A nonlinear user element (UEL) implementation for hyperelastic materials in Abaqus/Standard

Bibekananda Datta

Mechanical Engineering, Johns Hopkins University, Baltimore, MD 21211, USA

Analyses of large deformation of hyperelastic materials require a different finite element formulation than the small deformation analyses. This document briefly presents one of the approaches to develop finite element formulations for large deformation and its implementation in Abaqus/Standard using its user element *i.e.*, `UEL(...)` subroutine interface. Here, a second Piola-Kirchhoff stress-based total Lagrangian formulation has been adopted to implement using UEL. Some excerpts have been directly adopted or reprinted from Datta, 2024.

Interested readers should consult textbooks on nonlinear solid mechanics and continuum mechanics (Bonet & Wood, 2008; Bower, 2009; Holzapfel, 2000), nonlinear finite element analyses (Belytschko et al., 2014; de Borst et al., 2012; Reddy, 2015; Wriggers, 2008; Zienkiewicz et al., 2014), textbooks and tutorials on Fortran programming, and Abaqus documentation[1] (*SIMULIA User Assistance 2024: Abaqus*, 2024).

1 Summary of the large deformation finite element formulation

In referential (or material) configuration, the governing partial differential equation for stress equilibrium as well as the boundary conditions are given by,

$$\begin{aligned} \operatorname{Div}\mathbf{P} + \rho_R\mathbf{B} &= 0 \qquad && \text{in } \Omega_0, \\ \mathbf{u} &= \mathbf{g} && \text{on } \Gamma_g, \\ \mathbf{P}\cdot\mathbf{N} &= \mathbf{T} && \text{on } \Gamma_T. \end{aligned} \tag{1.1}$$

where, $\partial\Omega_0 = \Gamma_g \cup \Gamma_T$ is the boundary of the referential domain Ω_0, and Γ_g and Γ_T are two complementary subsurfaces. Here, $\mathbf{P}$ is the first Piola-Kirchhoff (PK-I) stress, $\mathbf{B}$ is the body force per unit mass, ρ_{R} is the mass density in the reference state, and $\mathbf{T}$ is the Piola-Kichhoff traction.

The global weak form in the reference configuration is given by,

$$\mathcal{W}_{\mathbf{u}}(\mathbf{u}) = -\int_{\Omega_0} \mathbf{P} : \operatorname{Grad}(\mathbf{W})\, dV + \int_{\Omega_0} \rho_{\mathrm{R}}\mathbf{B}\cdot\mathbf{W}\, dV + \int_{\Gamma_T} \mathbf{T}\cdot\mathbf{W}\, dS = 0, \tag{1.2}$$

where, $\mathbf{W}$ is the weight function that vanishes on $\Gamma_g \subset \partial\Omega_0$. By substituting the relation

[1] Electronic version of the Abaqus user manual is available at https://help.3ds.com.

$\mathbf{P} = \mathbf{FS}$ into the above weak form for an arbitrary test function, we can have,

$$\mathcal{W}_{\mathbf{u}}(\mathbf{u}) = -\int_{\Omega_0} \mathbf{FS} : \mathrm{Grad}(\mathbf{W})\ dV + \int_{\Omega_0} \rho_{\mathrm{R}}\mathbf{B}\cdot\mathbf{W}\ dV + \int_{\Gamma_T} \mathbf{T}\cdot\mathbf{W}\ dS = 0, \tag{1.3}$$

where, $\mathbf{S} = \mathbf{F}^{-1}\mathbf{P}$ is the second Piola-Kirchhoff stress and $\mathbf{F} = \mathrm{Grad}(\mathbf{X})$ is the deformation gradient.

Using the standard Galerkin discretization, the discretized element residual vector, $\mathbf{R}^a_{\mathbf{u}}$, corresponds to any degree of freedom (DOFs) within the element can be written as,

$$\begin{aligned} \mathbf{R}^a_{\mathbf{u}}(\mathbf{u}_e) &= -\int_{\Omega^e_0} \mathbf{FS} : \mathrm{Grad}(\mathbf{N}^a_{\mathbf{u}})\ dV + \int_{\Omega^e_0} \rho_{\mathrm{R}}\mathbf{N}^{a\top}_{\mathbf{u}}\mathbf{B}\ dV + \int_{\Gamma^e_T} \mathbf{N}^{a\top}_{\mathbf{u}}\mathbf{T}_e\ dS = 0, \\ \Rightarrow \mathbf{R}^a_{\mathbf{u}}(u^a_i) &= -\int_{\Omega^e_0} \frac{\partial N^a}{\partial X_I} F_{iJ}S_{JI}\ dV + \int_{\Omega^e_0} \rho_{\mathrm{R}}N^a B_I\ dV + \int_{\Gamma^e_T} N^a T^e_i\ dS = 0, \end{aligned} \tag{1.4}$$

