A CONSTITUTIVE MODEL FOR BOTH LOW AND HIGH STRAIN NONLINEARITIES IN HIGHLY FILLED ELASTOMERS AND IMPLEMENTATION WITH USER-DEFINED MATERIAL SUBROUTINES IN ABAQUS

TRAVIS W. HOHENBERGER,[1] RICHARD J. WINDSLOW,[2] NICOLA M. PUGNO,[1,3,4] JAMES J. C. BUSFIELD[1,*]

[1]SOFT MATTER GROUP, SCHOOL OF ENGINEERING & MATERIALS SCIENCE, QUEEN MARY UNIVERSITY OF LONDON, LONDON, UNITED KINGDOM
[2]SCHLUMBERGER LIMITED, HOUSTON, TX 77056
[3]LABORATORY OF BIO-INSPIRED & GRAPHENE NANO-MECHANICS, DEPARTMENT OF CIVIL, ENVIRONMENTAL, AND MECHANICAL ENGINEERING, UNIVERSITY OF TRENTO, TRENTO, ITALY
[4]KET-LAB, EDOARDO AMALDI FOUNDATION, VIA DEL POLITECNICO SNC, I-00133, ROME, ITALY

RUBBER CHEMISTRY AND TECHNOLOGY, Vol. 92, No. 4, pp. 653–686 (2019)

ABSTRACT

Strain energy functions (SEFs) are used to model the hyperelastic behavior of rubberlike materials. In tension, the stress–strain response of these materials often exhibits three characteristics: (i) a decreasing modulus at low strains (<20%), (ii) a constant modulus at intermediate strains, and (iii) an increasing modulus at high strains (>200%). Fitting an SEF that works in each regime is challenging when multiple or nonhomogeneous deformation modes are considered. The difficulty increases with highly filled elastomers because the small strain nonlinearity increases and finite-extensibility occurs at lower strains. One can compromise by fitting an SEF to a limited range of strain, but this is not always appropriate. For example, rubber seals in oilfield packers can exhibit low global strains but high localized strains. The Davies–De–Thomas (DDT) SEF is a good candidate for modeling such materials. Additional improvements will be shown by combining concepts from the DDT and Yeoh SEFs to construct a more versatile SEF. The SEF is implemented with user-defined material subroutines in Abaqus/Standard (UHYPER) and Abaqus/Explicit (VUMAT) for a three-dimensional general strain problem, and an approach to overcome a mathematically indeterminate stress condition in the unstrained state is derived. The complete UHYPER and VUMAT subroutines are also presented. [doi:10.5254/rct.19.80387]

INTRODUCTION

Rubbery materials undergoing large deformations exhibit geometric and material nonlinearities. Geometric nonlinearities arise from differences between the initial and deformed material configurations and can be addressed with nonlinear solid mechanics.[1] Material nonlinearities are evident in rubber's purely elastic (time-independent) stress–strain response, which is poorly modeled by Hooke's law. Nonlinearities also arise in rubber's viscoelastic responses. In this work, only an ideal, fully reversible hyperelastic response is considered because it is important in many applications such as setting of oilfield packer seals in wellbores.

Even with ideal hyperelasticity, the stress–strain responses of rubbery materials are loading mode dependent. For example, Figure 1a shows uniaxial tension (UT) and equibiaxial tension (ET) data for unfilled NR as tabulated by Simulia Corp.[2] using Treloar's data.[3] The stress–stretch response in ET is stiffer than that in UT, but the qualitative trends are similar. There is an initial reduction in stiffness followed by a linear range and finally an upward inflection that can be attributed to some combination of finite-extensibility of the polymer chains and strain-induced crystallization.[4] Figure 1a also shows equivalent uniaxial compression (UC) that, assuming incompressibility, can be computed from ET data as follows:

*Corresponding author. Ph: +44-(0)20-7882-8866; email: j.busfield@qmul.ac.uk

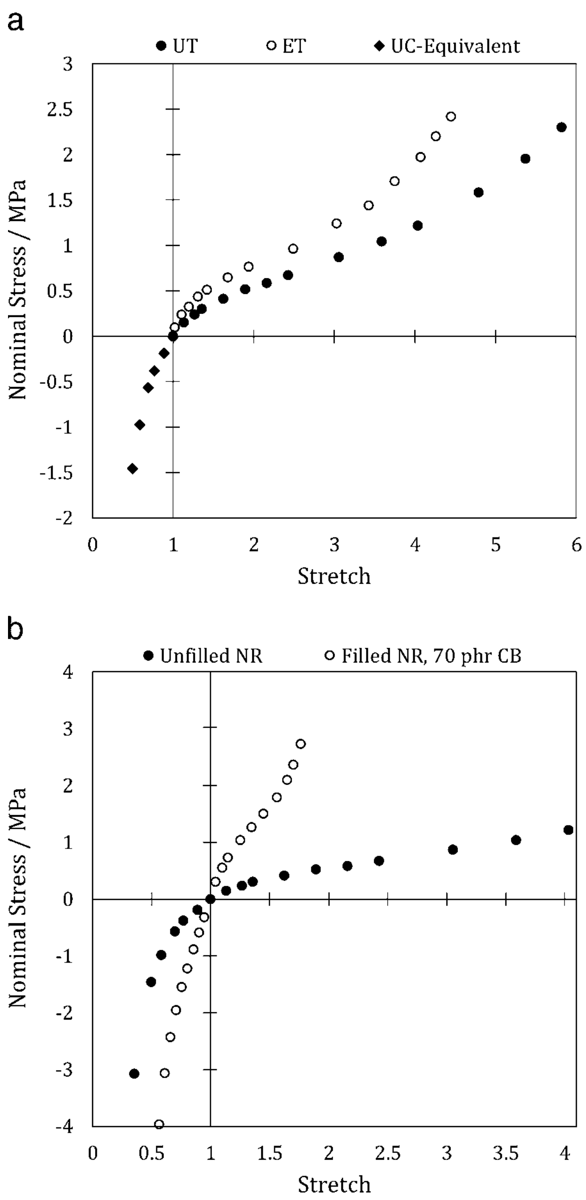


FIG. 1. — Stress–stretch responses with NR: (a) Treloar's unfilled rubber;[3] (b) Yeoh's filled rubber[7] compared with Treloar's unfilled rubber.

$$\lambda_{uc} = \lambda_{et}^{-2}$$

$$\sigma_{uc} = -\lambda_{et}^{3}\sigma_{et} \tag{1}$$

where λ's are stretches, σ's are nominal stresses, and subscripts indicate loading modes. Whereas the ET response shows an initial reduction in stiffness, the UC response shows a monotonic increase.This behavior is not an artefact of theoretical inaccuracies in Eq. 1 and has been reported by researchers who directly measured UC response.[5,6]

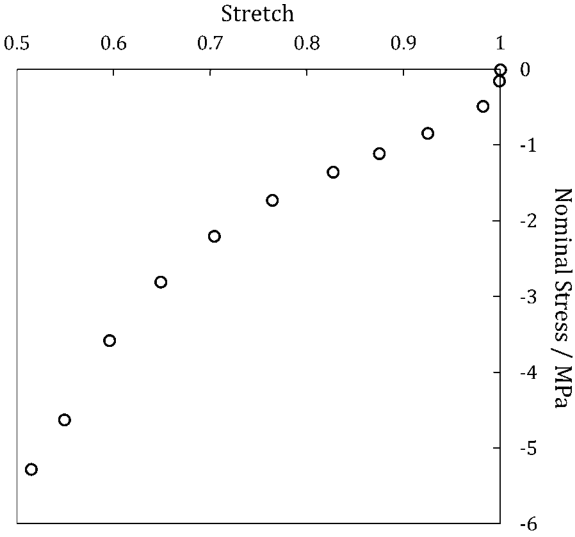


FIG. 2. — UC stress–stretch data[9] showing a low strain inflection for a filled HDR.

The importance of considering high strain nonlinearities usually increases as filler content increases. For example, Figure 1b compares Yeoh's data for NR filled with 70 parts per hundred of rubber (phr) of carbon black (CB)[7] to Treloar's data. The filled rubber is stiffer, and its upward inflection occurs at lower stretch. Yeoh's data also show a monotonic increase in stiffness during compression.

Low strain nonlinearities are more prominent in filled elastomers due to the Payne effect.[8] It causes a rapid reduction in stiffness at low strains due to breakdown of the filler network, and it results in more pronounced curvatures in UT stress–strain plots. In some cases, the effect is sufficient to cause an inflection in the UC stress–strain response and can be seen, for example, in the data of Amin et al. for high damping rubber (HDR) in Figure 2.[9]

In some applications, it is important to accurately model both low and high strain nonlinearities with inhomogeneous deformations. To illustrate, consider the oilfield packer seal in Figure 3a. It consists of three axisymmetric rubber sealing components (dark gray) and two metal anti-extrusion rings (light gray) installed around a tubular component. During setting, the seals and anti-extrusion devices are compressed between two rings. This causes the outer diameter of the seals to contact a sealing surface (Figure 3b) until the seals are fully packed off (Figure 3c). In this example, global nominal strains in the rubber are limited to ~30%, but local strains exceed 100% near the ends of the anti-extrusion rings. When simulating the setting process, it is critical to accurately model both global and local strains. Global strains ensure components move as desired, and local strains determine whether rubber will fracture during the setting process.

Figure 4 shows UT and UC stress–stretch data for a typical packer seal rubber made from a filled HNBR compound. The Payne effect is apparent in the low strain inflection of the UC data, and the UT data exhibits finite-extensibility beginning at a stretch of 1.4. The shear modulus, approximated as $G = E/3$, where E is Young's modulus, is 13 MPa. To emphasize this magnitude, Yeoh's filled NR data[7] and Fujikawa et al.'s SBR data with 20% volume CB[10] are also shown. Packer seal materials have a high modulus to facilitate deployment of anti-extrusion devices, resist extrusion during pressure application,[11] and mitigate elastic instabilities related to rapid gas decompression.[12]

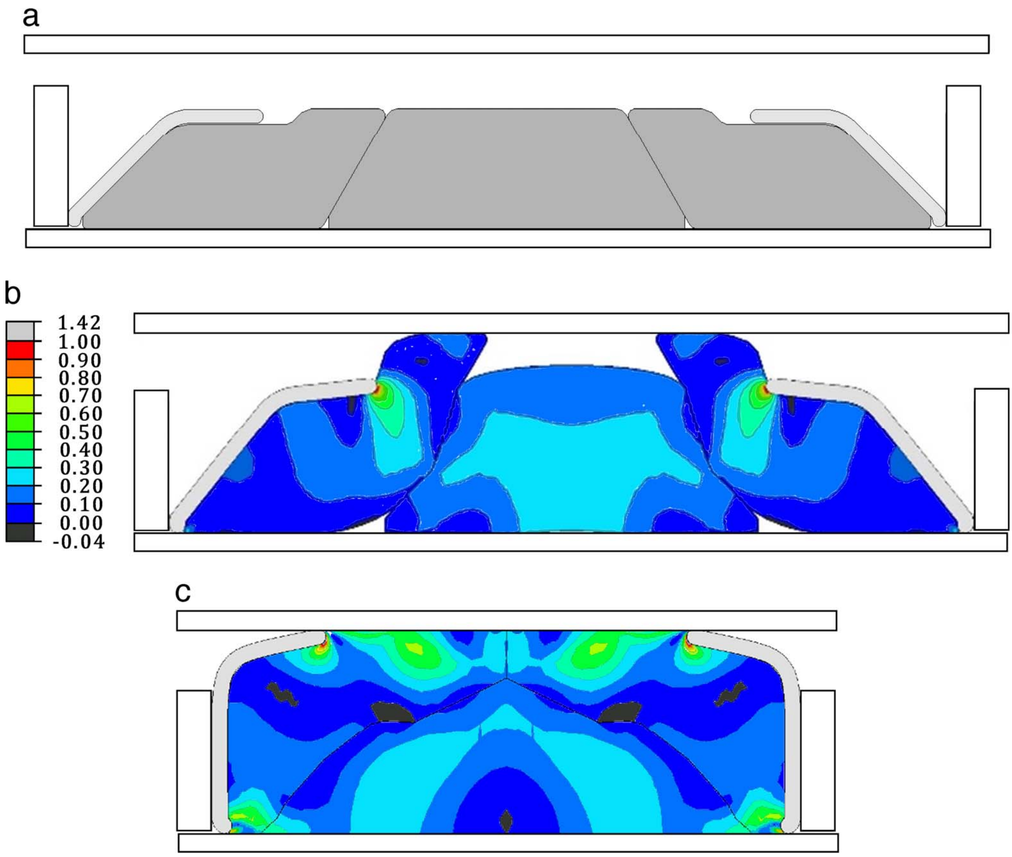


FIG. 3. — Packer seal assembly (a) prior to setting; (b) initiating contact on its outer diameter; (c) fully packed off. Maximum in-plane strain (nominal) is shown for the rubber elements in (b) and (c).

