FEniCS: Hyperelastic Formulations

Ida Ang: Edited April 11, 2023

Contents

1 Problem Definition

This document is based on the FEniCS hyperelasticity demonstration, which is formulated using the concept of minimization of potential energy. Hyperelastic models are a type of constitutive (stress-strain) model, where the stress and strain are not linearly related as they are in linear elastic models. Hyperelastic constitutive models can be used to describe materials such as rubbers, elastomers, and load bearing musculoskeletal tissues such as cartilage.

The goal of this document is to 1) provide more details of the theory, and 2) provide an alternate formulation where the strong form, or partial differential equation, is converted to the weak form, or integral equation, and written explicitly. This document provides the detailed steps for beginners who might have just been exposed to continuum mechanics and indicial notation. In direct notation, bolded uncapitalized notation refers to vectors, unbolded notation refers to scalar variables.

The original demonstration involves a three-dimensional cube, where a boundary condition is provided that twists the top part of the cube. See Fig. 1 for an example of reference and current configuration.

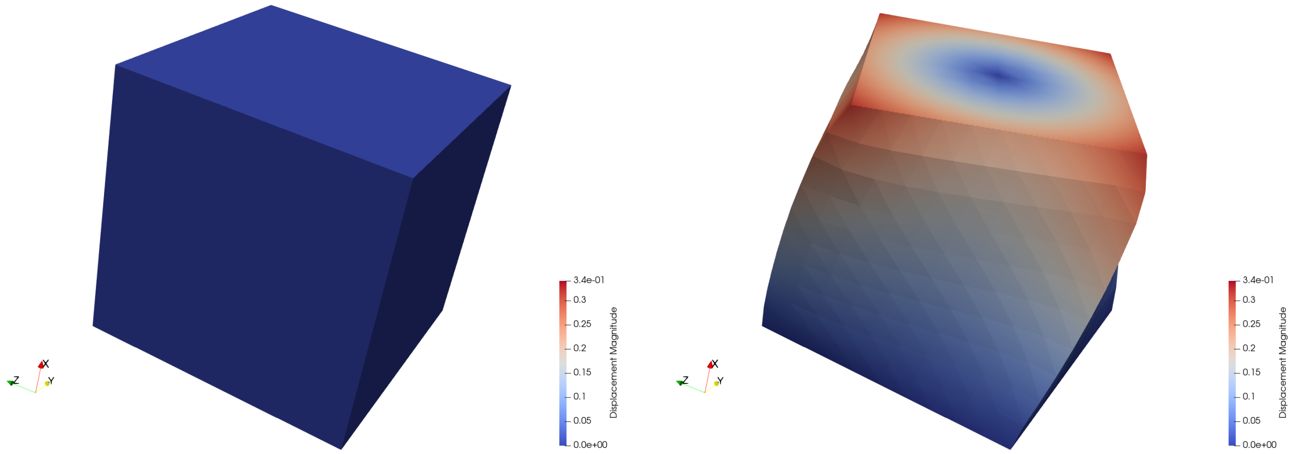

Figure 1: Paraview visualization of reference and final configuration of the cube subject to a twist.

2 Kinematics

The displacement, **u**, is the difference between points in the reference and points in the current configuration, which are denoted by **X** and **x** respectively.

$$\mathbf{u} = \mathbf{X} - \mathbf{x} \tag{1}$$

By taking the gradient in the reference configuration of Eq. 1.

$$\begin{gathered}\frac{\partial \mathbf{u}}{\partial \mathbf{X}} = \frac{\partial \mathbf{X}}{\partial \mathbf{X}} - \frac{\partial \mathbf{x}}{\partial \mathbf{X}} \\ \nabla \mathbf{u} = \mathbf{I} - \mathbf{F}\end{gathered}$$

Therefore, the deformation gradient **F** maps the reference configuration to the current configuration.

$$\mathbf{F} = \frac{\partial \mathbf{x}}{\partial \mathbf{X}} \rightarrow F_{ij} = \frac{\partial x_i}{\partial X_j} \tag{2}$$

and the following relationship is given, where **I** is the identity tensor.

$$\mathbf{F} = \mathbf{I} + \nabla \mathbf{u} \tag{3}$$

The right Cauchy Green (CG) tensor **C** is defined as follows

$$\mathbf{C} = \mathbf{F}^T \mathbf{F} \tag{4}$$

The invariants of the right CG tensor are defined as follows.

$$\begin{aligned} I_{\mathbf{C}} &= \operatorname{tr} \mathbf{C} \\ II_{\mathbf{C}} &= \frac{1}{2}(C_{kk}^2 - C_{ik}C_{ki}) \\ III_{\mathbf{C}} &= \det \mathbf{C} \end{aligned}$$

Therefore, we can characterize the volumetric ratio between the current and reference configuration as, where we identify a relationship.

