# Evaluation of J-integral 2D

This is the derivation of the J-Integral for the 2D case. In our case the crack front is not curved along the z-direction; however, the indenter and free surface result in variation in the stress field along the crack which makes the problem 3D. That being said, every specimen designed to get a stress intensity factor also has 3D effects and it is unclear if the 3D effects are substantial at the point of maximum dissipation, i.e. directly under the indenter tip. Regardless, it is worth understanding the 2D formulism before discussing the details of evaluating the 3D integral as the 3D integral is just a scheme to approximate the pointwise (2D) energy release rate. In the context of quasi-static analysis the J-integral is defined in two dimensions as



where  is a contour beginning on the bottom crack surface and ending on the top surface. The limit as  indicates that shrinks onto the crack tip and tractions on the crack are ignored,  is a unit vector in the direction of crack extension and is the outward normal to . , which is the Eshelby’s energy-momentum tensor is given for small strains by



For a linear elastic material,  is just the elastic strain energy density and no integral is needed in its evaluation; furthermore, there is no dependence on  i.e. we don’t have a graded material. Based on Abaqus documentation, the exported J integral in Abaqus should be not used at finite strains. To write in terms of finite strains, the right hand term would be written in terms of the engineering Piola-Kirchoff stress and the deformation gradient. We can recast in terms of the problem shown in Figure 1 and the integral becomes



where  is the traction on the crack faces and , which is an arbitrary smooth weighting function with the region enclosed , is equal  on  and equal 0 on . The equivalency of and can be understood in that  is chosen such that if the contours were evaluated  would disappear and as  is shrunk onto the crack tip the contribution to the contour integral  when there is a traction on the surface of the crack vanishes.

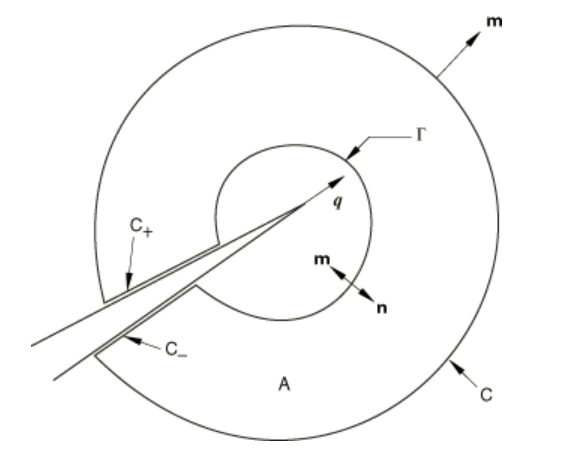


Figure . Closed contour enclosed a domain A that includes the crack-tip region as .

Applying the divergence theorem and assuming the crack surface is traction free we get the classical result for 2D



We are only interested in the crack along its propagation direction and is simplified for our geometry to



which is the result of (Shih et al., 1986) in the absence of thermal strain, body force and crack face traction.

# Evaluation of J-integral 3D

In 3D we are concerned with the pointwise energy release along a crack front which for just the second component of  has the form



and is schematically shown in figure 2.

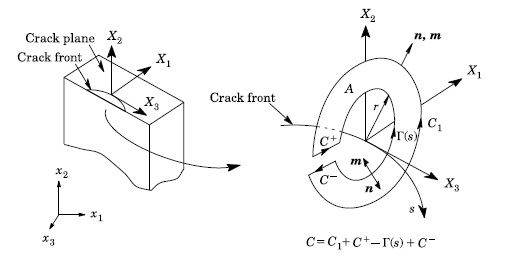


Figure . Schematic of  in . Schematic showing the relation of the problem to .

Note that the gradient terms in are with respect to the local frame pictured in Figure 2; however, we consider a straight crack front in our analysis and global frame coincides with the local frame.

In general, the vector integral is



where  on  and  is the unit normal to the crack-front at position .

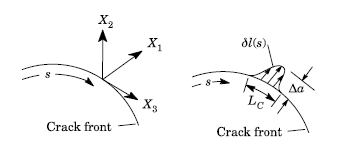


Figure . Virtual crack advance in the local X1-X3 plane at crack-front location s. Crack advance occurs in the X1-direction and is defined as in .

Consider a virtual displacement applied along the crack segment as pictured in Figure 3.



