Research paper

UMAT4COMSOL: An Abaqus user material (UMAT) subroutine wrapper for COMSOL

Sergio Lucarini [a,b,c], Emilio Martínez-Pañeda [d,a,*]

[a] *Department of Civil and Environmental Engineering, Imperial College, London SW7 2AZ, UK*
[b] *BCMaterials, Basque Center for Materials, Applications and Nanostructures, Leioa 48940, Spain*
[c] *Ikerbasque, Basque Foundation for Science, Bilbao 48009, Spain*
[d] *Department of Engineering Science, University of Oxford, Oxford OX1 3PJ, UK*

ARTICLE INFO

ABSTRACT

We present a wrapper that allows Abaqus user material subroutines (*UMAT*s) to be used as an *External Material* library in the software COMSOL Multiphysics. The wrapper, written in C language, transforms COMSOL's external material subroutine inputs and outputs into Fortran-coded Abaqus *UMAT* inputs and outputs, by means of a consistent variable transformation. This significantly facilitates conducting coupled, multi-physics studies employing the advanced material models that the solid mechanics community has developed over the past decades. We exemplify the potential of our new framework, UMAT4COMSOL, by conducting numerical experiments in the areas of elastoplasticity, hyperelasticity and crystal plasticity. The source code, detailed documentation and example tutorials are made freely available to download at www.empaneda.com/codes.

1. Introduction

Coupling multiple physical phenomena in continuum solids is a major research focus [1,2]. The mechanical behaviour of structures and industrial components is relatively well understood in inert environments but challenges arise as a result of material-environment interactions. Multi-physics structural integrity problems such as hydrogen embrittlement [3,4], corrosion [5,6], and oxidation-assisted fatigue [7, 8] continue to challenge scientists and engineers. Predicting these structural integrity problems requires coupling state-of-the-art material models with equations describing chemical, electrical and thermal phenomena [9–13]. In addition, numerous modern devices and concepts involve multi-physics environments and mechanical loads, which are either applied externally or induced by other physical phenomena, such as thermal expansion, chemical strains or magnetic forces. Examples include Li-Ion batteries [14,15], hydrogels [16], magnetorheological elastomers [17,18], and piezo-electric and piezo-resistive materials [19, 20], to name a few. As a result, there is a pressing need to develop computational tools for characterising the coupled behaviour of materials in multiphysics environments [21–23].

Abaqus [24] and COMSOL [25] have arguably been the most popular finite element packages for materials modelling and multi-physics simulations, respectively. For decades, the solid mechanics community

has developed Abaqus user material subroutines (*UMAT*s) to numerically implement new and advanced material models; e.g., in the context of hyperelasticity [26], elastoplasticity [27], damage [28], and crystal plasticity [29,30]. Abaqus *UMAT* subroutines are Fortran codes that describe constitutive material behaviour under either the small strains or finite strains convention. On the other hand, COMSOL stands out for its ability to handle coupled systems of partial differential equations, with multiple built-in physics modules that encompass most physical phenomena across the disciplines of chemistry, fluid flow, heat transfer, electromagnetism, structural mechanics and acoustics. Given the growing interest in material problems in multi-physics environments, there is great benefit in coupling these two tools. However, the COMSOL materials library does not currently support the automatic use of Abaqus *UMAT* subroutines as material models. Available materials in COMSOL are either built-in or can be programmed in C language as a user-defined *External Material* library. Therefore, our aim is to develop and share a wrapper that can take advantage of preprogrammed Abaqus *UMAT* material models and enable their use as *External Material* libraries in COMSOL. Both small and finite strains are considered, providing a general framework that can leverage the notable efforts made in materials model development with Abaqus

---

* Corresponding author at: Department of Engineering Science, University of Oxford, Oxford OX1 3PJ, UK.
E-mail addresses: s.lucarini@imperial.ac.uk (S. Lucarini), emilio.martinez-paneda@eng.ox.ac.uk (E. Martínez-Pañeda).

*UMAT*s, while also taking advantage of the versatility of COMSOL for multi-physics simulations. In this way, the present work contributes to ongoing efforts in the community aimed at connecting computational platforms [31–35].

The remainder of this paper is organised as follows. First, in Section 2, we proceed to describe the characteristics of the software that we have developed, UMAT4COMSOL. Then, in Section 3, usage instructions are provided. The potential of UMAT4COMSOL is demonstrated in Section 4 through examples encompassing small strain elastoplasticity, large strain hyperelasticity and crystal plasticity. Finally, concluding remarks end the manuscript in Section 5.

2. Software description: UMAT4COMSOL

UMAT4COMSOL is a C-coded subroutine that functions as a user-defined *External Material* in COMSOL, so as to describe the local mechanical behaviour of solid domains. The subroutine contains a call to an external Fortran-coded subroutine that corresponds to the Abaqus *UMAT*. During this process, the local state is inputted, and the constitutive equations describing the material behaviour are computed, with their outputs being returned to COMSOL for solving the multiphysics global problem.

UMAT4COMSOL's primary internal features are described below, starting with a concise explanation of the input requirements for both COMSOL *External Materials* and Abaqus *UMAT*s. Subsequently, the process of transferring input parameters from COMSOL *External Materials* to Abaqus *UMAT*s is elaborated upon. Finally, the output procedure, which involves a more detailed understanding of the various mathematical methods, is described.

2.1. Material models inputs and outputs

When modelling the mechanical behaviour of materials, the constitutive equations characterise the relation between the measure of the deformation and the stress state of the material. These equations can be linear, non-linear, time-dependent, and history-dependent.

The input to the material constitutive equations is typically a deformation measure. In the small strain formalism, the strain tensor ε_{ij} is used as the deformation measure, and is defined in terms of the displacement vector u_i as

$$\varepsilon_{ij} = \frac{1}{2}\left(u_{i,j} + u_{j,i}\right),\tag{1}$$

while the deformation gradient F_{ij}, given by

$$F_{ij} = \delta_{ij} + u_{i,j},\tag{2}$$

is typically adopted when considering finite strains. Here, δ_{ij} denotes the Dirac delta function and the derivative definition in Eq. (2) is defined in the reference configuration. Another class of inputs are the material properties, which are typically constant values. Additionally, the constitutive equations can be posed as a function of other strain measures dependent on the aforementioned ones, such as the left Cauchy–Green or Green–Lagrange tensors. In the case of history-dependent material behaviour, one should also consider a state variable vector α_i, that stores history information of the state of the material. Finally, in time-dependent problems, the time increment value is also stored, so as to calculate the strain rate measures and the evolution of state variables.

The outcome of the constitutive calculations is a calculated stress measure, which will be translated to forces at the element level. In the case of non-linear analyses, the mechanical equilibrium equation is usually linearised, requiring the definition of a consistent tangent modulus, which corresponds to the derivative of the adopted stress measure with respect to the adopted strain measure.

2.2. COMSOL external material

COMSOL *External Materials* are C-coded subroutines that are calculated at the Gauss point level on each iteration using a subroutine called `eval`. The inputs include the strain/deformation gradient trials `E`/`Fl`, along with the properties vector `par` and the deformation gradient at the previous time increment `FlOld`. The (pseudo)time steps/increments are defined in the `delta` variable, and a set of state variables are passed as a vector `state`. Once the constitutive equations are calculated in the core code, the results obtained for each (pseudo)time step/increment are passed using the stress vector `S`, the consistent tangent `D`/`Jac`, and the evolved state variables `state`.

2.3. Abaqus user material (UMAT) subroutine

Abaqus *UMAT*s are Fortran-coded subroutines that calculate the constitutive equations at the Gauss point level on every iteration. The inputs for the mechanical problem include the trial incremental quantities of strain/deformation trials at the current increment (`DSTRAN`/`DFGRD1`), the total strain/deformation measures at the end of the previous increment (`STRAN`/`DFGRD0`), the stresses at the previous time increment `STRESS`, and the material properties, which are contained in the `PROPS` vector. The (pseudo)time increments are defined in the `DTIME` variable, together with the current (pseudo)time value (`TIME`), and a set of state variables is passed as a vector in `STATEV`. Also, some constitutive relations in a finite strain context need the incremental rotation tensor variable `DROT`. After calculating the constitutive equations, the key outputs at the end of the (pseudo)time increment are the updated stress vector `STRESS`, the consistent tangent matrix `DDSDDE`, and the updated state variables vector `STATEV`.

