Application of Virtual Fields Method for the mechanical characterization of atherosclerotic plaques

by

A.M.A Abdollah

Application of Virtual Fields Method for the mechanical characterization of atherosclerotic plaques

by

Ariwan M.A. Abdollah

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Wednesday, June 1, 2022 at 01:00 PM.

Student number: 4225813
Delft University of Technology – Erasmus University Medical Center

Thesis committee:

Dr. ir. A. C. Akyildiz	Erasmus MC / TU Delft, Supervisor
Dr S. Kumar	TU Delft
Dr. ir. F.J.H. Gijsen	Erasmus MC / TU Delft

Preface

This work contains my master's thesis to obtain the degree of Master of Mechanical Engineering at Delft University of Technology. It was performed at the Erasmus Medical Center, under the Biomechanics Laboratory. Here I started beforehand with an internship on a different topic within atherosclerosis, followed by a literature study regarding the Virtual Fields Method in biomechanics. This was then the inspiration to apply the Virtual Fields Method to atherosclerotic lesions. This work, and the ones before, would not have been possible without some people, to whom I'd like to extend my gratitude.

First I'd like to thank my parents for their infinite love and helping hand. They were the safety net on which I could fall back whenever needed, they were there when things got tough, and they'll also be there to enjoy the good times to come. Thank you, mom and dad, your sacrifices years and years ago made all this (and more to come) possible. My brothers and sister also deserve thanks, they helped me stay motivated during this journey.

Next, I'd like to show appreciation to the BME group members. The friendliest group of colleagues I could ask for during my time at the Erasmus MC. It was lovely working with you and I hope to see you again in future endeavors.

Lastly, I'd like to thank my supervisor Ali Akyildiz. Our many, many discussions were a guiding force throughout my internship, literature review, and thesis. Thank you very much for being a wonderful supervisor. And best of luck with the newest little one in your family.

Ariwan M.A. Abdollah
May 2022

Contents

List of symbols & abbreviations

Symbols

a_j	=	Area of element j in the deformed configuration.
$\underline{\underline{A}}$	=	Internal virtual work tensor without the constitutive parameter (see Equation (17)).
$\underline{\underline{A}}_m$	=	Modified internal virtual tensor work without the constitutive parameter.
$\underline{\underline{A}}_{fr}$	=	Internal virtual work tensor without the constitutive parameter where full rank has been guaranteed.
$\underline{\underline{A}}_{fr2}$	=	Tensor $\underline{\underline{A}}_{fr}$ reduced to the linear independent rows only.
$\underline{b}$	=	External virtual work vector.
$\underline{b}_m$	=	External virtual work vector corresponding to modified tensor $\underline{\underline{A}}_m$.
$\underline{b}_{fr}$	=	External virtual work vector corresponding to tensor $\underline{\underline{A}}_{fr}$.
$\underline{b}_{fr2}$	=	External virtual work vector corresponding to tensor $\underline{\underline{A}}_{fr2}$.
$\underline{\underline{B}}$	=	Left Cauchy-Green deformation tensor.
$\underline{\underline{B}}^j$	=	Left Cauchy-Green deformation tensor of element j.
c_1	=	Constitutive parameter of the neo-Hookean material model.
c_1^j	=	Neo-Hookean constitutive parameter of element j.
$\underline{c}$	=	Constitutive parameter distribution vector.
$\underline{c}_{gt}$	=	Constitutive parameter distribution vector of the ground truth.
E	=	Young's modulus.
$\underline{\underline{F}}$	=	Deformation gradient tensor.
$\underline{\underline{I}}$	=	Identity tensor.
I_1	=	First invariant of the left Cauchy-Green deformation tensor $\underline{\underline{B}}$.
I_2	=	Second invariant of the left Cauchy-Green deformation tensor $\underline{\underline{B}}$.
j	=	Element number of the elements throughout a cross-section.
J	=	Total number of elements throughout a cross-section.
k_n	=	Exponent of virtual displacement field n.
l_o	=	Edge length of element o on the luminal side in the deformed configuration.
n	=	Virtual displacement field number.
$\underline{n}$	=	Local unit vector normal to the lumen-intima interface.
$\underline{n}_o$	=	Unit vector normal to the edge of intima element o on the luminal side in the deformed configuration.
N	=	Total number of virtual displacement fields.
o	=	Element number of the elements in the intima bordering the lumen.
O	=	Total number of elements in the intima bordering the lumen.
p	=	Hydrostatic pressure.
P	=	Intraluminal pressure.
$\underline{\underline{Q}}$	=	In-plane stiffness tensor.
r_w	=	Total number of elements in the wall component.
r_i	=	Total number of elements in the intima component.
r_l	=	Total number of elements in the lumen component.
r_c	=	Total number of elements in the calcified tissue component.
S	=	Surface area in the deformed configuration.
t	=	Cross-section thickness.
$\underline{T}$	=	Traction load vector.
$\underline{u}^*$	=	Virtual displacement field (VDF) vector.
$\underline{u}$	=	Actual displacement field vector.

V	=	Volume in the deformed configuration.
x	=	x-coordinate in the deformed configuration.
y	=	y-coordinate in the deformed configuration.

$\underline{\Delta}$	=	Difference between internal virtual work vector and external virtual work vector.
$\underline{\Delta}_{rel}$	=	Relative difference between internal virtual work vector and external virtual work. Relative to the external virtual work vector.
$\underline{\underline{\varepsilon}}$	=	Strain tensor.
θ	=	Angle by which the cross-sections are rotated.
λ	=	Eigenvalue.
ν	=	Poisson's ratio.
ξ	=	Cost function.
$\underline{\underline{\sigma}}$	=	Cauchy stress tensor.
$\underline{\underline{\sigma}}^j$	=	Cauchy stress tensor of element j.
Φ	=	Metric indicating the proximity of the calculated constitutive parameter distribution to the ground truth.
Ψ	=	Strain energy function.

Abbreviations

CS	=	Cross-section.
FE	=	Finite element.
GT	=	Ground truth.
M	=	Media.
NC	=	Necrotic core.
VDF	=	Virtual displacement field ($\underline{u}^*$).
VFM	=	Virtual fields method.

Abstract

Atherosclerosis is a lipid-driven inflammatory disease of the arteries. It causes increased thickness of the intima layer of an artery, due to build-up of, among others, lipid deposits. Eventually, this could cause rupture of the blood vessel's wall, leading to thrombus forming and possibly to clinical events. Clinically, the degree of stenosis (degree of narrowing of the lumen) is used as a metric for risk assessment. However, narrow arteries do not necessarily rupture, while wide arteries can still rupture. Since rupture is a mechanical event, (mechanical) stress in the arteries could be a better metric to assess rupture. Stress could be calculated through a chosen constitutive model. However, these constitutive models require patient-specific constitutive parameters, which presently are challenging to obtain.

To retrieve the patient-specific constitutive parameter(s), the Virtual Fields Method (VFM) could be used. This is a method based on the principle of virtual work. By minimizing the difference between the internal and external virtual work, the constitutive parameter(s) can be calculated. In this study, five atherosclerotic plaques were modeled with a neo-Hookean material model, subjected to intraluminal pressure in a Finite Element (FE) environment. The nodal deformations were used to apply the VFM in MATLAB. This was done at three levels: level 1 assumed that, for each model, a single constitutive parameter dictated the stress behavior of the entire atherosclerotic model. At level 2, the models were divided into four mechanically relevant components: intima, wall, calcified tissue, and lipid component. The ground truth for the constitutive parameter of each component was unique (120 kPa, 250 kPa, 5 kPa, and 423 kPa for intima, wall, lipid, and calcified tissue component respectively). For each of the components, a constitutive parameter was calculated. At the last level, the nodes provided by the FE model were used to create four-noded quadrilateral elements. For each element separately, a constitutive parameter was calculated. This level presents two sublevels: at the first sublevel (level 3a), the ground truth for the constitutive parameter of all elements was set at 100 kPa. At the second sublevel (level 3b), the ground truth for the constitutive parameter of an element depended on the component the element belonged to; the elements that belonged to the intima, wall, lipid, or calcified tissue component had a ground truth value for their constitutive parameter of 120 kPa, 250 kPa, 5 kPa, or 423 kPa, respectively.

The results showed how for the first level, the VFM was able to calculate the constitutive parameter within 0.2% *of the ground truth. For level 2, some of the atherosclerotic arteries' components were calculated very accurately (within* 2% *of their ground truth), while others differed greatly (by more than* 20% *of their ground truth). For level 3a, it was shown how a good initial guess for the minimization algorithm could produce very accurate results. For level 3b, a good initial guess was lacking and the minimization algorithm was not capable of finding results that were in close proximity to the ground truth.*

Clinically, the application of the VFM can be an interesting approach for assisting clinicians to decide on intervening. However, this still requires some improvements from the current work. The current study serves as a foundation upon which to build.

1 Introduction

1.1 Atherosclerosis

The human blood vessel consists of three layers: the adventitia, the media, and the intima. These are the outer, middle, and inner layers, respectively (Figure 1). Atherosclerosis is a disease of the vasculature that

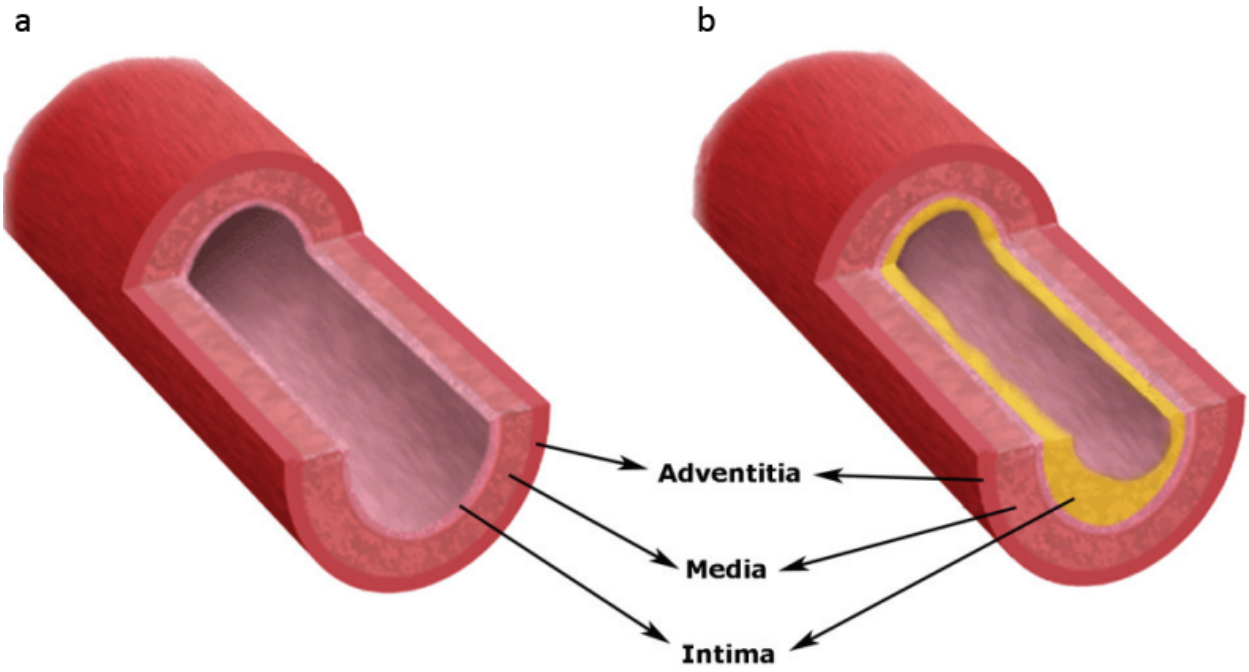

Figure 1: *a): Schematic cross-section of a healthy vessel, with the adventitia, media, and intima layers. b): Schematic cross-section of an atherosclerotic artery. Here the same layers are indicated. As can be seen, the intima layer is thickened. Modified from [1].*

can start at a young age in human life. Those affected with atherosclerosis have an intima layer that is increased in thickness mainly due to the build-up of lipid, cholesterol, and calcium [2]. At these lesion sites, also called atherosclerotic plaques, the vessel can eventually occlude or rupture, which could ultimately be fatal. Figure 2 shows a histological cross-section and a schematic representation of an atherosclerotic artery. It also shows a section of a healthy carotid artery cross-section. As can be seen, the intima layer relative to the media layer is much thicker for the atherosclerotic cross-section than the healthy cross-section.

As can be seen, due to the build-up of the necrotic core, the intima layer is much thicker (the necrotic core is the lipid-rich component). Worldwide, atherosclerosis is the leading cause of mortality and disability [3]. It can be caused by a high concentration of low-density lipoprotein. However, more often, it is induced by a low concentration of low-density lipoproteins, accompanied by a risk factor (e.g. smoking, hypertension, or genetic susceptibility) [3]. The abdominal aorta, coronary arteries, iliac arteries, and carotid bifurcations are typically most affected [3].

Atherosclerosis can reduce or obstruct coronary blood flow, although this is rarely fatal in absence of scarring of the heart muscle [3]. However, atherosclerosis can lead to rupture of the intima, which can cause clinical events. A significant concern with atherosclerosis is that it can grow without showing symptoms. This can lead to situations of acute clinical events (e.g. ischemic strokes, myocardial infarction) [1, 3]. Plaque rupture often occurs at the part of the intima separating the necrotic core and the lumen, also called the fibrous cap. Once rupture occurs here, the highly thrombogenic necrotic core can come into contact with the blood, which could lead to a heart attack or a stroke [1].

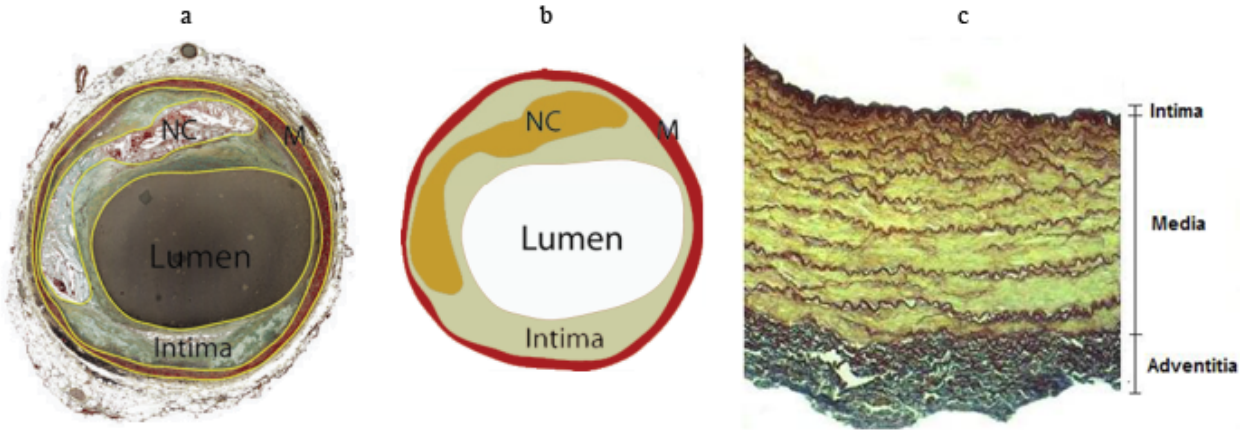

Figure 2: *a): Histological cross-section of a diseased artery. Modified from [1]. b): Schematic representation of a diseased artery. The necrotic core (NC), intima, media (M), and lumen have been indicated in the images. Modified from [1]. c) Part of a healthy carotid artery cross-section. It shows how the intima layer is smaller relative to the media layer for a healthy artery. Modified from [4].*

For clinicians, the level of stenosis (the extent of narrowing of the lumen) is used to decide on the risk for a plaque to lead to a clinical event, where a highly stenotic (narrowed) artery is considered at more risk. However, the level of stenosis is not sufficient for predicting the risk of rupture, since plaques with a low level of stenosis (wide arteries) can rupture, while plaques with a high level of stenosis (narrow arteries) might not rupture at all [2]. As plaque rupture is closely related to acute clinical manifestations [1], a better marker is needed for risk assessment.

Generally, when the stresses inside a material (caused by some external factor) exceed the material's strength, failure of the material occurs. Similarly, when the stresses caused by the blood pressure exceed the strength of the arterial tissue, rupture occurs. Rupture can occur spontaneously, but sometimes it is caused by other reasons, such as a temporary increase in emotional or physical stress [1, 3]. Since plaque rupture is a mechanical event, mechanical indicators, such as stress, might be better markers for the prediction of plaque rupture. Stress can be calculated through computational modeling. This requires several inputs: the geometry of the atherosclerotic plaque, the loading and boundary conditions, displacements, and the constitutive model. This is more challenging for atherosclerotic plaques, as their stress response is anisotropic and heterogeneous in nature. These inputs need to be as accurate as possible for accurate stress identification. For atherosclerotic plaques, the geometry can be found through imaging, the loading conditions can be measured, and the deformations can be measured through imaging. The constitutive model, however, requires patient-specific constitutive parameters. Currently, these patient-specific constitutive parameters are challenging to obtain.

1.2 Material characterization and virtual fields method

For material with simple stress-strain behavior, the constitutive parameters could be extracted through experimental methods. For instance, for metals under small deformations, a tension test could provide a stress-strain curve, with which the Young's modulus E could be calculated through the slope of the curve. For more complex geometries and material models, simple analytical solutions for material characterization might not be sufficient. A commonly used method for finding constitutive parameters in such cases is the inverse Finite Element (FE) method. Generally, this occurs by measuring the spatial deformation of an object, that underwent deformation due to a known load. The next step is to choose a constitutive model, for which constitutive parameters are required. These are yet unknown, thus an estimation is made (based on literature). Then, a computer model is made with the same geometry, loading conditions, and boundary conditions, and the simulation is started. The actual deformation is then recreated in the FE environment

using the computer model. If the difference between measured displacement and the calculated displacement (from FE) is below a predefined threshold, the estimated constitutive parameters are accepted. If they do not match, the estimated constitutive parameters are adjusted and the FE simulation is run again. This is repeated until the difference in displacements is below the predefined threshold. Once this is the case, the constitutive parameters are accepted.

The drawback of the inverse FE method is that it is a lengthy process. Each iteration requires a FE simulation. In FE, the computation time for a single simulation scales by dof^{α}, where dof is the degrees of freedom of the FE model and α is a constant dependent on the solver [5]. If the object under investigation is considered in three-dimensional geometry, with many elements, and a complex constitutive model is used, retrieving the constitutive parameters using inverse FE method could become too time-consuming.

Another method for calculating constitutive parameters is called the Virtual Fields Method (VFM). This is a method based on the virtual work equation that relates full-field spatial deformation to constitutive parameters. As this method has spatial deformation as one of its inputs, it can be patient-specific, even plaque-specific. The goal of the VFM is to calculate constitutive parameters, with which it is possible to locally calculate stresses inside a material. Furthermore, the VFM does not involve any simulations, therefore it is much less computationally expensive. This method relies on minimizing the difference between external virtual work (due to an external loading) and internal virtual work (as a reaction to the external loading). The VFM has previously been applied for the material characterization of all kinds of materials, such as fiber composites, metals, and wood [6]. In biomechanics, it has been applied to murine aortas [7, 8, 9, 10], human aortas [11, 12], human eardrum [13], human optic nerve head [5, 14], and canine left ventricle [15].

The goal of this thesis is to apply the VFM to simplified FE models of atherosclerotic plaques for the calculation of their constitutive parameter(s).

This was done by first performing FE simulations of several atherosclerotic plaques, where the constitutive parameters were chosen based on literature. The FE simulations synthetically generated the output of clinical imaging. The output of the FE simulations (nodal displacements) served as the input for the numerical application of the VFM performed in MATLAB (R2020a, The Mathworks Inc.), using in-house written script. This was done at 3 levels. At level 1, the atherosclerotic plaques were assumed to have a homogeneous constitutive parameter for the entire cross-section. At level 2, the components within the atherosclerotic plaques behaved according to their own constitutive parameter. At level 3, the atherosclerotic plaques were divided into elements based on the nodes provided by FE, where four neighboring nodes created a quadrilateral element. The stress response of each element was dictated by its own constitutive parameter. At this level, the constitutive parameter of each element was calculated separately. Level 3 was divided into two sublevels. At level 3a, the ground truths for the constitutive parameter of the elements all had the same value. At level 3b, the ground truth for the constitutive parameter of an element was dependent on the component to which that element belonged. I.e. the elements belonging to the intima component all had the same ground truth for the constitutive parameter, which was different from the ground truth for the constitutive parameter of elements in the lipid component. These ground truths were defined by the FE environment when the constitutive parameter of each component was given. At all levels, the result of applying the VFM should reflect the constitutive parameters given in the FE environment.

2 Method

2.1 Theory of Virtual Fields Method

The VFM is a method for the calculation of constitutive parameters. It commences from the virtual work equation. The virtual work equation for a continuum body subjected to (quasi-)static load with negligible body forces (in a three-dimensional Cartesian coordinate system) is given by [6]:

$$\overbrace{\int_V \underline{\underline{\sigma}} : \frac{\partial \underline{u}^*}{\partial \underline{x}} dV}^{\text{Internal virtual work}} = \overbrace{\int_S \underline{T} \cdot \underline{u}^* dS}^{\text{External virtual work}} \tag{1}$$

Here, $\underline{\underline{\sigma}}$ is the Cauchy stress tensor in the deformed configuration (double underlined variables indicate second-order tensors), $\underline{u}^*$ is the Virtual Displacement Field (VDF) vector (single underlined variables indicate vectors), $\underline{x}$ is the location vector in the deformed configuration ($\underline{x} = \begin{bmatrix} x \\ y \end{bmatrix}$, with x and y being the location coordinates), V is the volume of the deformed configuration, $\underline{T}$ is the traction load vector in the deformed configuration, and S is the surface area in the deformed configuration. The right-hand side of Equation (1) is the virtual work done by external forces (external virtual work) and the left-hand side is the virtual work done internally as a response (internal virtual work).

By choosing a constitutive model, the Cauchy stress tensor $\underline{\underline{\sigma}}$ is defined as a function of spatial variables and constitutive parameters. This means that the in the expression for virtual work (Equation (1)), there are 5 terms: constitutive parameters, spatial variables, VDF $\underline{u}^*$, its Jacobian $\frac{\partial \underline{u}^*}{\partial \underline{x}}$, and traction load vector $\underline{T}$. Thus, to calculate the constitutive parameters, information is required on the latter four. The externally applied traction load vector $\underline{T}$ should be known a priori. The spatial variable is calculated by first measuring full-field displacements, then using these measurements to calculate the right spatial variable. Which spatial variable is required, depends on the constitutive model. For instance, if an isotropic linear elastic material model is assumed, the full-field displacement data is used to calculate the strain $\underline{\underline{\varepsilon}}$.

In a physical sense, the VDF could be understood as a chosen mathematical expression that dictates how each material point of an object virtually deforms from the deformed configuration into the virtual configuration. In this virtual configuration, Equation (1) holds. The requirement for the VDF is that it is kinematically admissible, i.e. where the actual deformation is equal to zero, the virtual deformation must also be equal to zero. Note that the VDF $\underline{u}^*$ and (actual) full-field displacement $\underline{u}$ are not related. The VDF is a chosen mathematical expression, while the full-field deformation $\underline{u}$ measures how the material points of an object deform from the reference configuration to the deformed configuration. Strictly speaking, the VDF need not have a unit of distance. This is because the principle of virtual work (on which the VFM is based) is not an energy balance, but is equivalent to local equilibrium in combination with force boundary conditions for a given object. See [6] for further reading.

If the constitutive model contains more than 1 constitutive parameter, a single VDF is not enough. E.g. for a two-dimensional object modeled as an isotropic linear elastic material, two constitutive parameters dictate the material's stress behavior: Young's modulus E and Poisson's ratio ν. The constitutive model is then given by $\underline{\underline{\sigma}} = \underline{\underline{Q}}\,\underline{\underline{\varepsilon}}$ with $\underline{\underline{Q}} = \underline{\underline{Q}}(E, \nu)$, where $\underline{\underline{Q}}$ is the in-plane stiffness tensor [6]. In such a case, at least two VDFs are required, so that two equations are obtained to solve for the two unknowns. If the VDFs are not a linear combination of one another and they are kinematically admissible, two independent equations can be created using Equation (1), with which the two constitutive parameters can be calculated. The VDFs cannot be a linear combination of one another, as they will not be independent, and two independent equations are required if two unknowns are to be calculated.