Here  is the amplitude of an arbitrary displacement  in the direction. To first order, the energy release due to crack advance given in is



where is the energy released when crack segment  advances by . Substituting in we get

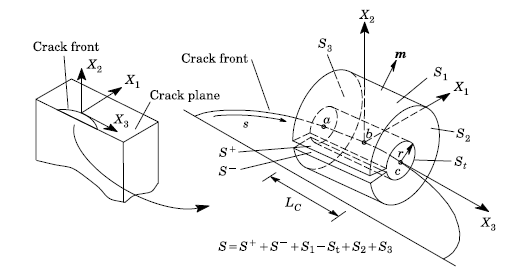


Combining the integrals is akin to extruding the contour into a surface, , which is a cylinder surrounding the crack front



wherein we note for completeness that  is removed because the dot product of a unit vector with itself is unity. Following the 2D formulism, we create an enclosed volume , , as pictured in Figure 4 and introduce a smooth vector field q which is  at and 0 on  and rewrite assuming no surface tractions as





Figure

Applying the divergence theorem, we obtain



Finally, we would like to relate  to  which can be done by assuming  does not very significantly over the interval . Rearranging we get



# Evaluation of J-integral with phase boundaries in 3D

Consider the contours in figure 5 at a position s along a crack front. The phase boundary contribution to the J integral in the absence of traction at the crack, body forces, thermal strain, and material property gradients is



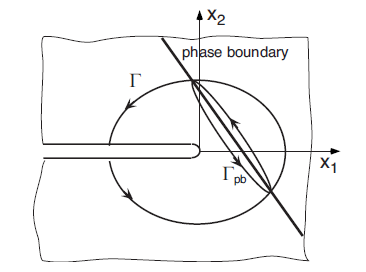


Figure . Contours for J-integral evaluation at a crack tip located near a phase boundary.

Substituting into we get



Extruding along s we obtain the surface integrals



While we can formulate as a volume integral around the phase boundary, this approach would preclude studying the exact. For this reason, we obtain the final form following the procedure outlined in the previous section as



Given is nonzero in the 2 direction can be written as



# Numerical integration of the J integral

The numerical integration of is performed using shape functions of the C3D20 elements. Consistent with the isoparametric formulation, we take  within an element as



On  and on, but in-between the boundaries  is between 0 and 1. A number of options for the shape of the virtual crack growth can be chosen. We choosing to evaluate at node M using a piecewise linear function as shown in Figure 6. The choice considers contributions to the release rate at the node from its surrounding elements. We do not evaluate  at the mid side nodes like Abaqus does. For quadratic elements and setting a mid-node to be zero, q must become negative between M-1 and M+1 and the mid nodes. Thus, the value of  will be slightly higher than it should be, but the spatial averaging along the crack front will be minimized. The spatial averaging under the indenter is a secondary effect there is no need for a more complicated script that evaluates computes  at the mid-nodes.

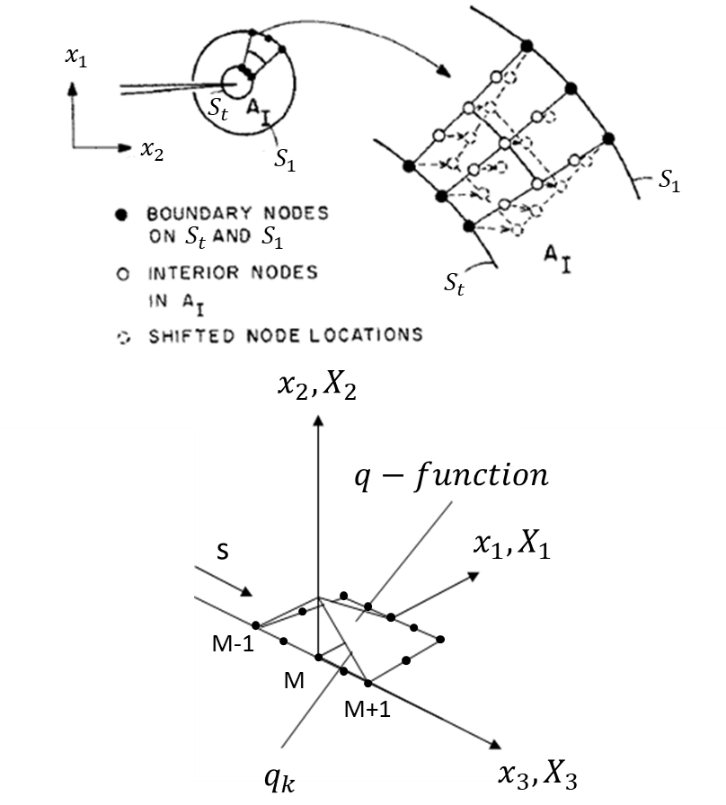


Figure . (top) is a schematic showing a linearly decreasing virtual shift of nodes in the 2 direction between S1 and St for the node set corresponding to M. (bottom) is a schematic showing how q varies along the crack.