Considering the strains in Figure 3 and the material response in Figure 4, modeling packer seals requires accurate simulation of both low and high strain nonlinearities. To this end, some existing models for the hyperelastic response of filled elastomers are discussed. Concepts from these are used to propose a simple phenomenological hyperelastic model that accurately models both low and high strain nonlinearities. Curve fitting by inspection is also demonstrated. Finite elasticity theory necessary for numerical implementation in Abaqus/Explicit is introduced, and UHYPER and VUMAT material subroutines are shown.

STRAIN ENERGY FUNCTIONS

The hyperelastic response of an isotropic rubbery material is often modeled with a strain energy function (SEF).[4] An SEF describes how energy is stored in a material as it deforms and encodes the stress–strain response in different loading modes. Dozens of SEFs have been proposed and extensively reviewed for unfilled rubbers. For example, 43 SEFs are reviewed by Steinmann et al.[13] and Dal et al.[14] However, there are no reviews as comprehensive as these for filled rubbers, although studies of limited scope are available.[7,10,15] Even considering typical studies on filled rubbers, the shear moduli of some oilfield sealing materials are exceptionally high and warrant separate attention.

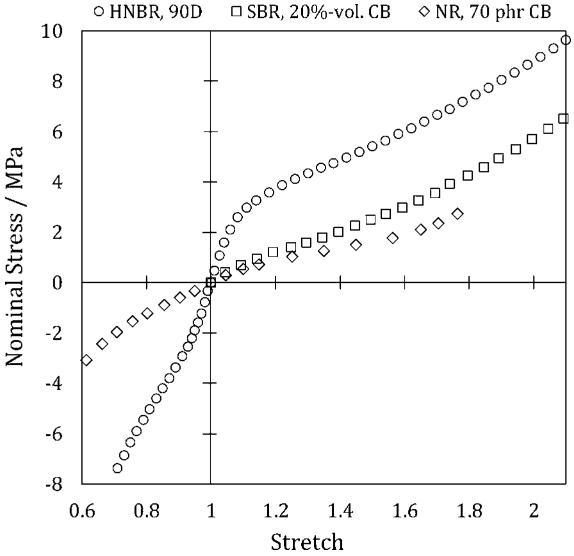


FIG. 4. — Stress–stretch data from an HNBR packer seal material, a 20% volume CB-filled SBR,[10] and a 70 phr CB-filled NR.[7]

When constructing SEFs, it is generally necessary to consider energy contributions due to both distortional (volume-preserving, isochoric) and volumetric (shape-preserving) deformations. Because the bulk modulus (κ) of rubber is much greater than its shear modulus, rubber is often modeled as incompressible and volumetric contributions are neglected.[4] This assumption is adopted initially, but is not accurate for highly constrained rubber seals.[16]

Using arguments based on the micromechanics of the polymer network of rubber, Treloar gave the simplest, widely-accepted SEF for rubbery materials:[17]

$$W = \frac{1}{2}NkT(\lambda_1^2 + \lambda_2^2 + \lambda_3^2 - 3) \tag{2}$$

where N is the number of polymer chains per unit volume, $\boldsymbol{k}$ is Boltzmann's constant, T is absolute temperature, and λ_i's are principal stretch ratios in principal directions. By adapting Hooke's law for finite deformations, Rivlin derived a modified form of Eq. 2 with $G = NkT$.[18] The SEFs are often called the (statistical) Gaussian model and neo-Hookean model, respectively. Adopting Rivlin's form, letting $\lambda_1 = \lambda$, and using the incompressibility condition ($\lambda_1\lambda_2\lambda_3 = 1$) to substitute $\lambda_2 = \lambda_3 = 1/\sqrt{\lambda}$ into Eq. 2, uniaxial stresses are computed as follows:

$$\sigma_u = \frac{dW}{d\lambda} = G(\lambda - \lambda^{-2}) \tag{3}$$

where σ_u is nominal stress and λ is principal stretch in the direction of the applied load. When fit to the initial modulus of the HNBR seal material, Eq. 3 grossly overpredicts stress magnitudes (Figure 5a). The overall fit can be improved by reducing the shear modulus, but the Payne effect and finite-extensibility are not captured. Table I shows errors from the neo-Hookean models using the following equation:

$$\text{Error} = \sqrt{\frac{1}{N_p}\sum_i \left[\left(\frac{\sigma_{i,\text{SEF}} - \sigma_{i,\text{data}}}{\sigma_{i,\text{data}}}\right)^2\right]} \tag{4}$$

where N_p is the total number of measurement points, $\sigma_{i,data}$ are stress data points, and $\sigma_{i,SEF}$ are stresses predicted by the SEF. The low and high strain limitations of the neo-Hookean SEF are well known. The latter, in particular, has received thorough attention, and the majority of SEFs that have been proposed can simulate finite-extensibility.[19,20]

To arrive at an SEF that can accurately simulate a strong Payne effect, first consider the general polynomial expansion given by Rivlin:[21]

$$W = \sum_{i,j,k=0}^{n} C_{ijk}(I_1 - 3)^i (I_2 - 3)^j (I_3 - 1)^k \tag{5}$$

where C_{ijk} are fitting parameters, $C_{000} = 0$, n is the model order, and (i, j, k) are positive integers. I_1, I_2, and I_3 are called invariants and may be computed from principal stretches as follows:

$$\begin{aligned} I_1 &= \lambda_1^2 + \lambda_2^2 + \lambda_3^2 \\ I_2 &= \lambda_1^2\lambda_2^2 + \lambda_2^2\lambda_3^2 + \lambda_1^2\lambda_3^2 \\ I_3 &= \lambda_1^2\lambda_2^2\lambda_3^2 \end{aligned} \tag{6}$$

For an incompressible material, $I_3 = 1$ and Eq. 5 simplifies to

$$W = \sum_{i,j=0}^{n} C_{ij}(I_1 - 3)^i (I_2 - 3)^j \tag{7}$$

The neo-Hookean SEF is recovered as the simplest particular form of Eq. 7, and researchers have proposed different polynomial expansions.[22] Invariant-based SEFs need not follow the series expansion form of Eq. 7,[23] but regardless of what form is chosen, several authors highlight the importance of retaining both I_1 and I_2 when constructing SEFs.[24] Some researchers have shown that in strain regimes of practical interest (up to ~100%), expansions strictly in terms of I_1 can be accurate.[25] For example, Yeoh proposed the following SEF and corresponding uniaxial stress for filled elastomers:[7]

$$W = C_{10}(I_1 - 3) + C_{20}(I_1 - 3)^2 + C_{30}(I_1 - 3)^3 \tag{8}$$

$$\sigma_u = \frac{dW}{d\lambda} = \frac{\partial W}{\partial I_1}\frac{\partial I_1}{\partial \lambda} = 2(\lambda - \lambda^{-2})\left[C_{10} + 2C_{20}(I_1 - 3) + 3C_{30}(I_1 - 3)^2\right] \tag{9}$$

When fitting the SEF, parameters are usually ordered with $C_{10} > |C_{20}| > C_{30}$ and only C_{20} is negative. Following these conventions, Figure 5b shows two Yeoh SEF fits to the HNBR data. Both fits are better than the neo-Hookean SEF (Table I), but neither accurately captures the Payne effect. Best fit parameters were determined using the Levenberg–Marquardt (LM) algorithm in Fortran as given by Press et al.,[26] and its extrapolated response is overly stiff. Recalling the strain contours of Figure 3, this is of little consequence for global strains in packer seals, but the strains (and hence energies) are in gross error locally. To mitigate this, the parameters were adjusted by inspection, but at the expense of low strain accuracy. There is not enough flexibility in the Yeoh SEF to get an accurate fit to the HNBR data at both low and high strains.

To better address the low strain nonlinearity in filled rubbers, Amin, Alam, and Okui (AAO) proposed an SEF of the following form:[9]

$$W = K_1(I_1 - 3) + K_2(I_1 - 3)^p + K_3(I_1 - 3)^q \tag{10}$$

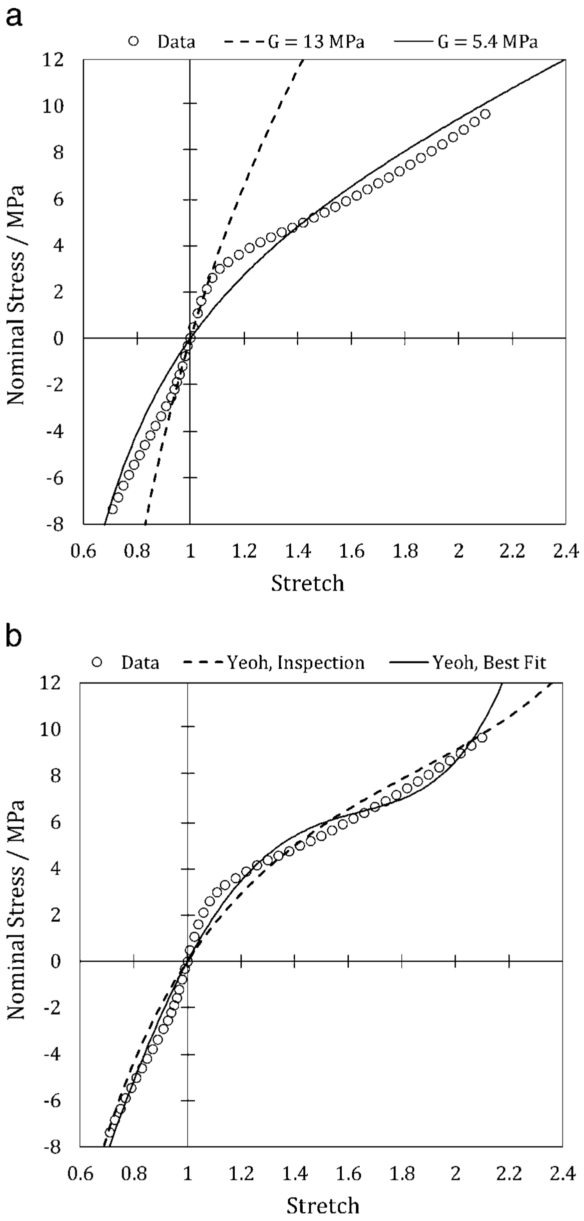


FIG. 5. — SEFs fit to data for HNBR seal material: (a) neo-Hookean; (b) Yeoh: coefficients by inspection, $(C_{10}, C_{20}, C_{30}) = (2.9, -0.15, 0.025)$ MPa, and best fit $(C_{10}, C_{20}, C_{30}) = (3.6, -0.84, 0.185)$ MPa.

where K_i's replace C_{ij}'s to avoid the convention of Eq. 7 in which subscripts correspond to integer exponents, and (p, q) are real exponents with constraints $1 \leq p \leq 2$ and $q \geq 2$. Yamashita and Kawabata had already proposed the first and third terms.[27] Amin added the second term and the constraint on p to better model the Payne effect. Figure 6a shows one of Amin's fits to the HDR material of Figure 2. Stresses have been computed following the same logic as Eq. 9:

TABLE I
ERRORS FOR DIFFERENT SEFS WHEN FIT TO HNBR SEALING MATERIAL DATA

Model	Fitting method	Comment	Error
Neo-Hookean	Inspection	$G = 13$ MPa	1.103
Neo-Hookean	Inspection	$G = 5.4$ MPa	0.282
Yeoh	Inspection	—	0.257
Yeoh	LM algorithm	—	0.188
gen-Yeoh	Inspection	—	0.043
DDT	LM algorithm	$D = 0$	0.042
DDT	LM algorithm then D by inspection	$D = 0.05$	0.034
gen-Yeoh	LM algorithm	—	0.027

$$\sigma_u = 2(\lambda - \lambda^{-2})\left[K_1 + pK_2(I_1 - 3)^{p-1} + qK_3(I_1 - 3)^{q-1}\right] \tag{11}$$

The AAO SEF has limited ability to simulate low strain inflections in UC data. Figure 6b illustrates this by plotting the same model as Figure 6a, but with the bounding limits of p (gray trendlines) replacing $p = 1.25$ (black trendline). No choice of p accurately reproduces the behavior at the lowest strains.

