Solving Contact Problems with Abaqus

DS UK Ltd, Coventry - March 2013

Stephen King

Tony Richards

Seminar Abstract

Contact interactions between different parts play a key role when simulating bolted assemblies, manufacturing processes, dynamic impact events, and various other systems. Accurately capturing these interactions is essential for solving many engineering problems. SIMULIA has developed state-of-the-art contact modeling capabilities in Abaqus.

Attend this seminar to learn the latest techniques and strategies for solving difficult contact problems with Abaqus. This seminar primarily focuses on Abaqus/Standard, with additional discussion of Abaqus/Explicit.

Topics include advantages of the general contact capability, accurate contact pressures, insight on numerical methods, tips for improving convergence, recent enhancements to the implicit dynamics procedure for contact models, and proper representation of physical details associated with contact.

Lectures

- Lecture 1: Introduction
- Lecture 2: Defining Contact in an Analysis
- Lecture 3: Numerical Methods for Contact
- Lecture 4: Contact Output and Diagnostics Tools (start)

(Lunch) 12:30pm – 1.30pm

- Lecture 4 (cont.): Contact Output and Diagnostics Tools (finish)
- Lecture 5: Convergence Topics
- Lecture 6: Contact in Abaqus/Explicit
- Lecture 7: More Features

Legal Notices

Printed in the U. S. A.

Introduction

Lecture 1

SIMULIA

Overview

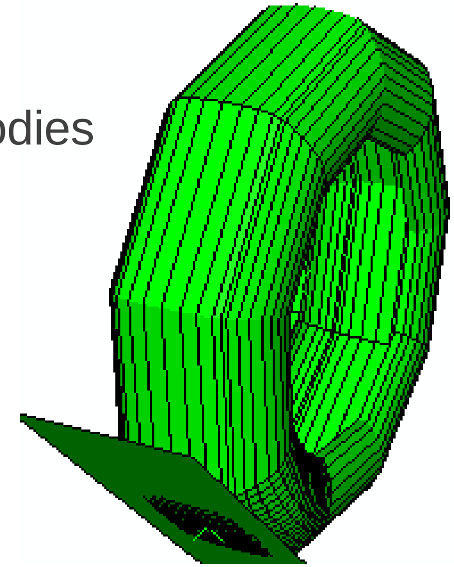
- General Considerations
- Evolution of Contact in Abaqus
- Contact Examples

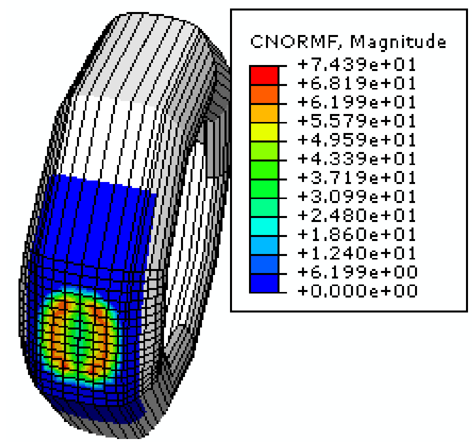
General Considerations

- **What is contact?**
  - Physically, contact involves interactions between bodies
    - Contact pressure resists penetration
    - Frictional stress resists sliding
    - Electrical, thermal interactions
  - Numerically, contact includes *severe nonlinearities*
    - Inequality conditions result in discontinuous "stiffness"
      - Gap distance: $d_{gap} \geq 0$
      - Frictional stress: $\tau \leq \mu p$
    - Conductance properties suddenly change when contact is established

Fairly intuitive

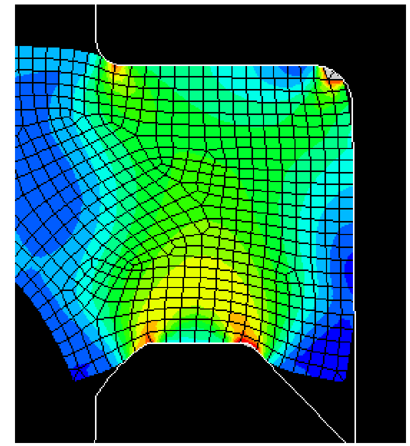
Numerically challenging

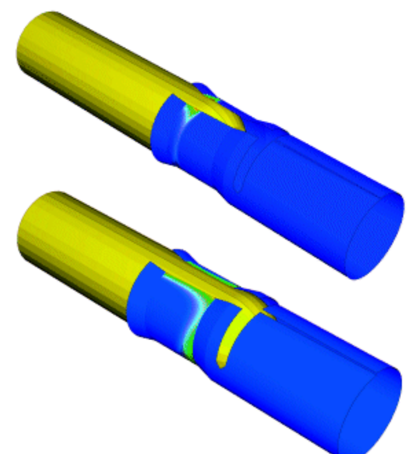




General Considerations

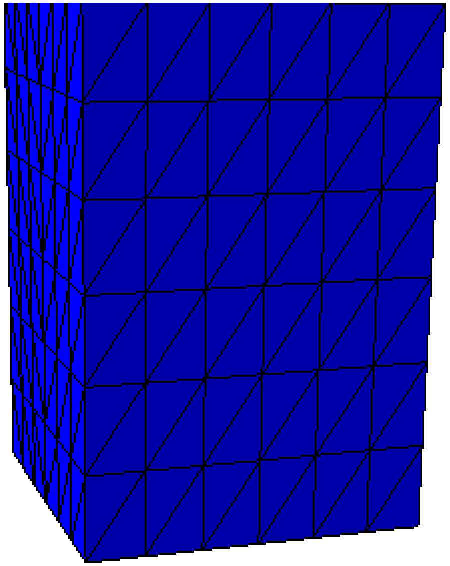
- **Various classifications of contact interactions can be considered**
  - Example: slender or bulky components
    - Bulky components:
      - Typically many nodes in contact at one time
      - Contact causes local deformation and shear, but it causes little bending
    - Slender components
      - Often relatively few nodes in contact at one time
      - Contact causes bending
      - Often more challenging





General Considerations

- **Classifications of contact interactions:**
  - Slender or bulky components
  - Deformable or rigid surfaces
  - Degree of confinement and compressibility of components
  - Two-body contact or self-contact
  - Amount of relative motion (small or finite sliding)
  - Amount of deformation
  - Underlying element type (1st or 2nd order)
  - Interaction properties (friction, thermal, etc.)
  - Which results are of interest and importance (e.g. contact stresses)



‘Ingredients’ of a Contact Model

- **Contact surfaces**
  - Surfaces over bodies that may experience contact
- **Contact interactions**
  - Which surfaces interact with one another?
- **Surface property assignments**
  - For example, contact thickness of a shell
- **Contact property models**
  - Examples: pressure vs. overclosure relationship, friction coefficient, conduction coefficients, etc.
- **Contact formulation aspects**
  - For example, can a small-sliding formulation be used?
- **Algorithmic contact controls**
  - Such as contact stabilization settings

Many of these aspect need not be explicitly specified

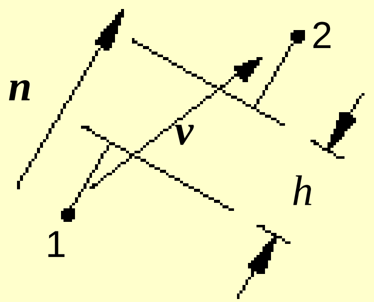
General Considerations

- **Physical and numerical aspects of contact modeling:**
  - User responsible for defining physical aspects of model
  - User and Abaqus control various numerical aspects
    - Many details (e.g., slender or bulky classification) need not be explicitly specified
    - Trend toward greater automation

Evolution of Contact Modeling in Abaqus

Evolution of Contact Modeling

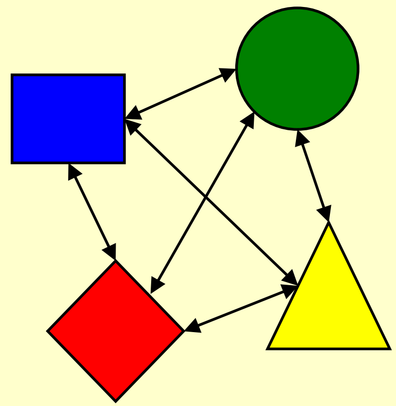
Contact elements (e.g., GAPUNI):



$$h = d + \boldsymbol{n} \cdot \left(\boldsymbol{u}^2 - \boldsymbol{u}^1 \right) \geq 0$$

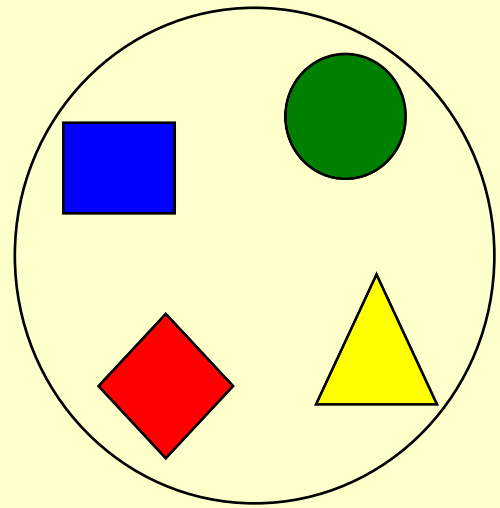
User-defined element for each contact constraint

Contact pairs:



Many pairings for assemblies

General contact:

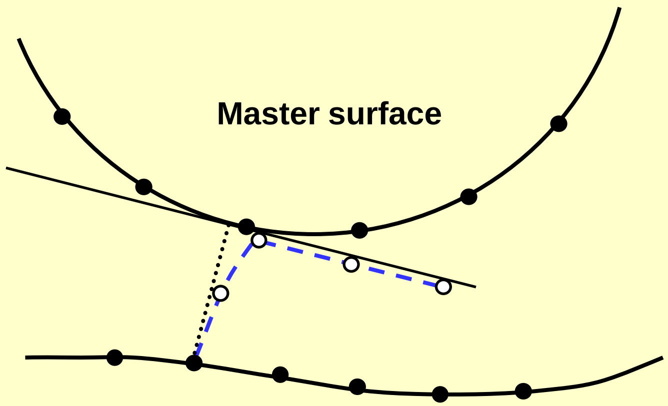


Model all interactions between free surfaces

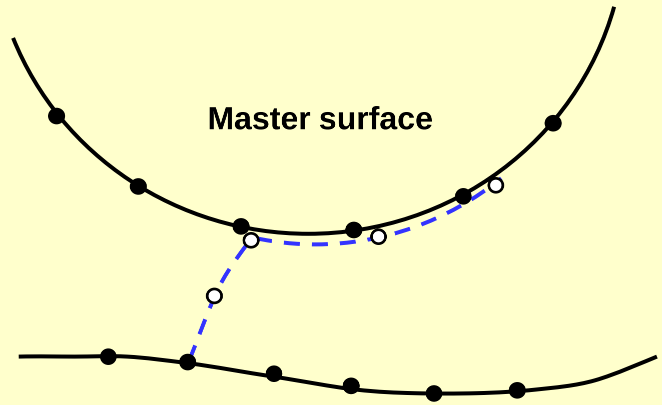
Trends over time

Evolution of Contact Modeling

Flat approximation of master surface per slave node:



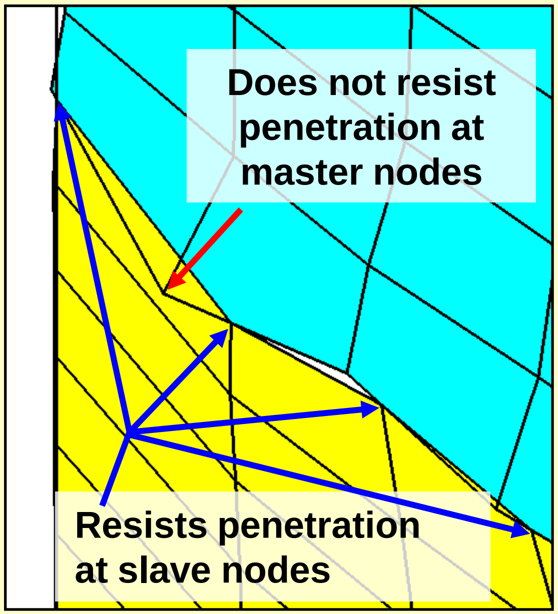
Realistic representation of master surface:



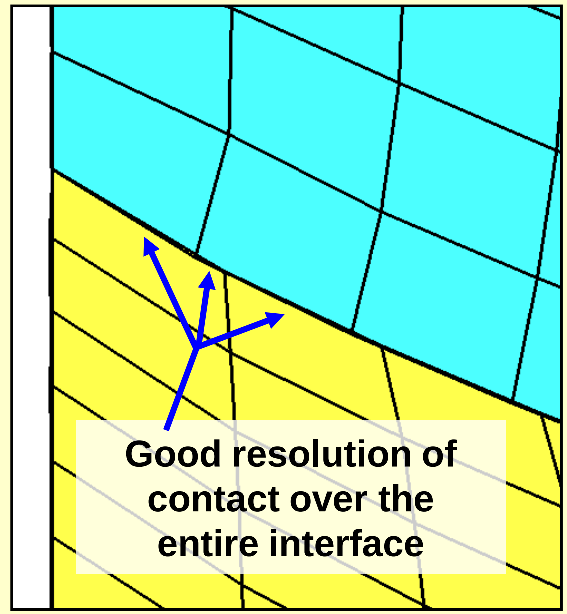
Trends over time

Evolution of Contact Modeling

Slave surface treated as collection of discrete points:



Constraints based on integrals over slave surface:



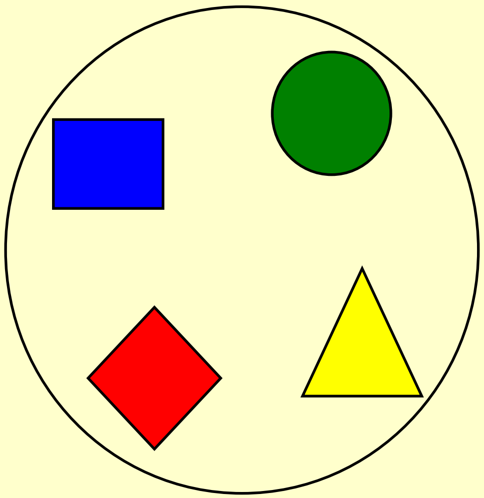
Trends over time

SIMULIA

Evolution of Contact Modeling

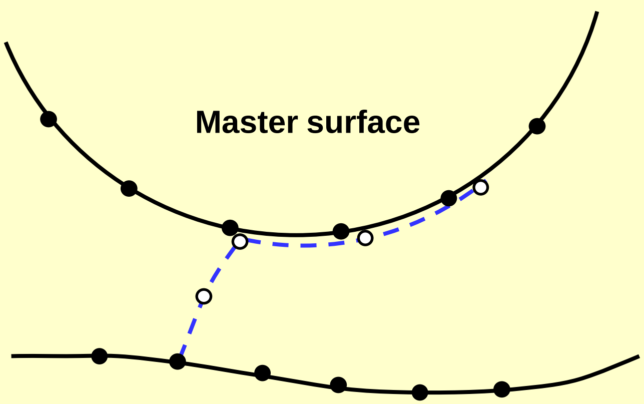
- **Goals: improve usability, accuracy, and performance**
  - More focus by user on physical aspects
    - Less on idiosyncrasies of numerical algorithms
  - Broad applicability
  - Large models (assemblies)

General contact:

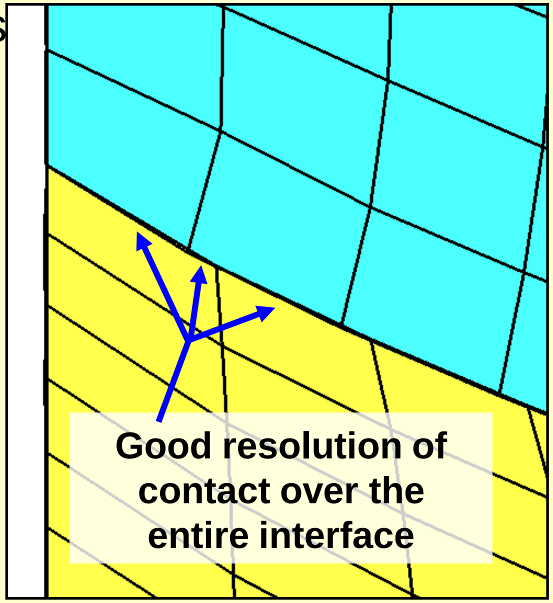


Model all interactions between free surfaces

Realistic representation of master surface:

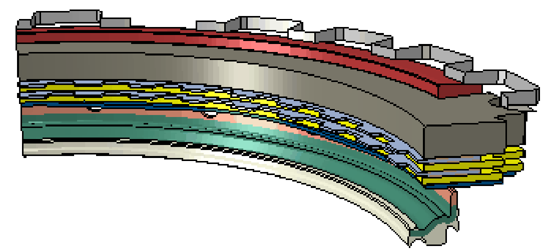


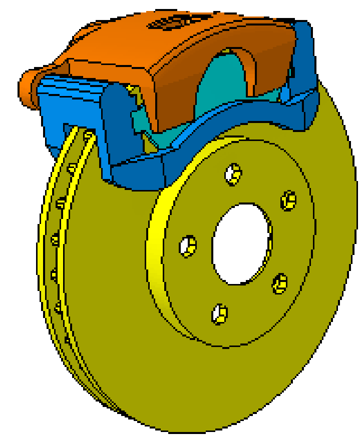
Constraints based on integrals over slave s

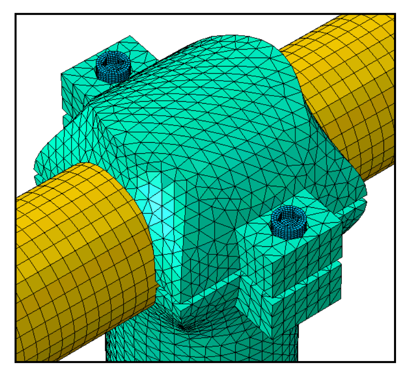


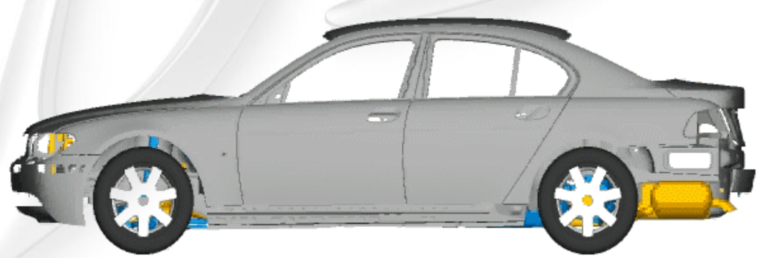
Evolution of Contact Modeling

- **General contact algorithm**
  - Contact domain spans multiple bodies (both rigid and deformable)
    - Default domain defined **automatically** via all-inclusive, element-based surface
  - Method geared toward models with multiple components and complex topology
    - Greater ease in defining contact model
- **Available in Abaqus/Explicit since 6.3**
- **Available in Abaqus/Standard since 6.8-EF**









Evolution of Contact Modeling

- **Transition to general contact nearly complete for Abaqus/Explicit**
  - Most Abaqus/Explicit analyses use general contact
  - Easy to use and robust
  - Accuracy, performance, and scalability as good or better than contact pairs
  - Some features available only in general contact
  - A few features available only with contact pairs

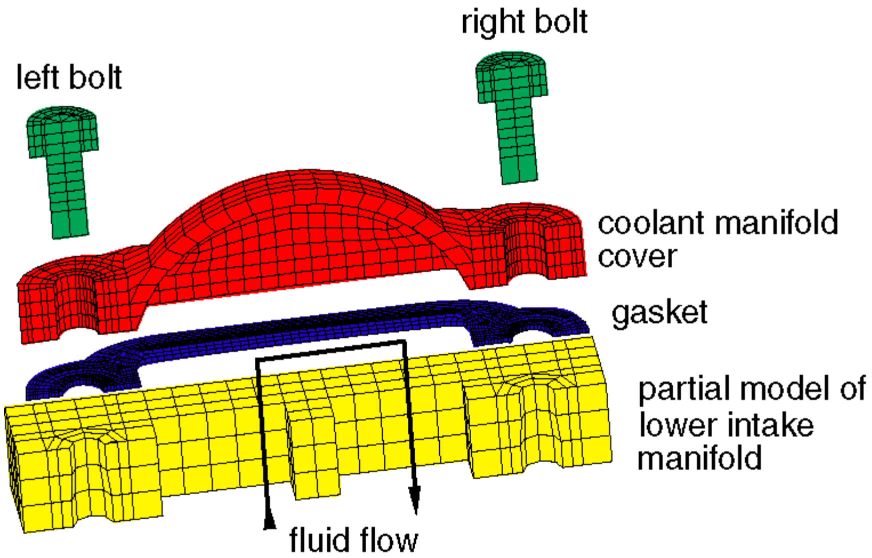
Evolution of Contact Modeling

- **Transitioning to general contact in Abaqus/Standard**
  - Good feedback
  - Easier to create model than contact pairs
  - Similar robustness and accuracy as contact pairs
  - Some extra contact tracking time, etc.
  - Contact pairs are required to access specific features not yet available with general contact
    - Analytical rigid surfaces
    - Node-based surfaces or surfaces on 3-D beams
    - Small-sliding formulation
    - See the Abaqus Analysis User's Manual
- **General contact and contact pairs can be used together**
  - General contact algorithm automatically avoids processing interactions treated with contact pairs

