UNIVERSITAT POLITÈCNICA DE CATALUNYA, BARCELONA

MSC. COMPUTATIONAL MECHANICS ERASMUS MUNDUS

ASSIGNMENT 3: FINITE DEFORMATION NONLINEAR ELASTICITY

Computational Solid Mechanics

Author:
Nikhil Dave

Date: June 7, 2018

Contents

List of Figures

1 Kirchhoff Saint-Venant material model

Isotropic linear elasticity can be derived from balance of linear momentum, the linearized strain displacement relation $\boldsymbol{\varepsilon} = \frac{1}{2}(\nabla \boldsymbol{u} + \nabla \boldsymbol{u}^T)$, and the stored elastic energy function

$$W(\boldsymbol{\varepsilon}) = \frac{\lambda}{2}(\text{tr}\boldsymbol{\varepsilon})^2 + \mu \ \text{tr}(\boldsymbol{\varepsilon}^2)$$

(1) Check that, the stress tensor obtained from $\boldsymbol{\sigma} = \partial W / \partial \boldsymbol{\varepsilon}$ agrees with the usual linear elasticity expression.

To derive this result, we would use the index notation and write,

$$\sigma_{ij} = \frac{\partial W}{\partial \varepsilon_{ij}}; \quad W = \frac{\lambda}{2}(\varepsilon_{ii})^2 + \mu \ \varepsilon_{jk}\varepsilon_{kj}$$

and therefore,

$$\sigma_{ij} = \frac{\partial W}{\partial \varepsilon_{ij}} = \frac{\partial}{\partial \varepsilon_{ij}}\left[\frac{\lambda}{2}(\varepsilon_{kk})^2 + \mu \ \varepsilon_{pq}\varepsilon_{qp}\right] = \frac{\lambda}{2}\frac{\partial}{\varepsilon_{ij}}(\varepsilon_{kk})^2 + \mu \frac{\partial}{\varepsilon_{ij}}[\varepsilon_{pq}\varepsilon_{qp}]$$

$$\sigma_{ij} = \frac{\lambda}{2} \ 2\varepsilon_{kk}\frac{\partial \varepsilon_{kk}}{\varepsilon_{ij}} + \mu \ \varepsilon_{qp}\frac{\partial \varepsilon_{qp}}{\partial \varepsilon_{ij}} = \lambda \varepsilon_{kk}\frac{\partial \varepsilon_{kk}}{\varepsilon_{ij}} + \mu \ \varepsilon_{qp}\frac{\partial \varepsilon_{pq}}{\partial \varepsilon_{pq}}\delta_{pi}\delta_{qj} + \mu \ \varepsilon_{pq}\frac{\partial \varepsilon_{qp}}{\partial \varepsilon_{qp}}\delta_{qi}\delta_{pj}$$

Knowing that $\partial \varepsilon_{ii} / \partial \varepsilon_{ii} = 1$, we have,

$$\sigma_{ij} = \lambda_{kk}\frac{\partial \varepsilon_{kk}}{\partial \varepsilon_{kk}}\delta_{ik}\delta_{jk} + \mu \varepsilon_{ji} + \mu \varepsilon_{ji} = \lambda \varepsilon_{kk}\delta_{ij} + 2\mu \varepsilon_{ij}$$

Hence, we get,

$$\boxed{\boldsymbol{\sigma} = \lambda \ \text{tr}(\varepsilon)\boldsymbol{I} + 2\mu\varepsilon} \tag{1}$$

Since the linearization of the Green-Lagrange strain tensor $\boldsymbol{E} = \frac{1}{2}(\boldsymbol{C} - \boldsymbol{Id})$ is the small strain tensor $\boldsymbol{\varepsilon}$, it is natural to extend isotropic elasticity to nonlinear elasticity as

$$W(\boldsymbol{E}) = \frac{\lambda}{2}(\text{tr}\boldsymbol{E})^2 + \mu \ \text{tr}(\boldsymbol{E}^2) \tag{2}$$

This hyperelastic model is called Kirchhoff Saint-Venant material model.

(2) According to the definition we gave in class about isotropy in nonlinear elasticity, is this model isotropic?

A model is isotropic if its constitutive behaviour is identical in any material direction. This implies that the relationship between the strain energy function, W and Right Cauchy-Green deformation tensor, $\boldsymbol{C}$ must be independent of the material axes chosen and, consequently, W must only be a function of the invariants of $\boldsymbol{C}$ as,

$$W(\boldsymbol{C}(\boldsymbol{X}), \boldsymbol{X}) = W(I_{\boldsymbol{C}}, II_{\boldsymbol{C}}, III_{\boldsymbol{C}}, \boldsymbol{X})$$

where the invariants of $\boldsymbol{C}$ are defined here as,

$$I_{\boldsymbol{C}} = \text{tr } \boldsymbol{C} = \boldsymbol{C} : \boldsymbol{I}$$
$$II_{\boldsymbol{C}} = \text{tr } \boldsymbol{C}\boldsymbol{C} = \boldsymbol{C} : \boldsymbol{C}$$
$$III_{\boldsymbol{C}} = \det \boldsymbol{C} = J^2$$

Since, the energy function in our case can be written as,

$$W(\boldsymbol{C}) = \frac{\lambda}{8}(\text{tr}[\boldsymbol{C} - \boldsymbol{I}])^2 + \frac{\mu}{4}\text{tr}[(\boldsymbol{C} - \boldsymbol{I})^2]$$

We see that the strain energy function depends on the invariant of $\boldsymbol{C}$ only. Hence, this model is isotropic.

(3) Derive the second Piola-Kirchhoff stress $\boldsymbol{S}$.

The second Piola-Kirchhoff stress tensor is given by the expression,

$$\boldsymbol{S} = \frac{\partial W(\boldsymbol{E})}{\partial \boldsymbol{E}} \tag{3}$$

We would again use the index notation to derive this result. Therefore, we have,

$$S_{ij} = \frac{\partial W}{\partial E_{ij}} = \frac{\partial}{\partial E_{ij}}\Big[\frac{\lambda}{2}(E_{kk})^2 + \mu\ E_{pq}E_{qp}\Big] = \lambda E_{kk}\frac{\partial E_{kk}}{\partial E_{ij}} + \mu\ E_{pq}\frac{\partial E_{qp}}{\partial E_{ij}} + \mu\ E_{qp}\frac{\partial E_{pq}}{\partial E_{ij}}$$

$$S_{ij} = \lambda E_{kk}\frac{\partial E_{kk}}{\partial E_{kk}}\delta_{ij}\delta_{jk} + \mu\ E_{pq}\frac{\partial E_{qp}}{\partial E_{qp}}\delta_{iq}\delta_{jp} + \mu\ E_{qp}\frac{\partial E_{pq}}{\partial E_{pq}}\delta_{ip}\delta_{jq} = \lambda E_{kk}\delta_{ij} + 2\mu E_{ij}$$

Hence, we get,

$$\boldsymbol{S} = \lambda\ \text{tr}(\boldsymbol{E})\boldsymbol{I} + 2\mu\boldsymbol{E} \tag{4}$$

Using the definition of Green -Lagrange strain tensor, we can write the above expression for the second Piola-Kirchhoff stress tensor as,

$$\boxed{\boldsymbol{S} = \frac{\lambda}{2}\text{tr}(\boldsymbol{C} - \boldsymbol{I})\boldsymbol{I} + \mu(\boldsymbol{C} - \boldsymbol{I})} \tag{5}$$

(4) For a uniform deformation of a rod aligned with the X axis ($x = \Lambda X,\ y = Y,\ z = Z$, where $\Lambda > 0$ is the stretch ratio along the X direction) derive the relation between the nominal normal stress P (the xX component of the first Piola-Kirchhoff stress) and the stretch ratio $\Lambda, P(\Lambda)$, and plot it.

Firstly, the deformation mapping is given as,

$$\boldsymbol{x} = \varphi(\boldsymbol{X}) = (\Lambda X,\ Y,\ Z) \tag{6}$$

Next, we compute the deformation gradient tensor from the expression,

$$F_{iJ} = \frac{\partial \varphi_i}{\partial X_J} \quad \Longrightarrow \quad \boldsymbol{F} = \begin{bmatrix} \Lambda & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{7}$$

As seen in equation (3), for calculating the second Piola-Kirchhoff stress tensor, we need the Green-Lagrange strain tensor, $\boldsymbol{E}$ which in turn requires the computation of the right Cauchy-Green deformation tensor, $\boldsymbol{C}$, given as,

$$\boldsymbol{C} = \boldsymbol{F}^T\boldsymbol{F} \quad \Longrightarrow \quad \boldsymbol{C} = \begin{bmatrix} \Lambda & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \Lambda & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \Lambda^2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Now, the Green-Lagrange strain tensor can be computed as,

$$\boldsymbol{E} = \frac{1}{2}(\boldsymbol{C} - \boldsymbol{I}) \quad \Longrightarrow \quad \boldsymbol{E} = \frac{1}{2}\begin{bmatrix} (\Lambda^2 - 1) & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Using the expression derived for the second Piola-Kirchhoff stress tensor in equation (4), we get,

$$\boldsymbol{S} = \frac{\lambda}{2}(\Lambda^2 - 1)\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} + 2\mu\,\frac{1}{2}\begin{bmatrix} (\Lambda^2 - 1) & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} = (\Lambda^2 - 1)\begin{bmatrix} (\lambda/2 + \mu) & 0 & 0 \\ 0 & \lambda/2 & 0 \\ 0 & 0 & \lambda/2 \end{bmatrix} \tag{8}$$