where, $[\mathbf{N}^a_{\mathbf{u}}]$ is the nodal matrix and $[\mathbf{N}_{\mathbf{u}}]$ is the element matrix of the interpolation functions, and for three-dimensional cases, they are given by,

$$\mathbf{N}^a_{\mathbf{u}} = \begin{bmatrix} N_i & 0 & 0 \\ 0 & N_i & 0 \\ 0 & 0 & N_i \end{bmatrix}, \quad \mathbf{N}_{\mathbf{u}} = \begin{bmatrix} N_1 & 0 & 0 & N_2 & 0 & 0 & \cdots & N_{\mathrm{nen}} & 0 & 0 \\ 0 & N_1 & 0 & 0 & N_2 & 0 & \cdots & 0 & N_{\mathrm{nen}} & 0 \\ 0 & 0 & N_1 & 0 & 0 & N_2 & \cdots & 0 & 0 & N_{\mathrm{nen}} \end{bmatrix}. \tag{1.5}$$

The vector representation of nodal degrees of freedom of an element, $\mathbf{u}_e$, and the Voigt vector of the second Piola-Kirchhoff stress, $\mathbf{S}$, are given by,

$$\begin{aligned} \mathbf{u}_e &= \begin{bmatrix} u^1_1 & u^1_2 & u^1_3 & u^2_1 & u^2_2 & u^2_3 & \cdots & \cdots & \cdots & u^{\mathrm{nen}}_1 & u^{\mathrm{nen}}_2 & u^{\mathrm{nen}}_3 \end{bmatrix}^{\top}_{\mathrm{nsd}*\mathrm{nen}\times 1}, \\ &\text{and } \boldsymbol{\sigma} = \begin{bmatrix} S_{11} & S_{22} & S_{33} & S_{23} & S_{13} & S_{12} \end{bmatrix}^{\top} \end{aligned} \tag{1.6}$$

Following the standard procedure of the Newton-Raphson method, the component of the element tangent matrix is given by,

$$\begin{aligned} \mathbf{k}^{ab}_{\mathbf{uu}} &= -\frac{\mathbf{R}^a_{\mathbf{u}}}{\partial \mathbf{u}^b}, \\ \Rightarrow k^{ab}_{u_i u_k} &= \int_{\Omega^e_0} \frac{\partial N^a}{\partial X_J} S_{JL}\delta_{ik} \frac{\partial N^b}{\partial X_L}\ dV + \int_{\Omega^e_0} \frac{\partial N^a}{\partial X_J}\left(F_{iI}\mathbb{C}_{IJKL}F_{kK}\right)\frac{\partial N^b}{\partial X_L}\ dV. \end{aligned} \tag{1.7}$$

where,

$$\mathbb{C}_{IJKL} = 2\frac{\partial S_{JI}}{\partial C_{KL}} = \frac{\partial S_{JI}}{\partial E_{KL}} \tag{1.8}$$

is defined as the material tangent tensor (or second elasticity tensor).

The matrix form of the element tangent matrix $[\mathbf{k}^e_{\mathbf{uu}}]$ and the element residual vector, $\mathbf{R}^e_{\mathbf{u}}$, are given by,

$$\begin{aligned}\mathbf{k}^e_{\mathbf{uu}} &= \int_{\Omega^e_0} \left(\mathbf{G}^\top_{\mathbf{u}} \Sigma_{\mathbf{S}} \mathbf{G}_{\mathbf{u}} + (\mathbf{B}_{\mathbf{u}} \boldsymbol{\Sigma}_{\mathbf{F}})^\top \mathbf{D}_{\mathbb{C}} (\mathbf{B}_{\mathbf{u}} \Sigma_{\mathbf{F}})\right)\, dV, \\ \mathbf{R}^e_{\mathbf{u}} &= -\int_{\Omega^e_0} (\mathbf{B}_{\mathbf{u}} \Sigma_{\mathbf{F}})^\top \mathbf{S}\, dV + \int_{\Omega^e_0} \rho_{\mathrm{R}} \mathbf{N}^\top_{\mathbf{u}} \mathbf{B}\, dV + \int_{\Gamma^e_T} \mathbf{N}^\top_{\mathbf{u}} \mathbf{T}_e\, dS.\end{aligned} \tag{1.9}$$