A drastic improvement in simulating the Payne effect was given by Davies, De, and Thomas (DDT).[25] They proposed an SEF and uniaxial stresses given by

$$W = K_1(I_1 - 3 + D^2)^m + K_3(I_1 - 3)^2 \tag{12}$$

$$\sigma_u = 2(\lambda - \lambda^{-2})\left[mK_1(I_1 - 3 + D^2)^{m-1} + 2K_3(I_1 - 3)\right] \tag{13}$$

where $0 < m \leq 1$ acts on the leading term of the SEF and $0 \leq D << 1$ is introduced for a reason that will be explained later. For the moment, let $D = 0$. The strength of the DDT SEF lies in its first term. When $m < 1$, stresses at the lowest strains are amplified and a strong Payne effect can be accurately modeled (Figure 7a).

Combining concepts from the Yeoh, AAO, and DDT SEFs, the following SEF and corresponding stresses are proposed:

$$W = K_1(I_1 - 3)^m + K_2(I_1 - 3)^p + K_3(I_1 - 3)^q \tag{14}$$

$$\sigma_u = 2(\lambda - \lambda^{-2})\left[mK_1(I_1 - 3)^{m-1} + pK_2(I_1 - 3)^{p-1} + qK_3(I_1 - 3)^{q-1}\right] \tag{15}$$

The model will be called the generalized Yeoh (gen-Yeoh) SEF.[28] It is conceptually similar to the invariant expansions given by Swanson[29] and Lopez-Pamies.[30] When fitting the SEF to highly filled materials such as the HNBR sealing material, the following constraints on parameters are useful: $K_1 > 0$; $K_2 \leq 0$; $K_3 \geq 0$; $K_1 > |K_2| > K_3$; $0.7 \leq m < 1$; $m < p < q$. Figure 7b shows the model's slight improvement over the DDT SEF in the stretch range of 0.7–0.92. Table II shows composite errors when fitting the AAO, DDT, and gen-Yeoh SEFs to Amin's data.

It is interesting that Amin imposed the constraint $1 \leq p \leq 2$ since allowing $p < 1$ better simulates the Payne effect. It is likely that he required $p \geq 1$ to avoid a mathematically indeterminate stress condition in the unstrained state. To illustrate, consider Eq. 15 with $(K_1, K_2, K_3) = (1, 0, 0)$, $(m, p, q) = (0.9, 2, 3)$, $\lambda = 1$, and $I_1 = 3$:

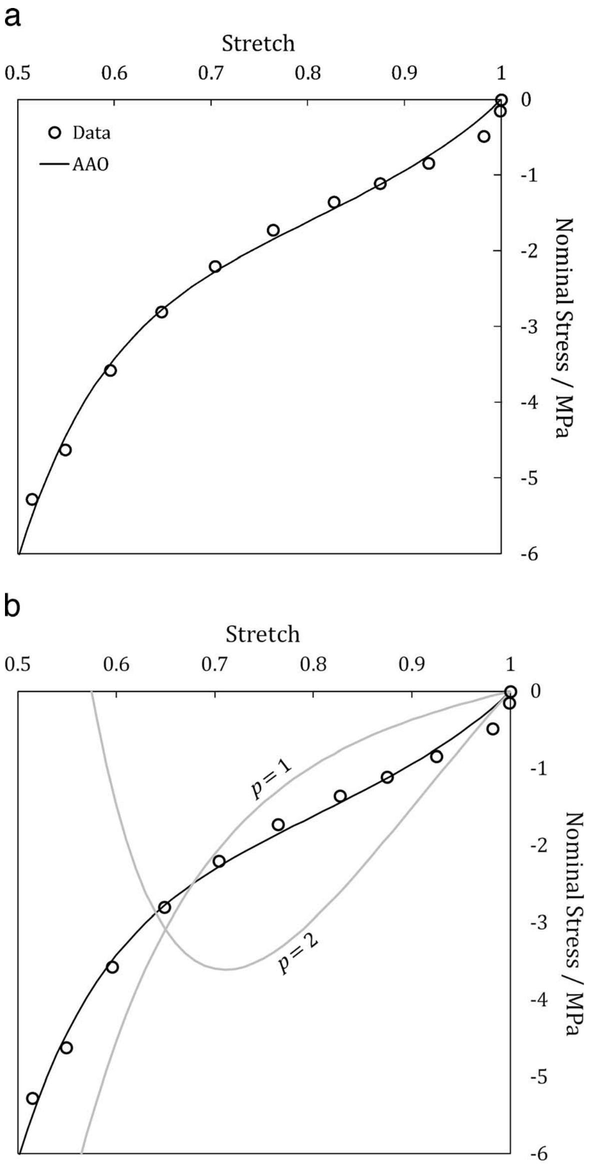


FIG. 6. — AAO SEF fit to UC data for HDR material: (a) $(K_1, K_2, K_3) = (2.35, -1.82, 0.37)$ MPa, $(p, q) = (1.25, 2)$; (b) with best fit $p = 1.25$, upper bound ($p = 2$), and lower bound ($p = 1$), all other parameters being the same.

$$\sigma_u = 2(1 - 1^{-2})\left[0.9(3-3)^{-0.1}\right] = \frac{(2)(0)(0.9)}{(3-3)^{0.1}} = \frac{0}{0} \tag{16}$$

Despite this indeterminate form, the stress must be zero in the unstrained state because of the physics being modeled. Nevertheless, the indeterminacy must be addressed for numerical implementation of the gen-Yeoh SEF. Refer to Appendices A and C for details on handling this issue with UHYPER and VUMAT subroutines. The DDT SEF circumvents the problem with parameter $D > 0$ because it guarantees a finite denominator in the stress equation.[31] An added benefit of D is that it allows precise tuning of stress at the very lowest strains, typically

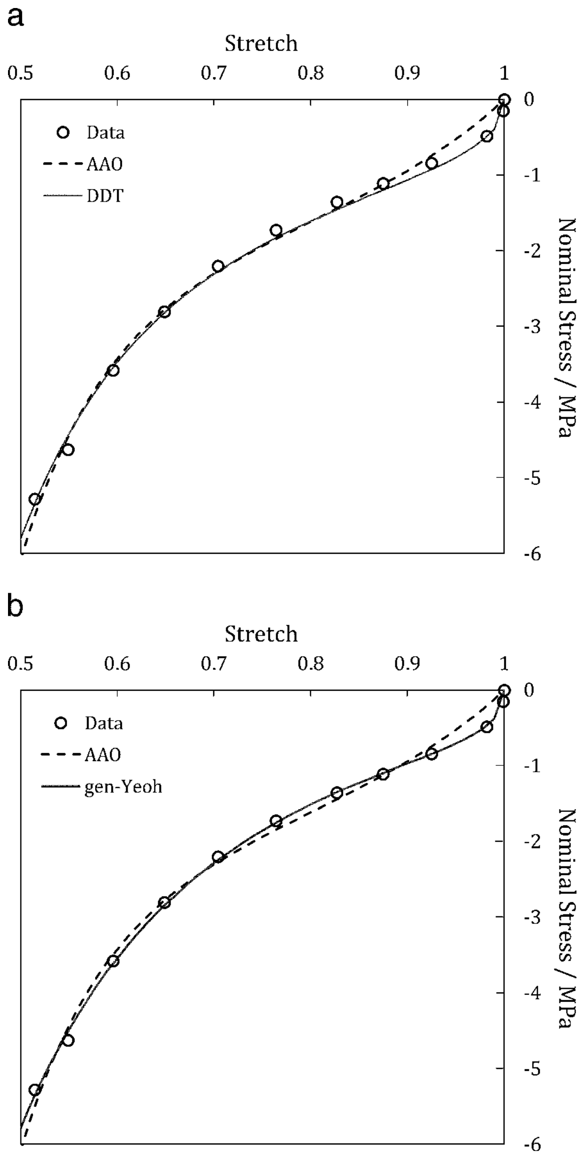


FIG. 7. — AAO SEF fit to Amin's HDR data and compared to (a) DDT SEF with $(K_1, K_3)=(0.81, 0.12)$ MPa, $(m, D)=(0.7, 0)$; (b) gen-Yeoh SEF with $(K_1, K_2, K_3)=(2.43, -1.87, 0.35)$ MPa, $(m, p, q)=(0.75, 0.8, 1.39)$.

TABLE II
ERRORS FOR DIFFERENT SEFS WHEN FIT TO AMIN'S HDR DATA

Model	Fitting method	Error
AAO	Best fit by Amin[9]	0.312
DDT	LM algorithm	0.051
gen-Yeoh	LM algorithm	0.013

in a range not discernable on a linear stress–stretch plot. However, $D > 0$ introduces finite energy in the rest state, so a modeler must ensure its contribution to the total system energy remains small.

CURVE FITTING THE DDT AND GEN-YEOH SEFS

Because of simple mathematical structure, the parameters of the DDT and gen-Yeoh SEFs can be determined to a good degree of accuracy by inspection. To illustrate, consider Figure 8a that plots data from the HNBR sealing material on log axes. The *x* axis is expressed in terms of the first invariant, and the *y* axis uses a measure of stiffness called reduced stress and for uniaxial deformation is given by

$$\widehat{\sigma} = \frac{\sigma_u}{\lambda - \lambda^{-2}} \tag{17}$$

For reference, the figure includes stretch values, λ_{ut}, that correspond to $\log(I_1 - 3)$. The UT and UC data follow similar trends that indicate they are well conditioned for fitting with I_1-based SEFs.[25] This condition does not always occur, so it is ideal to have data from more than one loading mode to confirm the approach.

To fit the gen-Yeoh SEF to the data, first adjust vertical position with K_1 and set the slope of the linear region with m (Figure 8b). Next assign a guess value $q = 2$ and adjust K_3 to capture finite-extensibility. Tune K_1, m, and q as necessary to improve the fit (Figure 8c). Finally, adjust K_2 and p to better tune the overall fit and adjust other parameters as necessary (Figure 8d). This last step is the most difficult because best fit parameters are not unique in nonlinear regression. As parameters are tuned, it is helpful to monitor an error metric (e.g., Eq. 4) and ensure parameter adjustments reduce the error. The data points less than $\log(I_1 - 3) = –2.5$ are subject to large measurement error, so they are not considered in the error equation during the fitting process.