$$J = \det \mathbf{F} \rightarrow J^2 = III_{\mathbf{C}} \tag{5}$$

for an incompressible material, $J = \det \mathbf{F} = 1$. A constraint on incompressible constitutive models involves

$$J - 1 = 0 \tag{6}$$

3 Stress Relationships

To calculate the first Piola Kirchoff (PK) stress, where $W(\mathbf{F}) = \widehat{W}(\mathbf{C})$

$$P_{ij} = \frac{\partial W}{\partial F_{ij}} \leftrightarrow \mathbf{P} = \frac{\partial W(\mathbf{F})}{\partial \mathbf{F}} \tag{7a}$$

$$P_{ij} = 2F_{ik}\frac{\partial \widehat{W}}{\partial C_{jk}} \leftrightarrow \mathbf{P} = 2\mathbf{F}\frac{\partial \widehat{W}(\mathbf{C})}{\partial \mathbf{C}} \tag{7b}$$

The Cauchy stress refers to a stress that characterizes the current or deformed configuration.

$$\sigma_{ij} = \frac{1}{J}F_{ik}\frac{\partial W(\mathbf{F})}{\partial F_{jk}} \tag{8a}$$

$$\sigma_{ij} = \frac{2}{J}F_{ip}F_{jq}\frac{\partial \widehat{W}(\mathbf{C})}{\partial C_{pq}} = \frac{1}{J}F_{ip}P_{jp} \tag{8b}$$

We note the relationship between the Cauchy stress and the 1st Pk stress.

$$\sigma_{ij} = \frac{1}{J}F_{ik}P_{jk} \leftrightarrow \boldsymbol{\sigma} = \frac{1}{J}\mathbf{F}\mathbf{P}^{\mathbf{T}}$$

Where we note that $\boldsymbol{\sigma} = \boldsymbol{\sigma}^{\boldsymbol{T}}$ because of symmetry, implying that $\mathbf{P}\mathbf{F}^{\mathbf{T}} = \mathbf{F}\mathbf{P}^{\mathbf{T}}$; therefore

$$\boldsymbol{\sigma} = \frac{1}{J}\mathbf{P}\mathbf{F}^{\mathbf{T}} \leftrightarrow \sigma_{ij} = \frac{1}{J}P_{ik}F_{jk} \tag{9}$$

This implies that we can also write the 1st PK stress as

$$\mathbf{P} = J\boldsymbol{\sigma}\mathbf{F}^{-\mathbf{T}} \tag{10}$$

Using the chain rule, we can obtain a generalized expression for the 1st PK stress

$$P_{ij} = 2F_{ik}\left[\frac{\partial \widehat{W}}{\partial I_{\mathbf{C}}}\frac{\partial I_{\mathbf{C}}}{\partial C_{kj}} + \frac{\partial \widehat{W}}{\partial II_{\mathbf{C}}}\frac{\partial II_{\mathbf{C}}}{\partial C_{kj}} + \frac{\partial \widehat{W}}{\partial III_{\mathbf{C}}}\frac{\partial III^{\mathbf{C}}}{\partial C_{kj}}\right] \tag{11}$$

For some special cases where $J = \det \mathbf{F}$, we can state this instead as:

$$P_{jp} = 2F_{jk}\left[\frac{\partial \widehat{W}}{\partial I_{\mathbf{C}}}\frac{\partial I_{\mathbf{C}}}{\partial C_{kp}} + \frac{\partial \widehat{W}}{\partial II_{\mathbf{C}}}\frac{\partial II_{\mathbf{C}}}{\partial C_{kp}}\right] + \frac{\partial W}{\partial J}\frac{\partial J}{\partial F_{jp}} \tag{12}$$

$$\sigma_{ij} = \frac{1}{J}F_{ip}P_{jp} \tag{13}$$

The symmetric second Piola-Kirchoff stress tensor is denoted by $\mathbf{S}$

$$\mathbf{P} = \mathbf{F}\mathbf{S} \leftrightarrow P_{ij} = F_{ik}S_{kj} \tag{14}$$

Where we know further that

$$\mathbf{S} = 2\frac{\partial \widehat{W}}{\partial \mathbf{C}} \tag{15}$$

The following relationship holds, where $W(\mathbf{F}) = \widehat{W}(\mathbf{C})$

$$\frac{\partial W}{\partial \mathbf{F}} = 2\mathbf{F}\frac{\partial \widehat{W}}{\partial \mathbf{C}} \to \frac{\partial W}{\partial F_{ij}} = 2F_{ik}\frac{\partial \widehat{W}}{\partial C_{kj}}$$

An important relationship is included

$$\frac{\partial \det \mathbf{F}}{\partial \mathbf{F}} = \det \mathbf{F}\mathbf{F}^{-\mathbf{T}} = J\mathbf{F}^{-\mathbf{T}} \tag{16}$$

4 Strain Energy Density Function

The next section outlines some common strain energy density functions. Note that μ and λ are Lamé Parameters and can be defined in terms of the Young's modulus, E, and Poisson's ratio ν

$$\mu = \frac{E}{2(1+\nu)} \tag{17a}$$

$$\lambda = \frac{E\nu}{(1+\nu)(1-2\nu)} \tag{17b}$$

4.1 Compressibility

The energy density function for compressible Neo-Hookean materials is:

$$W(\mathbf{F}) = \frac{\mu}{2}(I_{\mathbf{C}} - 3 - 2\ln J) + \frac{\lambda}{2}(\ln J)^2 \tag{18}$$