Here, $\mathbf{S}$ is the second Piola-Kirchhoff stress in Voigt vector form, $\Sigma_{\mathbf{F}}$ and $\Sigma_{\mathbf{S}}$ are expanded matrix forms of the deformation gradient, and PK-II stress tensor, respectively, $\mathbf{B}_{\mathbf{u}}$ is the symmetric gradient matrix (in the context of small deformation it is also known as strain-displacement matrix), $[\mathbf{G}_{\mathbf{u}}]$ is the non-symmetric gradient matrix, and $\mathbf{D}_{\mathbb{C}}$ is the Voigt matrix form of the material tangent.

$[\Sigma_{\mathbf{F}}]_{\mathrm{nsd}*\mathrm{nen}\times\mathrm{nsd}*\mathrm{nen}}$ is a square banded diagonal matrix of dimension $\mathrm{nsd}*\mathrm{nen} \times \mathrm{nsd}*\mathrm{nen}$, and for a three-dimensional case, it appears as,

$$\Sigma_{\mathbf{F}} = \begin{bmatrix} F_{11} & F_{12} & F_{13} & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ F_{21} & F_{22} & F_{23} & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ F_{31} & F_{32} & F_{33} & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ & & & & \cdots & & & & & \\ & & & & \cdots & & & & & \\ & & & & \cdots & & & & & \\ & & & & \cdots & & & & & \\ & & & & \cdots & & & & & \\ & & & & \cdots & & & & & \\ 0 & 0 & 0 & 0 & 0 & 0 & \cdots & F_{11} & F_{12} & F_{13} \\ 0 & 0 & 0 & 0 & 0 & 0 & \cdots & F_{21} & F_{22} & F_{23} \\ 0 & 0 & 0 & 0 & 0 & 0 & \cdots & F_{31} & F_{32} & F_{33} \end{bmatrix}_{\mathrm{nsd}*\mathrm{nen}\times\mathrm{nsd}*\mathrm{nen}} . \tag{1.10}$$

For a two-dimensional case, we need to reduce the size of $\Sigma_{\mathbf{F}}$.

$[\Sigma_{\mathbf{S}}]$ is the expanded stress matrix. For a two-dimensional and a three-dimensional case, it is given by,

$$\Sigma_{\mathbf{S}} = \begin{bmatrix} \mathbf{S}_{11} & 0 & \mathbf{S}_{12} & 0 \\ 0 & \mathbf{S}_{11} & 0 & \mathbf{S}_{12} \\ \mathbf{S}_{12} & 0 & \mathbf{S}_{22} & 0 \\ 0 & \mathbf{S}_{12} & 0 & \mathbf{S}_{22} \end{bmatrix}, \quad \Sigma_{\mathbf{S}} = \begin{bmatrix} \mathbf{S}_{11} & 0 & 0 & \mathbf{S}_{12} & 0 & 0 & \mathbf{S}_{13} & 0 & 0 \\ 0 & \mathbf{S}_{11} & 0 & 0 & \mathbf{S}_{12} & 0 & 0 & \mathbf{S}_{13} & 0 \\ 0 & 0 & \mathbf{S}_{11} & 0 & 0 & \mathbf{S}_{12} & 0 & 0 & \mathbf{S}_{13} \\ \mathbf{S}_{12} & 0 & 0 & \mathbf{S}_{22} & 0 & 0 & \mathbf{S}_{23} & 0 & 0 \\ 0 & \mathbf{S}_{12} & 0 & 0 & \mathbf{S}_{22} & 0 & 0 & \mathbf{S}_{23} & 0 \\ 0 & 0 & \mathbf{S}_{12} & 0 & 0 & \mathbf{S}_{22} & 0 & 0 & \mathbf{S}_{23} \\ \mathbf{S}_{13} & 0 & 0 & \mathbf{S}_{23} & 0 & 0 & \mathbf{S}_{33} & 0 & 0 \\ 0 & \mathbf{S}_{13} & 0 & 0 & \mathbf{S}_{23} & 0 & 0 & \mathbf{S}_{33} & 0 \\ 0 & 0 & \mathbf{S}_{13} & 0 & 0 & \mathbf{S}_{23} & 0 & 0 & \mathbf{S}_{33} \end{bmatrix}. \tag{1.11}$$

$[\Sigma_{\mathbf{S}}]$ has a dimension of $[\Sigma_{\mathbf{S}}]_{\text{nsd}^2\times\text{nsd}^2}$. It is also possible to represent $\mathbf{G}^a_{\mathbf{u}}$ and consequently $\mathbf{G}_{\mathbf{u}}$ and $\Sigma_{\mathbf{S}}$ matrices in alternative matrix forms which will essentially give the same result (de Borst et al., 2012; Reddy, 2015).

