



School of Engineering
Brown University

EN234: Computational methods in Structural and Solid Mechanics

Homework 9: Explicit Dynamics: modeling dynamic ductile fracture Due Wed Nov 18, 2015

Background problem The following preliminary coding exercise will be helpful in completing this weeks homework. Let

$$\phi(\sigma_e, p) = \left\{ \frac{\sigma_e^2}{Y^2} + 2q_1 f^* \cosh\left(\frac{3}{2} q_2 \frac{p}{Y}\right) - (1 + q_3 f^{*2}) \right\}^{1/2}$$

Where q_1, q_2, q_3, f^*, Y are constants. Let

$$\sigma_e = \sigma_e^* - \frac{3}{2(1+\nu)} E \Delta e_e \quad p = p^* - \frac{E}{3(1-2\nu)} \Delta \varepsilon_v$$

where σ_e^*, p^*, E, ν are constants.

Write a MATLAB code to solve the following equations for $\Delta e_e, \Delta \varepsilon_v$ by Newton-Raphson iteration (write your own Newton-Raphson loop, don't use the matlab equation solvers. The goal is to get the Newton loop to work so you can transfer it into your Fortran code)

$$F_1(\Delta e_e, \Delta \varepsilon_v) = \left[\left(\frac{\partial \phi}{\partial \sigma_e} \right)^2 + \frac{2}{9} \left(\frac{\partial \phi}{\partial p} \right)^2 \right]^{1/2} \left(\frac{\Delta e_e}{\Delta t \dot{\varepsilon}_0} \right) - \left(\frac{\partial \phi}{\partial \sigma_e} \right) \phi^m = 0$$

$$F_2(\Delta e_e, \Delta \varepsilon_v) = \left[\left(\frac{\partial \phi}{\partial \sigma_e} \right)^2 + \frac{2}{9} \left(\frac{\partial \phi}{\partial p} \right)^2 \right]^{1/2} \left(\frac{\Delta \varepsilon_v}{\Delta t \dot{\varepsilon}_0} \right) - \left(\frac{\partial \phi}{\partial p} \right) \phi^m = 0$$

where $\dot{\varepsilon}_0, m, \Delta t$ are constants. Use the following values for the constants (you can check with other values too but note that solutions only exist for $\phi(\sigma_e^*, p^*) > 0$):

$$E = 1000, \nu = 0.33, Y = 8000, \dot{\varepsilon}_0 = 0.01, m = 5,$$

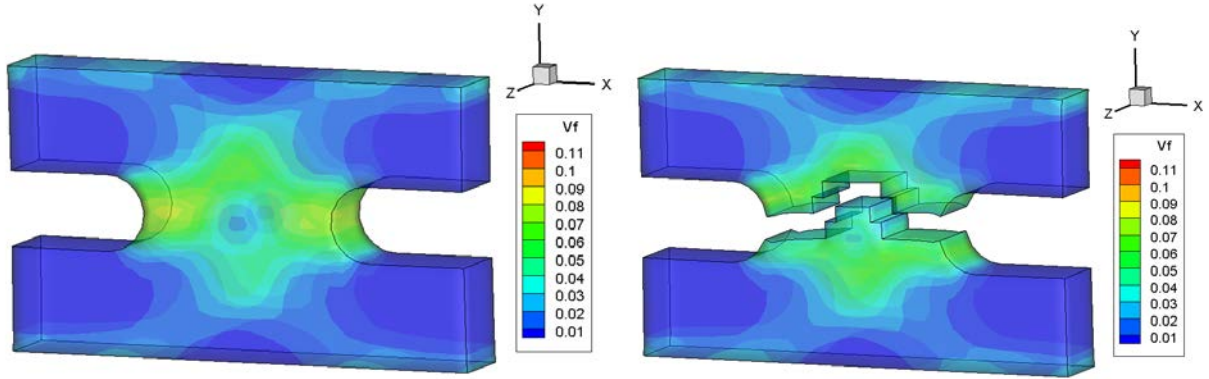
$$q_1 = 1.25, q_2 = 1.0, q_3 = 1.25,$$

$$f^* = 0.05, \sigma_e^* = 8500, p^* = 1000, \Delta t = 0.01$$

The Newton iterations require repeatedly solving the following linear equations for corrections $d\Delta e_e, d\Delta \varepsilon_v$