2.2 VFM applied to intraluminally pressurized atherosclerotic arteries

In this thesis, the atherosclerotic arteries are modeled as incompressible isotropic materials, that assume neo-Hookean material behavior. This is because biological tissue comprises about 70% of water (and thus can be considered incompressible) and the stress behavior of biological tissue does not behave linearly [16].

To calculate the Cauchy stress tensor $\underline{\underline{\sigma}}$ for this material, the strain energy function Ψ of the incompressible isotropic neo-Hookean material model must be differentiated w.r.t the first invariant (I_1) and second invariant (I_2) of the left Cauchy-Green deformation tensor $\underline{\underline{B}}$ [17]:

$$\underline{\underline{\sigma}} = -p\underline{\underline{I}} + 2\frac{\partial \Psi}{\partial I_1}\underline{\underline{B}} - 2\frac{\partial \Psi}{\partial I_2}\underline{\underline{B}}^{-1} \tag{2}$$

Here, p is the hydrostatic pressure, due to the incompressible behavior of biological tissue, $\underline{\underline{I}}$ is the identity tensor, and $\underline{\underline{B}}$ is the left Cauchy-Green deformation tensor, given by:

$$\underline{\underline{B}} = \underline{\underline{F}}\,\underline{\underline{F}}^T \tag{3}$$

Here T indicates the transpose and $\underline{\underline{F}}$ is the deformation gradient, defined as:

$$\underline{\underline{F}} = \frac{\partial \underline{x}}{\partial \underline{X}} \tag{4}$$

with $\underline{X}$ the location vector of the reference configuration. The first and second invariants of the left Cauchy-Green deformation tensor are defined as [17]:

$$I_1(\underline{\underline{B}}) = tr(\underline{\underline{B}}) = \lambda_1 + \lambda_2 + \lambda_3 \tag{5}$$

$$I_2(\underline{\underline{B}}) = \frac{1}{2}[tr(\underline{\underline{B}}) - tr(\underline{\underline{B}}^2)] \tag{6}$$

where λ_1, λ_2, and λ_3 are the eigenvalues of the left Cauchy-Green deformation tensor $\underline{\underline{B}}$. The strain energy function for incompressible isotropic neo-Hookean material is defined by [17]:

$$\Psi = c_1(I_1 - 3) \tag{7}$$

Here c_1 is the constitutive parameter. Combining Equation (2) and Equation (7) gives the Cauchy stress tensor for incompressible neo-Hookean materials:

$$\underline{\underline{\sigma}} = -p\underline{\underline{I}} + 2c_1\underline{\underline{B}} \tag{8}$$

Rewriting Equation (1) with this constitutive model gives:

$$\int_V (-p\underline{\underline{I}} + 2c_1\underline{\underline{B}}) : \frac{\partial \underline{u}^*}{\partial \underline{x}} dV = \int_S \underline{T} \cdot \underline{u}^* dS \tag{9}$$

Here, the traction load vector $\underline{T}$ is given as:

$$\underline{T} = \underline{n}P \tag{10}$$

where P is the intraluminal pressure and $\underline{n}$ is the local unit vector normal to the lumen-intima interface directed towards the lumen. The hydrostatic pressure $p(x, y, z)$ is non-uniform throughout the cross-section and is not of importance for the constitutive parameter characterization. Therefore, eliminating this variable is desirable. By choosing the VDF $\underline{u}^*$ such that $-p\underline{\underline{I}} : \frac{\partial \underline{u}^*}{\partial \underline{x}}$ is equal to zero, the hydrostatic pressure can be eliminated from the internal virtual work calculation, and thus need not be calculated. This requirement can be achieved if

$$tr(\frac{\partial \underline{u}^*}{\partial \underline{x}}) = 0 \tag{11}$$

In further analysis, it is assumed this condition holds, and thus the hydrostatic pressure term is omitted from the Cauchy stress expression. Section 2.5 gives the VDFs used for which this condition holds.

2.3 In-silico experiments through FE analysis

To retrieve constitutive parameters of the atherosclerotic plaques, computer models of the atherosclerotic plaques needed to be made. From histology images of atherosclerotic carotid arteries, two-dimensional FE atherosclerotic models were obtained. All these two-dimensional FE models operated under plane strain assumption and contained the four mechanically relevant components: diseased intima, calcified tissue, lipid, and wall (adventitia + media) [1]. The segmentation of the atherosclerotic plaques and the subsequent creation of the FE models were performed in previous work [18]. Experiments were performed in a FE environment (ABAQUS/Standard (Dassault systems)), where the two-dimensional nodal displacements were the output. The FE simulations represented the experimental full-field displacement measurements. A pressure load was applied to the lumen at 100 mmHg (13.33 kPa). The FE models were meshed with quadrilateral elements with an approximate edge length of 50 μm, unless specified otherwise. The FE models were bounded by fully encastering a single node on the outer edge of the adventitia, i.e. no rigid body displacements were allowed.

2.4 Constitutive parameter extraction at 3 levels

In this work, constitutive parameter calculation occurred at three levels. At level 1, also called the homogeneous approach, the FE models were considered homogeneous. Thus, one constitutive parameter described the stress behavior of the entire atherosclerotic plaque. In this level, the VFM was applied to calculate a single constitutive parameter c_1 per FE model.

At level 2, also called the segmentation approach, the FE models were divided into four mechanically relevant components: the intima component, the wall component, the lipid component, and the calcified tissue component. A distinct constitutive parameter represented each component. In this level, four constitutive parameters per FE model were calculated using the VFM. In the FE environment, $c_{1,wall} = 250$ kPa, $c_{1,intima} = 120$ kPa, $c_{1,lipd} = 5$ kPa, and $c_{1,calcifiedtissue} = 423$ kPa. These values were based on literature [19, 20] and served as ground truth. The VFM was used to find these values from the nodal displacements.

At level 3, also called the segmentation-less approach, the FE models were divided into many elements. These elements were created by the nodal points, where four neighboring nodes made up the corners of the quadrilateral element. Essentially, the mesh created by the FE environment was recreated, and for each element within the mesh, a constitutive parameter was calculated. The number of elements depended on the FE model and is included in Appendix 7.1. Each element had its constitutive parameter, thus the VFM was applied to each element. This level was further divided into two sublevels per FE model. At the first sublevel (level 3a), the ground truth for the constitutive parameter of all the elements was set at 100 kPa. At the second sublevel (level 3b), the ground truth for the constitutive parameter of an element was dictated by the component that element belonged to, where the same values as level 2 were adopted: The ground truth for elements that belong to the lipid component was 5 kPa; the ground truth for elements that belong to the calcified tissue component was 423 kPa, etc. Note that in both sublevels, the cross-section was not segmented into components. In the calculation of the constitutive parameters, elements within the same component had the freedom to differ from one another (unlike at level 2). Only in the second sublevel, the component an element belonged to, was used to set the ground truth of that element's constitutive parameter.

2.5 Virtual displacement fields

For the calculation of the constitutive parameters, (at least) one VDF is required per unknown constitutive parameter. At level 1, as described above, at least one VDF is required. For level 2, at least four VDFs are required. For level 3, this is at least equal to the number of elements, which differs per cross-section. On top of this, the VDFs u^* for all levels are not allowed to be a linear combination of each other; a superposition of two (or more) VDFs does not yield a new, independent VDF. Furthermore, to disregard the hydrostatic pressure $tr(\frac{\partial u^*}{\partial \underline{\underline{x}}}) = 0$ must hold.

The expression used for VDFs that takes all this into account was:

$$\underline{u}^{n*} = \begin{bmatrix} u_1^{n*} \\ u_2^{n*} \end{bmatrix} = \begin{bmatrix} \frac{x}{x^2+y^2}\left(\frac{x^2}{x^2+y^2}\right)^{k_n} \\ \frac{y}{x^2+y^2}\left(\frac{x^2}{x^2+y^2}\right)^{k_n} \end{bmatrix} = \begin{bmatrix} \frac{x^{2k_n+1}}{(x^2+y^2)^{k_n+1}} \\ \frac{yx^{2k_n}}{(x^2+y^2)^{k_n+1}} \end{bmatrix} \tag{12}$$

Here n indicates the (integer) number of the VDF and k_n is a random value with $1 < k_n < 10$ associated with VDF n. To cancel out the hydrostatic pressure $p(x, y)$ in the internal virtual work calculation, it can be verified that

$$tr\left(\frac{\partial \underline{u}^{n*}}{\partial \underline{x}}\right) = \frac{\partial u_1^{n*}}{\partial x} + \frac{\partial u_2^{n*}}{\partial y} = 0 \tag{13}$$

Proof of this is can be found in Appendix 7.2.

For the VDFs used, it is also required that the denominator cannot be equal to zero (see Equation (12)). This only happens for $x = 0 \wedge y = 0$. To make sure this cannot happen, the origin of the coordinate system was placed inside the lumen. There, no evaluation of the VDFs occurs (see right image in Figure 3), and thus the denominator in Equation (12) will never be zero.

Besides increasing the number of VDFs by choosing higher n values, the chosen VDFs can be reused by rotating the entire geometry of the atherosclerotic plaques by a certain angle θ. This changes the coordinates of the elements at which the internal and external virtual works are evaluated, thereby creating new independent virtual work equations.

Looking at the expression for the VDFs, it can be seen that for any value of k_n, x, and y, the unit of the VDF is m^{-1}. This is then not a unit of distance. However, this is also not a requirement for the VDFs. It directly influences the unit of the internal and external virtual work. Here, they did not have unit of work (Nm), but of pressure (kPa, see next section).

2.6 Numerical calculation

The output of the FE simulations was nodal displacements, with which elements were created. These formed the input to the calculations of constitutive parameters. These calculations were performed numerically in MATLAB, thus Equation (1) was discretized as

$$\sum_{j=1}^{J}(\underline{\underline{\sigma}}^j : \left.\frac{\partial \underline{u}^*}{\partial \underline{x}}\right|_j)ta_j = \sum_{o=1}^{O}(P\underline{n}_o \cdot \underline{u}^*|_o)tl_o \tag{14}$$

where j indicates the elements in the cross-section (with a total of J elements), o indicates the elements in the intima that border the lumen (with a total of O elements in the intima bordering the lumen), $\underline{\underline{\sigma}}^j$ is the Cauchy stress at the center of element j, $\underline{u}^*|_o$ is the VDF vector evaluated at the center of element o, $\left.\frac{\partial \underline{u}^*}{\partial \underline{x}}\right|_j$ is the Jacobian tensor of the VDF evaluated at element j, t is the thickness of the FE model, a_j is the area of element j in the deformed configuration, l_o is the edge length belonging to element o on the luminal side in the deformed configuration (meaning the edge of element o where traction load vector T is exerted by the blood). Traction load vector $\underline{T}$ is replaced by $\underline{n}_o P$ ($\underline{n}_o$ is the unit vector normal to edge length l_o). Filling in the expression for Cauchy stress for neo-Hookean materials (Equation (8), under the assumption that hydrostatic pressure is removed by the choice of the VDF) and dividing by thickness t for the two-dimension geometry of the cross-section gives:

$$\sum_{j=1}^{J}(2c_1^j\underline{\underline{B}}^j : \left.\frac{\partial \underline{u}^*}{\partial \underline{x}}\right|_j)a_j = \sum_{o=1}^{O}(P\underline{n}_o \cdot \underline{u}^*|_o)l_o \tag{15}$$

This expression shows how the unit for the internal and external virtual work is kPa. For instance, on the right-hand side, the unit for the VDF $\underline{u}^*|_o$ is m^{-1} (Equation (12)), that of edge length l_o is m, and that of the intraluminal pressure P is kPa. These multiplied give kPa.

If multiple constitutive parameters are to be calculated, multiple virtual displacement fields $\underline{u}^{n*}$ are required and multiple virtual work equations (Equation (15)) are created. This then leads to the following equation:

$$\underline{\underline{A}}\,\underline{c} = \underline{b} \tag{16}$$

where

$$\underline{\underline{A}} = \begin{bmatrix} (2\underline{\underline{B}}^1 : \frac{\partial \underline{u}^{1*}}{\partial \underline{x}}\Big|_1)a_1 & \dots & (2\underline{\underline{B}}^J : \frac{\partial \underline{u}^{1*}}{\partial \underline{x}}\Big|_J)a_J \\ \vdots & \ddots & \vdots \\ (2\underline{\underline{B}}^1 : \frac{\partial \underline{u}^{N*}}{\partial \underline{x}}\Big|_1)a_1 & \dots & (2\underline{\underline{B}}^J : \frac{\partial \underline{u}^{N*}}{\partial \underline{x}}\Big|_J)a_J \end{bmatrix} \tag{17}$$

$$\underline{c} = \begin{bmatrix} c_1^1 \\ \vdots \\ c_1^J \end{bmatrix} \tag{18}$$

$$\underline{b} = \begin{bmatrix} \sum_o^O (P\underline{n}_o \cdot \underline{u}_o^{1*}) l_o \\ \vdots \\ \sum_o^O (P\underline{n}_o \cdot \underline{u}_o^{N*}) l_o \end{bmatrix} \tag{19}$$

Here, $\underline{b}$ is the external virtual work vector and $\underline{c}$ is the constitutive parameter distribution vector. The cost function is defined by these three terms:

$$\xi = (\underline{\underline{A}}\,\underline{c} - \underline{b}) \cdot (\underline{\underline{A}}\,\underline{c} - \underline{b}) \tag{20}$$

The cost function was minimized by varying the constitutive parameter distribution vector $\underline{c}$ using linear least-squares algorithm. The minimization algorithm was restricted such that each calculated constitutive parameter entry $c_1^j \geq 0$. The number of rows in tensor $\underline{\underline{A}}$ is (at least) equal to the number of constitutive parameters that need to be calculated. The number of columns is equal to the number of elements created by the FE nodal points.

For the numerical calculations, Equations (17), (18), and (19) are used to calculate the constitutive parameters. For levels 1 and 2, Equation (18) was modified to reflect the level's circumstance. For the level 1, since the constitutive parameter of all the elements were the same, the constitutive parameter distribution vector $\underline{c}$ had entries that are all the same: $c_1^1 = c_1^2 = ... = c_1^J = c_1$. Therefore, the constitutive parameter distribution $\underline{c}$ looked as follows:

$$\underline{c} = \begin{bmatrix} c_1 \\ \vdots \\ c_1 \end{bmatrix} \tag{21}$$

For level 1 calculations, minimization was not required, as there was only 1 constitutive parameter to calculate. At level 1, the constitutive parameter was calculated through inversion. Even though there was only 1 unknown, there were still N VDFs used, and thus N equations were created. The average of these individual constitutive parameters was then reported as the calculated constitutive parameter of the cross-section.

For level 2, the ground truth of an element's constitutive parameter took one of four distinct values, depending on the component the element belonged to. The constitutive parameter distribution vector $\underline{c}$ was then subdivided into four groups. The wall component, intima component, lipid component, and calcified tissue component had r_w, r_i, r_l, r_c elements respectively. The constitutive distribution vector $\underline{c}$ then looked as follows:

$$\underline{c} = \begin{bmatrix} c_1^w \\ \vdots \\ c_1^w \\ c_1^i \\ \vdots \\ c_1^i \\ c_1^l \\ \vdots \\ c_1^l \\ c_1^c \\ \vdots \\ c_1^c \end{bmatrix} \tag{22}$$

where for r_w elements $c_1 = c_1^w$, for r_i elements $c_1 = c_1^i$, for r_l elements $c_1 = c_1^l$, and for r_c elements $c_1 = c_1^c$.

For level 3, the elements had the freedom to attain their own values, thus no modification of Equation (18) was required. The numerical calculations were performed using a custom-written MATLAB script. This has been included in Appendix 7.7.

To evaluate the accuracy of a calculated constitutive parameter distribution vector $\underline{c}$, a metric is required. To that end, the following metric was used:

$$\Phi = \frac{\sum_{j=1}^{J} |c_1^j - c_{gt}^j|}{\sum_{j=1}^{J} c_1^j} \tag{23}$$

where c_1^j and c_{gt}^j indicate the j-th value of the $\underline{c}$ and $\underline{c}_{gt}$ vectors respectively, with $\underline{c}_{gt}$ being the constitutive parameter distribution of the ground truth.

For $\Phi = 0$, $c_1^j = c_{gt}^j$, thus the calculated constitutive parameter distribution is equal to that of the ground truth. If $\Phi > 0$, the calculated constitutive parameter distribution vector differs from the ground truth. The larger the value of Φ, the larger the deviation of the calculated constitutive parameters from the ground truth, with a maximum of $\Phi = 1$.

3 Results

3.1 Representative cross-section

For the first cross-section, Figure 3a depicts the histology image, from which the schematic model (Figure 3b) and a FE model (Figure 3c) were created. The histology and models were created in previous work [18]. This FE model is a representative case for the other cross-sections. The FE mesh applied to the cross-sections had an approximated edge length of 50 μm. The creation of other FE models was also performed similarly [18] and are shown in Appendix 7.1.

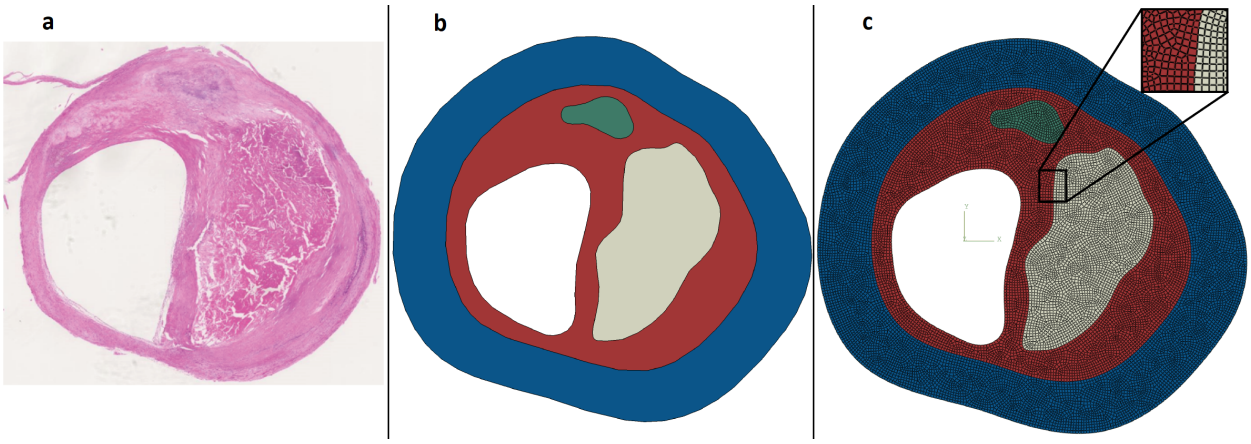

Figure 3: *a): Stained histology image of atherosclerotic artery cross-section 1. From this cross-section, a schematic model and FE model were created. b): Schematic model of the components of an atherosclerotic artery cross-section. The delineated regions all indicate a component. c): FE model of an atherosclerotic artery. The model was meshed with 17495 four-noded quadrilateral elements (black lines). The components were color-coded. The calcified tissue component can be seen in green, the wall component can be seen in blue, the lipid component is seen in gray, and the intima component can be seen in red. The inset shows the border between the intima component and lipid component. This depicts the individual four-noded quadrilateral elements in this region.*

3.2 Results for level 1: homogeneous approach

At level 1, the atherosclerotic arteries were considered homogeneous structures; the entire cross-section was dictated by one constitutive parameter. Since only one constitutive parameter was calculated, simple inversion was sufficient to calculate the constitutive parameter. To account for random errors (e.g. rounding errors), multiple VDFs were used to calculate the constitutive parameter many times. The average of these then represented the (definitive) constitutive parameter. In total $N = 3.5 * 10^4$ VDFs were used. Table 1 shows the calculated c_1 values for level 1. All constitutive parameter values are given in kPa. The ground truth was set at 100 kPa. As can be seen, the calculated c_1 value for each cross-section is very close to the ground truth value ($\pm$ 0.2%). For each cross-section, table 1 also gives the Φ (Equation (23)) value for each cross-section. As these values are very close to zero, it confirms the accurate calculation of the constitutive parameters.

3.3 Results for level 2: segmentation approach

At the second level, the atherosclerotic arteries were considered heterogeneous arteries, comprising of 4 components. Each component was represented by a distinct constitutive parameter. In this level, the VFM was applied to retrieve four constitutive parameters. Here, ministration was used to retrieve the constitutive

Table 1: *This table shows the calculated c_1 value for the level 1 approach, and the corresponding Φ value for each cross-section. The c_1 values are in unit of kPa. Here, the cross-sections (CS) were assumed to behave homogeneously. The ground truth was set at 100 kPa. As can be seen, the cross-sections are very close to the ground truth (GT).*

	GT	CS 1	CS2	CS3	CS4	CS5
c_1	100	100.04	100.02	100.18	100.03	100.07
Φ	0	$4.00 * 10^{-4}$	$2.00 * 10^{-4}$	$1.8 * 10^{-3}$	$3.00 * 10^{-}4$	$7.00 * 10^{-4}$

parameters. Again, $N = 3.5 * 10^4$ VDFs were used to create the same amount of virtual work equations. One of the inputs of the minimization algorithm was the initial guess of the constitutive parameter distribution. The minimization algorithm started from this initial guess. In this level, the initial guess for each of the four constitutive parameters was chosen at random between 0 and 1000 kPa. In level 2, the minimization algorithm was performed 100 times, each time with a different initial guess for all four components. The average constitutive parameter for each component of each iteration was reported in Table 2. The ground truth was set at 250 kPa, 120 kPa, 423 kPa, and 5 kPa for the wall component, intima component, calcified tissue component, and lipid component, respectively. For all cross-sections, the Φ value is also reported in Table 2.

Table 2: *This table shows the calculated c_1 values for the level 2 approach and the corresponding Φ value for each cross-section. Here the cross-sections (CS) behaved heterogeneously, by containing components that had different ground truths for their constitutive parameters. The ground truth (GT) has also been shown. All constitutive parameters are given in unit of kPa.*

	Intima	Wall	Lipid	Calcified tissue	Φ
GT	120	250	5	423	0
CS1	136.52	206.06	6.07	623.24	0.2693
CS2	121.05	251.00	5.03	419.81	0.0066
CS3	121.08	241.20	6.62	501.06	0.1030
CS4	120.36	255.87	2.47	408.81	0.0291
CS5	123.08	199.61	5.73	494.03	0.1523

As can be seen, the VFM was able to fairly accurately calculate the constitutive parameter of the individual components. The intima component was correctly calculated within 15% of the ground truth in all cross-sections. The wall component was mostly correct within 20% of the ground truth. When it comes to the lipid component, the relative error can be as large as 50%. However, this was not a large deviation in absolute terms. The ultimate goal of the VFM was for the absolute error to be small, which was the case. Lastly, the calcified tissue component showed large deviations from the ground truth. These were as big as 47%. Not only was this large in relative terms, but the absolute error of 200.24 kPa was also considerable.

The Φ values were in agreement with the differences between constitutive parameters and ground truths. Cross-section 1 had the highest Φ value ($\Phi = 0.2693$), as the c_1 value for the calcified tissue component differed greatly from that of the ground truth and likewise for the wall component. The second highest Φ value ($\Phi = 0.1523$) was from cross-section 5, due to the calcified tissue and wall component. Then came cross-section 3 ($\Phi = 0.1030$). This was due to the difference between the calculated c_1 value of the calcified tissue component and that of the ground truth. For cross-sections 2 and 4, the difference between Φ values was not very big, as their constitutive parameters were in close proximity to one another. These were also the smallest, as their calculated constitutive parameters were closest to the ground truth.

3.4 Results for level 3: segmentation-less approach

At the third level, the FE models of the atherosclerotic arteries were subdivided into four-noded quadrilateral elements. The stress response of each element was dictated by a single constitutive parameter. The VFM was used to calculate each element's constitutive parameter. At this level, the number of unknown constitutive parameters are significantly larger than the previous two levels. The number of VDFs required is dependent on the size of the cross-section and the approximate edge length of the elements. For the cross-section shown in Figure 3c, there were 17495 elements, each with an unknown constitutive parameter. Thus for this cross-section, at least the same amount of VDFs was required. With $N = 3.5 * 10^4$ VDFs, this condition was met for all cross-sections in this level.

Two sublevels are presented.

3.4.1 Level 3a: homogeneous segmentation-less approach

At the first sublevel, every element had the same ground truth value for the constitutive parameter. Here, that ground truth value for all the constitutive parameters was set at 100 kPa.