Similar to the interpolation function matrix, $[\mathbf{N}_{\mathbf{u}}]$, the symmetric gradient matrix, $[\mathbf{B}_{\mathbf{u}}]$, is also composed of repetitive sub-matrices, $[\mathbf{B}^a_{\mathbf{u}}]$. The matrix form of $[\mathbf{B}_{\mathbf{u}}]$ and $[\mathbf{B}^a_{\mathbf{u}}]$ are given by,

$$\mathbf{B}_{\mathbf{u}} = \begin{bmatrix}\mathbf{B}^1_{\mathbf{u}} & \mathbf{B}^2_{\mathbf{u}} & \mathbf{B}^3_{\mathbf{u}} & \cdots & \cdots & \mathbf{B}^{\text{nen}}_{\mathbf{u}}\end{bmatrix}_{\text{nstress}\times\text{nsd}*\text{nen}}, \quad \text{where,} \quad \mathbf{B}^a_{\mathbf{u}} = \begin{bmatrix} N^a_{,1} & 0 & 0 \\ 0 & N^a_{,2} & 0 \\ 0 & 0 & N^a_{,3} \\ 0 & N^a_{,3} & N^a_{,2} \\ N^a_{,3} & 0 & N^a_{,1} \\ N^a_{,2} & N^a_{,1} & 0 \end{bmatrix}_{\text{nstress}\times\text{nsd}} . \tag{1.12}$$

The non-symmetric gradient matrix, $[\mathbf{G}_{\mathbf{u}}]$, is also contains repetitive sub-matrices in the following form,

$$\mathbf{G}_{\mathbf{u}} = \begin{bmatrix}\mathbf{G}^1_{\mathbf{u}} & \mathbf{G}^2_{\mathbf{u}} & \mathbf{G}^3_{\mathbf{u}} & \cdots & \cdots & \mathbf{G}^{\text{nen}}_{\mathbf{u}}\end{bmatrix}_{\text{nsd}^2\times\text{nsd}*\text{nen}}, \tag{1.13}$$

where the sub-matrix, $[\mathbf{G}^a_{\mathbf{u}}]$, for two-dimensional and three-dimensional cases are given by,

$$\mathbf{G}^a_{\mathbf{u}} = \begin{bmatrix} N^a_{,1} & 0 \\ 0 & N^a_{,1} \\ N^a_{,2} & 0 \\ 0 & N^a_{,2} \end{bmatrix}_{\text{nsd}^2\times\text{nsd}}, \qquad \mathbf{G}^a_{\mathbf{u}} = \begin{bmatrix} N^a_{,1} & 0 & 0 \\ 0 & N^a_{,1} & 0 \\ 0 & 0 & N^a_{,1} \\ N^a_{,2} & 0 & 0 \\ 0 & N^a_{,2} & 0 \\ 0 & 0 & N^a_{,2} \\ N^a_{,3} & 0 & 0 \\ 0 & N^a_{,3} & 0 \\ 0 & 0 & N^a_{,3} \end{bmatrix}_{\text{nsd}^2\times\text{nsd}} . \tag{1.14}$$

Based on the finite element formulation presented in this document, a simplified pseudo-code or procedure for developing the Abaqus/Standard UEL subroutine is in **Procedure** 1.