In order to obtain the 1st PK stress, we can utilize Eq. 12 and use Eq. 16

$$\begin{aligned}
\mathbf{P} &= 2\mathbf{F}\left[\frac{\partial\widehat{W}}{\partial I_{\mathbf{C}}}\frac{\partial I_{\mathbf{C}}}{\partial\mathbf{C}} + \frac{\partial\widehat{W}}{\partial II_{\mathbf{C}}}\frac{\partial II_{\mathbf{C}}}{\partial\mathbf{C}}\right] + \frac{\partial W}{\partial J}\frac{\partial J}{\partial\mathbf{F}} \\
&= 2\mathbf{F}\frac{\partial\widehat{W}}{\partial I_{\mathbf{C}}}\frac{\partial I_{\mathbf{C}}}{\partial\mathbf{C}} + \frac{\partial W}{\partial J}\frac{\partial J}{\partial\mathbf{F}} \\
&= 2\mathbf{F}\frac{\mu}{2}\frac{\partial I_{\mathbf{C}}}{\partial\mathbf{C}} + \left[\frac{\mu}{2}\frac{\partial(-2\ln J)}{\partial J} + \frac{\lambda}{2}\frac{\partial(\ln J)^2}{\partial J}\right]\frac{\partial J}{\partial\mathbf{F}} \\
&= \mu\mathbf{F}\mathbf{I} + \left[\frac{\mu}{2}\frac{\partial(-2\ln J)}{\partial J} + \frac{\lambda}{2}\frac{\partial(\ln J)^2}{\partial J}\right]\frac{\partial J}{\partial\mathbf{F}} \\
&= \mu\mathbf{F} + \left[\frac{\mu}{2}\frac{-2}{J} + \frac{\lambda}{2}2(\ln J)\frac{1}{J}\right]J\mathbf{F}^{-\mathbf{T}} \\
&= \mu\mathbf{F} + \big[-\mu + \lambda(\ln J)\big]\mathbf{F}^{-\mathbf{T}}
\end{aligned}$$

Therefore, the 1st Piola-Kirchoff stress in the compressible case is given as,

$$\mathbf{P} = \mu(\mathbf{F} + \mathbf{F}^{-\mathbf{T}}) + \lambda(\ln J)\mathbf{F}^{-\mathbf{T}} \tag{19}$$

4.2 Incompressibility

For an incompressible material, $J = \det\mathbf{F} = 1$; therefore, Eq. 18 can be modified to:

$$\begin{aligned}
W(\mathbf{F}) &= \frac{\mu}{2}(I_{\mathbf{C}} - 3 - 2\ln(1)) + \frac{\lambda}{2}(\ln(1))^2 \\
W(\mathbf{F}) &= \frac{\mu}{2}(I_{\mathbf{C}} - 3)
\end{aligned}$$

The 1st PK associated with this equation is simply

$$\mathbf{P} = \mu\mathbf{F} \tag{20}$$

One way to enforce incompressibility is to use the penalty or lagrange method of enforcing the constraint.

4.2.1 Penalty Method

The penalty method enforces the constraint in Eq. 6 with a penalty parameter k_{pen}.

$$W(\mathbf{F}) = \frac{\mu}{2}(I_{\mathbf{C}} - 3) + \frac{k_{pen}}{2}(J-1)^2 \tag{21}$$

This is a simple implementation that does not require expansion of the solution sub-space to include a pressure variable as in the Lagrange method, but is dependent on the penalty parameter for enforcement of the constraint. If the penalty parameter is too high, the stiffness matrix can become ill-conditioned.

In order to obtain the 1st PK stress, we can utilize Eq. 12 and use Eq. 16

$$\begin{aligned}
\mathbf{P} &= 2\mathbf{F}\frac{\partial\widehat{W}}{\partial I_{\mathbf{C}}}\frac{\partial I_{\mathbf{C}}}{\partial\mathbf{C}} + \frac{\partial W}{\partial J}\frac{\partial J}{\partial\mathbf{F}} \\
&= 2\mathbf{F}\frac{\mu}{2}\frac{\partial I_{\mathbf{C}}}{\partial\mathbf{C}} + k_{pen}(J-1)\frac{\partial J}{\partial\mathbf{F}} \\
&= \mu\mathbf{F}\mathbf{I} + k_{pen}(J-1)J\mathbf{F}^{-\mathbf{T}}
\end{aligned}$$

Therefore, the 1st PK stress in the Lagrange case of enforcing incompressibility is given as,

$$\mathbf{P} = \mu\mathbf{F} + k_{pen}(J-1)J\mathbf{F}^{-\mathbf{T}} \tag{22}$$

4.2.2 Lagrange Method

To enforce incompressibility, introduce a Lagrange multiplier, p, which can be assumed to be a pressure-like term.

$$W(\mathbf{F}) = \frac{\mu}{2}(I_{\mathbf{C}} - 3) + \mathrm{p}(J-1) \tag{23}$$

This is an exact method of enforcing incompressibility, but it requires expansion of the solution subspace to a displacement-pressure formulation. Therefore, Taylor-Hood elements have to be used for numerical stability.