For the calculation of the constitutive parameters, it was helpful to make use of the homogeneous character of the cross-section. A good initial guess was found by starting from the notion that in this case, the entries in the constitutive parameter distribution vector $\underline{c}$ (Equation (18)) were all equal, i.e. $c_1^1 = c_1^2 = ... = c_1^J = c_1$. Therefore, Equation (15) was written as:

$$c_1 \sum_{j=1}^{J} (2\underline{\underline{B}}^j : \left.\frac{\partial \underline{u}^*}{\partial \underline{x}}\right|_j) a_j = \sum_{o=1}^{O} (P\underline{n}_o \cdot \underline{u}^*|_o) l_o \tag{24}$$

Using the nodal displacements, the deformation gradient $\underline{\underline{F}}$ was calculated, and subsequently left Cauchy-Green deformation tensor $\underline{\underline{B}} = \underline{\underline{F}}\,\underline{\underline{F}}^T$. Also, the nodal displacements allowed for the calculation of each element's areas a_e and the length l_o where the traction load was applied. Any VDF described by Equation (12) defined $\underline{u}^*|_o$ and $\left.\frac{\partial \underline{u}^*}{\partial \underline{x}}\right|_j$. The traction load vector $P\underline{n}_o$ was evaluated by knowledge of the luminal pressure (100 mmHg/13.33 kPa) and the unit vector normal to the edge of the intima on the luminal side created from the nodal displacements. The only unknown was c_1, thus simple inversion lead to an initial guess for the minimization algorithm. This initial guess was called c_{0j}

To minimize random errors (e.g. round-off errors), this was performed for multiple virtual work equations, using multiple VDFs, which lead to multiple initial guesses c_{0j}. The results of level 3a used in total $N = 3.5 * 10^4$ VDFs, with which $3.5 * 10^4$ c_{0j} values were calculated. Averaging over these resulted in the initial guess c_0 for all the elements. Essentially, the outcome of level 1 was used as an initial guess for level 3a.

Figure 4 shows the constitutive parameter distribution vector $\underline{c}$ calculated for cross-section 1. It shows the c_1 values (vertical axis) against the element number (horizontal axis), where each bar represents the constitutive parameter of a single element. As can be seen, most of the elements were calculated to have a value close to the ground truth. The ground truth is depicted by the red dashed line in the same figure. It shows how each element (horizontal axis) had a ground truth value of 100 kPa (vertical axis). The Φ value (Equation (23)) was equal to 0.0022. The proximity to zero shows that this was a very good constitutive parameter distribution. Cross-sections 2,4, and 5 showed a similar constitutive parameter distribution as Figure 4. Cross-section 3 ranged between 0.99 kPa and 278.3 kPa (see Appendix 7.3 for the bar graph). The Φ values for cross-sections 2-5 were $7.813 * 10^{-04}$, 0.0566, 0.00170, and 0.00180, respectively.

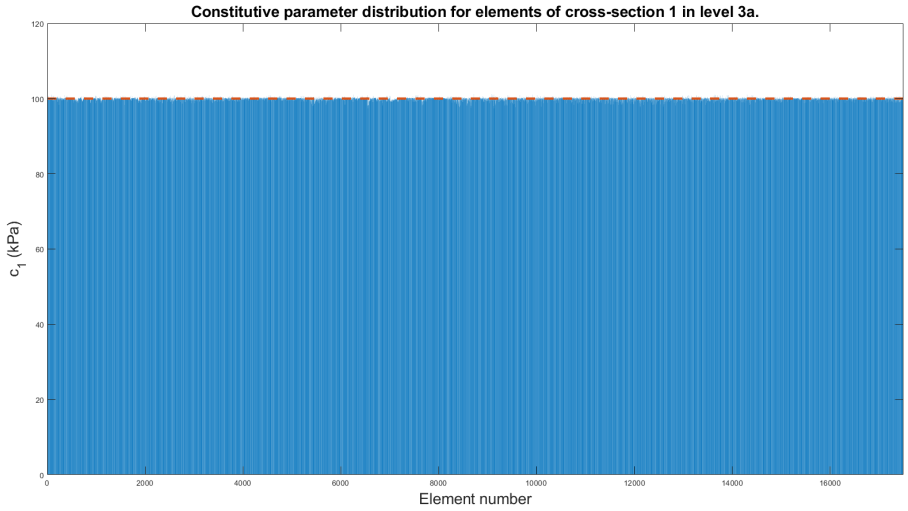

Figure 4: *This bar graph shows the calculated c_1 values of all the elements in cross-section 1 for the level 3a approach. As can be seen, the elements (horizontal axis) mostly hover around a constitutive parameter value (vertical axis) of 100 kPa. The red dashed line indicates the ground truth.*

3.4.2 Level 3b: heterogeneous segmentation-less approach

At the second sublevel, the number of constitutive parameters to be calculated was again equal to the number of elements. This time, however, the ground truth for the constitutive parameter of an element was dictated by the component that element belonged to (e.g. the ground truth value for the constitutive parameter of all the elements in the calcified tissue component was equal to 423 kPa). This has been illustrated in Figure 3c. As can be seen, the intima (red region) was divided into many elements. For each element in the intima component, a constitutive parameter was calculated, that has a different ground truth from, e.g., the constitutive parameters of the calcified tissue (green region). The red dashed line in Figure 5 shows the ground truth for this level. As can be seen, the elements are unified into four element groups, separated by the vertical black lines. The components have also been indicated in the figure. The ground truths for the elements in the intima component, wall component, lipid component, and calcified tissue component were set at 120 kPa, 250 kPa, 5 kPa, and 423 kPa, respectively. At this level, the approach did not allow for a priori knowledge of the grouping of the elements. As there is also no homogeneity in the cross-section, no useful initial guess was found for the minimization algorithm. Therefore, this had been set as a value between 0 and 1000 kPa, randomly assigned to each constitutive parameter.

Applying the VFM for this sublevel resulted in the bar graph shown in Figure 5. As can be seen, the results were not similar in any fashion to the desired result (red dashed line in the same figure), other than that the calculated c_1 values were all positive (which was imposed by the minimization algorithm). The Φ value was equal to 0.6128. This is very large, compared to the Φ value of level 3a (which was 0.0022).

It appears that the lipid component (elements 14430 to 17143) can be differentiated from the neighboring element sets (because of its comparatively low constitutive parameter values), however, the c_1 values differed greatly from the ground truth (5 kPa). The values ranged from 0.00 kPa to 657.28 kPa, with an average of 65.00 kPa. For this cross-section, it was clear that those elements made up a component, but the constitutive parameters that made up the component cannot be accurately calculated. This distinction of the lipid components is also seen in the other cross-section, except for cross-section 9 (see Appendix 7.4).

Comparing Figure 5 to the ground truth (red dashed line in the same figure), it may appear that the calcified tissue component could be discerned due to the constitutive parameters being distinctly higher than that of the lipid component. This, however, was solely due to the order of the components within this bar graph. If, for instance, the placing of the intima component and the calcified tissue component

was reversed (i.e. the calcified tissue component would have been next to the wall component), the calcified tissue component could not have been discerned from the wall component in Figure 5.

The constitutive parameter distributions of the cross-sections 2-5 showed similar results and are included in Appendix 7.4. Their Φ values were 0.5815, 0.6807, 0.5951, and 0.6999, for cross-section 2 - 5, respectively.

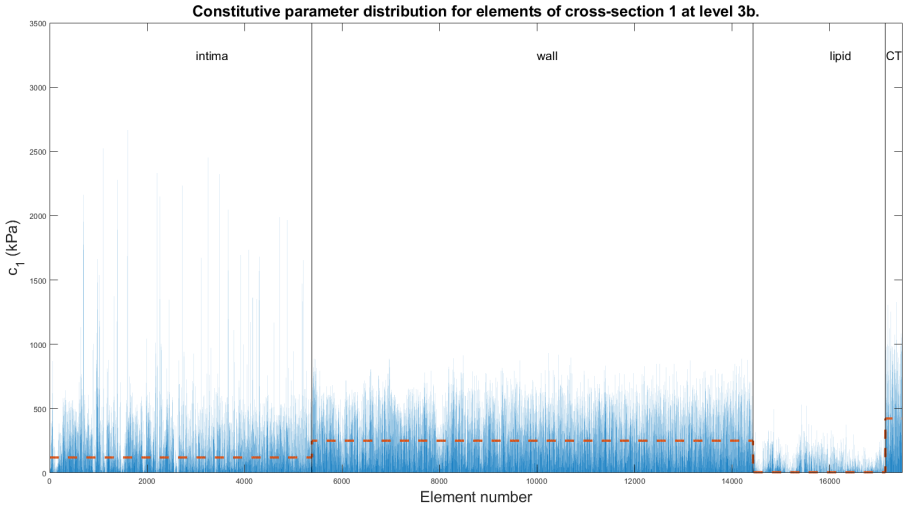

Figure 5: *This bar graph shows the calculated c_1 value of all the elements in cross-section 1 for the level 3b approach. As can be seen, the elements (horizontal axis) all had varying constitutive parameter values (vertical axis). It did not resemble the red dashed line, which was the ground truth. The black vertical lines delineate the figure into four groups of elements. Each group represents the elements in the corresponding component, e.g. the calcified tissue (CT) component can be found in the most-right group.*

4 Discussion

4.1 Discussion results level 1

For the first level, it is clear that the results were very close to the ground truth. At most the calculated constitutive parameter differed by 0.18 kPa. This is an error smaller than 0.2% compared to the ground truth. This accuracy means the VFM successfully retrieved the constitutive parameter value of the cross-sections.

4.2 Discussion results level 2

At this level, the results were fairly satisfactory, as most calculated c_1 values were within 20 kPa of the ground truth. Some components differed very much from the ground truth (> 75 kPa). For example, the calcified tissue of cross-section 1 differed 200.24 kPa from the ground truth.

Further analysis showed that if the k_n values for the VDFs (Equation (12)) were allowed to range from $1 < k_n < 60$, the results improved, as demonstrated in Table 3.

Table 3: *This table shows the calculated c_1 values for the level 2 approach, where the k_n value in the VDFs (Equation(12)) was allowed to range from $1 < k_n < 60$. The reported c_1 values are in unit of kPa. The ground truth (GT) is also included. For each cross-section, the Φ value is also included.*

	Intima	Wall	Lipid	Calcified tissue	Φ
GT	120	250	5	423	0
CS1	123.55	241.10	5.16	442.40	0.0394
CS2	120.49	250.47	4.95	421.59	0.0030
CS3	120.90	243.08	6.05	510.38	0.1093
CS4	120.05	251.80	5.32	418.51	0.0084
CS5	121.46	227.58	5.31	455.91	0.0705

As can be seen, almost all entries had values closer to the ground truth compared to the corresponding entries in Table 2. Some changed slightly, while others got more than 10% closer to the ground truth. Only a few differed greater from the ground truth than the corresponding entries in Table 2, but none of them significantly (< 10 kPa). This improvement was also seen in the Φ values. The Φ value of cross-section 3 increased by 0.0063, which is a small increase. The other Φ values all decreased by 0.2344, 0.0036, 0.0207, and 0.0818 for cross-section 1,2,4, and 5, respectively. This shows that the role the exponent plays is not insignificant. It is not entirely clear why this change of range resulted in (much) better results. This improvement implies further detailed investigation may shed some light on why this is the case and perhaps yield a strategy for choosing optimal exponents.

Furthermore, even though the constitutive parameters were much closer to the ground truth, the difference between the ground truth and the calculated constitutive parameter could still be large. For the calcified tissue component of cross-section 3 for instance, the difference between the ground truth and the calculated constitutive parameter was 87.38 kPa.

4.3 Discussion results level 3

4.3.1 Level 3a

For level 3a it was clear that the results have mostly been accurately calculated. As can be seen in the bar graph (Figure 4), the values were near 100 kPa. Figure 6 shows a histogram of all the calculated constitutive parameters. As can be seen, all constitutive parameters were within 4 kPa of the ground truth, indicated by the red dashed line.

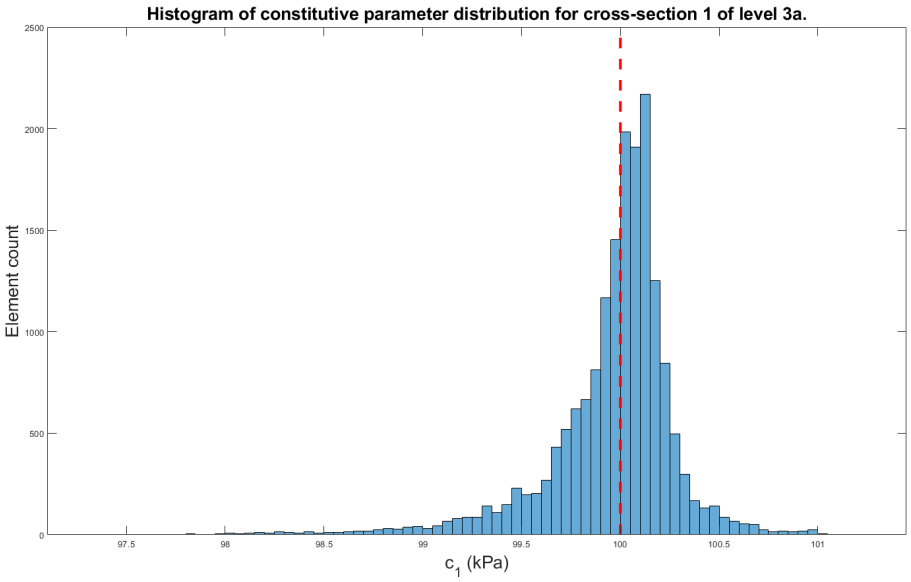

Figure 6: *Histogram of the calculated constitutive parameters for level 3a. Each bar represents a range of 0.05 kPa. All elements attained a value close to the ground truth of 100 kPa (indicated by the red dashed line). The range for the constitutive parameters is between 97.5 kPa and 101.5 kPa.*

The reason the constitutive parameters were not (exactly) 100 kPa is likely due to the difference between the calculated internal virtual work vector and external virtual work vector:

$$\underline{\Delta} = \underline{\underline{A}}\,\underline{c}_{gt} - \underline{b} \tag{25}$$

The minimization algorithm minimizes the cost function ξ (Equation (20)), which contains the internal virtual work vector and the external virtual work vector. If these two are not equal, there could be a constitutive parameter distribution with a lower ξ value than the ground truth. This can occur as the internal virtual work and external virtual work are approximated by sums, rather than calculated exactly through integrals (Equations (1) and (14)). Since the initial guess was reasonably close to the ground truth, the resulting calculated constitutive parameter distribution resembled the ground truth.

The role of the initial guess for the minimization algorithm is an important one and is demonstrated in Figure 7. This figure shows what the calculated constitutive parameter distribution would look if the initial guess for each element in the minimization algorithm was chosen at random between 0 and 1000 kPa, instead of using the knowledge of the homogeneous cross-section to find a good initial guess. As can be seen, the constitutive parameter distribution did not resemble the ground truth (red dashed line in Figure 7). In such a scenario, the Φ value was equal to 0.7255. This is a large increase compared to the scenario where a good initial guess was used. Then, the Φ value was equal to 0.0022.

For the discussion on level 3a, cross-section 1 is used as a representative case. The discussion points also hold for cross-sections 2, 4, and 5. For cross-section 3, the range in c_1 values was bigger (between 0.99 kPa and 278.3 kPa, see Appendix 7.3). This was most likely due to the condition number, the $\underline{\Delta}$, and the initial guess. The initial guess for the minimization algorithm was retrieved from Table 1. In Table 1, cross-section 3 has the biggest deviation from the ground truth. This bigger range is also reflected in the Φ values. These were 0.00220, $7.813 * 10^{-04}$, 0.0566, 0.00170, and 0.00180 for cross-sections 1-5, respectively. The Φ value of cross-section 3 is at least 25 times larger than that of the others, but it is still close to 0. This signifies that there are still lots of constitutive parameters close to the ground truth; 85% of the entries are within 10 kPa of the ground truth.

Furthermore, there is the condition number. The condition number is calculated by the ratio of the largest singular value to the lowest singular value of a second-order tensor. The singular values of any

second-order tensor $\underline{\underline{D}}$ can be found by taking the square root of the eigenvalues of the $\underline{\underline{D}}^T\underline{\underline{D}}$ tensor. The condition number indicates the sensitivity of a second-order tensor between the input and output. In other words, if the condition number is high, that means that a small change in the input can give a large change in the output. Here, the input was the initial guess. Since the condition number of tensor $\underline{\underline{A}}$ for cross-section 3 was $5.84 * 10^{18}$, this small deviation from the ground truth was enough to result in a larger range in the calculated constitutive parameters.

Lastly, there are $\underline{\Delta}$ and $\underline{\Delta}_{rel}$. Δ is the difference between internal and external virtual work vector (Equation (26)), and $\underline{\Delta}_{rel}$ is the difference between internal virtual work and external virtual work, relative to the external virtual work. In terms of subscripts:

$$\Delta_{rel,n} = \frac{\Delta_n}{b_n} \tag{26}$$

The n-th entry of $\underline{\Delta}_{rel}$ is given by the n-th entry of $\underline{\Delta}$ divided by the n-th entry of $\underline{b}$. $\underline{\Delta}$ and $\underline{\Delta}_{rel}$ were an order of magnitude larger than that of the other cross-sections (see Figure 25 in Appendix 7.5 and Figure 27 in Appendix 7.6). If the errors are larger, the initial guess less accurate, and the condition number is high, it is more difficult for the minimization algorithm to find a correct distribution than for the other cross-sections.

If the best value of Table 1 was used as an initial guess (which was 100.02 kPa, as it was closest to the ground truth), the range of the constitutive parameter distribution vector would be smaller (between 53.59 kPa and 148.64 kPa). The Φ value was then decreased to 0.0331.

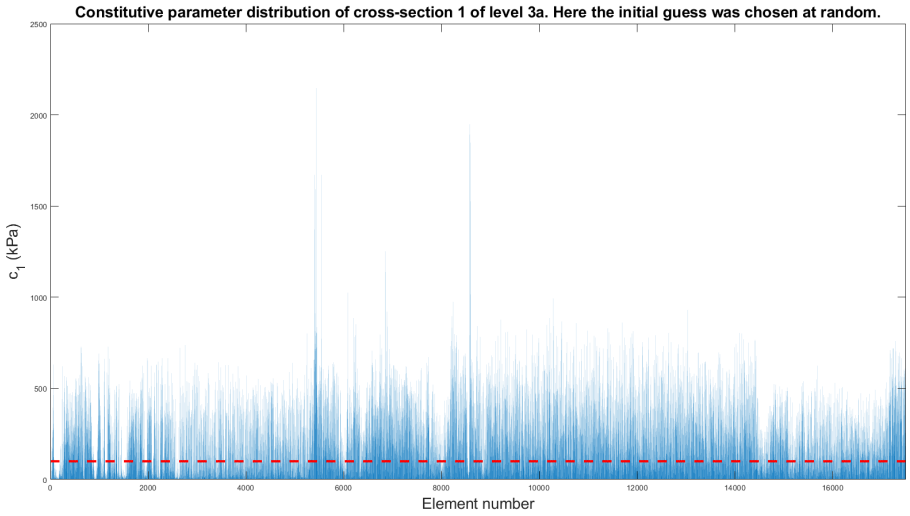

Figure 7: *This bar graph shows the calculated constitutive parameter distribution vector $\underline{c}$ distribution if the initial guess to the minimization algorithm was chosen at random between 0 and 1000 kPa for each constitutive parameter. As can be seen, the calculated c_1 values varied much and were not close to the ground truth value of 100 kPa. The ground truth is shown by the red dashed line.*

4.3.2 Level 3b

For level 3b, as can be seen in Figure 5, the results were not similar to the ground truth (red dashed line in the same figure). This is possibly due to three reasons: the difference between the internal virtual work vector and external virtual work vector, the rank, and the condition number. The discussion on level 3b is represented by cross-section 1, though it also holds for cross-sections 2-5.

$\underline{\Delta}$ *and* $\underline{\Delta}_{rel}$:

First, there is the difference between the internal virtual work vector and external virtual work vector $\underline{\Delta}$ (Equation (25)), and the relative difference between the internal virtual work vector and external virtual work vector $\underline{\Delta}_{rel}$(Equation (26)).

Ideally, each entry in $\underline{\Delta}$ (and thus $\underline{\Delta}_{rel}$) should be zero, but here they were not, due to the discretization of the integrals into sums (Equations (1) and (14)). For the constitutive parameter distribution calculated in Figure 5, the $\underline{\Delta}$ and $\underline{\Delta}_{rel}$ have been depicted in Figure 8. It shows the $\underline{\Delta}$ and $\underline{\Delta}_{rel}$ for each VDF used. As can be seen in the figure, both values were very low. Thus it can be expected that the $\underline{\Delta}$ and $\underline{\Delta}_{rel}$ did not play a significant role in the inaccurate calculation of the constitutive parameter distribution. Also, if a good initial guess is possible, level 3a showed that an accurate calculation of the constitutive parameter distribution is possible, despite having $\underline{\Delta}$ and $\underline{\Delta}_{rel}$ distribution in the same order as level 3b (see Appendix 7.5 for $\underline{\Delta}$ and $\underline{\Delta}_{rel}$ of level 3a).

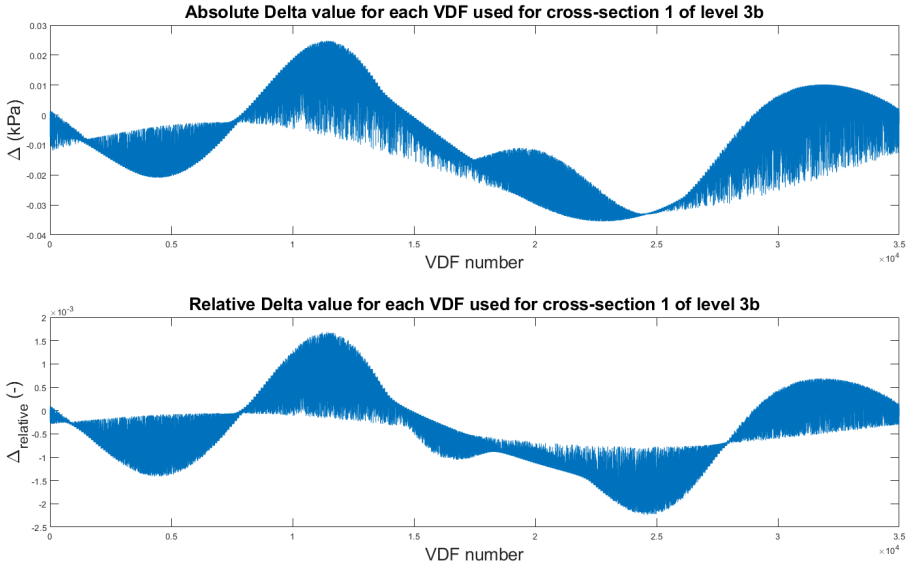

Figure 8: *This figure depicts* $\underline{\Delta}$ *and* $\underline{\Delta}_{rel}$ *for cross-section 1 at level 3b. The top plot depicts the difference vector* $\underline{\Delta}$ *(Equation* (25)*). As can be seen, the difference between the internal and external virtual work (vertical axis) is small for all VDFs (horizontal axis). The unit of* $\underline{\Delta}$ *is kPa, see Section 2.5 for the details of this. The bottom plot depicts the relative difference vector (Equation* (26)*). The relative difference (vertical axis) also is small for all VDFs (horizontal axis).*

Rank:

Next, the rank could explain the difference between $\underline{c}$ and $\underline{c}_{gt}$. The rank of a second-order tensor is the number of rows (or columns) in a second-order tensor that is independent of the other rows (or columns), i.e. the number of rows (or columns) that is not a linear combination of the other rows (or columns). In terms of the VFM, if tensor $\underline{\underline{A}}$ (Equation (17)) has rows that are a linear combination of other rows, this means that the one VDF is a linear combination of other VDFs.

For tensor $\underline{\underline{A}}$ that was used to create Figure 5, the rank was 1127, meaning $\underline{\underline{A}}$ tensor had (only) 1127 independent rows. As there are 17495 elements in this cross-section, full rank would have been attained at a rank of 17495.