2.4. Transfer of inputs

The code initially transforms COMSOL's input variables to conform to the convention of Abaqus *UMAT* subroutines, which requires a change in the strain format. Specifically, two aspects must be taken care of in small strain problems. One is that Abaqus stores shear strains as engineering shear strains; $\gamma_{xy} = 2\varepsilon_{xy}$. A second one is that the ordering of the shear components is different. Accordingly, one must change the COMSOL input strain tensor (`E` variable) to fit the *UMAT* convention, such that

$$\varepsilon_{xx}, \varepsilon_{yy}, \varepsilon_{zz}, \varepsilon_{yz}, \varepsilon_{xz}, \varepsilon_{xy} \rightarrow \varepsilon_{xx}, \varepsilon_{yy}, \varepsilon_{zz}, \gamma_{xy}, \gamma_{xz}, \gamma_{yz}\tag{3}$$

where $\gamma_{\square} = 2\varepsilon_{\square}$ and the left-hand side corresponds to the COMSOL notation. Here, one should note that COMSOL's E vector contains the total trial strains. Consequently, the incremental trial strains vector, `DSTRAN`, is computed by subtracting from E the total strain at the end of the previous increment, which has been stored as state variables and corresponds to the variable `STRAN` in the *UMAT* convention. Also, the stress at the previous time increment is saved in the state variable vector and assigned to the *UMAT* variable `STRESS` before computing the constitutive equations. For the finite strains version, the *UMAT* requires the deformation gradient F_{ij} before and after the (pseudo)time increment. COMSOL provides the deformation gradients `FlOld` and `Fl` to the external material subroutine as a C-order matrix. These matrices are transformed into a Fortran-order matrix and passed to the *UMAT* as `DFGRD0` and `DFGRD1`. Additionally, the incremental strain `DSTRAN` ($\Delta\varepsilon_{ij}$) and incremental rotation `DROT` ($R_{ij}^{\Delta t}$) are defined as auxiliary input variables. The `DSTRAN` components are computed via

$$\Delta\varepsilon_{ij} = 1/2\left(\Delta F_{ip}F_{t+\Delta t \mid pj}^{-1} + F_{t+\Delta t \mid pi}^{-1}\Delta F_{jp}\right),\tag{4}$$

defining $\Delta F_{ij} = F_{t+\Delta t \mid ij} - F_{t \mid ij}$ and forming a vector by using Eq. (3). The variable `DROT` is defined by

$$F_{ij}^{\Delta t} = R_{ip}^{\Delta t}U_{pj}^{\Delta t},\tag{5}$$

standing for the polar decomposition of $F_{ij}^{\Delta t} = F_{t+\Delta t} \mid_{ip} F_{t}^{-1}\mid_{pj}$. Finally, the values of the current (pseudo)time and (pseudo)time increment are obtained by making use of COMSOL's `delta` variable and updating a state variable. The COMSOL variable `delta` contains the current time increment and thus corresponds to the *UMAT* variable DTIME. Since the *UMAT* subroutine also takes as input the total time at the beginning of the increment (TIME variable), this is stored as the first component of the STATEV vector, which is updated on every increment based on `delta`.

2.5. Transfer of outputs

The stress vector and the consistent tangent matrix are the outputs of the *UMAT*s. In the small strain version, the stress variable (STRESS) can be directly transformed by rearranging stress vector shear components, as in Eq. (3), to match the required output format for the COMSOL solver (S variable). In a similar fashion, the consistent tangent matrix DDSDDE is reordered to match the arrangement of the shear components, as per (Eq. (3)), and then transposed to convert from a Fortran-ordered matrix to a C-ordered matrix D.

For the finite strain case, the outputs of the *UMAT* subroutine (the stresses and the consistent tangent matrix) need to be transformed before being transferred. In regards to the stresses, Abaqus *UMAT* outputs the Cauchy stress tensor σ_{ij} in the variable STRESS (as a vector, using Voigt notation), while COMSOL works with the second Piola–Kirchhoff stress S_{ij}, using the variable S. The transformation, STRESS → S, is defined using the following relation

$$S_{ij} = J F_{iq}^{-1} \sigma_{qp} F_{jp}^{-T},\tag{6}$$

where $J = \det\left(F_{ij}\right)$.

As described below, the transformation of the consistent tangent tensor (DDSDDE → Jac) in a finite strains context requires careful consideration. This is also a key step to guarantee the appropriate convergence rate. The material Jacobian obtained from the Abaqus *UMAT* user subroutine at finite strains is the tangent modulus tensor for the Jaumann rate of the Kirchhoff stress, denoted by C_{ijkl}^{Abaqus}, which is a fourth-order tensor defined as

$$\overset{\triangledown}{\tau}_{ij} = \dot{\tau}_{ij} - w_{ip}\tau_{pj} - \tau_{ip}w_{jp} = J\,C_{ijkl}^{Abaqus}d_{kl}\tag{7}$$

where τ_{ij} denotes the Kirchhoff stress, $\triangledown$ is the Jaumann rate, and w_{ij} and d_{ij} are the spin tensor and stretch tensor, respectively. In contrast, the COMSOL framework requires the material tangent, K_{ijkl}^{COMSOL}, which is given by

$$\dot{S}_{ij} = K_{ijkl}^{COMSOL}\dot{F}_{kl}.\tag{8}$$

We shall then derive an explicit expression that relates the two fourth-order tensors, C_{ijkl}^{Abaqus} and K_{ijkl}^{COMSOL}. Combining Eqs. (6) and (8), one reaches

$$K_{ijkl}^{COMSOL} = \frac{\partial S_{ij}}{\partial F_{kl}} = \frac{\partial\left(J F_{iq}^{-1}\sigma_{qp}F_{jp}^{-1}\right)}{\partial F_{kl}} = \frac{\partial\left(F_{iq}^{-1}\tau_{qp}F_{jp}^{-1}\right)}{\partial F_{kl}}.\tag{9}$$

Expanding the derivative of the product in Eq. (9) and expressing it in index notation, we obtain:

$$K_{ijkl}^{COMSOL} = \frac{\partial S_{ij}}{\partial F_{kl}} = \frac{\partial F_{iq}^{-1}}{\partial F_{kl}}\tau_{qp}F_{jp}^{-1} + F_{iq}^{-1}\frac{\partial\tau_{qp}}{\partial F_{kl}}F_{jp}^{-1} + F_{iq}^{-1}\tau_{qp}\frac{\partial F_{jp}^{-1}}{\partial F_{kl}}\tag{10}$$

This equation expresses the components of the COMSOL material tangent, K_{ijkl}^{COMSOL}, as a function of components that can be obtained using the definition of the Abaqus tangent modulus tensor C_{ijkl}^{Abaqus}, see Eq. (7). The first and third terms in Eq. (10) are the derivatives of the inverse of a tensor with respect to itself: $\partial F_{iq}^{-1}/\partial F_{kl} = -F_{ik}^{-1}F_{lq}^{-1}$. To obtain the second term in Eq. (10), we linearise the rate quantities from Eq. (7) by multiplying by an infinitesimal time increment and reordering terms, yielding

$$\delta\tau_{ij} = J\,C_{ijkl}^{Abaqus}\delta d_{kl} + \delta w_{ip}\tau_{pj} + \tau_{ip}\delta w_{jp},\tag{11}$$

where δd_{ij} and δw_{ij} are obtained as functions of F_{ij} and δF_{ij} via

$$\delta d_{ij} = \frac{1}{2}\delta F_{ip}F_{pj}^{-1} + \frac{1}{2}\delta F_{jp}F_{pi}^{-1}, \delta w_{ij} = \frac{1}{2}\delta F_{ip}F_{pj}^{-1} - \frac{1}{2}\delta F_{jp}F_{pi}^{-1}.\tag{12}$$

The Kirchhoff stress is also linearised with respect to the perturbation of the deformation gradient as

$$\delta\tau_{ip} = \frac{\partial\tau_{ip}}{\partial F_{kl}}\delta F_{kl}.\tag{13}$$

where $\delta\tau_{ij}$ and δF_{ij} are the Kirchhoff stress and the deformation gradient perturbations, respectively. Inserting the expressions of Eq. (12) into Eq. (11), we obtain

$$\begin{aligned}\delta\tau_{ip} = {} & \frac{J}{2}C_{ipkm}^{Abaqus}\delta F_{kl}F_{lm}^{-1} + \frac{J}{2}C_{ipmk}^{Abaqus}F_{lm}^{-1}\delta F_{kl}\\& + \frac{1}{2}I_{ipkq}\delta F_{kl}F_{lm}^{-1}\tau_{mq} - \frac{1}{2}I_{ipmq}F_{lm}^{-1}\delta F_{kl}\tau_{kq}\\& + \frac{1}{2}I_{ipqk}\tau_{qm}F_{lm}^{-1}\delta F_{kl} - \frac{1}{2}I_{ipqm}\tau_{qk}F_{lm}^{-1}\delta F_{kl}\end{aligned}\tag{14}$$

which we can be used to obtain $\partial\tau_{ip}/\partial F_{kl}$, considering Eq. (13). By considering the minor symmetries of C_{ijkl}^{Abaqus} and the symmetry of τ_{ij}, we can simplify the expression and arrive at the following definition of $\partial\tau_{ip}/\partial F_{kl}$

$$\frac{\partial\tau_{ip}}{\partial F_{kl}} = J\,C_{ipkm}^{Abaqus}F_{lm}^{-1} + \frac{1}{2}\delta_{ik}F_{lm}^{-1}\tau_{mp} - \frac{1}{2}F_{li}^{-1}\tau_{kp} + \frac{1}{2}\delta_{pk}F_{lm}^{-1}\tau_{im} - \frac{1}{2}F_{lp}^{-1}\tau_{ik}.\tag{15}$$

The relation between the COMSOL and Abaqus material tangents is then obtained by inserting Eq. (15) into Eq. (10), rendering

$$\begin{aligned}K_{ijkl}^{COMSOL} = {} & -F_{lq}^{-1}F_{ik}^{-1}\tau_{qp}F_{jp}^{-1} + F_{iq}^{-1}\Big(J\,C_{qpkm}^{Abaqus}F_{lm}^{-1} + \frac{1}{2}\delta_{qk}F_{lm}^{-1}\tau_{mp}\\& - \frac{1}{2}F_{lq}^{-1}\tau_{kp} + \frac{1}{2}\delta_{pk}F_{lm}^{-1}\tau_{qm} - \frac{1}{2}F_{lp}^{-1}\tau_{qk}\Big)F_{jp}^{-1} - F_{iq}^{-1}\tau_{qp}F_{lp}^{-1}F_{jk}^{-1}.\end{aligned}\tag{16}$$

The resulting COMSOL consistent tangent can then be transformed to Voigt notation and transposed from Fortran-order to C-order matrix, which is the required output to be used by the COMSOL solver.