Again, to obtain the 1st PK stress, we can use Eq. 12

$$\begin{aligned}\mathbf{P} &= 2\mathbf{F}\frac{\partial\widehat{W}}{\partial I_{\mathbf{C}}}\frac{\partial I_{\mathbf{C}}}{\partial\mathbf{C}} + \frac{\partial W}{\partial J}\frac{\partial J}{\partial\mathbf{F}} \\ &= 2\mathbf{F}\frac{\mu}{2}\frac{\partial I_{\mathbf{C}}}{\partial\mathbf{C}} + k_{pen}(J-1)\frac{\partial J}{\partial\mathbf{F}} \\ &= \mu\mathbf{F}\mathbf{I} + pJ\mathbf{F}^{-\mathbf{T}}\end{aligned}$$

Therefore,

$$\mathbf{P} = \mu\mathbf{F} + \mathrm{p}\mathbf{J}\mathbf{F}^{-\mathbf{T}} \tag{24}$$

5 Formulation

5.1 Potential Energy Minimization

An alternative approach to solving static problems is to consider the minimization of potential energy.

$$\min_{\mathbf{u}\in V}\Pi$$

where V is a suitable function space that satisfies the boundary conditions on the displacement $\mathbf{u}$. The total potential energy, Π, is given by the sum of the internal and external energy:

$$\begin{aligned}\Pi &= \Pi_{int} + \Pi_{ext} \\ &= \Big(\int_\Omega W(\mathbf{u})dx\Big) + \Big(-\int_\Omega \mathbf{B}\cdot\mathbf{u}\,dx - \int_{\partial\Omega}\mathbf{T}\cdot\mathbf{u}\,ds\Big)\end{aligned} \tag{25}$$

where W is the elastic stored energy density, $\mathbf{B}$ is the body force per unit reference volume and $\mathbf{T}$ is a traction force per unit reference area.

Minimization of the potential energy corresponds to the directional derivative of Π being zero for all possible variations of u. (Note, minimizing Π is equivalent to solving the balance of momentum problem.)

$$L(\mathbf{u};\mathbf{v}) = D_{\mathbf{v}}\Pi = \frac{d\Pi(\mathbf{u}+\boldsymbol{\epsilon}\mathbf{v})}{d\boldsymbol{\epsilon}}\Big|_{\boldsymbol{\epsilon}=0} = 0 \qquad \forall\mathbf{v}\in V \tag{26}$$

If we use Newton's method, we also want to find the Jacobian of Eq. 26

$$a(\mathbf{u};d\mathbf{u},\mathbf{v}) = D_{d\mathbf{u}}L = \frac{dL(\mathbf{u}+\boldsymbol{\epsilon}\,d\mathbf{u};\mathbf{v})}{d\boldsymbol{\epsilon}}\Big|_{\boldsymbol{\epsilon}=0} \tag{27}$$

5.2 Strong to Weak Form

This is another notation commonly used, where a capital index is used to indicate the referential configuration and a lowercase index is used to indicate the current configuration. The 1st PK stress is a tensor that represents both the referential and the current configuration, but the 2nd PK stress is a tensor purely in the referential configuration. The mechanical equilibrium equation is given as:

$$\nabla_X \mathbf{P} + \mathbf{B} = \mathbf{0} \quad \text{in } \Omega_0 \tag{28a}$$

$$\mathbf{u} = \mathbf{u_0} \quad \text{on } \Gamma_D \tag{28b}$$

$$\mathbf{P} \cdot \mathbf{N} = \mathbf{T} \quad \text{on } \Gamma_N \tag{28c}$$

where the Dirichlet and Neumann boundary conditions are given on there respective boundaries, and the body force and traction force are defined as $\mathbf{B}$ and $\mathbf{T}$ respectively. The mechanical equilibrium equation can be written in indicial notation:

$$P_{iJ,J} + B_i = 0 \rightarrow \frac{\partial P_{iJ}}{\partial X_J} + B_i = 0 \tag{29}$$

Take Eq. 29 and multiply by a test function, $\mathbf{v} = v_i \mathbf{e_i}$

$$\begin{aligned} \frac{\partial P_{iJ}}{\partial X_J} \mathbf{e_i} \cdot v_j \mathbf{e_j} &= -B_i \mathbf{e_i} \cdot v_j \mathbf{e_j} \\ \frac{\partial P_{iJ}}{\partial X_J} v_j \delta_{ij} &= -B_i v_j \delta_{ij} \\ \frac{\partial P_{iJ}}{\partial X_J} v_i &= -B_i v_i \quad \text{Integrate over domain} \\ \int_{\Omega_o} \frac{\partial P_{iJ}}{\partial X_J} v_i dV &= -\int_{\Omega_o} B_i v_i dV \end{aligned} \tag{30}$$

Use integration by parts:

$$\begin{aligned} (fg)' = f'g + fg' \rightarrow f'g &= (fg)' - fg' \\ \frac{\partial P_{iJ}}{\partial X_j} v_i &= (P_{iJ} v_i)_{,J} - P_{iJ} v_{i,J} \end{aligned}$$