Some care needs to be given to the choice of  between the contour regions, especially when the mesh is poor in some areas. The most accurate results will be from regions where the mesh quality is high, structured, and away from the crack tip. Here we choose q to be a linear function between the inner contour and outer contour surfaces e.g. see Figure 6 example for how q would be defined for the first ring of elements.

The gradient of q is computed at the gauss integration points using shape functions.



Similarly, the gradient of displacement is given by



For the surface and line integral, q is interpolated to gauss integrations points in the interface plane or crack front. In the next section, details of the integration procedure are given in terms of a transformation of the 3D shape functions. Mainly, we use for the line integral 2 Gaussian integration points, for surface integrals 3x3 Gaussian integration scheme, and for the volume integrals we use a 3x3x3 Gaussian integration. The discretized form of the domain expression for the energy release rate is



The integral in the denominator of is evaluate between M-1 and M and between M and M+1 as



In the case M is evaluated at a surface, the integral becomes



# Isoparametric formulization for C3D20 element

## General concepts

The basic idea of isoparametric finite element formulation is to achieve the relationship between element displacements at any point and the element nodal point displaces directly through the interpolation (shape) functions. For a general three-dimensional element, the coordinate interpolations are



where  are the global coordinate components at any point of the element with local (natural) coordinates , and , are the coordinates of the q element nodes in global coordinates. The interpolation functions  are defined in the element natural coordinates , , and  respectively. The natural coordinates vary from  to , thus the locations element nodes in the natural and global coordinate systems are available and natural coordinates are in the appropriate interval for gaussian quadrature. The fundamental property of the interpolation function is that  is unity at node  and zero at all other nodes.

Similar to coordinates, the displacements are defined as



where  are the displacements at any point of the element in global coordinates, and , are the displacements of the q element nodes in the global coordinate system. The transformation from local to global system for derivatives will be discussed later. The abaqus node ordering and local frame is schematically presented in figure 7.

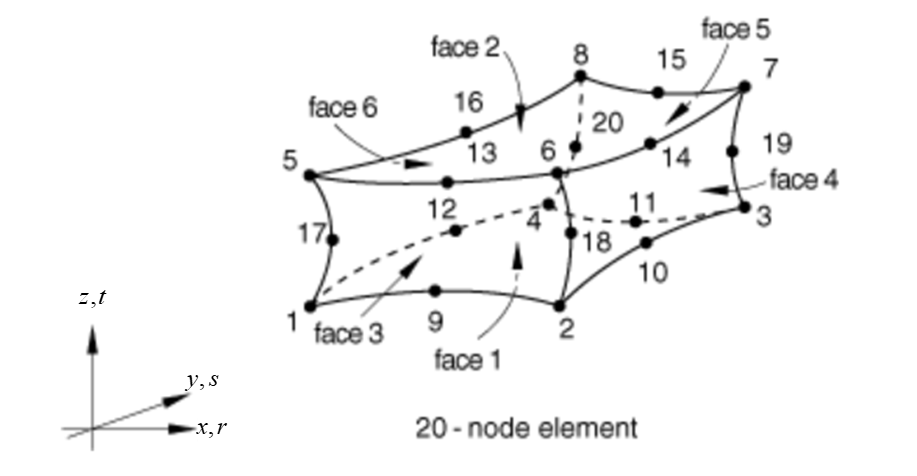


Figure . Node ordering, frame, and face labeling from abaqus [manual](https://help.3ds.com/2019/english/DSSIMULIA_Established/SIMACAEELMRefMap/simaelm-r-3delem.htm?ContextScope=all&id=24dc3c1d01ec40b294d5fb1d3d2d803e#Pg0) .

Using the node ordering presented in figure 1 the shape functions can be constructed by inspection:



where



Thus, the shape functions for the Abaqus C3D20 element can be constructed following and .

