int,x=-1..1)):
P3333:=evalf(1/2*int(P3333int,x=-1..1)):

Psphult[1,1]:=P1111:
Psphult[2,2]:=Psphult[1,1]:
Psphult[3,3]:=P3333:
Psphult[1,2]:=P1122:
Psphult[1,3]:=P1133:
Psphult[2,3]:=Psphult[1,3]:
Psphult[2,1]:=Psphult[1,2]:
Psphult[3,1]:=Psphult[1,3]:
Psphult[3,2]:=Psphult[2,3]:
Psphult[4,4]:=2*P2323:
Psphult[5,5]:=Psphult[4,4]:
Psphult[6,6]:=2*0.5*(P1111-P1122):

help_lac:=inv(I6+Psphult.(Clac-Cestult)):
Cestexvas:=((1-flac)*Cestult+flac*Clac.help_lac).inv((1-flac)*I6+flac*help_lac):

#bone microstructure
#-------------------
Cvas:=Matrix(6):

Pcylexvas:=Matrix(6):
P1111:=1/8*(5*Cestexvas[1,1]-3*Cestexvas[1,2])/Cestexvas[1,1]/(Cestexvas[1,1]-Cestexvas[1,2]):
P1122:=-1/8*(Cestexvas[1,1]+Cestexvas[1,2])/Cestexvas[1,1]/(Cestexvas[1,1]-Cestexvas[1,2]):
P2222:=P1111:
P2323:=1/8*1/(0.5*Cestexvas[4,4]):
P1313:=P2323:
```

```
P1212:=1/8*(3*Cestexvas[1,1]-Cestexvas[1,2])/Cestexvas[1,1]/(Cestexvas[1,1]-Cestexvas[1,2]):
Pcylexvas[1,1]:=P1111:
Pcylexvas[2,2]:=P2222:
Pcylexvas[1,2]:=P1122:
Pcylexvas[2,1]:=Pcylexvas[1,2]:
Pcylexvas[4,4]:=2*P2323:
Pcylexvas[5,5]:=2*P1313:
Pcylexvas[6,6]:=2*P1212:

help_vas:=inv(I6+Pcylexvas.(Cvas-Cestexvas)):
Cestmicro:=((1-fvas)*Cestexvas+fvas*Cvas.help_vas).inv((1-fvas)*I6+fvas*help_vas):

Destmicro:=inv(Cestmicro):
Eestmicro:=1/Destmicro[3,3];
EestmicroN:=1/Destmicro[1,1];
```

B.2 Determination of stress and strain states

This section includes the determination of the elastic, elastoplastic or ideal plastic stress and strain states in the specimen examined by Iyo et al. [39] and Sasaki et al. [67] using the stress resultants in Eqs. (4.5) and (4.6).

B.2.1 Elastoplastic and elastic stress and strain states in the longitudinal specimens examined by Iyo et al. [39]

```
restart;

#definition of material properties
#---------------------------------

Eel:=22842.75:                        # modulus of elasticity (in N/mm2)
Et:=15021.46:                         # gradient in stress-strain diagram after yield point in tension (in N/mm2)

fy:=11.09:                            # yield stress (in N/mm2)
h:=1:                                 # height in mm
b:=5:                                 # width in mm

M0:=27.21666667:                      # applied moment (in Nmm)


#determination of hpl with the help of the stress resultant N=int(sigma)dA=0
#---------------------------------------------------------------------------

A1:=k*Eel*0.5*(h-hpl-fy/(k*Eel))^2:
A2:=fy^2*0.5/(k*Eel):
A3:=fy*hpl:
A4:=k*Et*hpl^2*0.5:

N:=A2+A3+A4:

hpl1:=solve(N=A1,hpl);
hpl:=hpl1[2]:


#definition of h
#---------------

hy:=fy/(Eel*k):
```

```
ho:=h-hpl-hy:
hu:=hpl+hy:


#determination of d
#------------------

s1y:=k*Eel*hy:
s2y:=k*Et*hy+d:
d:=solve(s1y=s2y,d);


#determination of M
#------------------

s1:=k*Eel*z:
s2:=k*Et*z+d:

M:=int(s1*z*b,z=-ho..hy)+int(s2*z*b,z=hy..hu):

plot(M,k=2.97e-4..1e-3);


#determination of kappa
#----------------------

k1:=solve(M=M0,k);
k:=k1[1];


#solutions
#---------

M:=M;
hu:=hu;
ho:=ho;
hy:=hy;
hpl:=hpl;
sy1:=k*Eel*hy;
sy2:=k*Et*hy+d;
so:=k*Eel*ho;
su:=k*Et*hu+d;
eu:=k*hu;
eo:=-k*ho;


#defintition of elasto-plastic strain state
#------------------------------------------

h:=Vector([-ho,hy/2,hy,hy+(hu-hy)/2,hu]):
epsilon:=Vector(5):

for i from 1 to 5 do
   epsilon[i]:=k*h[i]:
end do:


#definition of elasto-plastis stress state
#-----------------------------------------

sigma:=Vector(5):

sigma[1]:=k*Eel*h[1]:
sigma[2]:=k*Eel*h[2]:
sigma[3]:=k*Eel*h[3]:
sigma[4]:=k*Et*h[4]+d:
sigma[5]:=k*Et*h[5]+d:
```

```
#determination of elastic strains
#---------------------------------

d:=Vector(5):
epsilon_pl:=Vector(5):
epsilon_el:=Vector(5):

for i from 1 to 5 do
   d[i]:=sigma[i]-Eel*epsilon[i]:
   epsilon_pl[i]:=-d[i]/Eel:
   epsilon_el[i]:=epsilon[i]-epsilon_pl[i]:
end do:

epsilon_el:=epsilon_el;
epsilon_pl:=epsilon_pl;
epsilon:=epsilon;
```

B.2.2 Elastic stress and strain states in the transverse specimens examined by Iyo et al. [39]

```
restart;

#definition of material properties
#---------------------------------

Eel:=12393.48:                    # modulus of elasticity (in N/mm2)

h:=1:                             # height (in mm)
b:=5:                             # width (in mm)

hel:=h/2:

M0:=22.23:                        # applied moment (in Nmm)


#determination of M
#------------------

s:=k*Eel*z:

M:=int(s*z*b,z=-h/2..h/2):

plot(M,k=2.97e-4..1e-3);


#determination of k
#------------------

k:=solve(M=M0,k);


#determination of elastic strains
#--------------------------------

h:=Vector([-hel,-hel/2,hel/2,hel]):

epsilon_el:=Vector(4):
for i from 1 to 4 do
   epsilon_el[i]:=k*h[i]:
end do:


#solutions
```

```
#---------

M:=M;
so:=k*Eel*(-hel);
su:=k*Eel*hel;

eo:=-k*hel;
eu:=k*hel;
epsilon_el:=epsilon_el;
```

B.2.3 Ideal plastic stress and strain states in the longitudinal specimens examined by Iyo et al. [39]

```
restart;

#definition of material properties
#-------------------------------

Eel:=12393.48:                    # modulus of elasticity (in N/mm2)
fyt:=11.0:                        # yield stress in tension (in N/mm2)
fyc:=51.2:                        # yield stress in compression (in N/mm2)

h:=1:                             # height (in mm)
b:=5:                             # width (in mm)

M0:=22.23:                        # applied moment (in Nmm)


#definition of h
#---------------

hyc:=fyc/(Eel*k):
hyt:=fyt/(Eel*k):
hplc:=h-hyc-hyt-hplt:
ho:=hplc+hyc:
hu:=hplt+hyt:


#determination of hpl with the help of the stress resultant N=int(sigma)dA=0
#-----------------------------------------------------------------------------

A1:=fyc*hplc:
A2:=0.5*fyc*hyc:
A3:=0.5*fyt*hyt:
A4:=fyt*hplt:

A12:=A1+A2:
A34:=A3+A4:

hplt:=solve(A12=A34,hplt);


#determination of M
#------------------

s1:=-fyc:
s2:=k*Eel*z:
s3:=fyt:
M:=int(s1*z*b,z=-ho..-hyc)+int(s2*z*b,z=-hyc..hyt)+int(s3*z*b,z=hyt..hu):

plot(M,k=2.97e-4..1e-3);
```

```
#determination of k
#------------------

k1:=solve(M=M0,k);
k:=k1[1];

#determination of elastic strains
#--------------------------------

epsilon_c:=-fyc/Eel:
epsilon_t:=fyt/Eel:

h:=Vector([-ho,-hyc,-hyc/2,hyt/2,hyt,hu]):

epsilon_el:=Vector(6):
epsilon_el[1]:=epsilon_c:
epsilon_el[2]:=epsilon_c:
epsilon_el[3]:=k*h[3]:
epsilon_el[4]:=k*h[4]:
epsilon_el[5]:=epsilon_t:
epsilon_el[6]:=epsilon_t:


#solutions
#---------

M:=M;
hu:=hu;
ho:=ho;
hyc:=hyc;
hyt:=hyt;
hplc:=hplc;
hplt:=hplt;
so:=-fyc;
su:=fyt;
syc:=-k*Eel*hyc;
syt:=k*Eel*hyt;
eo:=-k*ho;
eu:=k*hu;

epsilon_el:=epsilon_el;
```

B.2.4 Elastic stress and strain states in the longitudinal specimens examined by Sasaki et al. [67]

```
restart;

#definition of material properties
#---------------------------------

Eel:=22842.75:                      # modulus of elasticity (in N/mm2)

h:=0.5:                             # height (in mm)
b:=8:                               # width (in mm)

hel:=h/2:

M0:=1.369:                          # applied moment (in Nmm)


#determination of M
#------------------
```

```
s:=k*Eel*z:

M:=int(s*z*b,z=-h/2..h/2):

plot(M,k=2.97e-4..1e-3);


#determination of k
#------------------

k:=solve(M=M0,k);

#determination of elastic strains
#--------------------------------

h:=Vector([-hel,-hel/2,hel/2,hel]):

epsilon_el:=Vector(4):
for i from 1 to 4 do
   epsilon_el[i]:=k*h[i]:
end do:


#solutions
#---------

M:=M;
so:=k*Eel*(-hel);
su:=k*Eel*hel;
eo:=-k*hel;
eu:=k*hel;
epsilon_el:=epsilon_el;
```

B.3 Determination of Burgers parameters $\mu_{HA,KV}$, $\eta_{HA,KV}$ and $\eta_{HA,M}$

In this section, the source code of the program determining the Burgers parameters $\mu_{HA,KV}$, $\eta_{HA,KV}$ and $\eta_{HA,M}$ is shown:

```
restart:with(plots):with(linalg):with(LinearAlgebra):with(MTM):

#definition of calculation parameters
#-------------------------------------

kapproxmax:=4:                              #approximation order of gaver-wynn-rho algorithm for laplace inversion
Digits:=round(kapproxmax*2.1)+8:            #calculation precision
tmax:=100000:                               #time instant (in seconds) at end of observed viscoelastic response)
numberofintervals:=14:                      #number of intervals
tolerance:=1*10^(-(evalf(round(kapproxmax*2.1))+1)):


#defintion of material properties         (in GPa and GPa.s)
#--------------------------------

ccol3333:=17.9:       ccol1111:=11.7:
ccol1133:=7.1:        ccol1122:=5.1:      ccol1313:=3.3:
muhainst:=44.9:       kha:=82.6:
muwater:=0:           kwater:=2.3:

etahaM:=1.5e7:
etahaV:=1.5e5:
muhaV:=150:
```

```
ho:=0.4762991920:
hy:=0.1516248755:
hu:=0.5237008080:


#determination of volume fractions
#---------------------------------

flac:=0.020:
fvas:=0.050:
rhomu:=1.99:
fmupor:=fvas+flac:
rhoult:=(rhomu-1*fmupor)/(1-fmupor);

barfha:=0.46:
barfcol:=0.29:

tildeflac:=0.021:
ds:=-0.2000*rhoult+1.6580:
b:=1.47:
Dcol:=64:
vcol:=335.6:
vfib:=b*ds*5*Dcol:
barffib:=barfcol*vfib/vcol:
barfef:=1-barffib:

phihaef:=(1-barffib)/(1-barfcol):
checkfha:=phihaef*barfha/barfef:
checkfic:=1-checkfha:

brevefha:=barfha*(1-phihaef)/barffib:
brevefwetcol:=1-brevefha:

circlefcol:=barfcol/brevefwetcol:
circlefuw:=1-circlefcol:

fim:=1-circlefcol;
circlefcol:=circlefcol;
fwetcol:=brevefwetcol;
fha:=checkfha;
ffib:=barffib;
flac:=flac;
fvas:=fvas;


#input of experimental data (IYO 2004)
#-------------------------------------

Iy:=0.42:
epsilon_u:=0.0023:
l:=32:
h:=1:

E0:=14.2:
A1:=0.08:
t1:=49:
t2:=9.3e6:
B:=0.28:
G:=0.35:
Rexp:=array(1..numberofintervals+1):
Fexp:=array(1..numberofintervals+1):
step:=tmax/numberofintervals:

texp:=array(1..15):
texp[1]:=1:
texp[2]:=7.29568354608:
texp[3]:=31.8387278704:
texp[4]:=108.763941305:
```

```
texp[5]:=321.683062152:
texp[6]:=852.851887276:
texp[7]:=2030.99832312:
texp[8]:=4323.3497897:
texp[9]:=8269.20760983:
texp[10]:=14428.410878:
texp[11]:=23376.4603501:
texp[12]:=35720.7020611:
texp[13]:=52116.7122724:
texp[14]:=73280.7764716:
texp[15]:=100000:

for i from 1 to numberofintervals+1 do
   Rexp[i]:=E0*(A1*exp(-(texp[i]/t1)^B)+(1-A1)*exp(-(texp[i]/t2)^G)):
end do:

for i from 1 to numberofintervals+1 do
   Fexp[i]:=(4*Iy*Rexp[i]*epsilon_u)/(l*h/2):
end do:


#definition of strain state
#---------------------------

epsilon:= Vector(5, {(1) = -0.1525079723e-2, (2) = 0.2427466045e-3, (3) = 0.4854932089e-3, (4) = 0.8772152822e-3,
(5) = 0.1268937355e-2});


#definition of volumetric and deviatoric matrix
#----------------------------------------------

k:=1/3:
K:=Matrix([[k,k,k,0,0,0],[k,k,k,0,0,0],[k,k,k,0,0,0],[0,0,0,0,0,0],[0,0,0,0,0,0],[0,0,0,0,0,0]]):
J:=IdentityMatrix(6,6)-K:
I6:=IdentityMatrix(6,6):


#definition of Q, theta & phi for rotation
#-----------------------------------------

e1:=Vector([cos(phi)*cos(theta),sin(phi)*cos(theta),-sin(theta)]):
e2:=Vector([-sin(phi),cos(phi),0]):
e3:=Vector([cos(phi)*sin(theta),sin(phi)*sin(theta),cos(theta)]):

q11:=e1[1]:
q21:=e1[2]:
q31:=e1[3]:
q12:=e2[1]:
q22:=e2[2]:
q32:=e2[3]:
q13:=e3[1]:
q23:=e3[2]:
q33:=e3[3]:

a:=2/sqrt(2):
n:=sqrt(2):

Q:=Matrix(6):
Q[1,1]:=q11^2:
Q[1,2]:=q12^2:
Q[1,3]:=q13^2:
Q[1,4]:=a*q12*q13:
Q[1,5]:=a*q13*q11:
Q[1,6]:=a*q11*q12:
Q[2,1]:=q21^2:
Q[2,2]:=q22^2:
Q[2,3]:=q23^2:
Q[2,4]:=a*q22*q23:
Q[2,5]:=a*q23*q21:
```