Finally, we know that the first Piola-Kirchhoff stress tensor is given as,

$$\boldsymbol{P} = \boldsymbol{F}\boldsymbol{S} \tag{9}$$

Hence, by using the expressions derived in equations (7), (8) and (9), we get,

$$\boldsymbol{P} = \begin{bmatrix} \Lambda & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} (\Lambda^2 - 1) \begin{bmatrix} (\lambda/2 + \mu) & 0 & 0 \\ 0 & \lambda/2 & 0 \\ 0 & 0 & \lambda/2 \end{bmatrix} = (\Lambda^2 - 1)\begin{bmatrix} \Lambda(\lambda/2 + \mu) & 0 & 0 \\ 0 & \lambda/2 & 0 \\ 0 & 0 & \lambda/2 \end{bmatrix}$$

Therefore, the relation between the nominal normal stress and the stretch ratio is given as,

$$\boxed{P(\Lambda) = \Lambda(\Lambda^2 - 1)(\lambda/2 + \mu)} \tag{10}$$

(5) Is the relation $P(\Lambda)$ monotonic? If not, derive the critical stretch Λ_{crit} at which the model fails with zero stiffness. Does this critical stretch depend on the elastic constants? Show that the material does not satisfy the growth conditions

$$W(\boldsymbol{E}) \longrightarrow +\infty \quad \text{when} \quad J \longrightarrow 0^+.$$

Discuss your answers.

To find whether the relation $P(\Lambda)$ derived above is monotonic or not, we plot the $P(\Lambda)$ vs. Λ graph as shown in Figure 1. It is very evident from the graph, that the slope (derivative) of the function changes its sign and therefore making the relation non-monotonic.

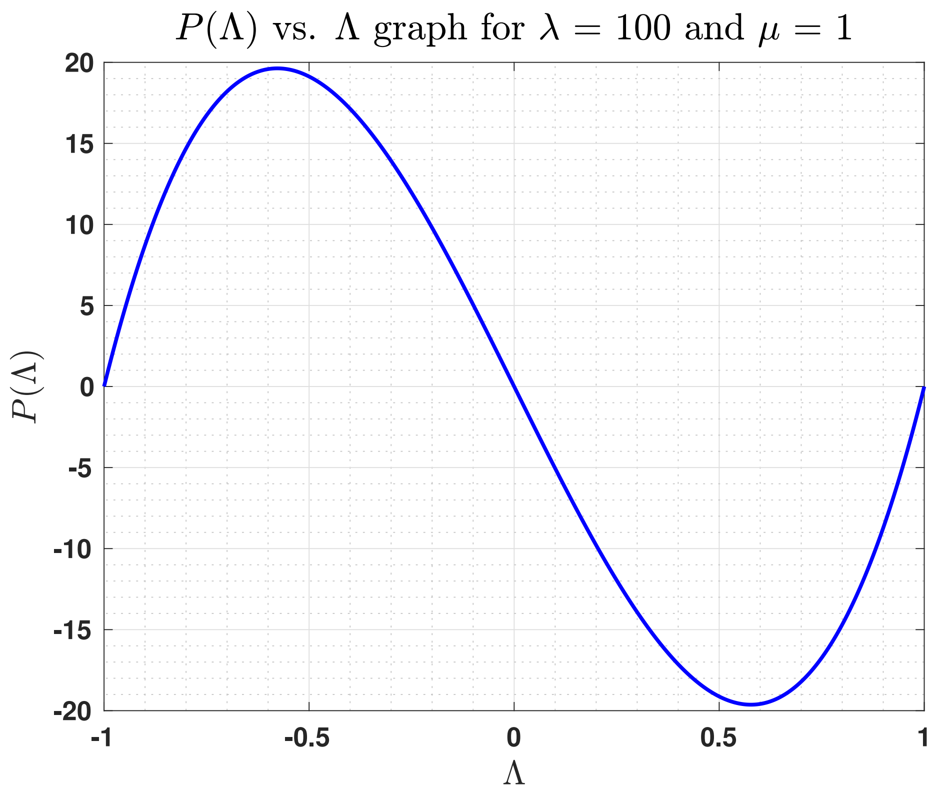


Figure 1: $P(\Lambda)$ vs. Λ graph for $\lambda = 100$ and $\mu = 1$.

Since the relation is not monotonic, we need to derive the critical stretch Λ_{crit}. To derive this, we need to satisfy the relation,

$$\frac{dP(\Lambda)}{d\Lambda} = 0$$

Using the relationship derived in equation (10), we get,

$$\frac{dP(\Lambda)}{d\Lambda} = \left(\frac{\lambda}{2} + \mu\right)(3\Lambda^2 - 1) = 0 \qquad \Longrightarrow \qquad \Lambda_{crit} = \pm\frac{\sqrt{3}}{3}$$

It is clear that the critical stretch, Λ_{crit} does not depend on the elastic constants.

Now, to check the growth conditions wee need to compute the Jacobian J and the energy function

$W(\boldsymbol{E})$. The Jacobian of the transformation is given as,

$$J = \det \boldsymbol{F} = \det \begin{bmatrix} \Lambda & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \Lambda \tag{11}$$

Next, we compute the components in the expression of the energy function given in equation (2).

$$\boldsymbol{E}^2 = \frac{1}{4} \begin{bmatrix} (\Lambda^2-1)^2 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \implies \text{tr}(\boldsymbol{E}^2) = \frac{1}{4}(\Lambda^2-1)^2$$

Also,

$$\text{tr}\boldsymbol{E} = \frac{1}{2}(\Lambda - 1) \implies (\text{tr}\boldsymbol{E})^2 = \frac{1}{4}(\Lambda-1)^2$$

Therefore, we have the energy function now given as,

$$W(\boldsymbol{E}) = \frac{\lambda}{2}(\text{tr}\boldsymbol{E})^2 + \mu\,\text{tr}(\boldsymbol{E}^2) = \frac{\lambda}{8}(\Lambda-1)^2 + \frac{\mu}{4}(\Lambda^2-1)^2 = \frac{\lambda}{8}(J-1)^2 + \frac{\mu}{4}(J^2-1)^2 \tag{12}$$

In order to check the growth conditions, we take the limit of the energy function as,

$$\lim_{J\to 0} W(\boldsymbol{E}) = \lim_{J\to 0} \frac{\lambda}{8}(J-1)^2 + \frac{\mu}{4}(J^2-1)^2 = \left(\frac{\lambda}{8} + \frac{\mu}{4}\right)$$

Since, the energy function does not grow to infinity instead gives a finite value dependent on the material parameters, μ and λ, we conclude that the material does not satisfy the growth conditions.

(6) Consider now the modified Kirchhoff Saint-Venant material model:

$$W(\boldsymbol{E}) = \frac{\lambda}{2}(\ln J)^2 + \mu\,\text{tr}(\boldsymbol{E}^2)$$

Does this model circumvent the drawbacks of the previous model?

For a modified Kirchhoff Saint-Venant material model, the Jacobian remains the same and so does the Green-Lagrange strain tensor. Thus we have again,

$$J = \det\boldsymbol{F} = \Lambda; \quad \text{tr}(\boldsymbol{E}^2) = \frac{1}{4}(\Lambda^2-1)^2$$

Therefore, we have the modified energy function as,

$$W(\boldsymbol{E}) = \frac{\lambda}{2}(\ln J)^2 + \frac{\mu}{4}(J^2-1)^2$$

Now, to check the growth conditions, we take the limit of the energy function as,

$$\lim_{J\to 0} W(\boldsymbol{E}) = \lim_{J\to 0} \frac{\lambda}{2}(\ln J)^2 + \frac{\mu}{4}(J^2-1)^2 = +\infty$$

This means the modified material model is able to circumvent the drawbacks of the previous model as the growth conditions are satisfied now.

(7) Implement the material model in Eq. (2) in the Matlab code. Perform the consistency test to check your implementation. Try to demonstrate the material instabilities of this model with a numerical example.

The material model in equation (2) was implemented in the Matlab code provided. To implement the model, we needed to express the constitutive tensor, $\mathbb{C}$ and the second Piola-Kirchhoff stress tensor, $\boldsymbol{S}$ in terms of the right Cauchy-Green tensor, $\boldsymbol{C}$. For this, we use the expression derived in equations (4) and (5) to get,

$$S_{ij} = \lambda E_{kk}\delta_{ij} + 2\mu E_{ij} = \frac{\lambda}{2}(C_{kk} - \delta_{kk})\delta_{ij} + \mu(C_{ij} - \delta_{ij})$$

Now for the constitutive tensor, $\mathbb{C}$, we have,

$$\mathbb{C}_{ijkl} = 2\frac{\partial S_{ij}}{\partial C_{kl}} = \lambda\frac{\partial(C_{mm} - \delta_{mm})}{\partial C_{kl}}\delta_{ij} + 2\mu\frac{\partial C_{ij}}{\partial C_{kl}}$$

$$\mathbb{C}_{ijkl} = \lambda\frac{\partial C_{mm}}{\partial C_{kl}}\delta_{ij} + 2\mu\frac{\partial C_{ij}}{\partial C_{kl}} = \lambda\frac{\partial C_{mm}}{\partial C_{mm}}\delta_{ij}\delta_{mk}\delta_{ml} + 2\mu\frac{\partial C_{ij}}{\partial C_{ij}}\delta_{ik}\delta_{lj} = \lambda\delta_{lk}\delta_{ij} + 2\mu\delta_{ik}\delta_{lj}$$

To check the consistency of the derived tensors and our implementation of the material model, the gradients and the hessian were computed using the provided function `Check_Derivatives` which gave negligible errors compared to the specified tolerance values. The implementation of the model is presented in the Appendix. To further validate the implemented material model, a numerical example was used wherein a slender beam was compressed to half of its length. Figure 2 shows the initial and deformed mesh of the beam.