$$\begin{bmatrix} \frac{\partial F_1}{\partial \Delta e_e} & \frac{\partial F_1}{\partial \Delta \varepsilon_v} \\ \frac{\partial F_2}{\partial \Delta e_e} & \frac{\partial F_2}{\partial \Delta \varepsilon_v} \end{bmatrix} \begin{bmatrix} d\Delta e_e \\ d\Delta \varepsilon_v \end{bmatrix} = - \begin{bmatrix} F_1 \\ F_2 \end{bmatrix}$$

The hard part of this problem is calculating and typing in the derivatives of F correctly (note that ϕ is a function of $\Delta \varepsilon_e, \Delta \varepsilon_v$ so the derivatives of F contain second derivatives of ϕ). You need to be extremely organized in both your calculations and designing the code to make the steps transparent. Fortunately as long as the iterations converge sufficiently rapidly (4-5 iterations are typical) you can sometimes get away with a few small errors, as long as F is correctly calculated!



In the remainder of this homework you will implement an explicit dynamic simulation of ductile fracture in a polycrystalline metal, using the famous ‘Gurson’ plasticity model, which accounts for nucleation and growth of voids in a metal. You can find a short discussion of the Gurson model [here](#), and some more detail of the model that will be implemented here in the ABAQUS theory manual.

Implicit dynamics with finite strains: Because ductile fracture generally involves large plastic strains, we must implement a finite strain version of the explicit dynamic finite element method. The procedure followed here is somewhat similar to ABAQUS/Explicit, except that we will use the Jaumann rather than Green-Naghdi measure of stress rate (a more extensive discussion of stress rate measures and the difference between them can be found [here](#))

The finite-strain version of explicit dynamics is based on the principle of virtual power

$$\int_{V_0} \tau_{ij} \delta L_{ij} dV + \int_{V_0} \rho_0 \frac{\partial v_i}{\partial t} \delta v_i - \int_{A_2} t_i \delta v_i = 0$$

where τ_{ij} is the Kirchhoff stress; δv_i is a virtual velocity (a test function) and $\delta L_{ij} = \partial \delta v_i / \partial y_j$ is the virtual velocity gradient. Since we are solving a metal plasticity problem we interpolate displacement and virtual velocity fields using the finite-strain version of the B-bar method, which yields a system of equations for the accelerations

$$M_{ab} \frac{d^2 u_i^b}{dt^2} = -R_i^a + F_i^a$$

$$M_{ab} = \int_{V_0} N^a N^b dV_0 \quad R_i^a = \int_{V_0} \tau_{mj} [\bar{F}_{kl}] \left[\delta_{im} \frac{\partial N^a}{\partial y_j} + \frac{\delta_{mj}}{n} \left(\frac{\partial N^a}{\partial y_i} - \frac{\partial N^a}{\partial y_i} \right) \right] dV_0$$

Where $n=2$ for a 2D problem and $n=3$ for a 3D problem. EN234FEA integrates these equations with respect to time using a Newmark scheme, with lumped mass matrix, $\beta_2 = 0$ and $\beta_1 = 1$. This gives the following algorithm:

At time $t=0$ the velocities $v_i^a(0)$ and accelerations $a_i^a(0)$ are initialized (the user can specify the velocity; the accelerations are zero).