Substitute into Eq. 30:

$$\begin{aligned} \int_{\Omega_o} (P_{iJ} v_i)_{,J} dV - \int_{\Omega_o} P_{iJ} v_{i,J} dV &= -\int_{\Omega_o} B_i v_i dV \quad \text{Use divergence theorem} \\ \int_{\partial\Omega_o} P_{iJ} N_J v_i dS - \int_{\Omega_o} P_{iJ} v_{i,J} dV &= -\int_{\Omega_o} B_i v_i dV \quad \text{Recognize traction } P_{iJ} N_J = T_i \\ \int_{\partial\Omega_o} T_i v_i dS - \int_{\Omega_o} P_{iJ} v_{i,J} dV &= -\int_{\Omega_o} B_i v_i dV \quad \text{Rearrange} \\ \int_{\Omega_o} P_{iJ} v_{i,J} dV &= \int_{\Omega_o} B_i v_i dV + \int_{\partial\Omega_o} T_i v_i dS \end{aligned}$$

Rewrite in direct notation to obtain weak form:

$$\int_{\Omega_o} \mathbf{P} : \mathrm{Grad}(\mathbf{v})\, dV = \int_{\Omega_o} \mathbf{B} \cdot \mathbf{v}\, dV + \int_{\Gamma_N} \mathbf{T} \cdot \mathbf{v}\, dS \tag{31}$$

Finally, in order to enforce incompressibility we use the constraint equation Eq. 6, where we can multiply this with a scalar test function

$$J - 1 = 0$$

$$\int_{\Omega} (J-1)\tau \, dV = 0.$$

Therefore the weak forms are as listed in Eq. 32

$$\int_{\Omega_o} \mathbf{P} : \mathrm{Grad}(\mathbf{v}) \, dV = \int_{\Omega_o} \mathbf{B} \cdot \mathbf{v} \, dV + \int_{\Gamma_N} \mathbf{T} \cdot \mathbf{v} \, dS \tag{32a}$$

$$\int_{\Omega_o} (J-1)\tau \, dV = 0 \tag{32b}$$

The bilinear and linear form can be stated where we have two test functions

$$a\big((\mathbf{u}, p), (\mathbf{v}, \tau)\big) = L\big((\mathbf{v}, \tau)\big) \quad \forall (\mathbf{v}, \tau) \in \Sigma_0 \times V$$

Therefore with Eq. 32, we have the bilinear and linear forms:

$$a\big((\mathbf{u}, p), (\mathbf{v}, \tau)\big) = \int_{\Omega_o} \big(\mathbf{P} : \mathrm{Grad}(\mathbf{v}) + (J-1)\tau\big) \, dV \tag{33}$$

$$L\big((\mathbf{v}, \tau)\big) = \int_{\Omega_o} \mathbf{B} \cdot \mathbf{v} \, dV + \int_{\partial\Omega_o} \mathbf{T} \cdot \mathbf{v} \, dS \tag{34}$$

The bilinear and linear form (Eq. 33 and 34) can also be combined into one statement, where we solve for two unknowns, displacement (**u**) and hydrostatic pressure (p).

$$a\big((\mathbf{u}, p), (\mathbf{v}, \tau)\big) - L\big((\mathbf{v}, \tau)\big) = \int_{\Omega_o} \big(\mathbf{P} : \mathrm{Grad}(\mathbf{v}) + (J-1)\tau\big) \, dV - \int_{\Omega_o} \mathbf{B} \cdot \mathbf{v} \, dV - \int_{\partial\Omega_o} \mathbf{T} \cdot \mathbf{v} \, dS \tag{35}$$

6 FEniCS Implementation

There are three codes provided, the first one demonstrates the energy minimization formulation for the displacement only formulation, with two choices of model, 1) a compressible neo-Hookean model and 2) an incompressible penalty formulation. The second code demonstrates a penalty formulation with an explicit weak form. The third code demonstrates a Lagrange formulation with an explicit weak form.

```
HyperelasticEnergy.py
HyperelasticDispForm.py
HyperelasticMixedForm.py
```

The domain is a unit cube where the x-direction faces towards the right and the y-direction is upwards by default in Paraview

$$\Omega = (0,1) \times (0,1) \times (0,1)$$

The left hand side is completely fixed, while a twist is provided to the right side of the cube. The function for the twist is specified in Eq. 36, which is why y and z direction are specified and the x-direction is zero.

$$\mathbf{u} = \left[0, s\bigg(\frac{1}{2} + (y - \frac{1}{2})\cos\theta - (z - \frac{1}{2})\sin\theta - y\bigg), s\bigg(\frac{1}{2} + (y - \frac{1}{2})\sin\theta - (z - \frac{1}{2})\cos\theta - z\bigg)\right] \tag{36}$$

Certain parameters of this equation can be increased over the course of the simulation, s is a scaling parameter, and θ is the angle of the twist provided. The first parameters provided

```
r = Expression(("scale*0.0",
   "scale*(.5 + (x[1] - .5)*cos(theta) - (x[2] - .5)*sin(theta) - x[1])",
   "scale*(.5 + (x[1] - .5)*sin(theta) + (x[2] - .5)*cos(theta) - x[2])"),
   scale = 0, theta = 0, degree=2)
```

where `scale` and `theta` can be adjusted in the loop according to

```
RampArray = np.linspace(0, 0.5, TotSteps)
ThetaArray = np.linspace(0, pi/3, TotSteps)
```