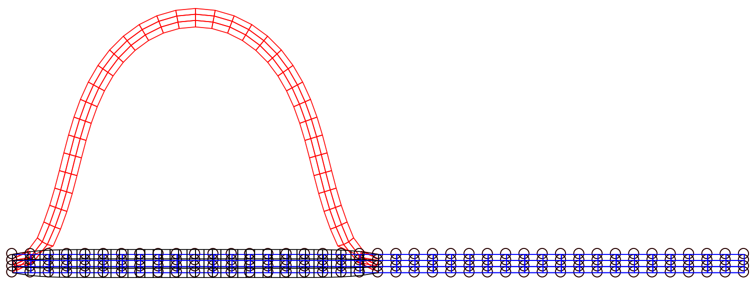


Figure 2: Initial and deformed mesh of the slender beam under compression using the Kirchhoff Saint-Venant material model.

Figure 3 shows the convergence plot for this model.

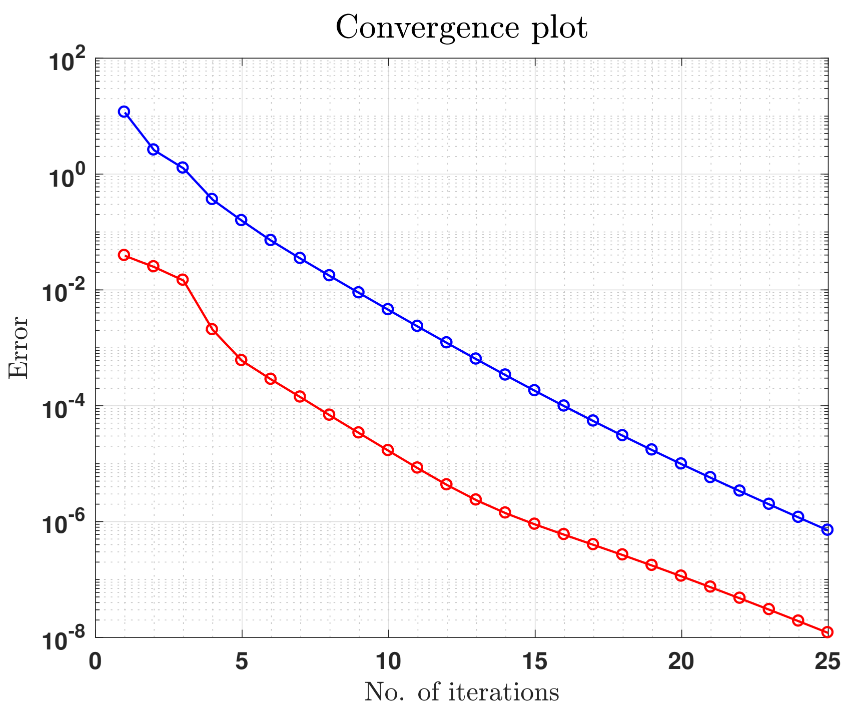


Figure 3: Convergence plot for the slender beam under compression using Kirchhoff Saint-Venant material model.

Next, the force-displacement graph for both the linear and nonlinear elasticity solution is plotted in Figure 4.

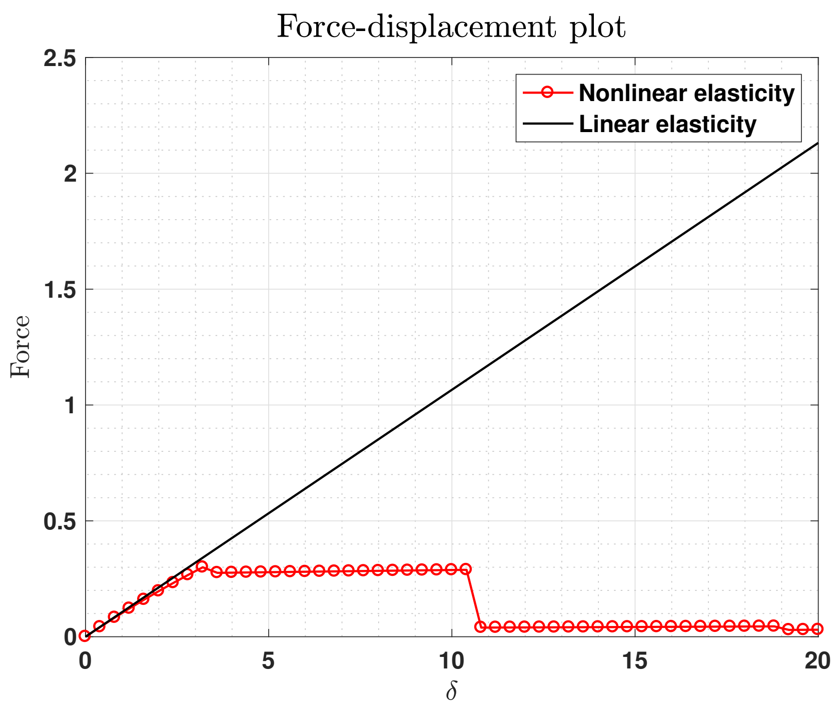


Figure 4: Force-displacement graph for linear and nonlinear elasticity solution of the slender beam under compression using Kirchhoff Saint-Venant material model.

The dramatic difference between the linear and nonlinear solution is clearly observed where at the critical point, a small increase in displacement suddenly decreases the force to a very small value and then remains constant thereafter making the model softer and deformable infinitely which is not physical in nature. As seen theoretically, this is due to the fact that the growth conditions are not satisfied with this material model.

In order to check how the modified Kirchoff Saint-Venant model evades the shortcomings of the previous model, we implemented the modified model in Matlab with the same approach. The implementation of the model is presented in the Appendix. Firstly, we calculated the second Piola-Kirchhoff stress tensor, $\boldsymbol{S}$ as,

$$S_{ij} = \frac{\partial W}{\partial E_{ij}} = -\mu\delta_{ij} + \mu C_{ij} + \lambda\,\ln J C^{-1}_{ij}$$

and the constitutive tensor, $\mathbb{C}$ as,

$$\mathbb{C}_{ijkl} = 2\frac{\partial S_{ij}}{\partial C_{kl}} = 2\mu\delta_{ikjl} + \lambda C^{-1}_{ij}C^{-1}_{kl} - 2\lambda\,\ln J(C^{-1}_{ik}C^{-1}_{jl} + C^{-1}_{il}C^{-1}_{jk})$$

After checking the consistency of the material implemented, we used the same numerical example with this modified material model. Figure 5 shows the initial and deformed mesh of the beam. It can be clearly observed that this material model fails as the slender beam is compressed to half of its length.

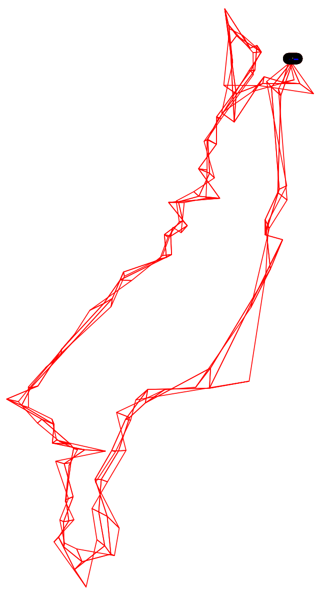


Figure 5: Initial and deformed mesh of the slender beam under compression using the modified Kirchhoff Saint-Venant material model.

The convergence plot for this model is shown in Figure 6.

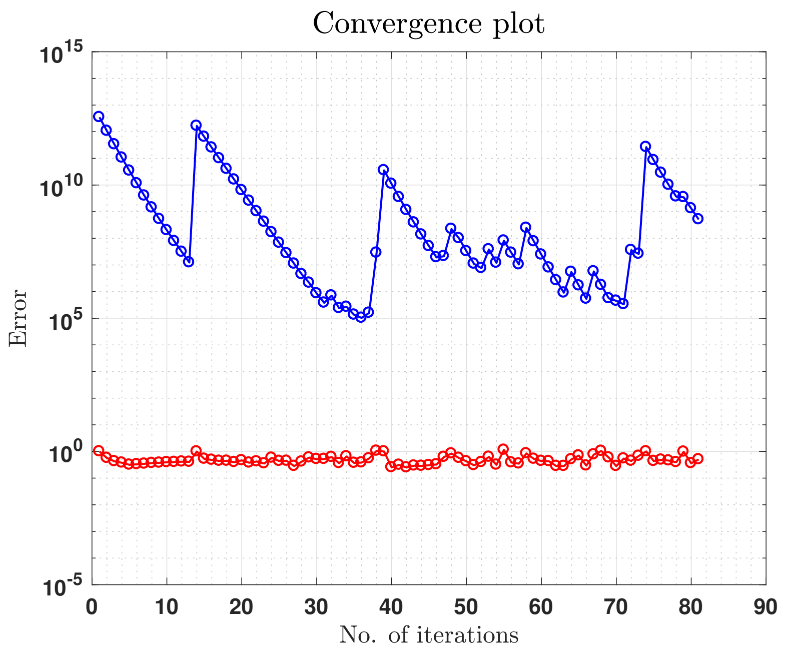


Figure 6: Convergence plot for the slender beam under compression using modified Kirchhoff Saint-Venant material model.