The time incrementation loop proceeds as follows:

1. Calculate $\Delta u_i^a = \Delta t \left(v_i^a(t) + \frac{\Delta t^2}{2} a_i^a(t) \right)$; impose any prescribed displacements.
2. Calculate $R_i^a(\Delta u_k^b)$

3. Calculate $a_i^a(t + \Delta t) = (-R_i^a + F_i^a) / M_{aa}$
4. Update $v_i^a(t + \Delta t) = v_i^a(t) + \Delta t a_i^a(t + \Delta t)$
5. Update $u_i^a(t + \Delta t) = u_i^a(t) + \Delta u_i^a$

This procedure is already coded in EN234FEA (the code is in explicit_dynamic_step.f90). You only need to write code to calculate $R_i^a(\Delta u_k^b)$, through the user subroutine.

To implement any new material model, you must calculate the Kirchhoff stress $\tau_{mj}[\bar{F}_{kl}]$ given a displacement increment Δu_i^a , and hence calculate the nodal forces $R_i^a(\Delta u_k^b)$. Once $\tau_{mj}[\bar{F}_{kl}]$ has been found, you can find $R_i^a(\Delta u_k^b)$ using the usual method.

In this homework, the stress will be calculated from the displacements using the constitutive relations for a porous plasticity model known as the 'Extended Gurson' model. This is very similar to the rate-dependent viscoplastic constitutive equation discussed in class, except that it also accounts for the nucleation, growth and coalescence of voids.

Summary of the Gurson model: Following the standard procedure in finite strain plasticity, we decompose the deformation gradient into elastic and plastic parts, which then allows us to decompose the total strain rate into elastic and plastic parts

$$\begin{aligned}
 F_{ij} &= F_{ik}^e F_{kj}^p \\
 \dot{\epsilon}_{ij} &= \text{sym}(\dot{F}_{ik} F_{kj}^{-1}) \quad W_{ij} = \text{skew}(\dot{F}_{ik} F_{kj}^{-1}) \\
 \dot{\epsilon}_{ij} &= \dot{\epsilon}_{ij}^e + \dot{\epsilon}_{ij}^p
 \end{aligned}$$

The plastic strain rate is related to the Kirchhoff stress by

$$\dot{\epsilon}_{ij}^p = \begin{cases} 0 & \phi < 0 \\ \frac{\dot{\epsilon}_0 \phi^m}{\sqrt{\left(\frac{\partial \phi}{\partial \sigma_e}\right)^2 + \frac{2}{9}\left(\frac{\partial \phi}{\partial p}\right)^2}} \left(\frac{\partial \phi}{\partial \sigma_e} \frac{3}{2} \frac{\tau_{ij}^D}{\sigma_e} + \frac{1}{3} \frac{\partial \phi}{\partial p} \delta_{ij} \right) & \phi > 0 \end{cases} \quad (1)$$

where Y is the yield stress of the matrix (we neglect strain hardening, so Y is constant), $\dot{\epsilon}_0, m$ are two material constants (a characteristic strain rate and the stress exponent), and

$$\begin{aligned}
 \phi &= \left\{ \frac{\sigma_e^2}{Y^2} + 2q_1 f^* \cosh\left(\frac{3}{2} q_2 \frac{p}{Y}\right) - (1 + q_3 f^{*2}) \right\}^{1/2} \\
 \tau_{ij}^D &= \tau_{ij} - \tau_{kk} \delta_{ij} / 3 \quad \sigma_e = \sqrt{3 \tau_{ij}^D \tau_{ij}^D / 2} \quad p = \tau_{kk} / 3
 \end{aligned}$$

Here, q_1, q_2, q_3 are three material properties, and f^* is a function that quantifies the loss of strength resulting from void growth. It is related to the volume fraction of voids in the material V_f by

$$f^* = \begin{cases} V_f & V_f < f_c \\ f_c + \frac{\bar{f}_F - f_c}{f_F - f_c} (V_f - f_c) & f_c < V_f < f_F \\ \bar{f}_F & V_f > f_F \end{cases} \quad (2)$$

