1 Theory

The material model being implemented is a version of viscoelasticity.

The general formulation is as follows:

$$
\begin{aligned}
\mathcal{F}\left(I_1, I_2, J, B, U\right) &= 2\left[\frac{1}{J^{2/3}}\left(\frac{\partial U}{\partial I_1} + I_1\frac{\partial U}{\partial I_2}\right)B_{ij} - \frac{I_1}{3}\frac{\partial U}{\partial I_1}\delta_{ij} - \frac{1}{J^{4/3}}\frac{\partial U}{\partial I_2}B_{ik}B_{kj}\right] \\
\tau_{ij} &= \mathcal{F}\left(\bar{I}_1, \bar{I}_2, J, B, U_\infty\right) + \mathcal{F}\left(\bar{I}_1^e, \bar{I}_2^e, J_e, B^e, U_T\right) + \frac{\partial U_\infty}{\partial J}\delta_{ij} + \frac{\partial U_T}{\partial J_e}\delta_{ij} \\
J &= J_e \\
\tau_{ij} &= \mathcal{F}\left(\bar{I}_1, \bar{I}_2, J, B, U_\infty\right) + \mathcal{F}\left(\bar{I}_1^e, \bar{I}_2^e, J, B^e, U_T\right) + \frac{\partial U_\infty}{\partial J}\delta_{ij} \\
\tau_{ij}^D &= \mathcal{F}\left(\bar{I}_1^e, \bar{I}_2^e, J, B^e, U_T\right) \\
\tau_e &= \sqrt{\frac{3}{2}\tau_{ij}^D\tau_{ij}^D} \\
\dot{\varepsilon}_{ij}^p &= \dot{\varepsilon_e}\left(\tau_e, I_1^e, I_2^e, T\right)\frac{3\tau_{ij}^D}{2\tau_e} \\
\dot{\varepsilon}_e &= \dot{\varepsilon}_0\left(\sqrt{I_1^e} - \sqrt{3}\right)^n\left(\frac{\tau_e}{\tau_0}\right)^m
\end{aligned}
$$

The material properties here are: $\dot{\varepsilon}_0 > 0$, $-1 < n < 0$, $\tau_0 > 0$, and $m > 0$.

In order for this formulation to mean something, however, a form needs to be chosen for both U_∞ and U_T. Usually, it is wise to select the same form for both with different constants.

In this project, I decided to implement viscoelasticity with the Arruda-Boyce potential. Thus, the updated (and final) formulation is below.

Material properties: μ_∞, β_∞, μ_T, β_T, K, $\dot{\varepsilon}_0 > 0$, $-1 < n < 0$, $\tau_0 > 0$, $m > 0$

$$
\begin{aligned}
U_\infty &= \mu_\infty\left(\frac{1}{2}\left(\bar{I}_1 - 3\right) + \frac{1}{20\beta_\infty^2}\left(\bar{I}_1^2 - 9\right) + \frac{11}{1050\beta_\infty^4}\left(\bar{I}_1^3 - 27\right)\right) + \frac{K}{2}\left(J-1\right)^2 \\
U_T &= \mu_T\left(\frac{1}{2}\left(\bar{I}_1^e - 3\right) + \frac{1}{20\beta_T^2}\left(\bar{I}_1^{e2} - 9\right) + \frac{11}{1050\beta_T^4}\left(\bar{I}_1^{e3} - 27\right)\right) \\
\mathcal{F}\left(I_1, I_2, J, B, U\right) &= 2\left[\frac{1}{J^{2/3}}\frac{\partial U}{\partial I_1}B_{ij} - \frac{I_1}{3}\frac{\partial U}{\partial I_1}\delta_{ij}\right] \\
\tau_{ij} &= \mathcal{F}\left(\bar{I}_1, \bar{I}_2, J, B, U_\infty\right) + \mathcal{F}\left(\bar{I}_1^e, \bar{I}_2^e, J, B^e, U_T\right) + \frac{\partial U_\infty}{\partial J}\delta_{ij} \\
\tau_{ij}^D &= \mathcal{F}\left(\bar{I}_1^e, \bar{I}_2^e, J, B^e, U_T\right) \\
\tau_e &= \sqrt{\frac{3}{2}\tau_{ij}^D\tau_{ij}^D} \\
\dot{\varepsilon}_{ij}^p &= \dot{\varepsilon_e}\left(\tau_e, I_1^e, I_2^e, T\right)\frac{3\tau_{ij}^D}{2\tau_e} \\
\dot{\varepsilon}_e &= \dot{\varepsilon}_0\left(\sqrt{I_1^e} - \sqrt{3}\right)^n\left(\frac{\tau_e}{\tau_0}\right)^m
\end{aligned}
$$