It is not clear why tensor $\underline{\underline{A}}$ is rank-deficient. Looking at the VDFs (Equation (12)), it is clear that from one VDF to the next, the difference is the exponent value k_n. Since between two VDFs, the difference is in

the exponent, and not a factor, this (inherently) should dispel any notion of linear dependency. Furthermore, each entry within tensor $\underline{\underline{A}}$ is subjected to element-specific properties (the left Cauchy-Green deformation tensor entries, the derivative of the VDF are calculated at the center of an element, and lastly the area is also element-specific, see Equation (17)).

Condition number:

Other than the rank, the condition number may also have an impact on the calculation of the constitutive parameter distribution. The condition number is calculated by the ratio of the largest singular value to the lowest singular value of a second-order tensor and indicates the sensitivity of a second-order tensor between the input and output. The tensor $\underline{\underline{A}}$ that lead to the creation of Figure 5 had a condition number of $3.33 * 10^{18}$. Such a high condition number means that if the initial guess (input) is too far from the ground truth, the minimization algorithm was not able to find the ground truth (output). At level 3b, the initial guess for the constitutive parameters was chosen as random values between 0 and 1000 kPa for each constitutive parameter. This was not sufficiently close to the ground truth for the minimization algorithm to find the ground truth.

High condition number and good initial guess:

A high condition number and good initial guess were exemplified at level 3a, where tensor $\underline{\underline{A}}$ had a very high condition number, at $1.96 * 10^{18}$ (in the same order as at level 3b). This time, a sufficiently close initial guess resulted in a solution that is similar to the ground truth (red dashed line vs bars in Figure 4).

For level 3b, finding a decent initial guess requires at least knowledge of the grouping of the elements into components, such as at level 2. Then, just like at level 2, the elements could be grouped into their components, and within a component, a homogeneous constitutive parameter can be calculated. If for all components the (homogeneous) constitutive parameters are calculated, this then serves as an initial guess at level 3b for all elements belonging to their respective component. In essence, if the grouping of the elements into components is known, the procedure is to use the output of level 2 as an input for level 3b.

If the values found in Table 2 for cross-section 1 were used as initial guesses, then the resulting constitutive distribution is depicted in Figure 9. As can be seen, in such a case the calculated constitutive parameter distribution $\underline{c}$ showed the grouping of the elements (four distinct groups of elements can be identified by their c_1 value). The average constitutive parameter for each element group was calculated as 135.96 kPa, 205.57 kPa, 6.45 kPa, and 589.27 kPa for the intima component, wall component, lipid component, and calcified tissue component, respectively. It seemed that the elements did not change much relative to the initial guess. This was due to the condition number being very high. This caused many local minima in the cost function ξ for the minimization algorithm to find. This again shows that in the presence of a large condition number, an accurate initial guess is required. If, for instance, the initial guess was retrieved from cross-section 1 in Table 3, the calculated constitutive parameter distribution seemed more aligned with the ground truth, see Figure 10. Here, the average values were 123.49 kPa, 241.07 kPa, 5.19 kPa, and 441.19 kPa for the intima component, wall component, lipid component, and calcified tissue component, respectively. This was much closer to the ground truth (120 kPa, 250 kPa, 5 kPa, and 432 kPa respectively). This improvement was also reflected in the Φ value of each constitutive parameter distribution. For Figure 9, this value was 0.1956, while for Figure 10 this was 0.0353, which is a large reduction.

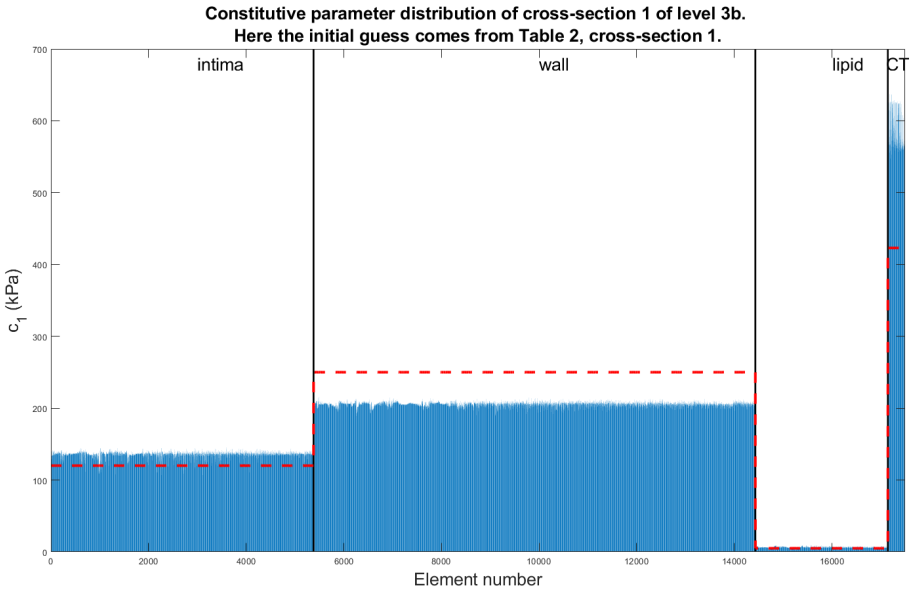

Figure 9: *This bar graph depicts the calculated constitutive parameter distribution for cross-section 1 (Figure 3), where the initial guess for the minimization was provided by Table 2 CS1. The red dashed line indicates the ground truth. The vertical lines divide the constitutive parameter distribution into four groups of elements. Each group of elements constitutes a component (intima, wall, lipid, or calcified tissue (CT)). These components have been indicated in the figure.*

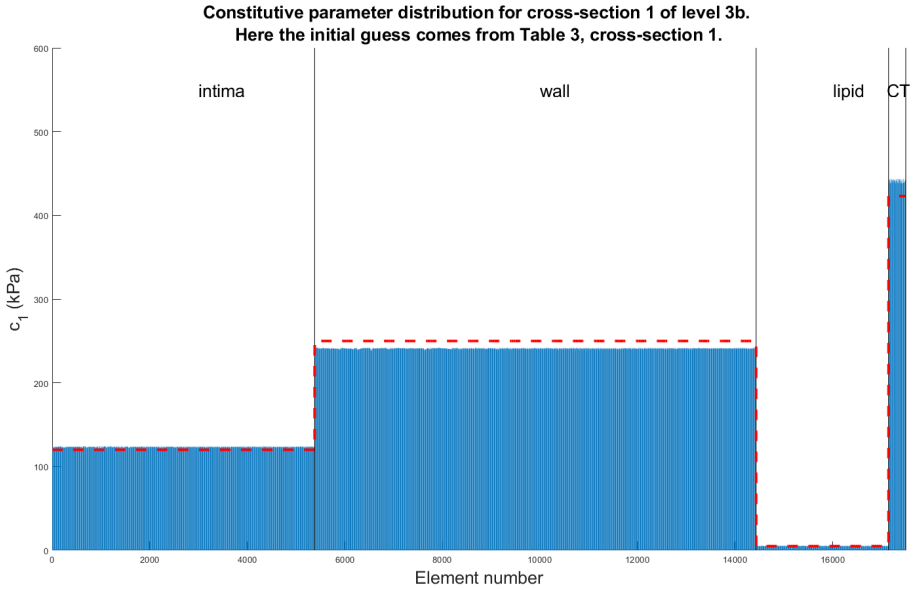

Figure 10: *This bar graph depicts the calculated constitutive parameter distribution for cross-section 1 (Figure 3), where the initial guess for the minimization was provided by Table 3 CS1. The red dashed line indicates the ground truth. The vertical lines divide the constitutive parameter distribution into four groups of elements. Each group of elements constitutes a component (intima, wall, lipid, or calcified tissue (CT)). These components have been indicated in the figure.*

Low condition number and random initial guess:

If the grouping of the elements into components is not known, the condition number must be lowered to get an accurate constitutive parameter distribution. To illustrate how a tensor $\underline{\underline{A}}$ with a low condition number would perform, a modified $\underline{\underline{A}}$ tensor is created:

$$\underline{\underline{A}}_m = \underline{\underline{A}}_m = \underline{\underline{A}} + \underline{\underline{I}} \tag{27}$$

where $\underline{\underline{I}}$ is the identity tensor (the identity tensor was appended with rows at the bottom containing zeros as tensor $\underline{\underline{A}}$ was not square). For cross-section 1, the modified tensor $\underline{\underline{A}}_m$ had a condition number of 2.59 and attained a rank of 17495. The condition number of $\underline{\underline{A}}_m$ was significantly lower than that of the original tensor $\underline{\underline{A}}$ (which was $3.33 * 10^{18}$). Since there are 17495 constitutive parameters to calculate, $\underline{\underline{A}}_m$ attained full rank.

Next, also the external virtual work vector $\underline{b}$ needed to be modified, as the modified internal virtual work vector $\underline{\underline{A}}_m \, \underline{c}_{gt}$ no longer was equal to the (original) external virtual work vector:

$$\underline{b}_m = \underline{\underline{A}}_m \, \underline{c}_{gt} \tag{28}$$

Using these modified tensor $\underline{\underline{A}}_m$ and the modified external virtual work vector $\underline{b}_m$ in the minimization algorithm lead to the constitutive parameter distribution depicted in Figure 11. As can be seen, the distribution calculated was very similar to the ground truth, even though the initial guess for each constitutive parameter was set at a random value between 0 and 1000 kPa. This was also represented by its Φ value (Equation (23)), which was equal to $4.60 * 10^{-9}$. This illustrates the importance of the condition number and rank.

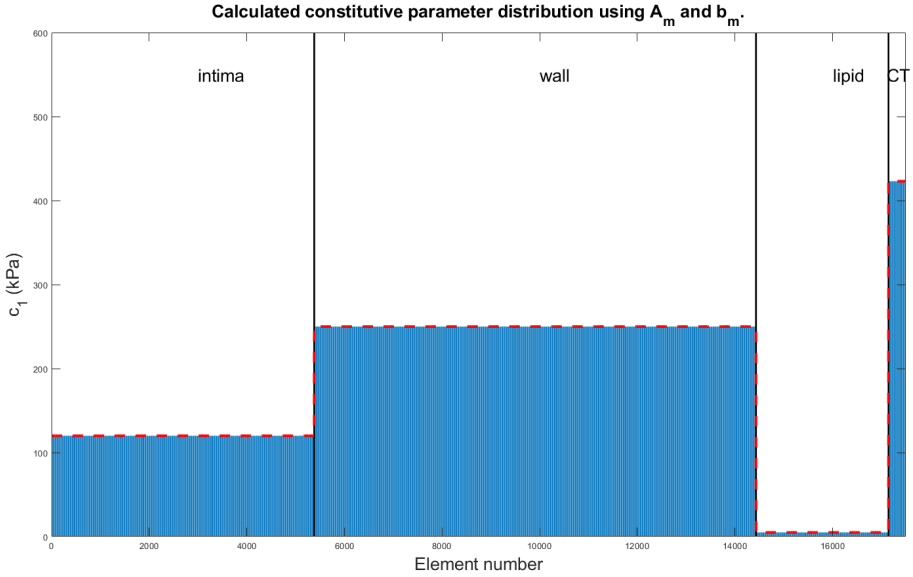

Figure 11: *This bar graph shows the constitutive parameter distribution if modified tensor $\underline{\underline{A}}_m$ and modified external virtual work vector $\underline{b}_m$ are used. The initial guess for each c_1 parameter in the minimization algorithm was set as a random value between 0 and 1000 kPa. The ground truth for such an approximate element size is represented by the red dashed line. As can be seen, the calculated constitutive parameters and the ground truth overlap much. The vertical lines divide the constitutive parameter distribution into 4 groups of elements. Each group of elements constitutes a component (intima, wall, lipid, or calcified tissue (CT)). These components have been indicated in the figure.*

The reason the condition number is drastically lowered by adding the identity tensor has to do with the definition of the condition number and the entries of the $\underline{\underline{A}}$ tensor. In the case of the $\underline{\underline{A}}$ tensor that

created Figure 5, the $\underline{\underline{A}}$ tensor had a size of 35000 x 17495. This means there are 35000x17495 $= 6.12 * 10^8$ entries. The vast majority of these entries (95.8%) had values between $-0.5 * 10^{-4}$ and $1.0 * 10^{-4}$, while the maximum absolute entry was $1.4 * 10^{-3}$. This showed that the entries of tensor $\underline{\underline{A}}$ were all small. If an identity tensor (with relatively large entries in the main diagonal) was added to such a tensor, the other entries were 'overshadowed' in their influence on the singular values. The singular values were then mostly dictated by the identity tensor, and by extension also the condition number. As the singular values of an identity tensor are all equal to 1, the modified tensor $\underline{\underline{A}}_m$ was expected to have a condition number close to 1. In a mathematical sense, the singular values of tensor $\underline{\underline{A}}_m$ is calculated by taking the square root of the eigenvalues of tensor $\underline{\underline{A}}_m \underline{\underline{A}}_m^T$. If tensor $\underline{\underline{A}}_m \underline{\underline{A}}_m^T$ mostly has small values on the non-main diagonal entries, and 1 plus a small value on the main diagonal, this is then approximately a square identity tensor, with singular values close to 1. The largest singular value divided by the smallest singular value (which is equal to the condition number) should then be close to 1.

Rank vs. condition number:

It was already shown that the influence of $\underline{\Delta}$ (Equation (25)) and $\underline{\Delta}_{rel}$ (Equation (26)) did not play a large role in the accurate calculation of the constitutive parameters due to their magnitude. It is then interesting to see which of the remaining two, rank or condition number, is more important. It should be noted that in the presence of a good initial guess, neither are necessary, as was shown in Figure 4. For the rest of this discussion, it was assumed the initial guess for each constitutive parameter was chosen at random between a value of 0 and 1000 kPa.

To investigate the influence of condition number and rank, two scenarios were to be tested: one where the condition number was high and the rank was full, and one where the condition number was low, while the rank was not full.

A method was found to deliver the first scenario. By including many more VDFs than unknowns, full rank was achieved. The tensor $\underline{\underline{A}}$ where full rank was achieved in this manner was called $\underline{\underline{A}}_{fr}$ and the corresponding external virtual work vector was called $\underline{b}_{fr}$. This was unfeasible when the elements in a cross-section had an approximate edge length of 50 μm as it was too computationally expensive. Changing the edge length to an approximate size of 400 micrometers drastically reduced the computational cost and allowed the investigation of this method. For cross-section 1, this translated to a cross-section with 420 elements. Using $N = 1 * 10^5$ VDFs, full rank was reached, thus tensor $\underline{\underline{A}}_{fr}$ had a rank of 420 after applying this method. Afterwards, the rows that spanned this very large tensor $\underline{\underline{A}}_{fr}$ and the corresponding rows in the external virtual work vector $\underline{b}_{fr}$ components were selected to create a new tensor $\underline{\underline{A}}_{fr2}$ and external virtual work vector $\underline{b}_{fr2}$. This resulted in a second-order tensor of size 420x420 with rank 420 and condition number $4.04 * 10^{11}$, and a vector of size 420x1.

Figure 12 depicts the calculated constitutive parameter distribution when tensor $\underline{\underline{A}}_{fr2}$ and virtual external work vector $\underline{b}_{fr2}$ were used in the minimization algorithm. The initial guess for each constitutive parameter was set at a random value between 0 and 1000 kPa. The ground truth is depicted in the same figure as a red dashed line. As can be seen, even though a full rank was achieved, the calculated constitutive parameter distribution did not resemble the ground truth. This is due to the large condition number. If the condition number is large, there are lots of local minima in the cost function ξ. The minimization algorithm can get 'stuck' in such a local minimum. This leads to the conclusion that in the absence of a good initial guess, full rank by itself is not enough for accurate constitutive parameter calculation.

In the second scenario, the condition number is to be low while the $\underline{\underline{A}}$ should be rank-deficient. No method was found that allowed for this. This was probably because the two properties are mutually exclusive: if the condition number is low, it implies full rank for tensor $\underline{\underline{A}}$. This could be explained as follows: If any second-order tensor $\underline{\underline{D}}$ is rank-deficient, the second-order tensor $\underline{\underline{E}} = \underline{\underline{D}}^T \underline{\underline{D}}$ is also rank-deficient. This then means that some eigenvalues λ are zero. The condition number is defined as:

$$cond(E) = \frac{\sqrt{\lambda_{max}}}{\sqrt{\lambda_{min}}} \tag{29}$$

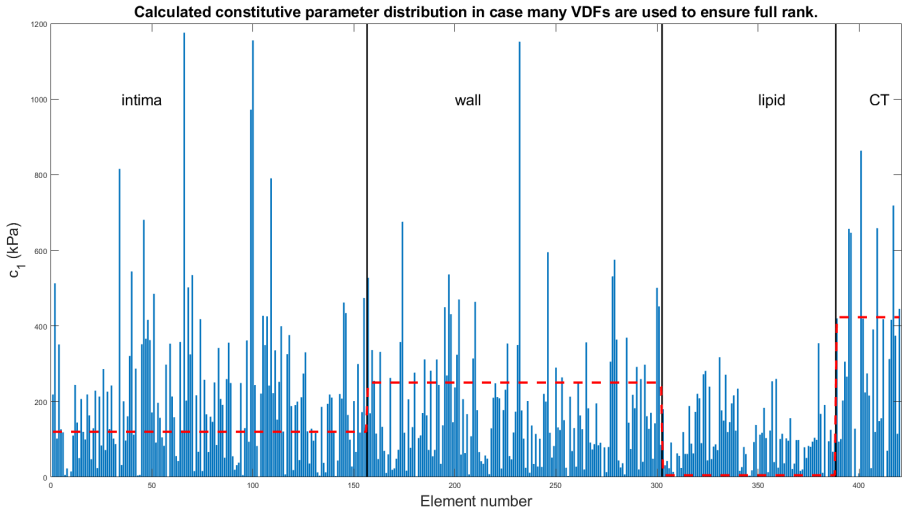

Figure 12: *This bar graph shows the constitutive parameter (vertical axis) distribution for elements (horizontal axis) of cross-section 1 if many more VDFs are used compared to the number of unknowns. For this cross-section, the approximate element size was set at 400 μm. The ground truth for such an approximate element size is represented by the red dashed line. The vertical lines divide the constitutive parameter distribution into four groups of elements. Each group of elements constitutes a component (intima, wall, lipid, or calcified tissue (CT)). These components have been indicated in the figure.*

As λ_{min} goes to zero, $cond(E)$ goes to infinity. It is therefore not possible to have a low condition number while the rank is deficient.

From these two scenarios and the fact that the best $\underline{c}$ distribution (based on Φ values) was calculated when the condition number was low (condition number of $\underline{\underline{A}}_m$ was 2.59), it is concluded that a low condition number is required for accurate constitutive parameter calculations (in absence of a good initial guess).

5 Recommendations

For level 3b, based on the successful calculation of the constitutive parameters using tensor $\underline{\underline{A}}_m$ and vector $\underline{b}_m$, it can be helpful to engineer tensor $\underline{\underline{A}}$ such that the main diagonal contains relatively large values. This would create a tensor that resembles an identity tensor, and thus ensure a low condition number.

The entries in tensor $\underline{\underline{A}}$ are made up of the left Cauchy-Green tensor $\underline{\underline{B}}$, the area a, and the Jacobian of the VDF $\frac{\partial \underline{u}^*}{\partial \underline{x}}$. Within a column in tensor $\underline{\underline{A}}$, the former two are the same for each row, and the only term that differs is the latter. Thus, within a column, the only way to influence the values in the rows is by influencing $\frac{\partial \underline{u}^*}{\partial \underline{x}}$. Taking a closer look on the first column of tensor $\underline{\underline{A}}$ (which is evaluated at the center of the first element), it must mean that $B_{11}^1 \frac{\partial u_1^{1*}}{\partial x} + 2B_{12}^1 \frac{\partial u_1^{1*}}{\partial y} + B_{22}^1 \frac{\partial u_2^{1*}}{\partial y}$ is large compared to $B_{11}^1 \frac{\partial u_1^{n*}}{\partial x} + 2B_{12}^1 \frac{\partial u_1^{n*}}{\partial y} + B_{22}^1 \frac{\partial u_2^{n*}}{\partial y}$ for $n > 1$. This could be achieved by making sure the Jacobian of the first VDF has got high values at the center of element 1, while the other VDFs have got low values for their Jacobian at the center of element 1. Note that the first VDF is also applied to the other elements in the first row of tensor $\underline{\underline{A}}$. The first VDF must thus be such that the first column in the first row has a relatively high value compared to the other columns in the first row. Basically, VDF q should have high Jacobian values at element w for $q = w$, and it should have low Jacobian values at elements $q \neq w$.

One of the requirements for the VDFs was that the trace of the Jacobian of the VDFs should be zero. In the entries of tensor $\underline{\underline{A}}$, the two terms in that trace are multiplied by left Cauchy-Green tensor components: $B_{11} \frac{\partial u_1^*}{\partial x} + B_{22} \frac{\partial u_2^*}{\partial y}$. B_{11} and B_{22} are often close to one another in value, thus their total contribution often is small. Since these left Cauchy-Green tensor components are more complex to control, it could be beneficial to focus on the term $\frac{\partial u_2^{1*}}{\partial y}$ to control the entries in tensor $\underline{\underline{A}}$. One method to influence the entry could then be to engineer $\frac{\partial u_2^{1*}}{\partial y}$ such that it would then have relatively high values compared to the other components in the main diagonal of tensor $\underline{\underline{A}}$.

From the discussion for level 2, it is apparent that the exponent k_n plays a significant role in the calculation of accurate constitutive parameters. In the discussion, going from $1 < k_n < 10$ to $1 < k_n < 60$ changes the constitutive parameters of level 2 drastically. Understanding and predicting which exponent range leads to more accurate results could greatly increase the accuracy of the constitutive parameter calculation.

The current constitutive model should be considered as a starting point. It is not accurate to describe heterogeneous, anisotropic atherosclerotic plaques with an isotropic constitutive model. Thus, it is recommended to expand to other constitutive models that are more representative of atherosclerotic plaques. A Holzapfel constitutive model or a Fung constitutive model could be more accurate, as these allow for anisotropy. These have previously been applied to murine ascending aorta [7, 8, 9] and a human ascending aorta [10].

An expansion of the current study is to check for heterogeneity within a component. At level 3b, each element within a component had the freedom to attain its own constitutive parameter. However, the ground truth of all elements within a component was the same. In reality, this is not accurate. Any component has heterogeneity within the component, as it is structurally not homogeneous. Therefore, having heterogeneity within a component would be one step closer to actual atherosclerotic plaques. This could be an interesting level 3c. This would making a good initial guess more challenging.

6 Conclusion

This study was focused on the characterization of constitutive parameters for two-dimensional FE atherosclerotic plaques using the VFM. The goal was to expand from a homogeneous approach to a segmentation-less element-wise approach. It was shown how up until the segmentation-less approach the VFM can be (very) successful in calculating the constitutive parameters. As soon as the deformed nodes are used to create elements, of which the constitutive parameters are calculated in a heterogeneous setting (level 3b), the application of the VFM is not accurate in calculating the constitutive parameters. In absence of a good initial guess, tensor $\underline{\underline{A}}$ (Equation (17)) must have a lowered condition number for this method to achieve acceptable results.

Presently, there are still some hurdles to overcome before the VFM could be used in clinical applications. While finding a method that could lower the condition number of tensor $\underline{\underline{A}}$ for a heterogeneous segmentation-less approach (level 3b) would be a major step forward, the current models (geometry, constitutive model, VFM) need to be expanded to more accurately represent atherosclerotic plaques. These more accurate models would allow for calculations of constitutive parameters that better represent the atherosclerotic plaques, with which more accurate stress distribution can be found. Once the current hurdles are overcome, and the models better represent the atherosclerotic plaques, the next step would then be to use experimental deformation data, rather than synthetically created in a FE environment. The present study, however, provides a foundation upon which to build.

References

[1] Ali Akyildiz. Biomechanical modeling of atherosclerotic plaques for risk assessment. 2013.

[2] Harm Nieuwstadt. Mri-based biomechanical modeling of carotid atherosclerotic plaques. 2015.

[3] Jacob Fog Bentzon, Fumiyuki Otsuka, Renu Virmani, and Erling Falk. Mechanisms of plaque formation and rupture. Circulation research, 114(12):1852–1866, 2014.

[4] D Valdez-Jasso, Harvey Thomas Banks, Mansoor A Haider, D Bia, Y Zocalo, RL Armentano, and Mette S Olufsen. Viscoelastic models for passive arterial wall dynamics. Advances in Applied Mathematics and Mechanics, 1(2):151–165, 2009.