Next, the force-displacement graph is plotted in Figure 7.

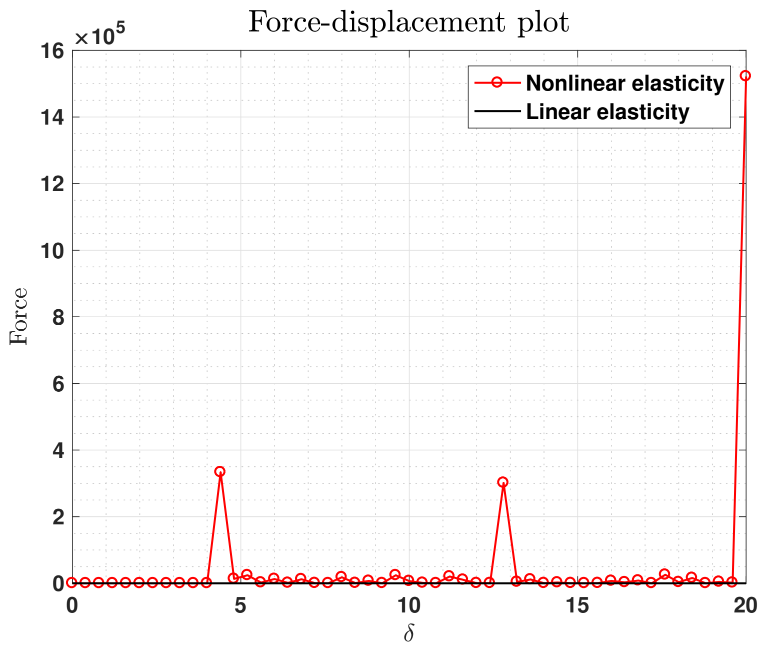


Figure 7: Force-displacement graph for linear and nonlinear elasticity solution of the slender beam under compression using modified Kirchhoff Saint-Venant material model.

As derived theoretically, it is observed that after surpassing the critical point, the force increases rapidly with small displacements in the model. As the growth conditions are satisfied for this material model, this is an expected result depicted the physical nature of the material.

2 Implementation of line-search

Implement a line-search algorithm to be used in combination with Newton's method. For this, I suggest you resort to Matlab's function `fminbd`, which performs 1D nonlinear minimization with bounds. You need to define a function `Ener_1D` that evaluates the energy along the line that passes through x in the direction of p (the descent search direction). The function `LineSearch` may include lines like the ones suggested next:

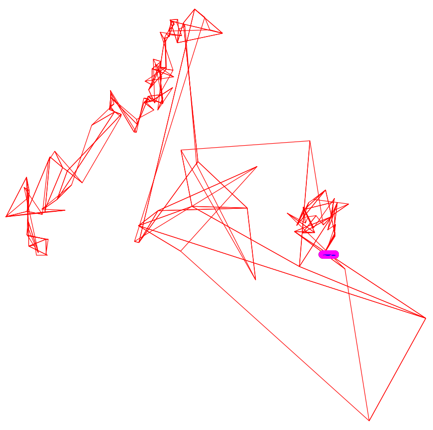
```
t=1;opts=optimset('TolX',options.TolX,'MaxIter',options.n_iter_max_LS);

t=fminbnd(@(t)Ener_1D(t,x_short,p),0,2,opts);

x_short=x_short+t*p;
```

Test the code with the examples where you expect buckling (the compression of the beams, or the deflection of the arch), and compare the results with and without linesearch.

Line search is an optimization tool that helps in identifying the minimum energy of the system by adding stability to the solution and thus leading to a better solution. In the line search strategy, the algorithm chooses a search direction s_k and tries to solve the following 1D minimization problem

$$\min_{t>0} f(x_k + t s_k)$$

where the scalar t is the step-length. The descent direction obtained is utilised to provide stability while solving the system of nonlinear equations when the Newton-Raphson method provides unstable (maximum energy) solution. This can be done by using the built-in Matlab function `fminbnd` or by using a backtracking method. The two examples where buckling is expected are tested for the Neo-Hookean material. First, the example of a slender beam under dead load is analysed. The initial and deformed mesh of the beam without and with line search are shown in Figures 8(a) and 8(b), respectively. It is evident from the results that in the case without line search, the simulation produces unstable, insignificant results and diverges, while in the other case, due to the stability provided by line search, we are able to capture the buckling modes of the slender beam.


(a) Mesh deformation without Line search

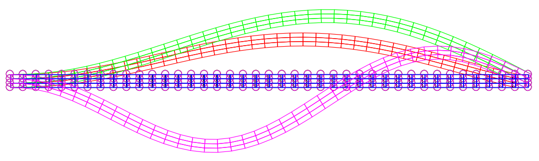
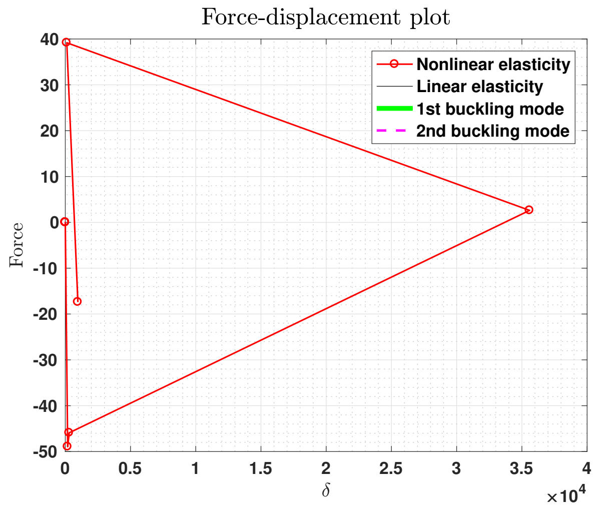
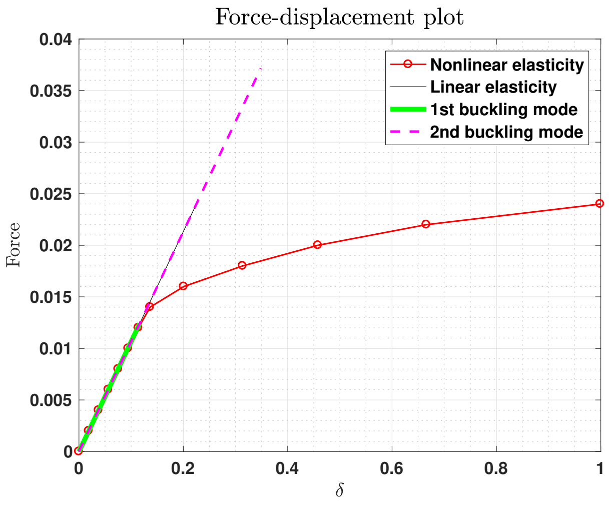

(b) Mesh deformation with Line search

Figure 8: Mesh for a slender beam under dead load using Neo-Hookean material model.

Next, we plot the Force-displacement curves for both the cases, without and with line search, in Figures 9(a) and 9(b), respectively. The similar analogy can be made with these plots, as in the case without line search the buckling event is not captured and the force value suddenly snaps during the simulation due to instability. When the line search is active, we can capture the force-displacement curve reasonably well in both the elastic and hyperelastic regions.



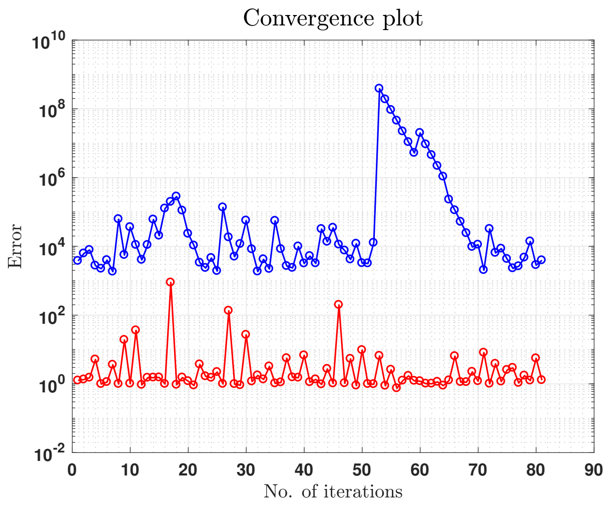
(a) Force-displacement curve without Line search



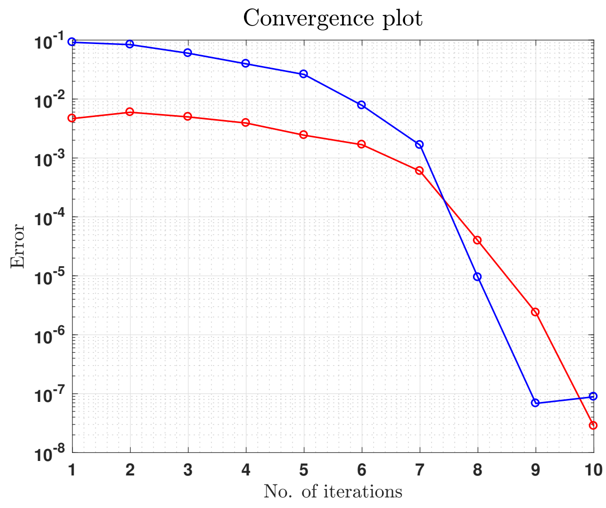
(b) Force-displacement curve with Line search

Figure 9: Force-displacement curves for a slender beam under dead load using Neo-Hookean material model.