Note the general way to declare aspects of the expression

```
Expression("x+y", x = #, y = #, degree = #)
```

The behavior of the FEniCS Form Compiler, FFC, can be adjusted by prescribing various parameters. Use the UFL Analyzer and Compiler System, UFLACS, which is the backend of FFC. Changes were made to parse in user parameters:

```
parameters.parse()
UserPar = Parameters("user")
UserPar.add("TotSteps", 20)
UserPar.parse()
```

From there, the following syntax can be used to define the parameter:

```
TotSteps = UserPar["TotSteps"]
```

Some parameters can be defined using the `Constant` syntax, which is used in order to avoid re-generation of C++ code when changing model parameters.

```
mu = Constant(E/(2*(1 + nu)))
```

Create a unit cube with 25 (24 + 1) vertices in one direction and 17 (16 + 1) vertices in the other two directions:

```
mesh = UnitCubeMesh(24, 16, 16)
```

Note: C++ syntax is used in the ‘CompiledSubDomain‘ function since the function will be automatically compiled into C++ code for efficiency.

```
left = CompiledSubDomain("near(x[0], side) && on_boundary", side = 0.0)
```

The Dirichlet (essential) boundary conditions are constraints on the function space V. Use the following definitions for the boundary conditions, where we have Dirichlet boundary conditions on the left and right surface of the cube and a potential Neumann boundary condition dependent on whether traction is specified.

$$\Gamma_{D_0} = 0 \times (0,1) \times (0,1)$$

$$\Gamma_{D_1} = 1 \times (0,1) \times (0,1)$$

6.1 Displacement Formulation

The displacement formulation only requires the definition of a function space with continuous piecewise linear vector polynomials. Note that VectorFunctionSpace creates a function space of vector fields. The dimension of the vector field (the number of components) is assumed to be the same as the spatial dimension, unless otherwise specified.

```
V = VectorFunctionSpace(mesh, "Lagrange", 1)
```

As another option, an additional tensor space, where DG = Discontinuous Galerkin, can be defined which is for projection of the stress. DG0 space refers to a scalar space for projection of $J = \det \mathbf{F}$ to determine how closely incompressibility is being maintained.

```
T_DG0 = TensorFunctionSpace(mesh, "DG", 0)
DG0 = FunctionSpace(mesh, "DG", 0)
```

In the `HyperelasticEnergy.py` file, two strain energy density equations are defined based on Eq. 18 and Eq. 21, where the user parameter `Model` determines which equation is used (same for the equations for the 1st PK Stress). The total potential energy can be set up as follows,

```
Pi = psi(Ic,J)*dx - dot(B, u)*dx - dot(T, u)*ds
WF = derivative(Pi, u, v)
Jc = derivative(WF, u, du)
```

In `HyperelasticDispForm`, the weak form is written explicitly for the penalty method of enforcing incompressibility. For the first term, the algorithm will estimate the polynomial degree and use many Gauss points to approximate the integral. Using the line `(metadata="quadrature_degree": 4)` allows us to specify the number of Gauss points explicitly. In this case, the weak form looks like:

```
WF = (inner(P(F), grad(v)))*dx(metadata="quadrature_degree":  4)
     - dot(B, v)*dx - dot(T, v)*ds
```

A for loop is set up to increase the `scale` and `theta` parameter within the expression. Read in the `numpy` library to create a linearly spaced array.

```
RampArray = np.linspace(0, 0.5, TotSteps)
ThetaArray = np.linspace(0, pi/3, TotSteps)
for (Step, Scale) in enumerate(RampArray):
  solve(WF == 0, u, bcs, J=Jc)
  r.scale = Scale
  r.theta = ThetaArray[Step]
```

Post-processing is also carried out where parameters of interest are saved to a text file. In this case, we save the step number, the scale parameter in the expression, and each direction of displacement at the point, (1,1,1).

```
PostProc = np.zeros((TotSteps, 5))
for (Step, Scale) in enumerate(RampArray):
  PostProc[Step] = np.array([Step, Scale,
                             u(1,1,1)[1], u(1,1,1)[2], DetF(1,1,1) ]
  np.savetxt(SaveDir + SimPar + '/PostProc.txt', PostProc)
```

Saving variables like this to a text file can allow for easy comparison of displacements (or other parameters of interest).