The shape functions for the C3D20 using the node labeling convention in figure 7 is



The derivatives of these shape functions  are derived with respect to r as



with respect to s as



and with respect to t as



To map the derivatives in the local coordinates to global coordinates or transform integrals from the global coordinate system to the local coordinate system, we need to build the Jacobian of the transformation in and using , , and



where each derivative is defined as



The derivatives for displacement and other quantities described by the shape functions can be constructed in the same manner.

An n-point gaussian integration is utilized in finite elements because it can be used to calculate the exact result of a polynomial function polynomials of degree 2n-1 or less by a suitable choice points  and weights . The integral is generally expressed in the local coordinates as



and the transform of the integrals from global coordinate systems to local coordinate system is



The C3D20 element uses a 3x3x3 gaussian integration (3 gauss points along each dimension) and the volume integral is exact. The ordering of the nodes is shown in figure 8.

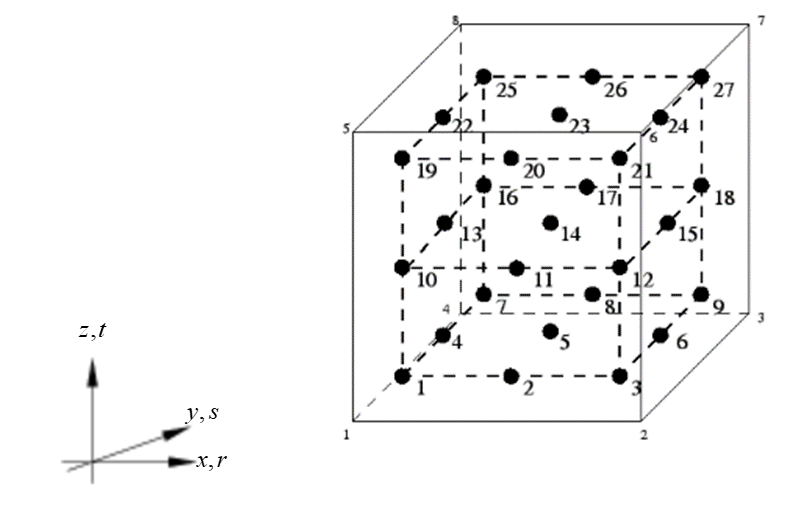


Figure . Integration point numbering for Abaqus C3D20 element taken from [here](http://web.mit.edu/calculix_v2.7/CalculiX/ccx_2.7/doc/ccx/node29.html).

The weights and locations of the gauss points for each dimension follow the 3-point Gauss rule and are summarized in Table 1 for the first three nodes and the derivation of the points can be found [here](https://www.sciencedirect.com/topics/engineering/gauss-quadrature). Matlab scripts for performing the Gauss integration can be found [here](http://jupiter.ethz.ch/~gfdteaching/femblockcourse/2015/other/KwonBang_Chapter6_2pagesperside.pdf) starting page 12.

Table 1. Worked out example for 3D 3-point quadrature rule.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Point 1 | Point 2 | Point 3 | |
| r  w |  | 0 |  |
| s  w |  |  |  | |
| t  w |  |  |  | |

In the case that we wish to interpolate from Gauss points to nodes we need to modify the basis of natural coordinates to the Gauss points. The problem is visualized in 2D in Figure 5. We observed based on Table 1 that when ,  and there a proportionality constant of . The variable which denotes the location in natural coordinates of the value being extrapolated remains the same as before but the natural nodal coordinates which are usually less than or equal to 1 become less than or equal to  (i.e. for node 3 in Figure 5, the nature coordinates are and we replace  in with) .

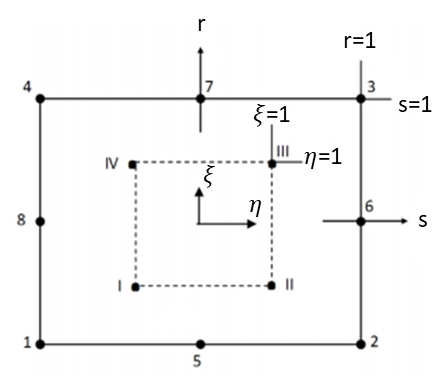


Figure . 2D schematic showing the two coordinate systems.