Procedure 1: A large deformation total Lagrangian UEL subroutine implementation in Abaqus/Standard

Input : `PROPS, COORDS, JELEM, JTYPE, NNODE, NDOFEL, TIME, DTIME, U, DU, V, A, PREDEF, JDLTYP, NDLOAD, MDLOAD, DDLMAG, ALMAG`
Output : `AMATRX, RHS, PNEWDT, ENERGY, SVARS`

Get `nInt` $\leftarrow$ `PROPS` and `nDim`, `nTens` $\leftarrow$ `JTYPE`
Initialize: $\{\mathbf{N}^a_{\mathbf{u}}, \mathbf{B}^a_{\mathbf{u}}, \mathbf{G}^a_{\mathbf{u}}, \mathbf{N}_{\mathbf{u}}, \mathbf{B}_{\mathbf{u}}, \mathbf{G}_{\mathbf{u}}, \Sigma_{\mathbf{F}}, \Sigma_{\mathbf{S}}, \mathbf{k}^e_{\mathbf{uu}}, \mathbf{R}^e_{\mathbf{u}}\} = 0$
Get nodal displacement vectors of the element: $\mathbf{u}_e, \Delta\mathbf{u}_e$
Reshape nodal coordinate and displacement vector in matrix form: $\overline{\mathbf{x}_e}, [\overline{\mathbf{u}}_e], \overline{\Delta\mathbf{u}_e}$
Get $w_{\text{int}}, \boldsymbol{\xi}_{\text{int}} \leftarrow$ SUBROUTINE `gaussQuadrtr`(`nDim`, `nNode`)
for $k = 1$ **to** `nInt` **do**
    Get $\mathbf{N}$, $\frac{\partial \mathbf{N_u}}{\partial \boldsymbol{\xi}} \leftarrow$ SUBROUTINE `interpFunc`(`nDim`, `nNode`)
    Calculate: $\frac{\partial \mathbf{X}}{\partial \boldsymbol{\xi}} = \overline{\mathbf{x}_e}\frac{\partial \mathbf{N_u}}{\partial \boldsymbol{\xi}}$ // map to the reference configuration to calculate **F**
    Calculate: $\frac{\partial \mathbf{N_u}}{\partial \mathbf{X}} = \frac{\partial \mathbf{N_u}}{\partial \boldsymbol{\xi}}\left(\frac{\partial \mathbf{X}}{\partial \boldsymbol{\xi}}\right)^{-1}$ and J_ξ
    Calculate deformation gradient: $\mathbf{F} = \mathbb{1} + [\overline{\mathbf{u}}_e]\frac{\partial \mathbf{N_u}}{\partial \mathbf{X}}$
    for $i = 1$ **to** `nNode` **do**
        Form $\mathbf{N}^a_{\mathbf{u}}(\mathtt{nDim}, \mathtt{NDOFEL})$, $\mathbf{B}^a_{\mathbf{u}}(\mathtt{nTens}, \mathtt{nDim})$, and $\mathbf{G}^a_{\mathbf{u}}(\mathtt{nDim}^2, \mathtt{nDim})$ matrices
        $\mathbf{N}_{\mathbf{u}}(1:\mathtt{nDim}, \mathtt{nDim}*(\mathtt{i}-1)+1:\mathtt{nDim}*\mathtt{i}) = \mathbf{N}^a_{\mathbf{u}}(1:\mathtt{nDim}, 1:\mathtt{nDim})$
        $\mathbf{B}_{\mathbf{u}}(1:\mathtt{nTens}, \mathtt{nDim}*(\mathtt{i}-1)+1:\mathtt{nDim}*\mathtt{i}) = \mathbf{B}^a_{\mathbf{u}}(1:\mathtt{nTens}, 1:\mathtt{nDim})$
        $\mathbf{G}_{\mathbf{u}}(1:\mathtt{nDim}^2, \mathtt{nDim}*(\mathtt{i}-1)+1:\mathtt{nDim}*\mathtt{i}) = \mathbf{G}^a_{\mathbf{u}}(1:\mathtt{nDim}^2, 1:\mathtt{nDim})$
    end
                                                  // end of nodal point loop
    Get $\mathbf{S}, \mathbf{e}, \boldsymbol{\sigma}, \mathbf{D}_{\mathbb{C}} \leftarrow$ SUBROUTINE `umatHyperelastic`(`PROPS`, $\mathbf{F}$) // UMAT returns the constitutive response
    Form $\Sigma_{\mathbf{F}}(\mathtt{nDim}*(\mathtt{i}-1)+1:\mathtt{nDim}*\mathtt{i}, \mathtt{nDim}*(\mathtt{j}-1)+1:\mathtt{nDim}*\mathtt{j}) = \mathbf{F}$
     // double nested for loop over nNode with i=j
    Form $\Sigma_{\mathbf{S}}(\mathtt{nDim}*(\mathtt{i}-1)+1:\mathtt{nDim}*\mathtt{i}, \mathtt{nDim}*(\mathtt{j}-1)+1:\mathtt{nDim}*\mathtt{j}) = S_{ij}\mathbb{1}$
     // double nested for loop over nDim
    Calculate: $\mathbf{k}^e_{\mathbf{uu}} = \mathbf{k}^e_{\mathbf{uu}} + w_{\text{int}}(k)\det(J_\xi)\left(\mathbf{G}^\top_{\mathbf{u}}\Sigma_{\mathbf{S}}\mathbf{G}_{\mathbf{u}} + (\mathbf{B}_{\mathbf{u}}\Sigma_{\mathbf{F}})^\top\mathbf{D}_{\mathbb{C}}(\mathbf{B}_{\mathbf{u}}\Sigma_{\mathbf{F}})\right)$
    Calculate: $\mathbf{R}^e_{\mathbf{u}} = \mathbf{R}^e_{\mathbf{u}} - w_{\text{int}}(k)\det(J_\xi)\left((\mathbf{B}_{\mathbf{u}}\Sigma_{\mathbf{F}})^\top\mathbf{S} - \rho_{\text{R}}\mathbf{N}^\top_{\mathbf{u}}\mathbf{b}\right)$
end
                                                  // end of integration point loop
Assign `AMATRX` $= \mathbf{k}^e_{\mathbf{uu}}$ and `RHS` $= \mathbf{R}^e_{\mathbf{u}}$