[5] Liang Zhang, Sri Gowtham Thakku, Meghna R Beotra, Mani Baskaran, Tin Aung, James CH Goh, Nicholas G Strouthidis, and Michael JA Girard. Verification of a virtual fields method to extract the mechanical properties of human optic nerve head tissues in vivo. Biomechanics and modeling in mechanobiology, 16(3):871–887, 2017.

[6] Fabrice Pierron and Michel Grédiac. The virtual fields method: extracting constitutive mechanical parameters from full-field deformation measurements. Springer Science & Business Media, 2012.

[7] Matthew R Bersi, Chiara Bellini, Paolo Di Achille, Jay D Humphrey, Katia Genovese, and Stéphane Avril. Novel methodology for characterizing regional variations in the material properties of murine aortas. Journal of biomechanical engineering, 138(7), 2016.

[8] Matthew R Bersi, Chiara Bellini, Jay D Humphrey, and Stéphane Avril. Local variations in material and structural properties characterize murine thoracic aortic aneurysm mechanics. Biomechanics and modeling in mechanobiology, 18(1):203–218, 2019.

[9] Matthew R Bersi, Víctor A Acosta Santamaría, Karl Marback, Paolo Di Achille, Evan H Phillips, Craig J Goergen, Jay D Humphrey, and Stéphane Avril. Multimodality imaging-based characterization of regional material properties in a murine model of aortic dissection. Scientific reports, 10(1):1–23, 2020.

[10] Stéphane Avril, Pierre Badel, and Ambroise Duprey. Anisotropic and hyperelastic identification of in vitro human arteries from full-field optical measurements. Journal of biomechanics, 43(15):2978–2985, 2010.

[11] Jin-Hwan Kim, Stéphane Avril, Ambroise Duprey, and Jean-Pierre Favre. Experimental characterization of rupture in human aortic aneurysms using a full-field measurement technique. Biomechanics and modeling in mechanobiology, 11(6):841–853, 2012.

[12] Solmaz Farzaneh, Olfa Trabelsi, and Stéphane Avril. Inverse identification of local stiffness across ascending thoracic aortic aneurysms. Biomechanics and modeling in mechanobiology, 18(1):137–153, 2019.

[13] Felipe Pires, Stéphane Avril, Julio Cordioli, Steve Vanlanduit, and Joris Dirckx. Local stiffness estimation of the human eardrum via the virtual fields method. In International Symposium on Computer Methods in Biomechanics and Biomedical Engineering, pages 248–255. Springer, 2019.

[14] Liang Zhang, Meghna R Beotra, Mani Baskaran, Tin A Tun, Xiaofei Wang, Shamira A Perera, Nicholas G Strouthidis, Tin Aung, Craig Boote, and Michael JA Girard. In vivo measurements of prelamina and lamina cribrosa biomechanical properties in humans. Investigative ophthalmology & visual science, 61(3):27–27, 2020.

[15] Renee Miller, Arunark Kolipaka, Martyn P Nash, and Alistair A Young. Estimation of transversely isotropic material properties from magnetic resonance elastography using the optimised virtual fields method. International journal for numerical methods in biomedical engineering, 34(6):e2979, 2018.

[16] Umar Sadat, Zhongzhao Teng, and Jonathan H Gillard. Biomechanical structural stresses of atherosclerotic plaques. Expert review of cardiovascular therapy, 8(10):1469–1481, 2010.

[17] Gerhard A Holzapfel, John J Mulvihill, Eoghan M Cunnane, and Michael T Walsh. Computational approaches for analyzing the mechanics of atherosclerotic plaques: a review. Journal of biomechanics, 47(4):859–869, 2014.

[18] Ronald van den Berg. A novel approach to estimating the material properties of atherosclerotic plaque tissue. Master's thesis, TU Delft, 2019.

[19] Donna M Ebenstein, Dezba Coughlin, Joan Chapman, Cheng Li, and Lisa A Pruitt. Nanomechanical properties of calcification, fibrous tissue, and hematoma from atherosclerotic plaques. Journal of Biomedical Materials Research Part A: An Official Journal of The Society for Biomaterials, The Japanese Society for Biomaterials, and The Australian Society for Biomaterials and the Korean Society for Biomaterials, 91(4):1028–1037, 2009.

[20] HA Nieuwstadt, S Fekkes, HHG Hansen, CL de Korte, Aad van der Lugt, JJ Wentzel, AFW van der Steen, and FJH Gijsen. Carotid plaque elasticity estimation using ultrasound elastography, mri, and inverse fea–a numerical feasibility study. Medical Engineering & Physics, 37(8):801–807, 2015.

7 Appendix

7.1 Histology, schematic, and FE model cross-sections

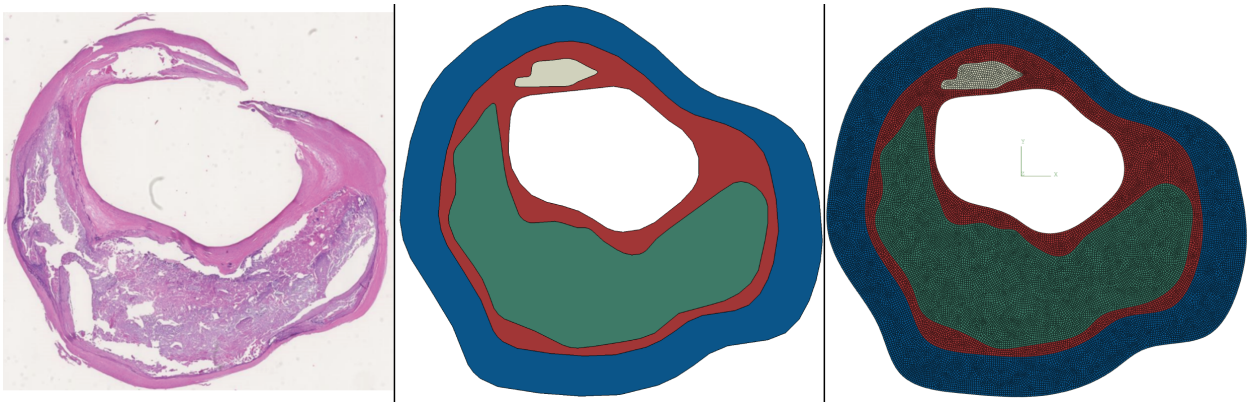

Figure 13: *Left: Stained histology image of an atherosclerotic artery cross-section 2. From these cross-sections, schematic and finite element models are created. Middle: Schematic image of the components of an atherosclerotic artery cross-section. The delineated regions all indicate a component. Right: FE model of an atherosclerotic artery. The model was meshed with 28509 four-noded quadrilateral elements (black lines). The components are color-coded. The calcified tissue component can be seen in green, the wall component can be seen in blue, the lipid component is seen in gray, and the intima component can be seen in red. This image also shows the origin of the coordinate system.*

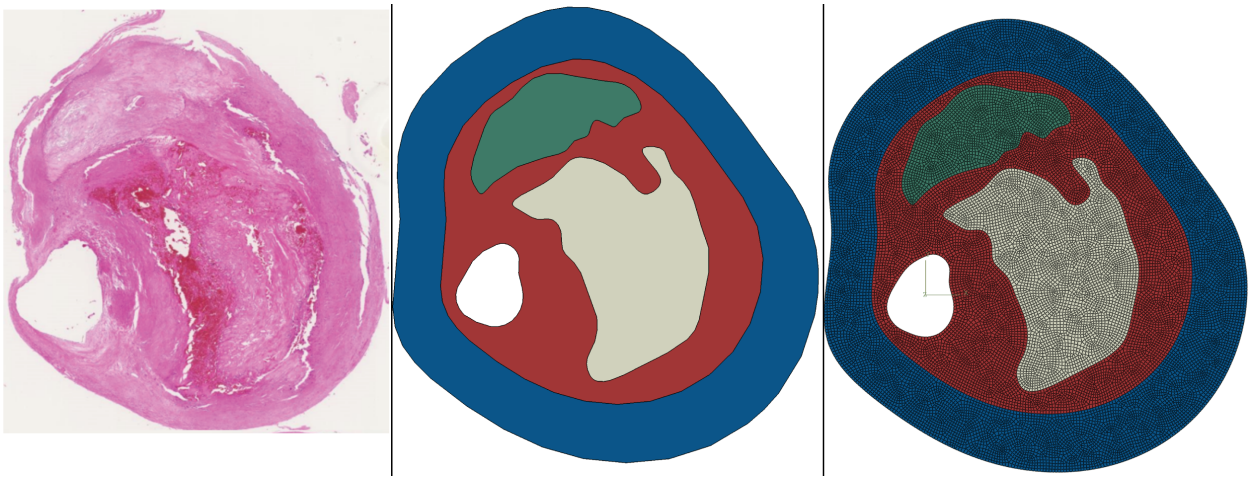

Figure 14: *Left: Stained histology image of an atherosclerotic artery cross-section 3. From these cross-sections, schematic and finite element models are created. Middle: Schematic image of the components of an atherosclerotic artery cross-section. The delineated regions all indicate a component. Right: FE model of an atherosclerotic artery. The model was meshed with 20381 four-noded quadrilateral elements (black lines). The components are color-coded. The calcified tissue component can be seen in green, the wall component can be seen in blue, the lipid component is seen in gray, and the intima component can be seen in red. This image also shows the origin of the coordinate system.*

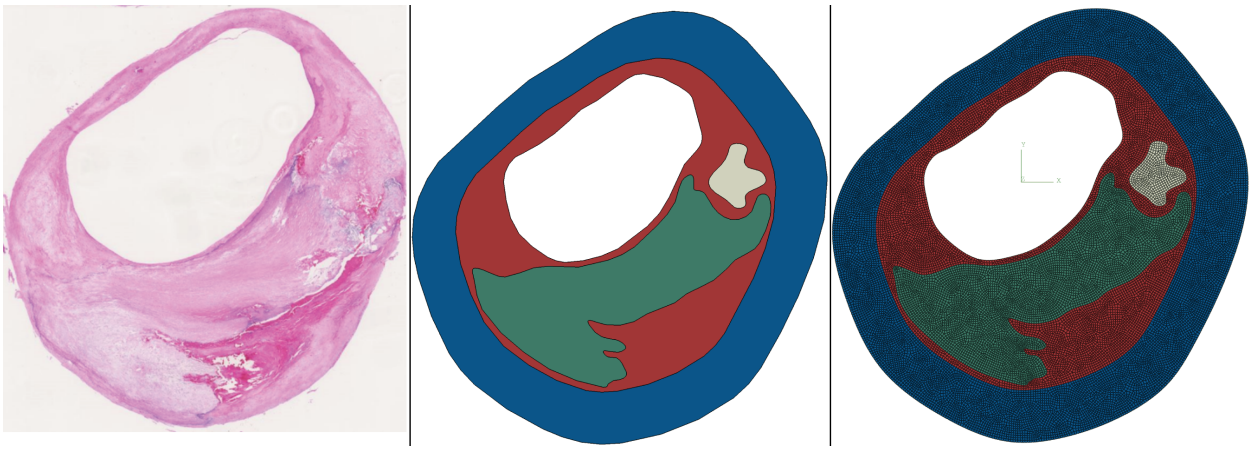

Figure 15: *Left: Stained histology image of an atherosclerotic artery cross-section 4. From these cross-sections, schematic and finite element models are created. Middle: Schematic image of the components of an atherosclerotic artery cross-section. The delineated regions all indicate a component. Right: FE model of an atherosclerotic artery. The model was meshed with 22952 four-noded quadrilateral elements (black lines). The components are color-coded. The calcified tissue component can be seen in green, the wall component can be seen in blue, the lipid component is seen in gray, and the intima component can be seen in red. This image also shows the origin of the coordinate system.*

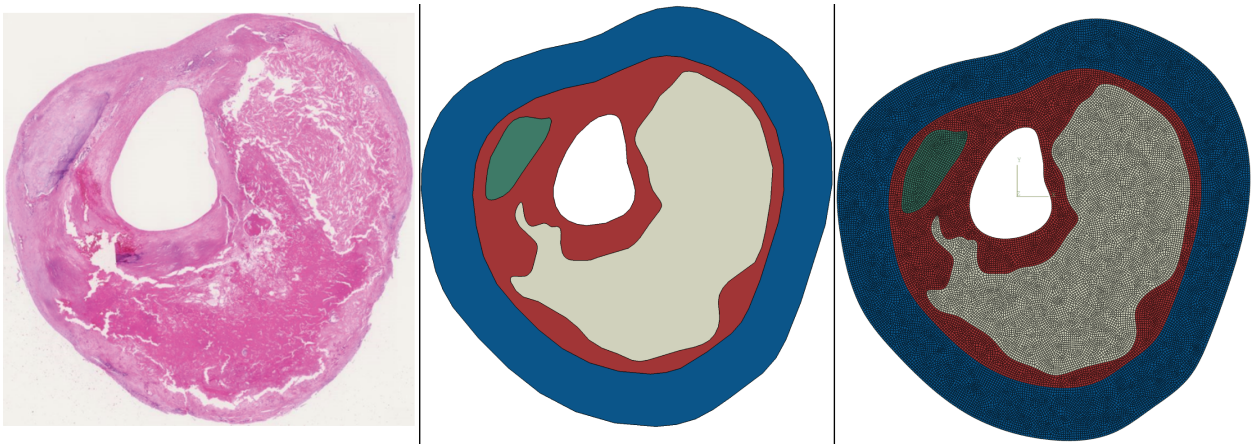

Figure 16: *Left: Stained histology image of an atherosclerotic artery cross-section 5. From these cross-sections, schematic and finite element models are created. Middle: Schematic image of the components of an atherosclerotic artery cross-section. The delineated regions all indicate a component. Right: FE model of an atherosclerotic artery. The model was meshed with 30770 four-noded quadrilateral elements (black lines). The components are color-coded. The calcified tissue component can be seen in green, the wall component can be seen in blue, the lipid component is seen in gray, and the intima component can be seen in red. This image also shows the origin of the coordinate system.*

7.2 Proof trace of Jacobian of the VDFs used is equal to zero.

This section shows how the trace of the Jacobian of the VDFs used is equal to zero. The VDFs are written as:

$$\underline{u}^{n*} = \begin{bmatrix} u_1^{n*} \\ u_2^{n*} \end{bmatrix} = \begin{bmatrix} \frac{x^{2k_n+1}}{(x^2+y^2)^{k_n+1}} \\ \\ \frac{yx^{2k_n}}{(x^2+y^2)^{k_n+1}} \end{bmatrix} \tag{30}$$

Next, the Jacobian of these VDFs is equal to:

$$\frac{\partial \underline{u}^{n*}}{\partial \underline{x}} = \begin{bmatrix} \frac{\partial u_1^{n*}}{\partial x} & \frac{\partial u_1^{n*}}{\partial y} \\ \\ \frac{\partial u_2^{n*}}{\partial x} & \frac{\partial u_2^{n*}}{\partial y} \end{bmatrix} \tag{31}$$

The trace of this tensor simply reads:

$$tr(\frac{\partial \underline{u}^{n*}}{\partial \underline{x}}) = \frac{\partial u_1^{n*}}{\partial x} + \frac{\partial u_1^{n*}}{\partial y} \tag{32}$$

The first partial derivative gives:

$$\frac{\partial u_1^{n*}}{\partial x} = \frac{\partial \frac{x^{2k_n+1}}{(x^2+y^2)^{k_n+1}}}{\partial x} = \frac{x^{2k_n}((2k_n+1)y^2 - x^2)}{(x^2+y^2)^{k_n+2}} \tag{33}$$

The second partial derivative

$$\frac{\partial u_2^{n*}}{\partial y} = \frac{\partial \frac{yx^{2k_n}}{(x^2+y^2)^{k_n+1}}}{\partial y} = -\frac{x^{2k_n}((2k_n+1)y^2 - x^2)}{(x^2+y^2)^{k_n+2}} \tag{34}$$

This shows how

$$\frac{\partial u_1^{n*}}{\partial x} = -\frac{\partial u_2^{n*}}{\partial y} \tag{35}$$

and thus showing that the trace of the Jacobian of the VDFs must be equal to zero, for any x, y, k_n, as long as $x^2 + y^2 \neq 0$. This has been ensured by placing the origin of the coordinate system in the lumen.

7.3 Level 3a results Cross-section 2-5

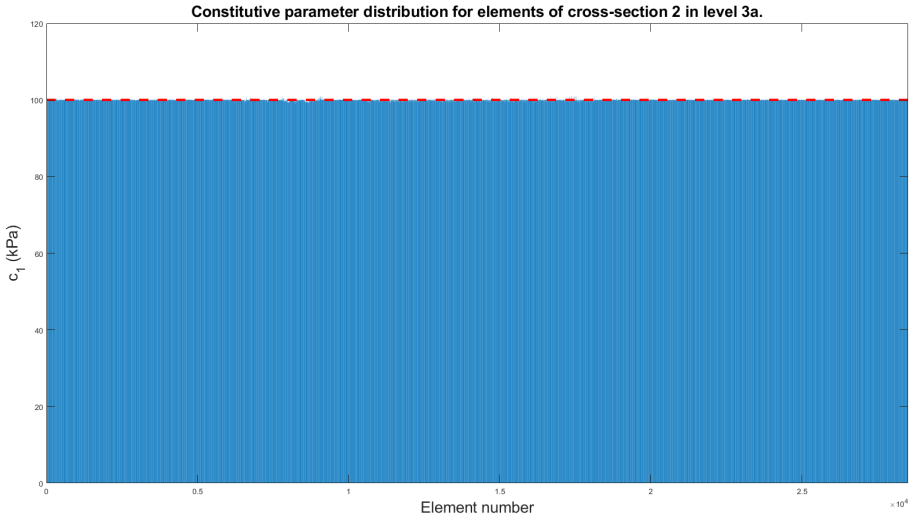

Figure 17: *This bar graph shows the calculated c_1 values of all elements in cross-section 2 for the level 3a approach. As can be seen, the elements (horizontal axis) mostly hover around a constitutive parameter value (vertical axis) of 100 kPa. The red dashed line indicates the ground truth.*

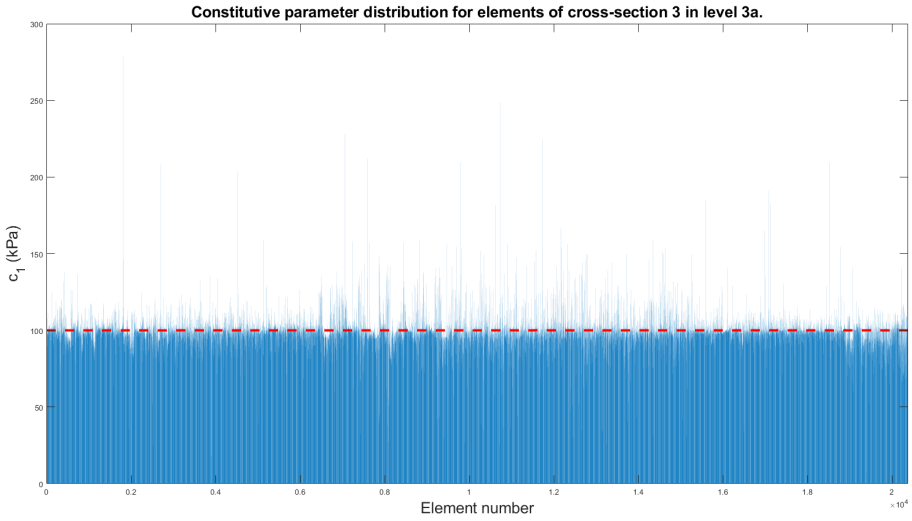

Figure 18: *This bar graph shows the calculated c_1 values of all elements in cross-section 3 for the level 3a approach. As can be seen, the elements (horizontal axis) sometimes have a larger or smaller constitutive parameter value (vertical axis) than the ground truth (red dashed line).*

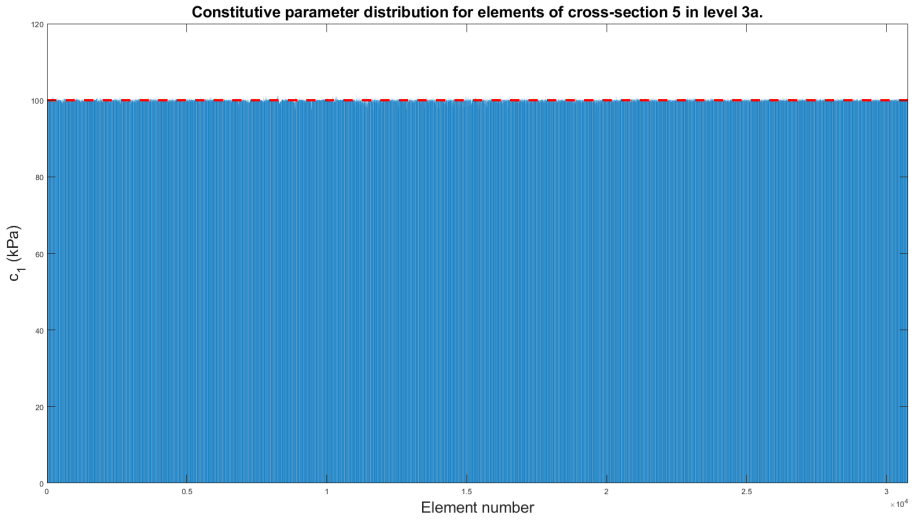

Figure 19: *This bar graph shows the calculated c_1 values of all elements in cross-section 4 for the level 3a approach. As can be seen, the elements (horizontal axis) mostly hover around a constitutive parameter value (vertical axis) of 100 kPa. The red dashed line indicates the ground truth.*

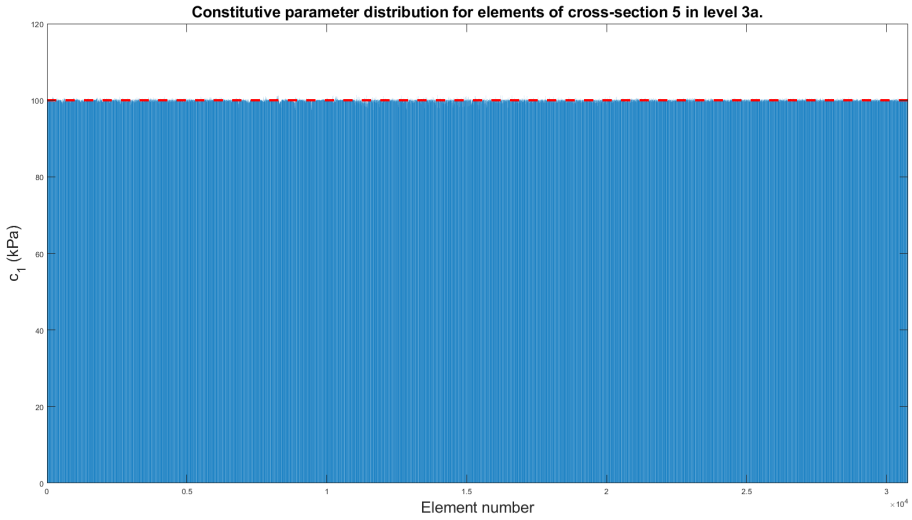

Figure 20: *This bar graph shows the calculated c_1 values of all elements in cross-section 5 for the level 3a approach. As can be seen, the elements (horizontal axis) mostly hover around a constitutive parameter value (vertical axis) of 100 kPa. The red dashed line indicates the ground truth.*