Contact Examples

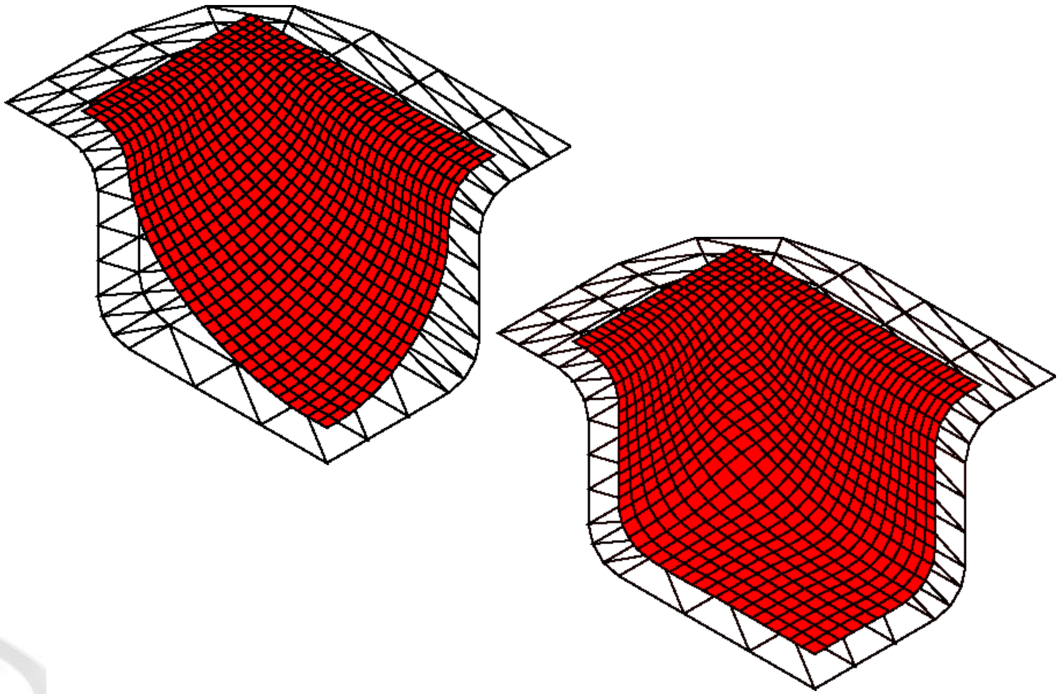
Contact Examples

- **Contact between linear elastic bodies with small relative motion**
  - Common design problems involving:
    - Small relative motion
    - Significant contact area
  - Typical examples:
    - Bearing design
    - Hard gaskets
    - Interference fits
  - Fretting (surface wear) is often a concern, requiring accurate resolution of contact stresses and stick/slip zones



Contact Examples

- **Deformable-to-rigid contact**
  - Finite sliding between surfaces (large displacements)
  - Finite strain of deforming components
  - Typical examples:
    - Rubber seals
    - Tire on road
    - Pipeline on seabed
    - Forming simulations (rigid die/mold, deformable component)

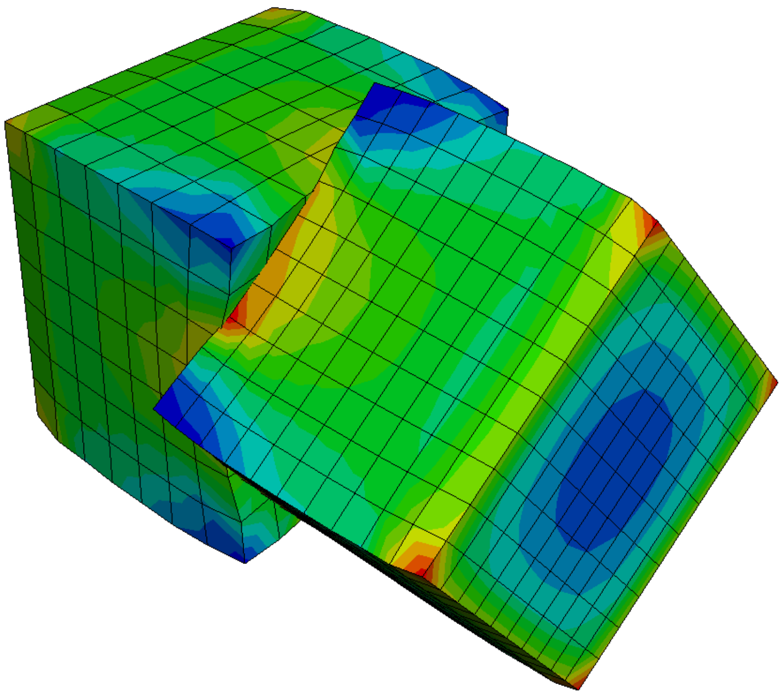


Example: metal forming simulation

Example taken from “Superplastic forming of a rectangular box,” Section 1.3.2 in the Abaqus Example Problems Manual

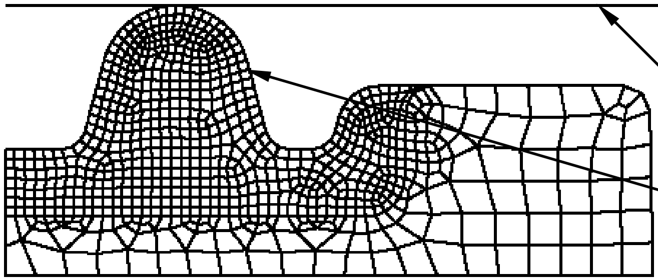
Contact Examples

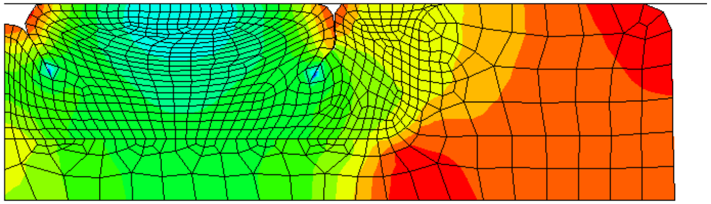
- **Finite-sliding contact between deformable bodies**
  - Most general category of contact
  - Example: twisting blocks
    - Press together and relative rotation of 90°



Contact Examples

- **Self-contact**
  - Type of finite-sliding, deformable-to-deformable contact
  - Contact of a single body with itself—often involves severe deformation
  - Sometimes adds CPU expense and numerical difficulty
  - General contact implementation somewhat like self-contact of surface spanning multiple bodies





Contour of minimum principal stress

Example: compression of a rubber gasket

Example taken from "Self-contact in rubber/foam components: rubber gasket," Example Problem 1.1.18 in the Abaqus Example Problems Manual

Lecture 1 Summary

Review of Topics Discussed in Lecture

- General Considerations
- Evolution of Contact in Abaqus
- Contact Examples

Defining Contact

Lecture 2

Overview

- Defining Surfaces
- Defining Contact Pairs
- Defining General Contact
- Representation of Curved Surfaces

Defining Surfaces

Surfaces

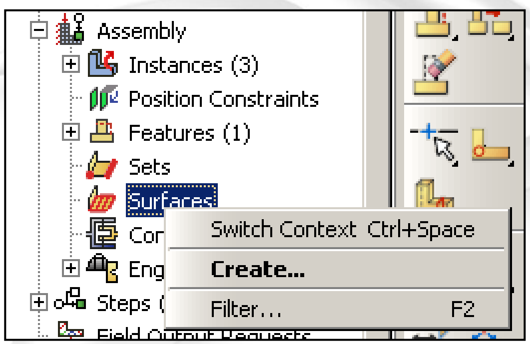
- **Various Abaqus features use surfaces**
  - Contact
  - Tie constraints
  - Surface loads
  - Cavity radiation
  - Bolt pre-tensioning
- **Various surface types exist in Abaqus**
  - Element-based (most common)
  - Node-based
  - Analytical rigid
  - Eulerian (not covering coupled Eulerian-Lagrangian analysis in this seminar)
- **Surface documentation**
  - Sections 2.3.1–2.3.6 of Abaqus Analysis User's Manual

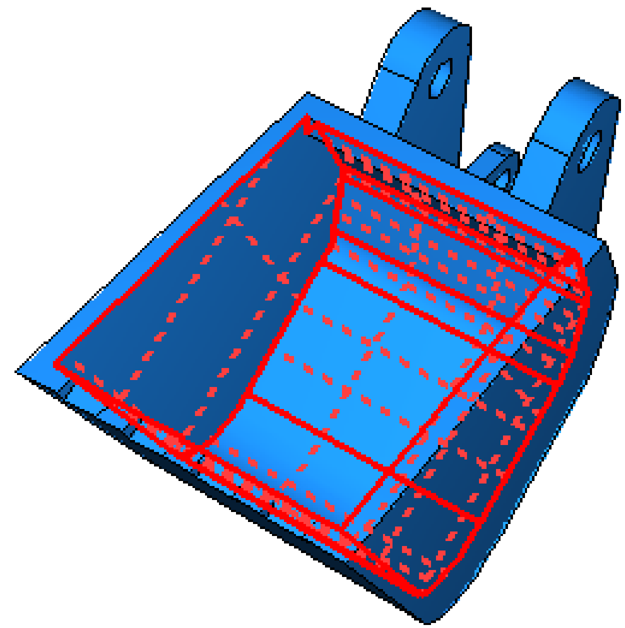
Surfaces

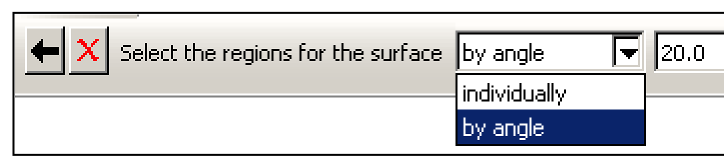
- **Abaqus/CAE interface**

  Solid bodies

  - Surface on solid defined by selecting appropriate region of exterior of the part
  - Regions can be selected individually or based on face angles

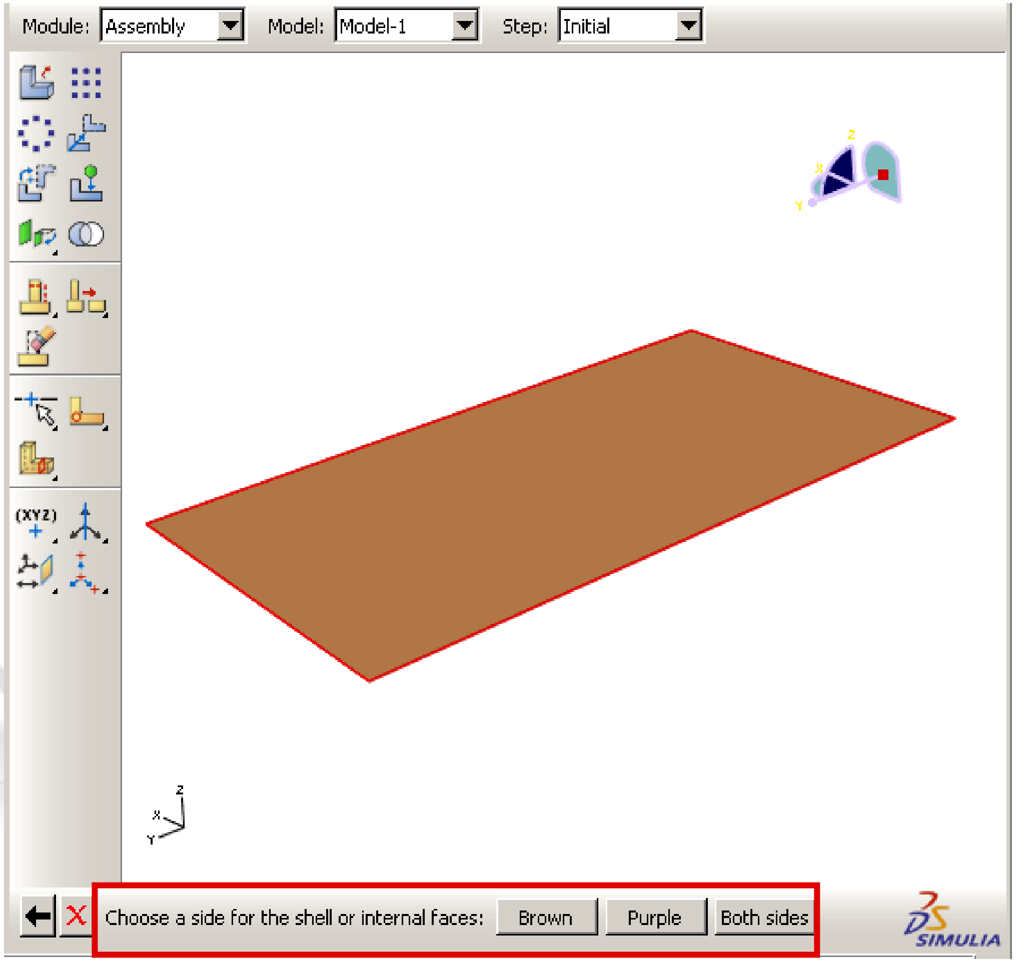






Surfaces

- **Abaqus/CAE interface**

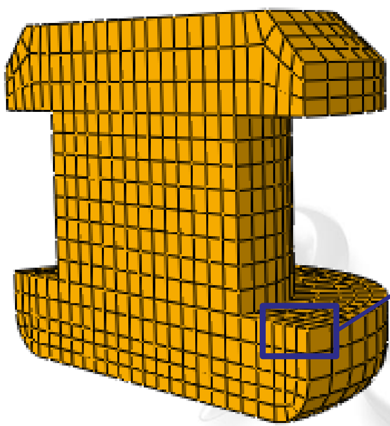


Shell-like surfaces may be:

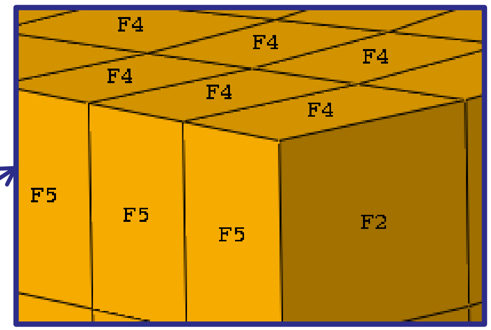
- On "positive" side of elements
- On "negative" side of elements
- Or, on both sides

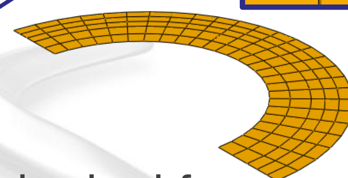
Surfaces

- **Element-based surfaces are composed of element faces**
  - Typically, on exposed faces of bodies

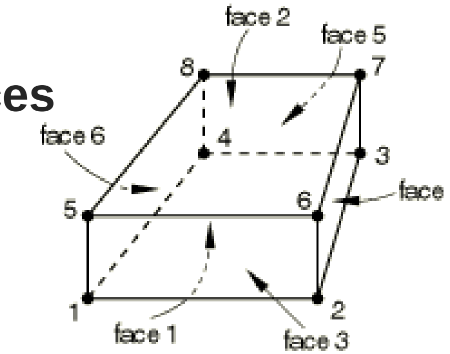


Close-up view with local face ID labels

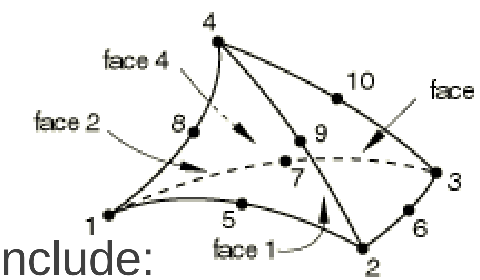




Example of surface defined over a portion of rivet



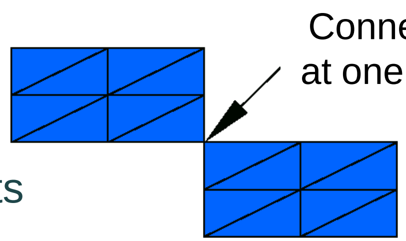
Local numbering conventions for brick and tet elements



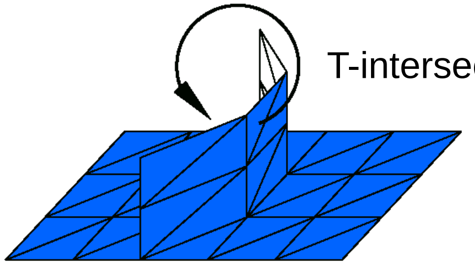
- Characteristics inherited from underlying elements include:
  - Deformable or rigid
  - Shell/membrane thickness
    - Some contact formulations account for this thickness
  - Representative stiffness
    - Influences some numerical aspects, such as penalty stiffness

Surface Restrictions

- **Mostly context-specific**
  - Depend on which features use the surface
- **Restrictions on surfaces used in contact definitions**
  - Depend on details of contact definition
    - Documented in Abaqus Analysis User's Manual
  - Trend toward fewer surface restrictions
    - Example: master surface connectivity requirements



Contact formulation	Discontinuous (or 3-D faces joined at only one node)	T-intersection (more than two faces per edge)
Finite-sliding, node-to-surface	Not allowed	Not allowed
Finite-sliding, surface-to-surface	Allowed	Allowed

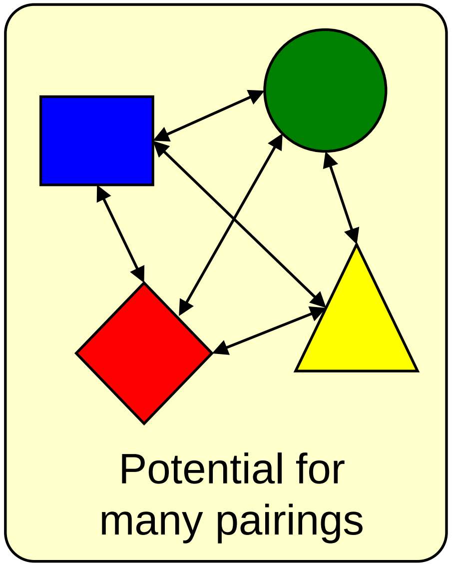


- **Example of a general restriction on element-based surfaces**
  - Parent elements cannot be a mixture of two-dimensional, axisymmetric, and three-dimensional elements

Defining Contact Pairs

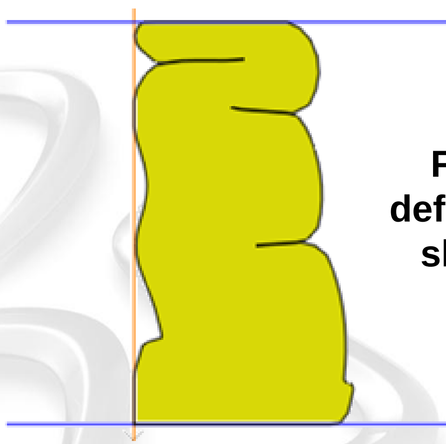
Defining Contact Pairs

- **Features of contact pairs defined by user:**
  - What constitutes each surface
  - Which pairs of surfaces will interact
  - Which surface is the master and which is the slave
  - Which surface interaction properties are relevant (e.g., friction)

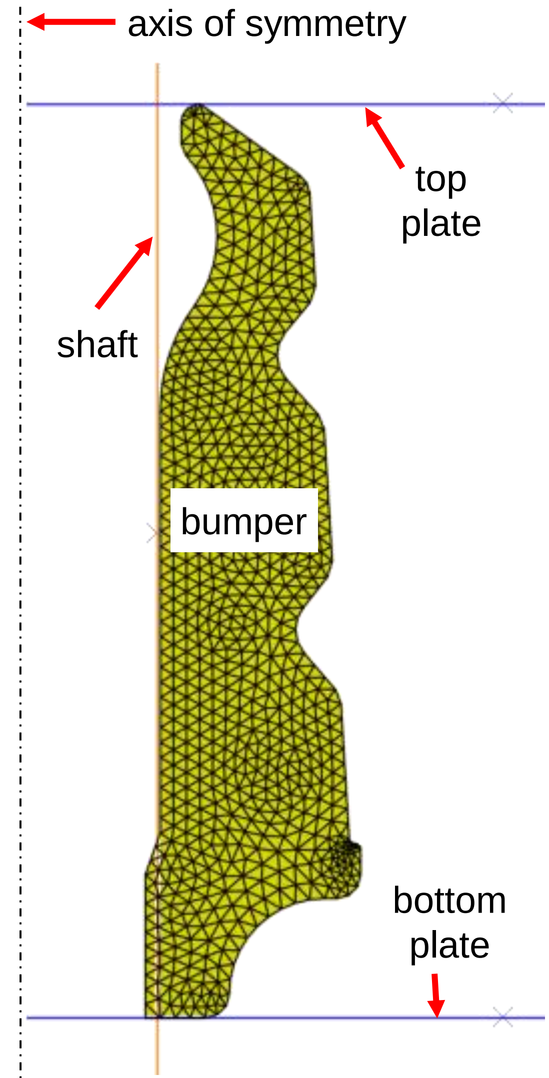


Defining Contact Pairs

- **Example: analysis of a jounce bumper**
  - Highly compressible component used in a vehicle's shock isolation system
    - Bumper folds as it is compressed, so self-contact is modeled



Final deformed shape



  - Analysis consists of two steps:

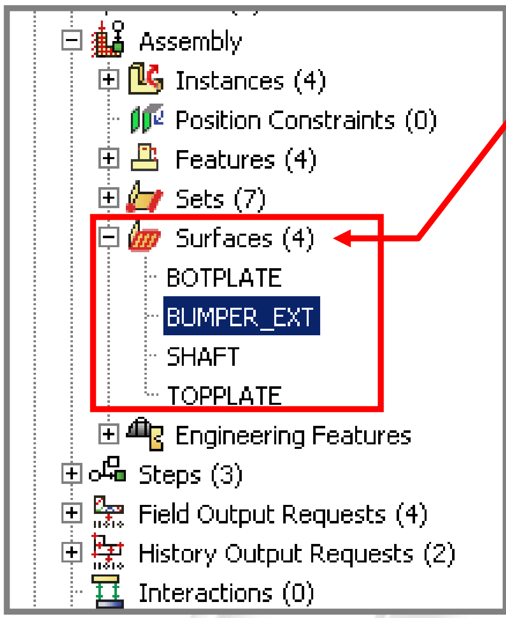
    Step 1 Resolve interference fit

    Step 2 Move the bottom plate up to compress the bumper

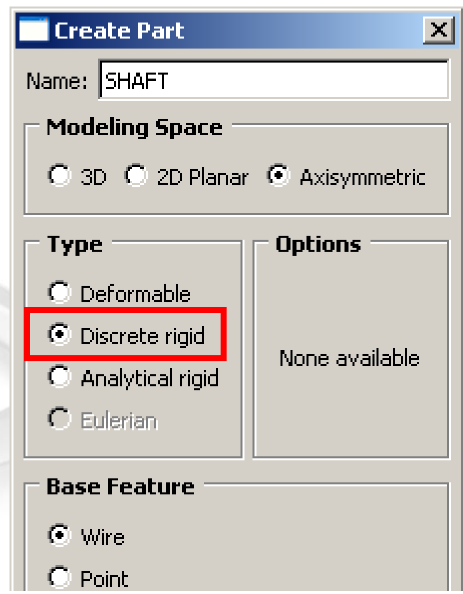
Defining Contact Pairs

1 Define surfaces (using Abaqus/CAE)

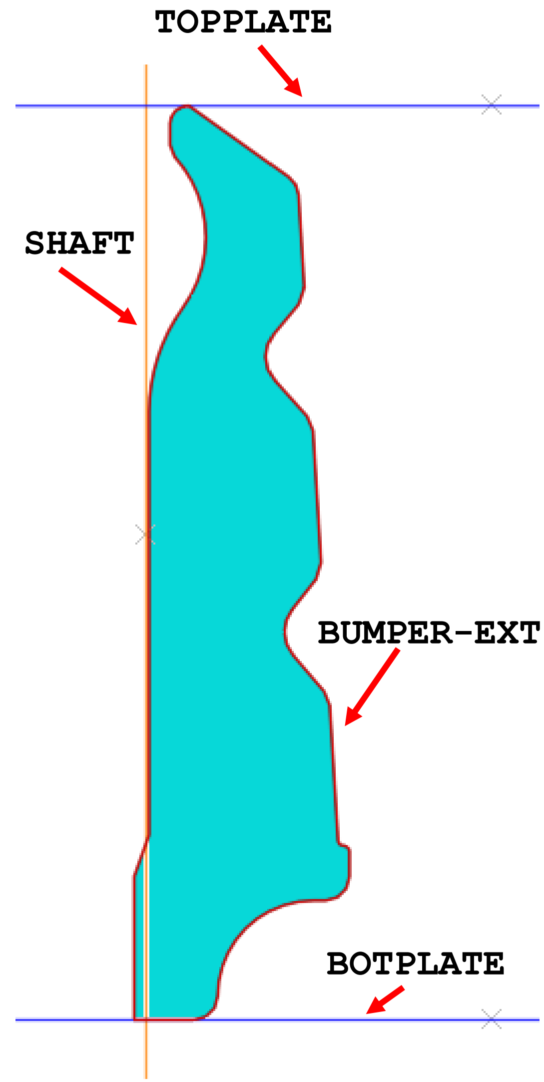
Double-click **Surfaces** to create a new surface



Model Tree



Create discrete rigid part



Defining Contact Pairs

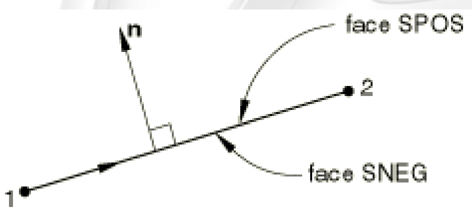
1 Define surfaces (using keywords)

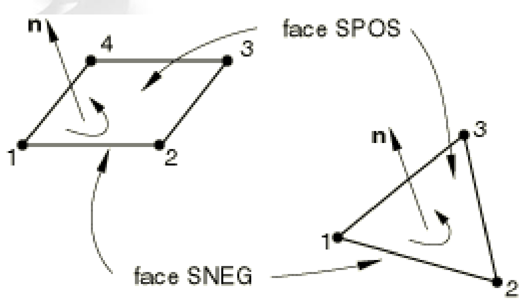
- Automatic free surface generation on bumper elements:

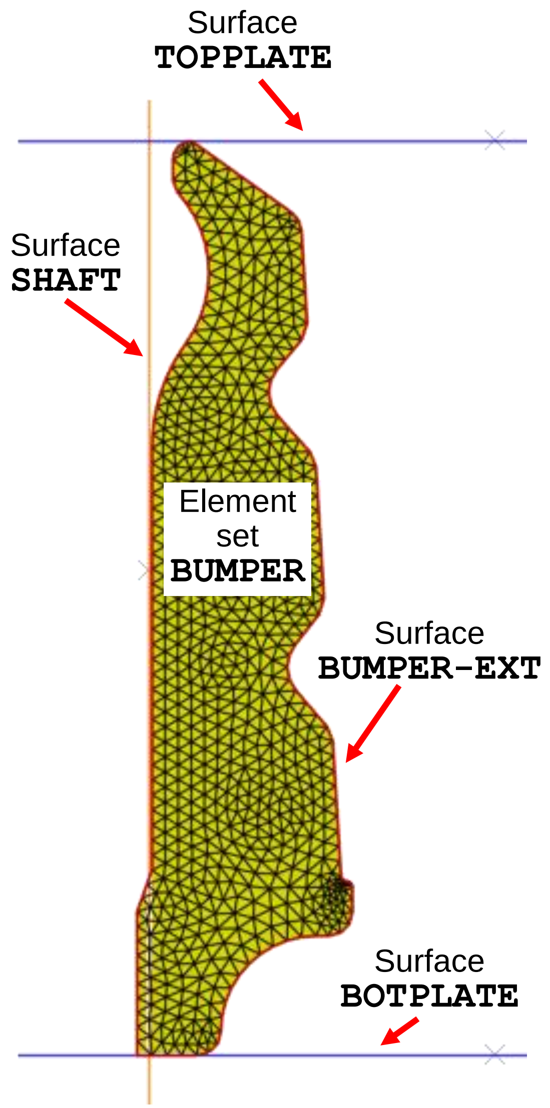
```
*SURFACE,NAME=BUMPER-EXT
 BUMPER,
```

- Discrete rigid surfaces:

```
*RIGID BODY, ELSET=BOTDIE, REF NODE=BOTRP
*SURFACE,NAME=BOTPLATE
 BOTDIE, SPOS
*RIGID BODY, ELSET=TOPDIE, REF NODE=TOPRP
*SURFACE, NAME=TOPPLATE
 TOPDIE, SPOS
*RIGID BODY, ELSET=SHAFTDIE, REF NODE=SHAFTRP
*SURFACE, NAME=SHAFT
 SHAFTDIE, SPOS
```





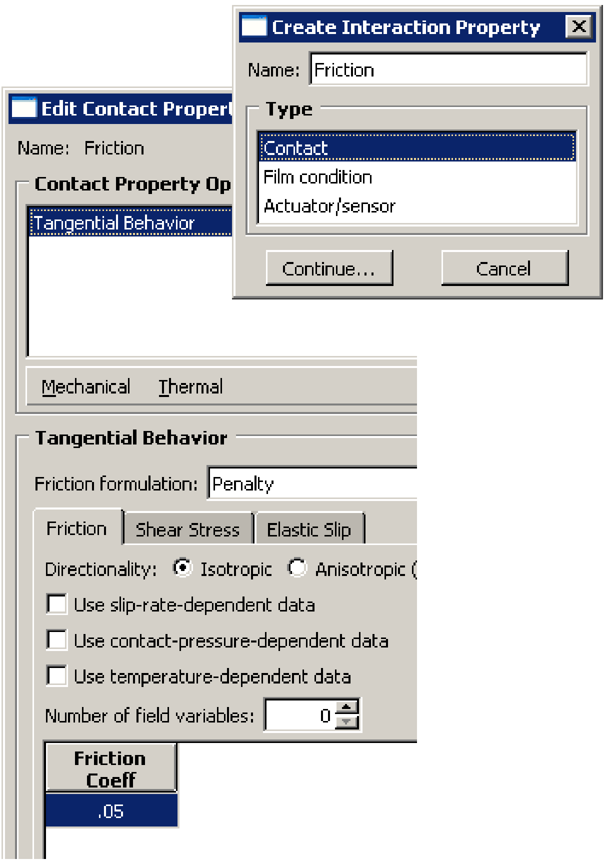


Defining Contact Pairs

2 Define contact properties

- Contact property definitions are the same for general contact and contact pairs
- Contact properties can include:
  - Friction
  - Contact damping
  - Pressure-overclosure relationships
- All contact pairs use the same interaction property in this example:

```
*SURFACE INTERACTION, NAME=Friction
*FRICTION
0.05,
```



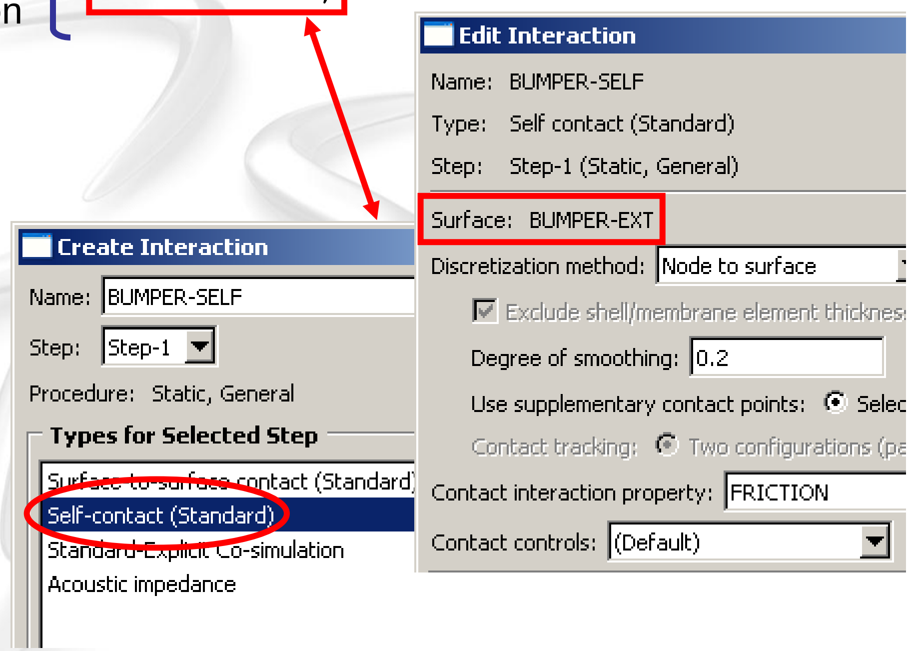
Defining Contact Pairs

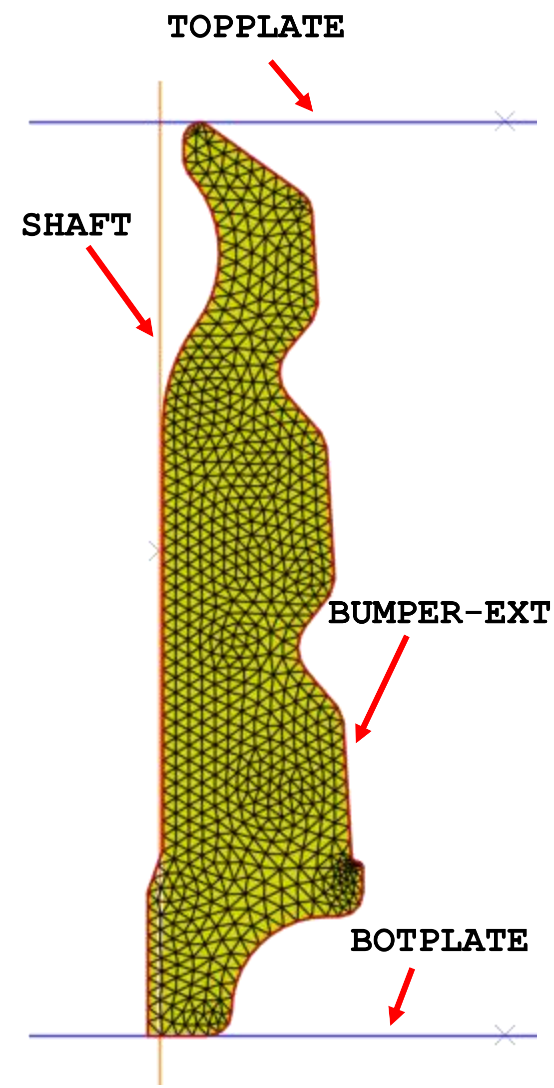
3 Define contact pairs

- Contact pair definition required for each pair of surfaces that can interact
    - Bumper self-contact:

include inside step definition

```
*CONTACT PAIR, INTERACTION=Friction
BUMPER-EXT,
```



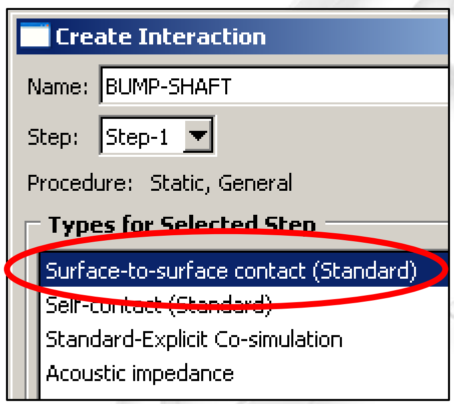


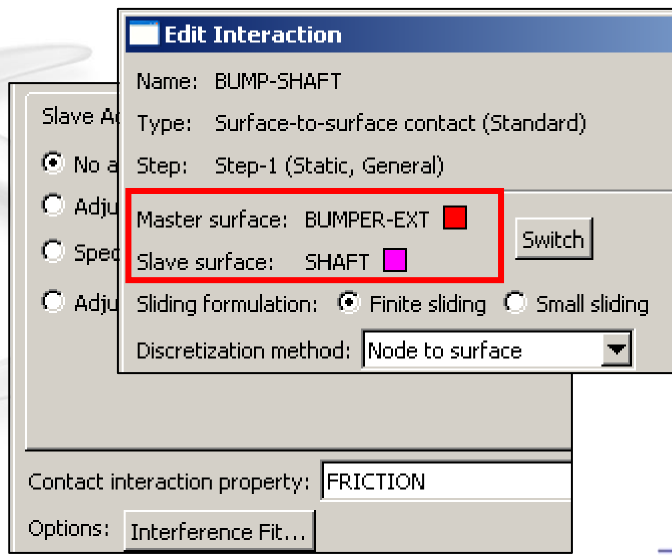
Defining Contact Pairs

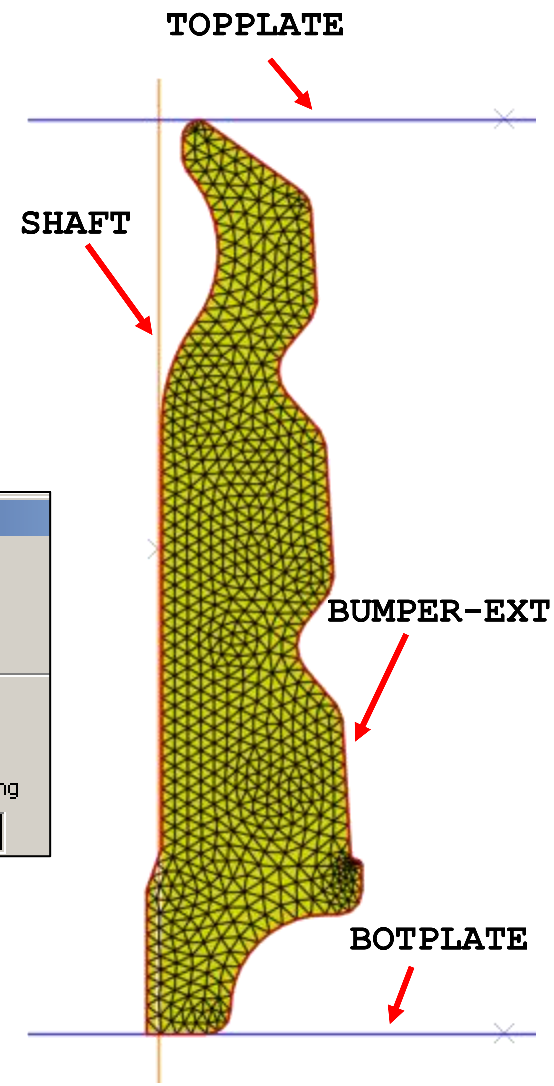
3 Define contact pairs

- Contact between the bumper and the rigid bodies:

```
*CONTACT PAIR, INTERACTION=Friction
 BUMPER-EXT, TOPPLATE
 BUMPER-EXT, BOTPLATE
 BUMPER-EXT, SHAFT
```

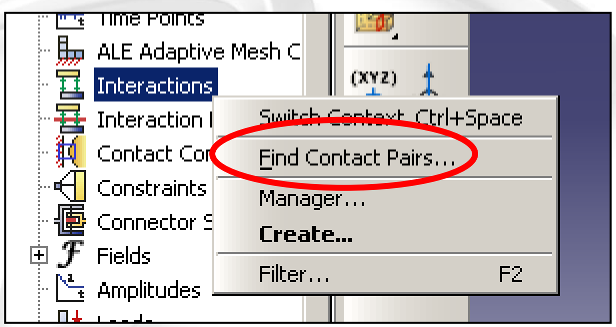


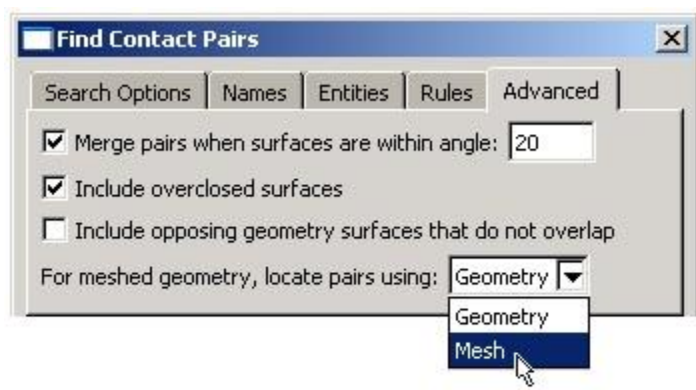




Defining Contact Pairs

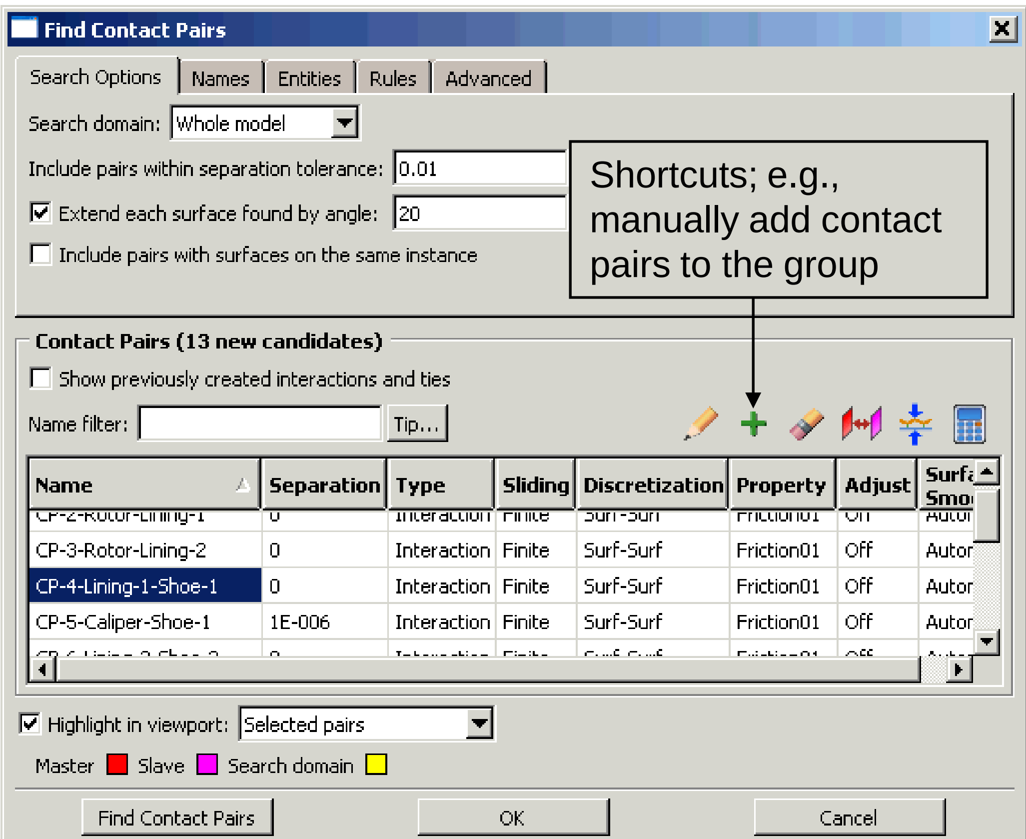
- **Automatic contact pair detection in Abaqus/CAE**
  - Automatic contact detection is a fast and easy way to define contact pairs and tie constraints in a three-dimensional model
  - Instead of individually selecting surfaces and defining the interactions between them, you can instruct Abaqus/CAE to locate automatically all surfaces in a model that are likely to interact ***based on initial proximity***
  - Can be used to define contact with shells, membranes, and solids
    - Including shell offset
    - Native or orphan mesh parts

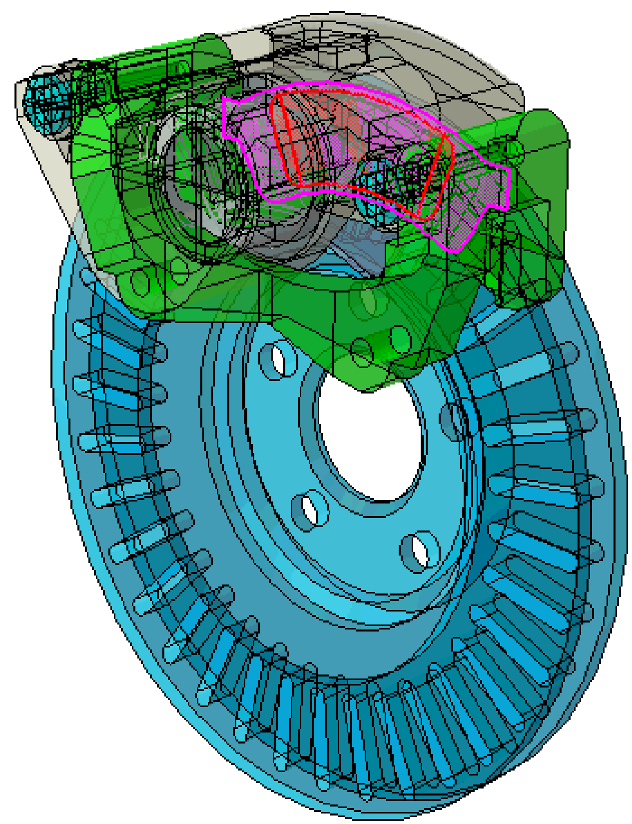




Defining Contact Pairs

- **Automatic contact pair detection in Abaqus/CAE**
  - Example: Disk brake
  - Tabular display of candidate contact pairs is provided
  - Various controls over selection criteria, etc.