```
Q[2,6]:=a*q21*q22:
Q[3,1]:=q31^2:
Q[3,2]:=q32^2:
Q[3,3]:=q33^2:
Q[3,4]:=a*q32*q33:
Q[3,5]:=a*q33*q31:
Q[3,6]:=a*q31*q32:
Q[4,1]:=n*q21*q31:
Q[4,2]:=n*q22*q32:
Q[4,3]:=n*q23*q33:
Q[4,4]:=q23*q32+q33*q22:
Q[4,5]:=q21*q33+q31*q23:
Q[4,6]:=q22*q31+q32*q21:
Q[5,1]:=n*q31*q11:
Q[5,2]:=n*q32*q12:
Q[5,3]:=n*q33*q13:
Q[5,4]:=q33*q12+q13*q32:
Q[5,5]:=q31*q13+q11*q33:
Q[5,6]:=q32*q11+q12*q31:
Q[6,1]:=n*q11*q21:
Q[6,2]:=n*q12*q22:
Q[6,3]:=n*q13*q23:
Q[6,4]:=q13*q22+q23*q12:
Q[6,5]:=q11*q23+q21*q13:
Q[6,6]:=q12*q21+q22*q11:

r:=1/2:
s:=(sqrt(5)+1)/4:
t:=(sqrt(5)-1)/4:

theta1:=evalf(arccos(t)):
phi1:=evalf(arccos(r/sin(theta1))):

theta2:=evalf(arccos(t)):
phi2:=evalf(arcsin(-s/sin(theta2))):

theta3:=evalf(arccos(t)):
phi3:=evalf(arccos(-r/sin(theta3))):

theta4:=evalf(arccos(t)):
phi4:=evalf(Pi-arcsin(-s/sin(theta4))):

theta5:=evalf(arccos(s)):
phi5:=evalf(arccos(t/sin(theta5))):

theta6:=evalf(arccos(s)):
phi6:=evalf(arcsin(-r/sin(theta6))):

theta7:=evalf(arccos(s)):
phi7:=evalf(arccos(-t/sin(theta7))):

theta8:=evalf(arccos(s)):
phi8:=evalf(Pi-arcsin(-r/sin(theta8))):

theta9:=evalf(arccos(r)):
phi9:=evalf(arccos(s/sin(theta9))):

theta10:=evalf(arccos(r)):
phi10:=evalf(arcsin(-t/sin(theta10))):

theta11:=evalf(arccos(r)):
phi11:=evalf(arccos(-s/sin(theta11))):

theta12:=evalf(arccos(r)):
phi12:=evalf(Pi-arcsin(-t/sin(theta12))):

theta13:=evalf(Pi/2):
phi13:=0:
```

```
theta14:=evalf(Pi/2):
phi14:=Pi/2:

theta15:=0:
phi15:=0:

Q1:=Matrix(6):
Q2:=Matrix(6):
Q3:=Matrix(6):
Q4:=Matrix(6):
Q5:=Matrix(6):
Q6:=Matrix(6):
Q7:=Matrix(6):
Q8:=Matrix(6):
Q9:=Matrix(6):
Q10:=Matrix(6):
Q11:=Matrix(6):
Q12:=Matrix(6):
Q13:=Matrix(6):
Q14:=Matrix(6):
Q15:=Matrix(6):

for i from 1 to 6 do
for j from 1 to 6 do
   Q1[i,j]:=eval(Q[i,j],[theta=theta1,phi=phi1]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q2[i,j]:=eval(Q[i,j],[theta=theta2,phi=phi2]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q3[i,j]:=eval(Q[i,j],[theta=theta3,phi=phi3]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q4[i,j]:=eval(Q[i,j],[theta=theta4,phi=phi4]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q5[i,j]:=eval(Q[i,j],[theta=theta5,phi=phi5]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q6[i,j]:=eval(Q[i,j],[theta=theta6,phi=phi6]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q7[i,j]:=eval(Q[i,j],[theta=theta7,phi=phi7]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q8[i,j]:=eval(Q[i,j],[theta=theta8,phi=phi8]):
end do:
end do:
```

```
for i from 1 to 6 do
for j from 1 to 6 do
   Q9[i,j]:=eval(Q[i,j],[theta=theta9,phi=phi9]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q10[i,j]:=eval(Q[i,j],[theta=theta10,phi=phi10]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q11[i,j]:=eval(Q[i,j],[theta=theta11,phi=phi11]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q12[i,j]:=eval(Q[i,j],[theta=theta12,phi=phi12]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q13[i,j]:=eval(Q[i,j],[theta=theta13,phi=phi13]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q14[i,j]:=eval(Q[i,j],[theta=theta14,phi=phi14]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q15[i,j]:=eval(Q[i,j],[theta=theta15,phi=phi15]):
end do:
end do:


#calculation of creep compliance
#-------------------------------

#wet collagen
#------------
Cwater:=3*kwater*K+2*muwater*J:
Cim:=Cwater:

Ccol:=Matrix(6):
Ccol[1,1]:=ccol1111:
Ccol[2,2]:=Ccol[1,1]:
Ccol[3,3]:=ccol3333:
Ccol[1,2]:=ccol1122:
Ccol[2,1]:=Ccol[1,2]:
Ccol[1,3]:=ccol1133:
Ccol[3,1]:=Ccol[1,3]:
Ccol[3,2]:=Ccol[3,1]:
Ccol[2,3]:=Ccol[3,2]:
Ccol[4,4]:=2*ccol1313:
Ccol[5,5]:=Ccol[4,4]:
Ccol[6,6]:=2*0.5*(Ccol[1,1]-Ccol[1,2]):

Pcylcol:=Matrix(6):
P1111:=1/8*(5*Ccol[1,1]-3*Ccol[1,2])/Ccol[1,1]/(Ccol[1,1]-Ccol[1,2]):
P1122:=-1/8*(Ccol[1,1]+Ccol[1,2])/Ccol[1,1]/(Ccol[1,1]-Ccol[1,2]):
```

```
P2222:=P1111:
P2323:=1/8*1/(0.5*Ccol[4,4]):
P1313:=P2323:
P1212:=1/8*(3*Ccol[1,1]-Ccol[1,2])/Ccol[1,1]/(Ccol[1,1]-Ccol[1,2]):
Pcylcol[1,1]:=P1111:
Pcylcol[2,2]:=P2222:
Pcylcol[1,2]:=P1122:
Pcylcol[2,1]:=Pcylcol[1,2]:
Pcylcol[4,4]:=2*P2323:
Pcylcol[5,5]:=2*P1313:
Pcylcol[6,6]:=2*P1212:

help_im:=inv(I6+Pcylcol.(Cim-Ccol)):
Cestwetcol:=((1-fim)*Ccol+fim*Cim.help_im).inv((1-fim)*I6+fim*help_im):

#fibril
#------
Cha:=3*kha*K+2*muhainst*J:

Cestfibnew:=Cestwetcol:
Cestfib:=(1+tolerance*10)*Cestfibnew:

while abs(Cestfib[1,1]-Cestfibnew[1,1])>tolerance do
   Cestfib:=Cestfibnew:

   Pcylfib:=Matrix(6):
   P1111:=1/8*(5*Cestfib[1,1]-3*Cestfib[1,2])/Cestfib[1,1]/(Cestfib[1,1]-Cestfib[1,2]):
   P1122:=-1/8*(Cestfib[1,1]+Cestfib[1,2])/Cestfib[1,1]/(Cestfib[1,1]-Cestfib[1,2]):
   P2222:=P1111:
   P2323:=1/8*1/(0.5*Cestfib[4,4]):
   P1313:=P2323:
   P1212:=1/8*(3*Cestfib[1,1]-Cestfib[1,2])/Cestfib[1,1]/(Cestfib[1,1]-Cestfib[1,2]):
   Pcylfib[1,1]:=P1111:
   Pcylfib[2,2]:=P2222:
   Pcylfib[1,2]:=P1122:
   Pcylfib[2,1]:=Pcylfib[1,2]:
   Pcylfib[4,4]:=2*P2323:
   Pcylfib[5,5]:=2*P1313:
   Pcylfib[6,6]:=2*P1212:

   C1111:=Cestfib[1,1]:
   C1122:=Cestfib[1,2]:
   C1133:=Cestfib[1,3]:
   C3333:=Cestfib[3,3]:
   C2323:=Cestfib[4,4]/2:

   Psphfib:=Matrix(6):

   D1:=(-2)*C1111^2*x^4*C3333+2*C2323^2*x^6*C3333-4*C1111*C2323^2*x^4-
          3*C1111^2*C2323*x^2+C1111^2*x^2*C3333+
          2*C1111*C2323^2*x^2-2*C2323*x^4*C1133^2-C1111*C1133^2*x^6+
          2*C1111*C1133^2*x^4+4*C2323^2*x^6*C1133-
          2*C1122*C1133^2*x^4+2*C2323*x^6*C1133^2+3*C1111^2*x^4*C2323+
            C1122*C1133^2*x^6-C1111^2*x^6*C2323+
          2*C1111*x^6*C2323^2+C1111^2*x^6*C3333-C1111*C1133^2*x^2-
          4*C2323^2*x^4*C1133+C1122*C1133^2*x^2+
            C1111^2*C2323-C1122*C1111*C2323-C1122*x^6*C1111*C3333+
          4*C1111*x^4*C2323*C1133-2*C1111*x^2*C2323*C1133-
          4*C1122*x^4*C2323*C1133+2*C1122*x^2*C2323*C1133+
          2*C1122*x^6*C2323*C1133-2*C1111*x^6*C2323*C1133-
          3*C1111*x^6*C2323*C3333+2*C1122*C1111*x^4*C3333-
            C1122*C2323*x^4*C3333-3*C1122*C1111*x^4*C2323-
            C1122*C1111*x^2*C3333+3*C1122*C1111*C2323*x^2+
          3*C1111*C2323*x^4*C3333+C1122*x^6*C1111*C2323+
            C1122*x^6*C2323*C3333:

   D2:=2*C2323*x^4*C1133+C2323*x^4*C3333+C1111*x^4*C2323-
       2*C2323*x^2*C1133-2*C1111*C2323*x^2+
         C1111*C2323+x^4*C1133^2-C1111*x^4*C3333-
```

```
        x^2*C1133^2+C1111*x^2*C3333:


   P1111int:=-((-5)*C1111*x^4*C3333-3*C1122*x^2*C3333-
                3*C1122*x^4*C2323+
                3*C1122*x^4*C3333+5*C1111*x^4*C2323-
               10*C1111*C2323*x^2+2*x^4*C1133^2+
                8*C2323*x^4*C3333-6*C2323^2*x^4+
                4*C2323*x^4*C1133+6*C1122*C2323*x^2+
                5*C1111*C2323+5*C1111*x^2*C3333-
                4*C2323*x^2*C1133+6*C2323^2*x^2-
                2*x^2*C1133^2-3*C1122*C2323)*(-1+x^2)/D1:

   P1122int:=(C1111*C2323-2*C1111*C2323*x^2+
              C1111*x^2*C3333+C1122*C2323-
            2*C1122*C2323*x^2+C1122*x^2*C3333+
              C1111*x^4*C2323-C1111*x^4*C3333+
              C1122*x^4*C2323-C1122*x^4*C3333-2*C2323^2*x^2+
            2*C2323^2*x^4-4*C2323*x^2*C1133+
            4*C2323*x^4*C1133-2*x^2*C1133^2+
            2*x^4*C1133^2)*(-1+x^2)/D1:

   P1133int:=(-1+x^2)*x^2*(C2323+C1133)/D2:

   P2323int:=(4*C1111*C2323*x^2-8*C2323*x^4*C1133-
              2*x^4*C1133^2-C1122*x^4*C3333-
              8*C1111*x^4*C2323+3*C1111*x^4*C3333+
              4*C1111*x^4*C1133-4*C1122*x^4*C1133+
              2*C1122*x^6*C1133-2*C1111*x^6*C1133+
                C1122*x^6*C1111-3*C1122*x^4*C1111+
              3*C1122*C1111*x^2-2*C1111*x^2*C1133+
              2*C1122*x^2*C1133+8*x^6*C2323*C1133-
              3*x^6*C1111*C3333+4*x^6*C2323*C3333+
              4*C1111*x^6*C2323+C1122*x^6*C3333+
              3*C1111^2*x^4-C1111^2*x^6+
              2*C1133^2*x^6-3*C1111^2*x^2+
                C1111^2-C1122*C1111)/D1:

   P3333int:=x^2*(x^2*C2323-C1111*x^2+
             C1111)/D2:

   P1111:=evalf(1/16*int(P1111int,x=-1..1)):
   P1122:=evalf(1/16*int(P1122int,x=-1..1)):
   P1133:=evalf(1/4*int(P1133int,x=-1..1)):
   P2323:=evalf(1/16*int(P2323int,x=-1..1)):
   P3333:=evalf(1/2*int(P3333int,x=-1..1)):

   Psphfib[1,1]:=P1111:
   Psphfib[2,2]:=Psphfib[1,1]:
   Psphfib[3,3]:=P3333:
   Psphfib[1,2]:=P1122:
   Psphfib[1,3]:=P1133:
   Psphfib[2,3]:=Psphfib[1,3]:
   Psphfib[2,1]:=Psphfib[1,2]:
   Psphfib[3,1]:=Psphfib[1,3]:
   Psphfib[3,2]:=Psphfib[2,3]:
   Psphfib[4,4]:=2*P2323:
   Psphfib[5,5]:=Psphfib[4,4]:
   Psphfib[6,6]:=2*0.5*(P1111-P1122):
   help_wetcol:=inv(I6+Pcylfib.(Cestwetcol-Cestfib)):
   help_ha:=inv(I6+Psphfib.(Cha-Cestfib)):
   Cestfibnew:=(fwetcol*Cestwetcol.help_wetcol+(1-fwetcol)*Cha.help_ha).
               inv(fwetcol*help_wetcol+(1-fwetcol)*help_ha):

end do:
Cestfib:=Cestfibnew:

#ultrastructure
#--------------
```

```
Pcylef:=Matrix(6):
P1111:=1/8*(5*Cestefsym[1,1]-3*Cestefsym[1,2])/Cestefsym[1,1]/(Cestefsym[1,1]-Cestefsym[1,2]):
P1122:=-1/8*(Cestefsym[1,1]+Cestefsym[1,2])/Cestefsym[1,1]/(Cestefsym[1,1]-Cestefsym[1,2]):
P2222:=P1111:
P2323:=1/8*1/(0.5*Cestefsym[4,4]):
P1313:=P2323:
P1212:=1/8*(3*Cestefsym[1,1]-Cestefsym[1,2])/Cestefsym[1,1]/(Cestefsym[1,1]-Cestefsym[1,2]):
Pcylef[1,1]:=P1111:
Pcylef[2,2]:=P2222:
Pcylef[1,2]:=P1122:
Pcylef[2,1]:=Pcylef[1,2]:
Pcylef[4,4]:=2*P2323:
Pcylef[5,5]:=2*P1313:
Pcylef[6,6]:=2*P1212:

help_fib:=inv(I6+Pcylef.(Cestfib-Cestefsym)):
Cestult:=((1-ffib)*Cestefsym+ffib*Cestfib.help_fib).inv((1-ffib)*I6+ffib*help_fib):

#extravascular bone material
#----------------------------
Clac:=Matrix(6):

C1111:=Cestultsym[1,1]:
C1122:=Cestultsym[1,2]:
C1133:=Cestultsym[1,3]:
C3333:=Cestultsym[3,3]:
C2323:=Cestultsym[4,4]/2:

Psphult:=Matrix(6):

D1:=(-2)*C1111^2*x^4*C3333+2*C2323^2*x^6*C3333-4*C1111*C2323^2*x^4-
       3*C1111^2*C2323*x^2+C1111^2*x^2*C3333+
       2*C1111*C2323^2*x^2-2*C2323*x^4*C1133^2-C1111*C1133^2*x^6+
       2*C1111*C1133^2*x^4+4*C2323^2*x^6*C1133-
       2*C1122*C1133^2*x^4+2*C2323*x^6*C1133^2+3*C1111^2*x^4*C2323+
         C1122*C1133^2*x^6-C1111^2*x^6*C2323+
       2*C1111*x^6*C2323^2+C1111^2*x^6*C3333-C1111*C1133^2*x^2-
       4*C2323^2*x^4*C1133+C1122*C1133^2*x^2+
         C1111^2*C2323-C1122*C1111*C2323-C1122*x^6*C1111*C3333+
       4*C1111*x^4*C2323*C1133-2*C1111*x^2*C2323*C1133-
       4*C1122*x^4*C2323*C1133+2*C1122*x^2*C2323*C1133+
       2*C1122*x^6*C2323*C1133-2*C1111*x^6*C2323*C1133-
       3*C1111*x^6*C2323*C3333+2*C1122*C1111*x^4*C3333-
         C1122*C2323*x^4*C3333-3*C1122*C1111*x^4*C2323-
         C1122*C1111*x^2*C3333+3*C1122*C1111*C2323*x^2+
       3*C1111*C2323*x^4*C3333+C1122*x^6*C1111*C2323+
         C1122*x^6*C2323*C3333:

D2:=2*C2323*x^4*C1133+C2323*x^4*C3333+C1111*x^4*C2323-
    2*C2323*x^2*C1133-2*C1111*C2323*x^2+
      C1111*C2323+x^4*C1133^2-C1111*x^4*C3333-
      x^2*C1133^2+C1111*x^2*C3333:

P1111int:=-((-5)*C1111*x^4*C3333-3*C1122*x^2*C3333-
               3*C1122*x^4*C2323+
               3*C1122*x^4*C3333+5*C1111*x^4*C2323-
              10*C1111*C2323*x^2+2*x^4*C1133^2+
               8*C2323*x^4*C3333-6*C2323^2*x^4+
               4*C2323*x^4*C1133+6*C1122*C2323*x^2+
               5*C1111*C2323+5*C1111*x^2*C3333-
               4*C2323*x^2*C1133+6*C2323^2*x^2-
               2*x^2*C1133^2-3*C1122*C2323)*(-1+x^2)/D1:

P1122int:=(C1111*C2323-2*C1111*C2323*x^2+
           C1111*x^2*C3333+C1122*C2323-
         2*C1122*C2323*x^2+C1122*x^2*C3333+
           C1111*x^4*C2323-C1111*x^4*C3333+
           C1122*x^4*C2323-C1122*x^4*C3333-2*C2323^2*x^2+
          2*C2323^2*x^4-4*C2323*x^2*C1133+
```

```
          4*C2323*x^4*C1133-2*x^2*C1133^2+
         2*x^4*C1133^2)*(-1+x^2)/D1:

P1133int:=(-1+x^2)*x^2*(C2323+C1133)/D2:

P2323int:=(4*C1111*C2323*x^2-8*C2323*x^4*C1133-
           2*x^4*C1133^2-C1122*x^4*C3333-
           8*C1111*x^4*C2323+3*C1111*x^4*C3333+
           4*C1111*x^4*C1133-4*C1122*x^4*C1133+
           2*C1122*x^6*C1133-2*C1111*x^6*C1133+
             C1122*x^6*C1111-3*C1122*x^4*C1111+
           3*C1122*C1111*x^2-2*C1111*x^2*C1133+
           2*C1122*x^2*C1133+8*x^6*C2323*C1133-
           3*x^6*C1111*C3333+4*x^6*C2323*C3333+
           4*C1111*x^6*C2323+C1122*x^6*C3333+
           3*C1111^2*x^4-C1111^2*x^6+
           2*C1133^2*x^6-3*C1111^2*x^2+
             C1111^2-C1122*C1111)/D1:

P3333int:=x^2*(x^2*C2323-C1111*x^2+
          C1111)/D2:

P1111:=evalf(1/16*int(P1111int,x=-1..1)):
P1122:=evalf(1/16*int(P1122int,x=-1..1)):
P1133:=evalf(1/4*int(P1133int,x=-1..1)):
P2323:=evalf(1/16*int(P2323int,x=-1..1)):
P3333:=evalf(1/2*int(P3333int,x=-1..1)):

Psphult[1,1]:=P1111:
Psphult[2,2]:=Psphult[1,1]:
Psphult[3,3]:=P3333:
Psphult[1,2]:=P1122:
Psphult[1,3]:=P1133:
Psphult[2,3]:=Psphult[1,3]:
Psphult[2,1]:=Psphult[1,2]:
Psphult[3,1]:=Psphult[1,3]:
Psphult[3,2]:=Psphult[2,3]:
Psphult[4,4]:=2*P2323:
Psphult[5,5]:=Psphult[4,4]:
Psphult[6,6]:=2*0.5*(P1111-P1122):

help_lac:=inv(I6+Psphult.(Clac-Cestultsym)):
Cestexvas:=((1-flac)*Cestultsym+flac*Clac.help_lac).inv((1-flac)*I6+flac*help_lac):

#bone microstructure
#-------------------
Cvas:=Matrix(6):

Pcylexvas:=Matrix(6):
P1111:=1/8*(5*Cestexvassym[1,1]-3*Cestexvassym[1,2])/Cestexvassym[1,1]/(Cestexvassym[1,1]-Cestexvassym[1,2]):
P1122:=-1/8*(Cestexvassym[1,1]+Cestexvassym[1,2])/Cestexvassym[1,1]/(Cestexvassym[1,1]-Cestexvassym[1,2]):
P2222:=P1111:
P2323:=1/8*1/(0.5*Cestexvassym[4,4]):
P1313:=P2323:
P1212:=1/8*(3*Cestexvassym[1,1]-Cestexvassym[1,2])/Cestexvassym[1,1]/(Cestexvassym[1,1]-Cestexvassym[1,2]):
Pcylexvas[1,1]:=P1111:
Pcylexvas[2,2]:=P2222:
Pcylexvas[1,2]:=P1122:
Pcylexvas[2,1]:=Pcylexvas[1,2]:
Pcylexvas[4,4]:=2*P2323:
Pcylexvas[5,5]:=2*P1313:
Pcylexvas[6,6]:=2*P1212:

help_vas:=inv(I6+Pcylexvas.(Cvas-Cestexvassym)):
Cestmicro:=((1-fvas)*Cestexvassym+fvas*Cvas.help_vas).inv((1-fvas)*I6+fvas*help_vas):


#definition of parameters for genetic algorithm
#----------------------------------------------
```

```
numberofmaximumevolutioncycles:=2500:
numberofactualevolutioncycle:=0:

initialscattermuhaV:=10:
initialscatteretahaM:=initialscattermuhaV*etahaM/muhaV:
initialscatteretahaV:=initialscattermuhaV*etahaV/muhaV:

scattermodification:=0:
lengthofsuccesscontrol:=5:
convergencersquare:=0.01:
evolution:=1:
successcounter:=0:
successarray:=Array(1..lengthofsuccesscontrol):
parameterchange:=0:


#optimization of rheological parameters by means of evolution strategy
#------------------------------------------------------------------------

while evolution=1 do
   numberofactualevolutioncycle:=numberofactualevolutioncycle+1:
   print("EVOLUTION CYCLE: ",numberofactualevolutioncycle):
   successcounter:=successcounter+1:
   if numberofactualevolutioncycle=1 then
      scatteretahaM:=initialscatteretahaM:
      scatteretahaV:=initialscatteretahaV:
      scattermuhaV:=initialscattermuhaV:
   elif numberofactualevolutioncycle>1 and scattermodification=1 then

      scatteretahaM:=newscatteretahaM:
      scatteretahaV:=newscatteretahaV:
      scattermuhaV:=newscattermuhaV:
      scattermodification:=0:
   end if:

   if parameterchange=1 then
      J3333parentgwr:=array(1..numberofintervals+1):
      for i from 1 to numberofintervals+1 do
         J3333parentgwr[i]:=J3333childgwr[i]:
         etahaM:=etahaMchild:
         etahaV:=etahaVchild:
         muhaV:=muhaVchild:
      end do:
   end if:
   etahaMchild:=etahaM+stats[random,normald](1)*scatteretahaM:
   etahaVchild:=etahaV+stats[random,normald](1)*scatteretahaV:
   muhaVchild:=muhaV+stats[random,normald](1)*scattermuhaV:
   print("etahaMparent",etahaM,"etahaVparent=",etahaV," muhaVparent=",muhaV);
   print("etahaMchild",etahaMchild,"etahaVchild=",etahaVchild," muhaVchild=",muhaVchild);
   print("scatteretahaM",scatteretahaM);
   print("scatteretahaV",scatteretahaV);
   print("ScattermuhaV",scattermuhaV);
   print("successarray",successarray);


#calculation of creep compliance for extrafibrilliar space in LC-domain
#------------------------------------------------------------------------

   t:=array(1..15):
   t[1]:=1:
   t[2]:=7.29568354608:
   t[3]:=31.8387278704:
   t[4]:=108.763941305:
   t[5]:=321.683062152:
   t[6]:=852.851887276:
   t[7]:=2030.99832312:
   t[8]:=4323.3497897:
   t[9]:=8269.20760983:
```

```
   t[10]:=14428.410878:
   t[11]:=23376.4603501:
   t[12]:=35720.7020611:
   t[13]:=52116.7122724:
   t[14]:=73280.7764716:
   t[15]:=100000:

   if numberofactualevolutioncycle=1 then
      for h from 1 to kapproxmax do
         for i from 1 to (numberofintervals+1) do
            for j from 1 to (h+1) do

#parent
#------
               p[i,j]:=evalf((h+(j-1))*log(2)/t[i]):
               muha[i,j]:=1/(1/muhainst+1/(p[i,j]*etahaM)+1/(p[i,j]*etahaV+muhaV)):

         Cwater_GLOBAL:=3*kwater*K+2*muwater*J:
         Cha_GLOBAL:=3*kha*K+2*muha[i,j]*J:

         tolerance:=1*10^(-round(kapproxmax*2.1)+5):
         Cestnew:=3*kha*K+2*muhainst*J:
         Cest:=(1+tolerance*10)*Cestnew:

         while abs(Cest[1,1]-Cestnew[1,1])>tolerance do
            Cest:=Cestnew:
            Dest:=inv(Cest):
            muestef:=1/(2*Dest[4,4]):
            Eestef:=1/Dest[1,1]:
            kestef:=(muestef*Eestef)/(3*(3*muestef-Eestef)):

            alphaef:=3*kestef/(3*kestef+4*muestef):
            betaef:=6*(kestef+2*muestef)/(5*(3*kestef+4*muestef)):
            Ssphef:=alphaef*K+betaef*J:
            Psphef:=Ssphef.inv(Cest):

            nup:=(3*kestef-2*muestef)/(6*kestef+2*muestef):
            Scylef:=Matrix(6):
            Scylef[1,1]:=(5-4*nup)/(8*(1-nup)):
            Scylef[2,2]:=Scylef[1,1]:
            Scylef[1,2]:=(-1+4*nup)/(8*(1-nup)):
            Scylef[2,1]:=Scylef[1,2]:
            Scylef[1,3]:=nup/(2*(1-nup)):
            Scylef[2,3]:=Scylef[1,3]:
            Scylef[4,4]:=2*(1/4):
            Scylef[5,5]:=Scylef[4,4]:
            Scylef[6,6]:=2*((3-4*nup)/(8*(1-nup))):
            Pcyl_LOCAL:=Scylef.inv(Cest):

            Pcyl1:=Q1.Pcyl_LOCAL.transpose(Q1):
            Pcyl2:=Q2.Pcyl_LOCAL.transpose(Q2):
            Pcyl3:=Q3.Pcyl_LOCAL.transpose(Q3):
            Pcyl4:=Q4.Pcyl_LOCAL.transpose(Q4):
            Pcyl5:=Q5.Pcyl_LOCAL.transpose(Q5):
            Pcyl6:=Q6.Pcyl_LOCAL.transpose(Q6):
            Pcyl7:=Q7.Pcyl_LOCAL.transpose(Q7):
            Pcyl8:=Q8.Pcyl_LOCAL.transpose(Q8):
            Pcyl9:=Q9.Pcyl_LOCAL.transpose(Q9):
            Pcyl10:=Q10.Pcyl_LOCAL.transpose(Q10):
            Pcyl11:=Q11.Pcyl_LOCAL.transpose(Q11):
            Pcyl12:=Q12.Pcyl_LOCAL.transpose(Q12):
            Pcyl13:=Q13.Pcyl_LOCAL.transpose(Q13):
            Pcyl14:=Q14.Pcyl_LOCAL.transpose(Q14):
            Pcyl15:=Q15.Pcyl_LOCAL.transpose(Q15):

            help_ha:=1/15*inv(I6+Pcyl1.(Cha_GLOBAL-Cest))+1/15*inv(I6+Pcyl2.(Cha_GLOBAL-Cest))+
            1/15*inv(I6+Pcyl3.(Cha_GLOBAL-Cest))+1/15*inv(I6+Pcyl4.(Cha_GLOBAL-Cest))+
            1/15*inv(I6+Pcyl5.(Cha_GLOBAL-Cest))+1/15*inv(I6+Pcyl6.(Cha_GLOBAL-Cest))+
            1/15*inv(I6+Pcyl7.(Cha_GLOBAL-Cest))+1/15*inv(I6+Pcyl8.(Cha_GLOBAL-Cest))+
```

```
            1/15*inv(I6+Pcyl9.(Cha_GLOBAL-Cest))+1/15*inv(I6+Pcyl10.(Cha_GLOBAL-Cest))+
            1/15*inv(I6+Pcyl11.(Cha_GLOBAL-Cest))+1/15*inv(I6+Pcyl12.(Cha_GLOBAL-Cest))+
            1/15*inv(I6+Pcyl13.(Cha_GLOBAL-Cest))+1/15*inv(I6+Pcyl14.(Cha_GLOBAL-Cest))+
            1/15*inv(I6+Pcyl15.(Cha_GLOBAL-Cest)):
            help_pores:=inv(I6+Psphef.(Cwater_GLOBAL-Cest)):
            Cestnew:=(fha*Cha_GLOBAL.help_ha+(1-fha)*Cwater_GLOBAL.help_pores).inv(fha*help_ha+(1-fha)*help_pores):

         end do:

         Cesteflc:=Cestnew:

         Cestultlc:=Matrix(6):
         Cestexvaslc:=Matrix(6):
         Cestmicrolc:=Matrix(6):

         for k from 1 to 6 do
         for l from 1 to 6 do
            Cestultlc[k,l]:=eval(Cestult[k,l],[Cestefsym=Cesteflc]):
         end do:
         end do:

         for k from 1 to 6 do
         for l from 1 to 6 do
            Cestexvaslc[k,l]:=eval(Cestexvas[k,l],[Cestultsym=Cestultlc]):
         end do:
         end do:

         for k from 1 to 6 do
         for l from 1 to 6 do
            Cestmicrolc[k,l]:=eval(Cestmicro[k,l],[Cestexvassym=Cestexvaslc]):
         end do:
         end do:

         Destmicrolc:=inv(Cestmicrolc):
         J3333parent[i,j]:=Destmicrolc[3,3]:

      end do:
   end do:


#calculation of GAVER-functionals
#---------------------------------

         for i from 1 to numberofintervals+1 do
            CFfuncK[i]:=0:
            for j from 0 to h do
               indexj:=j+1:
               CFfuncK[i]:=CFfuncK[i]+log(2)*h/t[i]*binomial(2*h,h)*(-1)^j*binomial(h,j)*
               J3333parent[i,indexj]/p[i,indexj]:
            end do:
         end do:
         for i from 1 to numberofintervals+1 do
            CFgaverfun[h,i]:=evalf(CFfuncK[i]):
         end do:
      end do:


#inversion of LC-transforms through WYNN-RHO-acceleration
#--------------------------------------------------------

      J3333parentgwr:=array(1..numberofintervals+1):
      for h from 1 to numberofintervals+1 do
         for i from 1 to kapproxmax+2 do
            for j from 1 to (kapproxmax+2) do
               if i=1 then
                  rhoCF[i,j]:=0:
               elif i=2 then
                  if j=1 then
                     rhoCF[i,j]:=0:
```

```
                elif j>1 and j<(kapproxmax+2) then
                   rhoCF[i,j]:=CFgaverfun[j-1,h]:
                end if:
             else
                if j<(kapproxmax+2-(i-2)) then
                   rhoCF[i,j]:=rhoCF[i-2,j+1]+(i-2)/(rhoCF[i-1,j+1]-rhoCF[i-1,j]):
                else
                   rhoCF[i,j]:=0:
                end if:
             end if:
          end do:
       end do:
       J3333parentgwr[h]:=evalf(rhoCF[kapproxmax+2,1]):
    end do:

             end if:


#child
#-----
 for h from 1 to kapproxmax do
       for i from 1 to (numberofintervals+1) do
          for j from 1 to (h+1) do

             p[i,j]:=evalf((h+(j-1))*log(2)/t[i]):
       muha[i,j]:=1/(1/muhainst+1/(p[i,j]*etahaMchild)+1/(p[i,j]*etahaVchild+muhaVchild)):

       Cwater_GLOBAL:=3*kwater*K+2*muwater*J:
       Cha_GLOBAL:=3*kha*K+2*muha[i,j]*J:

       tolerance:=1*10^(-round(kapproxmax*2.1)+5):
       Cestnew:=3*kha*K+2*muhainst*J:
       Cest:=(1+tolerance*10)*Cestnew:

       while abs(Cest[1,1]-Cestnew[1,1])>tolerance do
          Cest:=Cestnew:
          Dest:=inv(Cest):
          muestef:=1/(2*Dest[4,4]):
          Eestef:=1/Dest[1,1]:
          kestef:=(muestef*Eestef)/(3*(3*muestef-Eestef)):

          alphaef:=3*kestef/(3*kestef+4*muestef):
          betaef:=6*(kestef+2*muestef)/(5*(3*kestef+4*muestef)):
          Ssphef:=alphaef*K+betaef*J:
          Psphef:=Ssphef.inv(Cest):

          nup:=(3*kestef-2*muestef)/(6*kestef+2*muestef):
          Scylef:=Matrix(6):
          Scylef[1,1]:=(5-4*nup)/(8*(1-nup)):
          Scylef[2,2]:=Scylef[1,1]:
          Scylef[1,2]:=(-1+4*nup)/(8*(1-nup)):
          Scylef[2,1]:=Scylef[1,2]:
          Scylef[1,3]:=nup/(2*(1-nup)):
          Scylef[2,3]:=Scylef[1,3]:
          Scylef[4,4]:=2*(1/4):
          Scylef[5,5]:=Scylef[4,4]:
          Scylef[6,6]:=2*((3-4*nup)/(8*(1-nup))):
          Pcyl_LOCAL:=Scylef.inv(Cest):