Unlike in the case of the 2x2x2 integration and the C3D8 element, 3x3x3 integration scheme means that the form of the shape functions in is not valid for extrapolation from all of the integration points. We must create a mapping between the node convention and the integration point conventions in figures 3 and 4 which is summarized in Table 2. Essentially, some integration points that would correspond to a face node or center node, can be ignored.

Table 2. The mapping between element node ids (N) and corresponding integration point ids (GP) for extrapolation from integration point data.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| N | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| GP | 1 | 3 | 9 | 7 | 19 | 21 | 27 | 25 | 2 | 6 | 8 | 4 | 20 | 24 | 26 | 22 | 10 | 12 | 18 | 16 |

## Transformation of 3D shape functions for boundary integral

We can reduce the 3D formulism for the C3D20 element, however, the triquadratic shape functions do not reduce simply to biquadratic shape functions at the surface, rather there remains some dependence on the mid nodes e.g.



We can similarly reduce the shape functions for a line in a similar manner. The main strategy for numerical integration is to interpolate/extrapolate needed quantities to integration points on a surface or line that we want to perform integration on. Noting that our integration function varies in r and s and realizing that the order of the polynomial is simply taken as the product of variables, the degree of the polynomials in the shape function is 3 in the 2D case and the number of gauss point for exact integration is a 2x2 grid in the plane at t=1 for the example given. In the case of a line integral 2 gauss points are needed because the order is 2. The integral transforms from 3D to 2D area integrals and 1D line integrals are based on simple calculus and are presented as follows: for the 2D case assuming ds=0, the integral is given as



and for a line assuming dr=ds=0 as



All of the quantities needed to compute these integrals are available from the general shape functions and their derivatives.

# Validation of J integral without interfaces

## Problem statement

The J-integral computed for the indenter/crack problem by Abaqus has node sets that appear to be incorrectly defined based on the description of the methods and I think this is the reason why when we specify more than about 10 contours there is a memory issue.

To verify the 3D J integral procedure in the absence of interfaces, a simpler, well know problem of an through thickness crack in a infinite plate under uniaxial stress is utilized.

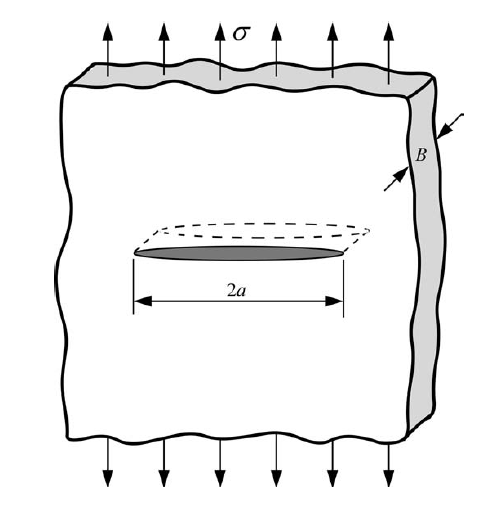


Figure . A schematic showing problem to be modeled.

The analytical solution for the problem for KI stress intensity factor is simply

.

# Model dimensions and modeling considerations

a\_crack=0.5,y\_halfsize=4,x\_halfsize=3, z\_halfsize=2. Symmetry conditions are utilized in the x, y, and z direction. A stress of 10000 is applied directly to the Y-Z plane at X=0 and the material properties are E=200e9 and v=0.32.

C3D20 are utilized at and around the crack with nodes collapsed to create a singularity. The mesh was designed around so that around the crack there are 10 rings of elements (See Figure below).

For the applied stress there is no difference between finite and small strain formulation results for  or . Increasing the applied stress has only scales the computed quantities.

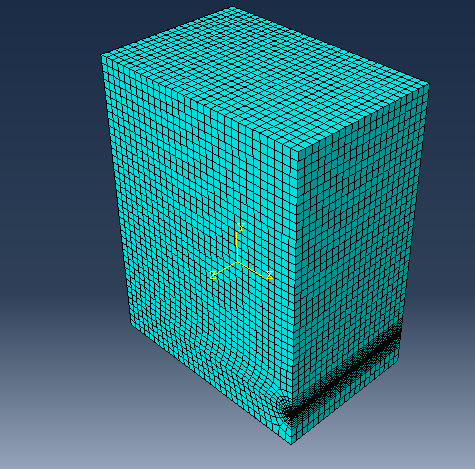


Figure . Mesh design.