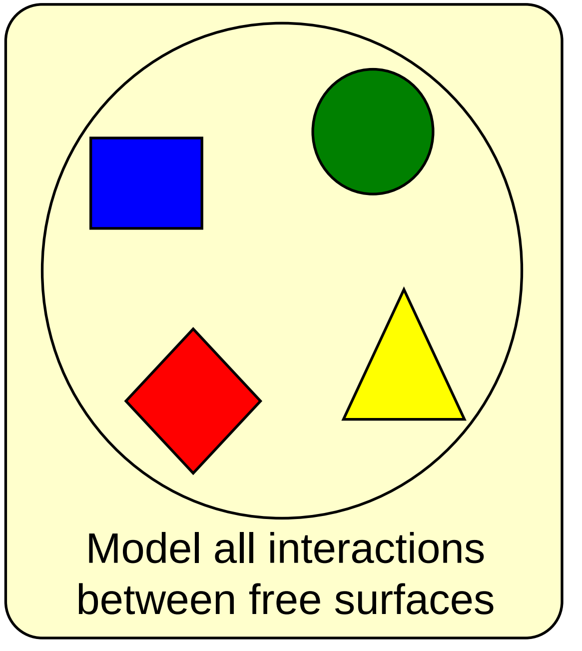
Defining General Contact

SIMULIA

Defining General Contact

- **General contact user interface allows for concise contact definition reflecting the physical description of the problem**
    - Contact definition can be expanded in complexity, as needed
    - Independent specification of contact interaction domain, contact properties, and surface attributes permitted
    - Minimal algorithmic controls required
- **General contact user interface is very similar for Abaqus/Explicit and Abaqus/Standard analyses**

Typical usage of general contact:



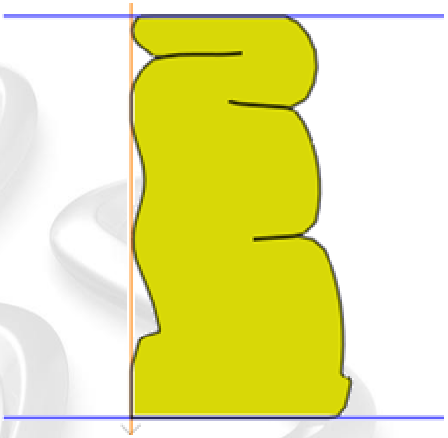
Defining General Contact

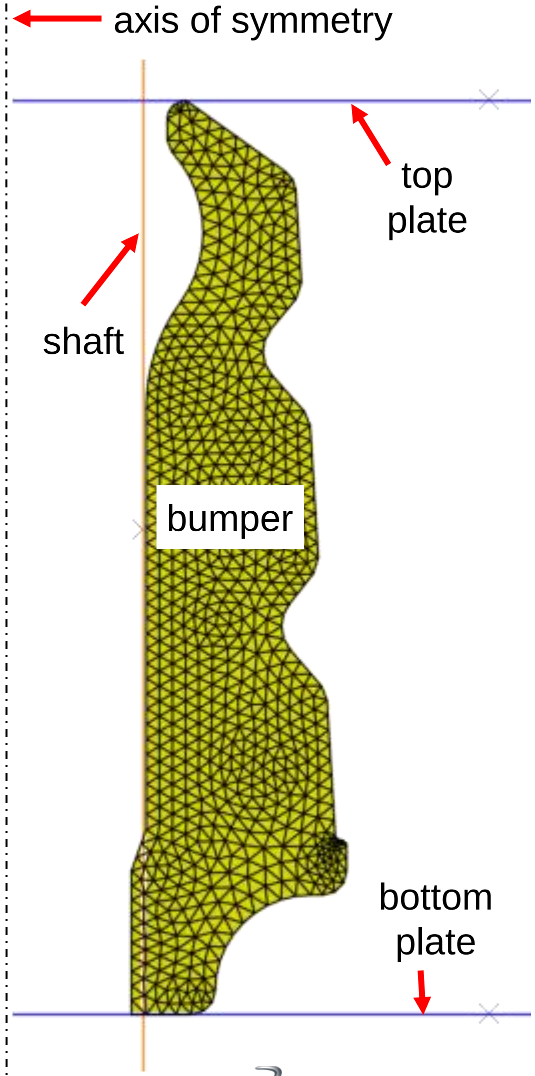
- **Examples of differences between general contact in Abaqus/Explicit and Abaqus/Standard**

Characteristic	Abaqus/Explicit	Abaqus/Standard
Primary formulation	Node-to-surface	Surface-to-surface
Master-slave roles	Balanced master-slave	Pure master-slave
Secondary formulation	Edge-to-edge	Edge-to-surface
2-D and axisymmetric	Not available	Available
Most aspects of contact definition	Step-dependent	Model data

Defining General Contact

- **Jounce bumper example using general contact**
  - Recall initial and final configurations (shown here)





Defining General Contact

- **Contact definition**

1) Begin the general contact definition

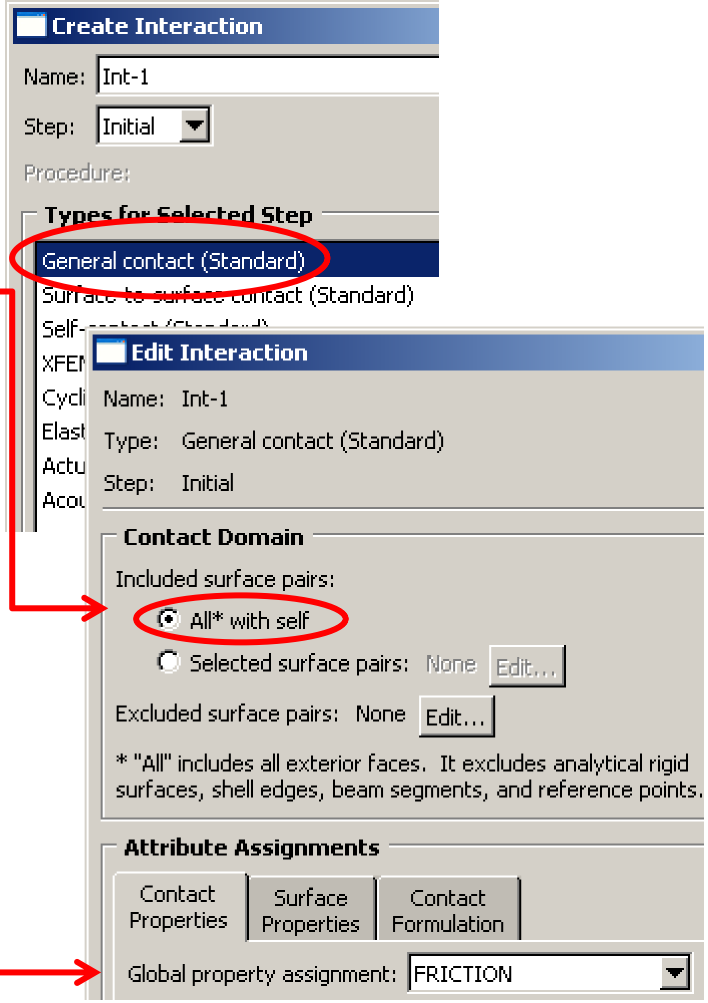
```
*Contact
*Contact Inclusions, ALL EXTERIOR
```

2) Specify “automatic” contact for the entire model

3) Assign global contact properties

```
*Contact Property Assignment
 ,   , FRICTION
```

Simple!

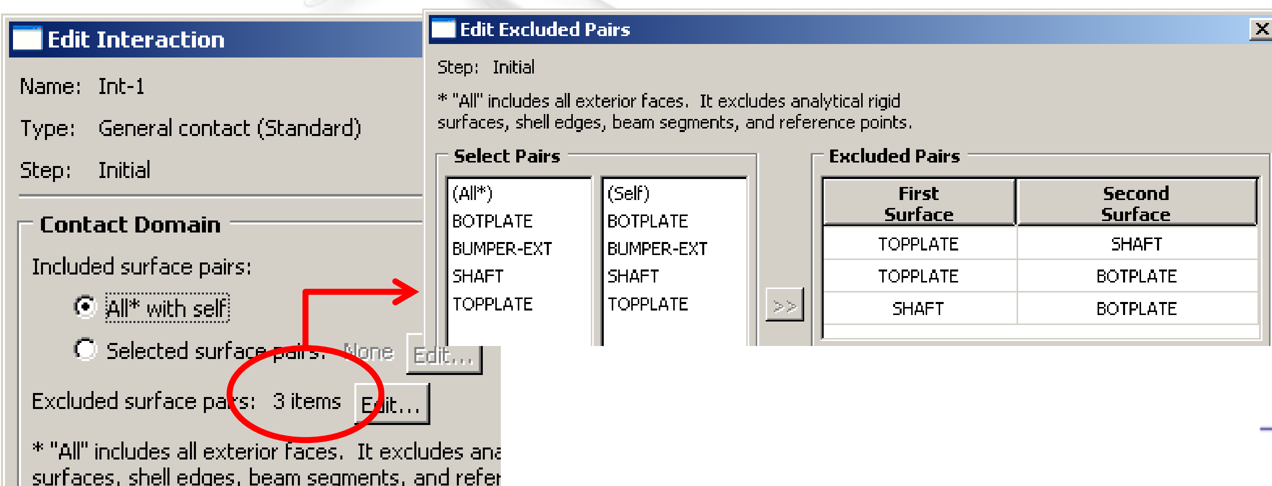


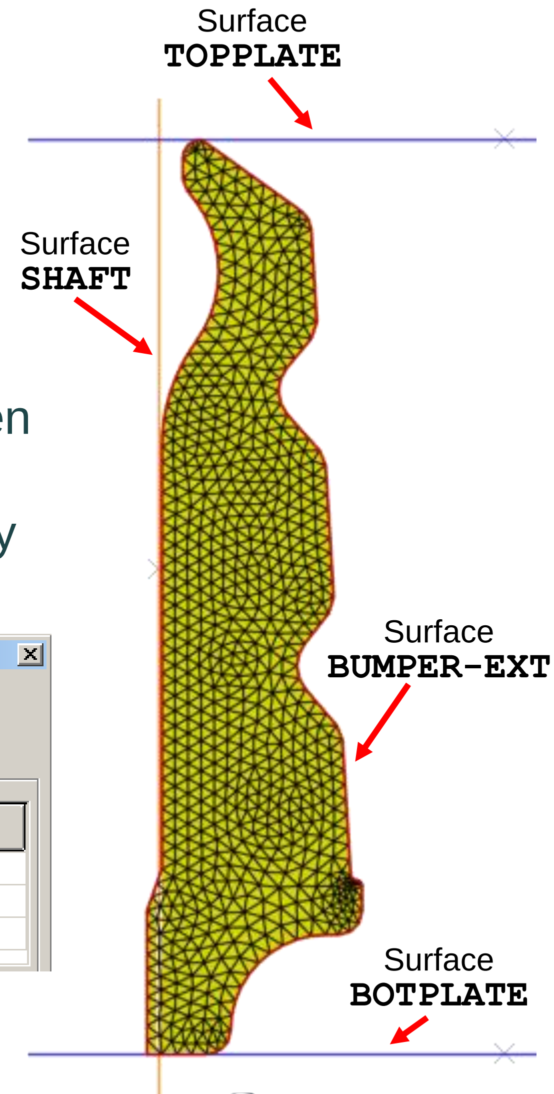
Defining General Contact

- **The contact definition can gradually become more detailed, as called for by the analysis**
  - Global/local friction coefficients and other contact properties can be defined
  - Pair-wise specification of contact domain (instead of ALL EXTERIOR) allowed
    - Contact inclusions and contact exclusions
  - User control of contact thickness (especially for shells) is provided
    - Surface properties
  - Contact initialization (initial adjustments, interference fits, etc.)

Defining General Contact

- **Fine-tuning contact domain**
  - General contact domain can be modified by including and/or excluding predefined surfaces
  - For example, exclude consideration of contact between rigid surfaces in this example
    - Not essential for this analysis (overlap between perpendicular surfaces not resolved with the surface-to-surface contact formulation used by general contact)





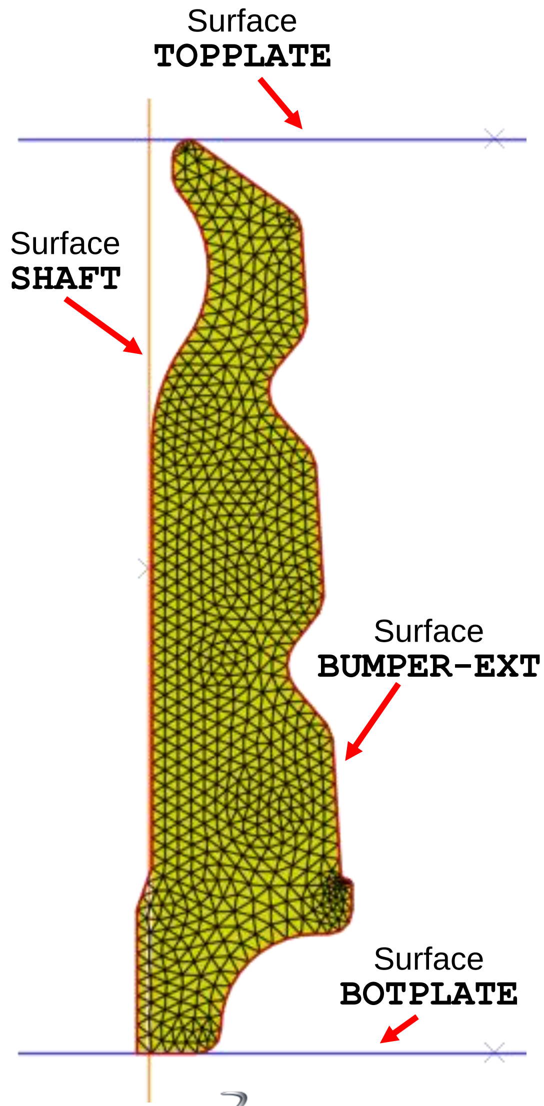
SIMULIA

Defining General Contact

- **Keyword interface for contact exclusions:**

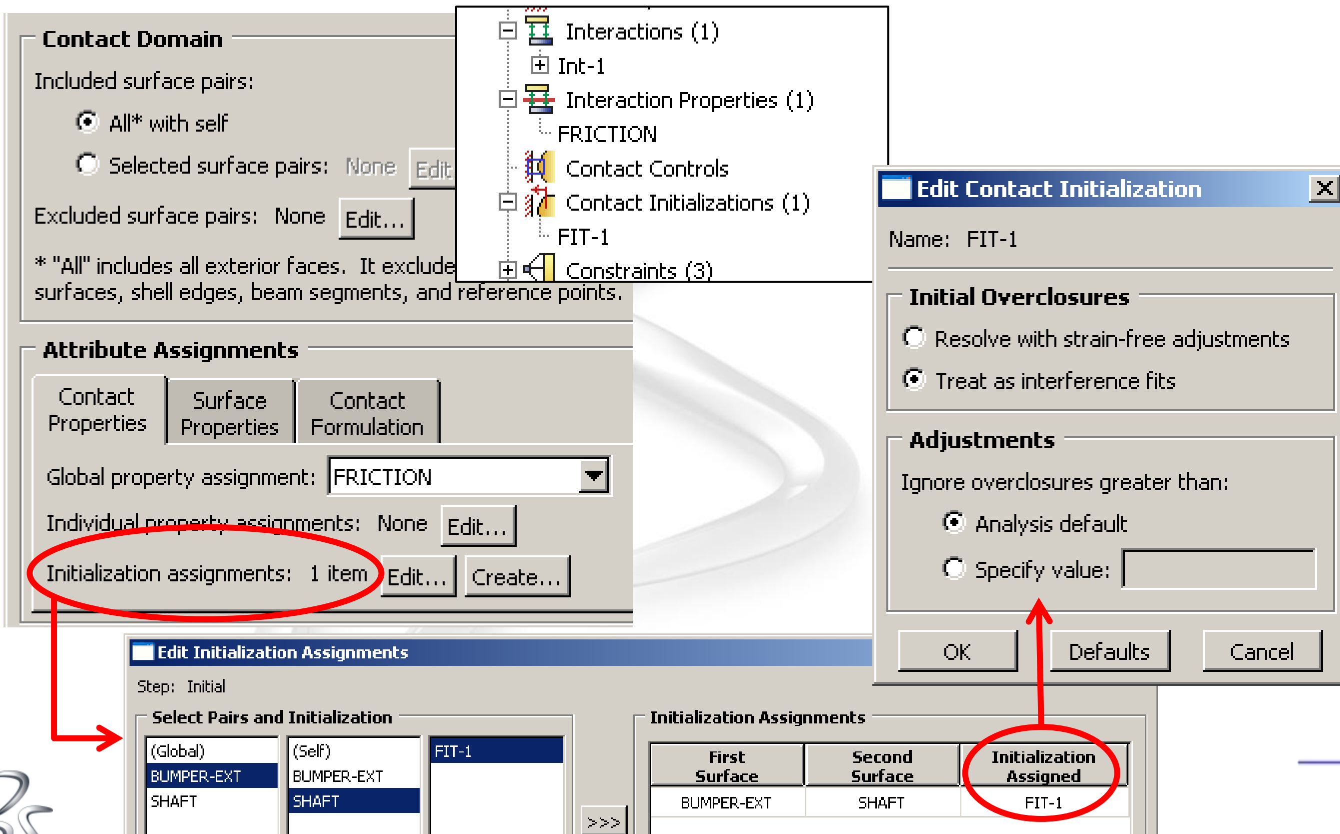
```
*Contact
*Contact Inclusions, ALL EXTERIOR
*Contact Exclusions
TOPPLATE , SHAFT
TOPPLATE , BOTPLATE
SHAFT , BOTPLATE
*Contact Property Assignment
 , , FRICTION
```

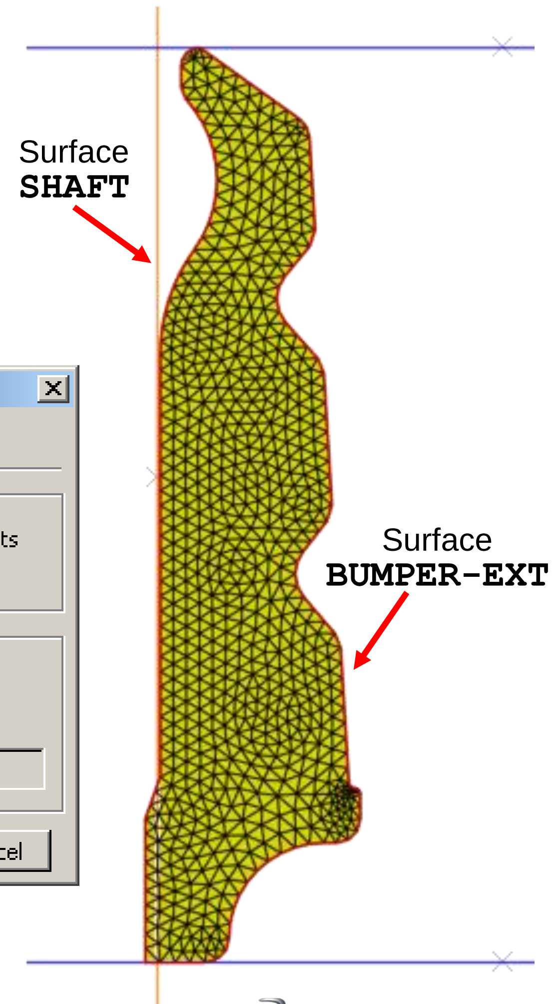
No effect on results of this analysis



Defining General Contact

- **Contact initialization**
  - The default behavior of general contact is to adjust small initial overclosures without strain
  - Can instead treat as interference fits

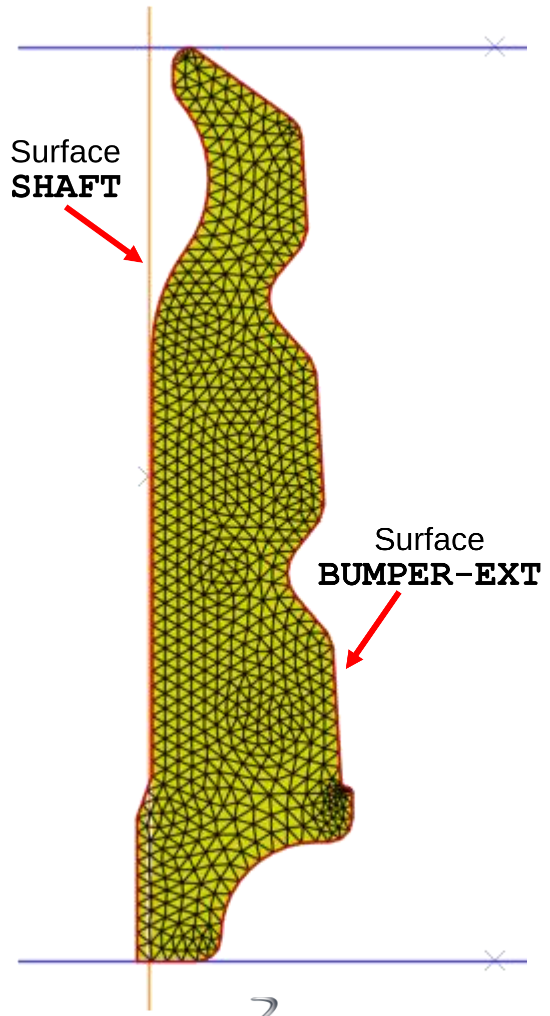




Defining General Contact

- **Contact initialization**
  - Keyword interface:

```
*Contact Initialization Data,
name=Fit-1, INTERFERENCE FIT
*Contact
*Contact Inclusions, ALL EXTERIOR
*Contact Property Assignment
 , , FRICTION
*Contact Initialization Assignment
 BUMPER-EXT, SHAFT, Fit-1
```