3. Usage instructions

This section provides a brief description of the source code assembly, which aims to aid in understanding the software and enable personalised development. The relevant operations to use the wrapper are also described. UMAT4COMSOL is made freely available to download at https://github.com/sergiolucarini, where future forks will be allowed to enable community developments, and www.empaneda.com/codes.

3.1. Organisation of the source code

UMAT4COMSOL is a C code that defines the `eval` function required in COMSOL for user-defined *External materials*. Two versions are available: a small strain version and a finite strain version. Both versions have a similar structure. One of the key elements of the code is an external `umat` function, which is linked to the Fortran *UMAT* subroutine. The first part of the code handles this `umat` function, which is followed by the standard header for a COMSOL user-defined external material subroutine, as per the COMSOL reference manual. Then, input variables are transformed using the operations described in Section 2.4. The core code calls the Abaqus *UMAT* subroutine to compute the constitutive equations, and the last part of the code converts the output data, as explained in Section 2.5, which is then passed to the COMSOL solver.
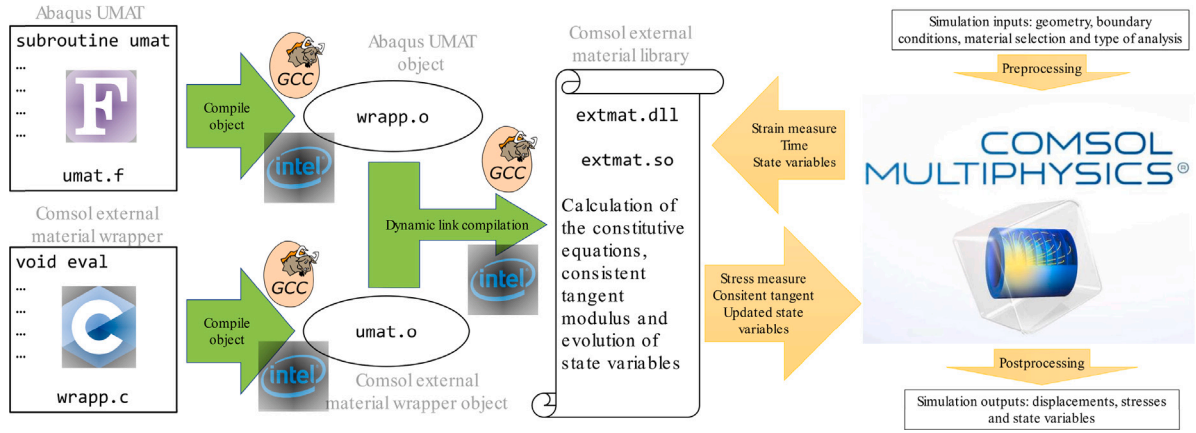
Fig. 1. Flow chart of UMAT4COMSOL.

3.2. Main usage

The main file is a wrapper in C language called `UMAT4COMSOL.c`. Once compiled, it can be used as a user-defined *External Material* library in COMSOL. For the compilation process, the `UMAT4COMSOL.c` and `umat.f` files should be in the same folder. The following commands can be used to compile the code:

```
gfortran -c umat.f
gcc -c UMAT4COMSOL.c
gcc -shared -o extmat.dll umat.o UMAT4COMSOL.o
 -lgfortran -lquadmath
```

Here, the compilation process uses open-source compilers, but other Fortran and C compilers can also be used as long as the equivalent compiling flags are used. Note that the scheme can accommodate *UMAT* subroutines written in both fixed Fortran (FORTRAN 77) and free Fortran (Fortran 90). The above commands generate two objects and a dynamic library called `extmat.dll` which is required for running simulations in COMSOL. If Linux is used, the extension of the library should be ".so" (i.e., `extmat.so`). It should also be noted that, for finite strain problems, a different wrapper is provided, named `UMAT4COMSOLfinite.c`.

To run non-linear simulations with external materials in COMSOL, one can use either the *Time-Dependent Solver* or the *Stationary Solver*. For small strains, the material should be introduced as a *General stress–strain* relation, and for finite strains, as a *General stress-deformation* relation. Then, the properties vector needs to be included in the Material model parameters vector, and the number of state variables and the initialisation values of the vector must be given. Finally, the path to the compiled dynamic library (`extmat.dll`) must be entered in the *External Material Library*'s field and imported into the COMSOL model.

To use external materials in the *Solid Mechanics* module, an *External Stress-Strain* relation node must be created for each phase in the domain and linked to the corresponding *External Material* created. After setting the boundary conditions and solver settings, the simulation is ready to be run and post-processed.

3.3. Sofware aspects

As stated previously, this wrapper relies on the definition of a COMSOL *External Material* library. A comprehensive scheme of the functioning of the framework is shown in Fig. 1. The procedure imports a compiled object, which encompasses Fortran and C code. The compiler choice is up to the user, taking into account the particularities of each compiler.

Once the compiled library is associated with the COMSOL input file, the wrapper will act as any material in COMSOL, enabling the use of the advanced functionalities that COMSOL offers. For non-linear analyses, COMSOL will access the *External Material* library at least twice in each iteration, once for obtaining the stress and a second time for obtaining the consistent tangent. This makes the COMSOL analysis less efficient compared to Abaqus since the latter is optimised to extract both at once.

4. Representative results and applications

To demonstrate the capabilities of UMAT4COMSOL, we provide representative results in the context of three areas of particular interest: elastoplasticity, large strain hyperelasticity and crystal plasticity; validating the outputs against Abaqus calculations. Thus, we first predict the behaviour of an elastoplastic plane stress holed plate undergoing small strains (Section 4.1). Then, in Section 4.2, we verify the finite strains implementation by considering a neo-Hookean hyperelastic model. Finally, in Section 4.3, we employ an advanced crystal plasticity model to showcase UMAT4COMSOL potential in handling complex constitutive equations. These are just three examples of the capabilities of UMAT4COMSOL but its applicability is universal, as it provides a non-intrusive connection between COMSOL and any material models available as Abaqus *UMAT*s. As an example of its potential, a fourth case study is included where a coupled problem is solved — the transport of hydrogen in a single crystal.

Of interest to all of these studies is the description of the tolerance criteria employed in Abaqus and COMSOL, as well as their comparison. Consider first the case of Abaqus. The tolerances for non-linear analysis in Abaqus are expressed in terms of maximum residual forces, such that equilibrium is achieved when this value is below a tolerance $\text{tol}^{\text{Abaqus}}$. The inequality reads as

$$\text{tol}^{\text{Abaqus}} > \frac{r^{max}}{\widetilde{q}}, \tag{17}$$

where r^{max} is the maximum value of the residual force vector and $\widetilde{q}$ is the overall time-averaged value of the spatially averaged force over the entire model. On the other hand, COMSOL considers that convergence has been achieved when the relative tolerance $\text{tol}^{\text{COMSOL}}$ exceeds the relative error computed as the weighted Euclidean norm. That is,

$$\text{tol}^{\text{COMSOL}} > \sqrt{\frac{1}{N} \sum_{i=1}^{N} \left(\frac{|E_i|}{\widetilde{W}} \right)^2}, \tag{18}$$