Table 1. Summary of calculated and for test case.

|  |  |  |  |
| --- | --- | --- | --- |
| Contours | 1 | 2 | 3-10 |
|  | 12707 | 12959 | 12964 |
|  | 7.2434E-04 | 7.5295E-04 | 7.5357E-04 |

The results are summarized in Table 1. The analytical solution for  is 12533, putting the error at 3.4%. Looking at the stress field in Figure 3, it is clear the plate is not big enough. Doubling the size of the model in X results in a little over 1% error and the contours regions are appropriately reconstructed by Abaqus, suggesting this simulation is good for verifying the computation of the J-integral.

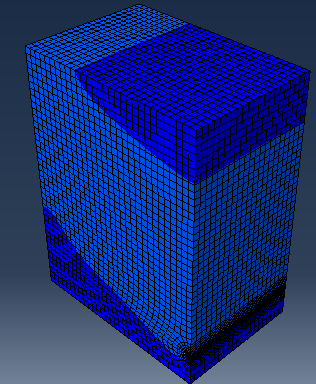


Figure . Mises stress contours.

# Computation of J-integral

The computation of the J-integral is performed with a set of python scripts designed for use with C3D20 elements only. The scripts can easily be adapted to other element types, but the shape functions and their derivatives must be added. The sets needed to compute the contours are added to ODB automatically. Only the node label for the first contour along the crack, the part instance, (e.g. Figure 4 below) and whether it is as at a symmetry plane or end is input by the user. To get a node label and the part instance go to Tools->Query select node and click on the node of interest.

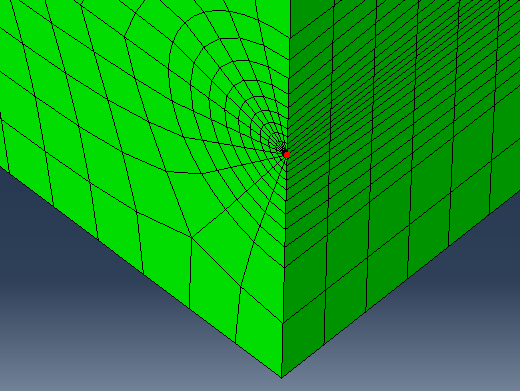


Figure . A valid choice of a node selected for J-integral computation.

Element sets are constructed starting at that node and going along the crack-front as shown in Figure 5.

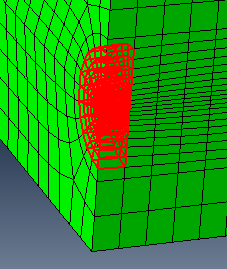
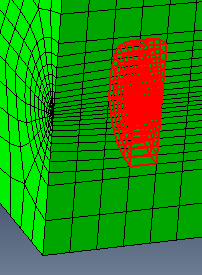
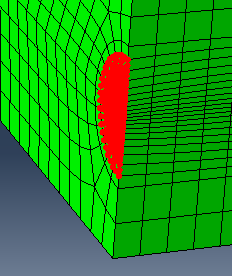
  

Figure . Example of contour level 10 element sets for slice 0 (left) and slice 3 (middle) for computation of volume J-integral. A procedure is performed when calculating the volume integral which makes it equivalent to the area integral of the nodeset shown in Abaqus (right). Abaqus does not use nodes when evaluating J, rather the node sets are a visual of the equivalent area integral which the J-integrals represent.

## Validation of shape functions

Abaqus allows the coordinates to be exported at nodal and element locations such as Gauss points or element nodal positions which should be the same as for nodal positions for some variables such as coordinates, making it the only variable readily available to verify the shape functions. Figure 6 gives the element and nodal layout of a two element stack pictured in Figure 7. For the validation we will look at the x component of the coordinates at time=0 in the simulation (no deformation). Note that the smallest value at nodal locations are zero, but element nodal positions somehow have a non-zero value. This seems odd at first since there is no interpolation between nodal and element nodal positions; however nodal values and element nodal values are different due to how they are populated. Abaqus appears to calculate element nodal values from the integration point values and this is the reason for the discrepancy in coordinate values in Figure 7.