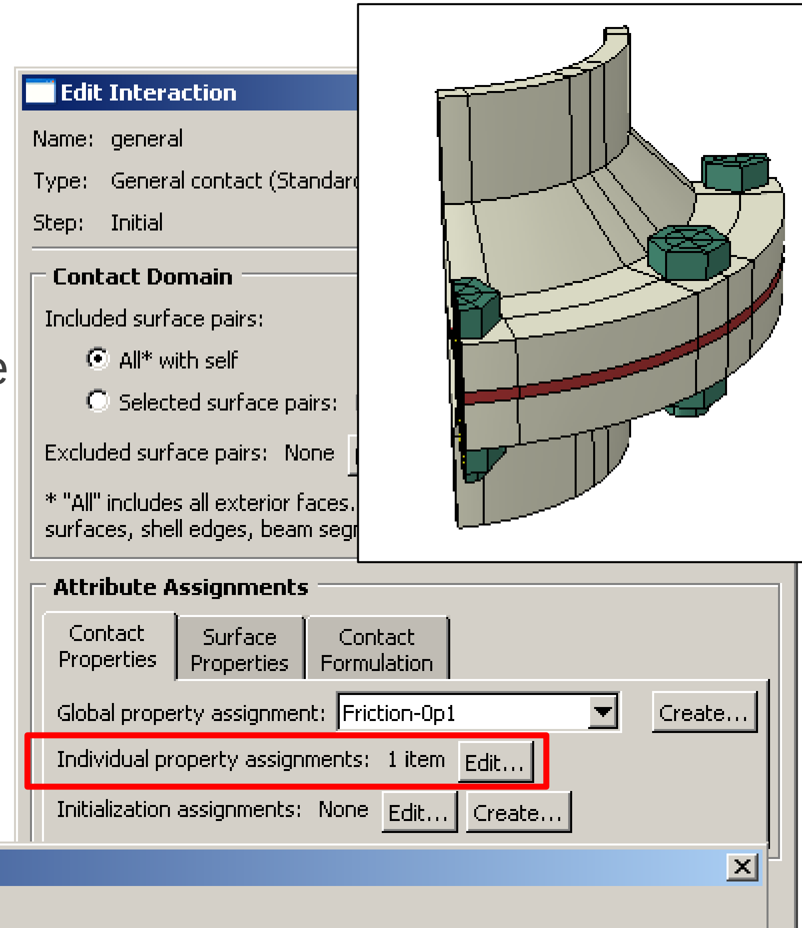
Defining General Contact

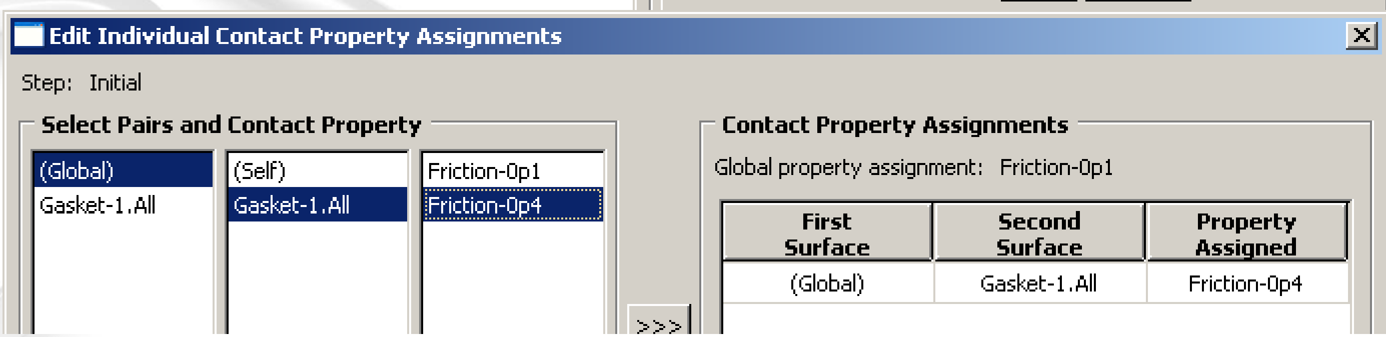
- **Contact properties**
  - Pertains to aspects such as:
    - Contact pressure-overclosure relationship
    - Friction
    - Contact damping
  - Defaults:
    - A “hard” pressure-overclosure relationship
      - No contact pressure until nodes are in contact
      - Unlimited contact pressure once contact has been established (enforced with a penalty method)
    - No friction
    - No contact damping
  - User can override contact property defaults globally and locally
    - Last assignment applies in case of conflicting assignments

Defining General Contact

- **Example: Bolted flange**
  - Coefficient of friction $\mu = 0.1$ for all contact interactions except for those involving the gasket ($\mu = 0.4$)

```
*Contact Property Assignment
 ,          , Friction-0p1
 , gasketAll, Friction-0p4
```



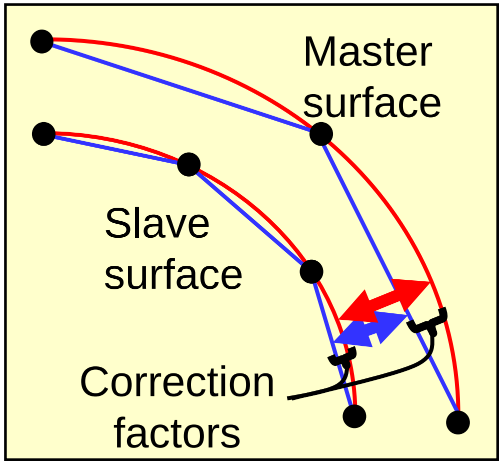


Representation of Curved Surfaces

SIMULIA

Representation of Curved Surfaces

- **Having faceted representations of curved surfaces is sometimes detrimental to accuracy and convergence**
  - Geometry corrections for the *surface-to-surface contact formulation* improve these aspects without degrading the per-iteration performance
    - Available for near-axisymmetric and near-spherical surfaces
    - Example applications on subsequent slides



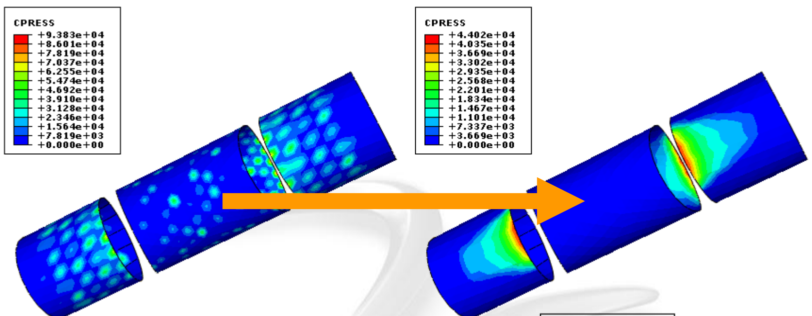
  - Whereas, surface-smoothing options for the *node-to-surface contact formulation* primarily target convergence issues associated with having discontinuous surface normals
    - But generally do not strive to represent exact initial geometry
    - Details depend on whether surfaces are 2-D or 3-D, rigid or deformable (not discussed in this seminar)

Will discuss contact formulations in next lecture

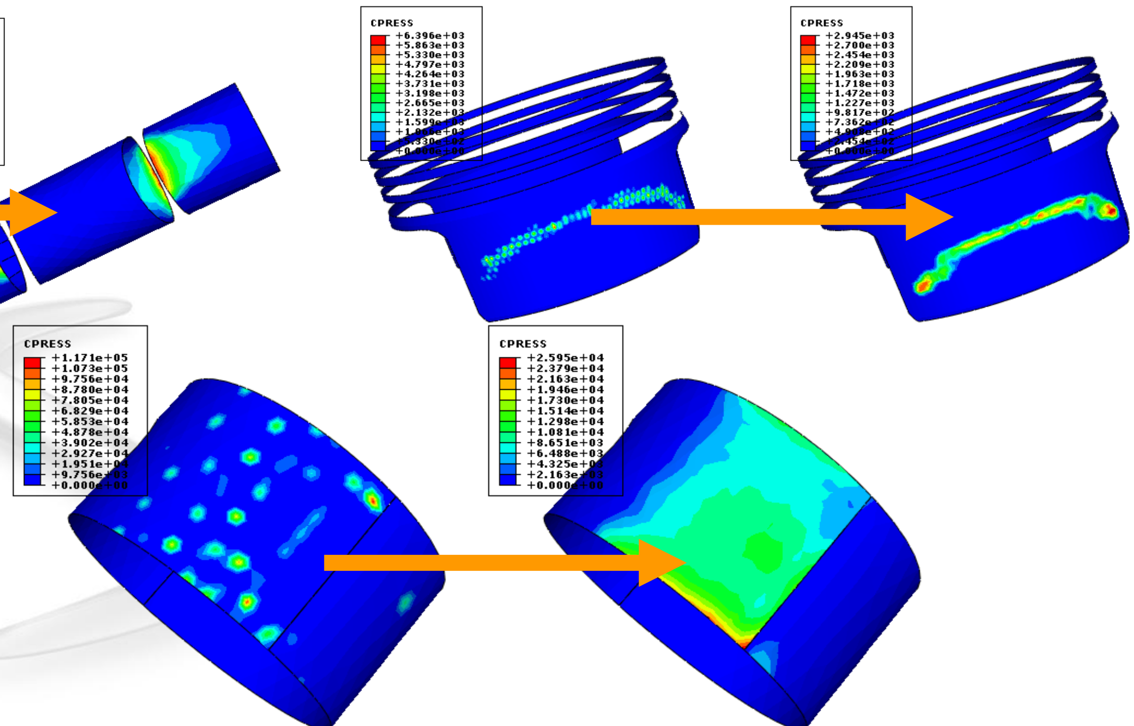
Representation of Curved Surfaces

- **Effect of geometric corrections in a piston application**

Rod and piston-to-pinion



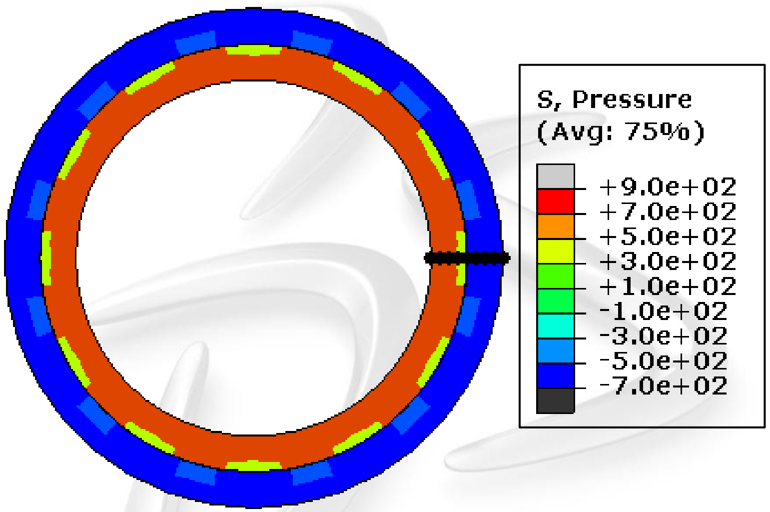
Piston-to-cylinder



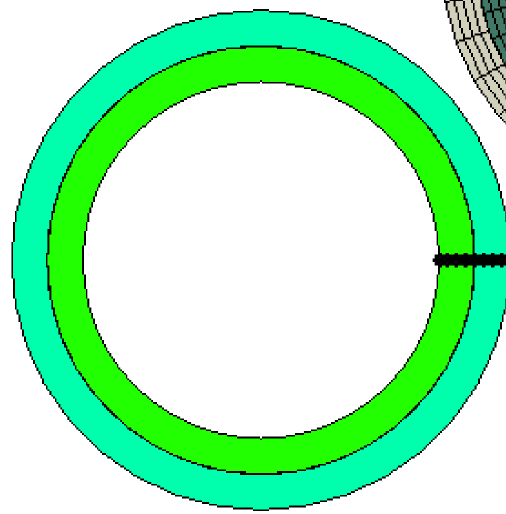
Cap and rod-to-crank

Representation of Curved Surfaces

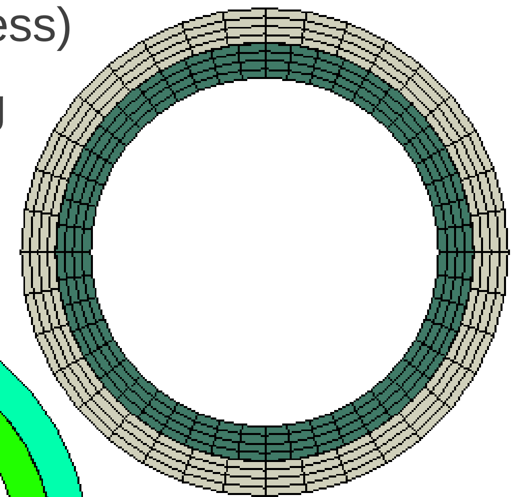
- **Example: Concentric rings with interference fit and finite sliding**
  - Spin inner ring after resolving interference (frictionless)
  - Analytical solution: Uniform pressure stress per ring



Noisy solution with faceted geometry

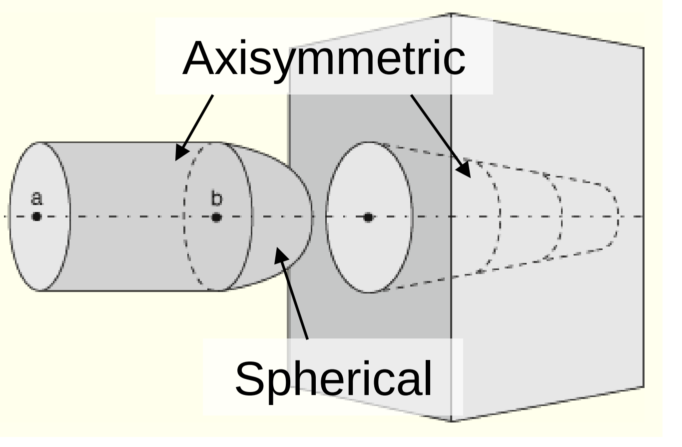


Accurate solution with geometric corrections



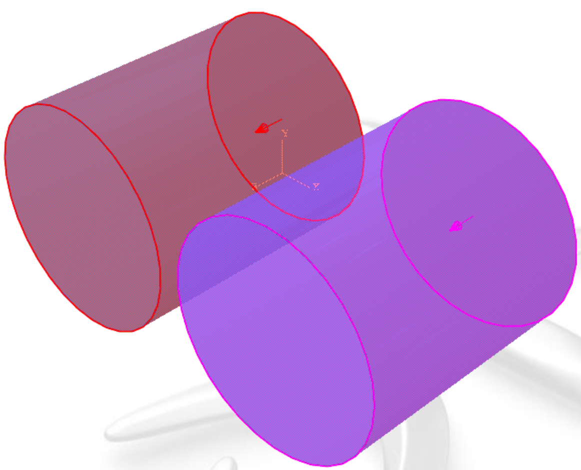
Representation of Curved Surfaces

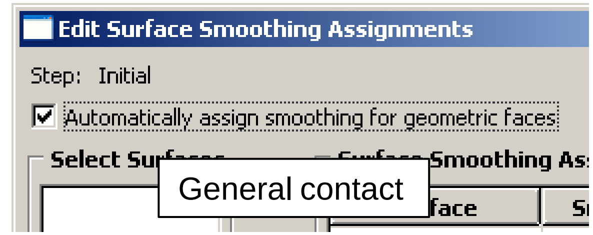
- **Applicability of geometric corrections:**
  - Significant effect for small-to-moderate deformation
    - Effect usually insignificant after large deformation
  - Small- or finite-sliding, surface-to-surface contact formulation
  - Applicable to the most-common curved geometries; portions of surface geometry must be approximately:
    - Circular in 2-D
    - Axisymmetric or spherical in 3-D

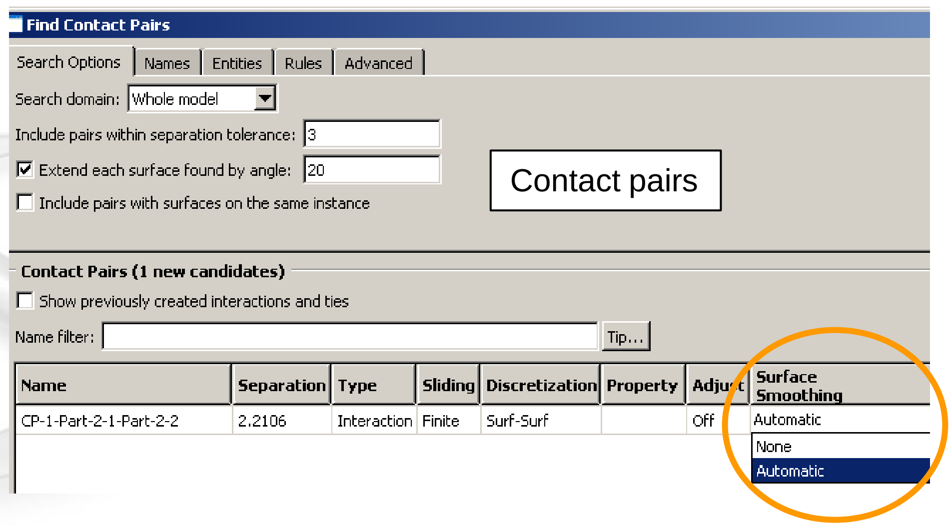


Representation of Curved Surfaces

- **Abaqus/CAE automatically detects these surfaces in native geometry models and applies appropriate smoothing method in contact interactions**







- **Benefits:**
  - Improved accuracy
    - Avoid need for matched nodes across contact interface
  - Reduced iteration count (sometimes)

Representation of Curved Surfaces

- **Keyword interface for general contact**

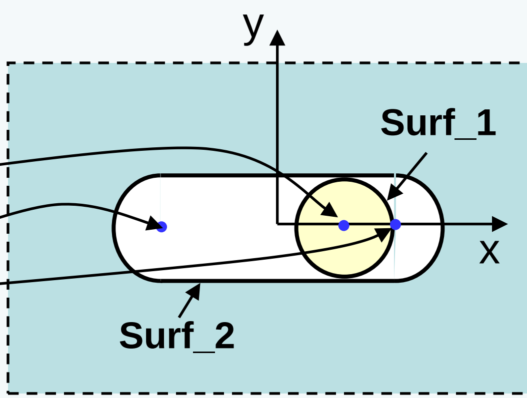
```
*Contact
*Contact Inclusions, All Exterior
*Surface Property Assignment, Property=Geometric Correction
 surface_name, CIRCUMFERENTIAL, Xa, Ya
 surface_name, CIRCUMFERENTIAL, Xa, Ya, Za, Xb, Yb, Zb
 surface_name, SPHERICAL, Xa, Ya, Za
```

2-D: Center of circle

3-D: 2 points on symmetry axis

3-D: Center of sphere

```
*Contact
*Contact Inclusions, All Exterior
*Surface Property Assignment, Prop=Geom
 Surf_1, CIRCUMFERENTIAL, 1.5, 0.0
 Surf_2a, CIRCUMFERENTIAL, -2.5, 0.0
 Surf_2b, CIRCUMFERENTIAL, 2.5, 0.0
```



Semi-circle on the left side of Surf_2

Semi-circle on the right side of Surf_2

Representation of Curved Surfaces

- **Keyword interface for surface-to-surface contact pairs**

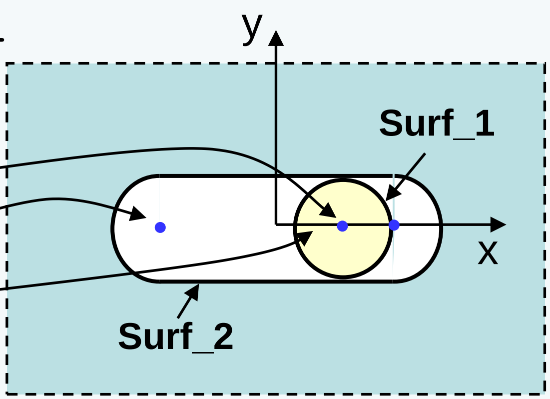
```
*Contact Pair, Type=Surface to Surface, Geometric
  Correction=smoothing_name
*Surface Smoothing, Name=smoothing_name
 slave_region, master_region, CIRCUMFERENTIAL, Xa, Ya
 slave_region, master_region, CIRCUMFERENTIAL, Xa, Ya, Za, Xb, Yb, Zb
 slave_region, master_region, SPHERICAL, Xa, Ya, Za
```

2-D: Center of circle

3-D: 2 points on symmetry axis

3-D: Center of sphere

```
*Contact Pair, Type=Surface, Geom=Smooth1
 Surf_1, Surf_2
*Surface smoothing, Name=Smooth1
 Surf_1, , CIRCUMFERENTIAL, 1.5, 0.0
  , Surf_2a, CIRCUMFERENTIAL, -2.5, 0.0
  , Surf_2b, CIRCUMFERENTIAL, 2.5, 0.0
```