where E_i is the estimated solution error vector of the ith degree of freedom, N is the number of degrees of freedom and $\widetilde{W}$ is the weight, which is determined by $\widetilde{W} = \max\left(|U_i|, S\right)$, where $|U_i|$ is the absolute value of the solution at that specific degree of freedom and S the absolute value of the average of the solution vector. For the numerical benchmarks a non-linear tolerance of $\text{tol}^{\text{Abaqus}} = 5 \times 10^{-3}$ and $\text{tol}^{\text{COMSOL}} = 1 \times 10^{-3}$ has been set for the Abaqus and COMSOL
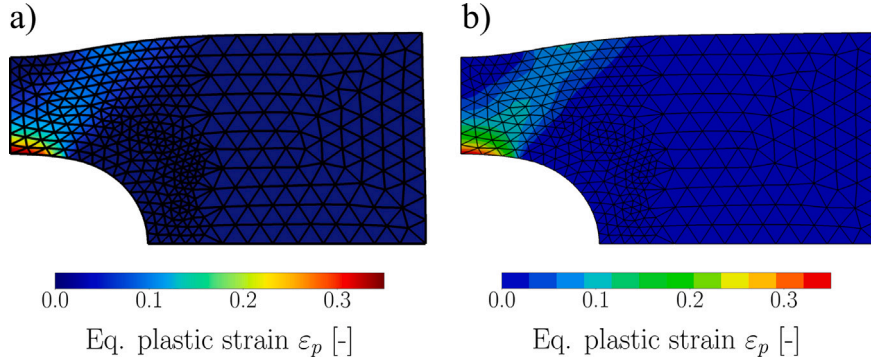
Fig. 2. Mechanical response of an elastoplastic holed plate: contour plots of the equivalent plastic strain. Results obtained: (a) with Abaqus, using an elastoplastic *UMAT*, and (b) with COMSOL, using the same *UMAT* and UMAT4COMSOL. The deformation has been scaled by a factor of 20.
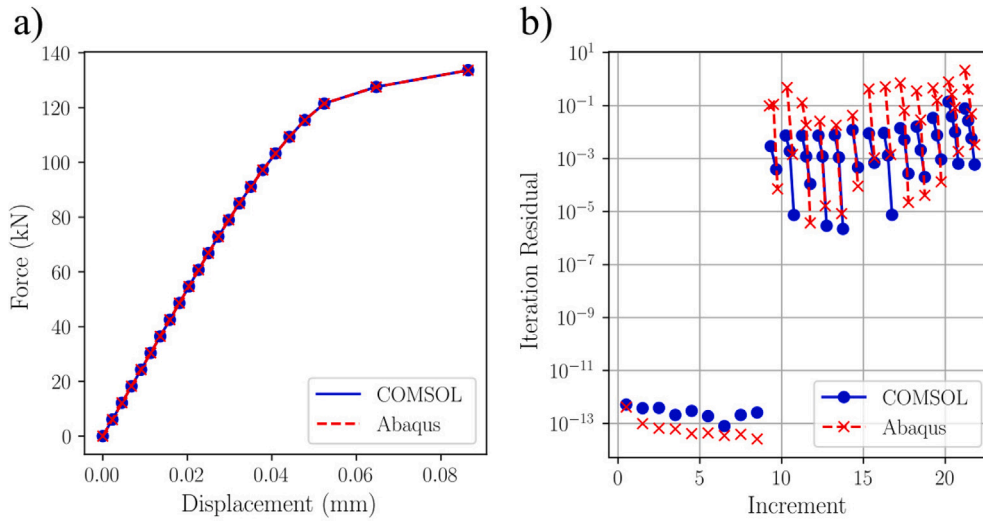

Fig. 3. Mechanical response of an elastoplastic holed plate: (a) Force versus displacement response, predicted with Abaqus (+*UMAT*) and with COMSOL (+*UMAT* and UMAT4COMSOL); and (b) convergence plots for both COMSOL and Abaqus solvers, showing the magnitude of the residual for each iteration as a function of the load increment. In (b), iterations are represented with equispaced subdivisions within each increment interval.

solvers, respectively. These are the default tolerances for both solvers. It is worth noting that while Abaqus also has a solution-based tolerance criterion, the residual-based one given in Eq. (17) is typically more restrictive.

4.1. Small strain elastoplastic model

In this example, we simulate the mechanical behaviour of a rectangular plate with a circular hole that is undergoing plastic deformation. The geometry and loading conditions are similar to the benchmark problem presented in Ref. [36]. The plate has a width of 36 mm and a height of 20 mm, and contains a hole of 5 mm radius at its centre. Due to symmetry, only one quarter of the model is simulated, with symmetry boundary conditions being applied on the left and bottom edges. The thickness of the plate is much smaller than the other dimensions, and the loads are confined to the plate plane, so plane stress conditions are assumed. The right edge of the plate is subjected to a traction that ramps linearly up to a maximum of 133.65 MPa. The plate is made of an isotropic elastoplastic material whose elastic response is characterised by a value of Young's modulus of $E = 70$ GPa and a Poisson's ratio of $\nu = 0.2$. In all case studies, we select the type of finite element consistently across platforms. In this specific benchmark, first-order plane stress Lagrange triangular elements with reduced integration are used in COMSOL, and accordingly, so-called CPS6 elements are used in the Abaqus calculations — the analogous choice. The material response is assumed to be characterised by the von

Mises plasticity, with a yield stress of $\sigma_y = 243$ MPa. Accordingly, the total strains are additively decomposed into elastic and plastic parts, $\varepsilon_{ij} = \varepsilon_{ij}^e + \varepsilon_{ij}^p$, and the Cauchy stresses are given by,

$$\sigma_{ij} = C_{ijkl}^e \varepsilon_{kl}^e \tag{19}$$

where C_{ijkl}^e is the elastic isotropic stiffness tensor. Linear isotropic hardening is assumed, with a tangent modulus of $h = 2171$ MPa. Accordingly, the yield condition is given by,

$$f\left(\sigma_{ij}\right) = \sqrt{\frac{3}{2} s_{ij} s_{ij}} - \left(\sigma_y + h\varepsilon_p\right) \tag{20}$$

where s_{ij} is the deviatoric part of the Cauchy stress, $s_{ij} = \mathrm{dev}\left(\sigma_{ij}\right)_{ij}$, and ε_p is the equivalent plastic strain, $\varepsilon_p = \int \sqrt{2/3\, \dot{\varepsilon}_{ij}^p \dot{\varepsilon}_{ij}^p}\, \mathrm{d}t$. The rate of the latter evolves as,

$$\dot{\varepsilon}_{ij}^p = \dot{\lambda} \frac{3}{2\sqrt{3}\, s_{pq} s_{pq}} s_{ij} \tag{21}$$

with $\dot{\lambda}$ being the plastic multiplier. The loading path is divided into 22 constant increments (using the *Stationary Solver* in COMSOL). The material constants are introduced as properties in the COMSOL *External Material* library and the plastic strain tensor is considered a state variable, initialised with zeros. Finite element calculations are run in both Abaqus and COMSOL using the same user material (*UMAT*) subroutine, with the COMSOL job exploiting the UMAT4COMSOL wrapper. The results obtained are given in Figs. 2 and 3.
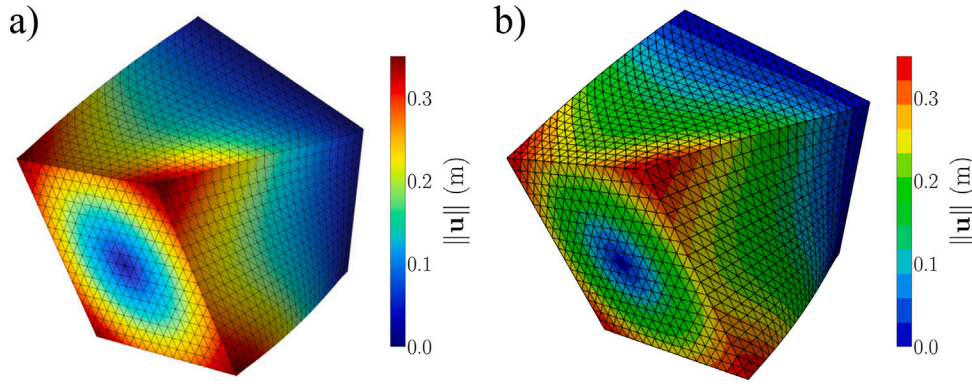
Fig. 4. Mechanical response of a twisted neo-Hookean cube: contour plots of the displacement field magnitude (L2 norm). Results obtained: (a) with Abaqus, using a non-linear hyperelastic *UMAT*, and (b) with COMSOL, using the same *UMAT* and UMAT4COMSOL.

The equivalent plastic strain contours obtained with COMSOL and Abaqus appear to be indistinguishable, see Fig. 2. The displacement solution difference between the two approaches is less than 0.6% in terms of relative L2-norm, indicating that the results obtained with Abaqus and COMSOL are identical. The perfect agreement obtained is also evident Fig. 3a, which shows the engineering stress–strain curves obtained with both packages.

The convergence rates are given in Fig. 3b. Specifically, the magnitudes of the residuals are provided for each iteration as a function of the load/time increment. As shown in Fig. 3b, after approximately ten load increments, two iterations are needed to achieve convergence in every increment. This is the same for both Abaqus and COMSOL, and the number of iterations is similar, demonstrating that both provide a similar convergence rate. Some differences are seen in the magnitude of the residuals but this should be considered with care as COMSOL uses the L2-norm of the residual vector to evaluate the convergence while Abaqus uses the ∞-norm of the out-of-balance forces.