7.4 Level 3b results Cross-section 2-5

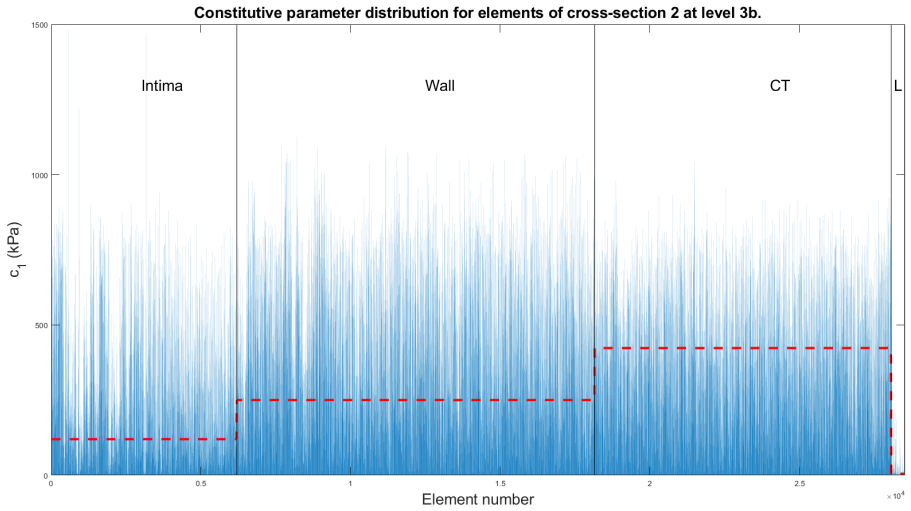

Figure 21: *This bar graph shows the calculated c_1 values of all elements in cross-section 2 for the level 3a approach. As can be seen, the elements (horizontal axis) all have varying constitutive parameter values (vertical axis). It does not resemble the red dashed line in any way, which is the ground truth. The black vertical lines delineate the figure into four groups of elements. Each group represents the elements in the corresponding component, e.g. the calcified tissue (CT) component can be found in the second-from-right group, and the lipid (L) component can be found in the most-right group.*

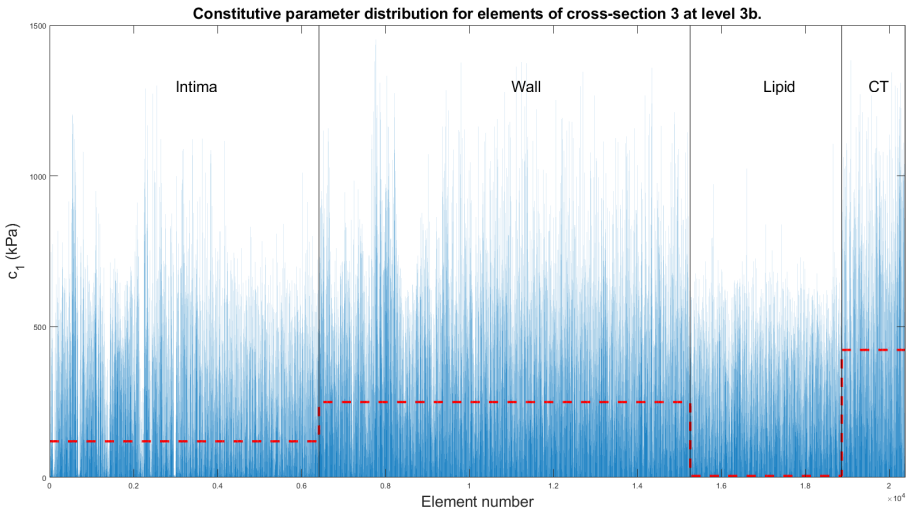

Figure 22: *This bar graph shows the calculated c_1 values of all elements in cross-section 3 for the level 3a approach. As can be seen, the elements (horizontal axis) all have varying constitutive parameter values (vertical axis). It does not resemble the red dashed line in any way, which is the ground truth. The black vertical lines delineate the figure into four groups of elements. Each group represents the elements in the corresponding component, e.g. the calcified tissue (CT) component can be found in the most-right group.*

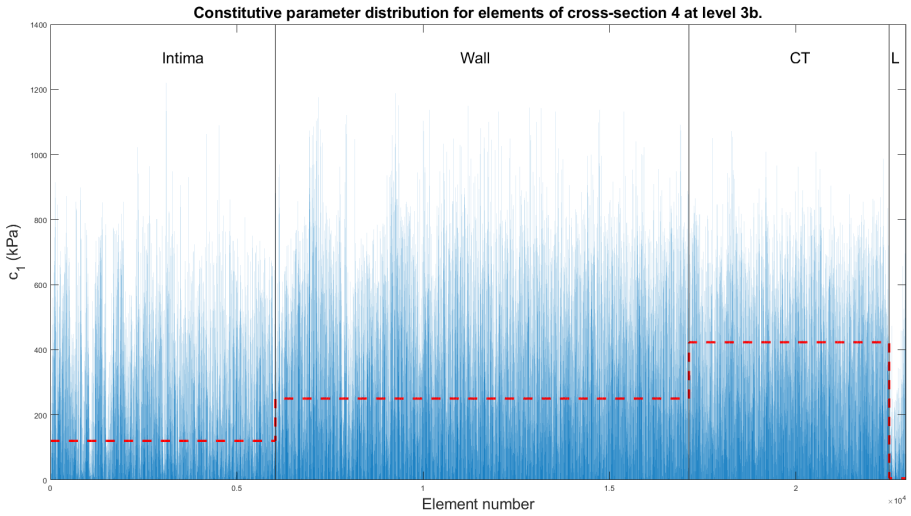

Figure 23: *This bar graph shows the calculated c_1 values of all elements in cross-section 4 for the level 3a approach. As can be seen, the elements (horizontal axis) all have varying constitutive parameter values (vertical axis). It does not resemble the red dashed line in any way, which is the ground truth. The black vertical lines delineate the figure into four groups of elements. Each group represents the elements in the corresponding component, e.g. the calcified tissue (CT) component can be found in the second-from-right group, and the lipid (L) component can be found in the most-right group.*

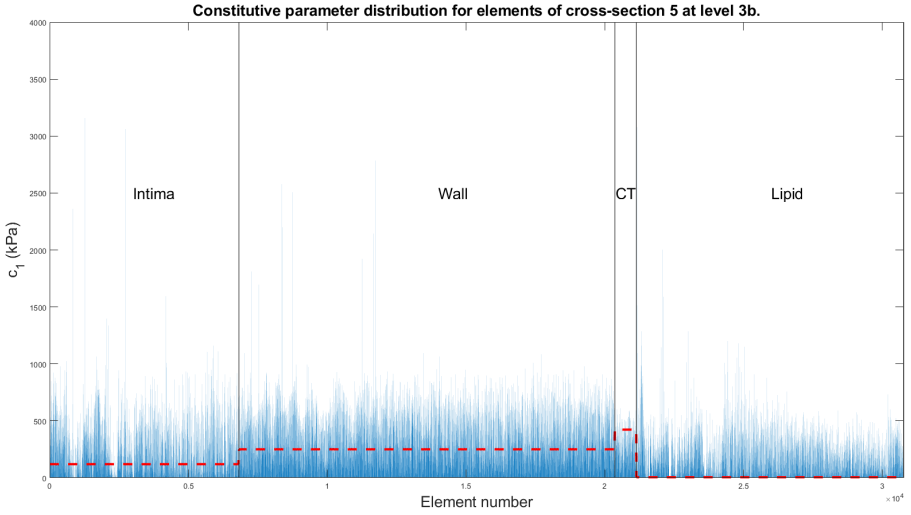

Figure 24: *This bar graph shows the calculated c_1 values of all elements in cross-section 5 for the level 3a approach. As can be seen, the elements (horizontal axis) all have varying constitutive parameter values (vertical axis). It does not resemble the red dashed line in any way, which is the ground truth. The black vertical lines delineate the figure into four groups of elements. Each group represents the elements in the corresponding component, the calcified tissue (CT) component can be found in the second-from-right group.*

7.5 Δ and Δ_{rel} for level 3a cross-section 1

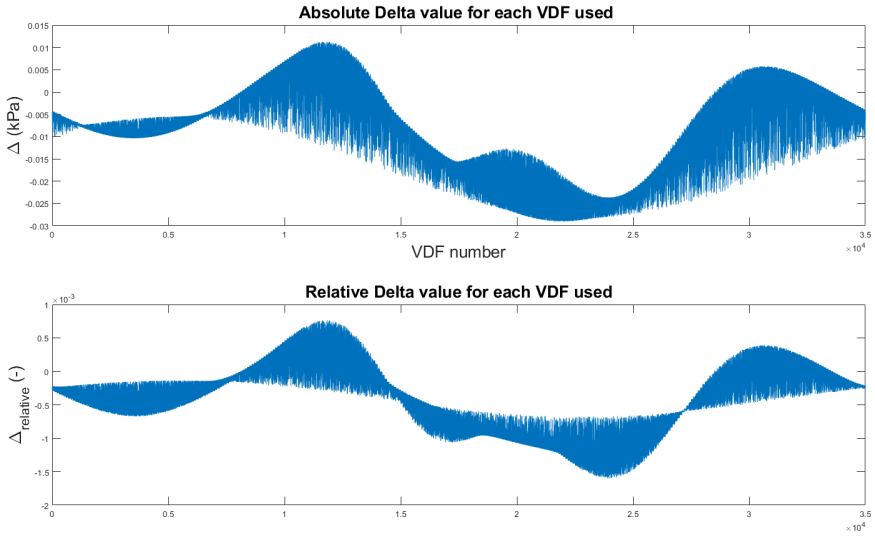

Figure 25: *This figure depicts $\underline{\Delta}$ and $\underline{\Delta}_{rel}$ for cross-section 1 at level 3a. The top plot depicts the difference vector $\underline{\Delta}$ (Equation (25)). As can be seen, the absolute difference between them is small. The bottom plot depicts the relative difference vector (Equation (26)). The relative difference also is small. The unit of $\underline{\Delta}$ is kPa, see Section 2.5 for the details of this.*

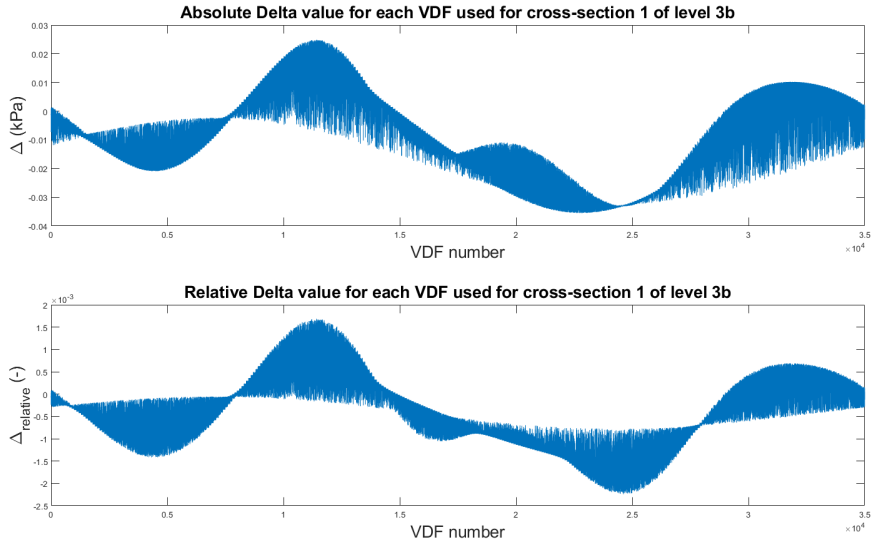

Figure 26: *This figure depicts $\underline{\Delta}$ and $\underline{\Delta}_{rel}$ for cross-section 1 at level 3b. The top plot depicts the difference vector $\underline{\Delta}$ (Equation (25)). As can be seen, the absolute difference between them is small. The bottom plot depicts the relative difference vector (Equation (26)). The relative difference also is small. This is the same figure as shown in Section 4.3.2. Comparing this figure to the one above, it is clear that $\underline{\Delta}$ for level 3a is in the same order as $\underline{\Delta}$ for level 3b. The same holds for the $\underline{\Delta}_{rel}$ of both sublevels. The unit of $\underline{\Delta}$ is kPa, see Section 2.5 for the details of this.*

7.6 Δ and Δ_{rel} for level 3a cross-section 3

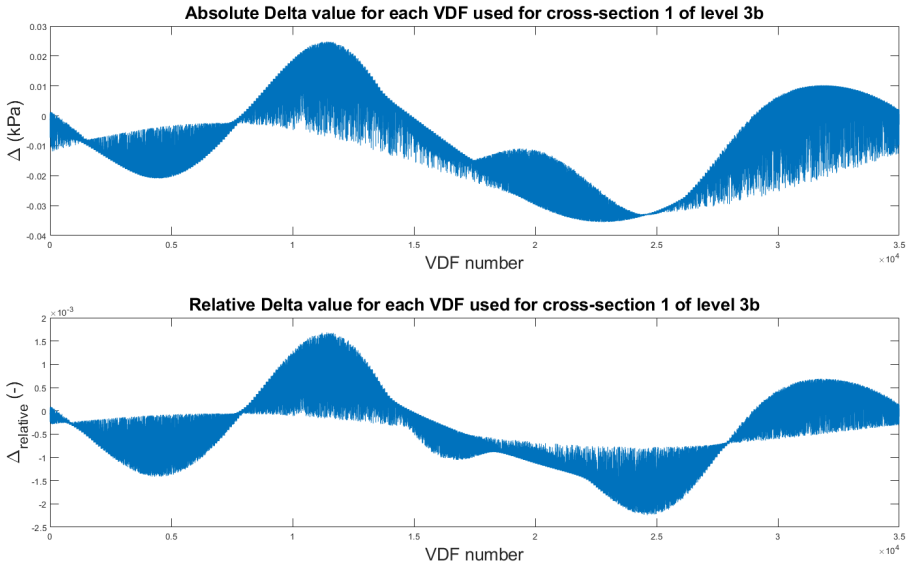

Figure 27: *This figure depicts $\underline{\Delta}$ and $\underline{\Delta}_{rel}$ for cross-section 3 at level 3a. The top plot depicts the difference vector $\underline{\Delta}$ (Equation (25)). The bottom plot depicts the relative difference vector (Equation (26)). This figure shows how $\underline{\Delta}$ and $\underline{\Delta}_{rel}$ are an order of magnitude larger than cross-section 1's $\underline{\Delta}$ and $\underline{\Delta}_{rel}$. The unit of $\underline{\Delta}$ is kPa, see Section 2.5 for the details of this.*

7.7 MATLAB script

The MATLAB script is divided into two. One set of scripts is meant for level 1 and 3a, while the other is meant for level 2 and 3b. The "dataload" files reference input and report files, created by ABAQUS.

7.7.1 Level 1 and 3a

main.m :

```
1
2  % clc
3  clear
4  % close all
5
6  c = fix(clock); totaltime=tic;  disp(['   Simulation started on ' ...
7  ,num2str(c(3)),'-',num2str(c(2)),'-', num2str(c(1)),' at ',num2str(c(4)) ,':' ,num2str(c(5))])
8
9  % % version  5 = Geo  5, 50 micron homogeneous
10 % % version  6 = Geo  6, 50 micron homogeneous
11 % % version  7 = Geo  7, 50 micron homogeneous
12 % % version  8 = Geo  8, 50 micron homogeneous
13 % % version  9 = Geo  9, 50 micron homogeneous
14
15
16 delt = 1000; % 1000 for KPa, 1 for MPa
17 P = 0.01333*delt;
18
19
20 minfield = 1;
21 field =100;
22 rots = 350;
23 fact = 9;
24 addpv = 1;
25 powervec = addpv+(fact*rand(rots*(field-minfield+1),1));
26
27 for version = 5:9
28 c = fix(clock);  disp(['   Version ', num2str(version), ' started at ' ...
29 ,num2str(c(3)),'-',num2str(c(2)),'-', num2str(c(1)),' at ',num2str(c(4)) ,':' ,num2str(c(5))])
30 dataloadlevel1
31
32 MaxRot=180-(180/(rots+1));
33 Delta = zeros(rots*(field-minfield+1),1); IV  ...
       W=zeros(rots*(field-minfield+1),length(Elset));  EVW=zeros(rots*(field-minfield+1),1);
34 clearvars 'IVW2'
35
36 for i = 1:rots;from = (i-1)*(field-minfield+1)+1; to = ...
       (i-1)*(field-minfield+1)+field-minfield+1;
37 rotnr = i;
38
39 theta = ((i-1)/rots)*MaxRot ;
40
41 R = [cosd(theta), -sind(theta); sind(theta) cosd(theta)];
42
43 coordn = R*[x';y']; COORDn = R*[X';Y'];
44 xn=coordn(1,:)'; yn = coordn(2,:)'; Xn=COORDn(1,:)'; Yn=COORDn(2,:)';
45
46 [Delta(from:to,1), IVW(from:to,:),EVW(from:to,1),¬] ...
       =Encalc4_2(xn,yn,Xn,Yn,lumn,Elset,field,minfield,delt,gt,powervec,rotnr,P) ;
47
48 end
49
50
51
52 %% Cvalue calculation level 1
```

```
53
54  IVW2(:,1) = sum(IVW,2);
55  cvalsLevel1{version} = EVW./IVW2  ;
56
57  cvalavgLevel1(version) = mean(cvalsLevel1{version});
58
59  disp(['   Level 3a: For Geo',num2str(version),', the c1 value found is ', ...
        num2str(cvalavgLevel1(version)), ' kPa '])
60
61
62  %% Cvalue calculation level 3a
63  % zeroth case is taken from section above.
64
65  options = optimoptions(@lsqlin,'Algorithm','trust-region-reflective');
66  lb=zeros(length(Elset),1);
67  ub=[];
68
69
70  x0 = cvalavgLevel1(version)*ones(length(Elset),1);
71  [xvals,res,¬,flag]=lsqlin(IVW,EVW,[],[],[],[],lb,[],x0,options);
72
73  disp(['   Lsqlin finished, now saving  '])
74
75
76  cvalues_level3a{version} = (xvals);
77  figure('keyPressFcn', @keyPressFcn);
78  bar(cvalues_level3a{version});
79  title(['Geo',num2str(version)])
80  savename = "workspace_version"+num2str(version)+"_Including_IVW";
81  save(savename, '-v7.3')
82  c = fix(clock);   disp(['   version ', num2str(version), ' finished at ' ...
83  ,num2str(c(3)),'-',num2str(c(2)),'-', num2str(c(1)),' at ',num2str(c(4)) ,':' ,num2str(c(5))])
84  end
85  c = fix(clock); disp(['   Total simulation time = ', num2str(toc(totaltime)),...
86  ' seconds and finished on ',num2str(c(3)),'-',num2str(c(2)),'-', num2str(c(1)) ...
87  ,' at ',num2str(c(4)) ,':' ,num2str(c(5)),])
88
89
90  %% Discussion
91  % The following section is commented out, as it is not necessary to have
92  % when runing the code for results. Uncomment for discussion plots.
93  % make sure to have the workspace mats in the current folder. Also make
94  % sure to have dataloadlevel2.m in here.
95
96
97  %%%%%%%%%%%%%%%%%%%%%%%%% Running Geo 5 (= cross section 1) again, but this time with
98  % random initial guess.
99
100 %     x0level3a_random_ini = delt*rand(length(Elset),1);
101 %     ...
        [xvalslevel3a_random_ini,reslevel3a_random_ini,¬,flaglevel3a_random_ini]=lsqlin(IVW,EVW ...
        ...
102 % ,[],[],[],[],lb,[],x0level3a_random_ini,options);
103
104 %%%%%%%%%%%%%%%%%%%%%%%%%
```

dataload1.m :

```
1       if version==5  % Geo 5 homogeneous 50 micron
2
3       Name_input_file='Geo5homo0050.inp';
4       report='Geo5homo0050report.rpt';
5       StartRowAllNodes=18; EndRowAllNodes=17832; Startreport=20; Endreport=17834;
```

```
    lumn=[
    4, 613, 614, 615, 616, 617, 618, 619, 620, 621, 622, 623, 624, 625, 626, 627  ...
    628, 629, 630, 631, 632, 633, 634, 635, 636, 637, 638, 639, 640, 641, 642, 643 ...
    644, 645, 646, 647, 648, 649, 650, 651, 652, 653, 654, 655, 656, 657, 658, 659 ...
    660, 661, 662, 663, 664, 665, 666, 667, 668, 669, 670, 671, 672, 673, 674, 675 ...
    676, 677, 678, 679, 680, 681, 682, 683, 684, 685, 686, 687, 688, 689, 690, 691 ...
    692, 693, 694, 695, 696, 697, 698, 699, 700, 701, 702, 703, 704, 705, 706, 707 ...
    708, 709, 710, 711, 712, 713, 714, 715, 716, 717, 718, 719, 720, 721, 722, 723 ...
    724, 725, 726, 727, 728, 729, 730, 731, 732, 733, 734, 735, 736, 737, 738, 739 ...
    740, 741, 742, 743, 744, 745, 746, 747, 748, 749, 750, 751, 752, 753, 754, 755 ...
    756, 757, 758, 759, 760, 761, 762, 763, 764, 765, 766, 767, 768, 769, 770, 771 ...
    772, 773, 774, 775, 776, 777, 778, 779, 780, 781, 782, 783, 784, 785, 786, 787 ...
    788, 789];
    lumn=[lumn lumn(1)];
    Geo5elset0050
    gt=0.1*ones(length(Elset),1);


    elseif version==6  % Geo 6 homogeneous 50 micron

    Name_input_file='Geo6homo0050.inp';
    report='Geo6homo0050report.rpt';
    StartRowAllNodes=18; EndRowAllNodes=28961; Startreport=20; Endreport=28963;
    lumn=[2, 501, 502, 503, 504, 505, 506, 507, 508, 509, 510, 511, 512, 513, 514, 515 ...
    516, 517, 518, 519, 520, 521, 522, 523, 524, 525, 526, 527, 528, 529, 530, 531 ...
    532, 533, 534, 535, 536, 537, 538, 539, 540, 541, 542, 543, 544, 545, 546, 547 ...
    548, 549, 550, 551, 552, 553, 554, 555, 556, 557, 558, 559, 560, 561, 562, 563 ...
    564, 565, 566, 567, 568, 569, 570, 571, 572, 573, 574, 575, 576, 577, 578, 579 ...
    580, 581, 582, 583, 584, 585, 586, 587, 588, 589, 590, 591, 592, 593, 594, 595 ...
    596, 597, 598, 599, 600, 601, 602, 603, 604, 605, 606, 607, 608, 609, 610, 611 ...
    612, 613, 614, 615, 616, 617, 618, 619, 620, 621, 622, 623, 624, 625, 626, 627 ...
    628, 629, 630, 631, 632, 633, 634, 635, 636, 637, 638, 639, 640, 641, 642, 643 ...
    644, 645, 646, 647, 648, 649, 650, 651, 652, 653, 654, 655, 656, 657, 658, 659 ...
    660, 661, 662, 663, 664, 665, 666, 667, 668, 669, 670, 671, 672, 673, 674, 675 ...
    676, 677, 678, 679, 680, 681, 682, 683, 684, 685, 686, 687, 688, 689, 690, 691 ...
    692, 693, 694, 695, 696, 697, 698, 699, 700, 701, 702, 703, 704, 705, 706, 707 ...
    708, 709, 710, 711, 712, 713, 714, 715, 716, 717, 718, 719, 720, 721, 722, 723  ...
    724, 725, 726, 727, 728, 729, 730, 731, 732, 733, 734, 735, 736, 737, 738, 739 ...
    740, 741, 742, 743, 744, 745, 746, 747, 748, 749, 750, 751, 752, 753, 754, 755];
    lumn=[lumn lumn(1)];
    Geo6elset0050
    gt=0.1*ones(length(Elset),1);



    elseif version==7  % Geo 7 homogeneous 50 micron

    Name_input_file='Geo7homo0050.inp';
    report='Geo7homo0050report.rpt';
    StartRowAllNodes=18; EndRowAllNodes=20672; Startreport=20; Endreport=20674;
    lumn=[2, 359, 360, 361, 362, 363, 364, 365, 366, 367, 368, 369, 370, 371, 372, 373  ...
    374, 375, 376, 377, 378, 379, 380, 381, 382, 383, 384, 385, 386, 387, 388, 389 ...
    390, 391, 392, 393, 394, 395, 396, 397, 398, 399, 400, 401, 402, 403, 404, 405 ...
    406, 407, 408, 409, 410, 411, 412, 413, 414, 415, 416, 417, 418, 419, 420, 421 ...
    422, 423, 424, 425, 426, 427, 428, 429, 430, 431, 432, 433, 434, 435, 436, 437];
    lumn=[lumn lumn(1)];
    Geo7elset0050
    gt=0.1*ones(length(Elset),1);



    elseif version==8  % Geo 8 homogeneous 50 micron

    Name_input_file='Geo8homo0050.inp';
    report='Geo8homo0050report.rpt';
```

```
    StartRowAllNodes=18; EndRowAllNodes=23361; Startreport=20; Endreport=23363;
    lumn=[   2, 423, 424, 425, 426, 427, 428, 429, 430, 431, 432, 433, 434, 435, 436, 437  ...
    438, 439, 440, 441, 442, 443, 444, 445, 446, 447, 448, 449, 450, 451, 452, 453 ...
    454, 455, 456, 457, 458, 459, 460, 461, 462, 463, 464, 465, 466, 467, 468, 469 ...
    470, 471, 472, 473, 474, 475, 476, 477, 478, 479, 480, 481, 482, 483, 484, 485 ...
    486, 487, 488, 489, 490, 491, 492, 493, 494, 495, 496, 497, 498, 499, 500, 501 ...
    502, 503, 504, 505, 506, 507, 508, 509, 510, 511, 512, 513, 514, 515, 516, 517 ...
    518, 519, 520, 521, 522, 523, 524, 525, 526, 527, 528, 529, 530, 531, 532, 533 ...
    534, 535, 536, 537, 538, 539, 540, 541, 542, 543, 544, 545, 546, 547, 548, 549 ...
    550, 551, 552, 553, 554, 555, 556, 557, 558, 559, 560, 561, 562, 563, 564, 565 ...
    566, 567, 568, 569, 570, 571, 572, 573, 574, 575, 576, 577, 578, 579, 580, 581 ...
    582, 583, 584, 585, 586, 587, 588, 589, 590, 591, 592, 593, 594, 595, 596, 597 ...
    598, 599, 600, 601, 602, 603, 604, 605, 606, 607, 608, 609, 610, 611, 612, 613 ...
    614, 615, 616, 617, 618, 619, 620, 621, 622, 623, 624, 625, 626, 627, 628, 629 ...
    630, 631, 632, 633, 634, 635, 636, 637, 638, 639, 640, 641, 642, 643, 644, 645 ...
    646, 647, 648, 649, 650, 651, 652, 653, 654, 655, 656, 657, 658, 659, 660, 661 ...
    662, 663, 664, 665];
    lumn=[lumn lumn(1)];
    Geo8elset0050
    gt=0.1*ones(length(Elset),1);


    elseif version==9  % Geo 9 homogeneous 50 micron

    Name_input_file='Geo9homo0050.inp';
    report='Geo9homo0050report.rpt';
    StartRowAllNodes=18; EndRowAllNodes=31147; Startreport=20; Endreport=31149;
    lumn=[
    2, 445, 446, 447, 448, 449, 450, 451, 452, 453, 454, 455, 456, 457, 458, 459 ...
    460, 461, 462, 463, 464, 465, 466, 467, 468, 469, 470, 471, 472, 473, 474, 475 ...
    476, 477, 478, 479, 480, 481, 482, 483, 484, 485, 486, 487, 488, 489, 490, 491 ...
    492, 493, 494, 495, 496, 497, 498, 499, 500, 501, 502, 503, 504, 505, 506, 507 ...
    508, 509, 510, 511, 512, 513, 514, 515, 516, 517, 518, 519, 520, 521, 522, 523 ...
    524, 525, 526, 527, 528, 529, 530, 531, 532, 533, 534, 535, 536, 537, 538, 539 ...
    540, 541, 542, 543, 544, 545, 546, 547, 548, 549, 550, 551, 552, 553, 554, 555 ...
    556, 557, 558, 559, 560, 561, 562, 563, 564, 565, 566, 567, 568, 569, 570, 571 ...
    572, 573, 574, 575, 576, 577, 578, 579, 580, 581];
    lumn=[lumn lumn(1)];
    Geo9elset0050
    gt=0.1*ones(length(Elset),1);


    else
    error('version number is not correct')

    end

    [¬,X,Y]=LoadAllNodesFun(Name_input_file,StartRowAllNodes,EndRowAllNodes);
    [¬,x,y]=LoadAllDefNodesFun(report,Startreport,Endreport);
```