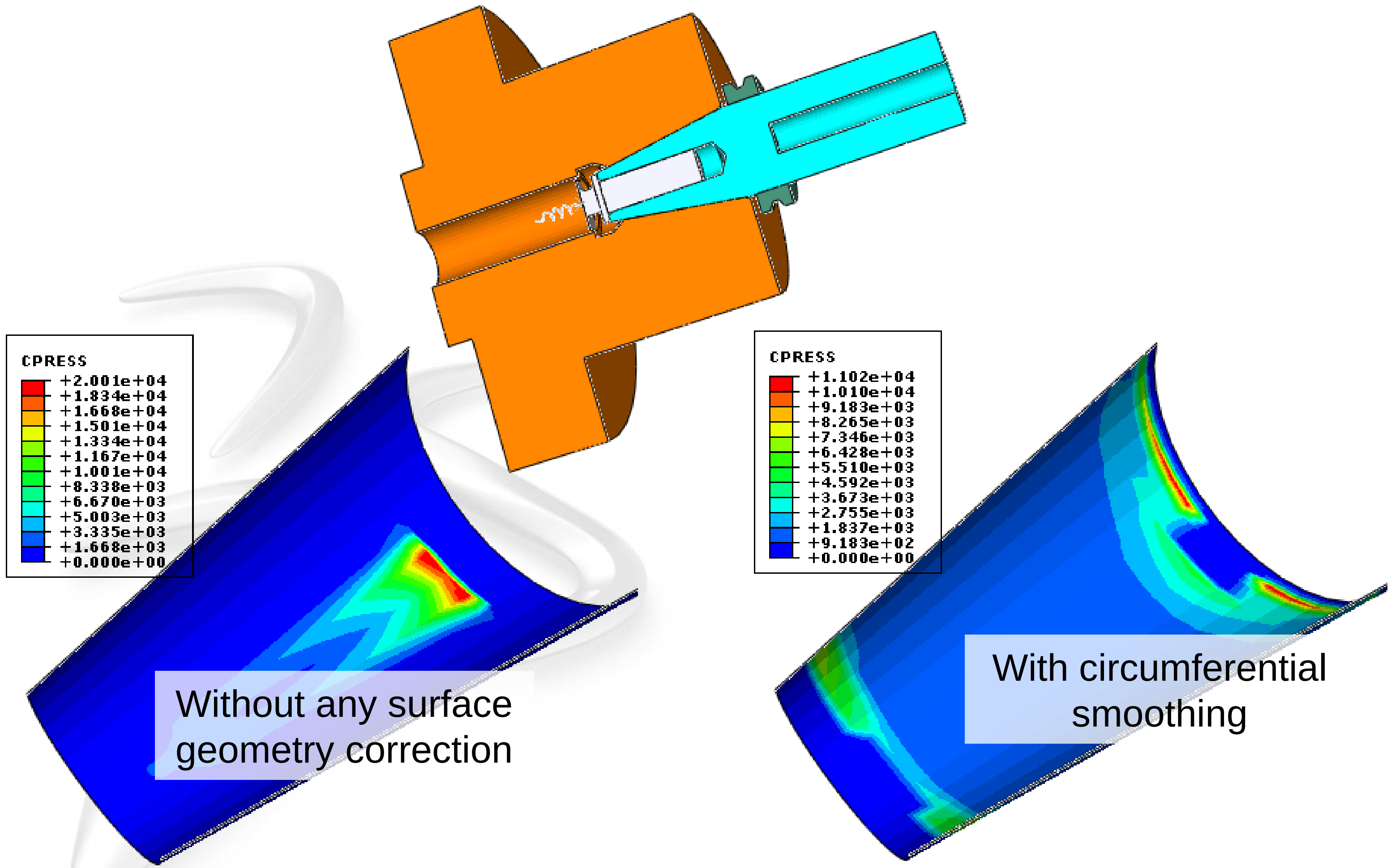


Semi-circle on the left side of Surf_2

Semi-circle on the right side of Surf_2

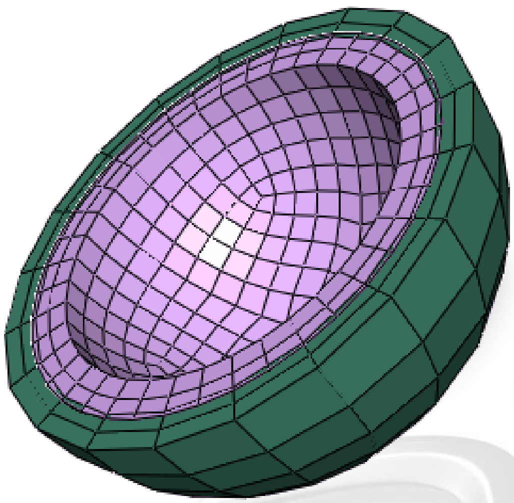
Representation of Curved Surfaces

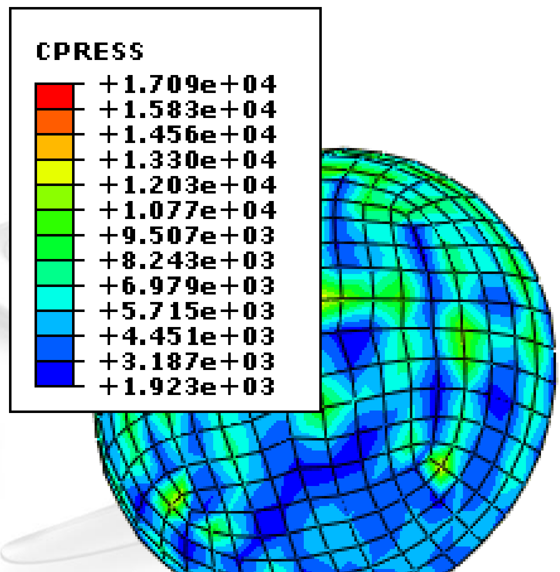
- **Example: Conical contact interface**



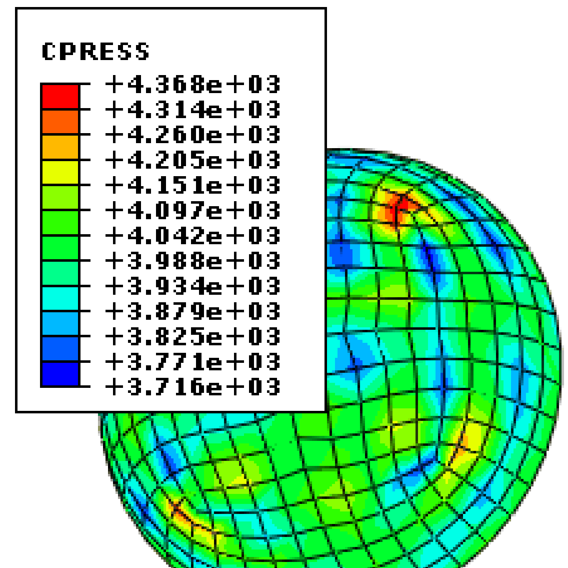
Representation of Curved Surfaces

- **Example: Spherical contact interface**
  - Uniform interference fit





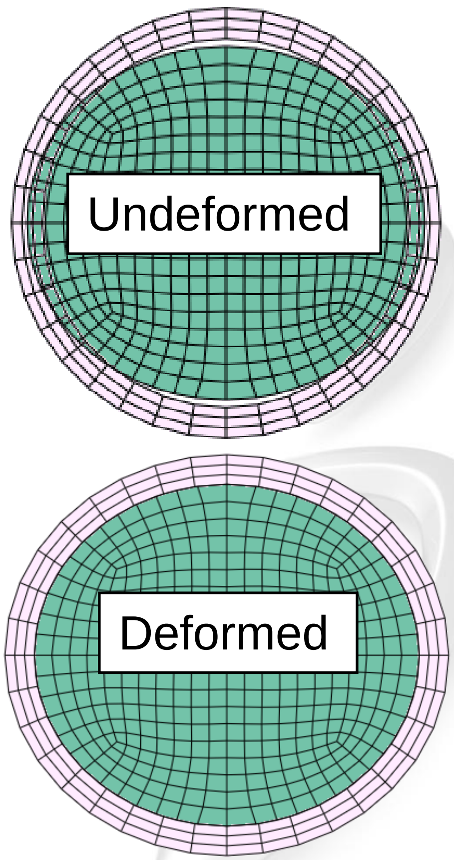
Without any surface geometry correction

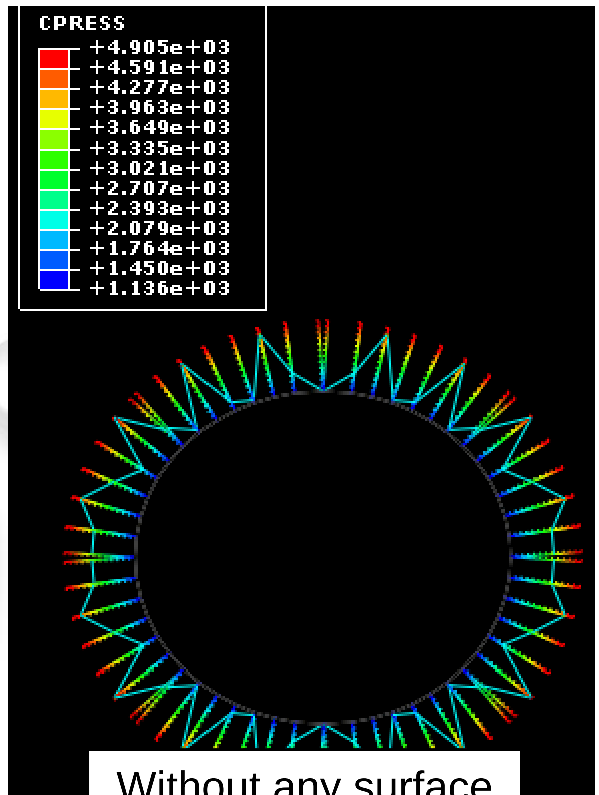


With spherical smoothing

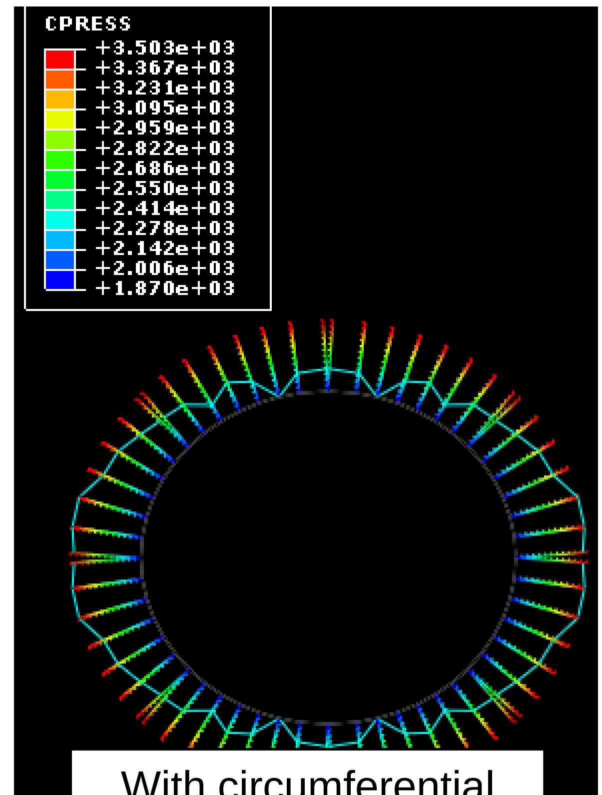
Representation of Curved Surfaces

- **Capability applicable even if surface geometry deviates somewhat from perfect cylinder, sphere, etc.**
  - Example: Interference fit between elliptical disk and circular ring





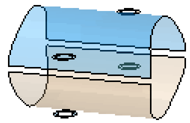
Without any surface geometry correction



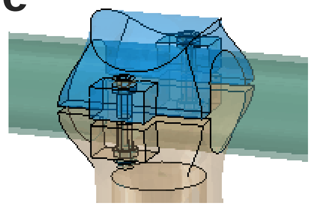
With circumferential correction

Representation of Curved Surfaces

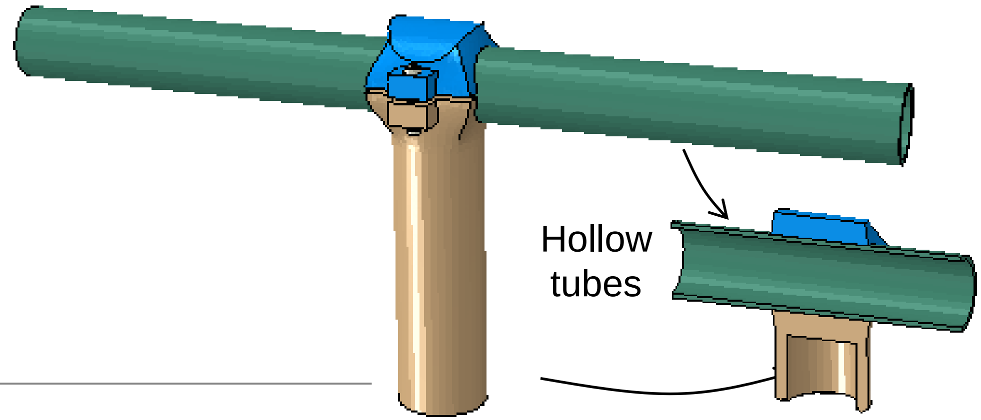
- **Clamp example**

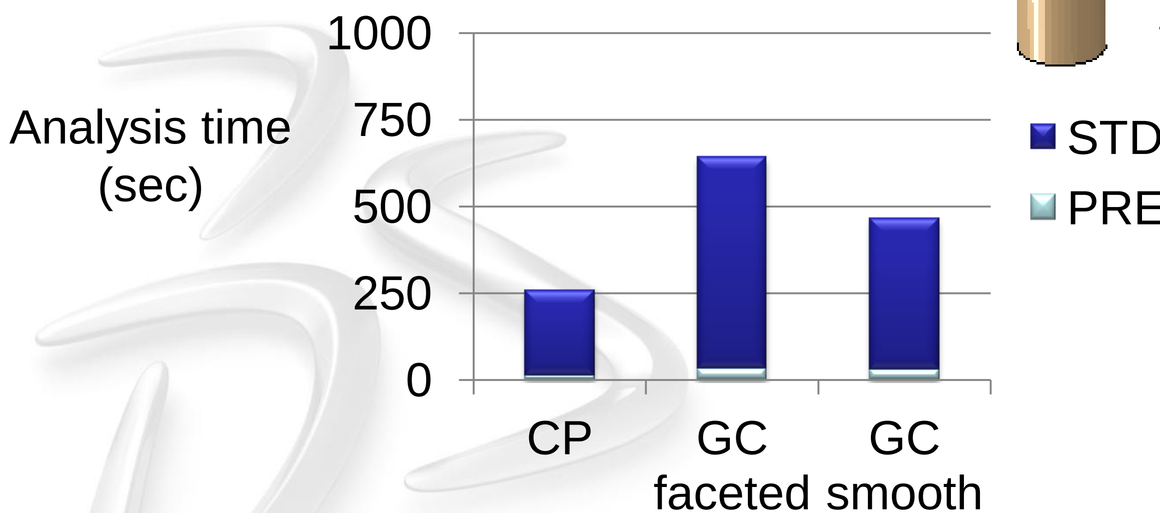


Slave surfaces of contact pairs



General contact internal surface

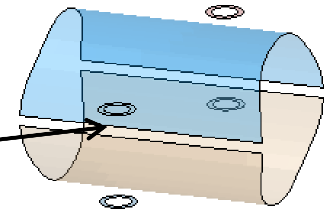




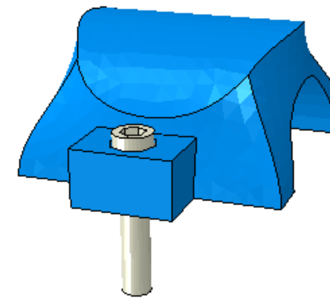
# iterations	36	55	41

Representation of Curved Surfaces

- **Clamp example (cont.)**
  - *Contact pair* model does not consider contact involving shank

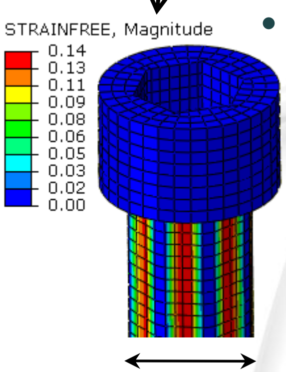


Slave surfaces of contact pairs



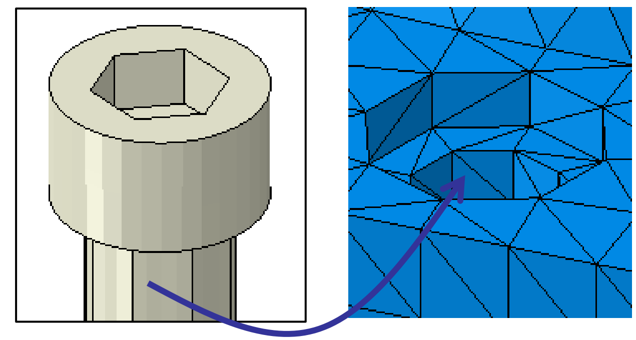
  - Coarse refinement near bolt holes

New output in Abaqus 6.9-EF



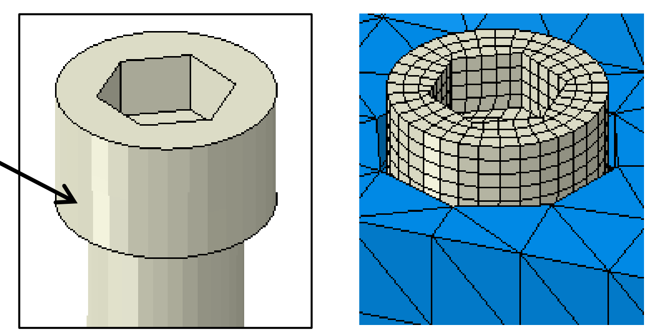
D=5

- Nonphysical initial overclosures for general contact without circumferential smoothing



Shank cross-section adjusted to conform to hole facets

- Realistic small gaps at these interfaces for general contact with circumferential corrections activated
  - Shank remains cylindrical
    - More realistic
    - “STRAINFREE” output is 0.0
  - Improved performance



Lecture 2 Summary

Review of Topics Discussed in this Lecture

- **Defining Contact Pairs**
- **Defining Surfaces for Contact Pairs**
- **Defining General Contact**
- **Representation of Curved Surfaces**

Numerical Methods

Lecture 3

Overview

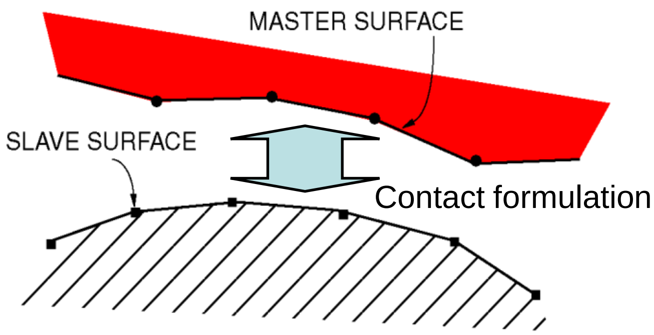
- **Contact Formulation Aspects**
  - Contact Discretization
  - Contact Enforcement
- **Incrementation and Newton Iterations**
- **Summary**

Contact Formulation Aspects

SIMULIA

Contact Formulation Aspects

- **Discretization**
  - How are constraints **formed**?
    - For example, how to calculate gap or penetration distances from nodal positions
    - Node-to-surface, surface-to-surface, and edge-to-surface formulations

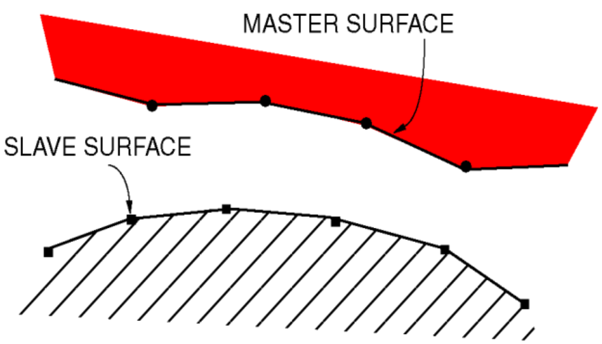


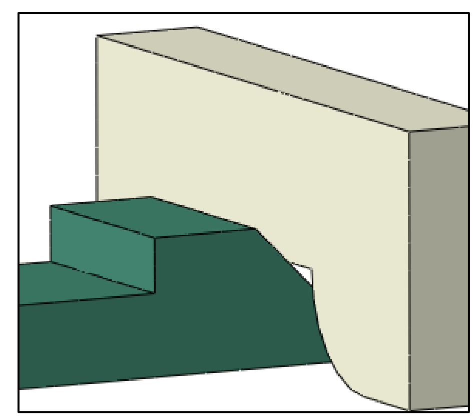
- **Enforcement**
  - How are constraints **enforced**?
    - For example, numerical method to resist penetrations
    - Direct (Lagrange multipliers) or penalty
- **Evolution of discretization**
  - How do constraints **evolve** upon sliding?
    - Rigorous, nonlinear evolution (“finite sliding”) vs. approximate (“small sliding”)

Contact Discretization

Contact Discretization

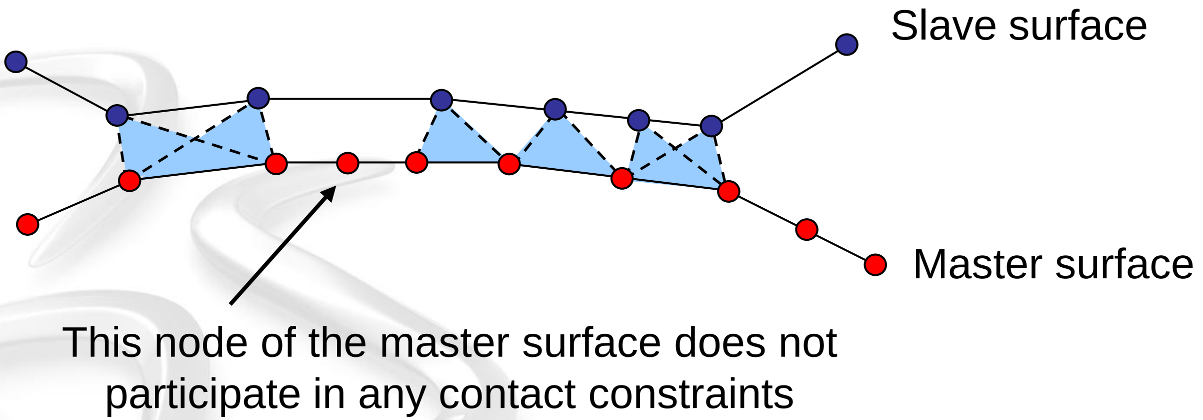
- **Node-to-surface technique**
  - Nodes on one surface (the slave surface) contact the segments on the other surface (the master surface)
  - Contact enforced at discrete points (slave nodes)
- **Surface-to-surface technique**
  - Contact enforced in an average sense over a region surrounding each slave node
  - Slave surface much more than just a collection of nodes
  - Fundamental to the development of general contact in Abaqus/Standard
- **Edge-to-surface technique**
  - Contact between a feature edge and a surface
  - Enforced in an average sense over portions of feature edges
  - Supplemental formulation for general contact starting in Abaqus/Standard 6.11