Here, f_c, f_F are two material parameters that represent the critical void volume fraction when voids start to coalesce, and when the material ultimately fails entirely, and

$$\bar{f}_F = \frac{q_1 + \sqrt{q_1^2 - q_3}}{q_3}$$

The volume fraction of voids is zero in the initial material. As the material is plastically deformed, voids nucleate and grow, which causes V_f to increase. It is governed by

$$\frac{dV_f}{dt} = (1 - V_f) \dot{\varepsilon}_{kk}^p + \frac{d\bar{\varepsilon}_{matrix}}{dt} \frac{f_N}{s_N \sqrt{2\pi}} \exp\left(-\frac{1}{2} \left[\frac{\bar{\varepsilon}_{matrix} - \varepsilon_N}{s_N} \right]^2\right) \quad (3)$$

where f_N, s_N, ε_N are material properties controlling the rate of void nucleation with plastic strain (voids nucleate rapidly when the strain reaches ε_N), and $\bar{\varepsilon}_{matrix}$ is the total accumulated plastic strain in the matrix, which evolves according to

$$\frac{d\bar{\varepsilon}_{matrix}}{dt} = \frac{\dot{\varepsilon}_0}{1 - V_f} \phi^m \left[\left(\frac{\partial \phi}{\partial \sigma_e} \right)^2 + \frac{2}{9} \left(\frac{\partial \phi}{\partial p} \right)^2 \right]^{-1/2} \left(\frac{\partial \phi}{\partial \sigma_e} \sigma_e + \frac{1}{3} \frac{\partial \phi}{\partial p} p \right)$$

The elastic strain rate is related to the so called ‘co-rotational’ stress rate by

$$\overset{\nabla}{\tau}_{ij} = \frac{E}{1 + \nu} \left\{ \dot{\varepsilon}_{ij}^e + \frac{\nu}{1 - 2\nu} \dot{\varepsilon}_{kk}^e \delta_{ij} \right\}$$

where

$$\overset{\nabla}{\tau}_{ij} = \frac{d\tau_{ij}}{dt} + \tau_{ik} W_{kj} - W_{ik} \tau_{kj}$$

(for a more detailed discussion of rate forms of constitutive equations for elastic-plastic solid see [here](#))

Implementing the dynamic fracture/plasticity model requires three steps:

- (1) Given the displacement increment Δu_i^a and the displacement u_i^a at the start of the step, you must calculate the strain increment $\Delta \varepsilon_{ij}$, the spin increment ΔW_{ij} and the rotation increment ΔR_{ij}
- (2) Given $\Delta \varepsilon_{ij}$, ΔR_{ij} , the stress, and state variables at the start of the increment, calculate the stress and state variables at the end of the increment

$$(3) \text{ Calculate the element residual forces } R_i^a = \int_{V_0} \tau_{mj} [\Delta \bar{F}_{kl}] \left[\delta_{im} \frac{\partial N^a}{\partial y_j} + \frac{\delta_{mj}}{n} \left(\frac{\partial N^a}{\partial y_i} - \frac{\partial N^a}{\partial y_i} \right) \right] dV_0$$

These steps are described in more detail below, with a rough code template:

Loop over the integration points:

1. Calculate the increment in deformation gradient $\Delta F_{ij} = \frac{\partial N^a}{\partial x_j} \Delta u_i^a$ (note there is no identity added, since this is the change in \mathbf{F})
2. Calculate the mid-point deformation gradient $F_{ij}^{(t+\Delta t/2)} = \delta_{ij} + \frac{\partial N^a}{\partial x_j} (u_i^a + \frac{1}{2} \Delta u_i^a)$ and the associated Jacobian $J = \det(\mathbf{F}^{(t+\Delta t/2)})$
3. Compute the contribution from the current integration point to the volume averaged Jacobian and the average volumetric strain increment

$$\eta = \frac{1}{V_{el}} \int_{V_{el}} J dV$$

$$\Delta \eta = \frac{1}{\eta V_{el}} \int_{V_{el}} J \Delta L_{kk} dV \quad \Delta L_{ij} = \Delta F_{ik} F_{kj}^{(t+\Delta t/2)-1}$$