          Pcyl1:=Q1.Pcyl_LOCAL.transpose(Q1):
          Pcyl2:=Q2.Pcyl_LOCAL.transpose(Q2):
          Pcyl3:=Q3.Pcyl_LOCAL.transpose(Q3):
          Pcyl4:=Q4.Pcyl_LOCAL.transpose(Q4):
          Pcyl5:=Q5.Pcyl_LOCAL.transpose(Q5):
          Pcyl6:=Q6.Pcyl_LOCAL.transpose(Q6):
          Pcyl7:=Q7.Pcyl_LOCAL.transpose(Q7):
          Pcyl8:=Q8.Pcyl_LOCAL.transpose(Q8):
          Pcyl9:=Q9.Pcyl_LOCAL.transpose(Q9):
          Pcyl10:=Q10.Pcyl_LOCAL.transpose(Q10):
```

```
            Pcyl11:=Q11.Pcyl_LOCAL.transpose(Q11):
            Pcyl12:=Q12.Pcyl_LOCAL.transpose(Q12):
            Pcyl13:=Q13.Pcyl_LOCAL.transpose(Q13):
            Pcyl14:=Q14.Pcyl_LOCAL.transpose(Q14):
            Pcyl15:=Q15.Pcyl_LOCAL.transpose(Q15):

            help_ha:=1/15*inv(I6+Pcyl1.(Cha_GLOBAL-Cest))+1/15*inv(I6+Pcyl2.(Cha_GLOBAL-Cest))+
            1/15*inv(I6+Pcyl3.(Cha_GLOBAL-Cest))+1/15*inv(I6+Pcyl4.(Cha_GLOBAL-Cest))+
            1/15*inv(I6+Pcyl5.(Cha_GLOBAL-Cest))+1/15*inv(I6+Pcyl6.(Cha_GLOBAL-Cest))+
            1/15*inv(I6+Pcyl7.(Cha_GLOBAL-Cest))+1/15*inv(I6+Pcyl8.(Cha_GLOBAL-Cest))+
            1/15*inv(I6+Pcyl9.(Cha_GLOBAL-Cest))+1/15*inv(I6+Pcyl10.(Cha_GLOBAL-Cest))+
            1/15*inv(I6+Pcyl11.(Cha_GLOBAL-Cest))+1/15*inv(I6+Pcyl12.(Cha_GLOBAL-Cest))+
            1/15*inv(I6+Pcyl13.(Cha_GLOBAL-Cest))+1/15*inv(I6+Pcyl14.(Cha_GLOBAL-Cest))+
            1/15*inv(I6+Pcyl15.(Cha_GLOBAL-Cest)):
            help_pores:=inv(I6+Psphef.(Cwater_GLOBAL-Cest)):
            Cestnew:=(fha*Cha_GLOBAL.help_ha+(1-fha)*Cwater_GLOBAL.help_pores).inv(fha*help_ha+(1-fha)*help_pores):

         end do:

         Cesteflc:=Cestnew:

         Cestultlc:=Matrix(6):
         Cestexvaslc:=Matrix(6):
         Cestmicrolc:=Matrix(6):

         for k from 1 to 6 do
         for l from 1 to 6 do
            Cestultlc[k,l]:=eval(Cestult[k,l],[Cestefsym=Cesteflc]):
         end do:
         end do:

         for k from 1 to 6 do
         for l from 1 to 6 do
            Cestexvaslc[k,l]:=eval(Cestexvas[k,l],[Cestultsym=Cestultlc]):
         end do:
         end do:

         for k from 1 to 6 do
         for l from 1 to 6 do
            Cestmicrolc[k,l]:=eval(Cestmicro[k,l],[Cestexvassym=Cestexvaslc]):
         end do:
         end do:

         Destmicrolc:=inv(Cestmicrolc):
         J3333child[i,j]:=Destmicrolc[3,3]:

      end do:
   end do:


#calvulation of GAVER-functionals
#---------------------------------

         for i from 1 to numberofintervals+1 do
            CFfuncKchild[i]:=0:
            for j from 0 to h do
               indexj:=j+1:
               CFfuncKchild[i]:=CFfuncKchild[i]+log(2)*h/t[i]*binomial(2*h,h)*(-1)^j*binomial(h,j)*
               J3333child[i,indexj]/p[i,indexj]:
            end do:
         end do:
         for i from 1 to numberofintervals+1 do
            CFgaverfunchild[h,i]:=evalf(CFfuncKchild[i]):
         end do:
      end do:


#inversion of LC-transforms through WYNN-RHO-acceleration
#--------------------------------------------------------
```

```
      J3333childgwr:=array(1..numberofintervals+1):
      for h from 1 to numberofintervals+1 do
         for i from 1 to kapproxmax+2 do
            for j from 1 to (kapproxmax+2) do
               if i=1 then
                  rhoCFchild[i,j]:=0:
               elif i=2 then
                  if j=1 then
                     rhoCFchild[i,j]:=0:
                  elif j>1 and j<(kapproxmax+2) then
                     rhoCFchild[i,j]:=CFgaverfunchild[j-1,h]:
                  end if:
               else
                  if j<(kapproxmax+2-(i-2)) then
                     rhoCFchild[i,j]:=rhoCFchild[i-2,j+1]+(i-2)/(rhoCFchild[i-1,j+1]-rhoCFchild[i-1,j]):
                  else
                     rhoCFchild[i,j]:=0:
                  end if:
               end if:
            end do:
         end do:
         J3333childgwr[h]:=evalf(rhoCFchild[kapproxmax+2,1]):
      end do:


#execution of evolution step
#---------------------------

#determination of stress state and moment
#----------------------------------------
   l:=32:
   b:=5:

   hz:=Vector([-ho,hy/2,hy,hy+(hu-hy)/2,hu]):

   s_parent:=matrix(5,numberofintervals+1):
   F_parent:=array(1..numberofintervals+1):
   s_child:=matrix(5,numberofintervals+1):
   F_child:=array(1..numberofintervals+1):

   for i from 1 to 5 do
      for j from 1 to numberofintervals+1 do
         s_parent[i,j]:=1/J3333parentgwr[j]*epsilon[i]:
         s_child[i,j]:=1/J3333childgwr[j]*epsilon[i]:
      end do:
   end do:

   for i from 1 to numberofintervals+1 do

      k1_parent:=(s_parent[3,i]-s_parent[2,i])/(hz[3]-hz[2]);
      k2_parent:=(s_parent[5,i]-s_parent[4,i])/(hz[5]-hz[4]);

      sy1_parent:=k1_parent*hz[3];
      sy2_parent:=k2_parent*hz[3]+d_parent;
      d_parent1:=solve(sy1_parent=sy2_parent,d_parent);

      s1_parent:=k1_parent*z;
      s2_parent:=k2_parent*z+d_parent1;
      M_parent:=int(s1_parent*z*b,z=hz[1]..hz[3])+int(s2_parent*z*b,z=hz[3]..hz[5]);

      F_parent[i]:=4*M_parent/l;

      k1_child:=(s_child[3,i]-s_child[2,i])/(hz[3]-hz[2]);
      k2_child:=(s_child[5,i]-s_child[4,i])/(hz[5]-hz[4]);

      sy1_child:=k1_child*hz[3];
      sy2_child:=k2_child*hz[3]+d_child;
      d_child1:=solve(sy1_child=sy2_child,d_child);
```

```
      s1_child:=k1_child*z;
      s2_child:=k2_child*z+d_child1;
      M_child:=int(s1_child*z*b,z=hz[1]..hz[3])+int(s2_child*z*b,z=hz[3]..hz[5]);

      F_child[i]:=4*M_child/l;
   end do:


#evolution
#---------
   r:=0:
   rchild:=0:
   expmean:=0:
   rsquarecounter:=0:

   for i from 1 to numberofintervals+1 do
      rsquarecounter:=rsquarecounter+1:
      expvalues[rsquarecounter]:=Fexp[i]:
      modelpredictions[rsquarecounter]:=F_parent[i]:
      modelpredictionschild[rsquarecounter]:=F_child[i]:
   end do:

   for i from 1 to rsquarecounter do
      expmean:=expmean+expvalues[i]/rsquarecounter:
   end do:

   sse:=array(1..rsquarecounter):
   ssechild:=array(1..rsquarecounter):
   sst:=array(1..rsquarecounter):

   for i from 1 to rsquarecounter do
      sst[i]:=(expvalues[i]):
      sse[i]:=(modelpredictions[i]-expvalues[i]):
      ssechild[i]:=(modelpredictionschild[i]-expvalues[i]):
   end do:
   for i from 1 to rsquarecounter do
   r:=r+sqrt((sse[i]/sst[i])^2);
   rchild:=rchild+sqrt((ssechild[i]/sst[i])^2);
   end do:
   print("R: ",r):
   print("R (child): ",rchild):
   if rchild<r then
      parameterchange:=1:
      successarray(successcounter):=successarray(successcounter)+1:
   else
      parameterchange:=0:
   end if:
   if successcounter=lengthofsuccesscontrol then
      successsum:=0:
      for a from 1 to lengthofsuccesscontrol do
         successsum:=successsum+successarray(a):
      end do:
      if successsum>=(lengthofsuccesscontrol/5) then
         scattermodification:=1:
         scatterfactor:=1/0.85:
      else
         scattermodification:=1:
         scatterfactor:=0.85:
      end if:

      newscatteretahaM:=scatteretahaM*scatterfactor:
      newscatteretahaV:=scatteretahaV*scatterfactor:
      newscattermuhaV:=scattermuhaV*scatterfactor:
      successcounter:=0:
      successarray:=Array(1..lengthofsuccesscontrol):
   end if:
   if r<=convergencersquare or numberofactualevolutioncycle=numberofmaximumevolutioncycles then
      evolution:=0:
```

```
   end if:
end do:
```

B.4 Model validation

The program described in this section calculates the relaxation forces on the basis of the experimental data according to [39] and [67]. In addition, the averaged standardized relative difference D is determined.

```
restart;with(plots):with(linalg):with(LinearAlgebra):with(MTM):

#definition of calculation parameters
#-------------------------------------

kapproxmax:=4:                              #approximation order of gaver-wynn-rho algorithm for laplace inversion
Digits:=round(kapproxmax*2.1)+8:            #calculation precision
tmax:=100000:                               #time instant (in seconds) at end of observed viscoelastic response)
numberofintervals:=14:                      #number of intervals -1
tolerance:=1*10^(-(round(kapproxmax*2.1)+1));


#definition of material properties         (in GPa and GPa.s)
#---------------------------------

ccol3333:=17.9:       ccol1111:=11.7:
ccol1133:=7.1:        ccol1122:=5.1:      ccol1313:=3.3:
muhainst:=44.9:       kha:=82.6:
muwater:=0:           kwater:=2.3:

etahaM:=1.284246403983642e7:
etahaV:=1.343032865581029e5:
muhaV:=179.5262919354894:


#determination of volume fractions
#---------------------------------

flac:=0.020:
fvas:=0.050:
rhomu:=1.99:
fmupor:=fvas+flac:
rhoult:=(rhomu-1*fmupor)/(1-fmupor);

barfha:=0.46:
barfcol:=0.29:

tildeflac:=0.021:
ds:=-0.2000*rhoult+1.6580:
b:=1.47:
Dcol:=64:
vcol:=335.6:
vfib:=b*ds*5*Dcol:
barffib:=barfcol*vfib/vcol:
barfef:=1-barffib:

phihaef:=(1-barffib)/(1-barfcol):
checkfha:=phihaef*barfha/barfef:
checkfic:=1-checkfha:

brevefha:=barfha*(1-phihaef)/barffib:
brevefwetcol:=1-brevefha:
```

```
circlefcol:=barfcol/brevefwetcol:
circlefuw:=1-circlefcol:

fim:=1-circlefcol;
circlefcol:=circlefcol;
fwetcol:=brevefwetcol;
fha:=checkfha;
ffib:=barffib;
flac:=flac;
fvas:=fvas;


#definition of volumetric and deviatoric matrix
#-----------------------------------------------

k:=1/3:
K:=Matrix([[k,k,k,0,0,0],[k,k,k,0,0,0],[k,k,k,0,0,0],[0,0,0,0,0,0],[0,0,0,0,0,0],[0,0,0,0,0,0]]):
J:=IdentityMatrix(6,6)-K:
I6:=IdentityMatrix(6,6):


#definition of Q, theta & phi for rotation
#-----------------------------------------

e1:=Vector([cos(phi)*cos(theta),sin(phi)*cos(theta),-sin(theta)]):
e2:=Vector([-sin(phi),cos(phi),0]):
e3:=Vector([cos(phi)*sin(theta),sin(phi)*sin(theta),cos(theta)]):

q11:=e1[1]:
q21:=e1[2]:
q31:=e1[3]:
q12:=e2[1]:
q22:=e2[2]:
q32:=e2[3]:
q13:=e3[1]:
q23:=e3[2]:
q33:=e3[3]:

a:=2/sqrt(2):
n:=sqrt(2):

Q:=Matrix(6):
Q[1,1]:=q11^2:
Q[1,2]:=q12^2:
Q[1,3]:=q13^2:
Q[1,4]:=a*q12*q13:
Q[1,5]:=a*q13*q11:
Q[1,6]:=a*q11*q12:
Q[2,1]:=q21^2:
Q[2,2]:=q22^2:
Q[2,3]:=q23^2:
Q[2,4]:=a*q22*q23:
Q[2,5]:=a*q23*q21:
Q[2,6]:=a*q21*q22:
Q[3,1]:=q31^2:
Q[3,2]:=q32^2:
Q[3,3]:=q33^2:
Q[3,4]:=a*q32*q33:
Q[3,5]:=a*q33*q31:
Q[3,6]:=a*q31*q32:
Q[4,1]:=n*q21*q31:
Q[4,2]:=n*q22*q32:
Q[4,3]:=n*q23*q33:
Q[4,4]:=q23*q32+q33*q22:
Q[4,5]:=q21*q33+q31*q23:
Q[4,6]:=q22*q31+q32*q21:
Q[5,1]:=n*q31*q11:
Q[5,2]:=n*q32*q12:
Q[5,3]:=n*q33*q13:
```

```
Q[5,4]:=q33*q12+q13*q32:
Q[5,5]:=q31*q13+q11*q33:
Q[5,6]:=q32*q11+q12*q31:
Q[6,1]:=n*q11*q21:
Q[6,2]:=n*q12*q22:
Q[6,3]:=n*q13*q23:
Q[6,4]:=q13*q22+q23*q12:
Q[6,5]:=q11*q23+q21*q13:
Q[6,6]:=q12*q21+q22*q11:

r:=1/2:
s:=(sqrt(5)+1)/4:
t:=(sqrt(5)-1)/4:

theta1:=evalf(arccos(t)):
phi1:=evalf(arccos(r/sin(theta1))):

theta2:=evalf(arccos(t)):
phi2:=evalf(arcsin(-s/sin(theta2))):

theta3:=evalf(arccos(t)):
phi3:=evalf(arccos(-r/sin(theta3))):

theta4:=evalf(arccos(t)):
phi4:=evalf(Pi-arcsin(-s/sin(theta4))):

theta5:=evalf(arccos(s)):
phi5:=evalf(arccos(t/sin(theta5))):

theta6:=evalf(arccos(s)):
phi6:=evalf(arcsin(-r/sin(theta6))):

theta7:=evalf(arccos(s)):
phi7:=evalf(arccos(-t/sin(theta7))):

theta8:=evalf(arccos(s)):
phi8:=evalf(Pi-arcsin(-r/sin(theta8))):

theta9:=evalf(arccos(r)):
phi9:=evalf(arccos(s/sin(theta9))):

theta10:=evalf(arccos(r)):
phi10:=evalf(arcsin(-t/sin(theta10))):

theta11:=evalf(arccos(r)):
phi11:=evalf(arccos(-s/sin(theta11))):

theta12:=evalf(arccos(r)):
phi12:=evalf(Pi-arcsin(-t/sin(theta12))):

theta13:=evalf(Pi/2):
phi13:=0:

theta14:=evalf(Pi/2):
phi14:=Pi/2:

theta15:=0:
phi15:=0:

Q1:=Matrix(6):
Q2:=Matrix(6):
Q3:=Matrix(6):
Q4:=Matrix(6):
Q5:=Matrix(6):
Q6:=Matrix(6):
Q7:=Matrix(6):
Q8:=Matrix(6):
Q9:=Matrix(6):
Q10:=Matrix(6):
```