2 Implementation

I implemented this material model as follows:

- Calculate $\Delta F_{ij} = \frac{\partial N^a}{\partial x_j} \Delta u_i^a$
- Calculate $F_{ij}^{n+1} = \delta_{ij} + \frac{\partial N^a}{\partial x_j} \left(u_i^a + \Delta u_i^a \right)$
- Calculate $\frac{\partial N^a}{\partial y_j} = \frac{\partial N^a}{\partial x_k} \frac{\partial x_k}{\partial y_j}$
- Recall F^p from state variable
- Calculate $F_{ij}^e = F_{ik}^{n+1} F_{kj}^{p^{-1}}$
- Calculate τ_{ij} using the formulæ above
- Calculate $\Delta F_{ij}^p = F_{ik}^{e^{-1}} \dot{\varepsilon}_{k\ell}^p \Delta t F_{\ell j}^{n+1}$
- Calculate $F^{p^{n+1}} = F^p + \Delta F^p$

3 Code

Listing 1: Relevant snippet of el_linelast_3D_basic subroutine

```
deltaF = matmul(reshape(dof_increment, (/3,n_nodes/)), dNdx(1:n_nodes,1:3))
F = identity + matmul(reshape(dof_total + dof_increment, (/3,n_nodes/)), dNdx(1:n_nodes,
    1:3))
call invert_small(F, Finv, J)
dNdy(1:n_nodes,1:3) = matmul(dNdx(1:n_nodes,1:3),Finv)
Fpvec = initial_state_variables(i:i + 8)
Fp = reshape(Fpvec, (/3,3/))
if (&
     Fp(1,1) == 0.d0 .AND. &
     Fp(1,2) == 0.d0 .AND. &
     Fp(1,3) == 0.d0 .AND. &
     Fp(2,1) == 0.d0 .AND. &
     Fp(2,2) == 0.d0 .AND. &
     Fp(2,3) == 0.d0 .AND. &
     Fp(3,1) == 0.d0 .AND. &
     Fp(3,2) == 0.d0 .AND. &
     Fp(3,3) == 0.d0) then
   Fp = identity
end if
call viscoelastic_stress(F,Fp,DTIME,element_properties,tauN,FpN)
updated_state_variables(i:i + 8) = reshape(FpN,(/9/))

call matToVec(tauN,tauNVec)

B = 0.d0

B(1,1:3*n_nodes-2:3) = dNdy(1:n_nodes,1)
B(2,2:3*n_nodes-1:3) = dNdy(1:n_nodes,2)
B(3,3:3*n_nodes:3)   = dNdy(1:n_nodes,3)
B(4,1:3*n_nodes-2:3) = dNdy(1:n_nodes,2)
B(4,2:3*n_nodes-1:3) = dNdy(1:n_nodes,1)
B(5,1:3*n_nodes-2:3) = dNdy(1:n_nodes,3)
B(5,3:3*n_nodes:3)   = dNdy(1:n_nodes,1)
B(6,2:3*n_nodes-1:3) = dNdy(1:n_nodes,3)
B(6,3:3*n_nodes:3)   = dNdy(1:n_nodes,2)

dNbarvec = reshape(dNbardy, (/3*n_nodes/))
dNvec = reshape(dNdy, (/3*n_nodes/))

Bbartmp = 0.d0
```

```
Bbartmp(1,1:3*n_nodes) = (1.d0/3.d0) * (dNbarvec - dNvec)
Bbartmp(2,1:3*n_nodes) = (1.d0/3.d0) * (dNbarvec - dNvec)
Bbartmp(3,1:3*n_nodes) = (1.d0/3.d0) * (dNbarvec - dNvec)

Bbar(1:6,1:3*n_nodes) = B + Bbartmp

element_residual(1:3*n_nodes) = element_residual(1:3*n_nodes) - matmul(transpose(B),tauNVec)
     * w(kint) * determinant
i = i + 9
```