4. Compute the contribution from the current integration point to the volume averaged spatial shape function derivatives

$$\frac{\partial N^a}{\partial y_i} = \frac{1}{\eta V_{el}} \int_{V_{el}} J \frac{\partial N^a}{\partial y_i} dV \quad \frac{\partial N^a}{\partial y_i} = \frac{\partial N^a}{\partial x_k} F_{ki}^{(t+\Delta t/2)-1}$$

5. Compute the contribution from the current integration point to the element volume.

End integration point loop

Divide the volume averaged quantities by the element volume

Loop over the integration points a second time

1. Calculate the increment in deformation gradient $\Delta F_{ij} = \frac{\partial N^a}{\partial x_j} \Delta u_i^a$
2. Calculate the mid-point deformation gradient $F_{ij}^{(t+\Delta t/2)} = \delta_{ij} + \frac{\partial N^a}{\partial x_j} (u_i^a + \frac{1}{2} \Delta u_i^a)$
3. Compute the corrected velocity gradient increment
$$\Delta \bar{L}_{ij} = \Delta F_{ik} F_{kj}^{(t+\Delta t/2)-1} + (\Delta \eta - \Delta F_{ik} F_{kl}^{(t+\Delta t/2)-1}) \delta_{ij} / 3$$
4. Compute the strain and spin increments $\Delta \varepsilon_{ij} = (\Delta \bar{L}_{ij} + \Delta \bar{L}_{ji}) / 2$ $\Delta W_{ij} = (\Delta \bar{L}_{ij} - \Delta \bar{L}_{ji}) / 2$
5. Calculate the rotation increment associated with the spin, which is given by $\Delta R_{ij} = \delta_{ij} + \int_t^{t+\Delta t} W_{ik} R_{kj} dt$.

The integral can be approximated using a mid-point rule

$$\dot{\mathbf{R}} \mathbf{R}^{-1} = \mathbf{W} \quad \Rightarrow \quad \Delta \mathbf{R} \approx \mathbf{I} + \frac{\Delta \mathbf{W} (\mathbf{I} + \Delta \mathbf{R})}{2}$$

$$\Rightarrow \Delta \mathbf{R} \approx (\mathbf{I} - \frac{\Delta \mathbf{W}}{2})^{-1} \left(\mathbf{I} + \frac{\Delta \mathbf{W}}{2} \right)$$

where \mathbf{I} is the identity matrix.

6. Calculate the Kirchoff stress at the end of the increment $\tau_{ij}^{(n+1)}$ and calculate the values of the updated material state variables for the current integration point (do this inside a subroutine – see below for the steps)
7. Calculate the contribution to the element force vector from the current integration point

$$\mathbf{R} = - \int_{V_0} \bar{\mathbf{B}}^T \boldsymbol{\tau} dV_0$$

where $\bar{\mathbf{B}}$ can be assembled as

$$\bar{\mathbf{B}} = \begin{bmatrix} \frac{\partial N^1}{\partial y_1} + \frac{1}{3} \left(\frac{\partial \bar{N}^1}{\partial y_1} - \frac{\partial N^1}{\partial y_1} \right) & \frac{1}{3} \left(\frac{\partial \bar{N}^1}{\partial y_2} - \frac{\partial N^1}{\partial y_2} \right) & \frac{1}{3} \left(\frac{\partial \bar{N}^1}{\partial y_3} - \frac{\partial N^1}{\partial y_3} \right) & \frac{\partial N^2}{\partial y_1} + \frac{1}{3} \left(\frac{\partial \bar{N}^2}{\partial y_1} - \frac{\partial N^2}{\partial y_1} \right) & -\frac{1}{3} \frac{\partial N^2}{\partial y_2} \\ \frac{1}{3} \left(\frac{\partial \bar{N}^1}{\partial y_1} - \frac{\partial N^1}{\partial y_1} \right) & \frac{\partial N^1}{\partial y_2} + \frac{1}{3} \left(\frac{\partial \bar{N}^1}{\partial y_2} - \frac{\partial N^1}{\partial y_2} \right) & \frac{1}{3} \left(\frac{\partial \bar{N}^1}{\partial y_3} - \frac{\partial N^1}{\partial y_3} \right) & etc & \\ \frac{1}{3} \left(\frac{\partial \bar{N}^1}{\partial y_1} - \frac{\partial N^1}{\partial y_1} \right) & \frac{1}{3} \left(\frac{\partial \bar{N}^1}{\partial y_2} - \frac{\partial N^1}{\partial y_2} \right) & & & \dots \\ \frac{\partial N^1}{\partial y_2} & \frac{\partial N^1}{\partial y_1} & & 0 & \end{bmatrix}$$