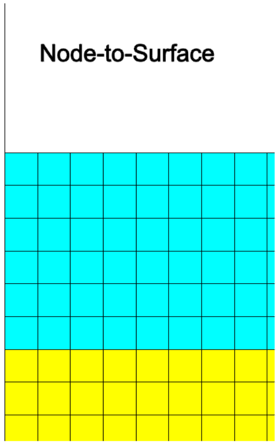
Contact Discretization

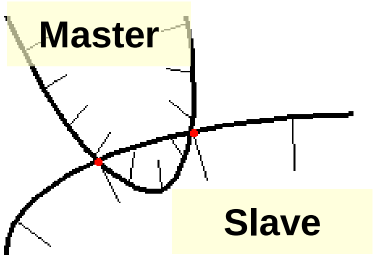
- **Node-to-surface (N-to-S) contact discretization**
  - Traditional “point-against-surface” method
  - Each potential contact constraint with this formulation involves a “slave” node and a “master” facet

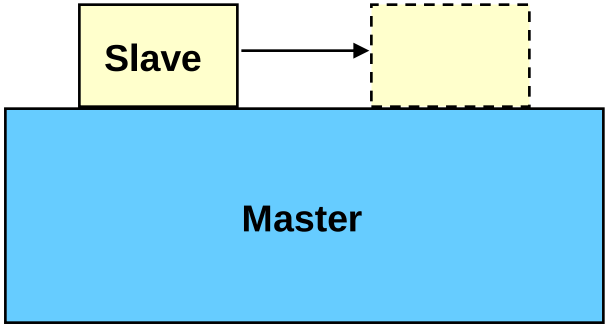


Contact Discretization

- **Key implications of node-to-surface formulation**
  - Slave nodes **cannot** penetrate master surface facets
  - Master nodes **are not explicitly restricted** from penetrating slave surface facets (and sometimes do penetrate the slave surface)
  - Refinement of slave surface helps avoid gross penetration of master nodes into slave surface
- **Guidelines for master and slave roles**
  - More-refined surface should act as slave surface
  - Stiffer body should be master
  - Active contact region should change most rapidly on master surface
    - Minimizes contact status changes



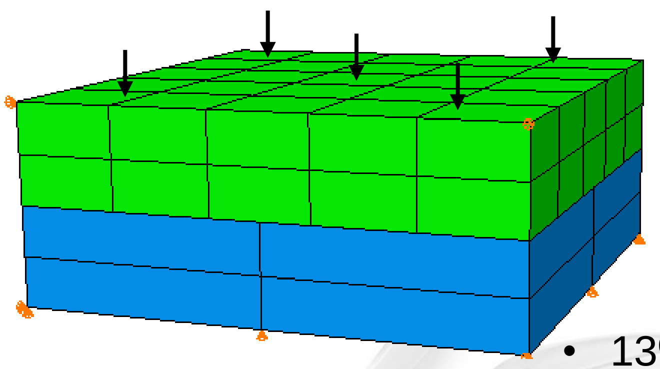


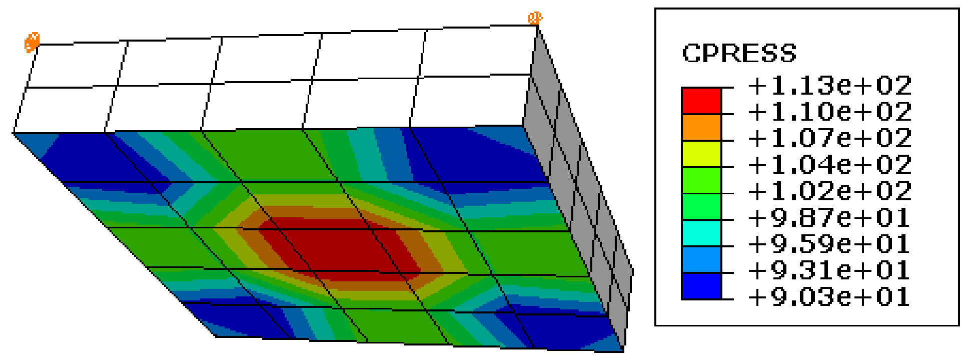


Contact Discretization

- **While refinement of slave surface leads to global accuracy, local contact stress oscillations may still be observed with N-to-S**

Uniform pressure load, σ=100

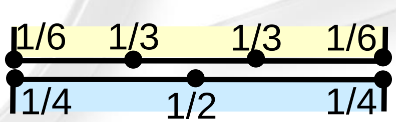




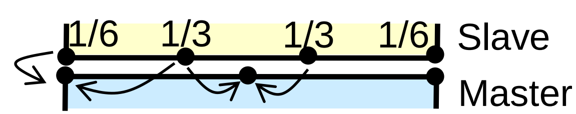
- 13% noise in CPRESS solution with N-to-S contact discretization if top block acts as slave (shown above)
- 31% CPRESS noise if bottom block acts as slave (not shown)

- 2-D example

Ideal contact force distribution factors (uniform pressure, linear elements):



Factors on master nodes assuming ideal factors on slave nodes:



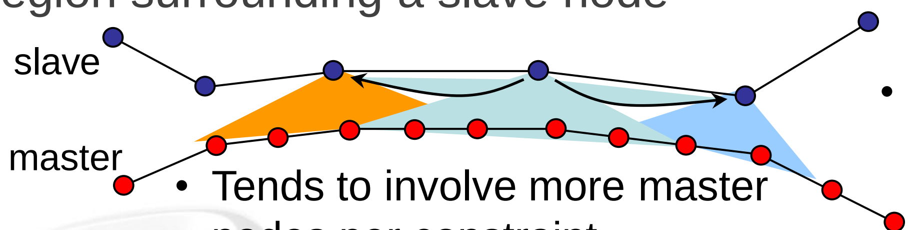
1/6×1 + ⅓×⅓ = 5/18 | 2×⅓×⅔ = 8/18 | 5/18

11% deviation

- "Matching meshes" across contact interface avoids this noise

Contact Discretization

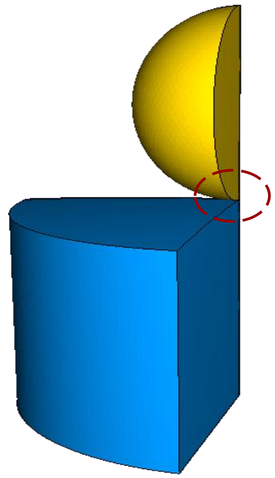
- **Surface-to-surface (S-to-S) contact discretization**
  - Each contact constraint is formulated based on an integral over the region surrounding a slave node



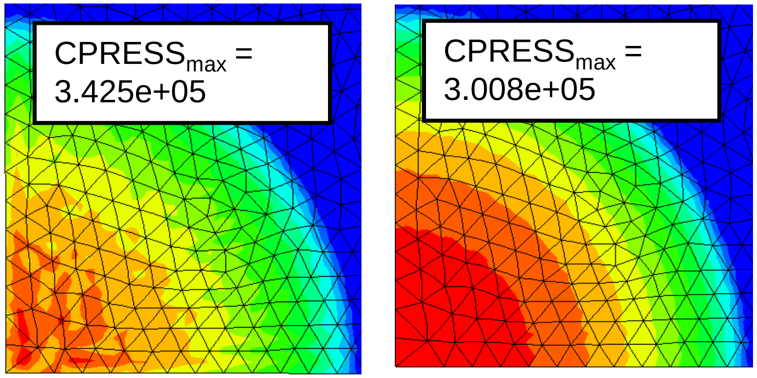
  - Tends to involve more master nodes per constraint
    - Especially if master surface is more refined than slave surface
  - Also involves coupling among slave nodes
  - Still best to have the more-refined surface act as slave
    - Better performance and accuracy
  - Benefits of surface-to-surface approach
    - Reduced likelihood of large localized penetrations
    - Reduced sensitivity of results to master and slave roles
    - More accurate contact stresses (without “matching meshes”)
    - Inherent smoothing (better convergence)

Contact Discretization

- **S-to-S discretization often improves accuracy of contact stresses**
  - Related to better distribution of contact forces among master nodes
  - Example: Classical Hertz contact problem:
    - Contact pressure contours much smoother and peak contact stress in very close agreement with the analytical solution using surface-to-surface approach



Analytical $CPRESS_{max}$ = 3.01e+05

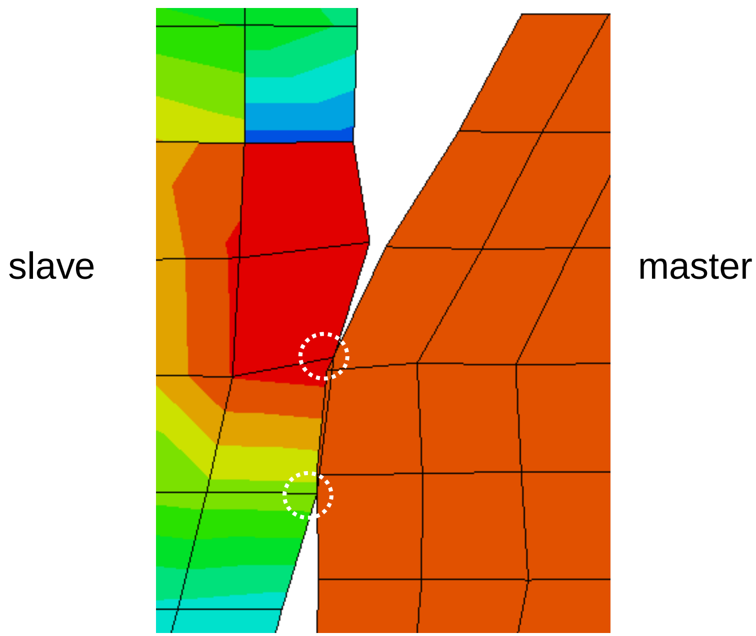


Node-to-surface　　Surface-to-surface

Contact Discretization

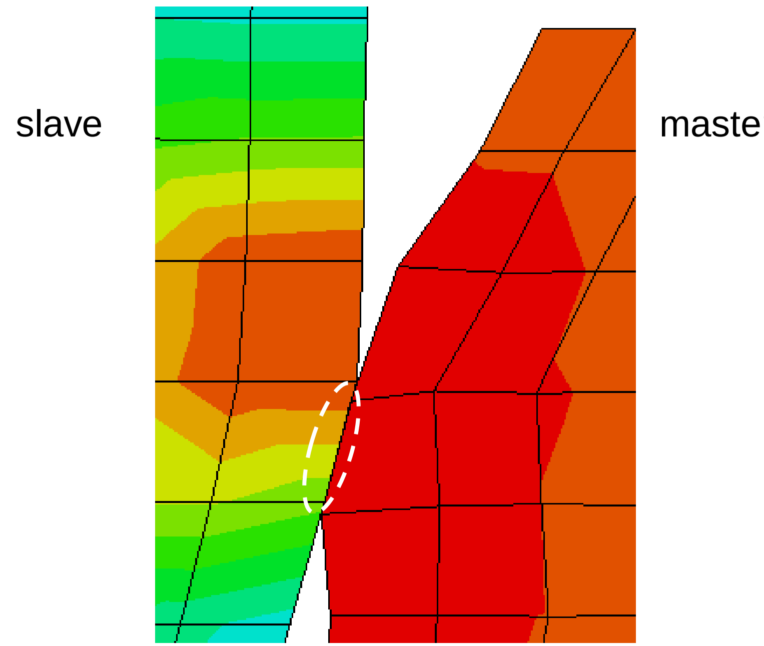
- **S-to-S discretization reduces likelihood of snagging**

Node-to-surface



Treating slave surface as collection of points can trigger snagging as slave nodes traverse a corner

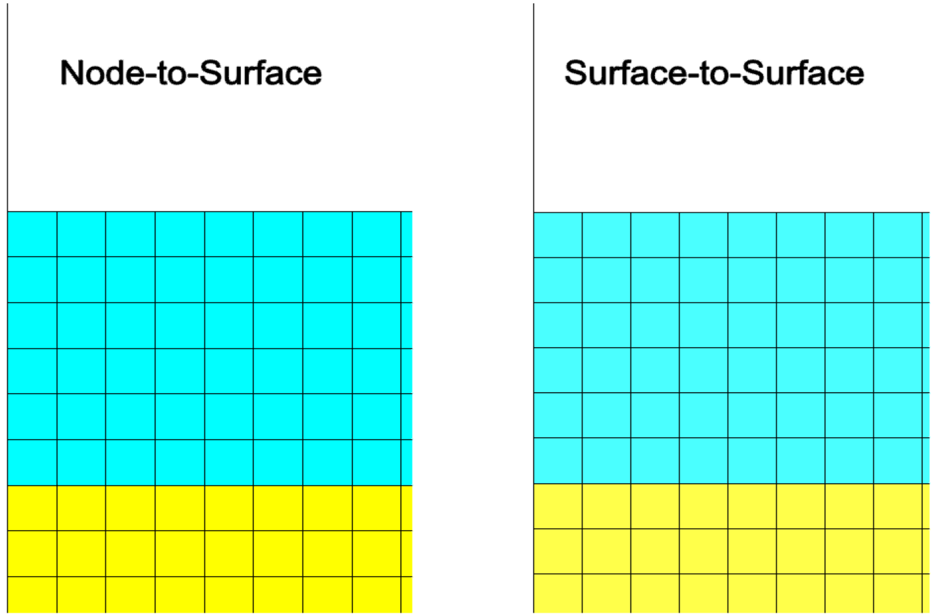
Surface-to-surface



Computing average penetrations and slips over finite regions has smoothing effect that avoids snagging

Contact Discretization

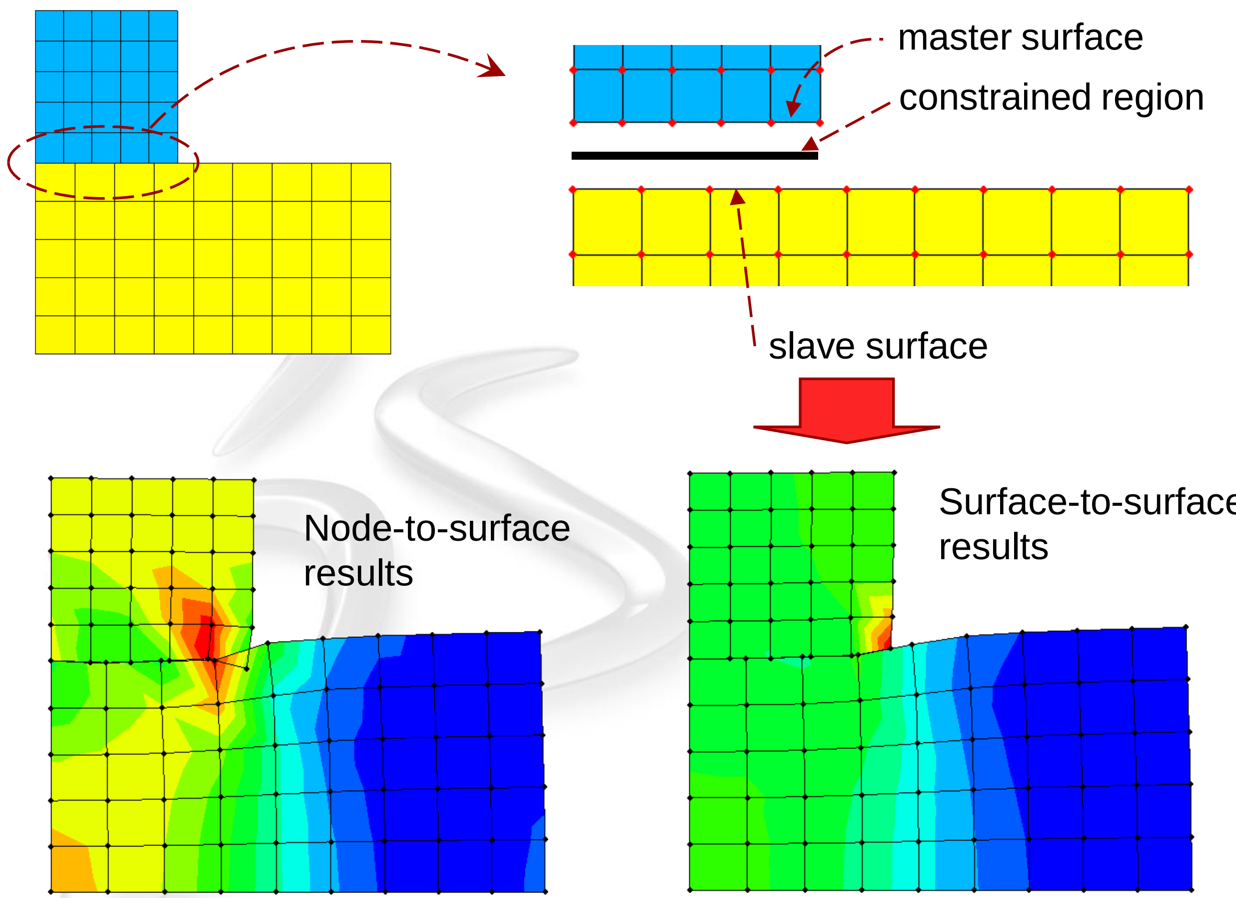
- **S-to-S discretization reduces likelihood of master nodes penetrating slave surface**



Contact Discretization

- **S-to-S discretization reduces likelihood of master nodes penetrating slave surface** (another example)

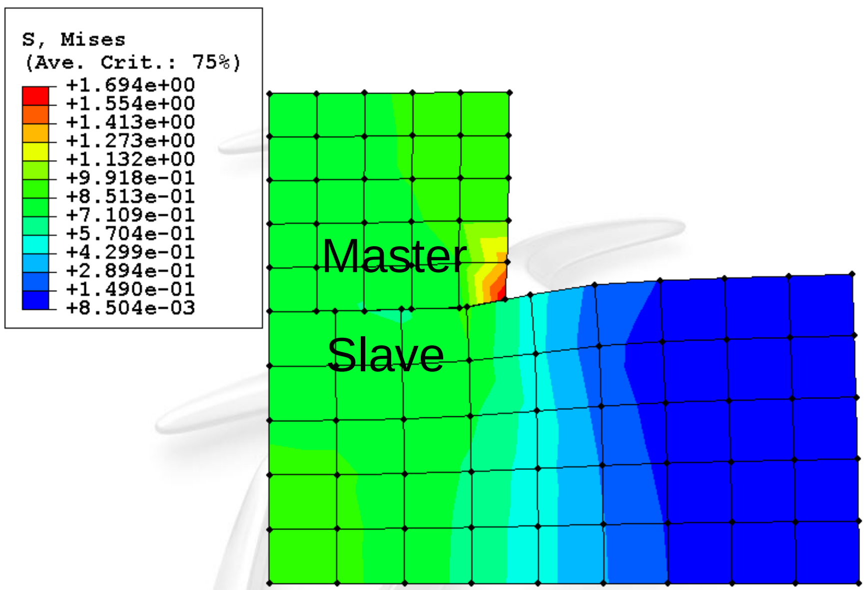
Non-ideal slave and master roles

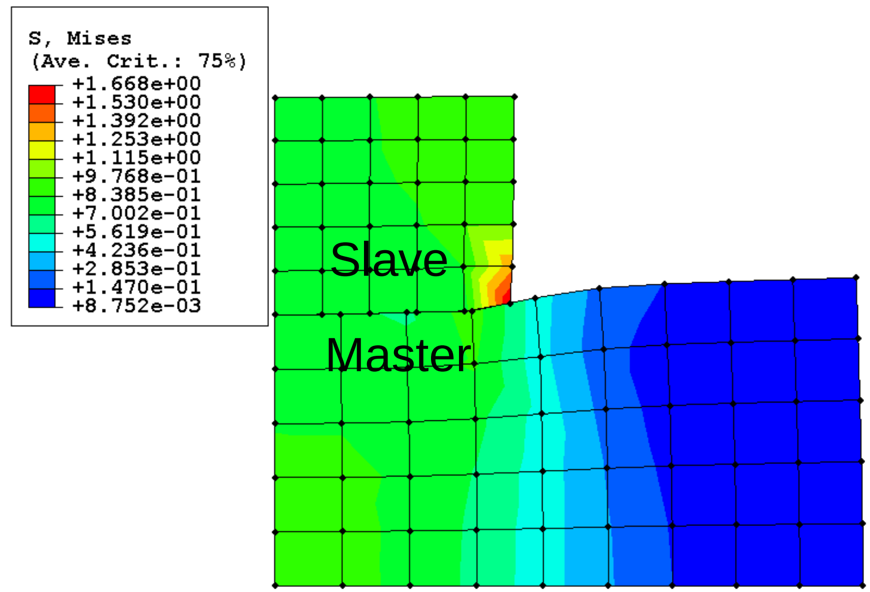


Some penetration may be observed at individual nodes; however, large, undetected penetrations of master nodes into slave surface do not occur

Contact Discretization

- **S-to-S discretization much less sensitive to choice of master and slave surfaces**
  - Results with S-to-S discretization nearly independent of master/slave roles in this example:



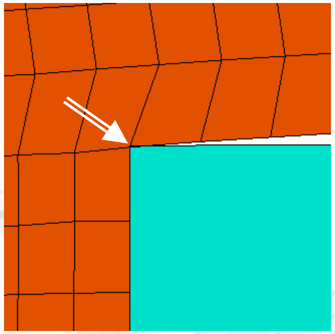


Choosing slave surface to be finer mesh will still yield better results; choosing the master surface to be more refined surface will tend to increase analysis cost

Contact Discretization

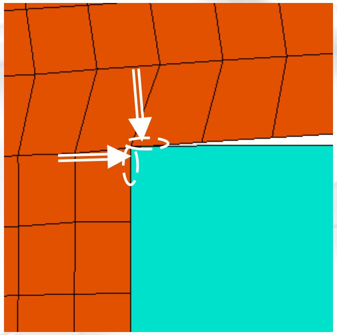
- **S-to-S discretization will generate multiple constraints at corners when appropriate**

<u>Node-to-surface</u>



- Single constraint in “average” normal direction at corner
  - Not stable
  - Leads to large penetrations and snagging
- Workaround: Two contact pairs