Finally, we compare the error obtained using both approaches in Figures 10(a) and 10(b). The non-converging results of the errors obtained in case of no line search verify our understanding of the instabilities in the method. On the other hand, the results obtained with the line search shows a quadratic convergence of the errors.



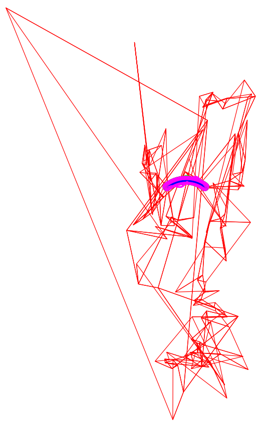
(a) Error obtained without Line search



(b) Error obtained with Line search

Figure 10: Error obtained for a slender beam under dead load using Neo-Hookean material model.

The second example analysed is the arch with dead load at the centre. Figures 11(a) and 11(b) shows the initial and deformed mesh of the arch without and with line search, respectively. A similar result could be seen in this example as well, as the solution is unstable and diverges when the line search is not used. However, using line search provides stability and we could capture the buckling modes of the arch.


(a) Mesh deformation without Line search

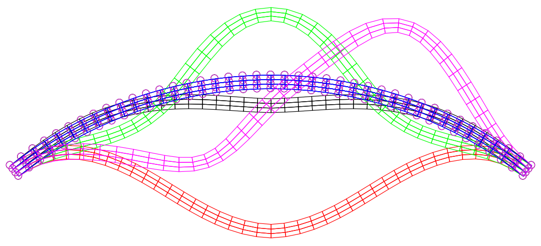
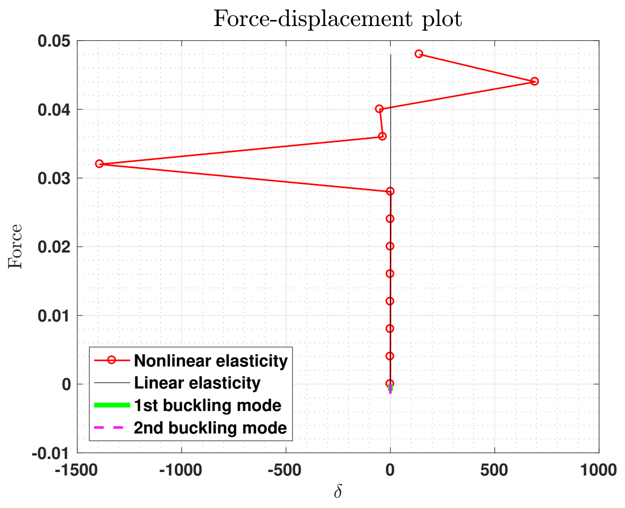
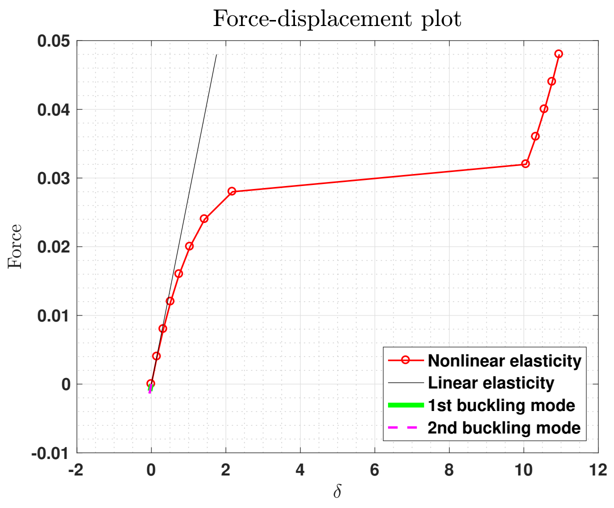

(b) Mesh deformation with Line search

Figure 11: Mesh for an arch with dead load at the center using Neo-Hookean material model.

The Force-displacement plot for both the cases are shown in Figures 12(a) and 12(b). Again, a similar relationship can be seen, as in the case without line search we get an insignificant force-displacement behaviour. But when the line search is active, we could capture the curve reasonably well although the buckling is represented by a snap behaviour in the force-displacement behaviour wherein a small increase in force results in sudden displacement in the model.



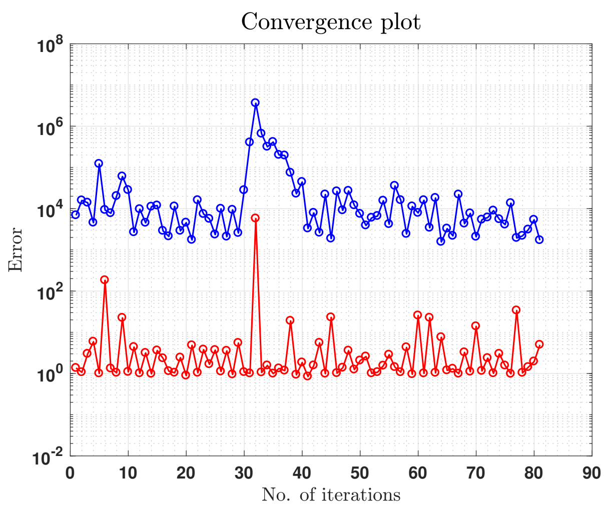
(a) Force-displacement curve without Line search



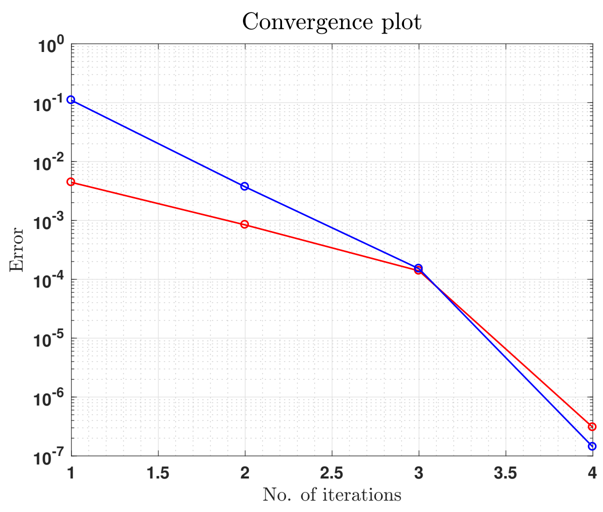
(b) Force-displacement curve with Line search

Figure 12: Force-displacement curves for an arch with dead load at the centre using Neo-Hookean material model.

Finally, Figures 13(a) and 13(b) represent the error obtained using both approaches. It is clearly seen again how the case without line search diverges and the use of line search provides stability and converges to the solution at a very good rate.



(a) Error obtained without Line search



(b) Error obtained with Line search

Figure 13: Error obtained for an arch with dead load at the centre using Neo-Hookean material model.

3 Implementation of a material model

The code you are given implements a plane-strain finite element method for finite deformation elasticity. A compressible Neo-Hookean material is already in place (modeling a slightly porous rubber for instance), whose strain energy density (or hyper-elastic potential) is

$$W(\boldsymbol{C}) = \frac{1}{2}\lambda_0(\ln J)^2 - \mu_0 \ln J + \frac{1}{2}\mu_0(\text{trace } \boldsymbol{C} - 3)$$

This constitutive model is isotropic. Note that, since we are considering plane strain, we can use a 2×2 reduced right Cauchy-Green deformation tensor and replace trace $\boldsymbol{C}$ - 3 by trace $\boldsymbol{C}$ - 2 in the above equation.

We want to consider now an anisotropic material, more specifically, a transversely isotropic material. We consider a material constitutive law for a rubber reinforced by fibers, all aligned in the same direction in such a way that perpendicular to the fibers, the material remains isotropic. The orientation of the fibers is given in the reference configuration by a unit vector $\boldsymbol{N}^{fib}$. Such a model depends on the principal invariants of $\boldsymbol{C}$, and additionally by the fourth invariant

$$I_4(\boldsymbol{C}) = \boldsymbol{N}^{fib} \cdot \boldsymbol{C} \cdot \boldsymbol{N}^{fib} = C_{IJ} N_I^{fib} N_J^{fib}$$

More specifically,

$$W(\boldsymbol{C}) = \frac{1}{2}\mu_0(\text{trace } \boldsymbol{C} - 3) - \mu_0 \ln J + \kappa \mathcal{G}(J) + c_0\left\{\exp\left[c_1(\sqrt{I_4(\boldsymbol{C})} - 1)^4)\right] - 1\right\}$$

where μ_0, κ, c_0 and c_1 are material parameters, and $\mathcal{G}(J)$ provides the volumetric response of the material. We consider

$$\mathcal{G}(J) = \frac{1}{4}(J^2 - 1 - 2\,\mathrm{ln}J)$$

The last term in the strain energy function specifies the contribution to the deformation energy of the fibers, and as typical in biological fibers, with this model these become stiffer the more deformed they are.

(a) Provide expressions for the second Piola-Kirchhoff stress tensor and for the elasticity tensor required in the linearization of the equilibrium equations.

The implementation of this material model requires computation of the second Piola-Kirchhoff tensor and the constitutive tensor.