4.2. Finite strain neo-Hookean hyperelastic model

The case study aims to verify the large strain implementation and showcase the use of UMAT4COMSOL with another material model: non-linear hyperelasticity. To this end, the behaviour resulting from twisting a three-dimensional cube made of a neo-Hookean material is studied. The characteristic length of the cube is 1 m, as in the hyperelasticity example presented in Ref. [37]. The constitutive equation of the neo-Hookean hyperelastic model is given by,

$$\sigma_{ij} = \frac{1}{J} \frac{E}{2(1+\nu)} \left(F_{iq} F_{jq} - \frac{1}{3} \delta_{ij} F_{kq} F_{kq} \right) + \frac{E}{3(1-2\nu)} (J-1) \delta_{ij} \tag{22}$$

and the material parameters used are $E = 10^6$ Pa and $\nu = 0.3$. These are introduced as properties to the COMSOL *External Material* library. The unitary cuboid is twisted by 60 degrees, and the boundary conditions are given, on one face, by Dirichlet boundary conditions of rotation with respect to the centre and, on the opposite face, by symmetry boundary conditions. For consistency, the calculations are obtained using first-order Lagrange tetrahedral elements with reduced integration in both COMSOL and Abaqus. This element type is denoted C3D4 in Abaqus. Two loading cases are studied: one with a single load increment and another one splitting the load into 10 constant load increments. Calculations are conducted using stationary solvers in both Abaqus and COMSOL.

The predicted deformed shapes of the twisted cube are given in Fig. 4, showing the contours of the displacement field magnitude. Again, no differences are observed in the results. The contours are also insensitive to the number of load increments employed. Importantly, the relative error in the displacement solution between the Abaqus-

and COMSOL-based calculations is below 0.3%, as measured by the L2-norm. Moreover, this difference does not increase when using only one pseudo-time increment.

The non-linear convergence of both solvers is comparable, as shown in Fig. 5, where the magnitude of the residual for each iteration is provided for both the 10-increment (Fig. 5a) and the 1-increment (Fig. 5b) studies. In both loading cases, COMSOL and Abaqus require a similar number of solver iterations to obtain a converged solution. As in the previous case study (Section 4.1), some quantitative differences can be observed in the magnitude of the residuals, which are intrinsically related to their definition.

4.3. Finite strain crystal plasticity

The last benchmark deals with the deformation of a metallic polycrystal where the material response is characterised using a crystal plasticity material model [38]. This benchmark has been chosen for mainly two reasons: (i) its complexity, as the crystal plasticity *UMAT* involves a large number of state variables and exhibits dependencies on the loading path and the strain rate, and (ii) its relevance, as it is not currently possible to run crystal plasticity studies in COMSOL despite its importance in many multi-physics problems. The chosen crystal plasticity material model is elastic-viscoplastic and assumes that plastic deformation only occurs by plastic shear along the slip systems. The deformation gradient is multiplicatively decomposed into elastic and plastic parts,

$$F_{ij} = F_{ip}^e F_{pj}^p \tag{23}$$

For a slip system α, the plastic slip rate ($\dot{\gamma}^\alpha$) is given in terms of the resolved shear stress (τ^α) through the following phenomenological power-law,

$$\dot{\gamma}^\alpha = \dot{\gamma}_0 \left| \frac{\tau^\alpha}{\tau_c^\alpha} \right|^n \tag{24}$$

where $\dot{\gamma}$ is the reference strain rate, and n is an exponent describing the dependence of strain rate on the stress. In Eq. (24), the critical resolved shear stress on the α slip system is the denominator τ_c^α, which evolves following the Asaro-Needleman hardening rule [39], defined by

$$\dot{\tau}_c^\alpha = \sum_\beta q_{\alpha\beta} h_0 \sec^2 \left| \frac{h_0 \sum_\delta \int_0^t |\dot{\gamma}^\delta| \, \mathrm{d}t}{\tau_s - \tau_0} \right| \dot{\gamma}^\beta \tag{25}$$

Accordingly, the plastic part evolves following the sum of plastic slips as:

$$\dot{F}_{ij}^p = \sum_\alpha \dot{\gamma}^\alpha s_i^\alpha n_q^\alpha \, F_{qj}^p \tag{26}$$

where s^α and n^α are unit vectors, respectively parallel to the slip direction and normal to the slip plane. This crystal plasticity model

Table 1
Crystal plasticity parameters adopted in the simulations.

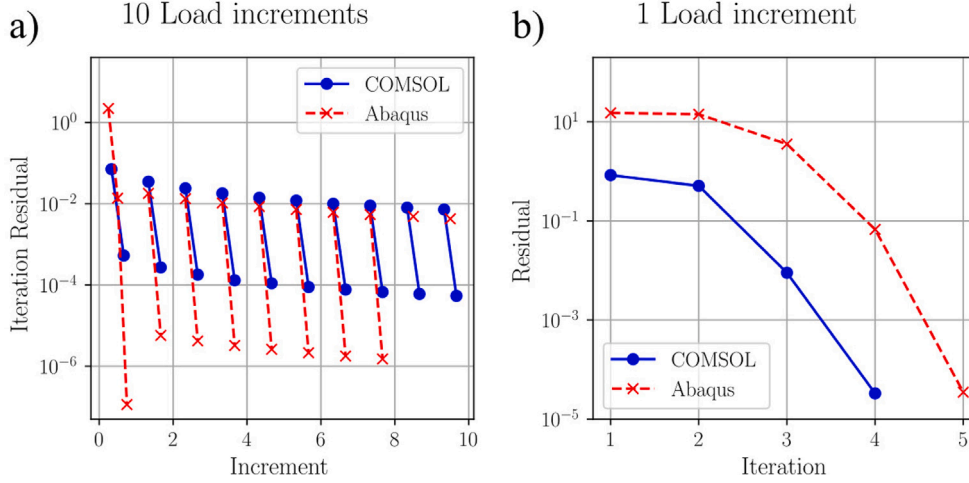| Parameter	C_{11} [GPa]	C_{12} [GPa]	C_{44} [GPa]	h_0 [MPa]	τ_s [MPa]	τ_0 [MPa]	$\dot{\gamma}_0$ [–]	n [–]	$q_{\alpha\beta}$ [–]
Magnitude	168.4	121.4	75.4	541.4	109.5	60.8	0.001	10	1

Fig. 5. Convergence plots of twisting cuboid simulation using a neo-Hookean *UMAT* material in both solvers, Abaqus and COMSOL. Subfigure (a) shows the residual evolution in 10 pseudo-time increments, while subfigure (b) provides convergence plots for 1 pseudo-time increment. In (a), iterations are represented with equispaced subdivisions within each increment interval.
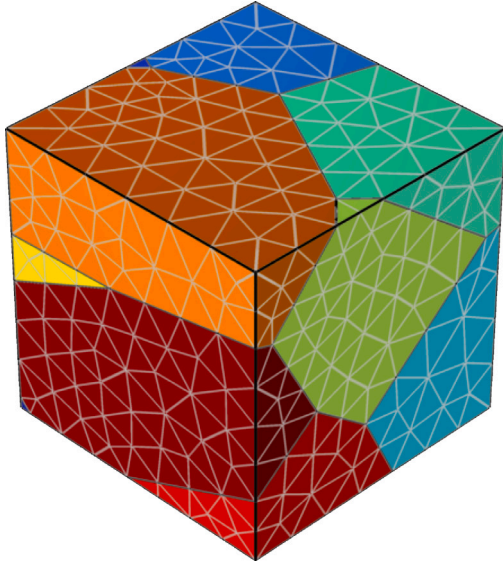

Fig. 6. Boundary value problem: crystal plasticity-based polycrystal containing 10 grains.

represents a face-centred cubic metal lattice with 12 octahedral slip systems and the parameters used are shown in Table 1.

The boundary value problem is a unit cube containing 10 randomly oriented grains, each behaving as a single crystal with elastic-viscoplastic behaviour, see Fig. 6. This is a common geometry in the crystal plasticity community [38]. One *External Material* must be defined in COMSOL for each grain, as while they all use the same material model, they require different input properties due to the different orientations. Second-order Lagrange tetrahedral elements are used for the polycrystal domain discretisation in both COMSOL and Abaqus (C3D10). Here, it should be noted that to ensure consistency in the number of integration points used, the reduced integration flag must be activated in COMSOL as this implies the use of 4 integration points per element, what Abaqus denotes as a fully integrated quadratic

tetrahedral element. The boundary conditions are chosen to mimic a uniaxial strain-controlled tensile test, with the displacement imposed on one face of the cube and the opposite face fixed. A final elongation of 5% is imposed. Given the rate-dependent nature of the problem, COMSOL's *Time Dependent* solver is used, with the time being discretised into 20 uniform increments of 1 s. The results obtained are shown in Figs. 7 and 8.

As in the previous case studies, the results obtained appear to be identical, independently of the finite element package used. No visible differences are observed in the deformed shape or the stress contours, see Fig. 7. Due to the different grain orientations, various locations of stress concentration can be observed, and these are present in both COMSOL and Abaqus predictions. From a quantitative viewpoint, the differences in the displacement solution fields observed are below 0.1%, validating our approach and implementation.