In Figure 8d, the double-log plot amplifies errors at the lowest strains, making them appear deceptively large. Figure 9a shows the SEF fits well at the lowest strains when linear axes are used. Figure 9b shows a gen-Yeoh fit using the LM algorithm which is almost identical to the fit determined by inspection. Figure 9b also shows that stress predictions using the fitted coefficients in planar and equibiaxial loading modes behave reasonably even though data in those modes were not considered.

Finally, Figure 10 shows curve fits with the gen-Yeoh and DDT SEFs on a double-log plot. All parameters were determined with the LM algorithm except D, which was manually added to correct the lowest strains in the DDT SEF. Errors for the different fits are in Table I.

FINITE ELASTICITY THEORY FOR A HYPERELASTIC VUMAT

Finite elasticity theory necessary for numerical implementation of the gen-Yeoh SEF in Abaqus/Explicit is explored. A more detailed account of the subject is given by Holzapfel.[1]

The configuration of a body can be described by position vectors that locate every material point in the body with respect to a fixed coordinate system. Upon deformation, a material point, P, in a body with position vector $\mathbf{X}$ in a reference (undeformed) configuration moves to a deformed position, $\mathbf{x}$ (Figure 11). The strain at the material point can be approximated by determining how a differential line element $\mathrm{d}\mathbf{S}$ deforms to $\mathrm{d}\mathbf{s}$. The differential vectors are related by

$$\mathrm{d}\mathbf{s} = \frac{\partial \mathbf{x}}{\partial \mathbf{X}} \mathrm{d}\mathbf{S} = \mathbf{F}\mathrm{d}\mathbf{S} \tag{18}$$

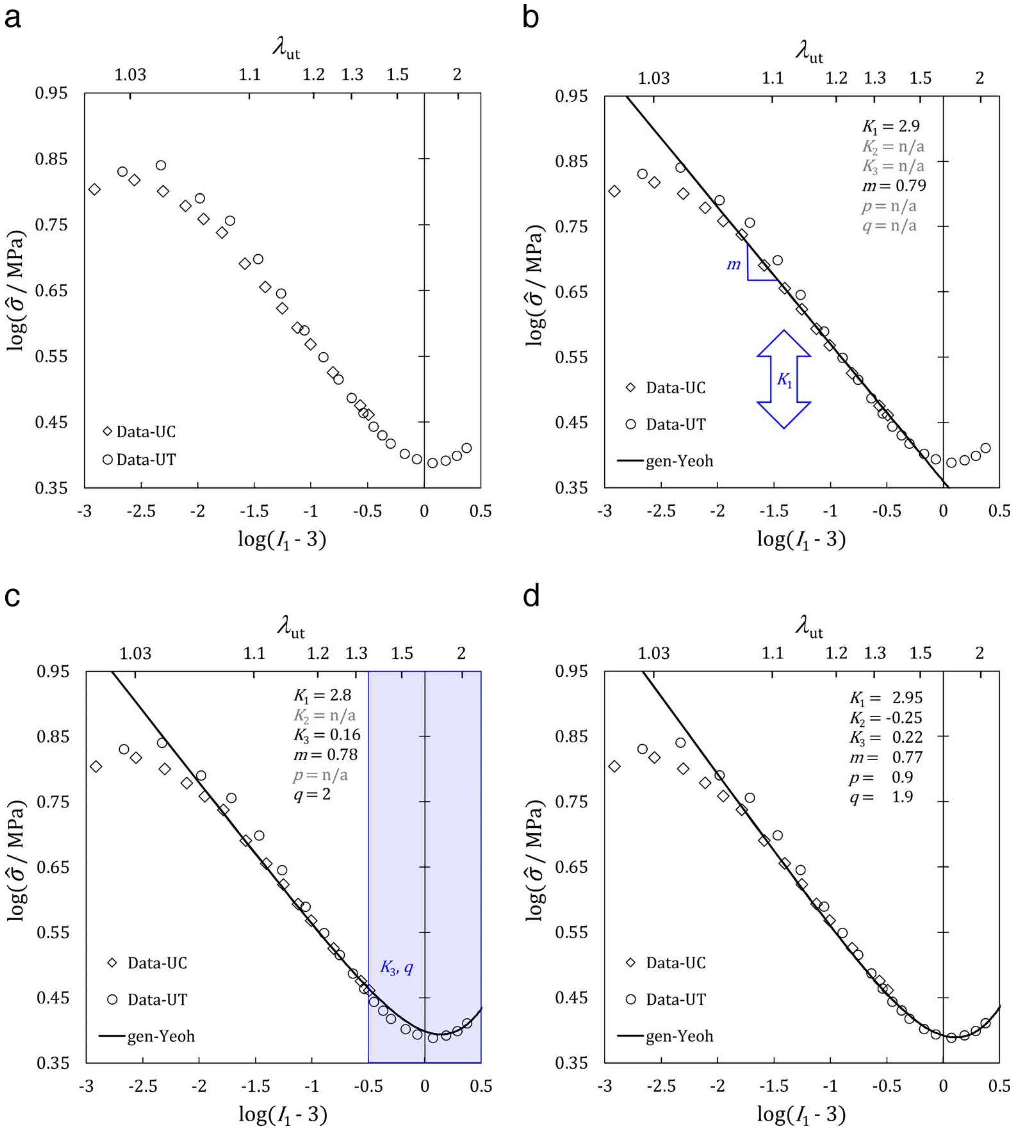


FIG. 8. — Fitting the gen-Yeoh SEF by inspection: (a) uniaxial data; (b) setting K_1 and m; (c) setting K_3 and q; (d) setting K_2 and p.

where $\mathbf{F} \equiv \partial\mathbf{x}/\partial\mathbf{X}$ is a second-order tensor called the deformation gradient. The collection of all material points in a body and their local deformation describes the kinematics of the entire body.

The deformation gradient encodes all information regarding strains and rigid body rotations. However, it does not account for rigid body translations because the vectors d$\mathbf{S}$ and d$\mathbf{s}$ always have the same magnitude and direction for any rigid translation. In solid mechanics, one is often most concerned with strains because only they contribute to stresses. Therefore, it is helpful to separate rigid body rotations from strains in the deformation gradient. This is achieved through the polar decomposition of $\mathbf{F}$:

$$\mathbf{F} = \mathbf{RU} \tag{19}$$

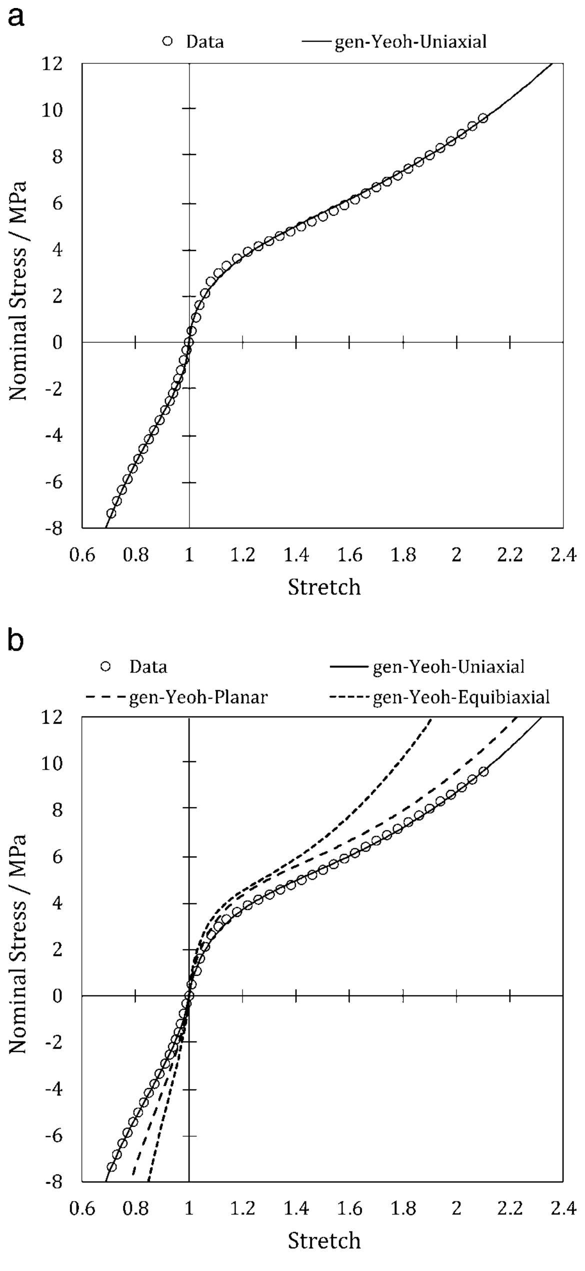


FIG. 9. — Stress–stretch plots with the gen-Yeoh SEF for HNBR sealing material on linear axes: (a) same parameters as Figure 8d; (b) parameters from LM algorithm, $(K_1, K_2, K_3) = (5.38, -2.85, 0.4)$ MPa, $(m, p, q) = (0.89, 1.08, 1.85)$.

where **R** is the orthogonal rotation tensor and **U** is the right stretch tensor. The stretch tensor is symmetric and provides a direct measure of strains. It has principal values that can be computed from the tensor invariants of **U**:

$$i_1 = \lambda_1 + \lambda_2 + \lambda_3 \qquad i_2 = \lambda_1\lambda_2 + \lambda_2\lambda_3 + \lambda_1\lambda_3 \qquad i_3 = \lambda_1\lambda_2\lambda_3 \tag{20}$$

where λ_i's are principal stretches[32] and i_3 captures volume changes during deformation and is called

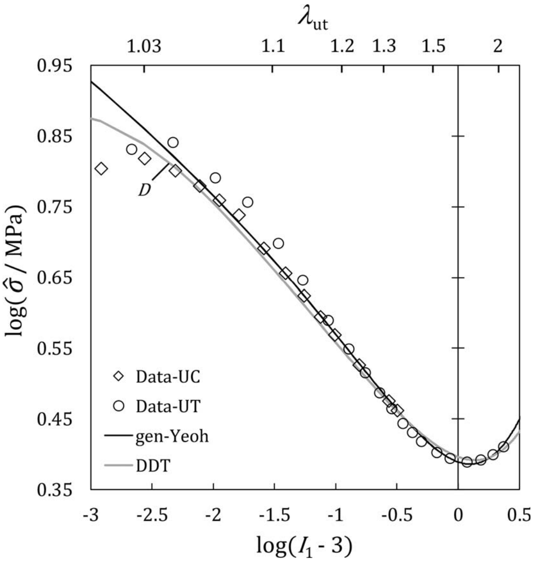


FIG. 10. — Final curve fits for DDT and gen-Yeoh SEFs for HNBR sealing material; gen-Yeoh parameters: same as Figure 9b; DDT parameters: $(K_1, K_3) = (2.78, 0.16)$ MPa, $(m, D) = (0.78, 0.05)$.

the volume ratio, Jacobian, or Jacobian determinant. It may be expressed as follows:

$$J = \det\mathbf{F} = \lambda_1\lambda_2\lambda_3 \tag{21}$$

where det is the determinant operator. In addition to the stretch tensor, other strain tensors may be defined for mathematical convenience. For example, the left Cauchy–Green strain tensor will be useful:

$$\mathbf{B} = \mathbf{F}\mathbf{F}^{\mathrm{T}} = \mathbf{R}\mathbf{U}^2\mathbf{R}^{\mathrm{T}} \tag{22}$$

where T is the transpose operator. The second equality follows from properties $(\mathbf{RU})^{\mathrm{T}} = \mathbf{U}^{\mathrm{T}}\mathbf{R}^{\mathrm{T}}$ and $\mathbf{U}^{\mathrm{T}} = \mathbf{U}$ due to symmetry. This strain tensor is simply a rotation of the square of the stretch tensor. The eigenvalues of $\mathbf{B}$ are squares of the principal stretches from $\mathbf{U}$, and its invariants are identical to those in Eq. 6:

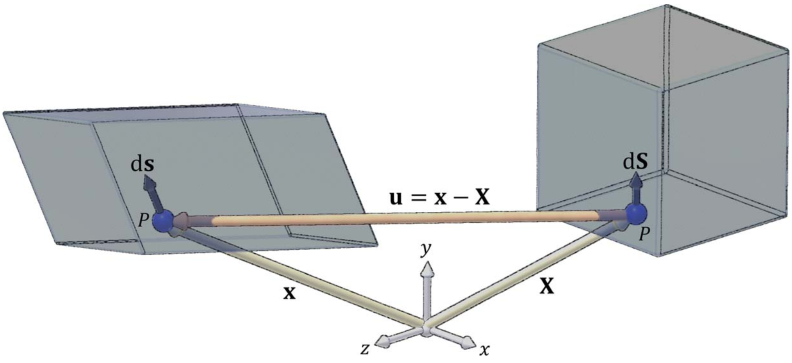


FIG. 11. — Vectors associated with finite deformation of differential line element d**S** at point *P* in a body.

$$I_1 = \lambda_1^2 + \lambda_2^2 + \lambda_3^2 \qquad I_2 = \lambda_1^2\lambda_2^2 + \lambda_2^2\lambda_3^2 + \lambda_1^2\lambda_3^2 \qquad I_3 = \lambda_1^2\lambda_2^2\lambda_3^2 \tag{23}$$

For compressible hyperelasticity, it is often convenient to split an SEF into isochoric and volumetric contributions:

$$W = W_{\text{iso}}(\overline{\mathbf{B}}) + W_{\text{vol}}(J) \tag{24}$$

where $\bar{\mathbf{B}}$ is the modified left Cauchy–Green strain tensor. It excludes any volumetric energy contributions and is computed from $\mathbf{B}$ as follows:

$$\overline{\mathbf{B}} = J^{-2/3}\mathbf{B} \tag{25}$$

Invariants of this modified strain tensor may be expressed compactly as follows:

$$\bar{I}_1 = \text{tr}(\overline{\mathbf{B}})$$

$$\bar{I}_2 = \frac{1}{2}\left[\left(\text{tr}(\overline{\mathbf{B}})\right)^2 - \text{tr}(\overline{\mathbf{B}}^2)\right]$$

$$\bar{I}_3 = \det(\overline{\mathbf{B}}) = 1 \tag{26}$$

where tr is the trace operator. These equations can be expanded to the same form as Eq. 23 with modified principal stretches $\bar{\lambda}_i$'s replacing λ_i's. Another useful definition is the modified stretch tensor:

$$\overline{\mathbf{U}} = J^{-1/3}\mathbf{U} \tag{27}$$

Implementing the expressions above, Bergström derives the following stress equation:[33]

$$\boldsymbol{\sigma} = \frac{2}{J}\left(\frac{\partial W}{\partial \bar{I}_1}\right)\left(\overline{\mathbf{B}} - \frac{1}{3}\text{tr}(\overline{\mathbf{B}})\mathbf{I}\right) + \frac{\partial W}{\partial J}\mathbf{I} \tag{28}$$

where $\boldsymbol{\sigma}$ is the Cauchy (true) stress tensor and $\mathbf{I}$ is the identity tensor. Equation 28 expresses stresses with respect to coordinates in the deformed configuration. In Abaqus/Explicit, VUMAT subroutines require stresses to be expressed in the reference configuration. This is achieved by using the rotation tensor to convert the stress equation to its corotational form:[33]

$$\boldsymbol{\sigma}_{\text{co}} = \mathbf{R}^{\text{T}}\boldsymbol{\sigma}\mathbf{R} \tag{29}$$

Combining Eqs. 22, 25, 27, 28, and 29 and applying the properties $\mathbf{R}^{\text{T}}\mathbf{R} = \mathbf{R}\mathbf{R}^{\text{T}} = \mathbf{I}$ and tr($\mathbf{R}\bar{\mathbf{U}}^2\mathbf{R}^{\text{T}}$ = tr($\bar{\mathbf{U}}^2$), corotational stresses may be expressed as follows:

$$\boldsymbol{\sigma}_{\text{co}} = \frac{2}{J}\left(\frac{\partial W}{\partial \bar{I}_1}\right)\left(\overline{\mathbf{U}}^2 - \frac{1}{3}\text{tr}(\overline{\mathbf{U}}^2)\mathbf{I}\right) + \frac{\partial W}{\partial J}\mathbf{I} \tag{30}$$

The forms of Eqs. 28 and 30 are identical, but it is important to recognize that $\bar{\mathbf{B}} \neq \bar{\mathbf{U}}^2$; each tensor returns stress in different configurations.

Before implementing the theory above, the gen-Yeoh SEF of Eq. 14 must be cast into a compressible form:

$$W = K_1(\bar{I}_1 - 3)^m + K_2(\bar{I}_1 - 3)^p + K_3(\bar{I}_1 - 3)^q + \frac{1}{D_1}(J-1)^2 \tag{31}$$

where $D_1 = 2/\kappa$ has been defined to follow an Abaqus convention. The following steps will execute a hyperelastic VUMAT with the gen-Yeoh SEF in Abaqus:

1. Compute the strain tensor, $\mathbf{B}^* = \mathbf{U}^2$.
2. Compute the volume ratio, $J = \det(\mathbf{U})$.
3. Compute the modified strain tensor, $\bar{\mathbf{B}}^* = J^{-2/3}\mathbf{B}^*$.
4. Compute derivatives of the strain energy function, $\partial W/\partial \bar{I}_1$ and $\partial W/\partial J$.
5. Compute corotational stresses, $\boldsymbol{\sigma}_{co} = 2/J\ (\partial W/\partial \bar{I}_1)(\bar{\mathbf{B}}^* - 1/3\mathrm{tr}(\bar{\mathbf{B}}^*)\mathbf{I}) + (\partial W/\partial J)\mathbf{I}$.
6. Compute the internal energy density, for example, with direct application of the SEF.

As a mathematical shortcut, the strain tensor $\mathbf{B}^*$ has been defined to remove rotations from the left Cauchy–Green strain tensor of Eq. 22, and a slightly altered version of Eq. 30 is used. The corotational stresses computed in the fifth step are identical to those from Eq. 30.

VALIDATION OF THE GEN-YEOH UHYPER AND VUMAT SUBROUTINES

Details on UHYPER and VUMAT subroutines and their implementation are given in Appendices A through D. This section summarizes results. The UHYPER subroutine for the gen-Yeoh SEF was initially tested with the best fit Yeoh parameters from Figure 5b and compared to Abaqus's built-in Yeoh model. A Poisson's ratio (ν) of 0.495 was assumed, and it was converted to compressibility parameter D_1:[34]

$$D_1 = \frac{3(1-2\nu)}{G(1+\nu)} = 2.78707 \times 10^{-3}\mathrm{MPa}^{-1} \tag{32}$$

where $G = 2C_{10}$ in the Yeoh model and $G = 2K_1$ in the gen-Yeoh model.

Homogeneous modes of deformation, including simple shear, were tested on a unit cube with one linear, hybrid, reduced integration brick element (C3D8RH in Abaqus nomenclature). With only one exception, stresses, strains, energy density, and volume matched to nine decimal places, the maximum precision in Abaqus's visualization module. In the pure deformation modes (e.g., ET), finite stresses were computed in the principal direction in which no load was applied, and these differed in the eighth decimal place. These stresses are an artefact of the numerical solution and only amounted to 0.001% of the stress values in the directions of applied deformation. These artificial stresses also arose with a fully incompressible element, so the issue does not arise solely from compressibility in the material model.

To test inhomogeneous deformations, a unit cube (1 mm^3) was meshed with 20^3 elements. The cube was fixed on its bottom surface and twisted through 60° on its top surface. Figure 12 shows maximum principal nominal strains in the cube. All maximum and minimum principal stresses and strains matched to nine decimal places. Element volumes and energy densities also matched to nine decimal places.

The gen-Yeoh model was run without issue using the best fit parameters in Figure 9b and $D_1 = 1.86495 \times 10^{-3}$ MPa^{-1}. Simulation times with the built-in Yeoh model, UHYPER Yeoh model, and UHYPER gen-Yeoh model were 161, 163, and 165 s, respectively, using full nodal precision and eight processors on a 12 core Intel Xeon E5-2620 CPU.

A density of 1 g/cm^3 was used for simulations in Abaqus/Explicit. Mass scaling was required to prevent the stable time increment from becoming too small for the computer's numerical precision. Scaling factors up to 10^7 affected stress, strain, volume, and strain energy density less than 1%. A factor of 10^3 was used for final validation of the VUMAT subroutine.

When comparing the built-in and VUMAT Yeoh models in homogeneous modes of deformation, artificial stresses again occurred in the directions in which no loads were applied. However, they were two orders of magnitude lower than their Abaqus/Standard counterparts. Consequently, some small but negligible differences between the built-in and VUMAT solutions were found with homogeneous deformations.

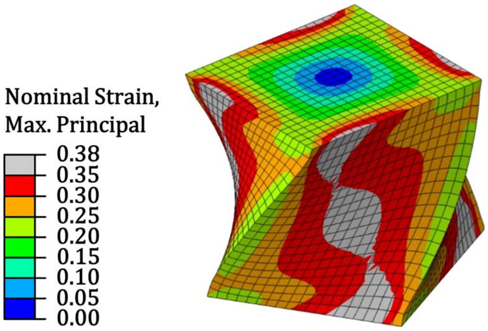


FIG. 12. — Maximum principal nominal strain contours in cube when twisted 60°.

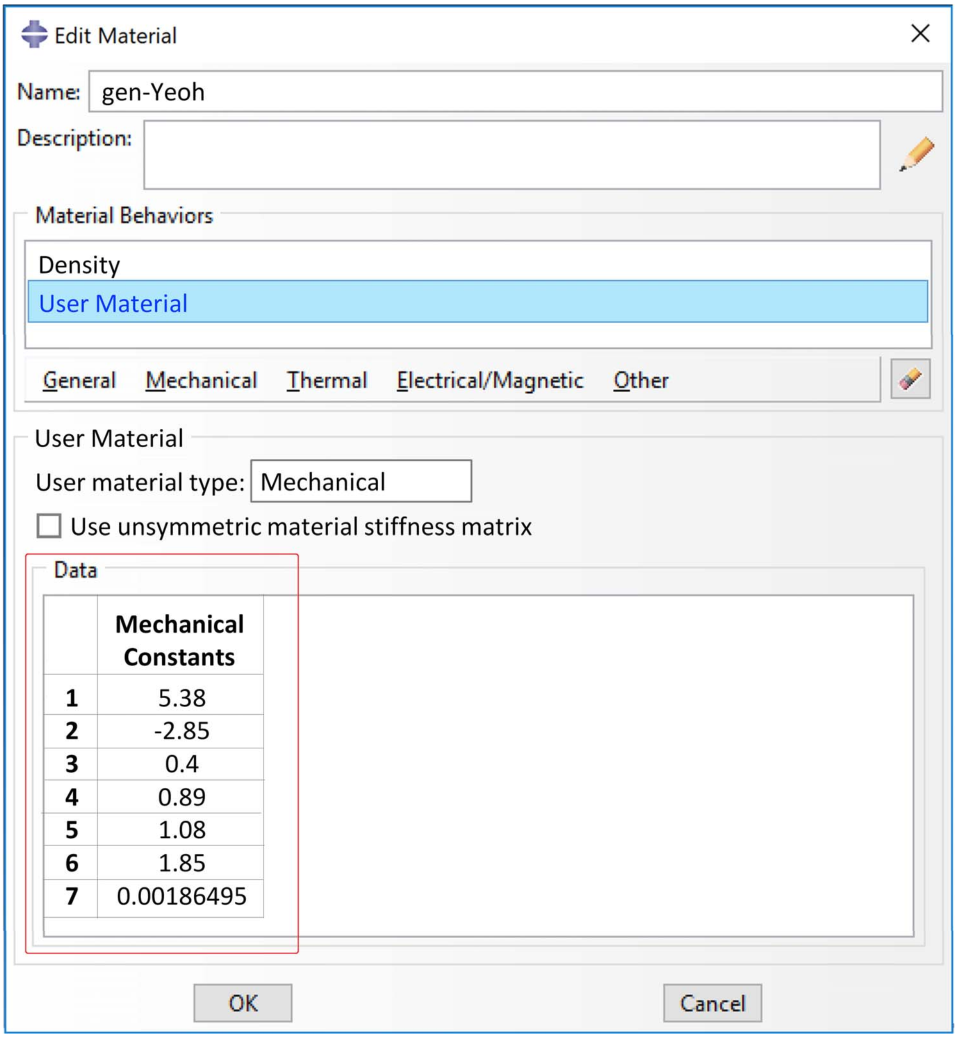
Small discrepancies were found in the twisted cube solutions when using the built-in and VUMAT Yeoh models. These discrepancies are negligible in terms of practical engineering design. Table III summarizes the percent differences for some selected field outputs with the different material solution techniques.