This is the same B-bar matrix you used earlier, except that the derivatives are with respect to \mathbf{y} instead of \mathbf{x} .

End integration point loop

Stress update algorithm

The stress is updated using the approach discussed in class for small-strain plasticity problems (some of you may have implemented a small strain plasticity model in HW6 already). Given $\Delta \varepsilon_{ij}, \tau_{ij}^{(n)}, \bar{\varepsilon}_{matrix}^{(n)}, V_f^{(n)}$, we wish to determine $\tau_{ij}^{(n+1)}, \bar{\varepsilon}_{matrix}^{(n+1)}, V_f^{(n+1)}$, as follows:

1. Compute the elastic predictors for the deviatoric and hydrostatic stress

$$S_{ij}^* = \frac{E}{1+\nu} \Delta e_{ij} + \Delta R_{ik} \tau_{kl}^{Dn} \Delta R_{jl} \quad p^* = \frac{1}{3} \tau_{kk}^{(n)} + \frac{E}{3(1-2\nu)} (\Delta \varepsilon_{kk})$$

$$\Delta e_{ij} = \Delta \varepsilon_{ij} - \Delta \varepsilon_{kk} \delta_{ij} / 3 \quad \tau_{ij}^{Dn} = \tau_{ij}^{(n)} - \frac{1}{3} \tau_{kk}^{(n)}$$

2. Calculate

$$\phi = \left\{ \frac{\sigma_e^{*2}}{Y^2} + 2q_1 f^* \cosh \left(\frac{3}{2} q_2 \frac{p^*}{Y} \right) - (1 + q_3 f^{*2}) \right\}^{1/2}$$

$$\sigma_e^* = \sqrt{3 S_{ij}^* S_{ij}^* / 2}$$

$$f^* = \begin{cases} V_f^{(n)} & V_f^{(n)} < f_c \\ f_c + \frac{\bar{f}_F - f_c}{f_F - f_c} (V_f^{(n)} - f_c) & f_c < V_f^{(n)} < f_F \\ \bar{f}_F & V_f^{(n)} > f_F \end{cases}$$

If $\phi < 0$, the increment is elastic, and the stress follows immediately as

$$\tau_{ij}^{n+1} = S_{ij}^* + p^* \delta_{ij}$$

3. If $\phi > 0$, solve the following two equations for the magnitudes of the deviatoric and volumetric plastic strains $\Delta e_e, \Delta \varepsilon_v$ (using Newton-Raphson iteration – you did this in MATLAB already)

$$\left[\left(\frac{\partial \phi}{\partial \sigma_e} \right)^2 + \frac{2}{9} \left(\frac{\partial \phi}{\partial p} \right)^2 \right]^{1/2} \left(\frac{\Delta e_e}{\Delta t \dot{\varepsilon}_0} \right) - \left(\frac{\partial \phi}{\partial \sigma_e} \right) \phi^m = 0$$

$$\left[\left(\frac{\partial \phi}{\partial \sigma_e} \right)^2 + \frac{2}{9} \left(\frac{\partial \phi}{\partial p} \right)^2 \right]^{1/2} \left(\frac{\Delta \varepsilon_v}{\Delta t \dot{\varepsilon}_0} \right) - \left(\frac{\partial \phi}{\partial p} \right) \phi^m = 0$$

where ϕ is computed using the void volume fraction at the start of the increment, together with the following expressions for p and σ_e

$$\sigma_e = \sigma_e^* - \frac{3}{2} \frac{E}{(1+\nu)} \Delta e_e \quad p = p^* - \frac{E}{3(1-2\nu)} \Delta \varepsilon_v$$