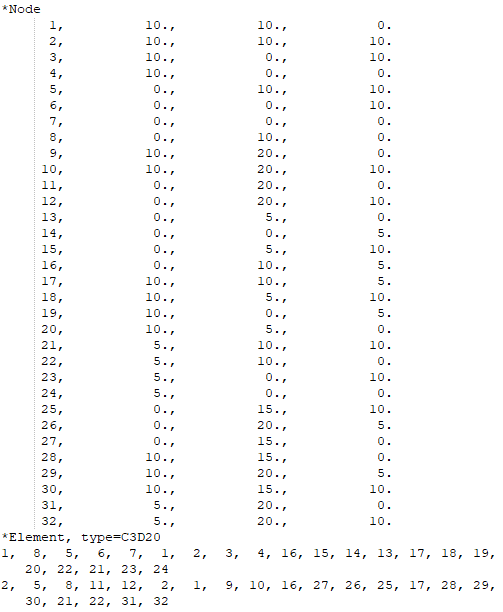


Figure . Node and element definitions in Abaqus input file form.

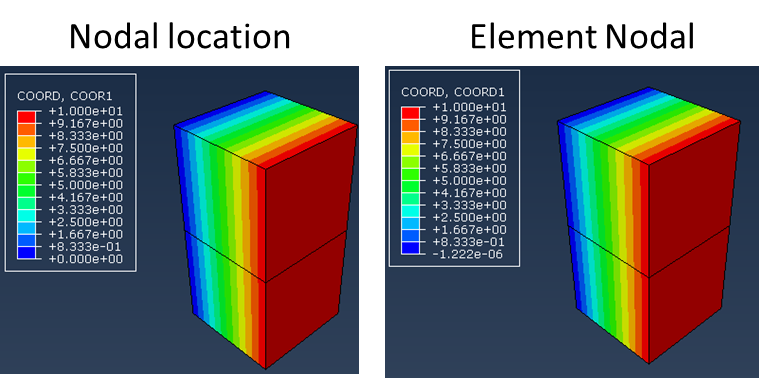


Figure . Comparison of coordinates in the undeformed state for nodal position and element nodal position. They should be the same, but data has gone from nodal, to element integration points and back to element nodal positions creating these errors.

Figure 8 compares the same components computed by Abaqus and by our implementation of the shape functions for nodal coordinates interpolated to Gauss points. Good agreement is observed, and the use of double precision increases the accuracy of the computation.

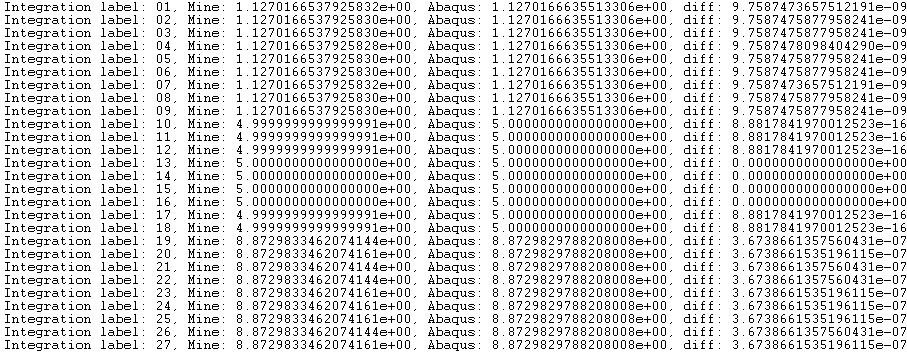


Figure 8. Comparison of coordinates interpolated from nodal coordinates to gauss point coordinates using shape functions. The actual error for our calculations is around 1e-16 since the ground truth for the integration point coordinates is 1.127016653792583 , 5.0, and 8.872983346207416.

We are also interested in mapping data at integration points to nodal positions using the shape functions. A comparison of values and errors (absolute difference from ground truth) between our implementation and Abaqus are provided in Figure 9.

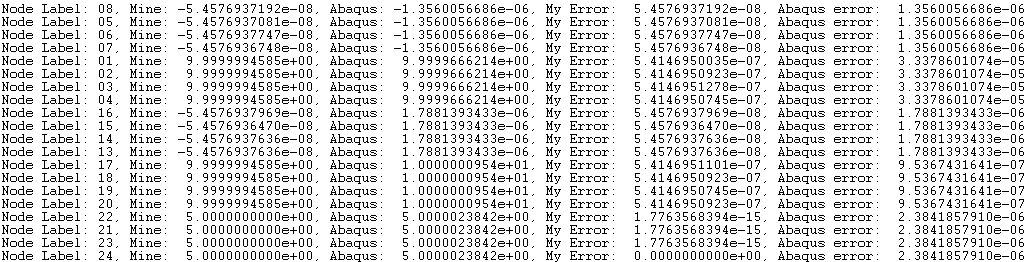


Figure 9. Comparison of extrapolation values and error from coordinates defined at nodal locations. Compared to the ground truth (coordinates at ), our implementation does similar or better than Abaqus.