LoadAllNodesFun.m:

```
    function ...
        [nodecoordinates,X,Y]=LoadAllNodesFun(Name_input_file,StartRowAllNodes,EndRowAllNodes)
    %% Loading input geometry

    % % This interesting function reads the input file.


    %% Initialize variables. Outside geometry
    filename = Name_input_file;
    delimiter = ',';
    %The first and the last line of the nodes in the ABAQUS input file:
```

```
    startRow = StartRowAllNodes;
    endRow = EndRowAllNodes;

    %% Format string for each line of text:
    %   column1: double (%f)
    %   column2: double (%f)
    %   column3: double (%f)
    %   column4: double (%f)
    % For more information, see the TEXTSCAN documentation.
    formatSpec = '%f%f%f%*s%[^\n\r]';

    %% Open the text file.
    fileID = fopen(filename,'r');

    %% Read columns of data according to format string.
    % This call is based on the structure of the file used to generate this
    % code. If an error occurs for a different file, try regenerating the code
    % from the Import Tool.
    dataArray = textscan(fileID, formatSpec, endRow-startRow+1, 'Delimiter', delimiter, ...
        'EmptyValue' ,NaN,'HeaderLines', startRow-1, 'ReturnOnError', false);

    %% Close the text file.
    fclose(fileID);

    %% Post processing for unimportable data.
    % No unimportable data rules were applied during the import, so no post
    % processing code is included. To generate code which works for
    % unimportable data, select unimportable cells in a file and regenerate the
    % script.

    %% Create output variable
    nodecoordinates = [dataArray{1:end-1}];
    X = dataArray{2}; Y = dataArray{3};
    %% Clear temporary variables
    clearvars filename delimiter startRow endRow formatSpec fileID dataArray ans;

    end
```

LoadAllDefNodesFun.m:

```
function [nodedefcoordinates,x,y]=LoadAllDefNodesFun(xydef,Startxydef,Endxydef)
%% Loading input geometry

%% Initialize variables. Outside geometry
filename = xydef;
delimiter = ',';
%The first and the last line of the nodes in the ABAQUS input file:
startRow = Startxydef;
endRow = Endxydef;

%% Format string for each line of text:
%   column1: double (%f)
%   column2: double (%f)
%   column3: double (%f)
%   column4: double (%f)
% For more information, see the TEXTSCAN documentation.
formatSpec = '%f%f%f%*s%[^\n\r]';

%% Open the text file.
fileID = fopen(filename,'r');

%% Read columns of data according to format string.
% This call is based on the structure of the file used to generate this
% code. If an error occurs for a different file, try regenerating the code
```

```
% from the Import Tool.
dataArray = textscan(fileID, formatSpec, endRow-startRow+1, 'Delimiter', delimiter, ...
    'EmptyValue' ,NaN,'HeaderLines', startRow-1, 'ReturnOnError', false);

%% Close the text file.
fclose(fileID);

%% Post processing for unimportable data.
% No unimportable data rules were applied during the import, so no post
% processing code is included. To generate code which works for
% unimportable data, select unimportable cells in a file and regenerate the
% script.

%% Create output variable
nodedefcoordinates = [dataArray{1:end-1}];
x=dataArray{2};y=dataArray{3};
%% Clear temporary variables
clearvars filename delimiter startRow endRow formatSpec fileID dataArray ans;

end
```

virfield_4.m:

```
function u = virfield_4(field,minfield,xmid,ymid ,powervec,rotnr)
u=zeros(size(xmid,1),field-minfield+1);
% cutting the element edge in correct number of lines.


for k = minfield:field
k2 = (rotnr-1)*(field-minfield+1)+k-minfield+1;
a=powervec(k2);
%     if rem(k2,2)== 0 % Even, 2 4 6 8
u(:,2*k-(2*minfield)+1:2*k-2*minfield+2) = ...
    [xmid./(xmid.^2+ymid.^2),ymid./(xmid.^2+ymid.^2)].*...
(xmid.^2./(xmid.^2+ymid.^2)).^(a);
%     elseif rem(k2,2)≠ 0 % uneven --> The k-1 is required to turn the exponents 3 5 7 9 ...
    to 2 4 6 8
%         u(:,2*k-(2*minfield)+1:2*k-2*minfield+2) =  ...
    [xmid./(xmid.^2+ymid.^2),ymid./(xmid.^2+ymid.^2)].*...
%             (ymid.^2./(xmid.^2+ymid.^2)).^(a);
%     end
end



end
```

virstrain_4.m:

```
function [e11, e22, e12] = virstrain_4(xcen, ycen, field,minfield,powervec,rotnr)
x=xcen; y=ycen;e11 = zeros(size(xcen,1),field-minfield+1);e12 = ...
    zeros(size(xcen,1),field-minfield+1);
e22 = zeros(size(xcen,1),field-minfield+1);
for k = minfield:field
a=powervec((rotnr-1)*(field-minfield+1)+k-minfield+1);

e11(:,k-minfield+1) =  ((x.^2./(x.^2 + y.^2)).^a.*(2.*a.*y.^2 - x.^2 + y.^2))./(x.^2 + ...
    y.^2).^2;
e22(:,k-minfield+1) = -((x.^2./(x.^2 + y.^2)).^a.*(2.*a.*y.^2 - x.^2 + y.^2))./(x.^2 + ...
    y.^2).^2;
```

```
e12(:,k-minfield+1) = -(y.*(x.^2./(x.^2 + y.^2)).^a.*(a.*x.^2 - a.*y.^2 + ...
    2.*x.^2))./(x.*(x.^2 + y.^2).^2);


end
```

keyPressFcn.m:

```
%% Version 2.0



function N(obj,eve)
if eve.Character == '.'
xl = xlim; yl = ylim;
h =  findobj('type','figure');
A=zeros(1,length(h));
for i = 1:length(h)
A(i) = h(i).Number;
end
A=sort(A);
ix = find(A==obj.Number);
if ix == length(A)
n = A(1);
else
n = A(ix+1);
end
figure(n);
xlim(xl); ylim(yl);
end
if eve.Character == ','
xl = xlim; yl = ylim;
h =  findobj('type','figure');
A=zeros(1,length(h));
for i = 1:length(h)
A(i) = h(i).Number;
end
A=sort(A);
ix = find(A==obj.Number);
if ix == 1
n = A(end);
else
n = A(ix-1);
end
figure(n);
xlim(xl); ylim(yl);
end

if eve.Character == '2'

h =  findobj('type','figure');
A=zeros(1,length(h));
for i = 1:length(h)
A(i) = h(i).Number;
end
A=sort(A);
ix = find(A==obj.Number);
if ix == length(A)
n = A(1);
else
n = A(ix+1);
end
```

```
figure(n);

end
if eve.Character == '1'

h =  findobj('type','figure');
A=zeros(1,length(h));
for i = 1:length(h)
A(i) = h(i).Number;
end
A=sort(A);
ix = find(A==obj.Number);
if ix == 1
n = A(end);
else
n = A(ix-1);
end
figure(n);

end
end
```

7.7.2 Level 2 and 3b

main.m :

```
% clc
clear
% close all

c = fix(clock); totaltime=tic;  disp(['   Simulation started on ' ...
,num2str(c(3)),'-',num2str(c(2)),'-', num2str(c(1)),' at ',num2str(c(4)) ,':' ...
    ,num2str(c(5)),])
timeforEn =tic;

% % version  5 = Geo  5, 50 micron heterogeneous
% % version  6 = Geo  6, 50 micron heterogeneous
% % version  7 = Geo  7, 50 micron heterogeneous
% % version  8 = Geo  8, 50 micron heterogeneous
% % version  9 = Geo  9, 50 micron heterogeneous

delt = 1000; % 1000 for KPa, 1 for MPa
P = 0.01333*delt;


minfield =1;
field =100; %1520 %
rots = 350;  % 30 geeft rank = 187 %
fact = 9;
addpv = 1.0; % add to powervec.
% powervec = addpv+(fact*rand(rots*(field-minfield+1),1));

powervec = addpv+(fact*rand(rots*(field-minfield+1),1));

for version = 5:9
c = fix(clock); totaltime=tic;  disp(['   Version ', num2str(version), ' started at ' ...
,num2str(c(3)),'-',num2str(c(2)),'-', num2str(c(1)),' at ',num2str(c(4)) ,':' ...
    ,num2str(c(5)),])
dataloadlevel2_3b

MaxRot=180-(180/(rots+1));
Delta = zeros(rots*(field-minfield+1),1); ...
    IVW=zeros(rots*(field-minfield+1),length(Elset));  EVW=zeros(rots*(field-minfield+1),1);
clearvars 'IVW4'
```

```
36
37 for i = 1:rots;from = (i-1)*(field-minfield+1)+1; to = ...
       (i-1)*(field-minfield+1)+field-minfield+1;
38 rotnr = i;
39 theta = ((i-1)/rots)*MaxRot ;
40 R = [cosd(theta), -sind(theta); sind(theta) cosd(theta)];
41 coordn = R*[x';y']; COORDn = R*[X';Y'];
42 xn=coordn(1,:)'; yn = coordn(2,:)'; Xn=COORDn(1,:)'; Yn=COORDn(2,:)';
43 [Delta(from:to,1), IVW(from:to,:),EVW(from:to,1),¬] ...
       =Encalc4_2(xn,yn,Xn,Yn,lumn,Elset,field,minfield,delt,gt,powervec,rotnr,P) ;
44 end
45
46 %%
47
48 IVW4(:,1) = sum(IVW(:, plaquenr),2); IVW4(:,2) = sum(IVW(:,wallnr),2);
49 IVW4(:,3) = sum(IVW(:,lipidnr),2); IVW4(:,4) = sum(IVW(:,calcnr),2);
50 condIVW4 = cond(IVW4); rankIVW4 = rank(IVW4);
51 disp(['     Geo',num2str(version),    'cond IVW4 = ', num2str( condIVW4),',  rank IVW4 = ...
       ', num2str( rankIVW4)])
52
53 %%
54
55 options = optimoptions(@lsqlin,'Display','none','Algorithm','trust-region-reflective');
56 lb=zeros(4,1);
57 ub=[];
58 % Level 2 calculation
59 for i = 1 :100
60 x0level2(:,i) =1000*rand(4,1);
61 [cvalslevel2(i,:),reslevel2(i,1),¬,flaglevel2(i,1)]=lsqlin(IVW4,EVW,[],[],[],[],lb,[], ...
62 x0level2(:,i),options);
63 end
64
65 c_valueslevel2(version,:) = mean(cvalslevel2);
66
67 % Level 3b calculation
68
69 x0level3b = delt*rand(length(Elset),1);
70 lb=zeros(length(Elset),1);
71 [cvalslevel3b,reslevel3b,¬,flaglevel3b]=lsqlin(IVW,EVW,[],[],[],[],lb,[],x0level3b,options);
72
73 disp(['   Both lsqlin finished, now saving  '])
74
75 cvalues_level3b{version} = (cvalslevel3b);
76 figure('keyPressFcn', @keyPressFcn);
77 bar(cvalues_level3b{version});
78 title(['Geo',num2str(version)])
79 savename = "workspace_level2_and_3b_version"+num2str(version)+"_Including_IVW";
80 save(savename, '-v7.3')
81
82 c = fix(clock);   disp(['   version ', num2str(version), ' finished at ' ...
83 ,num2str(c(3)),'-',num2str(c(2)),'-', num2str(c(1)),' at ',num2str(c(4)) ,':' ,num2str(c(5))])
84 end
85
86 %% Table creation
87 tabIntima = c_valueslevel2(5:9,1);
88 tabWall =   c_valueslevel2(5:9,2);
89 tabLipid = c_valueslevel2(5:9,3);
90 tabCalc = c_valueslevel2(5:9,4);
91 Cross_sections = {'Geo5';'Geo6';'Geo7';'Geo8';'Geo9';};
92 T = table(Cross_sections,tabIntima,tabWall,tabLipid,tabCalc)
93
94
95 c = fix(clock); disp(['   Total simulation time = ', num2str(toc(totaltime)),...
96 ' seconds and finished on ',num2str(c(3)),'-',num2str(c(2)),'-', num2str(c(1)) ...
97 ,' at ',num2str(c(4)) ,':' ,num2str(c(5)),])
```

```
98
99  %% Discussion
100 % The following section is commented out, as it is not necessary to have
101 % when runing the code for results. Uncomment for discussion plots.
102 % make sure to have the workspace mats in the current folder. Also make
103 % sure to have dataloadlevel2_3b.m in here.
104
105 %%%%%%%%%%%%%%%%%%%%%% better initial guess for level 3a, initial guess from Table 2
106 % name = "workspace_level2_and_3b_version9_Including_IVW";
107 % c_valueslevel2=load(name, 'c_valueslevel2');c_valueslevel2 = c_valueslevel2.c_valueslevel2;
108 %
109 % version =5;
110 %
111 % dataloadlevel2_3b
112 % x0_x0_from_level2 = zeros(length(Elset),1); x0_x0_from_level2(plaquenr)= ...
        c_valueslevel2(5,1);
113 % x0_x0_from_level2(wallnr)= c_valueslevel2(5,2); x0_x0_from_level2(lipidnr)= ...
        c_valueslevel2(5,3);
114 % x0_x0_from_level2(calcnr)= c_valueslevel2(5,4);
115 % options = optimoptions(@lsqlin,'Display','none','Algorithm','trust-region-reflective');
116 % lb=zeros(length(Elset),1);
117 % ub=[];
118 % [cvals_x0_from_level2,res_x0_from_level2,¬,flag_x0_from_level2]=lsqlin(IVW,EVW ...
119 % ,[],[],[],[],lb,[],x0_x0_from_level2,options);
120
121 %%%%%%%%%%%%%%%%%%%%%% creation of level 2, letting 1<k<60 (Table 3)
122
123 % minfield =1;
124 % field =100; %1520 %
125 % rots = 350;  % 30 geeft rank = 187 %
126 % fact = 59;
127 % addpv = 1.0; % add to powervec.
128 % powervec_if_1_k_60 = addpv+(fact*rand(rots*(field-minfield+1),1));
129 %
130 % for version = 5:9
131 %     c = fix(clock); totaltime=tic;  disp(['   Version ', num2str(version), ' started at ...
      ' ...
132 %     ,num2str(c(3)),'-',num2str(c(2)),'-', num2str(c(1)),' at ',num2str(c(4)) ,':' ...
      ,num2str(c(5)),])
133 %     dataloadlevel2_3b
134 %
135 %     MaxRot=180-(180/(rots+1));
136 %     Delta = zeros(rots*(field-minfield+1),1); ...
      IVW=zeros(rots*(field-minfield+1),length(Elset));  EVW=zeros(rots*(field-minfield+1),1);
137 %     clearvars 'IVW4'
138 %
139 %     for i = 1:rots;from = (i-1)*(field-minfield+1)+1; to = ...
      (i-1)*(field-minfield+1)+field-minfield+1;
140 %         rotnr = i;
141 %         theta = ((i-1)/rots)*MaxRot ;
142 %         R = [cosd(theta), -sind(theta); sind(theta) cosd(theta)];
143 %         coordn = R*[x';y']; COORDn = R*[X';Y'];
144 %         xn=coordn(1,:)'; yn = coordn(2,:)'; Xn=COORDn(1,:)'; Yn=COORDn(2,:)';
145 %         [Delta(from:to,1), IVW(from:to,:),EVW(from:to,1),¬] ...
      =Encalc4_2(xn,yn,Xn,Yn,lumn,Elset,field,minfield,delt,gt,powervec,rotnr,P) ;
146 %     end
147 %
148 %     %%
149 %
150 %     IVW4(:,1) = sum(IVW(:, plaquenr),2); IVW4(:,2) = sum(IVW(:,wallnr),2);
151 %     IVW4(:,3) = sum(IVW(:,lipidnr),2); IVW4(:,4) = sum(IVW(:,calcnr),2);
152 %     condIVW4 = cond(IVW4); rankIVW4 = rank(IVW4);
153 %     disp(['     Geo',num2str(version),    'cond IVW4 = ', num2str( condIVW4),',  rank ...
      IVW4 = ', num2str( rankIVW4)])
154 %
```

```
%     %%
%
%     options = optimoptions(@lsqlin,'Display','none','Algorithm','trust-region-reflective');
%     lb=zeros(4,1);
%     ub=[];
%
%     for i = 1 :100
%         x0level2_if_1_k_60(:,i) =1000*rand(4,1);
% [cvalslevel_if_1_k_602(i,:),reslevel2_if_1_k_60(i,1),¬,flaglevel2_if_1_k_60(i,1)]= ...
% lsqlin(IVW4,EVW,[],[],[],[],lb,[],x0level2_if_1_k_60(:,i),options);
%     end
%     c_valueslevel2_if_1_k_60(version,:) = mean(cvalslevel2_if_1_k_60);
% end
%
%  tabIntima2 = c_valueslevel2_if_1_k_60(5:9,1);
%  tabWall2 =   c_valueslevel2_if_1_k_60(5:9,2);
%  tabLipid2 = c_valueslevel2_if_1_k_60(5:9,3);
%  tabCalc2 = c_valueslevel2_if_1_k_60(5:9,4);
%  Cross_sections2 = {'Geo5';'Geo6';'Geo7';'Geo8';'Geo9';};
%  T2 = table(Cross_sections2,tabIntima2,tabWall2,tabLipid2,tabCalc2)

%%%%%%%%%%%%%%%%%%%%%%% Using results of table 3 (i.e. 1<k<60) as initial
% guess for level 3b:
% x0_x0_from_level2_T3 = zeros(length(Elset),1);
% x0_x0_from_level2_T3(plaquenr)= c_valueslevel2_if_1_k_60(5,1);
% x0_x0_from_level2_T3(wallnr)= c_valueslevel2_if_1_k_60(5,2);
% x0_x0_from_level2_T3(lipidnr)= c_valueslevel2_if_1_k_60(5,3);
% x0_x0_from_level2_T3(calcnr)= c_valueslevel2_if_1_k_60(5,4);
% options = optimoptions(@lsqlin,'Display','none','Algorithm','trust-region-reflective');
%     lb=zeros(length(Elset),1);
%     ub=[];
% [cvals_x0_from_level2_T3,res_x0_from_level2_T3,¬,flag_x0_from_level2_T3]=lsqlin( ...
%IVW,EVW,[],[],[],[],lb,[],x0_x0_from_level2_T3,options);

%%%%%%%%%%%%%%%%%%%%%%% creating IVWmod and EVWmod, and calculating
% cvals_with_IVWmod


% IVWmod = IVW+[eye(length(Elset)); zeros(field*rots-length(Elset),length(Elset))]; ...
    EVWmod=IVWmod*gt*delt;
% options = optimoptions(@lsqlin,'Display','none','Algorithm','trust-region-reflective');
%     lb=zeros(length(Elset),1);
%     ub=[];
% x0_IVWmod = delt*rand(length(Elset),1);
% [cvals_IVWmod,res_IVWmod,¬,flag_IVWmod]=lsqlin(IVWmod,EVWmod ...
%,[],[],[],[],lb,[],x0_IVWmod,options);


%%%%%%%%%%%%%%%%%%%%%%% Calculating c-value distribution with full rank
% ensured (400 micrometer mesh size)


% version = 10;
% delt = 1000; % 1000 for KPa, 1 for MPa
% P = 0.01333*delt;
%
%
% minfield =1;
% field =200; %
% rots = 500;  %
% fact = 9;
% addpv = 1.0; %
%
% powervec_400mu = addpv+(fact*rand(rots*(field-minfield+1),1));
%     dataloadlevel2_3b
```

```
%
%     MaxRot=180-(180/(rots+1));
%     Delta_400mu = zeros(rots*(field-minfield+1),1); ...
    IVW_400mu=zeros(rots*(field-minfield+1),length(Elset));  ...
    EVW_400mu=zeros(rots*(field-minfield+1),1);
%
%
%     for i = 1:rots;from = (i-1)*(field-minfield+1)+1; to = ...
    (i-1)*(field-minfield+1)+field-minfield+1;
%         rotnr = i;
%         theta = ((i-1)/rots)*MaxRot ;
%         R = [cosd(theta), -sind(theta); sind(theta) cosd(theta)];
%         coordn = R*[x';y']; COORDn = R*[X';Y'];
%         xn=coordn(1,:)'; yn = coordn(2,:)'; Xn=COORDn(1,:)'; Yn=COORDn(2,:)';
%         [Delta_400mu(from:to,1), IVW_400mu(from:to,:),EVW_400mu(from:to,1),¬] ...
    =Encalc4_2(xn,yn,Xn,Yn,lumn,Elset,field,minfield,delt,gt,powervec_400mu,rotnr,P) ;
%     end
%
%
%
% [¬,B]=licols(IVW_400mu');
% IVW2=IVW_400mu(B,:);
% EVW2gt=IVW2*gt*delt;
% EVW2 = EVW_400mu(B);
%
% rankIVW2 =rank(IVW2);
% condIVW2=cond(IVW2);
%
% options = optimoptions(@lsqlin,'Display','none','Algorithm','trust-region-reflective');
% lb=zeros(length(Elset),1);
% ub=[];
%
% x0_400mu = delt*rand(length(Elset),1);
% [cvals_400mu,res_400mu,¬,flag_400mu]=lsqlin(IVW2,EVW2,[],[],[],[],lb,[],x0_400mu,options);
```