The predicted macroscopic stress–strain curve is shown in Fig. 8a. The results from Abaqus and COMSOL perfectly overlap. The convergence of the simulations is evaluated in Fig. 8b, which displays the evolution of the equilibrium error estimator. A quadratic or quasi-quadratic rate of convergence is observed. Both COMSOL and Abaqus exhibit a similar performance although on this occasion Abaqus appears to provide a better performance in the last increments. This is likely to be related to the different methods employed for estimating the relative residual (L2-norm in COMSOL and ∞-norm in Abaqus).

4.4. Coupled hydrogen diffusion and deformation in a single crystal

Finally, UMAT4COMSOL is used to model a problem of growing scientific and technological interest: the coupling between hydrogen diffusion and material deformation in a single crystal. Hydrogen is famed for 'embrittlement' metallic materials, significantly reducing their ductility, fracture toughness and fatigue crack growth resistance, and understanding these hydrogen-material interactions is critical to the safe deployment of hydrogen energy infrastructure [40,41]. To date, no commercial finite element package enables the coupling of hydrogen transport and crystal plasticity, a key element to gaining fundamental understanding and designing hydrogen-resistant microstructures. As
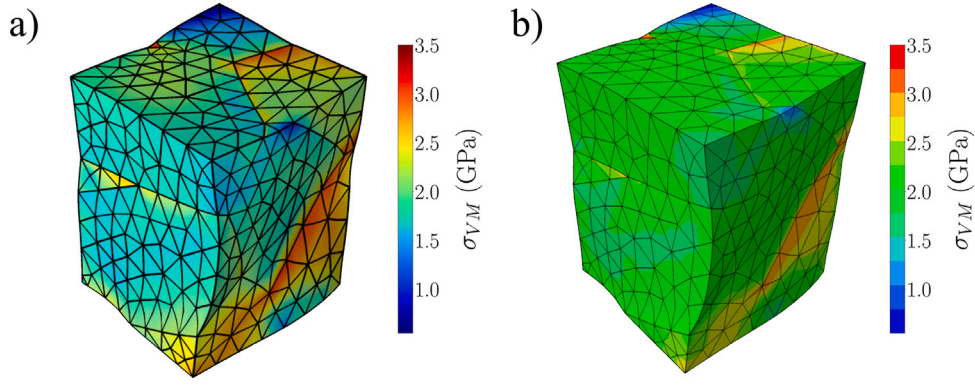
Fig. 7. Crystal plasticity-based predictions of polycrystalline deformation: contour plots of the von Mises stress (in GPa). Results obtained: (a) with Abaqus, using a crystal plasticity *UMAT*, and (b) with COMSOL, using the same *UMAT* and UMAT4COMSOL. The deformation has been scaled by a factor of 5.
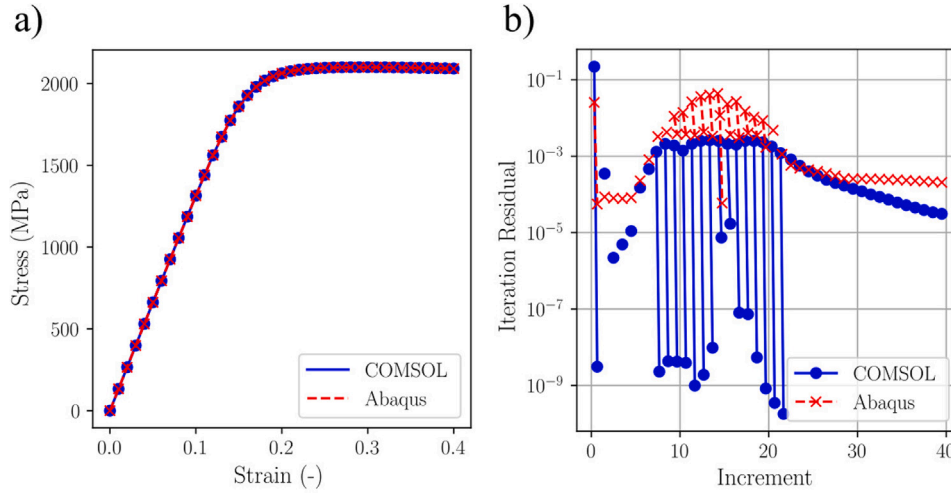

Fig. 8. Crystal plasticity-based predictions of polycrystalline deformation: (a) resultant macroscopic stress–strain curve, as predicted with Abaqus (+*UMAT*) and with COMSOL (+*UMAT* and UMAT4COMSOL); and (b) convergence plots for both COMSOL and Abaqus solvers, showing the magnitude of the residual for each iteration as a function of the load increment. In (b), iterations are represented with equispaced subdivisions within each increment interval.

discussed above, COMSOL does not include any crystal plasticity constitutive model, and Abaqus does not have the ability to conduct coupled deformation-diffusion. Thus, the ability of UMAT4COMSOL to break new ground is here showcased by exploiting COMSOL's abilities to run coupled analyses and the existing crystal plasticity material libraries available as Abaqus *UMAT* subroutines.

The analysis focuses on the behaviour of a single crystal containing a blunted crack of opening 0.01 mm. As depicted in Fig. 9, small scale yielding conditions are assumed and a so-called boundary layer formulation is adopted, whereby a remote K_I field is imposed by prescribing the displacement of the outer nodes based on William's linear elasticity solution (see, e.g., Ref. [42]). Finite strain conditions are assumed. The outer radius is taken to be sufficiently large so that it has no effect on the solution ($R = 150$ mm) and only half of the problem is simulated, taking advantage of symmetry. As shown in Fig. 9, the mesh is refined near the crack tip, to resolve the stress and strain gradients taking place there. This is a classic benchmark in the fracture mechanics community, which has also been comprehensively used in the study of hydrogen embrittlement in isotropic elastic–plastic solids [43,44]. However, to the best of the author's knowledge, this work constitutes its first application to single crystals. The material behaviour is described by the crystal plasticity model employed in Section 4.3, using the same material parameters (Table 1). The transport of hydrogen is simulated by using an extended version of Fick's law and assuming a state of equilibrium between the hydrogen located in lattice and trapping sites - so-called Oriani's equilibrium conditions [45]. Accordingly, the total

hydrogen content C is assumed to additively decomposed into the hydrogen concentration in lattice sites C_L and the hydrogen concentration in trapping sites C_T. Only one type of trap is considered: dislocations, and its evolution with the plastic deformation is defined by integrating the crystal plasticity model into the experimentally derived expression first proposed by Kumnick and Johnson [46], such that the trap density equals,

$$\log N_T = 23.26 - 2.33 \exp\left(-5.5 \sum_\alpha \gamma^\alpha\right) \tag{27}$$

where γ^α is the accumulated plastic shear strain of the crystal plane α.

Hydrogen diffusion through the crystal lattice is driven by gradients of concentration and of hydrostatic stress σ_h, bringing a second coupling with the mechanical problem. The balance equation for C_L reads,

$$\frac{\partial C_L}{\partial t} + \frac{\partial C_T}{\partial t} + \nabla \cdot \left(-D_L \nabla C_L + \frac{D_L \bar{V}_H}{RT} C_L \nabla \sigma_h\right) = 0 \tag{28}$$

where D_L is the lattice diffusion coefficient, T is the absolute temperature, R is the gas constant, and V_H is the partial molar volume of hydrogen. Then, denoting N_L as the density of lattice sites and W_B as the trap binding energy, Oriani's equilibrium results in the following relation between trapped and lattice hydrogen concentrations [47]