We know that the second Piola-Kirchhoff tensor is given by $\boldsymbol{S} = 2\frac{\partial W}{\partial \boldsymbol{C}}$. Using the given expression for the strain energy function, W, we can write the second Piola-Kirchhoff tensor as,

$$\boldsymbol{S} = 2\frac{\partial}{\partial \boldsymbol{C}}\Big[\frac{1}{2}\mu_0(\mathrm{tr}\boldsymbol{C} - 3) - \mu_0 \mathrm{ln}J + \kappa\mathcal{G}(J) + c_0\Big\{\exp\Big(c_1(\sqrt{I_4(\boldsymbol{C})} - 1)^4\Big)\Big\}\Big]$$

We get,

$$\boldsymbol{S} = 2\Big[\frac{\mu_0}{2}\frac{\partial \mathrm{tr}\boldsymbol{C}}{\partial \boldsymbol{C}} - \frac{\mu_0}{J}\frac{\partial J}{\partial \boldsymbol{C}} + \frac{\kappa}{4}\Big(2J\frac{\partial J}{\partial \boldsymbol{C}} - \frac{2}{J}\frac{\partial J}{\partial \boldsymbol{C}}\Big) + 2c_0c_1\frac{\exp(c_1[\sqrt{I_4} - 1]^4)(\sqrt{I_4} - 1)^3}{\sqrt{I_4}}\frac{\partial I_4}{\partial \boldsymbol{C}}\Big]$$

Using the index notations and knowing the relations,

$$\frac{\partial \mathrm{tr}\boldsymbol{C}}{\partial \boldsymbol{C}} = \boldsymbol{I}; \quad \frac{\partial J}{\partial \boldsymbol{C}} = \frac{J}{2}\boldsymbol{C}^{-1}; \quad \frac{\partial I_4}{\partial C_{ij}} = N_i N_j$$

we get,

$$S_{ij} = \mu_0(\delta_{ij} - \boldsymbol{C}^{-1}_{ij}) + \frac{\kappa}{2}(J^2 - 1)\boldsymbol{C}^{-1}_{ij} + 4c_0c_1\,\exp(c_1[\sqrt{I_4} - 1]^4)\frac{(\sqrt{I_4} - 1)^3}{\sqrt{I_4}}N_iN_j$$

Similarly, the constitutive tensor, $\mathbb{C}$ can be derived as,

$$\mathbb{C}_{ijkl} = 2\frac{\partial S_{ij}}{\partial C_{kl}} = \kappa J^2 C^{-1}_{kl}C^{-1}_{ij} - \frac{\kappa}{2}(J^2 - 1 - \mu_0)(C^{-1}_{ik}C^{-1}_{jl} + C^{-1}_{il}C^{-1}_{jk})$$

$$+8c_0c_1\frac{(\sqrt{I_4} - 1)^2}{I_4}\,\exp\Big(c_1[\sqrt{I_4} - 1]^4\Big)\Big(2c_1[\sqrt{I_4} - 1]^4 + \frac{3}{2} - \frac{\sqrt{I_4} - 1}{2\sqrt{I_4}}\Big)N_iN_jN_kN_l$$

(b) Implement this material model into the code. The code is prepared for this model (`material=2`), including the definition of the material parameters in `preprocessing.m`.

The material model is incorporated into the provided code using the derivatives obtained in the previous section. The implementation is given in the Appendix. The material (`material=2`) is defined with the function `transv_isotr_i` for `i = 1,2,3`.

(c) Check the correctness (consistency test) of your implementation by running the script `Check_Derivatives.m` with `material=2`. This script checks the gradient of the energy (out-of-balance forces) and the Hessian of the energy (tangent stiffness matrix) by numerical differentiation. Check also that when solving a mechanical problem with this mode, Newton's method converges quadratically.

The consistency check to analyse the correctness of the implemented code is performed by running the provided script `Check_Derivatives` with `material=2`. It was noted that the gradient and hessian of the energy yields error less than the tolerance implying that the second Piola-Kirchhoff tensor and the constitutive tensor are implemented correctly. Hence it was decided to use the implementation to plot the convergence plot of the errors using provided numerical examples. Figure 14 shows the convergence plot using `example=0` where the quadratic convergence of Newton's method could be clearly seen.

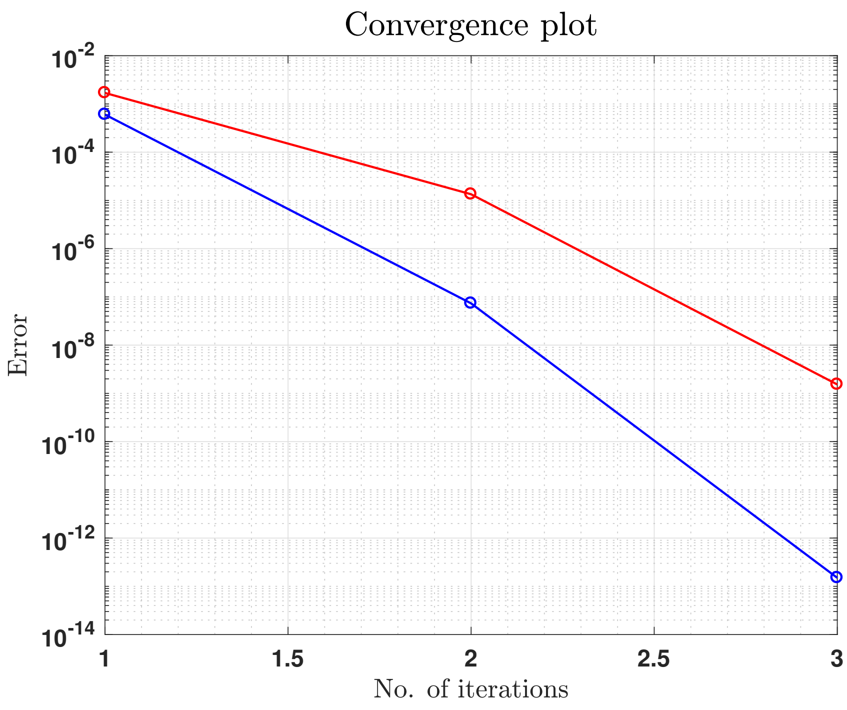


Figure 14: Error convergence plot for the implemented transversely isotropic material model.

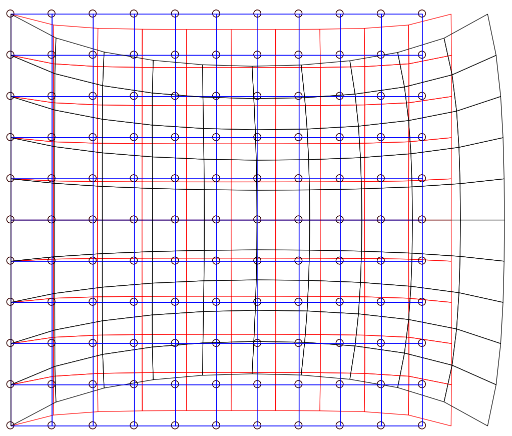
(d) Solve `example=0`, a dead load applied on an elastic block in tension, with a few representative orientations of the fibers. Consider $\theta = 0$ (fibers aligned with the loading direction), $\theta = \pi/6$, $\theta = \pi/4$ and $\theta = \pi/2$ (fibers perpendicular to the loading direction), where

$$\boldsymbol{N}^{fib} = \left[\cos\theta, \sin\theta\right]^T$$

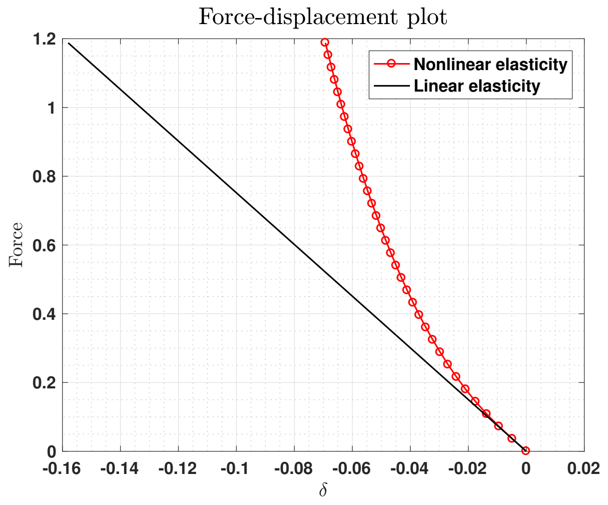
Explain the results from a mechanical viewpoint. Note that if mechanical instabilities are expected, you should use a solver involving line-search.

The implementation considering a transversely isotropic material is used to solve `example=0` for different fiber directions. We consider a material constitutive law for a rubber reinforced by fibers such that all fibres are aligned in the same direction and the material remains isotropic in the perpendicular direction to the fibers.

The results obtained for the fibre direction $\theta = 0$ are shown in Figure 15 where the fibres are aligned in the direction of the applied force. Since the material behaves isotropic in the perpendicular direction to the fibre direction, we expect similar results for the case with fibre direction $\theta = \pi/2$. Figure 16 shows the results obtained for this case where we observe that the model behaves similar to the previous case, as expected. It is interesting to see that this material behaves stiffer as typical in biological fibers.