```
Q11:=Matrix(6):
Q12:=Matrix(6):
Q13:=Matrix(6):
Q14:=Matrix(6):
Q15:=Matrix(6):

for i from 1 to 6 do
for j from 1 to 6 do
   Q1[i,j]:=eval(Q[i,j],[theta=theta1,phi=phi1]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q2[i,j]:=eval(Q[i,j],[theta=theta2,phi=phi2]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q3[i,j]:=eval(Q[i,j],[theta=theta3,phi=phi3]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q4[i,j]:=eval(Q[i,j],[theta=theta4,phi=phi4]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q5[i,j]:=eval(Q[i,j],[theta=theta5,phi=phi5]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q6[i,j]:=eval(Q[i,j],[theta=theta6,phi=phi6]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q7[i,j]:=eval(Q[i,j],[theta=theta7,phi=phi7]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q8[i,j]:=eval(Q[i,j],[theta=theta8,phi=phi8]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q9[i,j]:=eval(Q[i,j],[theta=theta9,phi=phi9]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q10[i,j]:=eval(Q[i,j],[theta=theta10,phi=phi10]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q11[i,j]:=eval(Q[i,j],[theta=theta11,phi=phi11]):
```

```
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q12[i,j]:=eval(Q[i,j],[theta=theta12,phi=phi12]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q13[i,j]:=eval(Q[i,j],[theta=theta13,phi=phi13]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q14[i,j]:=eval(Q[i,j],[theta=theta14,phi=phi14]):
end do:
end do:

for i from 1 to 6 do
for j from 1 to 6 do
   Q15[i,j]:=eval(Q[i,j],[theta=theta15,phi=phi15]):
end do:
end do:


#calculation of creep compliance
#-------------------------------

#wet collagen
#------------
Cwater:=3*kwater*K+2*muwater*J:
Cim:=Cwater:

Ccol:=Matrix(6):
Ccol[1,1]:=ccol1111:
Ccol[2,2]:=Ccol[1,1]:
Ccol[3,3]:=ccol3333:
Ccol[1,2]:=ccol1122:
Ccol[2,1]:=Ccol[1,2]:
Ccol[1,3]:=ccol1133:
Ccol[3,1]:=Ccol[1,3]:
Ccol[3,2]:=Ccol[3,1]:
Ccol[2,3]:=Ccol[3,2]:
Ccol[4,4]:=2*ccol1313:
Ccol[5,5]:=Ccol[4,4]:
Ccol[6,6]:=2*0.5*(Ccol[1,1]-Ccol[1,2]):

Pcylcol:=Matrix(6):
P1111:=1/8*(5*Ccol[1,1]-3*Ccol[1,2])/Ccol[1,1]/(Ccol[1,1]-Ccol[1,2]):
P1122:=-1/8*(Ccol[1,1]+Ccol[1,2])/Ccol[1,1]/(Ccol[1,1]-Ccol[1,2]):
P2222:=P1111:
P2323:=1/8*1/(0.5*Ccol[4,4]):
P1313:=P2323:
P1212:=1/8*(3*Ccol[1,1]-Ccol[1,2])/Ccol[1,1]/(Ccol[1,1]-Ccol[1,2]):
Pcylcol[1,1]:=P1111:
Pcylcol[2,2]:=P2222:
Pcylcol[1,2]:=P1122:
Pcylcol[2,1]:=Pcylcol[1,2]:
Pcylcol[4,4]:=2*P2323:
Pcylcol[5,5]:=2*P1313:
Pcylcol[6,6]:=2*P1212:

help_im:=inv(I6+Pcylcol.(Cim-Ccol)):
Cestwetcol:=((1-fim)*Ccol+fim*Cim.help_im).inv((1-fim)*I6+fim*help_im):

#fibril
```

```
#------
Cha:=3*kha*K+2*muhainst*J:

Cestfibnew:=Cestwetcol:
Cestfib:=(1+tolerance*10)*Cestfibnew:

while abs(Cestfib[1,1]-Cestfibnew[1,1])>tolerance do
   Cestfib:=Cestfibnew:

   Pcylfib:=Matrix(6):
   P1111:=1/8*(5*Cestfib[1,1]-3*Cestfib[1,2])/Cestfib[1,1]/(Cestfib[1,1]-Cestfib[1,2]):
   P1122:=-1/8*(Cestfib[1,1]+Cestfib[1,2])/Cestfib[1,1]/(Cestfib[1,1]-Cestfib[1,2]):
   P2222:=P1111:
   P2323:=1/8*1/(0.5*Cestfib[4,4]):
   P1313:=P2323:
   P1212:=1/8*(3*Cestfib[1,1]-Cestfib[1,2])/Cestfib[1,1]/(Cestfib[1,1]-Cestfib[1,2]):
   Pcylfib[1,1]:=P1111:
   Pcylfib[2,2]:=P2222:
   Pcylfib[1,2]:=P1122:
   Pcylfib[2,1]:=Pcylfib[1,2]:
   Pcylfib[4,4]:=2*P2323:
   Pcylfib[5,5]:=2*P1313:
   Pcylfib[6,6]:=2*P1212:

   C1111:=Cestfib[1,1]:
   C1122:=Cestfib[1,2]:
   C1133:=Cestfib[1,3]:
   C3333:=Cestfib[3,3]:
   C2323:=Cestfib[4,4]/2:

   Psphfib:=Matrix(6):

   D1:=(-2)*C1111^2*x^4*C3333+2*C2323^2*x^6*C3333-4*C1111*C2323^2*x^4-
          3*C1111^2*C2323*x^2+C1111^2*x^2*C3333+
          2*C1111*C2323^2*x^2-2*C2323*x^4*C1133^2-C1111*C1133^2*x^6+
          2*C1111*C1133^2*x^4+4*C2323^2*x^6*C1133-
          2*C1122*C1133^2*x^4+2*C2323*x^6*C1133^2+3*C1111^2*x^4*C2323+
            C1122*C1133^2*x^6-C1111^2*x^6*C2323+
          2*C1111*x^6*C2323^2+C1111^2*x^6*C3333-C1111*C1133^2*x^2-
          4*C2323^2*x^4*C1133+C1122*C1133^2*x^2+
            C1111^2*C2323-C1122*C1111*C2323-C1122*x^6*C1111*C3333+
          4*C1111*x^4*C2323*C1133-2*C1111*x^2*C2323*C1133-
          4*C1122*x^4*C2323*C1133+2*C1122*x^2*C2323*C1133+
          2*C1122*x^6*C2323*C1133-2*C1111*x^6*C2323*C1133-
          3*C1111*x^6*C2323*C3333+2*C1122*C1111*x^4*C3333-
            C1122*C2323*x^4*C3333-3*C1122*C1111*x^4*C2323-
            C1122*C1111*x^2*C3333+3*C1122*C1111*C2323*x^2+
          3*C1111*C2323*x^4*C3333+C1122*x^6*C1111*C2323+
            C1122*x^6*C2323*C3333:

   D2:=2*C2323*x^4*C1133+C2323*x^4*C3333+C1111*x^4*C2323-
       2*C2323*x^2*C1133-2*C1111*C2323*x^2+
         C1111*C2323+x^4*C1133^2-C1111*x^4*C3333-
         x^2*C1133^2+C1111*x^2*C3333:


   P1111int:=-((-5)*C1111*x^4*C3333-3*C1122*x^2*C3333-
                  3*C1122*x^4*C2323+
                  3*C1122*x^4*C3333+5*C1111*x^4*C2323-
                 10*C1111*C2323*x^2+2*x^4*C1133^2+
                  8*C2323*x^4*C3333-6*C2323^2*x^4+
                  4*C2323*x^4*C1133+6*C1122*C2323*x^2+
                  5*C1111*C2323+5*C1111*x^2*C3333-
                  4*C2323*x^2*C1133+6*C2323^2*x^2-
                  2*x^2*C1133^2-3*C1122*C2323)*(-1+x^2)/D1:

   P1122int:=(C1111*C2323-2*C1111*C2323*x^2+
              C1111*x^2*C3333+C1122*C2323-
            2*C1122*C2323*x^2+C1122*x^2*C3333+
```

```
              C1111*x^4*C2323-C1111*x^4*C3333+
              C1122*x^4*C2323-C1122*x^4*C3333-2*C2323^2*x^2+
            2*C2323^2*x^4-4*C2323*x^2*C1133+
            4*C2323*x^4*C1133-2*x^2*C1133^2+
            2*x^4*C1133^2)*(-1+x^2)/D1:

   P1133int:=(-1+x^2)*x^2*(C2323+C1133)/D2:

   P2323int:=(4*C1111*C2323*x^2-8*C2323*x^4*C1133-
              2*x^4*C1133^2-C1122*x^4*C3333-
              8*C1111*x^4*C2323+3*C1111*x^4*C3333+
              4*C1111*x^4*C1133-4*C1122*x^4*C1133+
              2*C1122*x^6*C1133-2*C1111*x^6*C1133+
                C1122*x^6*C1111-3*C1122*x^4*C1111+
              3*C1122*C1111*x^2-2*C1111*x^2*C1133+
              2*C1122*x^2*C1133+8*x^6*C2323*C1133-
              3*x^6*C1111*C3333+4*x^6*C2323*C3333+
              4*C1111*x^6*C2323+C1122*x^6*C3333+
              3*C1111^2*x^4-C1111^2*x^6+
              2*C1133^2*x^6-3*C1111^2*x^2+
                C1111^2-C1122*C1111)/D1:

   P3333int:=x^2*(x^2*C2323-C1111*x^2+
             C1111)/D2:

   P1111:=evalf(1/16*int(P1111int,x=-1..1)):
   P1122:=evalf(1/16*int(P1122int,x=-1..1)):
   P1133:=evalf(1/4*int(P1133int,x=-1..1)):
   P2323:=evalf(1/16*int(P2323int,x=-1..1)):
   P3333:=evalf(1/2*int(P3333int,x=-1..1)):

   Psphfib[1,1]:=P1111:
   Psphfib[2,2]:=Psphfib[1,1]:
   Psphfib[3,3]:=P3333:
   Psphfib[1,2]:=P1122:
   Psphfib[1,3]:=P1133:
   Psphfib[2,3]:=Psphfib[1,3]:
   Psphfib[2,1]:=Psphfib[1,2]:
   Psphfib[3,1]:=Psphfib[1,3]:
   Psphfib[3,2]:=Psphfib[2,3]:
   Psphfib[4,4]:=2*P2323:
   Psphfib[5,5]:=Psphfib[4,4]:
   Psphfib[6,6]:=2*0.5*(P1111-P1122):
   help_wetcol:=inv(I6+Pcylfib.(Cestwetcol-Cestfib)):
   help_ha:=inv(I6+Psphfib.(Cha-Cestfib)):
   Cestfibnew:=(fwetcol*Cestwetcol.help_wetcol+(1-fwetcol)*Cha.help_ha).
               inv(fwetcol*help_wetcol+(1-fwetcol)*help_ha):

end do:
Cestfib:=Cestfibnew:

#ultrastructure
#---------------
Pcylef:=Matrix(6):
P1111:=1/8*(5*Cestefsym[1,1]-3*Cestefsym[1,2])/Cestefsym[1,1]/(Cestefsym[1,1]-Cestefsym[1,2]):
P1122:=-1/8*(Cestefsym[1,1]+Cestefsym[1,2])/Cestefsym[1,1]/(Cestefsym[1,1]-Cestefsym[1,2]):
P2222:=P1111:
P2323:=1/8*1/(0.5*Cestefsym[4,4]):
P1313:=P2323:
P1212:=1/8*(3*Cestefsym[1,1]-Cestefsym[1,2])/Cestefsym[1,1]/(Cestefsym[1,1]-Cestefsym[1,2]):
Pcylef[1,1]:=P1111:
Pcylef[2,2]:=P2222:
Pcylef[1,2]:=P1122:
Pcylef[2,1]:=Pcylef[1,2]:
Pcylef[4,4]:=2*P2323:
Pcylef[5,5]:=2*P1313:
Pcylef[6,6]:=2*P1212:

help_fib:=inv(I6+Pcylef.(Cestfib-Cestefsym)):
```

```
Cestult:=((1-ffib)*Cestefsym+ffib*Cestfib.help_fib).inv((1-ffib)*I6+ffib*help_fib):

#extravascular bone material
#---------------------------
Clac:=Matrix(6):

C1111:=Cestultsym[1,1]:
C1122:=Cestultsym[1,2]:
C1133:=Cestultsym[1,3]:
C3333:=Cestultsym[3,3]:
C2323:=Cestultsym[4,4]/2:

Psphult:=Matrix(6):

D1:=(-2)*C1111^2*x^4*C3333+2*C2323^2*x^6*C3333-4*C1111*C2323^2*x^4-
       3*C1111^2*C2323*x^2+C1111^2*x^2*C3333+
       2*C1111*C2323^2*x^2-2*C2323*x^4*C1133^2-C1111*C1133^2*x^6+
       2*C1111*C1133^2*x^4+4*C2323^2*x^6*C1133-
       2*C1122*C1133^2*x^4+2*C2323*x^6*C1133^2+3*C1111^2*x^4*C2323+
         C1122*C1133^2*x^6-C1111^2*x^6*C2323+
       2*C1111*x^6*C2323^2+C1111^2*x^6*C3333-C1111*C1133^2*x^2-
       4*C2323^2*x^4*C1133+C1122*C1133^2*x^2+
         C1111^2*C2323-C1122*C1111*C2323-C1122*x^6*C1111*C3333+
       4*C1111*x^4*C2323*C1133-2*C1111*x^2*C2323*C1133-
       4*C1122*x^4*C2323*C1133+2*C1122*x^2*C2323*C1133+
       2*C1122*x^6*C2323*C1133-2*C1111*x^6*C2323*C1133-
       3*C1111*x^6*C2323*C3333+2*C1122*C1111*x^4*C3333-
         C1122*C2323*x^4*C3333-3*C1122*C1111*x^4*C2323-
         C1122*C1111*x^2*C3333+3*C1122*C1111*C2323*x^2+
       3*C1111*C2323*x^4*C3333+C1122*x^6*C1111*C2323+
         C1122*x^6*C2323*C3333:

D2:=2*C2323*x^4*C1133+C2323*x^4*C3333+C1111*x^4*C2323-
    2*C2323*x^2*C1133-2*C1111*C2323*x^2+
      C1111*C2323+x^4*C1133^2-C1111*x^4*C3333-
      x^2*C1133^2+C1111*x^2*C3333:


P1111int:=-((-5)*C1111*x^4*C3333-3*C1122*x^2*C3333-
               3*C1122*x^4*C2323+
               3*C1122*x^4*C3333+5*C1111*x^4*C2323-
              10*C1111*C2323*x^2+2*x^4*C1133^2+
               8*C2323*x^4*C3333-6*C2323^2*x^4+
               4*C2323*x^4*C1133+6*C1122*C2323*x^2+
               5*C1111*C2323+5*C1111*x^2*C3333-
               4*C2323*x^2*C1133+6*C2323^2*x^2-
               2*x^2*C1133^2-3*C1122*C2323)*(-1+x^2)/D1:

P1122int:=(C1111*C2323-2*C1111*C2323*x^2+
            C1111*x^2*C3333+C1122*C2323-
          2*C1122*C2323*x^2+C1122*x^2*C3333+
            C1111*x^4*C2323-C1111*x^4*C3333+
            C1122*x^4*C2323-C1122*x^4*C3333-2*C2323^2*x^2+
           2*C2323^2*x^4-4*C2323*x^2*C1133+
           4*C2323*x^4*C1133-2*x^2*C1133^2+
          2*x^4*C1133^2)*(-1+x^2)/D1:

P1133int:=(-1+x^2)*x^2*(C2323+C1133)/D2:

P2323int:=(4*C1111*C2323*x^2-8*C2323*x^4*C1133-
            2*x^4*C1133^2-C1122*x^4*C3333-
            8*C1111*x^4*C2323+3*C1111*x^4*C3333+
            4*C1111*x^4*C1133-4*C1122*x^4*C1133+
            2*C1122*x^6*C1133-2*C1111*x^6*C1133+
              C1122*x^6*C1111-3*C1122*x^4*C1111+
            3*C1122*C1111*x^2-2*C1111*x^2*C1133+
            2*C1122*x^2*C1133+8*x^6*C2323*C1133-
            3*x^6*C1111*C3333+4*x^6*C2323*C3333+
            4*C1111*x^6*C2323+C1122*x^6*C3333+
```

```
           3*C1111^2*x^4-C1111^2*x^6+
           2*C1133^2*x^6-3*C1111^2*x^2+
             C1111^2-C1122*C1111)/D1:

P3333int:=x^2*(x^2*C2323-C1111*x^2+
          C1111)/D2:

P1111:=evalf(1/16*int(P1111int,x=-1..1)):
P1122:=evalf(1/16*int(P1122int,x=-1..1)):
P1133:=evalf(1/4*int(P1133int,x=-1..1)):
P2323:=evalf(1/16*int(P2323int,x=-1..1)):
P3333:=evalf(1/2*int(P3333int,x=-1..1)):

Psphult[1,1]:=P1111:
Psphult[2,2]:=Psphult[1,1]:
Psphult[3,3]:=P3333:
Psphult[1,2]:=P1122:
Psphult[1,3]:=P1133:
Psphult[2,3]:=Psphult[1,3]:
Psphult[2,1]:=Psphult[1,2]:
Psphult[3,1]:=Psphult[1,3]:
Psphult[3,2]:=Psphult[2,3]:
Psphult[4,4]:=2*P2323:
Psphult[5,5]:=Psphult[4,4]:
Psphult[6,6]:=2*0.5*(P1111-P1122):

help_lac:=inv(I6+Psphult.(Clac-Cestultsym)):
Cestexvas:=((1-flac)*Cestultsym+flac*Clac.help_lac).inv((1-flac)*I6+flac*help_lac):

#bone microstructure
#-------------------
Cvas:=Matrix(6):

Pcylexvas:=Matrix(6):
P1111:=1/8*(5*Cestexvassym[1,1]-3*Cestexvassym[1,2])/Cestexvassym[1,1]/(Cestexvassym[1,1]-Cestexvassym[1,2]):
P1122:=-1/8*(Cestexvassym[1,1]+Cestexvassym[1,2])/Cestexvassym[1,1]/(Cestexvassym[1,1]-Cestexvassym[1,2]):
P2222:=P1111:
P2323:=1/8*1/(0.5*Cestexvassym[4,4]):
P1313:=P2323:
P1212:=1/8*(3*Cestexvassym[1,1]-Cestexvassym[1,2])/Cestexvassym[1,1]/(Cestexvassym[1,1]-Cestexvassym[1,2]):
Pcylexvas[1,1]:=P1111:
Pcylexvas[2,2]:=P2222:
Pcylexvas[1,2]:=P1122:
Pcylexvas[2,1]:=Pcylexvas[1,2]:
Pcylexvas[4,4]:=2*P2323:
Pcylexvas[5,5]:=2*P1313:
Pcylexvas[6,6]:=2*P1212:

help_vas:=inv(I6+Pcylexvas.(Cvas-Cestexvassym)):
Cestmicro:=((1-fvas)*Cestexvassym+fvas*Cvas.help_vas).inv((1-fvas)*I6+fvas*help_vas):


#calculation of creep compliance for extrafibrilliar space in LC-space
#---------------------------------------------------------------------

t:=array(1..15):
t[1]:=1:
t[2]:=7.29568354608:
t[3]:=31.8387278704:
t[4]:=108.763941305:
t[5]:=321.683062152:
t[6]:=852.851887276:
t[7]:=2030.99832312:
t[8]:=4323.3497897:
t[9]:=8269.20760983:
t[10]:=14428.410878:
t[11]:=23376.4603501:
t[12]:=35720.7020611:
t[13]:=52116.7122724:
```

```
t[14]:=73280.7764716:
t[15]:=100000:

#extrafibrilliar space
#---------------------
for h from 1 to kapproxmax do
   for i from 1 to numberofintervals+1 do
      for j from 1  to (h+1) do

         p[i,j]:=evalf((h+(j-1))*log(2)/t[i]):
         muha[i,j]:=1/(1/muhainst+1/(p[i,j]*etahaM)+1/(p[i,j]*etahaV+muhaV)):

         Cwater_GLOBAL:=3*kwater*K+2*muwater*J:
         Cha_GLOBAL:=3*kha*K+2*muha[i,j]*J:

         tolerance:=1*10^(-round(kapproxmax*2.1)+5):
         Cestnew:=3*kha*K+2*muhainst*J:
         Cest:=(1+tolerance*10)*Cestnew:

         while abs(Cest[1,1]-Cestnew[1,1])>tolerance do
            Cest:=Cestnew:
            Dest:=inv(Cest):
            muestef:=1/(2*Dest[4,4]):
            Eestef:=1/Dest[1,1]:
            kestef:=(muestef*Eestef)/(3*(3*muestef-Eestef)):

            alphaef:=3*kestef/(3*kestef+4*muestef):
            betaef:=6*(kestef+2*muestef)/(5*(3*kestef+4*muestef)):
            Ssphef:=alphaef*K+betaef*J:
            Psphef:=Ssphef.inv(Cest):

            nup:=(3*kestef-2*muestef)/(6*kestef+2*muestef):
            Scylef:=Matrix(6):
            Scylef[1,1]:=(5-4*nup)/(8*(1-nup)):
            Scylef[2,2]:=Scylef[1,1]:
            Scylef[1,2]:=(-1+4*nup)/(8*(1-nup)):
            Scylef[2,1]:=Scylef[1,2]:
            Scylef[1,3]:=nup/(2*(1-nup)):
            Scylef[2,3]:=Scylef[1,3]:
            Scylef[4,4]:=2*(1/4):
            Scylef[5,5]:=Scylef[4,4]:
            Scylef[6,6]:=2*((3-4*nup)/(8*(1-nup))):
            Pcyl_LOCAL:=Scylef.inv(Cest):

            Pcyl1:=Q1.Pcyl_LOCAL.transpose(Q1):
            Pcyl2:=Q2.Pcyl_LOCAL.transpose(Q2):
            Pcyl3:=Q3.Pcyl_LOCAL.transpose(Q3):
            Pcyl4:=Q4.Pcyl_LOCAL.transpose(Q4):
            Pcyl5:=Q5.Pcyl_LOCAL.transpose(Q5):
            Pcyl6:=Q6.Pcyl_LOCAL.transpose(Q6):
            Pcyl7:=Q7.Pcyl_LOCAL.transpose(Q7):
            Pcyl8:=Q8.Pcyl_LOCAL.transpose(Q8):
            Pcyl9:=Q9.Pcyl_LOCAL.transpose(Q9):
            Pcyl10:=Q10.Pcyl_LOCAL.transpose(Q10):
            Pcyl11:=Q11.Pcyl_LOCAL.transpose(Q11):
            Pcyl12:=Q12.Pcyl_LOCAL.transpose(Q12):
            Pcyl13:=Q13.Pcyl_LOCAL.transpose(Q13):
            Pcyl14:=Q14.Pcyl_LOCAL.transpose(Q14):
            Pcyl15:=Q15.Pcyl_LOCAL.transpose(Q15):

            help_ha:=1/15*inv(I6+Pcyl1.(Cha_GLOBAL-Cest))+1/15*inv(I6+Pcyl2.(Cha_GLOBAL-Cest))+
            1/15*inv(I6+Pcyl3.(Cha_GLOBAL-Cest))+1/15*inv(I6+Pcyl4.(Cha_GLOBAL-Cest))+
            1/15*inv(I6+Pcyl5.(Cha_GLOBAL-Cest))+1/15*inv(I6+Pcyl6.(Cha_GLOBAL-Cest))+
            1/15*inv(I6+Pcyl7.(Cha_GLOBAL-Cest))+1/15*inv(I6+Pcyl8.(Cha_GLOBAL-Cest))+
            1/15*inv(I6+Pcyl9.(Cha_GLOBAL-Cest))+1/15*inv(I6+Pcyl10.(Cha_GLOBAL-Cest))+
            1/15*inv(I6+Pcyl11.(Cha_GLOBAL-Cest))+1/15*inv(I6+Pcyl12.(Cha_GLOBAL-Cest))+
            1/15*inv(I6+Pcyl13.(Cha_GLOBAL-Cest))+1/15*inv(I6+Pcyl14.(Cha_GLOBAL-Cest))+
            1/15*inv(I6+Pcyl15.(Cha_GLOBAL-Cest)):
            help_pores:=inv(I6+Psphef.(Cwater_GLOBAL-Cest)):
```

```
            Cestnew:=(fha*Cha_GLOBAL.help_ha+(1-fha)*Cwater_GLOBAL.help_pores).inv(fha*help_ha+(1-fha)*help_pores):

         end do:

         Cesteflc:=Cestnew:

         Cestultlc:=Matrix(6):
         Cestexvaslc:=Matrix(6):
         Cestmicrolc:=Matrix(6):

         for k from 1 to 6 do
         for l from 1 to 6 do
            Cestultlc[k,l]:=eval(Cestult[k,l],[Cestefsym=Cesteflc]):
         end do:
         end do:

         for k from 1 to 6 do
         for l from 1 to 6 do
            Cestexvaslc[k,l]:=eval(Cestexvas[k,l],[Cestultsym=Cestultlc]):
         end do:
         end do:

         for k from 1 to 6 do
         for l from 1 to 6 do
            Cestmicrolc[k,l]:=eval(Cestmicro[k,l],[Cestexvassym=Cestexvaslc]):
         end do:
         end do:

         R1111[i,j]:=Cestmicrolc[1,1]:
         R3333[i,j]:=Cestmicrolc[3,3]:
         R1122[i,j]:=Cestmicrolc[1,2]:
         R1133[i,j]:=Cestmicrolc[1,3]:
         R2323[i,j]:=Cestmicrolc[4,4]/2:

         Destmicrolc:=inv(Cestmicrolc):
         J1111[i,j]:=Destmicrolc[1,1]:
         J3333[i,j]:=Destmicrolc[3,3]:
         J1122[i,j]:=Destmicrolc[1,2]:
         J1133[i,j]:=Destmicrolc[1,3]:
         J2323[i,j]:=Destmicrolc[4,4]/2:

      end do:
   end do:


#calculation of GAVER-functionals
#--------------------------------

   for i from 1 to (numberofintervals+1) do
      R1111funcK[i]:=0:
      R3333funcK[i]:=0:
      R1122funcK[i]:=0:
      R1133funcK[i]:=0:
      R2323funcK[i]:=0:

      J1111funcK[i]:=0:
      J3333funcK[i]:=0:
      J1122funcK[i]:=0:
      J1133funcK[i]:=0:
      J2323funcK[i]:=0:

      for j from 0 to h do
         indexj:=j+1:
         R1111funcK[i]:=R1111funcK[i]+log(2)*h/t[i]*binomial(2*h,h)*(-1)^j*binomial(h,j)*R1111[i,indexj]/p[i,indexj]:
         R3333funcK[i]:=R3333funcK[i]+log(2)*h/t[i]*binomial(2*h,h)*(-1)^j*binomial(h,j)*R3333[i,indexj]/p[i,indexj]:
         R1122funcK[i]:=R1122funcK[i]+log(2)*h/t[i]*binomial(2*h,h)*(-1)^j*binomial(h,j)*R1122[i,indexj]/p[i,indexj]:
         R1133funcK[i]:=R1133funcK[i]+log(2)*h/t[i]*binomial(2*h,h)*(-1)^j*binomial(h,j)*R1133[i,indexj]/p[i,indexj]:
         R2323funcK[i]:=R2323funcK[i]+log(2)*h/t[i]*binomial(2*h,h)*(-1)^j*binomial(h,j)*R2323[i,indexj]/p[i,indexj]:
         J1111funcK[i]:=J1111funcK[i]+log(2)*h/t[i]*binomial(2*h,h)*(-1)^j*binomial(h,j)*J1111[i,indexj]/p[i,indexj]:
```

```
         J3333funcK[i]:=J3333funcK[i]+log(2)*h/t[i]*binomial(2*h,h)*(-1)^j*binomial(h,j)*J3333[i,indexj]/p[i,indexj]:
         J1122funcK[i]:=J1122funcK[i]+log(2)*h/t[i]*binomial(2*h,h)*(-1)^j*binomial(h,j)*J1122[i,indexj]/p[i,indexj]:
         J1133funcK[i]:=J1133funcK[i]+log(2)*h/t[i]*binomial(2*h,h)*(-1)^j*binomial(h,j)*J1133[i,indexj]/p[i,indexj]:
         J2323funcK[i]:=J2323funcK[i]+log(2)*h/t[i]*binomial(2*h,h)*(-1)^j*binomial(h,j)*J2323[i,indexj]/p[i,indexj]:
      end do:
   end do:
   for i from 1 to (numberofintervals+1) do
      R1111gaverfun[h,i]:=evalf(R1111funcK[i]):
      R3333gaverfun[h,i]:=evalf(R3333funcK[i]):
      R1122gaverfun[h,i]:=evalf(R1122funcK[i]):
      R1133gaverfun[h,i]:=evalf(R1133funcK[i]):
      R2323gaverfun[h,i]:=evalf(R2323funcK[i]):
      J1111gaverfun[h,i]:=evalf(J1111funcK[i]):
      J3333gaverfun[h,i]:=evalf(J3333funcK[i]):
      J1122gaverfun[h,i]:=evalf(J1122funcK[i]):
      J1133gaverfun[h,i]:=evalf(J1133funcK[i]):
      J2323gaverfun[h,i]:=evalf(J2323funcK[i]):
   end do:
end do:


#inversion of LC-transforms through WYNN-RHO-acceleration
#-------------------------------------------------------

R1111gwr:=array(1..numberofintervals+1):
R3333gwr:=array(1..numberofintervals+1):
R1122gwr:=array(1..numberofintervals+1):
R1133gwr:=array(1..numberofintervals+1):
R2323gwr:=array(1..numberofintervals+1):

J1111gwr:=array(1..numberofintervals+1):
J3333gwr:=array(1..numberofintervals+1):
J1122gwr:=array(1..numberofintervals+1):
J1133gwr:=array(1..numberofintervals+1):
J2323gwr:=array(1..numberofintervals+1):

for h from 1 to (numberofintervals+1) do
   for i from 1 to (kapproxmax+2) do
      for j from 1 to (kapproxmax+2) do
         if i=1 then
            rhoR1111[i,j]:=0:
            rhoR3333[i,j]:=0:
            rhoR1122[i,j]:=0:
            rhoR1133[i,j]:=0:
            rhoR2323[i,j]:=0:

            rhoJ1111[i,j]:=0:
            rhoJ3333[i,j]:=0:
            rhoJ1122[i,j]:=0:
            rhoJ1133[i,j]:=0:
            rhoJ2323[i,j]:=0:
         elif i=2 then
            if j=1 then
               rhoR1111[i,j]:=0:
               rhoR3333[i,j]:=0:
               rhoR1122[i,j]:=0:
               rhoR1133[i,j]:=0:
               rhoR2323[i,j]:=0:

               rhoJ1111[i,j]:=0:
               rhoJ3333[i,j]:=0:
               rhoJ1122[i,j]:=0:
               rhoJ1133[i,j]:=0:
               rhoJ2323[i,j]:=0:
            elif j>1 and j<(kapproxmax+2) then
               rhoR1111[i,j]:=R1111gaverfun[j-1,h]:
               rhoR3333[i,j]:=R3333gaverfun[j-1,h]:
               rhoR1122[i,j]:=R1122gaverfun[j-1,h]:
               rhoR1133[i,j]:=R1133gaverfun[j-1,h]:
```

```
            rhoR2323[i,j]:=R2323gaverfun[j-1,h]:

            rhoJ1111[i,j]:=J1111gaverfun[j-1,h]:
            rhoJ3333[i,j]:=J3333gaverfun[j-1,h]:
            rhoJ1122[i,j]:=J1122gaverfun[j-1,h]:
            rhoJ1133[i,j]:=J1133gaverfun[j-1,h]:
            rhoJ2323[i,j]:=J2323gaverfun[j-1,h]:
         end if:
      else
         if j<(kapproxmax+2-(i-2)) then
            rhoR1111[i,j]:=rhoR1111[i-2,j+1]+(i-2)/(rhoR1111[i-1,j+1]-rhoR1111[i-1,j]):
            rhoR3333[i,j]:=rhoR3333[i-2,j+1]+(i-2)/(rhoR3333[i-1,j+1]-rhoR3333[i-1,j]):
            rhoR1122[i,j]:=rhoR1122[i-2,j+1]+(i-2)/(rhoR1122[i-1,j+1]-rhoR1122[i-1,j]):
            rhoR1133[i,j]:=rhoR1133[i-2,j+1]+(i-2)/(rhoR1133[i-1,j+1]-rhoR1133[i-1,j]):
            rhoR2323[i,j]:=rhoR2323[i-2,j+1]+(i-2)/(rhoR2323[i-1,j+1]-rhoR2323[i-1,j]):

            rhoJ1111[i,j]:=rhoJ1111[i-2,j+1]+(i-2)/(rhoJ1111[i-1,j+1]-rhoJ1111[i-1,j]):
            rhoJ3333[i,j]:=rhoJ3333[i-2,j+1]+(i-2)/(rhoJ3333[i-1,j+1]-rhoJ3333[i-1,j]):
            rhoJ1122[i,j]:=rhoJ1122[i-2,j+1]+(i-2)/(rhoJ1122[i-1,j+1]-rhoJ1122[i-1,j]):
            rhoJ1133[i,j]:=rhoJ1133[i-2,j+1]+(i-2)/(rhoJ1133[i-1,j+1]-rhoJ1133[i-1,j]):
            rhoJ2323[i,j]:=rhoJ2323[i-2,j+1]+(i-2)/(rhoJ2323[i-1,j+1]-rhoJ2323[i-1,j]):
         else
            rhoR1111[i,j]:=0:
            rhoR3333[i,j]:=0:
            rhoR1122[i,j]:=0:
            rhoR1133[i,j]:=0:
            rhoR2323[i,j]:=0:

            rhoJ1111[i,j]:=0:
            rhoJ3333[i,j]:=0:
            rhoJ1122[i,j]:=0:
            rhoJ1133[i,j]:=0:
            rhoJ2323[i,j]:=0:
         end if:
      end if:
   end do:
   R1111gwr[h]:=evalf(rhoR1111[kapproxmax+2,1]):
   R3333gwr[h]:=evalf(rhoR3333[kapproxmax+2,1]):
   R1122gwr[h]:=evalf(rhoR1122[kapproxmax+2,1]):
   R1133gwr[h]:=evalf(rhoR1133[kapproxmax+2,1]):
   R2323gwr[h]:=evalf(rhoR2323[kapproxmax+2,1]):