Discrepancies were expected when testing inhomogeneous deformations because Abaqus/Explicit uses the Jaumann objective stress rate with built-in material models and the Green–Naghdi objective stress rate with VUMAT models. These have differences when finite rotations and shear occur simultaneously.[35] Furthermore, Vorel and Bažant argue that both of these stress rates are not generally accurate in numerical simulations.[36] They recommend converting to the Truesdell stress rate, but this has not been pursued here. Nevertheless, it is a topic worth further exploration as it may reconcile the discrepancies between the solutions of the built-in and VUMAT Yeoh models. This could become particularly important for problems with larger shear and rotation.

The gen-Yeoh model ran without issue using the VUMAT and the same material parameters as the gen-Yeoh UHYPER subroutine. Using double precision with eight solution domains, simulation times with the built-in Yeoh, VUMAT Yeoh, and VUMAT gen-Yeoh models were 33.15, 31.35, and 39.32 min, respectively. The VUMAT Yeoh model ran faster than the built-in Yeoh model. This result was repeatable and indicates that the VUMAT code may be simpler than the built-in routine, perhaps because the VUMAT is written specifically for the three-dimensional

TABLE III
PERCENT DIFFERENCE IN SELECTED FIELD OUTPUT VALUES USING BUILT-IN AND VUMAT YEOH MODELS

Field output	Built-in Yeoh	VUMAT Yeoh	Difference, %
Principal stress, MPa			
Maximum	8.7532	8.7839	0.35
Minimum	1.3431	1.3477	0.34
Principal strain			
Maximum	0.38557	0.38581	0.06
Minimum	−0.018584	−0.018599	0.08
Volume, mm^3			
Maximum	1.2580×10^{-4}	1.2582×10^{-4}	0.02
Minimum	1.2458×10^{-4}	1.2459×10^{-4}	0.01
Energy density, mJ/mm^3			
Maximum	1.3919	1.3913	0.04
Minimum	5.6965×10^{-3}	5.7079×10^{-3}	0.20



Name: gen-Yeoh

Description:

Material Behaviors

Density

User Material

User Material

User material type: Mechanical

□ Use unsymmetric material stiffness matrix

Data

	Mechanical Constants
1	5.38
2	-2.85
3	0.4
4	0.89
5	1.08
6	1.85
7	0.00186495

FIG. 13. — Material parameters read by props(n) variable in a VUMAT.

(3D) case. The gen-Yeoh model takes significantly longer because computations are more complex with its non-integer exponents.

CONCLUSIONS

Building on concepts from the Yeoh, AAO, and DDT SEFs, the gen-Yeoh SEF has been introduced. It is well suited to capture both low and high strain nonlinearities in highly filled elastomeric materials such as those in oilfield packer seals. In particular, the SEF accurately models materials that have a strong Payne effect. Curve fitting by inspection and with the Levenberg–Marquardt algorithm has shown that both techniques yield good fits even though model parameters are not unique. Finite elasticity theory that is necessary to implement a compressible hyperelastic VUMAT has been given, and a procedure for efficient numerical execution has been described. Codes to implement a compressible form of the gen-Yeoh SEF in Abaqus/Standard and Abaqus/

Explicit have been provided. Finite-element solutions with the built-in and user-defined Yeoh models agree well. Small differences were found in the Abaqus/Explicit solutions for inhomogeneous deformations. It is possible that these arise from differences in the objective stress rates used with built-in and VUMAT material models. Although the discrepancy is of interest, it is of little consequence for the problem studied. It could become important to reconcile the discrepancies as the magnitude of combined shear and rotation increases. Finally, the UHYPER and VUMAT Yeoh model subroutines have execution times similar to Abaqus's built-in Yeoh models, but the gen-Yeoh subroutine in Abaqus/Explicit takes longer to run when non-integer exponents are used.

ACKNOWLEDGEMENTS

The authors thank Schlumberger, Ltd. for sponsorship of this work. N.M.P. is supported by the European Commission under the Graphene Flagship Core 2 Grant 785219 (WP14 "Composites") and FET Proactive "Neurofibres" Grant 732344 as well as by the Italian Ministry of Education, University and Research under the "Departments of Excellence" Grant L. 232/2016, the ARS01-01384-PROSCAN Grant, and the PRIN-20177TTP3S.

REFERENCES

[1] G. A. Holzapfel, *Nonlinear Solid Mechanics: A Continuum Approach for Engineering*, John Wiley & Sons, Ltd., New York, 2000.

[2] "Section 3.1.4, Fitting of Rubber Test Data," in *Abaqus 6.14 Benchmarks Guide*, Dassault Systèmes Simulia Corp., Providence, RI, 2014.

[3] L. R. G. Treloar, *Trans. Faraday Soc*. **40**, 59 (1944).

[4] L. R. G. Treloar, *The Physics of Rubber Elasticity*, Oxford University Press, Oxford, UK, 1975.

[5] O. H. Yeoh, RUBBER CHEM. TECHNOL. **66**, 754 (1993).

[6] P. A. Przybylo and E. M. Arruda, RUBBER CHEM. TECHNOL. **71**, 730 (1998).

[7] O. H. Yeoh, RUBBER CHEM. TECHNOL. **63**, 792 (1990).

[8] A. R. Payne and R. E. Whittaker, RUBBER CHEM. TECHNOL. **44**, 440 (1971).

[9] A. F. S. M. Amin, M. S. Alam, and Y. Okui, *Mech. Mater*. **34**, 75 (2002).

[10] M. Fujikawa, N. Maeda, J. Yamabe, and M. Koishi, RUBBER CHEM. TECHNOL, in press.

[11] R. J. Windslow and J. J. C. Busfield, *Soft Mater.* **17**, (2019).

[12] R. J. Windslow, "Computational Modelling of Fracture Processes in Elastomeric Seals," Ph.D. Thesis, Queen Mary University of London, 2018.

[13] P. Steinmann, M. Hossain, and G. Possart, *Arch. Appl. Mech*. **82**, 1183 (2012).

[14] H. Dal, Y. Badienia, K. Acikgoz, and F. A. Denli, "A Comparative Study on Hyperelastic Constitutive Models on Rubber: State of the Art after 2006," in *Constitutive Models for Rubber XI*, Proceedings of the 11th European Conference on Constitutive Models for Rubber, Nantes, France, June 25–27, 2019, B. Huneau, J.-B. Le Cam, Y. Marco, and E. Verron, Eds., CRC Press, Boca Raton, FL, 2019.

[15] F. Carleo, E. Barbieri, R. Whear, and J. J. C. Busfield, *Polymers* **10**, 988 (2018).

[16] T. J. Peng and R. F. Landel, *J. Appl. Phys*. **46**, 2599 (1975).

[17] L. R. G. Treloar, *Trans. Faraday Soc*. **39**, 241 (1943).

[18] R. S. Rivlin, *Philos. Trans. R. Soc. Lond. Ser. A* **240**, 459 (1948).

[19] R. W. Ogden, *Proc. R. Soc. Lond. Ser. A* **326**, 565 (1972).

[20] E. Arruda and M. Boyce, *J. Mech. Phys. Solids* **41**, 389 (1993).

[21] R. S. Rivlin, "Large Elastic Deformations," in *Rheology: Theory and Applications*, F. R. Eirich, Ed., Elsevier, New York, 1956, Vol. 1, Ch. 10.

[22] A. G. James, A. Green, and G. M. Simpson, *J. Appl. Polym. Sci.* **19**, 2033 (1975).

[23] A. N. Gent, RUBBER CHEM. TECHNOL. **69**, 59 (1996).

[24] A. F. S. M. Amin, S. I. Wiraguna, A. R. Bhuiyan, and Y. Okui, *J. Eng. Mech.* **132**, 54 (2006).

[25] C. K. L. Davies, D. K. De, and A. G. Thomas, RUBBER CHEM. TECHNOL. **67**, 716 (1994).

[26] W. Press, B. Flannery, S. Teukolsky, and W. Vetterlin, *Numerical Recipes in FORTRAN 90: The Art of Parallel Scientific Computing*, 2nd Ed., Cambridge University Press, Cambridge, UK, 1992, Section 15.5.

[27] Y. Yamashita and S. Kawabata, *J. Soc. Rubber Sci. Technol. Jpn.* **65**, 517 (in Japanese) (1992).

[28] T. W. Hohenberger, R. J. Windslow, N. Pugno, and J. J. C. Busfield, *Constitutive Models for Rubber XI*, Proceedings of the 11th European Conference on Constitutive Models for Rubber, Nantes, France, June 25–27, 2019, B. Huneau, J.-B. Le Cam, Y. Marco, and E. Verron, Eds., CRC Press, Boca Raton, FL, 2019.

[29] S. Swanson, *J. Engr. Mater. Tech. (ASME)* **107**, 110 (1985).

[30] O. Lopez-Pamies, *C. R. Méca.* **338**, 3 (2010).

[31] J. J. C Busfield and A. G. Thomas, RUBBER CHEM. TECHNOL. **72**, 876 (1999).

[32] D. J. Steigmann, *Math. Mech. Solids* **7**, 393 (2002).

[33] J. S. Bergström, *Mechanics of Solid Polymers: Theory and Computational Modeling*, Elsevier, London, 2015.

[34] "Section 22.5.1, Hyperelastic Behavior of Rubberlike Materials," in *Abaqus 6.14 Analysis User's Guide*, Dassault Systèmes Simulia Corp., Providence, RI, 2014.

[35] "Section 1.5.3, Stress Rates," in *Abaqus 6.14 Theory Guide*, Dassault Systèmes Simulia Corp., Providence, RI, 2014.

[36] J. Vorel and Z. P. Bažant, *Adv. Eng. Softw.* **72**, 3 (2014).

[37] "Section 1.1.38, UHYPER"; "Section 1.1.20, VUMAT," in *Abaqus 6.14 User Subroutines Reference Guide*, Dassault Systèmes Simulia Corp., Providence, RI, 2014.

[38] N. Elyasi, K. K. Taheri, K. Narooei, and A. K. Taheri, *Biomech. Model Mechanobiol.* **16**, 1077 (2017).

[39] H. Khajehsaeid, S. Reese, J. Arghavani, and R. Naghdabadi, *Acta Mech.* **227**, 1969 (2016).

[40] S. A. Chester, "VUMAT and UMAT for a neo-Hookean material," 2008, https://web.njit.edu/~sac3/Software.html.