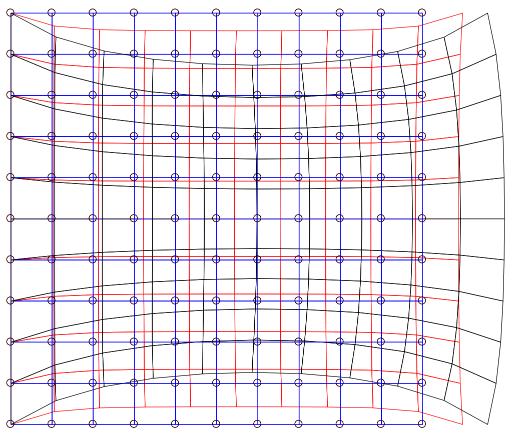


(a) Initial and deformed mesh

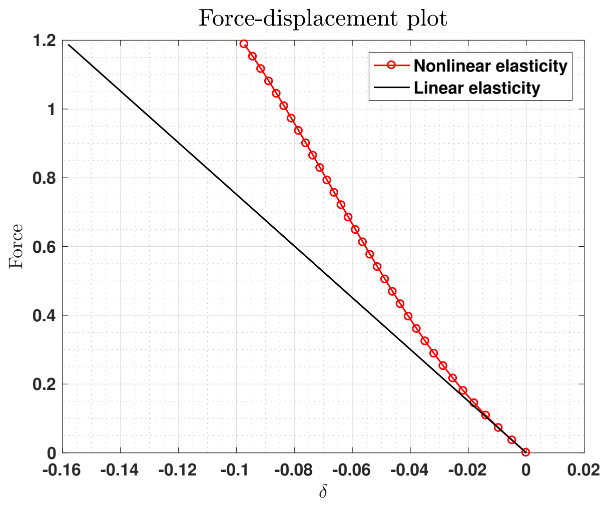


(b) Force-displacement curve

Figure 15: Initial, deformed mesh and force-displacement curve for $\theta = 0$.



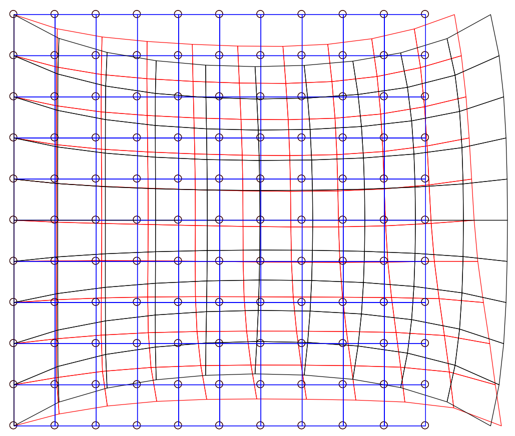
(a) Initial and deformed mesh



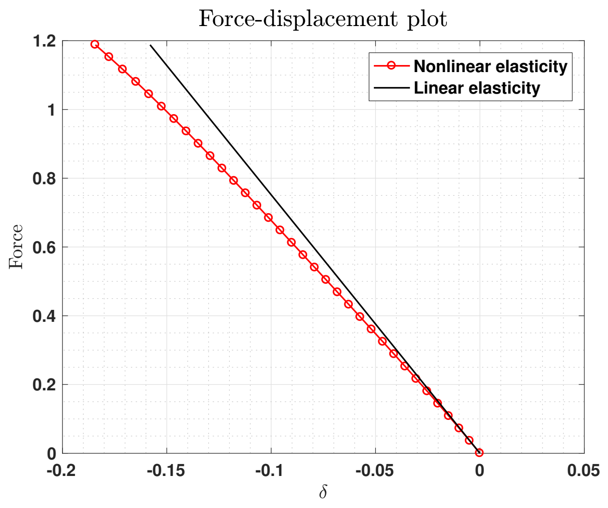
(b) Force-displacement curve

Figure 16: Initial, deformed mesh and force-displacement curve for $\theta = \pi/2$.

Figures 17 and 18 show the results obtained when the fibres are aligned in $\theta = \pi/6$ and $\theta = \pi/4$, respectively. We notice that the fibers behave less stiff as they resist lesser than in the previous case making it a not optimal. This means it is possible to deform the material more in this orientation than in the case where the fibers are oriented orthogonal or parallel to the applied force making these materials an interesting choice in future research.

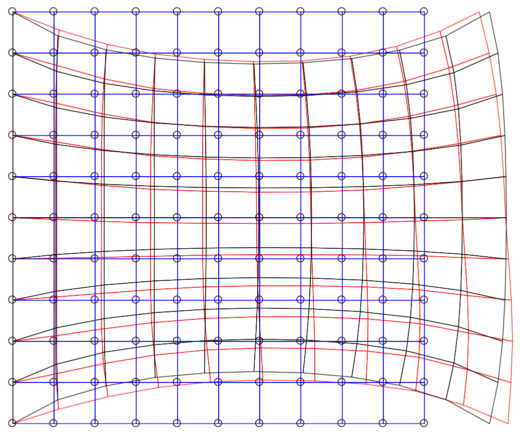


(a) Initial and deformed mesh

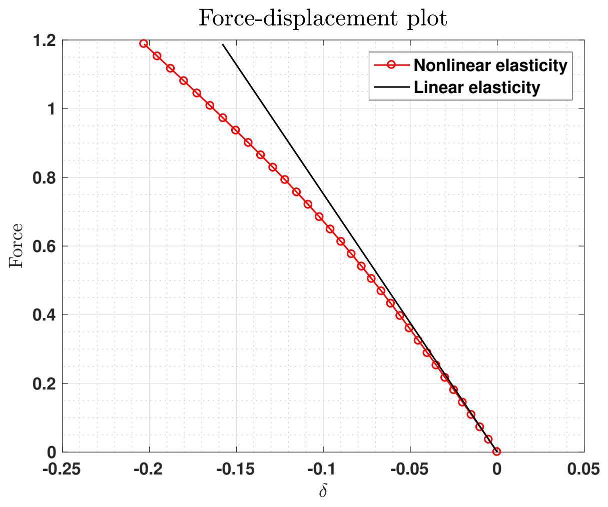


(b) Force-displacement curve

Figure 17: Initial, deformed mesh and force-displacement curve for $\theta = \pi/6$.



(a) Initial and deformed mesh



(b) Force-displacement curve

Figure 18: Initial, deformed mesh and force-displacement curve for $\theta = \pi/4$.

4 Appendix

Kirchhoff Saint-Venant function 1: St_Venant_1

```

function [W]=St_Venant_1(C,lambda,mu,icode)

% Green-Lagrange strain tensor
E = (C-[1 1 0])/2;
trE = E(1) + E(2);
Esq = E.^2;
trEsq = Esq(1) + Esq(2);

% Energy
W = 0.5*lambda*(trE)^2+mu*trEsq;
end
```

Kirchhoff Saint-Venant function 2: St_Venant_2

```

function [W,S]=St_Venant_2(C,lambda,mu,icode)

% Green-Lagrange strain tensor
E = (C-[1 1 0])/2;
trE = E(1)+E(2);
Esq = E.^2;
trEsq = Esq(1)+Esq(2);

% Energy
W = 0.5*lambda*(trE)^2+mu*trEsq;

% Second Piola-Kirchhoff tensor
S = lambda*trE*[1 1 0]+2*mu*E;
end
```

Kirchhoff Saint-Venant function 3: St_Venant_3

```

function [W,S,CC] = St_Venant_3(C,lambda,mu,icode)

% Green-Lagrange strain tensor
E = (C-[1 1 0])/2;
trE = E(1)+E(2);
Esq = E.^2;
trEsq = Esq(1)+Esq(2);

% Energy
```

```
W = 0.5*lambda*(trE)^2+mu*trEsq;

% Second Piola-Kirchhoff tensor
S = lambda*trE*[1 1 0]+2*mu*E;

% Constitutive tensor
CC = zeros(3);
CC(1,1) = lambda+2*mu; CC(1,2) = lambda; CC(1,3) = 0;
CC(2,1) = lambda; CC(2,2) = lambda+2*mu; CC(2,3) = 0;
CC(3,1) = 0; CC(3,2) = 0; CC(3,3) = 2*mu;
end
```

Modified Kirchhoff Saint-Venant function 1: Mod_St_Venant_1

```

function [W]=Mod_St_Venant_1(C,lambda,mu,icode)

% Green-Lagrange strain tensor
E = (C-[1 1 0])/2;
Esq = E.^2;
trEsq = Esq(1) + Esq(2);

% Jacobian
J2 = C(1)*C(2)-C(3)*C(3);
J = sqrt(J2);

% Energy
W = lambda/2*(log(J))^2+mu*trEsq;
end
```

Modified Kirchhoff Saint-Venant function 2: Mod_St_Venant_2

```

function [W,S]=Mod_St_Venant_2(C,lambda,mu,icode)

% Green-Lagrange strain tensor
E = (C-[1 1 0])/2;
Esq = E.^2;
trEsq = Esq(1)+Esq(2);

% Jacobian
J2 = C(1)*C(2)-C(3)*C(3);
J = sqrt(J2);

% Energy
W = lambda/2*(log(J))^2+mu*trEsq;

% Inverse of C
```

```
C_inv = [C(2) C(1) -C(3)]/J2;

% Second Piola-Kirchhoff tensor
S = zeros(1,3);
S(1) = mu*(C(1)-1)+lambda*log(J)*C_inv(1);
S(2) = mu*(C(2)-1)+lambda*log(J)*C_inv(2);
S(3) = mu*(C(3))+lambda*log(J)*C_inv(3);
end
```