6.2 Mixed Formulation

There are two unknowns in our formulation, displacement and hydrostatic pressure, $\mathbf{u}$, p Therefore, define a vector space of degree 2 for displacement and degree 1 for pressure.

```
V_CG2 = VectorElement("Lagrange", mesh.ufl_cell(), 2)
V_CG1 = FiniteElement("Lagrange", mesh.ufl_cell(), 1)
```

Taylor-Hood elements, `TH`, are elements defined where one vector space is one degree higher than the other. These type of elements are stable for this formulation. In our case, the vector space for displacement is one degree higher than that of pressure:

```
TH = MixedElement([V_CG2, V_CG1])
V = FunctionSpace(mesh, TH)
```

The only difference from the displacement formulation involves splitting the test functions and the function into the displacement and pressure. Note that this is a shallow copy not a deep copy. An example of a deep copy is shown within the for loop.

```
(v_u, v_p) = split(v)
(u, p) = split(w)
```

The weak form has the additional equation included for enforcing incompressibility. Note that body force and traction are constants, the automatic inference algorithm will use 1 Gauss point to represent those integrals. If we don't specify the quadrature degree, a warning will appear about computational time:

```
WF = (inner(P(F), grad(v_u))
     + inner(J-1., v_p))*dx(metadata="quadrature_degree":  4)
     - dot(B, v_u)*dx - dot(T, v_u)*ds
```

In this specific formulation, different solver parameters are setup.

```
J_o = derivative(WF, w, du)
varproblem = NonlinearVariationalProblem(WF, w, bcs, J=J_o)
for (Step, Scale) in enumerate(RampArray):
   solver.solve()
   (u, p) = w.split()
```

This split function is a deep copy, which is important for saving the solution for visualization. A post-processing file is also created in the same manner as the other codes to provide consistency in results.

7 Results

The code `PostProc.py` accesses the text files of interest. The displacement in the y and z direction at point $[1, 1, 1]$ is tracked and plotted next to the expression applied through a Dirichlet boundary condition. The displacement results for all these formulations, in order, are given in Fig. 2. The comparison of the displacement results from all the formulations shows that they produce results consistent to each others. They also display some deviation from the applied expression.

While the displacement results are consistent, the incompressibility parameter $J = det\mathbf{F}$ is not necessarily enforced at $J = 1$. First, we can examine the results from `HyperelasticDispForm.py` plotted in Fig. 3A, where we can see how incompressibility is more closely maintained at 1

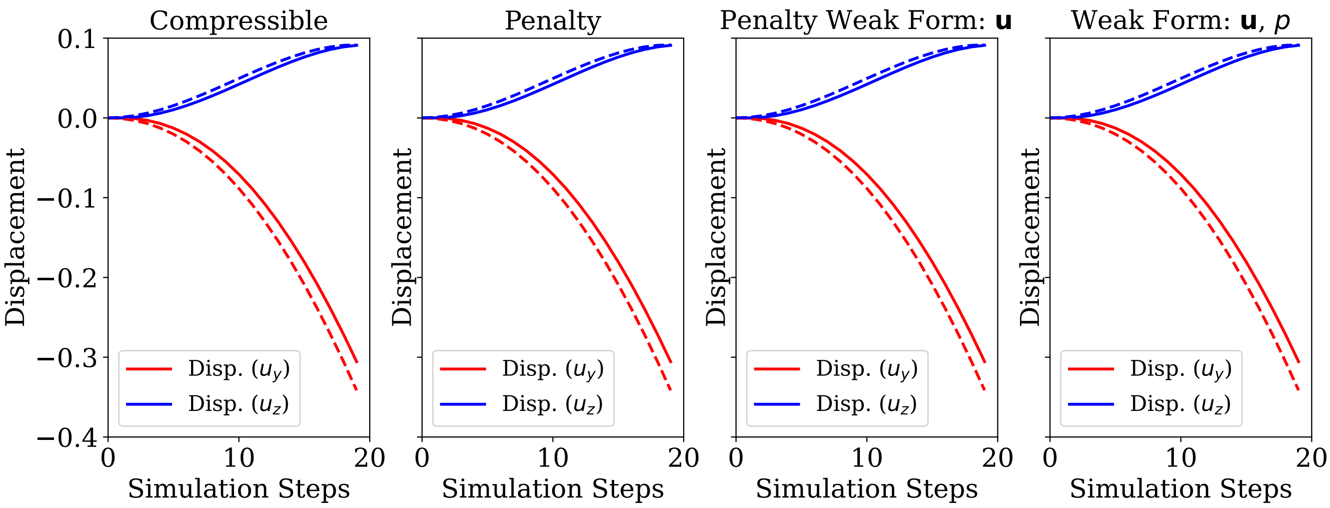

Figure 2: Expression applied is demonstrated with a dashed line while displacement results are displayed with a solid line. A) Comparison of results from HyperelasticEnergy.py where different constitutive models are applied in a displacement-only formulation. B) Comparison of displacement (penalty) and displacement pressure (Lagrange) formulation results where the weak form is explicitly defined, where $\nu = 0.49$.

as the Poisson's ratio increases. Next, we can compare this result to the energy minimization formulation in Fig. 3B, where we see identical results for the same mesh parameters, $N_x = N_y = N_z = 10$