[41] "Writing User Subroutines with ABAQUS," Dassault Systèmes Simulia Corp., Providence, RI, 2019. https://companion.3ds.com/CompanionManager/ui/#/course/en/222402//false/false/publicSearch//abaqus//.

[*Received August 2019, Revised October 2019*]

APPENDIX A: IMPLEMENTING THE GEN-YEOH STRAIN ENERGY FUNCTION IN ABAQUS/STANDARD WITH A UHYPER SUBROUTINE

Implementing the gen-Yeoh SEF with a UHYPER subroutine only requires a user to specify Eq. 31 and appropriate derivatives,[37] which are as follows:

$$\frac{\partial W}{\partial \bar{I}_1} = mK_1(\bar{I}_1 - 3)^{m-1} + pK_2(\bar{I}_1 - 3)^{p-1} + qK_3(\bar{I}_1 - 3)^{q-1} \tag{33}$$

$$\frac{\partial^2 W}{\partial \bar{I}_1^2} = (m^2 - m)K_1(\bar{I}_1 - 3)^{m-2} + (p^2 - p)K_2(\bar{I}_1 - 3)^{p-2} + (q^2 - q)K_3(\bar{I}_1 - 3)^{q-2} \tag{34}$$

$$\frac{\partial W}{\partial J} = \frac{2}{D_1}(J - 1) \tag{35}$$

$$\frac{\partial^2 W}{\partial J^2} = \frac{2}{D_1} \tag{36}$$

The mathematical indeterminacy highlighted in Eq. 16 arises if *m*, *p*, or *q* are less than 1. Because the UHYPER subroutine requires the second derivative of the SEF, the numerical issue also arises if *m*, *p*, or *q* are non-integer and less than 2. The following logic can address the indeterminacy:

```
IF$\left(\bar{I}_1 = 3 \;\text{ AND }\; (m < 1 \;\text{ OR }\; p < 1 \;\text{ OR }\; q < 1)\right)$THEN
    $\frac{\partial W}{\partial \bar{I}_1} = 1$
ELSE
    $\frac{\partial W}{\partial \bar{I}_1} = mK_1(\bar{I}_1 - 3)^{m-1} + pK_2(\bar{I}_1 - 3)^{p-1} + qK_3(\bar{I}_1 - 3)^{q-1}$
END IF
IF$\left(\bar{I}_1 = 3 \;\text{ AND }\; (m < 2 \;\text{ OR }\; p < 2 \;\text{ OR }\; q < 2)\right)$THEN
    $\frac{\partial^2 W}{\partial \bar{I}_1^2} = 0$
ELSE
    $\frac{\partial^2 W}{\partial \bar{I}_1^2} = (m^2 - m)K_1(\bar{I}_1 - 3)^{m-2} + (p^2 - p)K_2(\bar{I}_1 - 3)^{p-2} + (q^2 - q)K_3(\bar{I}_1 - 3)^{q-2}$
END IF
```

Even though $\partial W/\partial \bar{I}_1 \rightarrow \infty$ when, for example, $m < 1$, the first IF-THEN statement assigns a finite value to $\partial W/\partial \bar{I}_1$ because Abaqus will not initialize a solution if $\partial W/\partial \bar{I}_1$ is too close to zero or tends to infinity. The choice of finite value does not appear to affect the solution as long as convergence is achieved. Abaqus can initialize a solution with $\partial^2 W/\partial \bar{I}_1^2 = 0$, so that has been used. The full code for the UHYPER subroutine is in Appendix B.

APPENDIX B: GEN-YEOH UHYPER SUBROUTINE

Disclaimer: Abaqus is sensitive to the location of the column in which lines start in subroutine files, and the code shown does not reflect proper alignment. If problems are encountered during compilation, confirm appropriate spacing with the templates and validated subroutines provided in Abaqus's documentation.

```
C **********************************************************************
C  UHYPER for gen-Yeoh Strain-Energy Function
C  ------------------------------------------
C  Author:   Travis Hohenberger
C  Date:     2019-07-11
C  E-mail:   twhohen@gmail.com
C **********************************************************************
C
C  Strain-energy function:
C
C  W = K1*(I1-3)^m + K2*(I1-3)^p + K3*(I1-3)^q + (1/D1)*(J-1)^2
C
C  where K1, K2, K3, m, p, q are distortional fitting parameters and D1
C  is a volumetric fitting parameter. I1 is the first invariant of the
C  modified stretch tensor. J is the volumetric ratio.
C
C **********************************************************************
          SUBROUTINE UHYPER(BI1,BI2,AJ,U,UI1,UI2,UI3,TEMP,NOEL,CMNAME,
         1                  INCMPFLAG,NUMSTATEV,STATEV,NUMFIELDV,
         2                  FIELDV,FIELDVINC,NUMPROPS,PROPS)
    C
          INCLUDE 'ABA_PARAM.INC'
    C
          CHARACTER*80 CMNAME
          DIMENSION U(2),UI1(3),UI2(6),UI3(6),STATEV(*),FIELDV(*),
         1          FIELDVINC(*),PROPS(*)
    C
    C     PARAMETERS
```

```
C     ----------
      REAL*8     zero,      one,      two,      three
      PARAMETER( zero=0.d0, one=1.d0, two=2.d0, three=3.d0 )
C
C     LOCAL VARIABLES
C     ---------------
      REAL*8 k1, k2, k3, em, pe, qu, d1
C
C **********************************************************************
C -------------------------- MODEL PARAMETERS --------------------------
C **********************************************************************
      k1 =  5.38
      k2 = -2.85
      k3 =  0.4
      em =  0.89
      pe =  1.08
      qu =  1.85
      d1 =  0.00186495
C
C **********************************************************************
C ----------------------- STRAIN-ENERGY FUNCTION -----------------------
C **********************************************************************
      IF (d1.GT.zero) THEN
         U(1) =   k1*(BI1-three)**em +
     $            k2*(BI1-three)**pe +
     $            k3*(BI1-three)**qu +
     $            1/d1*(AJ-one)**two
      ELSE
         U(1) =   k1*(BI1-three)**em +
     $            k2*(BI1-three)**pe +
     $            k3*(BI1-three)**qu
      END IF
C
```

```
C **********************************************************************
C -- IF-THEN statement initializes dU/dI1 to finite value if EM < 1.0 --
C **********************************************************************
      IF (BI1.EQ.three .AND. em.LT.one) THEN
        UI1(1) = one
      ELSE
        UI1(1) = em*k1*(BI1-three)**(em-one) +
     $           pe*k2*(BI1-three)**(pe-one) +
     $           qu*k3*(BI1-three)**(qu-one)
      END IF
C
C **********************************************************************
C ---------- IF-THEN statement prevents d2U/dI1 --> Infinity -----------
C **********************************************************************
      IF (BI1.EQ.three .AND.
     $   (em.LT.two .OR. pe.LT.two .OR. qu.LT.two)) THEN
        UI2(1) = zero
      ELSE
        UI2(1) = em*(em-one)*k1*(BI1-three)**(em-two) +
     $           pe*(pe-one)*k2*(BI1-three)**(pe-two) +
     $           qu*(qu-one)*k3*(BI1-three)**(qu-two)
      END IF
C
C **********************************************************************
C ------------------ DERIVATIVES OF COMPRESSIBLE TERM ------------------
C **********************************************************************
      IF (d1.GT.zero) THEN
         UI1(3) = two/d1*(AJ-one)
         UI2(3) = two/d1
      ELSE
         UI1(3) = zero
         UI2(3) = zero
      END IF
```

```
C
C **********************************************************************
C ----------------- SET NON-ESSENTIAL VARIABLES TO ZERO ----------------
C **********************************************************************
      U(2)   = zero
      UI1(2) = zero
      UI2(2) = zero
      UI2(4) = zero
      UI2(5) = zero
      UI2(6) = zero
      UI3(1) = zero
      UI3(2) = zero
      UI3(3) = zero
      UI3(4) = zero
      UI3(5) = zero
      UI3(6) = zero
C
      RETURN
C
  END SUBROUTINE UHYPER
```

APPENDIX C: IMPLEMENTING THE GEN-YEOH STRAIN ENERGY FUNCTION IN ABAQUS/EXPLICIT WITH A VUMAT SUBROUTINE

VUMATs have been implemented by many researchers, but detailed codes are rarely published.[38,39] Some neo-Hookean VUMATs can be found. Bergström provides a partially complete but obsolete example,[33] and Chester gives a partial code that, while correct, is not computationally optimal.[40] Simulia Corp. also provides some code blocks for the neo-Hookean SEF in training materials in its 3DS Academy.[41] However, finding a complete and validated code for a hyperelastic VUMAT is not easy. To close this gap, the complete code for 3D implementation of the gen-Yeoh VUMAT is provided in Appendix D, and some clarifications to understand the code are in this section.

Guidelines on writing a VUMAT can be found in Abaqus documentation.[37] The Cauchy stress tensor, stretch tensor, and strain increment tensor are stored in vectors with six components. For example, Cauchy stress components are as follows:

$$\boldsymbol{\sigma} = \begin{bmatrix} \sigma_{11} & \sigma_{12} & \sigma_{13} \\ \sigma_{21} & \sigma_{22} & \sigma_{23} \\ \sigma_{31} & \sigma_{32} & \sigma_{33} \end{bmatrix} = (\sigma_{11}, \sigma_{22}, \sigma_{33}, \sigma_{12}, \sigma_{23}, \sigma_{13}) = (\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5, \sigma_6) \tag{37}$$

where the symmetry of $\sigma_{ij} = \sigma_{ji}$ has been invoked to reduce storage space.

When a model uses more than one element, Abaqus may process multiple material points during a VUMAT call. These are stored in an Abaqus-defined parameter, nblock, that increases the dimension of the stress, stretch, and strain increment vectors.

Explicit analyses require calculation of a stable time increment to advance the solution. The increment is initialized with a linearly elastic approximation for the material during the first call of the VUMAT. This is completed by using Abaqus-defined strain increments ($d\varepsilon_i$), the initial stress vector (σ'_i), and elastic material parameters (G, κ) to calculate the new stress vector (σ''_i) as shown in the following logic:

```
IF(totalTime = 0)THEN

  DO k = 1, nblock

    $\sigma''_{k,1} = \sigma'_{k,1} + 2G(d\varepsilon_{k,1}) + \left(\kappa - \frac{2}{3}G\right)(\varepsilon_{k,1} + \varepsilon_{k,2} + \varepsilon_{k,3})$

    $\sigma''_{k,2} = \sigma'_{k,2} + 2G(d\varepsilon_{k,2}) + \left(\kappa - \frac{2}{3}G\right)(\varepsilon_{k,1} + \varepsilon_{k,2} + \varepsilon_{k,3})$

    $\sigma''_{k,3} = \sigma'_{k,3} + 2G(d\varepsilon_{k,3}) + \left(\kappa - \frac{2}{3}G\right)(\varepsilon_{k,1} + \varepsilon_{k,2} + \varepsilon_{k,3})$

    $\sigma''_{k,4} = \sigma'_{k,4} + 2G(d\varepsilon_{k,4})$

    $\sigma''_{k,5} = \sigma'_{k,5} + 2G(d\varepsilon_{k,5})$

    $\sigma''_{k,6} = \sigma'_{k,6} + 2G(d\varepsilon_{k,6})$

  END DO
  RETURN
END IF
```

Next, the hyperelastic block of the code executes with the steps at the end of the section on finite elasticity. In some cases, the linear elastic step does not sufficiently deform elements and stress indeterminacy occurs in the hyperelastic coding block due to numerical precision or truncation. To avoid this, the following logic can be applied:

```
IF$\left((\bar{I}_1 - 3) < 10^{-12}\right)$THEN

  $\frac{\partial W}{\partial \bar{I}_1} = 0$

ELSE

  $\frac{\partial W}{\partial \bar{I}_1} = mK_1(\bar{I}_1 - 3)^{m-1} + pK_2(\bar{I}_1 - 3)^{p-1} + qK_3(\bar{I}_1 - 3)^{q-1}$

END IF
```

The user must set the threshold at which $\partial W/\partial \bar{I}_1 = 0$. Larger values help convergence but introduce larger rounding error. Smaller values do the opposite.