Modified Kirchhoff Saint-Venant function 3: Mod_St_Venant_3

```

function [W,S,CC]=Mod_St_Venant_3(C,lambda,mu,icode)

% Green-Lagrange strain tensor
E = (C-[1 1 0])/2;
Esq = E.^2;
trEsq = Esq(1)+Esq(2);

% Jacobian
J2 = C(1)*C(2)-C(3)*C(3);
J = sqrt(J2);

% Energy
W = lambda/2*(log(J))^2+mu*trEsq;

% Inverse of C
C_inv = [C(2) C(1) -C(3)]/J2;

% Second Piola-Kirchhoff tensor
S = zeros(1,3);
S(1) = mu*(C(1)-1)+lambda*log(J)*C_inv(1);
S(2) = mu*(C(2)-1)+lambda*log(J)*C_inv(2);
S(3) = mu*(C(3))+lambda*log(J)*C_inv(3);

% Constitutive tensor
CC = zeros(3,3);
CC(1,1) = 2*mu + lambda*C_inv(1)*C_inv(1)-(2*lambda*log(J))*...
                                 (C_inv(1)*C_inv(1)+C_inv(1)*C_inv(1));

CC(1,2) = lambda*C_inv(2)*C_inv(1)-(2*lambda*log(J))*...
                                 (C_inv(3)*C_inv(3)+C_inv(3)*C_inv(3));

CC(1,3) = lambda*C_inv(3)*C_inv(1)-(2*lambda*log(J))*...
                                 (C_inv(1)*C_inv(3)+C_inv(1)*C_inv(3));

CC(2,2) = 2*mu+ lambda*C_inv(2)*C_inv(2)-(2*lambda*log(J))*...
                                 (C_inv(2)*C_inv(2)+C_inv(2)*C_inv(2));

CC(2,3) = lambda*C_inv(3)*C_inv(2)-(2*lambda*log(J))*...
```

```
                                    (C_inv(2)*C_inv(3)+C_inv(2)*C_inv(3));

CC(3,3) = 2*mu+ lambda*C_inv(3)*C_inv(3)-(2*lambda*log(J))*...
                                    (C_inv(1)*C_inv(2)+C_inv(3)*C_inv(3));
CC(2,1) = CC(1,2);
CC(3,1) = CC(1,3);
CC(3,2) = CC(2,3);
end
```

Line search case included in function Equilibrate.m

```

  case 1,  %Newton-Raphson with line search
    iter=0;
    err_x=100;
    err_f=100;
    [Ener,grad_E,Hess_E] = Ener_short(x_short,3);
    while (iter<=options.n_iter_max) & ...
        ( (err_x>options.tol_x) | ...
        (err_f>options.tol_f))
       iter=iter+1;
      dx = -Hess_E\grad_E;
      % Implemented part for line search option
      if options.linesearch==1
        dir_der=dx'*grad_E;
        if (dir_der)>0 % check if direction is upward
           disp('ascent direction')
           dx=-dx;
           dir_der=-dir_der; % reverse the direction
        end
        [x_short t]=LineSearch(x_short,dx,Ener,dir_der,options);
      else
        t=1;
        x_short=x_short+dx;
      end
      [Ener,grad_E,Hess_E] = Ener_short(x_short,3);
      err_x=abs(t)*norm(dx)/norm(x_short);
      err_f=norm(grad_E);
      err_plot=[err_plot err_x];
      err_plot1=[err_plot1 err_f];
      %fprintf('Iteration %i, errors %e %e \n', iter,err_x,err_f)
    end
    %Check positive definiteness
    if options.info==3
      [V,D] = eig(Hess_E);
      D=diag(D);
      if ((min(D))<=-1e-6*abs(max(D)))
           fprintf('Warning, the Hessian has a negative eigenvalue \n')
      end
    end
```

Transversely isotropic model function 1: transv_isotr_1

```

function [W]=transv_isotr_1(C,c0,c1,kappa,mu,N)

Jsq = C(1)*C(2)-C(3)*C(3); % Jacobian square
J = sqrt(Jsq);
logJ = log(J);
trC = C(1)+C(2); % trace of C

% Volumetric response of the material
G = 0.25*(Jsq-1-2*logJ);

% Fourth invariant
I4 = C(1)*N(1)*N(1)+C(2)*N(2)*N(2)+2*C(3)*N(1)*N(2);

% Energy
W = mu*0.5*(trC-2)-mu*logJ+kappa*G+c0*(exp(c1*(sqrt(I4)-1)^4)-1);
end
```

Transversely isotropic model function 2: transv_isotr_2

```

function [W,S]=transv_isotr_2(C,c0,c1,kappa,mu,N)

Jsq = C(1)*C(2)-C(3)*C(3); % Jacobian square
J = sqrt(Jsq);
logJ = log(J);

trC = C(1)+C(2); % trace of C
C_inv = [C(2) C(1) -C(3)]/Jsq; % Inverse of C
Id = [1 1 0]; % Voigt notation of identity matrix
N_fib_tensor = [N(1)*N(1), N(2)*N(2), N(1)*N(2)];

% Volumetric response of the material
G = 0.25*(Jsq-1-2*logJ);

% Fourth invariant
I4 = C(1)*N(1)*N(1)+C(2)*N(2)*N(2)+2*C(3)*N(1)*N(2);

% Energy
W = mu*0.5*(trC-2)-mu*logJ+kappa*G+c0*(exp(c1*(sqrt(I4)-1)^4)-1);

% Second Piola-Kirchhoff tensor
S = mu*(Id-C_inv)+kappa/2*(Jsq*C_inv-C_inv)+4*c0*c1*...
            ((sqrt(I4)-1)^3/sqrt(I4)*exp(c1*(sqrt(I4)-1)^4))*N_fib_tensor;
end
```

Transversely isotropic model function 3: transv_isotr_3

```

function [W,S,CC]=transv_isotr_3(C,c0,c1,kappa,mu,N)

Jsq = C(1)*C(2)-C(3)*C(3); % Jacobian square
J = sqrt(Jsq);
logJ = log(J);

trC = C(1)+C(2); % trace of C
C_inv = [C(2) C(1) -C(3)]/Jsq; % Inverse of C
Id = [1 1 0]; % Voigt notation of identity matrix
N_fib_tensor = [N(1)*N(1), N(2)*N(2), N(1)*N(2)];

% Derivatives
der_C11=-1/2*(C_inv(1)*C_inv(1)+C_inv(1)*C_inv(1));
der_C22=-1/2*(C_inv(2)*C_inv(2)+C_inv(2)*C_inv(2));
der_C33=-1/2*(C_inv(1)*C_inv(2)+C_inv(3)*C_inv(3));
der_C12=-1/2*(C_inv(3)*C_inv(3)+C_inv(3)*C_inv(3));
der_C13=-1/2*(C_inv(1)*C_inv(3)+C_inv(1)*C_inv(3));
der_C23=-1/2*(C_inv(2)*C_inv(3)+C_inv(2)*C_inv(3));

% Volumetric response of the material
G = 0.25*(Jsq-1-2*logJ);

% Fourth invariant
I4 = C(1)*N(1)*N(1)+C(2)*N(2)*N(2)+2*C(3)*N(1)*N(2);

% Energy
W = mu*0.5*(trC-2)-mu*logJ+kappa*G+c0*(exp(c1*(sqrt(I4)-1)^4)-1);

% Second Piola-Kirchhoff tensor
S = mu*(Id-C_inv)+kappa/2*(Jsq*C_inv-C_inv)+4*c0*c1*...
            ((sqrt(I4)-1)^3/sqrt(I4)*exp(c1*(sqrt(I4)-1)^4))*N_fib_tensor;

% Constitutive tensor
value=(8*c0*c1*((sqrt(I4)-1)^2)/I4)*exp(c1*((sqrt(I4)-1)^4))*...
                (2*c1*((sqrt(I4)-1)^4)+(3/2)-((sqrt(I4)-1)/(2*sqrt(I4))));

CC(1,1)=(-2*mu + kappa*(Jsq-1))*der_C11 + kappa*Jsq*C_inv(1)*C_inv(1) +...
                                 value*N_fib_tensor(1)*N_fib_tensor(1);

CC(2,2)=(-2*mu + kappa*(Jsq-1))*der_C22 + kappa*Jsq*C_inv(2)*C_inv(2) +...
                                 value*N_fib_tensor(2)*N_fib_tensor(2);

CC(3,3)=(-2*mu + kappa*(Jsq-1))*der_C33 + kappa*Jsq*C_inv(3)*C_inv(3) +...
                                 value*N_fib_tensor(3)*N_fib_tensor(3);

CC(1,2)=(-2*mu + kappa*(Jsq-1))*der_C12 + kappa*Jsq*C_inv(2)*C_inv(1) +...
                                 value*N_fib_tensor(1)*N_fib_tensor(2);

CC(1,3)=(-2*mu + kappa*(Jsq-1))*der_C13 + kappa*Jsq*C_inv(1)*C_inv(3) +...
```

```
                                value*N_fib_tensor(1)*N_fib_tensor(3);

CC(2,3)=(-2*mu + kappa*(Jsq-1))*der_C23 + kappa*Jsq*C_inv(2)*C_inv(3) +...
                                value*N_fib_tensor(2)*N_fib_tensor(3);

CC(2,1)=CC(1,2);
CC(3,1)=CC(1,3);
CC(3,2)=CC(2,3);
end
```