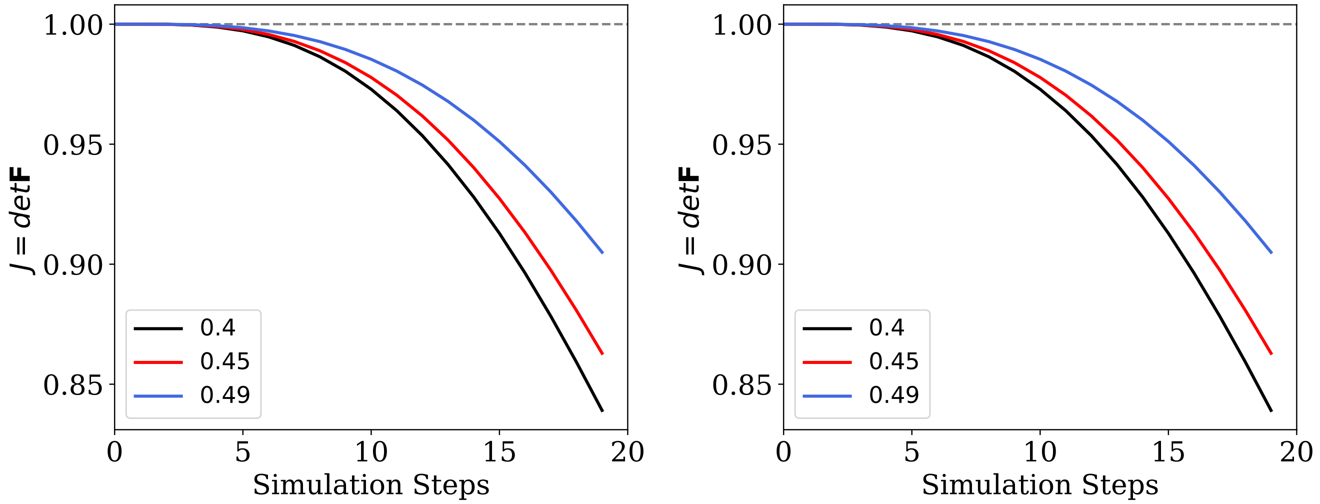

Figure 3: Incompressibility results at point (1,1,1) of the cube. A) Energy formulation B) Plotted for varying Poisson's ratio for the penalty formulation with explicit weak form. Higher Poisson's ratio leads to a higher bulk modulus which maintains the incompressibility constraint more strictly. B) Comparison of incompressibility constraints for different formulations. First two listed work by potential energy minimization, and last two listed contain the explicit weak form statement.

Fig. 4 demonstrates the evolution of the incompressibility constraint where $\nu = 0.49$ among the different formulations. As expected, the compressible formulation does not enforce $J = 1$, and the Lagrange formulation holds the incompressibility constraint the most strictly.

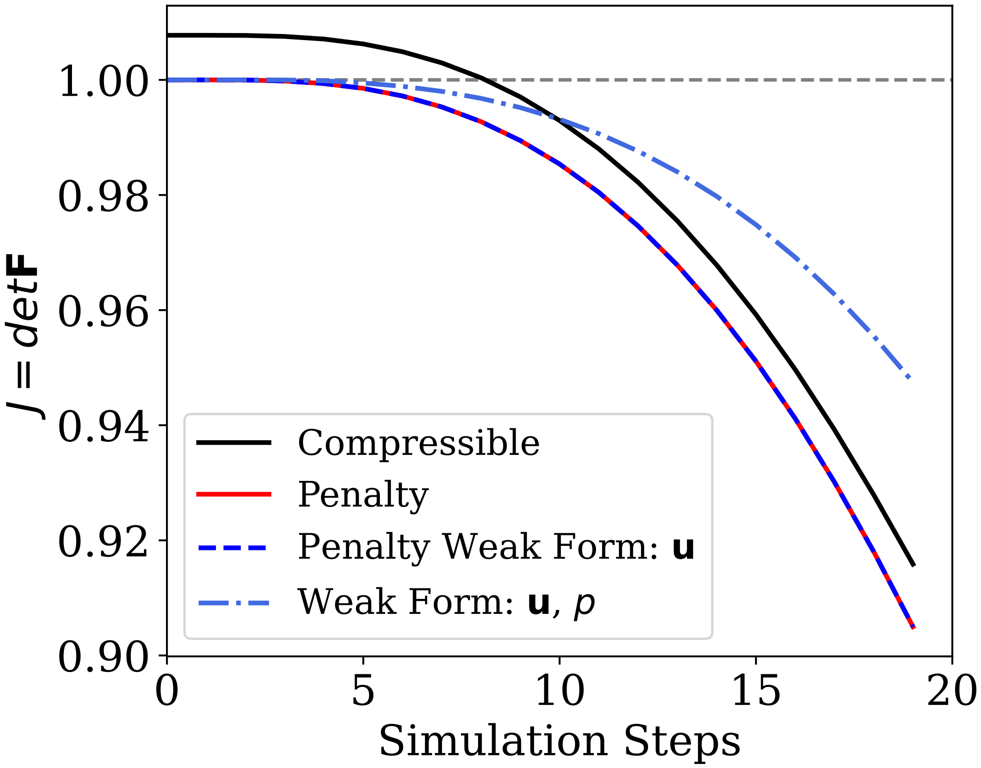

Figure 4: Comparison of incompressibility constraints for different formulations. First two listed work by potential energy minimization, and last two listed contain the explicit weak form statement.