4. The stress at the end of the increment can then be calculated from

$$\tau_{ij}^{n+1} = S_{ij}^* - \Delta e_e \frac{E}{1+\nu} \frac{3}{2} \frac{S_{ij}^*}{\sigma_e^*} + \left(p^* - \frac{E}{3(1-2\nu)} \Delta \varepsilon_v \right) \delta_{ij}$$

5. Finally, the void volume fraction and matrix strain can be updated as

$$\bar{\varepsilon}_{matrix}^{(n+1)} = \bar{\varepsilon}_{matrix}^{(n)} + \Delta \bar{\varepsilon}_{matrix}$$

$$V_f^{n+1} = 1 + (V_f^{(n)} - 1) \exp(-\Delta \varepsilon_v) + \frac{f_N \Delta \bar{\varepsilon}_{matrix}}{s_N \sqrt{2\pi}} \exp \left(-\frac{1}{2} \left[\frac{\bar{\varepsilon}_{matrix}^{(n)} - \varepsilon_N}{s_N} \right]^2 \right)$$

$$\Delta \bar{\varepsilon}_{matrix} = \begin{cases} 0 & \phi < 0 \\ \frac{\dot{\varepsilon}_0 \Delta t}{1 - V_f^{(n)}} \phi^m \left[\left(\frac{\partial \phi}{\partial \sigma_e} \right)^2 + \frac{2}{9} \left(\frac{\partial \phi}{\partial p} \right)^2 \right]^{-1/2} \left(\frac{\partial \phi}{\partial \sigma_e} \sigma_e + \frac{1}{3} \frac{\partial \phi}{\partial p} p \right) & \phi > 0 \end{cases}$$

Implementation Notes

- To implement an explicit dynamic element in EN234FEA, you must write two user-subroutines (i) a routine to compute the element residual force (there is an example in the el_linelast_3dbasic.f90 file). This routine must also compute values for the updated material state variables. The subroutine arguments are shown below.

```
subroutine el_linelast_3dbasic_dynamic(lmn, element_idenfier, n_nodes, node_property_list, &
n_properties, element_properties, element_coords, length_coord_array, &
dof_increment, dof_total, length_dof_array, &
n_state_variables, initial_state_variables,
updated_state_variables, element_residual, element_deleted) ! Output variables
```

(ii) the usual routine to project stresses and other field variables (eg the void volume fraction) to the nodes.

- The Gurson material has 13 material properties: $E, \nu, Y, \dot{\epsilon}_0, m, q_1, q_2, q_3, f_N, \epsilon_N, s_N, f_c, f_F$
- The material has the following state variables: stress components τ_{ij} , the matrix strain $\bar{\epsilon}_{matrix}$ and the void volume fraction V_f . The values of these variables need to be stored, and updated, for at each integration point in the element. In EN234FEA, all state variables for an element are stored in two 1-D vectors: the user subroutine will provide the values of state variables at the start of the step in a vector called `initial_state_variables`; your user-subroutine must return their values at the end of the step in a variable called `updated_state_variables`. You can use any storage scheme you like for your variables as long as they are consistent. For example, in my code I extract the state variables for the first integration point as

```
stress0 = initial_state_variables(1:6) ! Stress at start of increment
ematrix = initial_state_variables(7)
Vf = initial_state_variables(8)
```

The data for the second integration point starts at `initial_state_variables(9)`, and so on.