$$C_T = N_T \frac{\frac{C_L}{N_L} \exp\left(\frac{W_B}{RT}\right)}{1 + \frac{C_L}{N_L} \exp\left(\frac{W_B}{RT}\right)} \tag{29}$$
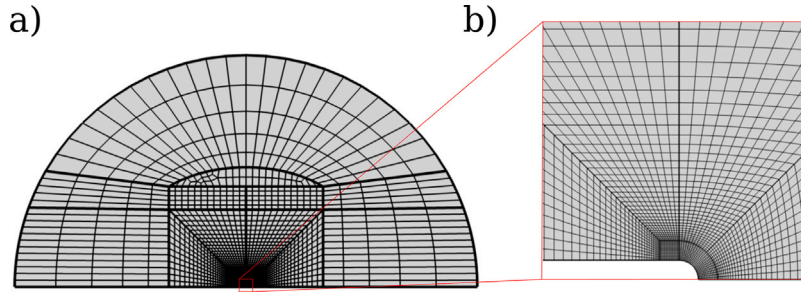
Fig. 9. Boundary layer formulation and finite element mesh used to describe the coupled deformation-diffusion crack tip behaviour of a single crystal exposed to a hydrogen-containing environment; (a) full geometry, and (b) detail of the crack tip region.
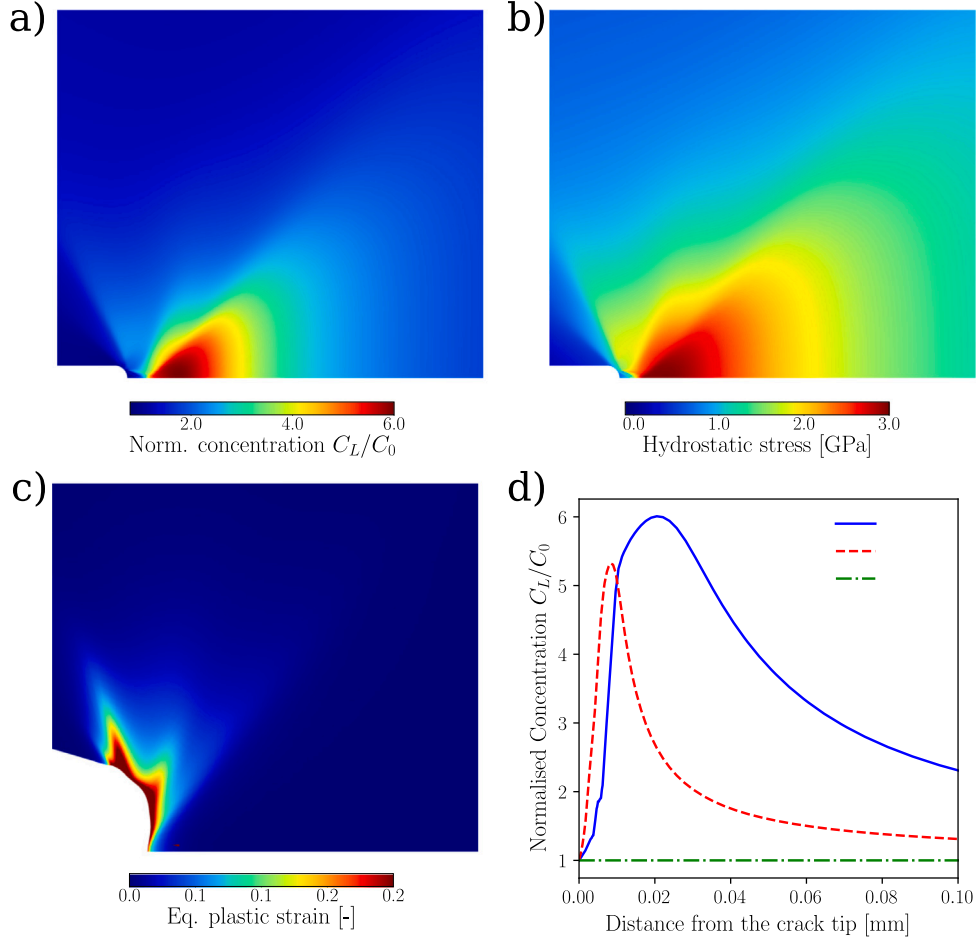

Fig. 10. Using UMAT4COMSOL to gain insight into the coupled deformation-diffusion behaviour of a single crystal exposed to a hydrogen-containing environment. Near crack tip contours of: (a) normalised lattice concentration C_L/C_0, (b) hydrostatic stress σ_h, and (c) equivalent plastic strain, with the deformation being scaled by a factor of 10. Subfigure (d) shows the quantitative changes in the C_L distribution ahead of the crack tip due to applied load and time.

Eqs. (28) and (29) can be readily considered in COMSOL by either by extending their *Transport of Dilute Species* module (as in Ref. [48]), or by using their new (as per version 6.2) hydrogen transport capabilities, based on the implementation by Hageman and Martínez-Pañeda [49].

We particularise the model to the study of a single crystal with lattice diffusion coefficient $D_L = 1.27 \times 10^{-8}$ m^2/s, density of lattice sites $N_L = 8.469$ mol/m^3, and partial molar volume $V_H = 2 \times 10^{-6}$ m^3/mol [43]. The trap binding energy associated with dislocations is taken to be $W_B = 60$ kJ/mol [46]. The single crystal is oriented parallel to the (001) direction, implying no rotations from the local to the global axis are applied. The temperature is assumed to be $T = 300$ K. The outer displacement is linearly increased with time until reaching an applied intensity factor of $K_I = 65.8$ MPa$\sqrt{m}$ at 96 s. The hydrogen concentration initial and boundary conditions are as follows; no flux

is assumed everywhere except for the crack surface, where a fixed concentration of $C_0 = 0.00346$ mol/m^3 is imposed, which is also the initial hydrogen concentration in the metal lattice. The computational domain is discretised with second-order Lagrange triangular elements with full integration and a multi-pass staggered (segregated) solution scheme is adopted. No comparison with Abaqus is provided, as this sort of analysis cannot be conducted using only *UMAT* subroutines (a UMATHT [50] or UEL [51] are needed). The results obtained are shown in Fig. 10, in terms of contours of normalised hydrogen concentration (C_L/C_0), hydrostatic stress (σ_h) and equivalent plastic strain $\varepsilon_p = \sum_\alpha \gamma^\alpha$.

The results obtained, given in Fig. 10, show the expected trends. The lattice hydrogen content (Fig. 10a) is highest in the regions of high hydrostatic stress (Fig. 10b). Since a conventional model is used, the

stress and concentration peaks are slightly ahead of the crack tip; this would change if the role of strain gradients is accounted for [52,53]. The equivalent plastic strain is highest near the crack tip, as shown in Fig. 10c, where the deformation has been scaled by a factor of 10. Accordingly, the hydrogen content trapped in dislocations (not shown) also increases as the crack tip is approached. Finally, Fig. 10d shows more quantitative estimates of normalised lattice hydrogen content ahead of the crack tip, showing the expected sensitivity of C_L to time and to the applied mechanical load. In terms of the convergence, the rate obtained is similar to those when using built-in elasto-plastic COMSOL materials.

5. Conclusions

We have presented UMAT4COMSOL, a new wrapper that links COMSOL C-coded *External Material* library with Abaqus Fortran-coded user material (*UMAT*) subroutines. This enables bringing to the multiphysics environment of COMSOL the advanced material models that the solid mechanics community has developed and implemented through Abaqus *UMAT* subroutines. Thus, UMAT4COMSOL enables the combination of advanced multi-physics and material modelling tools, as necessary to predict the degradation of Li-Ion batteries, the corrosion of metals and the oxidation of superalloys, among others. We validate our framework and demonstrate its potential by addressing three case studies of particular relevance: (i) the elastoplastic behaviour of a holed plate, (ii) the finite strain non-linear hyperelastic deformation of a twisted cube, and (iii) the crystal plasticity-based prediction of microscopic and macroscopic deformation in a polycrystalline solid. The results show that predictions obtained using COMSOL and UMAT4COMSOL are identical to those obtained using Abaqus and the associated *UMAT* subroutine. Moreover, convergence rates also exhibit a remarkable agreement, despite the different characteristics of each solver. To further highlight the potential of UMAT4COMSOL, the coupled deformation-diffusion behaviour of a single crystal exposed to a hydrogen-containing environment is simulated using a crystal plasticity *UMAT* subroutine and COMSOL's multi-physics capabilities. This analysis, of increasing technological importance, highlights the ability of the wrapper presented to assist in extending the state-of-the-art in materials and coupled physical modelling.

CRediT authorship contribution statement

Sergio Lucarini: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Data curation, Writing – original draft, Visualization. **Emilio Martínez-Pañeda:** Conceptualization, Methodology, Investigation, Resources, Writing – review & editing, Supervision, Project administration, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The software developed, UMAT4COMSOL, is made freely available together with example case studies and documentation. UMAT4COMSOL can be downloaded from the website of the research group (https://www.imperial.ac.uk/mechanics-materials/codes/), a website repository (https://www.empaneda.com/codes) and a GitHub repository (https://github.com/sergiolucarini).

Acknowledgements

References

[1] Martínez-Pañeda E. Progress and opportunities in modelling environmentally assisted cracking. RILEM Tech Lett 2021;6:70–7.

[2] Mianroodi JR, Shanthraj P, Liu C, Vakili S, Roongta S, Siboni NH, Perchikov N, Bai Y, Svendsen B, Roters F, Raabe D, Diehl M. Modeling and simulation of microstructure in metallic systems based on multi-physics approaches. npj Comput Mater 2022;8(1):1–15.

[3] Martínez-Pañeda E, Niordson CF, Gangloff RP. Strain gradient plasticity-based modeling of hydrogen environment assisted cracking. Acta Mater 2016;117:321–32.

[4] Gobbi G, Colombo C, Miccoli S, Vergani L. A fully coupled implementation of hydrogen embrittlement in FE analysis. Adv Eng Softw 2019;135(April):102673.

[5] Cui C, Ma R, Martínez-Pañeda E. A phase field formulation for dissolution-driven stress corrosion cracking. J Mech Phys Solids 2021;147:104254.

[6] Ansari TQ, Huang H, Shi S-Q. Phase field modeling for the morphological and microstructural evolution of metallic materials under environmental attack. npj Comput Mater 2021;7:143.