   J1111gwr[h]:=evalf(rhoJ1111[kapproxmax+2,1]):
   J3333gwr[h]:=evalf(rhoJ3333[kapproxmax+2,1]):
   J1122gwr[h]:=evalf(rhoJ1122[kapproxmax+2,1]):
   J1133gwr[h]:=evalf(rhoJ1133[kapproxmax+2,1]):
   J2323gwr[h]:=evalf(rhoJ2323[kapproxmax+2,1]):
end do:


#IYO-longitudinal (see [1])
#----------------

#definition of elastic strain state
#----------------------------------
epsilon:= Vector(5, {(1) = -0.1525079723e-2, (2) = 0.2427466045e-3, (3) = 0.4854932089e-3, (4) = 0.8772152822e-3,
(5) = 0.1268937355e-2});

ho:=0.4762991920:                  #in mm
hy:=0.1516248755:                  #in mm
hu:=0.5237008080:                  #in mm

#determination of stress state and moment
#----------------------------------------
l:=32:                             #in mm
b:=5:                              #in mm
hz:=Vector([-ho,hy/2,hy,hy+(hu-hy)/2,hu]):
```

```
s:=array(1..5,1..numberofintervals+1):
F:=array(1..numberofintervals+1):
M:=array(1..numberofintervals+1):

for i from 1 to 5 do
   for j from 1 to numberofintervals+1 do
      s[i,j]:=1/J3333gwr[j]*epsilon[i]:
   end do:
end do:

for i from 1 to numberofintervals+1 do
   k1:=(s[3,i]-s[2,i])/(hz[3]-hz[2]);
   k2:=(s[5,i]-s[4,i])/(hz[5]-hz[4]);

   sy1:=k1*hz[3];
   sy2:=k2*hz[3]+d;
   d1:=solve(sy1=sy2,d);

   s1:=k1*z;
   s2:=k2*z+d1;
   M[i]:=int(s1*z*b,z=hz[1]..hz[3])+int(s2*z*b,z=hz[3]..hz[5]);

   F[i]:=4*M[i]/l;
end do:

#input of experimental data
#--------------------------
Iy:=0.42:                                          #data from [1]
epsilon_u:=0.0023:
lexp:=32:
hexp:=1:
bexp:=5:
E0:=14.2:
A1:=0.08:
t1:=49:
t2:=9.3e6:
B:=0.28:
G:=0.35:

Rexp:=array(1..numberofintervals+1):
Fexp:=array(1..numberofintervals+1):
Mexp:=array(1..numberofintervals+1):

texp:=array(1..15):
texp[1]:=1:
texp[2]:=7.29568354608:
texp[3]:=31.8387278704:
texp[4]:=108.763941305:
texp[5]:=321.683062152:
texp[6]:=852.851887276:
texp[7]:=2030.99832312:
texp[8]:=4323.3497897:
texp[9]:=8269.20760983:
texp[10]:=14428.410878:
texp[11]:=23376.4603501:
texp[12]:=35720.7020611:
texp[13]:=52116.7122724:
texp[14]:=73280.7764716:
texp[15]:=100000:

for i from 1 to numberofintervals+1 do
   Rexp[i]:=E0*(A1*exp(-(texp[i]/t1)^B)+(1-A1)*exp(-(texp[i]/t2)^G)):
end do:

for i from 1 to numberofintervals+1 do
   s1:=Rexp[i]*epsilon_u:
   k:=s1/(hexp/2):
   sel:=k*z:
   Mexp[i]:=int(sel*z*bexp,z=-hexp/2..hexp/2):
```

```
   Fexp[i]:=(4*Mexp[i])/lexp:
end do:

#D
#--
   expmean:=0:
   r:=0:
   rsquarecounter:=0:

   for i from 1 to numberofintervals+1 do
      rsquarecounter:=rsquarecounter+1:
      expvalues[rsquarecounter]:=Fexp[i]:
      modelpredictions[rsquarecounter]:=F[i]:
   end do:

   for i from 1 to rsquarecounter do
      expmean:=expmean+expvalues[i]/rsquarecounter:
   end do:

   sse:=array(1..rsquarecounter):
   sst:=array(1..rsquarecounter):

   for i from 1 to rsquarecounter do
      sst[i]:=(expvalues[i]):
      sse[i]:=(modelpredictions[i]-expvalues[i]):
   end do:
   for i from 1 to rsquarecounter do
      r:=r+sqrt((sse[i]/sst[i])^2):
   end do:

   D:=r/(numberofintervals+1);

#output of data
#--------------
writedata("t_IYO_long.dat",t):
writedata("F_IYO_long.dat",F);
writedata("texp_IYO_long.dat",texp);
writedata("Fexp_IYO_long.dat",Fexp);


#IYO-transversal, elastic (see [1])
#----------------------------------

#definition of elastic strain state
#----------------------------------
epsilon:= Vector(4, {(1) = -0.2152422080e-2, (2) = -0.1076211040e-2, (3) = 0.1076211040e-2, (4) = 0.2152422080e-2});

#determination of stress state and moment
#----------------------------------------

hel:=0.5:                          #in mm
l:=32:                             #in mm
b:=5:                              #in mm
hz:=Vector([-hel,-hel/2,hel/2,hel]):

s:=array(1..4,1..numberofintervals+1):
F:=array(1..numberofintervals+1):
M:=array(1..numberofintervals+1):

for i from 1 to 4 do
   for j from 1 to numberofintervals+1 do
      s[i,j]:=1/J1111gwr[j]*epsilon[i]:
   end do:
end do:

for i from 1 to numberofintervals+1 do
   k:=(s[3,i]-s[2,i])/(hz[3]-hz[2]);

   s1:=k*z;
```

```
   M[i]:=int(s1*z*b,z=hz[1]..hz[4]);

   F[i]:=4*M[i]/l;
end do:

#input of experimental data
#---------------------------
Iy:=0.42:                                          # data from [1]
epsilon_u:=0.0023:
lexp:=32:
hexp:=1:
bexp:=5:
E0:=11.6:
A1:=0.11:
t1:=50:
t2:=6.4e6:
B:=0.26:
G:=0.37:
Rexp:=array(1..numberofintervals+1):
Fexp:=array(1..numberofintervals+1):
Mexp:=array(1..numberofintervals+1):

texp:=array(1..15):
texp[1]:=1:
texp[2]:=7.29568354608:
texp[3]:=31.8387278704:
texp[4]:=108.763941305:
texp[5]:=321.683062152:
texp[6]:=852.851887276:
texp[7]:=2030.99832312:
texp[8]:=4323.3497897:
texp[9]:=8269.20760983:
texp[10]:=14428.410878:
texp[11]:=23376.4603501:
texp[12]:=35720.7020611:
texp[13]:=52116.7122724:
texp[14]:=73280.7764716:
texp[15]:=100000:

for i from 1 to numberofintervals+1 do
   Rexp[i]:=E0*(A1*exp(-(texp[i]/t1)^B)+(1-A1)*exp(-(texp[i]/t2)^G)):
end do:

for i from 1 to numberofintervals+1 do
   s1:=Rexp[i]*epsilon_u:
   k:=s1/(hexp/2):
   sel:=k*z:
   Mexp[i]:=int(sel*z*bexp,z=-hexp/2..hexp/2):
   Fexp[i]:=(4*Mexp[i])/lexp:
end do:

#D
#--
   expmean:=0:
   r:=0:
   rsquarecounter:=0:

   for i from 1 to numberofintervals+1 do
      rsquarecounter:=rsquarecounter+1:
      expvalues[rsquarecounter]:=Fexp[i]:
      modelpredictions[rsquarecounter]:=F[i]:
   end do:

   for i from 1 to rsquarecounter do
      expmean:=expmean+expvalues[i]/rsquarecounter:
   end do:

   sse:=array(1..rsquarecounter):
   sst:=array(1..rsquarecounter):
```

```
   for i from 1 to rsquarecounter do
      sst[i]:=(expvalues[i]):
      sse[i]:=(modelpredictions[i]-expvalues[i]):
   end do:
   for i from 1 to rsquarecounter do
      r:=r+sqrt((sse[i]/sst[i])^2):
   end do:

   D:=r/(numberofintervals+1);

#output of data
#--------------
writedata("t_IYO_transel.dat",t):
writedata("F_IYO_transel.dat",F);
writedata("texp_IYO_transel.dat",texp);
writedata("Fexp_IYO_transel.dat",Fexp);


#IYO-transversal, ideal plastic (see [1])
#----------------------------------------

#definition of elastic strain state
#----------------------------------
epsilon:= Vector(6, {(1) = -0.4131204472e-2, (2) = -0.4131204472e-2, (3) = -0.2065602236e-2, (4) = 0.4437817303e-3,
(5) = 0.8875634608e-3, (6) = 0.8875634608e-3});

ho := 0.2340948109:                          #in mm
hu := 0.7659051892:                          #in mm
hyc := 0.1458204944:                         #in mm
hyt := 0.03132862185:                        #in mm

#determination of stress state and moment
#----------------------------------------
l:=32:                                       #in mm
b:=5:                                        #in mm
hz:=Vector([-ho,-hyc,-hyc/2,hyt/2,hyt,hu]):

s:=array(1..5,1..numberofintervals+1):
F:=array(1..numberofintervals+1):
M:=array(1..numberofintervals+1):

for i from 1 to 5 do
   for j from 1 to numberofintervals+1 do
      s[i,j]:=1/J1111gwr[j]*epsilon[i]:
   end do:
end do:

for i from 1 to numberofintervals+1 do
   k1:=(s[3,i]-s[2,i])/(hz[3]-hz[2]);
   k2:=(s[5,i]-s[4,i])/(hz[5]-hz[4]);

   sy1:=k1*hz[3];
   sy2:=k2*hz[3]+d;
   d1:=solve(sy1=sy2,d);

   s1:=k1*z;
   s2:=k2*z+d1;
   M[i]:=int(s1*z*b,z=hz[1]..hz[3])+int(s2*z*b,z=hz[3]..hz[5]);

   F[i]:=4*M[i]/l;
end do:

#input of experimental data
#--------------------------
Iy:=0.42:                                    #data from [1]
epsilon_u:=0.0023:
lexp:=32:
hexp:=1:
```

```
bexp:=5:
E0:=11.6:
A1:=0.11:
t1:=50:
t2:=6.4e6:
B:=0.26:
G:=0.37:
Rexp:=array(1..numberofintervals+1):
Fexp:=array(1..numberofintervals+1):
Mexp:=array(1..numberofintervals+1):

texp:=array(1..15):
texp[1]:=1:
texp[2]:=7.29568354608:
texp[3]:=31.8387278704:
texp[4]:=108.763941305:
texp[5]:=321.683062152:
texp[6]:=852.851887276:
texp[7]:=2030.99832312:
texp[8]:=4323.3497897:
texp[9]:=8269.20760983:
texp[10]:=14428.410878:
texp[11]:=23376.4603501:
texp[12]:=35720.7020611:
texp[13]:=52116.7122724:
texp[14]:=73280.7764716:
texp[15]:=100000:

for i from 1 to numberofintervals+1 do
   Rexp[i]:=E0*(A1*exp(-(texp[i]/t1)^B)+(1-A1)*exp(-(texp[i]/t2)^G)):
end do:

for i from 1 to numberofintervals+1 do
   s1:=Rexp[i]*epsilon_u:
   k:=s1/(hexp/2):
   sel:=k*z:
   Mexp[i]:=int(sel*z*bexp,z=-hexp/2..hexp/2):
   Fexp[i]:=(4*Mexp[i])/lexp:
end do:

#D
#--
   expmean:=0:
   r:=0:
   rsquarecounter:=0:

   for i from 1 to numberofintervals+1 do
      rsquarecounter:=rsquarecounter+1:
      expvalues[rsquarecounter]:=Fexp[i]:
      modelpredictions[rsquarecounter]:=F[i]:
   end do:

   for i from 1 to rsquarecounter do
      expmean:=expmean+expvalues[i]/rsquarecounter:
   end do:

   sse:=array(1..rsquarecounter):
   sst:=array(1..rsquarecounter):

   for i from 1 to rsquarecounter do
      sst[i]:=(expvalues[i]):
      sse[i]:=(modelpredictions[i]-expvalues[i]):
   end do:
   for i from 1 to rsquarecounter do
      r:=r+sqrt((sse[i]/sst[i])^2):
   end do:

   D:=r/(numberofintervals+1);
```

```
#output of data
#--------------
writedata("t_IYO_transip.dat",t):
writedata("F_IYO_transip.dat",F);
writedata("texp_IYO_transip.dat",texp);
writedata("Fexp_IYO_transip.dat",Fexp);


#SASAKI - cantilever (see [2])
#-----------------------------

#definition of elastic strain state
#----------------------------------
epsilon:= Vector(4, {(1) = -0.1797944643e-3, (2) = -0.8989723216e-4, (3) = 0.8989723216e-4, (4) = 0.1797944643e-3});

hel:=0.25:                          #in mm

#determination of stress state and moment
#----------------------------------------
l:=40:                              #in mm
b:=8:                               #in mm
hz:=Vector([-hel,-hel/2,hel/2,hel]):

s:=array(1..4,1..numberofintervals+1):
F:=array(1..numberofintervals+1):
M:=array(1..numberofintervals+1):

for i from 1 to 4 do
   for j from 1 to numberofintervals+1 do
      s[i,j]:=1/J3333gwr[j]*epsilon[i]:
   end do:
end do:

for i from 1 to numberofintervals+1 do
   k:=(s[3,i]-s[2,i])/(hz[3]-hz[2]);

   s1:=k*z;
   M[i]:=int(s1*z*b,z=hz[1]..hz[4]);

   F[i]:=M[i]/l;
end do:

#input of experimental data
#--------------------------
w:=0.5:                              #data from [2]
Iy:=8.33e-2:
lexp:=40:
hexp:=0.5:
bexp:=8:
E0:=17.5254:
A1:=0.17:
A2:=0.83:
t1:=1.68e3:
t2:=5.08e5:
B:=0.53:

Rexp:=array(1..numberofintervals+1):
Fexp:=array(1..numberofintervals+1):
Mexp:=array(1..numberofintervals+1):

texp:=array(1..15):
texp[1]:=1:
texp[2]:=7.29568354608:
texp[3]:=31.8387278704:
texp[4]:=108.763941305:
texp[5]:=321.683062152:
texp[6]:=852.851887276:
texp[7]:=2030.99832312:
texp[8]:=4323.3497897:
```

```
texp[9]:=8269.20760983:
texp[10]:=14428.410878:
texp[11]:=23376.4603501:
texp[12]:=35720.7020611:
texp[13]:=52116.7122724:
texp[14]:=73280.7764716:
texp[15]:=100000:

for i from 1 to numberofintervals+1 do
   Rexp[i]:=E0*(A1*exp(-(texp[i]/t1)^B)+A2*exp(-(texp[i]/t2))):
end do:

for i from 1 to numberofintervals+1 do
   Fexp[i]:=(3*Rexp[i]*Iy*w)/(l^3):
end do:

#D
#--
   expmean:=0:
   r:=0:
   rsquarecounter:=0:

   for i from 1 to numberofintervals+1 do
      rsquarecounter:=rsquarecounter+1:
      expvalues[rsquarecounter]:=Fexp[i]:
      modelpredictions[rsquarecounter]:=F[i]:
   end do:

   for i from 1 to rsquarecounter do
      expmean:=expmean+expvalues[i]/rsquarecounter:
   end do:

   sse:=array(1..rsquarecounter):
   sst:=array(1..rsquarecounter):

   for i from 1 to rsquarecounter do
      sst[i]:=(expvalues[i]):
      sse[i]:=(modelpredictions[i]-expvalues[i]):
   end do:
   for i from 1 to rsquarecounter do
      r:=r+sqrt((sse[i]/sst[i])^2):
   end do:

   D:=r/(numberofintervals+1);

#output of data
#--------------
writedata("t_SAS_cant.dat",t):
writedata("F_SAS_cant.dat",F);
writedata("texp_SAS_cant.dat",texp);
writedata("Fexp_SAS_cant.dat",Fexp);
```

Bibliography

[1] Maple Version 13. Waterloo maple software, Ontario, Canada <www.maplesoft.com>(2009).

[2] J. Abate and P. Valkó. Multi-precision laplace transform inversion. *International Journal for Numerical Methods in Engineering*, 60(5):979 – 993, 2004.

[3] M. Abramowitz and I. Stegun. *Handbook of mathematical functions with formulas, graphs, and mathematical tables*. Dover, New York, 9th edition, 1973.

[4] O. Akkus, A. Polyakova-Akkus, F. Adar, and M.B. Schaffler. Aging of microstructural compartments in human compact bone. *Journal of Bone and Mineral Research*, 18(6):1012 – 1019, 2003.

[5] Anonymous. Astm d 790m-82: Standard test methods for flexural properties of unreinforced and reinforced plastics and electrical insulating materials (metric). *1983 Annual Book of ASTM Standards, ASTM, Philadelphia*, 1983.

[6] R.B. Ashman, S.C. Cowin, W.C. van Buskirk, and Rice J.C. A continuous wave technique for the measurement of the elastic properties of cortical bone. *Journal of Biomechanics*, 17(5):349 – 361, 1984.

[7] R.B. Ashman and J.Y. Rho. Elastic modulus of trabecular bone material. *Journal of Biomechanics*, 21(3):177 – 181, 1988.

[8] R. Bhowmik, K. Katti, and D.R. Katti. Mechanics of molecular collagen is influenced by hydroxyapatite in naturial bone. *Journal of Materials Science*, 42:8824 – 8837, 2007.

[9] G. Boivin and P.J. Meunier. The degree of mineralization of bone tissue measured by computerized quantitative contact microradiography. *Calcified Tissue International*, 70:503 – 511, 2002.

[10] L. Boltzmann. Zur theorie der elastischen nachwirkung (concerning the theory of the elastic aftereffect. *Sitzungsberichte der Mathematisch-Naturwissenschaftlichen Classe der Kaiserlichen Akademie der Wissenschaften*, 70(2):275 – 306, 1874. (in German).

[11] E. Bossy, M. Talmant, F. Peyrin, L. Akrout, P. Cloetens, and P. Laugier. An in vitro study of the ultrasonic axial transmission technique at the radius: 1 MHz velocity measurements are sensitive to both mineralization and introcortical porosity. *Journal of Bone and Mineral Research*, 19(9):1548 – 1556, 2004.

[12] V. Bousson, C. Bergot, A. Meunier, F. Barbot, C. Parlier-Cuau, A.-M. Laval-Jeantet, and J.-D. Laredo. CT of the middiaphyseal femur: Cortical bone mineral density and relation to porosity. *Radiology*, 217:179–187, 2000.

[13] S.W. Bowman, L.J. Gibson, W.C. Hayes, and T.A. McMahon. Results from demineralized bone creep tests suggest that collagen is responsible for the creep behavior of bone. *Journal of Biomechanical Engineering*, 121:253 – 258, 1999.

[14] J.D. Currey. Mechanical properties of bone tissues with greatly differing functions. *Journal of Biomechanics*, 12:313 – 319, 1979.

[15] J.D. Currey. The design of of mineralized hard tissues for their mechanical functions. *Journal of Experimental Biology*, 292:3285 – 3294, 1999.

[16] J.D. Currey. Incompatible mechanical properties in compact bone. *Journal of Theoretical Biology*, 231:569 – 580, 2004.

[17] J.D. Currey and G. Butler. The mechanical properties of bone tissues in children. *Journal of Bone and Joint Surgery*, 57-A(6):810 – 814, 1975.

[18] S. Cusack and A. Miller. Determination of the elastic constants of collagen by Brillouin light scattering. *Journal of Molecular Biology*, 135:39 – 51, 1979.

[19] C. Donolato. Analytical and numerical inversion of the laplace-carson transform by a differential method. *Computer Physics Communications*, 145:298 – 309, 2002.

[20] J.D. Eshelby. The determination of the elastic field of an ellipsoidal inclusion, and related problems. *Proceedings of the Royal Society London, Series A*, 241:376 – 396, 1957.

[21] M. Fondrk, E. Bahniuk, D.T. Davy, and C. Michaels. Some viscoplastic characteristics of bovine and human cortical bone. *Journal of Biomechanics*, 21:623 – 630, 1988.

[22] P. Fratzl, S. Schreiber, and K. Klaushofer. Bone mineralization as studied by small-angle X-ray scattering. *Connective Tissue Research*, 34(4):247 – 254, 1996.

[23] A. Fritsch, L. Dormieux, Ch. Hellmich, and J. Sanahuja. Mechanical behaviour of hydroxyapatite biomaterials: An experimentally validated micromechanical model for elasticity and strength. *Journal of Biomedical Materials Research Part A*, 88A:149 – 161, 2009.

[24] A. Fritsch and Ch. Hellmich. 'Universal' microstructural patterns in cortical and trabecular, extracellular and extravacular bone materials: Micromechanics-based prediction of anisotropic elasticity. *Journal of Theoretical Biology*, 244:597 – 620, 2007.

[25] A. Fritsch, Ch. Hellmich, and L. Dormieux. Ductile sliding between mineral crystals followed by rupture of collagen crosslinks: experimentally supported micromechanical explanation of bone strength. *Journal of Theoretical Biology*, 260(2):230 – 252, 2009.

[26] D. Gaver Jr. Observing stochastic processes and approximate transform inversion. *Annals of Operations Research*, 14:444 – 459, 1966.

[27] R.S. Gilmore and J.L. Katz. Elastic properties of apatites. *Journal of Materials Science*, 17:1131 – 1141, 1982.

[28] M. Gurtin and E. Sternberg. On the linear theory of viscoelasticity. *Archive for Rational Mechanics and Analysis*, 11:291 – 356, 2002.

[29] Ch. Hellmich, J.-F. Barthélémy, and L. Dormieux. Mineral-collagen interactions in elasticity of bone ultrastructure - a continuum micromechanics approach. *European Journal of Mechanics - A/Solids*, 23:783 – 810, 2004.

[30] Ch. Hellmich, C. Kober, and B. Erdmann. Micromechancs-based conversion of ct-data into anisotropic elasticity tensors, applied to fe simulations of a mandible. *Annals of Biomedical Engineering*, 36:108 – 122, 2008.

[31] Ch. Hellmich and F.-J. Ulm. Are mineralized tissues open crystal foams reinforced by crosslinked collagen? – some energy arguments. *Journal of Biomechanics*, 35:1199 – 1212, 2002.

[32] Ch. Hellmich and F.-J. Ulm. A micromechanical model for the ultrastructural stiffness of mineralized tissue. *Journal of engineering mechanics (ASCE)*, 128(8):898 – 908, 2002.

[33] Ch. Hellmich and F.-J. Ulm. A micromechanical model for the ultrastructural stiffness of mineralized tissues. *Journal of Engineering Mechanics (ASCE)*, 128(8):898 – 908, 2002.

[34] A.V. Hershey. The elasticity of an isotropic aggregate of anisotropic cubic crystals. *Journal of Applied Mechanics (ASME)*, 21:236 – 240, 1954.

[35] R. Hill. Elastic properties of reinforced solids: some theoretical principles. *Journal of the Mechanics and Physics of Solids*, 11:357 – 362, 1963.

[36] R. Hill. Continuum micro-mechanics of elastoplastic polycrystals. *Journal of Mechanics and Physics of Solids*, 13(2):89 – 101, 1965.

[37] A.J. Hodge and J.A. Petruska. Recent studies with the electron microscope on ordered aggregates of the tropocollagen molecule. In G.N. Ramachandran, editor, *Aspects of Protein Structure – Proceedings of a Symposium held in Madras 14 - 18 January 1963 and organized by the University of Madras, India*, pages 289 – 300. Academic Press, London and New York, 1963.

[38] J.L. Holden, J.G. Clement, and P.P. Phakey. Age and temperature related changes to the ultrastructure and composition of human bone mineral. *Journal of Bone and Mineral Research*, 10:1400 – 1409, 1995.

[39] T. Iyo, N. Sasaki, and M. Nakata. Anisotropic viscoelastic properties of cortical bone. *Journal of Biomechanics*, 37:1433 – 1437, 2004.

[40] J.L. Katz. Anisotropy of Young's modulus of bone. *Nature*, 283:106 – 107, 1980.

[41] J.L. Katz and K. Ukraincik. On the anisotropic elastic properties of hydroxyapatite. *Journal of Biomechanics*, 4:221 – 227, 1971.

[42] W. Knauss. Viscoelasticity and the time-dependent fracture of polymers. *Comprehensive Structural Integrity*, 7:383 – 428, 2.

[43] R.S. Lakes. *Viscoelastic Solids*. CRC Press Boca Raton, London, New York, Washington, D.C., 1959.

[44] N. Laws. The determination of stress and strain concentrations at an ellipsoidal inclusion in an anisotropic material. *Journal of Elasticity*, 7(1):91 – 97, 1977.

[45] S. Lees, J.M. Ahern, and M. Leonard. Parameters influencing the sonic velocity in compact calcified tissues of various species. *Journal of the Acoustical Society of America*, 74(1):28 – 33, 1983.

[46] S. Lees and C.L. Davidson. The role of collagen in the elastic properties of calcified tissue. *Journal of Biomechanics*, 10:473 – 486, 1977.

[47] S. Lees, D. Hanson, E.A. Page, and H.A. Mook. Comparison of dosage-dependent effects of beta-aminopropionitrile, sodium fluoride, and hydrocortisone on selected physical properties of cortical bone. *Journal of Bone and Mineral Research*, 9(9):1377 – 1389, 1994.

[48] S. Lees, J.D. Heeley, and P.F. Cleary. A study of some properties of a sample of bovine cortical bone using ultrasound. *Calcified Tissue International*, 29:107 – 117, 1979.

[49] S. Lees, K.S. Prostak, V.K. Ingle, and K. Kjoller. The loci of mineral in turkey leg tendon as seen by atomic force microscope and electron microscopy. *Calcified Tissue International*, 55:180 – 189, 1994.

[50] R.B. Martin and D.L. Boardman. The effects of collagen fiber orientation, porosity, density, and mineralization on bovine cortical bone bending properties. *Journal of Biomechanics*, 26(9):1047 – 1054, 1993.

[51] M. Mauch, J.D. Currey, and A.J. Sedman. Creep fracture in bones with different stiffnesses. *Journal of Biomechanics*, 25:11 – 16, 1992.

[52] R.N. McCarthy, L.B. Jeffcott, and R.N. McCartney. Ultrasound speed in equine cortical bone: effects of orientation, density, porosity and temperature. *Journal of Biomechanics*, 23(11):1139 – 1143, 1990.

[53] A.E. Melnis, B.O. Ozola, and P.A. Moorlat. Comparative characteristics of the creep properties of human compact bone tissue under various specimen storage and testing conditions. *Mekhanika Kompozitnykh Materialov*, 3:515 – 521, 1981. Translated.

[54] A. Miller. Collagen: the organic matrix of bone. *Philosophical Transactions of the Royal Society in London Series B*, 304:455 – 477, 1984.

[55] T. Mori and K. Tanaka. Average stress in matrix and average elastic energy of materials with misfitting inclusions. *Acta Metallurgica*, 21(5):571 – 574, 1973.

[56] F.R.N. Nabarro. The time constant of logarithmic creep and relaxation. *Material Science and Engineering*, A309-A310:227 – 228, 2001.

[57] J.C. Nadeau and M. Ferrari. Invariant tensor-to-matrix mappings for evaluation of tensorial expressions. *Journal of Elasticity*, 52:43 – 61, 1998.

[58] H.C. Park and R.S. Lakes. Cosserat micromechanics of human bone: strain redistribution by a hydration sensitive constituent. *Journal of Biomechanics*, 19:385 – 397, 1986.

[59] F. Perus. Continuum microviscoelasticity: A new model for basic creep of concrete. unpublished, 2009.

[60] F. Peters, K. Schwarz, and M. Epple. The structure of bone studied with synchrotron x-ray diffraction, x-ray absorption spectroscopy and thermal analysis. *Thermochimica Acta*, 361:131 – 138, 2000.

[61] B. Pichler, Ch. Hellmich, and J. Eberhardsteiner. Spherical and acicular representation of hydrates in a micromechanical model for cement paste: prediction of early-age elasticity and strength. *Acta Mechanica*, 203:137 – 162, 2009.

[62] D.T. Reilly and A.H. Burstein. The elastic and ultimate properties of compact bone tissue. *Journal of Biomechanics*, 8:393 – 405, 1975.

[63] C.M. Rimnac, A.A. Petko, T.J. Santner, and T.M. Wright. The effect of temperature, stress and microstructure on the creep of compact bovine bone. *Journal of Biomechanics*, 26(3):219 – 228, 1993.

[64] C.M. Rimnac, A.A. Petko, and T.M. Wright. Creep of compact bone: effects of temperature and stress on bovine lamellar microstructure. *International Transactions in Operational Research*, 18:152, 1991.

[65] P. Roschger, H.S. Gupta, A. Berzlanovich, G. Ittner, D.W. Dempster, P. Fratzl, F. Cosman, M. Parisien, R. Lindsay, J.W. Nieves, and K. Klaushofer. Constant mineralization density distribution in cancellous human bone. *Bone*, 32:316 – 323, 2003.

[66] J. Salen¸con. Viscoelasticit (viscoelasticity). *Presses de l'Ecole Nationale des Ponts et Chausses, Paris*, 1983. (in French).

[67] N. Sasaki, Y. Nakayama, M. Yoshikawa, and A. Enyo. Stress relaxation function of bone and bone collagen. *Journal of Biomechanics*, 26(12):1369 – 1376, 1993.

[68] S. Scheiner and Ch. Hellmich. Continuum microviscoelasticity model for aging basic creep of early-age concrete. *Journal of Engineering Mechanics*, 135(4):307 – 323, 2009.

[69] F. Schwarzl and L. Struik. Analysis of relaxation measurements. *Advances in Molecular Relaxation Processes*, 1(3):201 – 255, 1968.

[70] H. Schwefel. *Numerische Optimierung von Computer-Modellen mittels der Evolution-Strategie*. Birkhuser, Basel, Switzerland, 1977. (in German).

[71] A. Stroud. Approximate calculation of multiple integrals. *Prentice-Hall, Englewood Cliffs*, 1971.

[72] P. Suquet, editor. *Continuum micromechanics*. Springer, Wien – New York, 1997.

[73] D. TerHaar. A phenomenological theory of visco-elastic behaviour. *Physica (Amsterdam)*, 16(9):719 – 737, 1950.

[74] P. Valkó and J. Abate. Comparison of sequence accelerators for the gaver method of numerical laplace transform inversion. *Computers and Mathematics with Applications*, 48:629 – 636, 2004.

[75] M. Vandamme and F.-J. Ulm. Viscoelastic solutions for conical indentations. *International Journal of Solids and Structures*, 43(10):3142 – 3165, 2006.

[76] K. Wakashima and H. Tsukamoto. Mean-field micromechanics model and its application to the analysis of thermomechanical behaviour of composite materials. *Materials Science and Engineering A*, 146(1-2):291 – 316, 1991.

[77] D. Widder. *The Laplace Transform*. Princeton University Press, 2^{nd} edition, 1946.

[78] D. Zahn, O. Hochrein, A. Kawska, J. Brickmann, and R. Kniep. Towards an atomistic understanding of apatite-collagen biomaterials: linking molecular simulation studies of complex-, crystal- and composite-formation to experimental findings. *Journal of Material Science*, 42:8966 – 8973, 2007.

[79] A. Zaoui. *Structural morphology and constitutive behavior of microheterogeneous materials*, pages 291 – 347. In Suquet [72], 1997.

[80] A. Zaoui. Continuum micromechanics: Survey. *Journal of Engineering Mechanics (ASCE)*, 128(8):808 – 816, 2002.

List of Figures

List of Tables