- It is best to separate the calculations into element operations (calculating shape function derivatives, deformation measures, etc) and material operations (calculating the stress). There are various ways to do this, but the basic idea is to write a subroutine that calculates the updated stress and updated state variables for just one integration point, given values of the state variables, the strain increment, and the rotation increment, with a subroutine of the form

```
subroutine gurson(element_properties,n_properties,n_state_variables,initial_state_variables, &
updated_state_variables,dstrain,dRot,stress1)
  use Types
  use ParamIO
  use Globals, only : TIME, DTIME

  implicit none

  integer, intent( in )      :: n_properties
  integer, intent( in )      :: n_state_variables

  real (prec), intent( in )  :: element_properties(n_properties)
  real (prec), intent( in )  :: initial_state_variables(n_state_variables)
  real (prec), intent( in )  :: dstrain(3,3)
  real (prec), intent( in )  :: dRot(3,3)

  real (prec), intent( out ) :: stress1(6)
  real (prec), intent( out ) :: updated_state_variables(n_state_variables)
```

Note that `n_state_variables` here is the number of state variables for ONE INTEGRATION POINT, not all of them, and you must slice the vector that stores the full set of variables for all the integration points to pass the correct set to this subroutine.

- The condition $\phi > 0$ is best checked numerically as $\phi > \epsilon_{ps}$, where ϵ_{ps} is a small number (10^{-8}) to avoid divide by zero errors.
- The ‘Element Utilities’ module in EN234FEA defines a function to compute $\Delta R_{ik} \tau_{kl}^{Dn} \Delta R_{jl}$

```
stress1 = rotatesymvec(stress0,dR)
```

Here `dR` is a 3x3 orthogonal matrix and `stress0` is a vector containing components of a symmetric tensor `s=[s11,s22,s33,s12,s13,s23]`.

- Both EN234FEA and ABAQUS allow elements to be deleted during an explicit dynamic computation, to simulate material failure. In EN234FEA this is accomplished by setting the

‘element_deleted’ variable to .true. inside the user-element subroutine. You could delete elements when $V_f > f_F$ for all the integration points in the element.

- You are coding and running a fairly sophisticated FEA model in this homework, so debugging might take a bit of work. The following procedure will test your code in stages:

1. Run your code with only two elements (eg with the mesh in Gurson_3d_dynamic.in), and parameter values that will produce an elastic material, eg

$$E = 1000, \nu = 0.33, Y = 8000, \dot{\epsilon}_0 = 0.01, m = 5,$$

$$q_1 = 1.25, q_2 = 1.0, q_3 = 1.25,$$

$$f_N = 0, \epsilon_N = 0.05, s_N = 0.05, f_c = 0.15, f_F = 0.25$$

Use a mass density $\rho = 10$ and a time-step of 0.01 units. You can apply any sensible boundary conditions. Your code should reproduce the results in linear_elastic_dynamic_3d.in (with the same boundary conditions).

2. Run your 2 element test with plasticity, but no void nucleation, eg.

$$E = 1000, \nu = 0.33, Y = 0.8, \dot{\epsilon}_0 = 0.01, m = 5,$$

$$q_1 = 1.25, q_2 = 1.0, q_3 = 1.25,$$

$$f_N = 0, \epsilon_N = 0.05, s_N = 0.05, f_c = 0.15, f_F = 0.25$$

Check the convergence of the Newton iterations in the plasticity model – they should converge after 2-4 iterations. If this does not happen there is probably an error in the derivatives you are using in the Newton-Raphson loop (but hopefully you caught any problems in the preliminary MATLAB code you wrote, so this part should work).

3. If step (2) works repeat the 2 element test with void nucleation – try parameters

$$E = 1000, \nu = 0.33, Y = 0.8, \dot{\epsilon}_0 = 0.01, m = 5,$$

$$q_1 = 1.25, q_2 = 1.0, q_3 = 1.25,$$

$$f_N = 0.05, \epsilon_N = 0.05, s_N = 0.05, f_c = 0.15, f_F = 0.25$$

(suppress element deletion for this test). Check the convergence of your Newton iterations. In this simulation you should see the stress in the elements drop as the material fails.

4. If test (3) works, you can try running the simulation shown in the figure. An input file called notch_fracture_dynamic.in has been provided for this purpose. You may need to change the order of the material property definition to be consistent with your code.
5. **Optional:** if you are curious you can make your own fracture specimen and (virtually) break it. The simulation will run faster if you compile and run your code in release mode instead of debug mode.