Listing 2: Newly defined subroutine

```
subroutine viscoelastic_stress(F,Fp,DTIME,element_properties,tauN,FpN)
  use Types
  use Element_Utilities, only : invert_small
  implicit none
  real (prec), intent (in) :: F(3,3), Fp(3,3)
  real (prec), intent (in) :: element_properties(9)
  real (prec), intent (in) :: DTIME
  real (prec), intent (out) :: tauN(3,3), FpN(3,3)

  real (prec) :: muinf, betainf, muT, betaT, K, edot0, n, tau0, m
  real (prec) :: identity(3,3), Fe(3,3), BN(3,3), Be(3,3), tauND(3,3), edotp(3,3)
  real (prec) :: Feinv(3,3), Fpinv(3,3), deltaFp(3,3), throw(3,3), FpNinv(3,3)
  real (prec) :: I1, I1e, dUinfdI1, dUTdI1e, taue, edote, throwV, Bkk, Bekk, J

  muinf = element_properties(1)
  betainf = element_properties(2)
  muT = element_properties(3)
  betaT = element_properties(4)
  K = element_properties(5)
  edot0 = element_properties(6)
  tau0 = element_properties(7)
  n = element_properties(8)
  m = element_properties(9)

  identity = reshape((/ 1.d0, 0.d0, 0.d0, 0.d0, 1.d0, 0.d0, 0.d0, 0.d0, 1.d0 /), (/3,3/))

  call invert_small(Fp, Fpinv, throwV)

  Fe = matmul(F, Fpinv)

  call invert_small(F, throw, J)

  BN = matmul(F, transpose(F))

  Be = matmul(Fe, transpose(Fe))

  call trace(BN,Bkk)

  call trace(Be,Bekk)

  I1 = Bkk / (J ** (2.d0 / 3.d0))
  I1e = Bekk / (J ** (2.d0 / 3.d0))

  dUinfdI1 = muinf * &
       (1.d0 / 2.d0 + &
       (2.d0 / (20.d0 * betainf ** 2.d0)) * I1 + &
       (33.d0 / (1050.d0 * betainf ** 4.d0)) * I1 ** 2.d0)

  dUTdI1e = muT * &
       (1.d0 / 2.d0 + &
       (2.d0 / (20.d0 * betaT ** 2.d0)) * I1e + &
       (33.d0 / (1050.d0 * betaT ** 4.d0)) * I1e ** 2.d0)

  tauND = 2.d0 * dUTdI1e * ((1.d0 / (J ** (2.d0 / 3.d0))) * Be - I1e / 3.d0 * identity)
```

```
  tauN = &
       2.d0 * dUinfdI1 * ((1.d0 / (J ** (2.d0 / 3.d0))) * BN - I1 / 3.d0 * identity) + &
       tauND + &
       K * (J - 1.d0) * identity

  call trace(matmul(tauND, transpose(tauND)),taue)

  taue = sqrt(3.d0 / 2.d0) * sqrt(taue)

  edote = edot0 * (sqrt(I1e) - sqrt(3.d0)) ** n * (taue / tau0) ** m

  if (taue <= 1e-16 .OR. (sqrt(I1e) - sqrt(3.d0)) <= 1e-16) then
     edotp = 0.d0
  else
     edotp = edote * (3.d0 / (2.d0 * taue)) * tauND
  end if

  call invert_small(Fe,Feinv,throwV)

  deltaFp = matmul(matmul(Feinv,edotp * DTIME), F)

  FpN = Fp + deltaFp

end subroutine viscoelastic_stress
```