The current implementation includes 3D continuum solid elements and plane-strain elements, but body force and traction loading were not included. Since Abaqus does not natively support viewing element output from the user element, a layer of additional dummy elements with `UVARM(...)` subroutine from Abaqus was used to view the results.

2 Constitutive models for hyperelastic materials

In this implementation, two different types of constitutive models have been included to represent the mechanical behavior of hyperelastic materials: (a) Neo-Hookean model and (b) Arruda-Boyce model. For both models, the coupled strain energy density representation has been adopted to model quasi-incompressible behaviors. Users can specify the bulk modulus of the materials to be a few orders of magnitude larger than the shear modulus to enforce quasi-incompressibility. However, it should be noted that these elements will behave poorly under certain deformation states because of *volumetric locking*.

2.1 Neo-Hookean material

The strain energy density, Ψ, for a quasi-incompressible Neo-Hookean type material is given by,

$$\Psi = \frac{\mu}{2}(I_1 - 3 - 2\ln J) + \frac{\kappa}{2}(\ln J)^2. \tag{2.1}$$

where, $I_1 = \mathrm{tr}(\mathbf{C})$ and $J = \det(\mathbf{F})$, and μ and κ are the material parameters representing the shear modulus and bulk modulus.

The second Piola-Kirchhoff stress, $\mathbf{S}$, and the material tangent, $\mathbb{C}$, are given by,

$$\begin{aligned} \mathbf{S} &= \mu(\mathbb{1} - \mathbf{C}^{-1}) + \kappa(\ln J)\mathbf{C}^{-1}, \\ \mathbb{C}_{IJKL} = 2\frac{\partial S_{IJ}}{C_{KL}} &= \kappa C^{-1}_{IJ}C^{-1}_{KL} + (\mu - \kappa\ln J)\left(C^{-1}_{IK}C^{-1}_{JL} + C^{-1}_{JK}C^{-1}_{IL}\right). \end{aligned} \tag{2.2}$$

2.2 Arruda-Boyce material

The strain energy density, Ψ, for a quasi-incompressible Arruda-Boyce type model is given by,

$$\begin{aligned} \Psi = \mu\left[\lambda_{\mathrm{L}}^2\left(\frac{\lambda_c\beta_c}{\lambda_{\mathrm{L}}} + \ln\frac{\beta_c}{\sinh\beta_c}\right) - \left(\frac{\lambda_{\mathrm{L}}}{3}\right)\ln J\right] + \frac{\kappa}{2}(\ln J)^2, \\ \text{where, } \beta_c = \mathcal{L}^{-1}\left(\frac{\lambda_c}{\lambda_{\mathrm{L}}}\right), \quad \text{and} \quad \lambda_c = \sqrt{\frac{I_1}{3}}, \end{aligned} \tag{2.3}$$