dataloadlevel2_3b.m:

```
if  version ==5 % Geo 5

Name_input_file='Geo5hetero0050.inp';
report='Geo5hetero0050report.rpt';
StartRowAllNodes=18; EndRowAllNodes=17832; Startreport=20; Endreport=17834;
lumn=[
4, 613, 614, 615, 616, 617, 618, 619, 620, 621, 622, 623, 624, 625, 626, 627 ...
628, 629, 630, 631, 632, 633, 634, 635, 636, 637, 638, 639, 640, 641, 642, 643 ...
644, 645, 646, 647, 648, 649, 650, 651, 652, 653, 654, 655, 656, 657, 658, 659 ...
660, 661, 662, 663, 664, 665, 666, 667, 668, 669, 670, 671, 672, 673, 674, 675 ...
676, 677, 678, 679, 680, 681, 682, 683, 684, 685, 686, 687, 688, 689, 690, 691 ...
692, 693, 694, 695, 696, 697, 698, 699, 700, 701, 702, 703, 704, 705, 706, 707 ...
708, 709, 710, 711, 712, 713, 714, 715, 716, 717, 718, 719, 720, 721, 722, 723 ...
724, 725, 726, 727, 728, 729, 730, 731, 732, 733, 734, 735, 736, 737, 738, 739 ...
740, 741, 742, 743, 744, 745, 746, 747, 748, 749, 750, 751, 752, 753, 754, 755 ...
756, 757, 758, 759, 760, 761, 762, 763, 764, 765, 766, 767, 768, 769, 770, 771 ...
772, 773, 774, 775, 776, 777, 778, 779, 780, 781, 782, 783, 784, 785, 786, 787 ...
788, 789];
lumn=[lumn lumn(1)];
Geo5elset0050;

gt = zeros(length(Elset),1);

wallnr   =  5381:14429;
plaquenr =     1: 5380;
lipidnr  = 14430:17143;
```

```
calcnr   = 17144:17495;

gt( plaquenr,1) = 0.120; % Plaque
gt(wallnr,1) = 0.250; % Wall
gt(lipidnr,1) = 0.005; % Lipid
gt(calcnr,1) = 0.423; % Calc


elseif version == 6 % Geo 6

Name_input_file='Geo6hetero0050.inp';
report='Geo6hetero0050report.rpt';
StartRowAllNodes=18; EndRowAllNodes=28961; Startreport=20; Endreport=28963;
lumn=[
2, 501, 502, 503, 504, 505, 506, 507, 508, 509, 510, 511, 512, 513, 514, 515 ...
516, 517, 518, 519, 520, 521, 522, 523, 524, 525, 526, 527, 528, 529, 530, 531 ...
532, 533, 534, 535, 536, 537, 538, 539, 540, 541, 542, 543, 544, 545, 546, 547 ...
548, 549, 550, 551, 552, 553, 554, 555, 556, 557, 558, 559, 560, 561, 562, 563 ...
564, 565, 566, 567, 568, 569, 570, 571, 572, 573, 574, 575, 576, 577, 578, 579 ...
580, 581, 582, 583, 584, 585, 586, 587, 588, 589, 590, 591, 592, 593, 594, 595 ...
596, 597, 598, 599, 600, 601, 602, 603, 604, 605, 606, 607, 608, 609, 610, 611 ...
612, 613, 614, 615, 616, 617, 618, 619, 620, 621, 622, 623, 624, 625, 626, 627 ...
628, 629, 630, 631, 632, 633, 634, 635, 636, 637, 638, 639, 640, 641, 642, 643 ...
644, 645, 646, 647, 648, 649, 650, 651, 652, 653, 654, 655, 656, 657, 658, 659 ...
660, 661, 662, 663, 664, 665, 666, 667, 668, 669, 670, 671, 672, 673, 674, 675 ...
676, 677, 678, 679, 680, 681, 682, 683, 684, 685, 686, 687, 688, 689, 690, 691 ...
692, 693, 694, 695, 696, 697, 698, 699, 700, 701, 702, 703, 704, 705, 706, 707 ...
708, 709, 710, 711, 712, 713, 714, 715, 716, 717, 718, 719, 720, 721, 722, 723 ...
724, 725, 726, 727, 728, 729, 730, 731, 732, 733, 734, 735, 736, 737, 738, 739 ...
740, 741, 742, 743, 744, 745, 746, 747, 748, 749, 750, 751, 752, 753, 754, 755   ];
lumn=[lumn lumn(1)];
Geo6elset0050;

gt = zeros(length(Elset),1);
wallnr   = 6203:18156 ;
plaquenr =   1:6202   ;
lipidnr  =  28058:28509;
calcnr   =  18157:28057;

gt(plaquenr,1) = 0.120; % Plaque
gt(wallnr,1) = 0.250; % Wall
gt(lipidnr,1) = 0.005; % Lipid
gt(calcnr,1) = 0.423; % Calc


elseif  version == 7 % Geo 7

Name_input_file='Geo7hetero0050.inp';
report='Geo7hetero0050report.rpt';
StartRowAllNodes=18; EndRowAllNodes= 20672 ; Startreport=20; Endreport= 20674 ;
lumn=[
2, 359, 360, 361, 362, 363, 364, 365, 366, 367, 368, 369, 370, 371, 372, 373 ...
374, 375, 376, 377, 378, 379, 380, 381, 382, 383, 384, 385, 386, 387, 388, 389 ...
390, 391, 392, 393, 394, 395, 396, 397, 398, 399, 400, 401, 402, 403, 404, 405 ...
406, 407, 408, 409, 410, 411, 412, 413, 414, 415, 416, 417, 418, 419, 420, 421 ...
422, 423, 424, 425, 426, 427, 428, 429, 430, 431, 432, 433, 434, 435, 436, 437 ...
];
lumn=[lumn lumn(1)];
Geo7elset0050;

gt = zeros(length(Elset),1);
wallnr   =  6418:  15260  ;
plaquenr =   1: 6417    ;
lipidnr  =  15261:  18870   ;
calcnr   =   18871:  20381 ;
```

```
92
93  gt(plaquenr,1) = 0.120; % Plaque
94  gt(wallnr,1) = 0.250; % Wall
95  gt(lipidnr,1) = 0.005; % Lipid
96  gt(calcnr,1) = 0.423; % Calc
97
98
99
100 elseif version == 8 % Geo 8
101
102 Name_input_file='Geo8hetero0050.inp';
103 report='Geo8hetero0050report.rpt';
104 StartRowAllNodes=18; EndRowAllNodes=23361; Startreport=20; Endreport=23363;
105 lumn=[
106 2, 423, 424, 425, 426, 427, 428, 429, 430, 431, 432, 433, 434, 435, 436, 437 ...
107 438, 439, 440, 441, 442, 443, 444, 445, 446, 447, 448, 449, 450, 451, 452, 453 ...
108 454, 455, 456, 457, 458, 459, 460, 461, 462, 463, 464, 465, 466, 467, 468, 469 ...
109 470, 471, 472, 473, 474, 475, 476, 477, 478, 479, 480, 481, 482, 483, 484, 485 ...
110 486, 487, 488, 489, 490, 491, 492, 493, 494, 495, 496, 497, 498, 499, 500, 501 ...
111 502, 503, 504, 505, 506, 507, 508, 509, 510, 511, 512, 513, 514, 515, 516, 517 ...
112 518, 519, 520, 521, 522, 523, 524, 525, 526, 527, 528, 529, 530, 531, 532, 533 ...
113 534, 535, 536, 537, 538, 539, 540, 541, 542, 543, 544, 545, 546, 547, 548, 549 ...
114 550, 551, 552, 553, 554, 555, 556, 557, 558, 559, 560, 561, 562, 563, 564, 565 ...
115 566, 567, 568, 569, 570, 571, 572, 573, 574, 575, 576, 577, 578, 579, 580, 581 ...
116 582, 583, 584, 585, 586, 587, 588, 589, 590, 591, 592, 593, 594, 595, 596, 597 ...
117 598, 599, 600, 601, 602, 603, 604, 605, 606, 607, 608, 609, 610, 611, 612, 613 ...
118 614, 615, 616, 617, 618, 619, 620, 621, 622, 623, 624, 625, 626, 627, 628, 629 ...
119 630, 631, 632, 633, 634, 635, 636, 637, 638, 639, 640, 641, 642, 643, 644, 645 ...
120 646, 647, 648, 649, 650, 651, 652, 653, 654, 655, 656, 657, 658, 659, 660, 661 ...
121 662, 663, 664, 665   ];
122 lumn=[lumn lumn(1)];
123 Geo8elset0050;
124
125 gt = zeros(length(Elset),1);
126 wallnr   =  6038:  17132;
127 plaquenr =     1:  6037;
128 lipidnr  = 22503:  22952;
129 calcnr   = 17133:  22502;
130
131 gt( plaquenr,1) = 0.120; % Plaque
132 gt(wallnr,1) = 0.250; % Wall
133 gt(lipidnr,1) = 0.005; % Lipid
134 gt(calcnr,1) = 0.423; % Calc
135
136 elseif version == 9 % Geo 9
137
138 Name_input_file='Geo9hetero0050.inp';
139 report='Geo9hetero0050report.rpt';
140 StartRowAllNodes=18; EndRowAllNodes=31147; Startreport=20; Endreport=31149;
141 lumn=[
142 2, 445, 446, 447, 448, 449, 450, 451, 452, 453, 454, 455, 456, 457, 458, 459 ...
143 460, 461, 462, 463, 464, 465, 466, 467, 468, 469, 470, 471, 472, 473, 474, 475 ...
144 476, 477, 478, 479, 480, 481, 482, 483, 484, 485, 486, 487, 488, 489, 490, 491 ...
145 492, 493, 494, 495, 496, 497, 498, 499, 500, 501, 502, 503, 504, 505, 506, 507 ...
146 508, 509, 510, 511, 512, 513, 514, 515, 516, 517, 518, 519, 520, 521, 522, 523 ...
147 524, 525, 526, 527, 528, 529, 530, 531, 532, 533, 534, 535, 536, 537, 538, 539 ...
148 540, 541, 542, 543, 544, 545, 546, 547, 548, 549, 550, 551, 552, 553, 554, 555 ...
149 556, 557, 558, 559, 560, 561, 562, 563, 564, 565, 566, 567, 568, 569, 570, 571 ...
150 572, 573, 574, 575, 576, 577, 578, 579, 580, 581   ];
151 lumn=[lumn lumn(1)];
152 Geo9elset0050;
153
154 gt = zeros(length(Elset),1);
155 wallnr   =  6823:  20367;
156 plaquenr =     1:   6822;
```

```
lipidnr  = 21141:  30770;
calcnr   = 20368:  21140;

gt( plaquenr,1) = 0.120; % Plaque
gt(wallnr,1) = 0.250; % Wall
gt(lipidnr,1) = 0.005; % Lipid
gt(calcnr,1) = 0.423; % Calc

elseif version == 10 % Geo 10

Name_input_file='Geo5hetero0400.inp';
report='Geo5hetero0400report.rpt';
StartRowAllNodes=10; EndRowAllNodes=470; Startreport=20; Endreport=480;
lumn=[
4,  99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113 ...
114, 115, 116, 117, 118, 119, 120, 121]; %];
lumn= [lumn lumn(1)];
Geo5elset0400;

gt = zeros(length(Elset),1);

plaquenr=(1   : 156); % Plaque
wallnr=(157 :302); % Wall
lipidnr=(303:388); % Lipid
calcnr=(389:420); % Calcified

gt( plaquenr,1) = 0.120; % Plaque
gt(wallnr,1) = 0.250; % Wall
gt(lipidnr,1) = 0.005; % Lipid
gt(calcnr,1) = 0.423; % Calc
else
error('version number is not correct')

end

[¬,X,Y]=LoadAllNodesFun(Name_input_file,StartRowAllNodes,EndRowAllNodes);
[¬,x,y]=LoadAllDefNodesFun(report,Startreport,Endreport);
```

LoadAllNodesFun.m:

```
function ...
    [nodecoordinates,X,Y]=LoadAllNodesFun(Name_input_file,StartRowAllNodes,EndRowAllNodes)
%% Loading input geometry

% % This interesting function reads the input file.


%% Initialize variables. Outside geometry
filename = Name_input_file;
delimiter = ',';
%The first and the last line of the nodes in the ABAQUS input file:
startRow = StartRowAllNodes;
endRow = EndRowAllNodes;

%% Format string for each line of text:
%   column1: double (%f)
%   column2: double (%f)
%   column3: double (%f)
%   column4: double (%f)
% For more information, see the TEXTSCAN documentation.
formatSpec = '%f%f%f%*s%[^\n\r]';

%% Open the text file.
```

```
fileID = fopen(filename,'r');

%% Read columns of data according to format string.
% This call is based on the structure of the file used to generate this
% code. If an error occurs for a different file, try regenerating the code
% from the Import Tool.
dataArray = textscan(fileID, formatSpec, endRow-startRow+1, 'Delimiter', delimiter, ...
    'EmptyValue' ,NaN,'HeaderLines', startRow-1, 'ReturnOnError', false);

%% Close the text file.
fclose(fileID);

%% Post processing for unimportable data.
% No unimportable data rules were applied during the import, so no post
% processing code is included. To generate code which works for
% unimportable data, select unimportable cells in a file and regenerate the
% script.

%% Create output variable
nodecoordinates = [dataArray{1:end-1}];
X = dataArray{2}; Y = dataArray{3};
%% Clear temporary variables
clearvars filename delimiter startRow endRow formatSpec fileID dataArray ans;

end
```

LoadAllDefNodesFun.m:

```
function [nodedefcoordinates,x,y]=LoadAllDefNodesFun(xydef,Startxydef,Endxydef)
%% Loading input geometry

%% Initialize variables. Outside geometry
filename = xydef;
delimiter = ',';
%The first and the last line of the nodes in the ABAQUS input file:
startRow = Startxydef;
endRow = Endxydef;

%% Format string for each line of text:
%   column1: double (%f)
%   column2: double (%f)
%   column3: double (%f)
%   column4: double (%f)
% For more information, see the TEXTSCAN documentation.
formatSpec = '%f%f%f%*s%[^\n\r]';

%% Open the text file.
fileID = fopen(filename,'r');

%% Read columns of data according to format string.
% This call is based on the structure of the file used to generate this
% code. If an error occurs for a different file, try regenerating the code
% from the Import Tool.
dataArray = textscan(fileID, formatSpec, endRow-startRow+1, 'Delimiter', delimiter, ...
    'EmptyValue' ,NaN,'HeaderLines', startRow-1, 'ReturnOnError', false);

%% Close the text file.
fclose(fileID);

%% Post processing for unimportable data.
% No unimportable data rules were applied during the import, so no post
% processing code is included. To generate code which works for
% unimportable data, select unimportable cells in a file and regenerate the
```

```
% script.

%% Create output variable
nodedefcoordinates = [dataArray{1:end-1}];
x=dataArray{2};y=dataArray{3};
%% Clear temporary variables
clearvars filename delimiter startRow endRow formatSpec fileID dataArray ans;

end
```

Encalc4_2.m

```
% Allereerst controle of de getallen van EVW wel kloppen. Voor element 8 ga
% ik de EVW checken, nodes 4 en 99 zijn lumennodes, de andere tweee zijn
% 223 en 224.

function [Delta, IVW,EVW,Ar] ...
=Encalc4_2(x,y,X,Y,lumn,Elset,field,minfield,delt,gt,powervec,rotnr,P)


% allereerst de traction load T: T=nP, P = 0.0133, n = vector dat richting
% de lumen center wijst ? of andersom? ik denk juist richting het center.




%% EVW calculation

% % % dl calculation:

% for i = 1:length(lumn)-1
%
% end

% % % calculation of T
% % calculation of n
% calculation of vector:
n=zeros(length(lumn)-1,2); ymid=zeros(length(lumn)-1,1); xmid=zeros(length(lumn)-1,1);
dl = sqrt( (x(lumn(2:end)) - x(lumn(1:end-1)) ).^2  + (y(lumn(2:end)) - ...
    y(lumn(1:end-1))).^2  );
vec2 = [x(lumn(2:end))-x(lumn(1:end-1)),y(lumn(2:end))-y(lumn(1:end-1))];
vecR2 = ([0 -1 ; 1 0]*vec2')';
for i = 1:length(lumn)-1
n(i,:) = vecR2(i,:)/norm(vecR2(i,:));
end

xmid = (x(lumn(2:end))+x(lumn(1:end-1)))/2;
ymid = (y(lumn(2:end))+y(lumn(1:end-1)))/2;
% % calculation of T:
T = n*P;


% % % calculation of u^*

% calculation of midpoints in lumenelements:


%
% u=[uindividual1,uindividual2];

u2 = virfield_4(field,minfield,xmid,ymid,powervec,rotnr);


```

```
% % % calculation of EVW:

% EVWindividual = ((T(:,1).*u(:,1)+T(:,2).*u(:,2)).*dl);
EVWindividual2 = ((T(:,1).*u2(:,1:2:size(u2,2)-1)+T(:,2).*u2(:,2:2:size(u2,2))).*dl)';


% EVW = sum(EVWindividual);
EVW = sum(EVWindividual2,2);


%% IVW calculation


% % % Calculation of sigma = 2*B*c

% % calculation of DG
Ar=zeros(length(Elset),1); B11=zeros(length(Elset),1);B22=zeros(length(Elset),1);
B12=zeros(length(Elset),1); ycen=zeros(length(Elset),1);xcen=zeros(length(Elset),1);
Ar(:,1)  = polyarea(x(Elset(:,2:end))', y(Elset(:,2:end))') ;
f11 = zeros(length(Elset),1); f22 = zeros(length(Elset),1); f21 = zeros(length(Elset),1);
f12 = zeros(length(Elset),1);


for i = 1:length(Elset)
xdisp = x(Elset(i,2:end));    Xdisp = X(Elset(i,2:end));
ydisp = y(Elset(i,2:end));    Ydisp = Y(Elset(i,2:end));
u = xdisp-Xdisp;
v = ydisp-Ydisp;
dudr = 0.25*(-u(1)+u(2)+u(3)-u(4));
duds = 0.25*(-u(1)-u(2)+u(3)+u(4));
dvdr = 0.25*(-v(1)+v(2)+v(3)-v(4));
dvds = 0.25*(-v(1)-v(2)+v(3)+v(4));

mat1 = [dudr duds ; dvdr dvds ] ;

dXdr = 0.25*(-Xdisp(1)+Xdisp(2)+Xdisp(3)-Xdisp(4));
dXds = 0.25*(-Xdisp(1)-Xdisp(2)+Xdisp(3)+Xdisp(4));
dYdr = 0.25*(-Ydisp(1)+Ydisp(2)+Ydisp(3)-Ydisp(4));
dYds = 0.25*(-Ydisp(1)-Ydisp(2)+Ydisp(3)+Ydisp(4));

mat2 = [dXdr dXds; dYdr dYds];

DGtemp = mat1*mat2^-1+eye(2);


f11(i,1)  = DGtemp(1,1);f12(i,1)  = DGtemp(1,2); f21(i,1)  = DGtemp(2,1); f22(i,1)  = ...
    DGtemp(2,2);
B11(i) = f11(i).^2+f12(i).^2;
B12(i) = f11(i).*f21(i)+f12(i).*f22(i);
B22(i) = f22(i).^2+f21(i).^2;


end

% % calculation of B


% % calculation of sigma:
S11 = 2*B11;
S12 = 2*B12;
S22 = 2*B22;

% % % Calculation of e*;

% % centerpoints of each element:
```

```
for i = 1:length(Elset)

xcen(i) = mean(x(Elset(i,2:end)));
ycen(i) = mean(y(Elset(i,2:end)));

end


[e112, e222, e122] = virstrain_4(xcen, ycen, field,minfield  ,powervec,rotnr);
% % % IVW per element:

% IVW = ((S11.*e11+2*S12.*e12+S22.*e22)).*Ar;
IVW = (((S11.*e112+2*S12.*e122+S22.*e222)).*Ar)';

% disp(['   rotnr = ',num2str(rotnr),', sumxycen = ', ...
    num2str(sum(abs(xcen))+sum(abs(ycen))),', sumxymid = ', ...
    num2str(sum(abs(xmid./(xmid.^2+ymid.^2)))+sum(abs(ymid./(xmid.^2+ymid.^2))))])

Delta =  IVW *gt*delt-EVW;

end
```

virfield_4.m

```
function u = virfield_4(field,minfield,xmid,ymid ,powervec,rotnr)
u=zeros(size(xmid,1),field-minfield+1);


for k = minfield:field
k2 = (rotnr-1)*(field-minfield+1)+k-minfield+1;
a=powervec(k2);
%     if rem(k2,2)== 0 % Even, 2 4 6 8
u(:,2*k-(2*minfield)+1:2*k-2*minfield+2) = ...
    [xmid./(xmid.^2+ymid.^2),ymid./(xmid.^2+ymid.^2)].*...
(xmid.^2./(xmid.^2+ymid.^2)).^(a);
%     elseif rem(k2,2)≠ 0 % uneven --> The k-1 is required to turn the exponents 3 5 7 9 ...
    to 2 4 6 8
%         u(:,2*k-(2*minfield)+1:2*k-2*minfield+2) =  ...
    [xmid./(xmid.^2+ymid.^2),ymid./(xmid.^2+ymid.^2)].*...
%             (ymid.^2./(xmid.^2+ymid.^2)).^(a);
%     end
end

end
```

virstrain_4.m

```
function [e11, e22, e12] = virstrain_4(xcen, ycen, field,minfield,powervec,rotnr)
x=xcen; y=ycen;e11 = zeros(size(xcen,1),field-minfield+1);e12 = ...
    zeros(size(xcen,1),field-minfield+1);
e22 = zeros(size(xcen,1),field-minfield+1);
for k = minfield:field
a=powervec((rotnr-1)*(field-minfield+1)+k-minfield+1);

e11(:,k-minfield+1) =  ((x.^2./(x.^2 + y.^2)).^a.*(2.*a.*y.^2 - x.^2 + y.^2))./(x.^2 + ...
    y.^2).^2;
e22(:,k-minfield+1) = -((x.^2./(x.^2 + y.^2)).^a.*(2.*a.*y.^2 - x.^2 + y.^2))./(x.^2 + ...
    y.^2).^2;
e12(:,k-minfield+1) = -(y.*(x.^2./(x.^2 + y.^2)).^a.*(a.*x.^2 - a.*y.^2 + ...
```

```
        2.*x.^2))./(x.*(x.^2 + y.^2).^2);

end
```

keyPressFcn.m:

```
%% Version 2.0


function N(obj,eve)
if eve.Character == '.'
xl = xlim; yl = ylim;
h =  findobj('type','figure');
A=zeros(1,length(h));
for i = 1:length(h)
A(i) = h(i).Number;
end
A=sort(A);
ix = find(A==obj.Number);
if ix == length(A)
n = A(1);
else
n = A(ix+1);
end
figure(n);
xlim(xl); ylim(yl);
end
if eve.Character == ','
xl = xlim; yl = ylim;
h =  findobj('type','figure');
A=zeros(1,length(h));
for i = 1:length(h)
A(i) = h(i).Number;
end
A=sort(A);
ix = find(A==obj.Number);
if ix == 1
n = A(end);
else
n = A(ix-1);
end
figure(n);
xlim(xl); ylim(yl);
end

if eve.Character == '1'

h =  findobj('type','figure');
A=zeros(1,length(h));
for i = 1:length(h)
A(i) = h(i).Number;
end
A=sort(A);
ix = find(A==obj.Number);
if ix == length(A)
n = A(1);
else
n = A(ix+1);
end
figure(n);

end
if eve.Character == '2'
```

```
58
59  h =  findobj('type','figure');
60  A=zeros(1,length(h));
61  for i = 1:length(h)
62  A(i) = h(i).Number;
63  end
64  A=sort(A);
65  ix = find(A==obj.Number);
66  if ix == 1
67  n = A(end);
68  else
69  n = A(ix-1);
70  end
71  figure(n);
72
73  end
74  end
```