4 Results

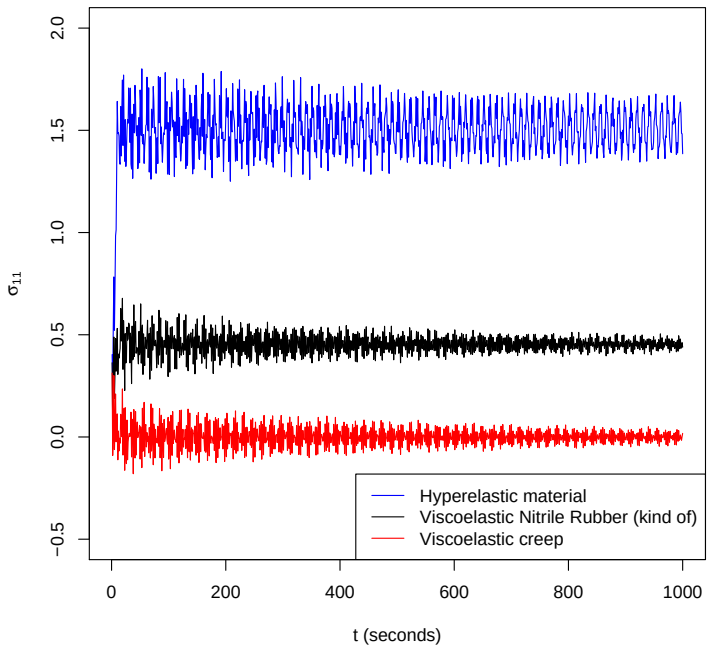

Figure 1: Comparison of different material parameters and expected behavior

As one can see in Figure 1, I ran three tests to determine if my implementation was working correctly. In all three tests, I obtained the expected behavior. In the hyperelastic case, as expected, the oscillations do not die down as the system is conservative and there is no loss of energy (there is some numerical damping though). By comparison, in the nitrile rubber case, we see the oscillations dying down. This makes sense

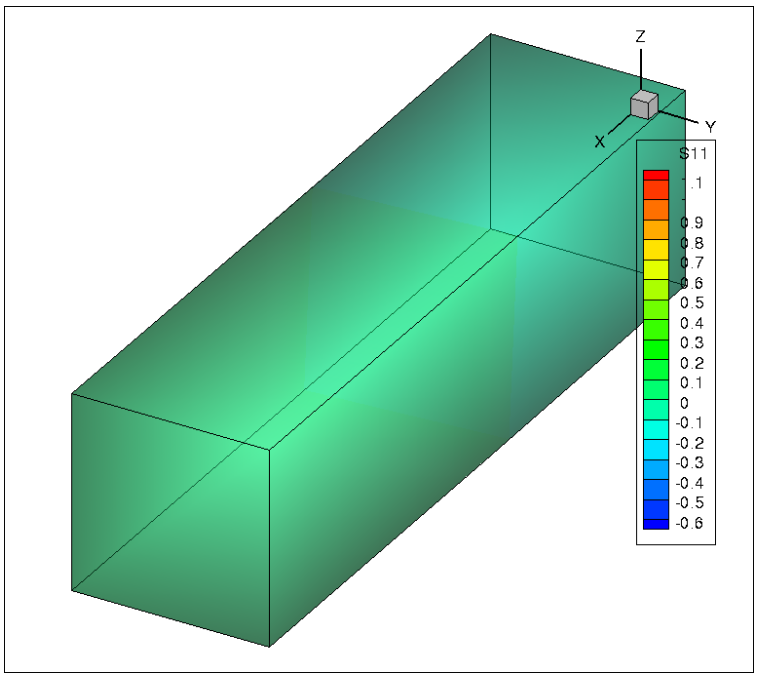

Figure 2: Full stress state of viscoelastic creep simulation

because the plasticity of the material leads to dissipation in energy. In the steady-state creep case, we see the oscillations quickly die down and a slight non-zero stress.

In the hyperelastic case, the general equation reduces to:

$$
\begin{aligned}
\dot{\varepsilon}^p &= 0 \\
F^e &= F \\
B &= B^e \\
\bar{I}_1 &= \bar{I}_1^e \\
\tau_{ij} &= \mathcal{F}\left(\bar{I}_1, J, B, U_\infty\right) + \mathcal{F}\left(\bar{I}_1, J, B, U_T\right) + \frac{\partial U_\infty}{\partial J}\delta_{ij} \\
\mathcal{F}\left(\bar{I}_1, J, B, U_\infty\right) &= 2\frac{\partial U_\infty}{\partial \bar{I}_1}\left[\frac{1}{J^{2/3}}B_{ij} - \frac{\bar{I}_1}{3}\delta_{ij}\right] \\
\mathcal{F}\left(\bar{I}_1, J, B, U_T\right) &= 2\frac{\partial U_T}{\partial \bar{I}_1}\left[\frac{1}{J^{2/3}}B_{ij} - \frac{\bar{I}_1}{3}\delta_{ij}\right] \\
\tau_{ij} &= 2\left[\frac{1}{J^{2/3}}B_{ij} - \frac{\bar{I}_1}{3}\delta_{ij}\right]\left(\frac{\partial U_\infty}{\partial \bar{I}_1} + \frac{\partial U_T}{\partial \bar{I}_1}\right) + K\left(J-1\right)\delta_{ij} \\
\tau_{11} &= 2\left[\frac{1}{J^{2/3}}B_{11} - \frac{\bar{I}_1}{3}\right]\left(\frac{\partial U_\infty}{\partial \bar{I}_1} + \frac{\partial U_T}{\partial \bar{I}_1}\right) + K\left(J-1\right) \\
\tau_{22} &= 2\left[\frac{1}{J^{2/3}}B_{22} - \frac{\bar{I}_1}{3}\right]\left(\frac{\partial U_\infty}{\partial \bar{I}_1} + \frac{\partial U_T}{\partial \bar{I}_1}\right) + K\left(J-1\right) \\
\tau_{33} &= 2\left[\frac{1}{J^{2/3}}B_{33} - \frac{\bar{I}_1}{3}\right]\left(\frac{\partial U_\infty}{\partial \bar{I}_1} + \frac{\partial U_T}{\partial \bar{I}_1}\right) + K\left(J-1\right) \\
\tau_{22} &= \tau_{33} = 0 \\
B_{22} &= B_{33}
\end{aligned}
$$