[7] Reuchet J, Remy L. Fatigue oxidation interaction in a superalloy - application to life prediction in high temperature low cycle fatigue. Metall Trans A Phys Metall Mater Sci 1983;14(A (1):141–9.

[8] Leo Prakash DG, Walsh MJ, Maclachlan D, Korsunsky AM. Crack growth micromechanisms in the IN718 alloy under the combined influence of fatigue, creep and oxidation. Int J Fatig 2009;31(11–12):1966–77.

[9] Martínez-Pañeda E, Golahmar A, Niordson CF. A phase field formulation for hydrogen assisted cracking. Comput Methods Appl Mech Engrg 2018;342:742–61.

[10] Kristensen PK, Niordson CF, Martínez-Pañeda E. A phase field model for elastic-gradient-plastic solids undergoing hydrogen embrittlement. J Mech Phys Solids 2020;143:104093.

[11] Valverde-González A, Martínez-Pañeda E, Quintanas-Corominas A, Reinoso J, Paggi M. Computational modelling of hydrogen assisted fracture in polycrystalline materials. Int J Hydrogen Energy 2022;47(75):32235–51.

[12] Cui C, Ma R, Martínez-Pañeda E. A generalised, multi-phase-field theory for dissolution-driven stress corrosion cracking and hydrogen embrittlement. J Mech Phys Solids 2022;166:104951.

[13] Hageman T, Martínez-Pañeda E. Stabilising effects of lumped integration schemes for the simulation of metal-electrolyte reactions. J Electrochem Soc 2023;170(2):021511.

[14] Boyce AM, Martínez-Pañeda E, Wade A, Zhang YS, Bailey JJ, Heenan TM, Brett PR, Dan JL. Shearing, cracking predictions of lithium-ion battery electrodes by X-ray computed tomography and modelling. J Power Sources 2022;526:231119.

[15] Ai W, Wu B, Martínez-Pañeda E. A multi-physics phase field formulation for modelling fatigue cracking in lithium-ion battery electrode particles. J Power Sources 2022;544:231805.

[16] Pan Z, Brassart L. Constitutive modelling of hydrolytic degradation in hydrogels. J Mech Phys Solids 2022;167(May):105016.

[17] Rambausek M, Mukherjee D, Danas K. A computational framework for magnetically hard and soft viscoelastic magnetorheological elastomers. Comput Methods Appl Mech Engrg 2022;391:114500.

[18] Moreno-Mateos MA, Hossain M, Steinmann P, Garcia-Gonzalez D. Hard magnetics in ultra-soft magnetorheological elastomers enhance fracture toughness and delay crack propagation. J Mech Phys Solids 2023;173:105232.

[19] Wu JY, Chen WX. Phase-field modeling of electromechanical fracture in piezoelectric solids: Analytical results and numerical simulations. Comput Methods Appl Mech Engrg 2021;387:114125.

[20] Quinteros L, García-Macías E, Martínez-Pañeda E. Electromechanical phase-field fracture modelling of piezoresistive CNT-based composites. Comput Methods Appl Mech Engrg 2023;407:115941.

[21] Nie JH, Hopkins DA, Chen YT, Hsieh HT. Development of an object-oriented finite element program with adaptive mesh refinement for multi-physics applications. Adv Eng Softw 2010;41(4):569–79.

[22] Patzák B, Rypl D, Kruis J. MuPIF-A distributed multi-physics integration tool. Adv Eng Softw 2013;60–61:89–97.

[23] Zhao Y, Stein P, Bai Y, Al-siraj M, Yang Y, B-xXu. A review on modeling of electro-chemo-mechanics in lithium-ion batteries. J Power Sources 2019;413:259–83.

[24] ABAQUS/standard user's manual, version 2022. United States: Dassault Systèmes Simulia Corp; 2022.

[25] COMSOL multiphysics® v. 6.1, COMSOL multiphysics reference manual. Stockholm, Sweden: COMSOL AB; 2022.

[26] Sun W, Chaikof EL, Levenston ME. Numerical approximation of tangent moduli for finite element implementations of nonlinear hyperelastic material models. J Biomech Eng 2008;130(6):061003.

[27] Ramasubramanian M, Wang Y. A computational micromechanics constitutive model for the unloading behavior of paper. Int J Solids Struct 2007;44(22):7615–32.

[28] Navidtehrani Y, Betegón C, Martínez-Pañeda E. A unified Abaqus implementation of the phase field fracture method using only a user material subroutine. Materials 2021;14(8):1913.

[29] Huang Y. A user-material subroutine incorporating single crystal plasticity in the abaqus finite element program. In: Harvard university report. 1991, p. 1–24.

[30] Kysar JW. Addendum to a user-material subroutine incorporating single crystal plasticity in the abaqus finite element program. In: Harvard university report. 1997, p. 1–3.

[31] Segurado J, Llorca J. Simulation of the deformation of polycrystalline nanostructured ti by computational homogenization. Comput Mater Sci 2013;76:3–11.

[32] Helfer T, Michel B, Proix J-M, Salvo M, Sercombe J, Casella M. Introducing the open-source mfront code generator: Application to mechanical behaviours and material knowledge management within the pleiades fuel element modelling platform. Comput Math Appl 2015;70(5):994–1023.

[33] Portillo D, del Pozo D, Rodríguez-Galán D, Segurado J, Romero I. Muesli-a material universal library. Adv Eng Softw 2017;105:1–8.

[34] Papazafeiropoulos G, Muñiz-Calvente M, Martínez-Pañeda E. Abaqus2Matlab: A suitable tool for finite element post-processing. Adv Eng Softw 2017;105:9–16.

[35] Helfer T, Bleyer J, Frondelius T, Yashchuk I, Nagel T, Naumov D. The 'mfront generic interface support' project. J Open Source Softw 2020;5(48):2003.

[36] Zienkiewicz OC, Taylor RL, Zhu JZ. The finite element method: its basis and fundamentals. seventh edn. Oxford: Butterworth-Heinemann; 2013.

[37] Logg A, Mardal K-A, Wells G. Automated solution of differential equations by the finite element method: The FEniCS book. Lecture notes in computational science and engineering, vol. 84, Springer Berlin, Heidelberg; 2012.

[38] Segurado J, Lebensohn RA, LLorca J. Chapter one - computational homogenization of polycrystals. In: Hussein MI, editor. Advances in crystals and elastic metamaterials, part 1. Advances in applied mechanics, vol. 51, Elsevier; 2018, p. 1–114.

[39] Peirce D, Asaro R, Needleman A. An analysis of nonuniform and localized deformation in ductile single crystals. Acta Metall 1982;30(6):1087–119.

[40] Gangloff RP, Somerday BP. Gaseous hydrogen embrittlement of materials in energy technologies. Cambridge: Woodhead Publishing Limited; 2012.

[41] del Busto S, Betegón C, Martínez-Pañeda E. A cohesive zone framework for environmentally assisted fatigue. Eng Fract Mech 2017;185:210–26.

[42] Martínez-Pañeda E, Betegón C. Modeling damage and fracture within strain-gradient plasticity. Int J Solids Struct 2015;59:208–15.

[43] Sofronis P, McMeeking RM. Numerical analysis of hydrogen transport near a blunting crack tip. J Mech Phys Solids 1989;37(3):317–50.

[44] Martínez-Pañeda E, Díaz A, Wright L, Turnbull A. Generalised boundary conditions for hydrogen transport at crack tips. Corros Sci 2020;173:108698.

[45] Oriani R. The diffusion and trapping of hydrogen in steel. Acta Metall 1970;18(1):147–57.

[46] Kumnick AJ, Johnson HH. Deep trapping states for hydrogen in deformed iron. Acta Metall 1980;28(1):33–9.

[47] Hageman T, Martínez-Pañeda E. A phase field-based framework for electro-chemo-mechanical fracture: Crack-contained electrolytes, chemical reactions and stabilisation. Comput Methods Appl Mech Eng 2023;415:116235.

[48] Díaz A, Alegre J, Cuesta I, Martínez-Pañeda E. A comsol framework for predicting hydrogen embrittlement - part i: coupled hydrogen transport. 2024, submitted for publication.

[49] Hageman T, Martínez-Pañeda E. An electro-chemo-mechanical framework for predicting hydrogen uptake in metals due to aqueous electrolytes. Corros Sci 2022;208:110681.

[50] Fernández-Sousa R, Betegón C, Martínez-Pañeda E. Analysis of the influence of microstructural traps on hydrogen assisted fatigue. Acta Mater 2020;199:253–63.

[51] Kristensen PK, Niordson CF, Martínez-Pañeda E. Applications of phase field fracture in modelling hydrogen assisted failures. Theor Appl Fract Mech 2020;110:102837.

[52] Niordson CF, Kysar JW. Computational strain gradient crystal plasticity. J Mech Phys Solids 2014;62:31–47.

[53] Martínez-Pañeda E, del Busto S, Niordson CF, Betegón C. Strain gradient plasticity modeling of hydrogen diffusion to the crack tip. Int J Hydrogen Energy 2016;41(24):10265–74.