Parameters for the hyperelastic material model can be directly specified in the subroutine, read from the input file, or read from Abaqus's .cae file. If this latter option is preferred, the material properties must be specified as shown in Figure 13.

The material properties are then read with the following statements in the subroutine:

```
C MATERIAL PROPERTIES
C ———————————————————————
k1 = props(1)
k2 = props(2)
k3 = props(3)
em = props(4)
pe = props(5)
qu = props(6)
d1 = props(7)
```

When updating the internal energy density, the strain energy function can be directly applied. Alternatively, the following equation adopted from Abaqus documentation correctly increments the energy:

$$W = \frac{1}{2}\Big\{(\sigma'_{\mathrm{k},1} + \sigma''_{\mathrm{k},1})d\varepsilon_{\mathrm{k},1} + (\sigma'_{\mathrm{k},2} + \sigma''_{\mathrm{k},2})d\varepsilon_{\mathrm{k},2} + (\sigma'_{\mathrm{k},3} + \sigma''_{\mathrm{k},3})d\varepsilon_{\mathrm{k},3} \\ +2\Big[(\sigma'_{\mathrm{k},4} + \sigma''_{\mathrm{k},4})d\varepsilon_{\mathrm{k},4} + (\sigma'_{\mathrm{k},5} + \sigma''_{\mathrm{k},5})d\varepsilon_{\mathrm{k},5} + (\sigma'_{\mathrm{k},6} + \sigma''_{\mathrm{k},6})d\varepsilon_{\mathrm{k},6}\Big]\Big\}$$

Finally, plane strain and axisymmetric versions of the code can be built by removing any references to the fifth and sixth elements of the stress, stretch, and strain increment vectors. Alternatively, one can follow Bergström's example and implement logic to handle two-dimensional and 3D cases with a single subroutine.[33] A plane stress subroutine requires additional modification to account for out-of-plane strains.

APPENDIX D: GEN-YEOH VUMAT SUBROUTINE

Disclaimer: Abaqus is sensitive to the location of the column in which lines start in subroutine files, and the code shown does not reflect proper alignment. If problems are encountered during compilation, confirm appropriate spacing with the templates and validated subroutines provided in Abaqus's documentation.

```
C **********************************************************************
C  VUMAT for gen-Yeoh Strain-Energy Function
C  -----------------------------------------
C  Authors:  Travis Hohenberger & Richard Windslow
C  Date:     2019-07-11
C  E-mail:   twhohen@gmail.com
C **********************************************************************
C
C  Strain-energy function:
C
C  W = K1*(I1-3)^m + K2*(I1-3)^p + K3*(I1-3)^q + (1/D1)*(J-1)^2
C
C  where K1, K2, K3, m, p, q are distortional fitting parameters and D1
C  is a volumetric fitting parameter. I1 is the first invariant of the
C  modified stretch tensor. J is the volumetric ratio.
C
C **********************************************************************
          SUBROUTINE VUMAT(
     1     nblock, ndir, nshr, nstatev, nfieldv, nprops, lanneal,
     2     stepTime, totalTime, dt, cmname, coordMp, charLength,
     3     props, density, strainInc, relSpinInc,
     4     tempOld, stretchOld, defgradOld, fieldOld,
     5     stressOld, stateOld, enerInternOld, enerInelasOld,
     6     tempNew, stretchNew, defgradNew, fieldNew,
     7     stressNew, stateNew, enerInternNew, enerInelasNew )
C
      INCLUDE 'vaba_param.inc'
C
      DIMENSION props(nprops), density(nblock), coordMp(nblock,*),
     1          charLength(nblock), strainInc(nblock,ndir+nshr),
     2          relSpinInc(nblock,nshr), tempOld(nblock),
     3          stretchOld(nblock,ndir+nshr),
     4          defgradOld(nblock,ndir+nshr),
     5          fieldOld(nblock,nfieldv), stressOld(nblock,ndir+nshr),
     6          stateOld(nblock,nstatev), enerInternOld(nblock),
     7          enerInelasOld(nblock), tempNew(nblock),
     8          stretchNew(nblock,ndir+nshr),
     9          defgradNew(nblock,ndir+nshr),
     1          fieldNew(nblock,nfieldv),
     2          stressNew(nblock,ndir+nshr), stateNew(nblock,nstatev),
     3          enerInternNew(nblock), enerInelasNew(nblock)
C
      CHARACTER*80 cmname
C
C     PARAMETERS
C     ----------
      REAL*8    oneThrd, half, twoThrd, one, two, three, thresh
      PARAMETER(oneThrd=1.d0/3.d0, half=0.5d0, twoThrd=2.d0/3.d0,
     $          one=1.d0, two=2.d0, three=3.d0, thresh=10.d0**-12.d0)
C
C     LOCAL VARIABLES
C     ---------------
      REAL*8 g0      , k0     , twoG   , lmda   , trace  , d1      ,
     $       k1      , k2     , k3     , em     , pe     , qu      ,
     $       Bxx     , Byy    , Bzz    , Bxy    , Bxz    , Byz     ,
     $       BbarXX  , BbarYY , BbarZZ , BbarXY , BbarXZ , BbarYZ  ,
     $       dBbarXX , dBbarYY, dBbarZZ, dBbarXY, dBbarXZ, dBbarYZ ,
     $       J       , J23    , duDi1  , duDi3  , I1     , g1      ,
     $       p0      , j1     , u1
C
      k1 = props(1)
      k2 = props(2)
      k3 = props(3)
      em = props(4)
      pe = props(5)
      qu = props(6)
      d1 = props(7)
C
      g0 = two * k1
      k0 = two / d1
C
C     *******************************************************************
C     ------------ INITIALIZE MATERIAL AS LINEARLY ELASTIC --------------
C     *******************************************************************
C
      twoG = two * g0
      lmda = k0 - twoG * oneThrd
C
      IF (totalTime.EQ.0.0) THEN
C
         DO k = 1,nblock
            trace = strainInc(k,1) + strainInc(k,2) + strainInc(k,3)
            stressNew(k,1) = stressOld(k,1) + twoG*strainInc(k,1) +
     $                       lmda*trace
            stressNew(k,2) = stressOld(k,2) + twoG*strainInc(k,2) +
     $                       lmda*trace
            stressNew(k,3) = stressOld(k,3) + twoG*strainInc(k,3) +
     $                       lmda*trace
            stressNew(k,4) = stressOld(k,4) + twoG*strainInc(k,4)
            stressNew(k,5) = stressOld(k,5) + twoG*strainInc(k,5)
            stressNew(k,6) = stressOld(k,6) + twoG*strainInc(k,6)
         END DO
C
         RETURN
C
      END IF
C
C     ******************************************************************
C     ----------- START LOOP FOR MATERIAL POINT CALCULATIONS -----------
C     ******************************************************************
C
      DO k = 1,nblock
C
C        CALCULATE LEFT CAUCHY-GREEN STRAIN TENSOR, B = U*U
C        --------------------------------------------------
         Bxx = stretchNew(k,1) * stretchNew(k,1) +
     $         stretchNew(k,4) * stretchNew(k,4) +
     $         stretchNew(k,6) * stretchNew(k,6)
         Byy = stretchNew(k,2) * stretchNew(k,2) +
     $         stretchNew(k,4) * stretchNew(k,4) +
     $         stretchNew(k,5) * stretchNew(k,5)
         Bzz = stretchNew(k,3) * stretchNew(k,3) +
     $         stretchNew(k,5) * stretchNew(k,5) +
     $         stretchNew(k,6) * stretchNew(k,6)
         Bxy = stretchNew(k,1) * stretchNew(k,4) +
     $         stretchNew(k,4) * stretchNew(k,2) +
     $         stretchNew(k,6) * stretchNew(k,5)
         Bxz = stretchNew(k,1) * stretchNew(k,6) +
     $         stretchNew(k,4) * stretchNew(k,5) +
     $         stretchNew(k,6) * stretchNew(k,3)
         Byz = stretchNew(k,4) * stretchNew(k,6) +
     $         stretchNew(k,2) * stretchNew(k,5) +
     $         stretchNew(k,5) * stretchNew(k,3)
```

```
C
C        CALCULATE J = |F| = |U|
C        -----------------------
C
         J =    stretchNew(k,1) *
     $        ( stretchNew(k,2) * stretchNew(k,3)   -
     $          stretchNew(k,5) * stretchNew(k,5) ) +
     $          stretchNew(k,4) *
     $        ( stretchNew(k,5) * stretchNew(k,6)   -
     $          stretchNew(k,3) * stretchNew(k,4) ) +
     $          stretchNew(k,6) *
     $        ( stretchNew(k,4) * stretchNew(k,5)   -
     $          stretchNew(k,2) * stretchNew(k,6) )
C
C        CALCULATE MODIFIED STRAIN TENSOR, Bbar = J^(-2/3)*B
C        ---------------------------------------------------
         J23 = J**(-twoThrd)
C
         BbarXX = J23 * Bxx
         BbarYY = J23 * Byy
         BbarZZ = J23 * Bzz
         BbarXY = J23 * Bxy
         BbarXZ = J23 * Bxz
         BbarYZ = J23 * Byz
C
C        FIRST INVARIANT OF Bbar = tr(Bbar)
C        ----------------------------------
         I1 = BbarXX + BbarYY + BbarZZ
C
C        DEVIATORIC PART OF Bbar
C        -----------------------
         p0 = oneThrd * I1
C
```

```
         dBbarXX = BbarXX - p0
         dBbarYY = BbarYY - p0
         dBbarZZ = BbarZZ - p0
         dBbarXY = BbarXY
         dBbarXZ = BbarXZ
         dBbarYZ = BbarYZ
C
C        DERIVATIVES OF STRAIN-ENERGY FUNCTION
C        -------------------------------------
         j1 = I1 - three
C
         IF (j1.LT.thresh) THEN
            duDi1 = zero
         ELSE
            duDi1 = em * k1 * j1**(em-one) +
     $              pe * k2 * j1**(pe-one) +
     $              qu * k3 * j1**(qu-one)
         END IF
C
         duDi3 = two/d1 * (J - one)
C
C        COROTATIONAL CAUCHY (TRUE) STRESSES
C        -----------------------------------
         g1 = two/J * duDi1
C
         stressNew(k,1) = g1 * dBbarXX + duDi3
         stressNew(k,2) = g1 * dBbarYY + duDi3
         stressNew(k,3) = g1 * dBbarZZ + duDi3
         stressNew(k,4) = g1 * dBbarXY
         stressNew(k,5) = g1 * dBbarYZ
         stressNew(k,6) = g1 * dBbarXZ
C
C        UPDATE SPECIFIC INTERNAL ENERGY
```

```
C        -------------------------------
         u1 = half * ( (stressOld(k,1)+stressNew(k,1))*strainInc(k,1) +
     $                 (stressOld(k,2)+stressNew(k,2))*strainInc(k,2) +
     $                 (stressOld(k,3)+stressNew(k,3))*strainInc(k,3) +
     $                  two * ( (stressOld(k,4) + stressNew(k,4))*
     $                           strainInc(k,4) +
     $                          (stressOld(k,5) + stressNew(k,5))*
     $                           strainInc(k,5) +
     $                          (stressOld(k,6) + stressNew(k,6))*
     $                           strainInc(k,6) ) )
C
         enerInternNew(k) = enerInternOld(k) + u1 / density(k)
C
      END DO
C
      RETURN
C
      END SUBROUTINE VUMAT
```