The second Piola-Kirchhoff stress, $\mathbf{S}$, is given by,

$$\mathbf{S} = \mu\left(\frac{\lambda_{\mathrm{L}}}{3\lambda_c}\beta_c\right)\mathbb{1} - \left[\frac{\mu\lambda_{\mathrm{L}}}{3} - \kappa(\ln J)\right]\mathbf{C}^{-1}. \tag{2.4}$$

Using index notation, the material tangent, $\mathbb{C}$, can be written as,

$$\begin{aligned}\mathbb{C}_{IJKL} &= 2\frac{\partial S_{IJ}}{C_{KL}}, \\ &= \frac{\mu}{9\lambda_c^2}\left(\frac{\partial \beta_c}{\partial\left(\frac{\lambda_c}{\lambda_{\mathrm{L}}}\right)} - \frac{\lambda_{\mathrm{L}}}{\lambda_c}\beta_c\right)\delta_{IJ}\delta_{KL} + \kappa C_{IJ}^{-1}C_{KL}^{-1} \\ &\quad + \left[\mu\left(\frac{\lambda_{\mathrm{L}}}{3\lambda_c}\right) - \kappa(\ln J)\right]\left(C_{IK}^{-1}C_{JL}^{-1} + C_{JK}^{-1}C_{IL}^{-1}\right)\end{aligned} \tag{2.5}$$

Remark 1. The built-in hyperelastic material models available in Abaqus/Standard are represented by an uncoupled strain energy density formulation in which the deformation gradient is also split into deviatoric and volumetric parts.

3 Modeling in Abaqus

If and only if the user element has the same topology as any standard built-in element available in Abaqus, we can build the primary model in Abaqus/CAE, export the input file (`.inp`), and modify it to execute with Abaqus/Standard solver. Building the primary model is a standard exercise, and it is strongly recommended to check the option **Do not use parts and assemblies in input files** from the **model attributes** drop-down menu before exporting the input file of a model. Once the input is available, it must be modified following the instructions in the Abaqus user manual.

The first step is to include the element definition for the user elements in the input file by removing the standard element defined with `*Element` keyword in generating the model.

```
*User Element, Type=< >, Nodes=< >, Coordinates=< >, Properties=<
   >, Iproperties=< >
<list of degrees of freedom>
*Element, type=< >
<list of element nodal connectivity>
```

The second step is to define the properties of the user elements by element sets by removing standard keywords `*Section` and `*Material`.

```
*Uel property, Elset= < >
<list of properties>
```

The third step is optional, but if the user wants to visualize the results in Abaqus/Viewer, a set of overlaying dummy elements needs to be added to the input file.

```
*Element, Type=< >
<list of element nodal connectivity>
*Elset, Elset=elDummy
```

```
< >
*Solid section, Elset=elDummy, Material=Dummy
*Material, Name=Dummy
*Elastic
1.e-20
*User output variables
< >
```

The fourth step is to request element-level output in load steps.

```
*Element output, Elset=< >
uvarm
```

Bibliographic references

Belytschko, T., Liu, W. K., Moran, B., & Elkhodary, K. I. (2014). *Nonlinear Finite Elements for Continua and Structures* (2nd ed.). John Wiley & Sons, Inc.

Bonet, J., & Wood, R. D. (2008). *Nonlinear Continuum Mechanics for Finite Element Analysis* (2nd ed.). Cambridge University Press.

Bower, A. F. (2009). *Applied Mechanics of Solids* (1st ed.). CRC Press.

Datta, B. (2024). *An Abaqus user element (UEL) implementation of linear elastostatics*. Zenodo. https://doi.org/10.5281/zenodo.11075088

de Borst, R., Crisfield, M. A., Remmers, J. J. C., & Verhoosel, C. V. (2012). *Non-linear Finite Element Analysis of Solids and Structures* (2nd ed.). John Wiley & Sons, Inc.

Holzapfel, G. (2000). *Nonlinear Solid Mechanics: A Continuum Approach for Engineering* (1st ed.). John Wiley & Sons, Inc.

Reddy, J. N. (2015). *An Introduction to Nonlinear Finite Element Analysis* (1st ed.). Oxford University Press.

SIMULIA User Assistance 2024: Abaqus. (2024). Dassault Systèmes. Providence, RI.

Wriggers, P. (2008). *Nonlinear Finite Element Methods* (1st ed.). Springer.

Zienkiewicz, O. C., Taylor, R. L., & Fox, D. (2014). *The Finite Element Method for Solid and Structural Mechanics* (7th ed.). Butterworth-Heinemann.