Several conclusions can be drawn: 1) Nodal data probably should be mapped from nodal storage containers and not from the element nodal storage due to element nodal values apparently being computed from the integration point. Abaqus appears to enforce this by restricting availability of displacement and nodal quantities at element storage locations. 2) We can use double precision and get slightly better extrapolation/interpolation compared to Abaqus. The Abaqus interpolation/extrapolation appears to be accurate to about 7 digits.

## Validation of shape function derivatives

To validation the shape derivatives, a UMAT linear elastic material was utilized and in the UMAT the deformation gradient was exported at each integration point using SDV. Then using the nodal coordinates in the first frame and the nodal displacement values at some time, a comparison is generated for the deformation gradient. In Figure 10 is the comparison for the F11 component of the deformation gradient.

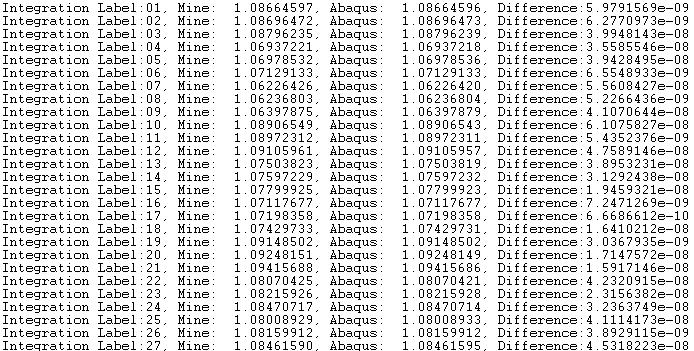


Figure . F11 component of the deformation gradient as calculated by the shape functions and by Abaqus for the UMAT subroutine

Figure . Probing collapsed element used by Abaqus for the crack tip.

## Validation of volume and surface integrals using shape functions

Performing surface integrals is verified from a simple micromechanical model in which the relationship between volume and surface integrals is given by



In  is the outward normal of the surface and  is the coordinate of stress on the surface. To perform the integral in the isoparametric space, we need to determine the face of the element on the boundary and interpolate stresses to integration points on the boundary. The integration points are chosen to be gauss points with sufficient number to integrate triquadratic element shape functions accurately and the integration is taken using and . Because of the dependence on position in the surface integral case, coordinates must be interpolated to integration points as well as stress.

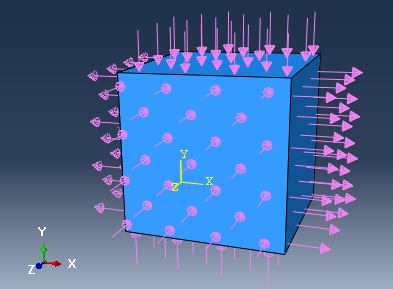


Figure 19. Picture of micromechanical model used surface integration test.

Pressures were specified on 11, 22, and 33 as pictured in Figure 19 for a fully solid, linear elastic cube of material. The results for volume and surface integrals show excellent agreement with the applied values and are summarized in Table #. These values represent the best-case scenario in which mesh is completely uniform, double precision is utilized, and exact integration is used. I believe we would have to change simulation tolerances to get lower error.

Table #. Summary of integrated values of stress.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Applied | Volume Value | Volume % error | Surface Value | Surface % error |
| S11, 1000 | 999.999961478 | 3.848319059e-06 | 1000.00014855 | 1.484025881e-05 |
| S22, -10000 | -10000.0003872 | 3.871850517e-06 | -10000.0003579 | 3.579004809e-06 |
| S33,-5000 | -4999.99993172 | 1.365267521e-06 | -5000.00000578 | 1.154860950e-07 |

# Building the element and node sets for Jm

The element and node sets needed to evaluate the J integral are built using..

Shih, C.F., Moran, B., Nakamura, T., 1986. Energy release rate along a three-dimensional crack front in a thermally stressed body. International Journal of Fracture 30, 79-102.