$$
\begin{aligned}
\bar{I}_1 &= \frac{B_{11}+B_{22}+B_{33}}{J^{2/3}} \\
F &= \begin{pmatrix} \lambda & 0 & 0 \\ 0 & \alpha & 0 \\ 0 & 0 & \alpha \end{pmatrix} \\
B &= \begin{pmatrix} \lambda^2 & 0 & 0 \\ 0 & \alpha^2 & 0 \\ 0 & 0 & \alpha^2 \end{pmatrix} \\
J &= \lambda\alpha^2 \\
\bar{I}_1 &= \frac{\lambda^2+2\alpha^2}{\left(\lambda\alpha^2\right)^{2/3}} \\
&= \lambda^{4/3}\alpha^{-4/3} + 2\alpha^{2/3}\lambda^{-2/3} \\
c_1 &= \frac{\mu_\infty + \mu_T}{2} = 0.51 \\
c_2 &= \frac{1}{10}\left(\frac{\mu_\infty}{\beta_\infty^2} + \frac{\mu_T}{\beta_T^2}\right) = 5.368 \times 10^{-3} \\
c_3 &= \frac{33}{1050}\left(\frac{\mu_\infty}{\beta_\infty^4} + \frac{\mu_T}{\beta_t^4}\right) = 9.665 \times 10^{-5} \\
\frac{\partial U_\infty}{\partial \bar{I}_1} + \frac{\partial U_T}{\partial \bar{I}_1} &= c_1 + c_2\bar{I}_1 + c_3\bar{I}_1^2 \\
\tau_{22} = \tau_{33} &= \frac{2\left(\alpha^2-\lambda^2\right)}{3\left(\lambda\alpha^2\right)^{2/3}}\left(c_1 + c_2\bar{I}_1 + c_3\bar{I}_1^2\right) + K\left(\lambda\alpha^2 - 1\right) = 0 \\
0 &= 2\left(\alpha^2-\lambda^2\right)\left(c_1 + c_2\bar{I}_1 + c_3\bar{I}_1^2\right) + 3K\left(\lambda\alpha^2\right)^{2/3}\left(\lambda\alpha^2 - 1\right) \\
\lambda &= \frac{3}{2} \\
K &= 5 \\
\alpha &= 0.85654598737271609801 \\
\tau_{11} &= 1.5238330922577697096
\end{aligned}
$$

This matches closely with the average stress in the computational results.

In the steady-state creep case, the general equation reduces to:

$$
\begin{aligned}
U_\infty &= 0 \\
\tau_{ij} &= \mathcal{F}\left(\bar{I}_1^e, J, B^e, U_T\right)
\end{aligned}
$$

From this equation, it is easy to see that the stress should tend to 0 as $F \to F^p$, which is what happens in the simulation as shown in Figure 1 (Figure 1 only shows σ_{11}, but the same is true for the other stress components as shown in Figure 2).

Qualitatively, the results make sense as well. The hyperelastic case should have the highest stress because $B^e = B$. In all other cases, $B^e < B$, which means $\tau_{ij}^{\text{viscoelastic}} < \tau_{ij}^{\text{hyperelastic}}$.

Given both the quantitative result (comparison of steady-state creep simulation and theory) and the qualitative results (as given above), it seems safe to say that the implementation works.