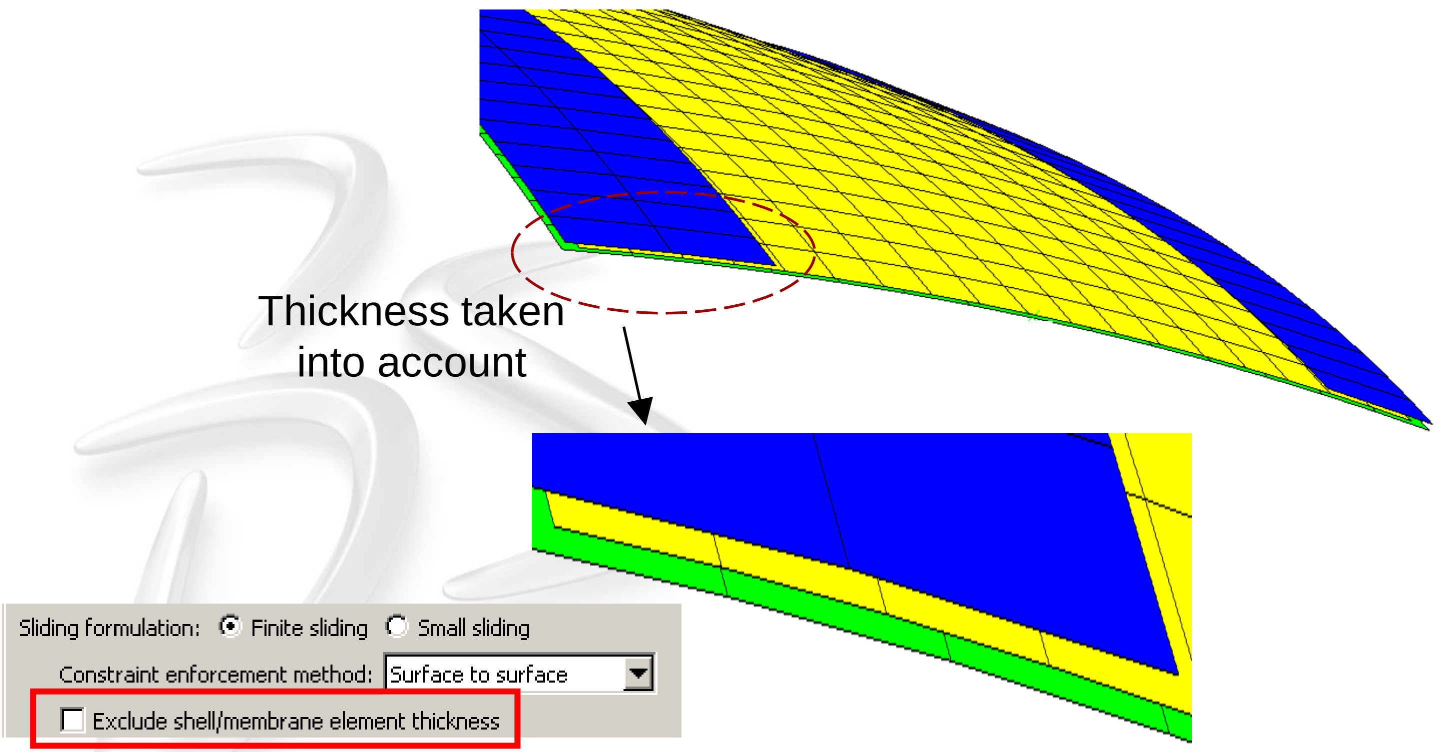
<u>Surface-to-surface</u>



- Two constraints are generated at corner (even if one contact pair)
  - See arrows near corner
  - Accurate and stable
- No smoothing of surface normals

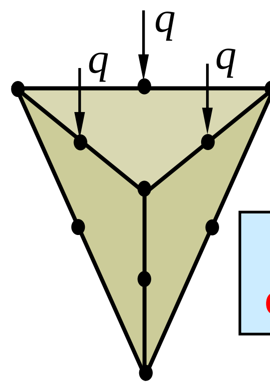
Contact Discretization

- **S-to-S discretization takes into consideration shell and membrane thicknesses when performing contact calculations**
  - N-to-S considers this effect only for the small-sliding formulation



Contact Discretization

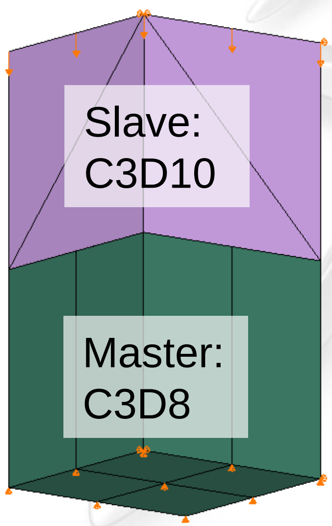
- **S-to-S discretization is fundamentally sound for situations in which quadratic elements underlie slave surface**
- **N-to-S struggles with some quadratic element types**
  - Related to:
    - Discrete treatment of slave surface
    - “Consistent” force distribution for element
  - Workarounds (with pros and cons):
    - C3D10M, supplementary constraints, etc.

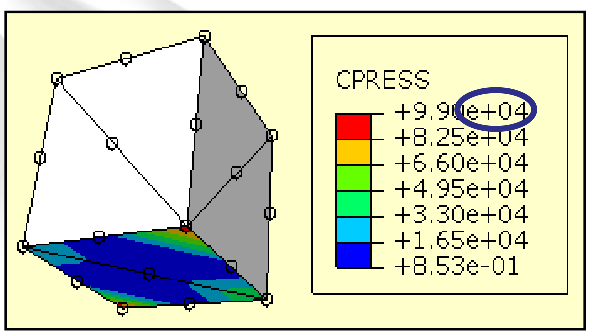


$$q = \frac{1}{3} pA$$

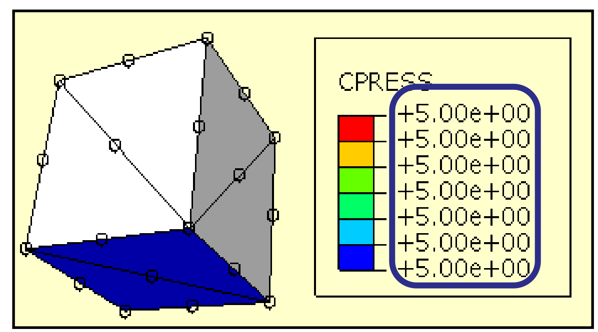
Zero force at corner nodes

Uniaxial pressure loading of 5.0





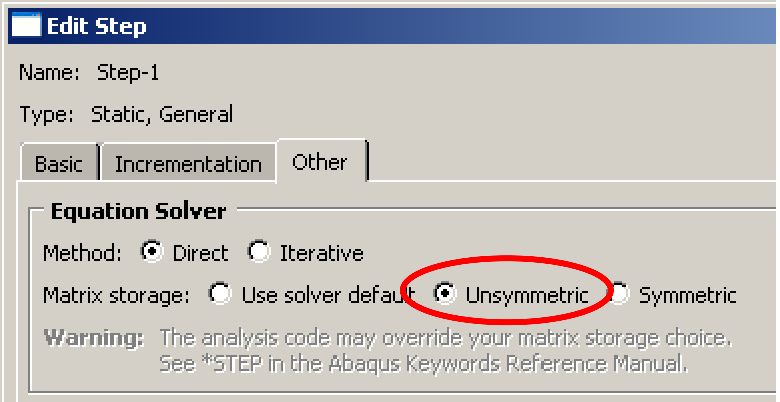
Node-to-surface



Surface-to-surface

Contact Discretization

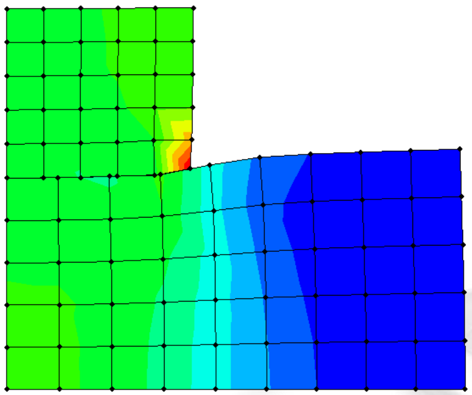
- **S-to-S discretization has *greater* tendency to generate unsymmetric stiffness terms where master and slave surface are not approximately parallel to each other**
  - Use of unsymmetric solver is sometimes necessary to avoid convergence difficulties

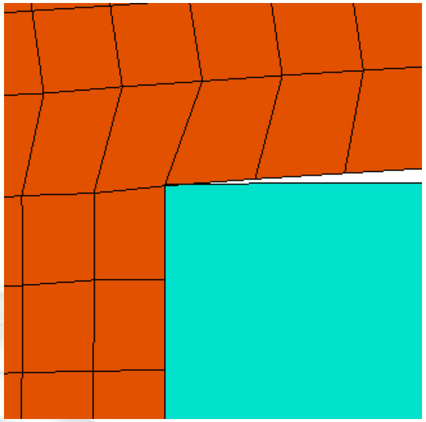


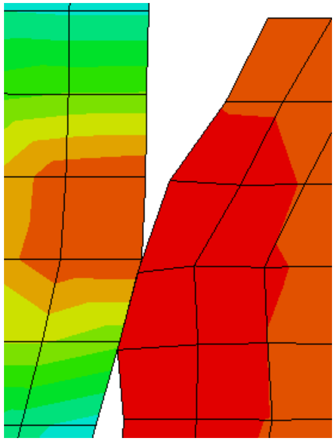

```
*STEP, UNSYMM=YES
```

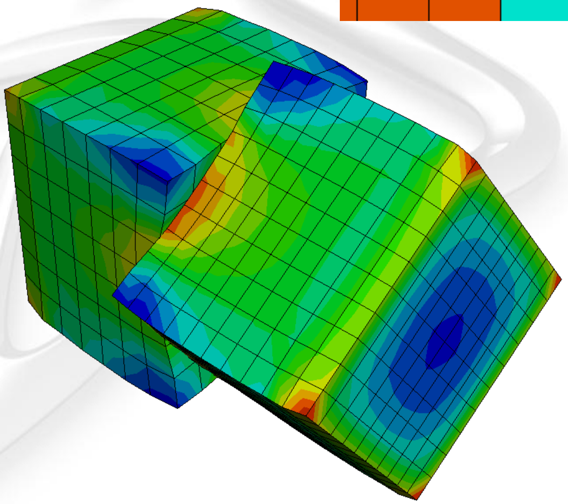
Contact Discretization

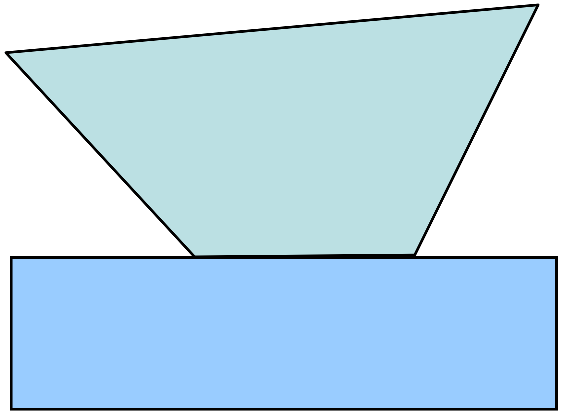
- **S-to-S discretization works best when contacting surfaces have nearly opposing normals**
  - Works well for many cases involving corners





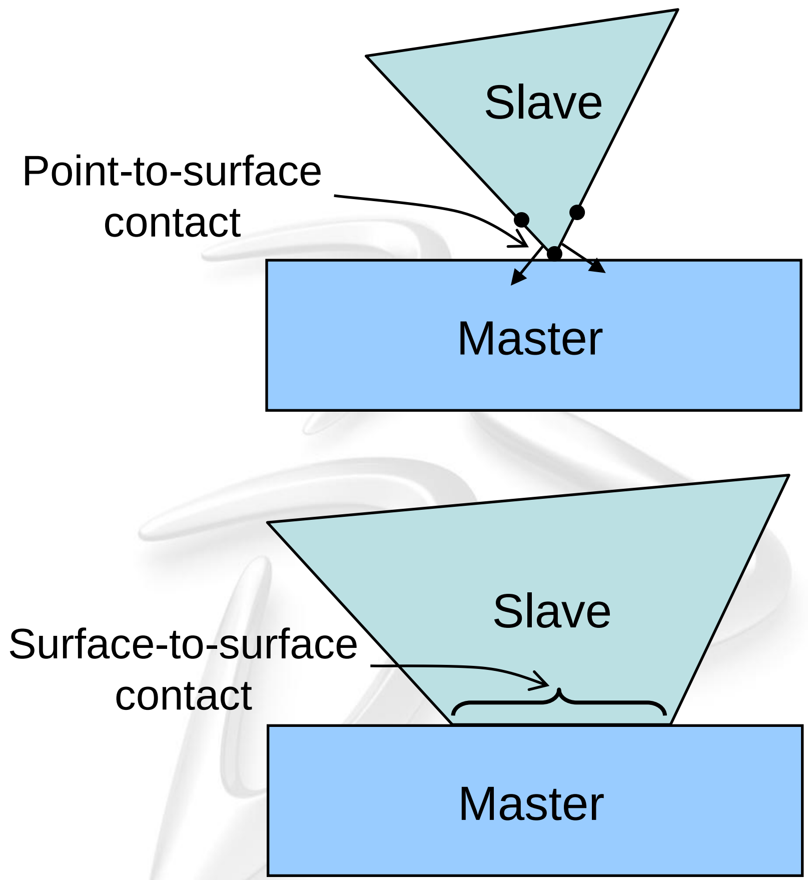






Contact Discretization

- **Surface-to-surface discretization, however, has difficulty resolving point-to-surface contact**

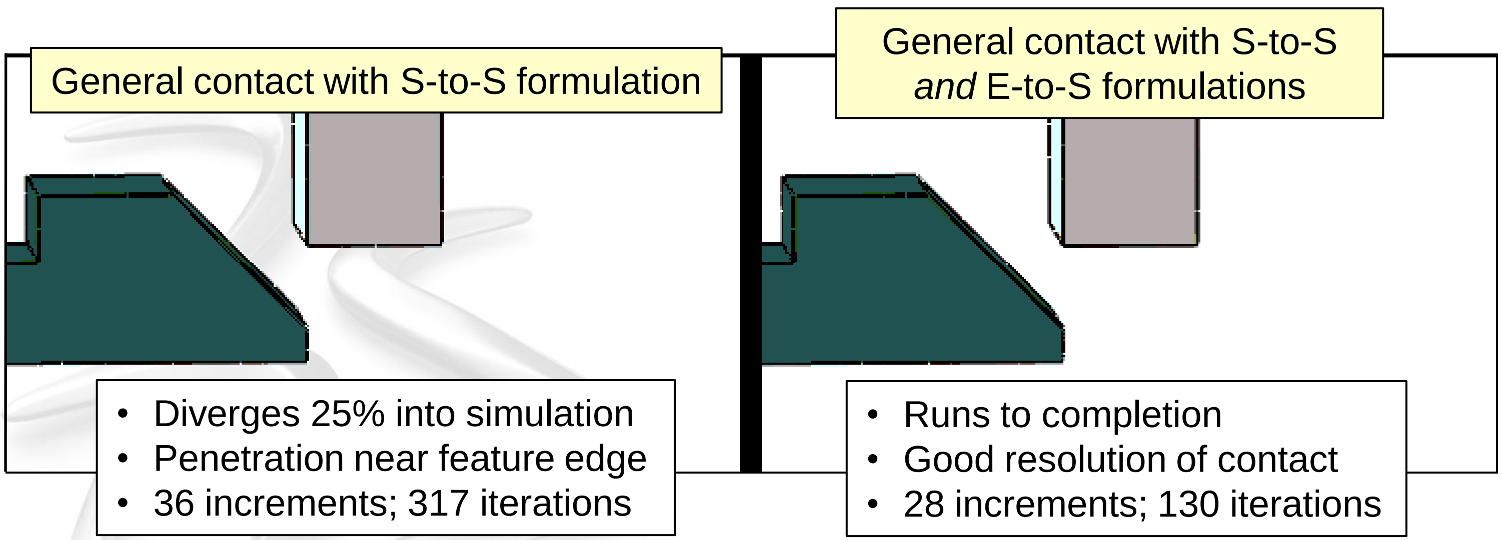


Surface-to-surface formulation:

- Penetrations averaged over finite regions
- Contact normal based on slave surface normal

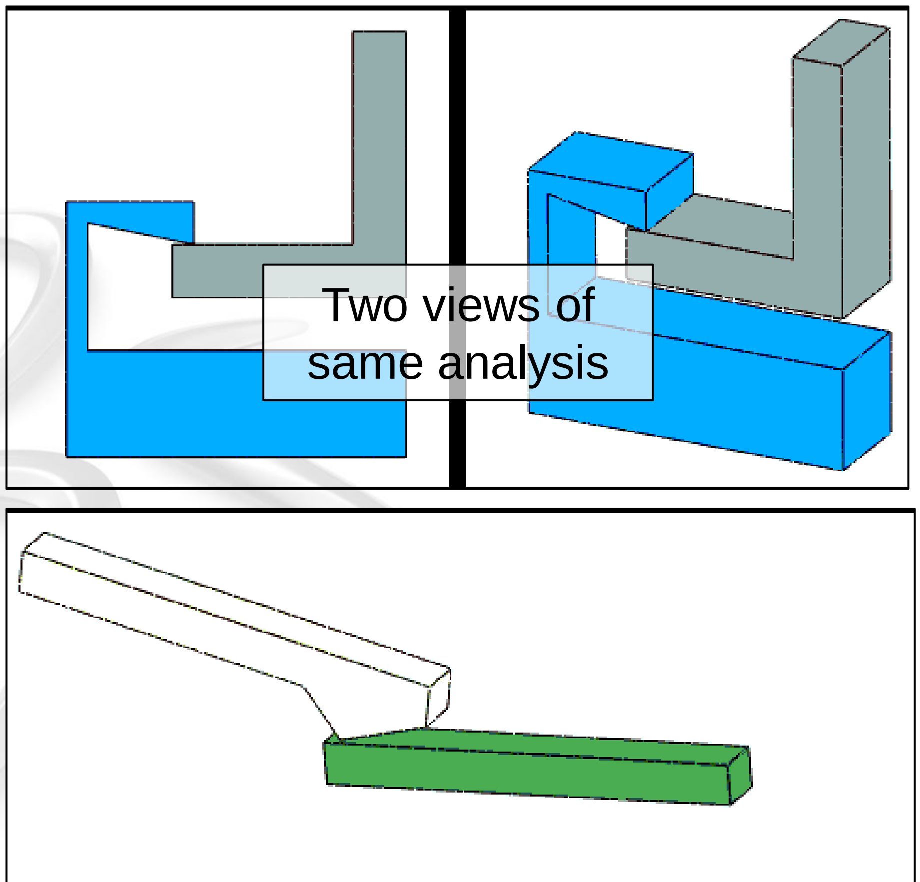
Contact Discretization

- **Supplemental edge-to-surface formulation for general contact:**
  - New in Abaqus/Standard 6.11; non-default in this first release
  - Good for enforcing certain contacts for which surface-to-surface formulation struggles



Contact Discretization

- **Supplemental edge-to-surface formulation for general contact:**
  - Additional examples



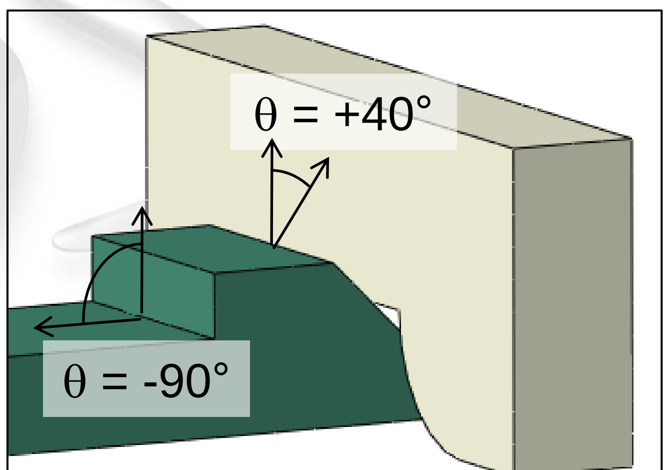
Contact Discretization

- **Limitations of edge-to-surface in Abaqus 6.11**
  - 3D, solid edges only; general contact only
  - Not supported in Abaqus/CAE
  - Not yet active by default
- **Keyword interface** (like Abaqus/Explicit)

  *Surface Property Assignment, Property=Feature Edge Criteria

  surface_name, *cut-off angle* (between facet normals, in degrees)

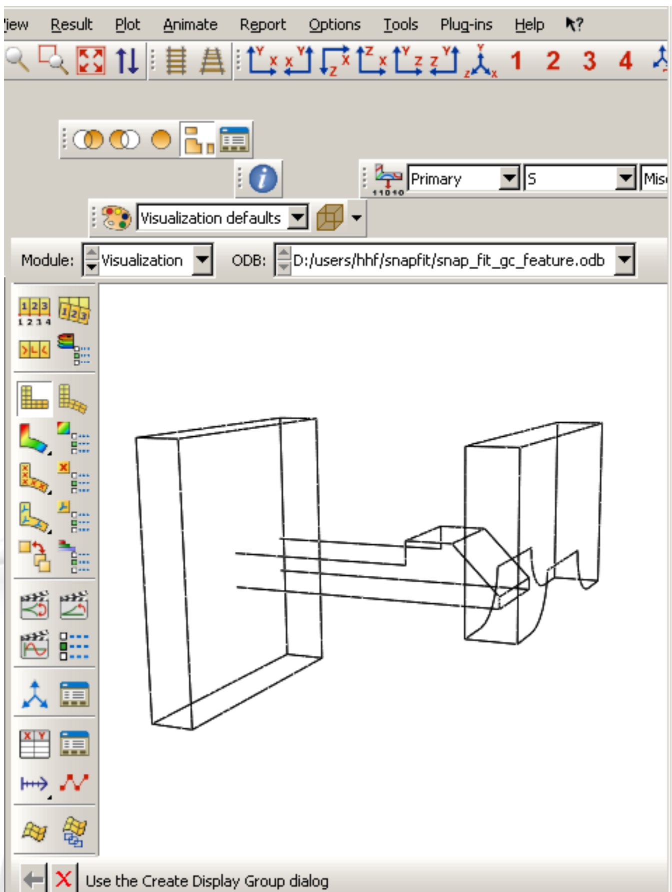
Include edges where $\theta \geq \theta_{cut\text{-}off}$

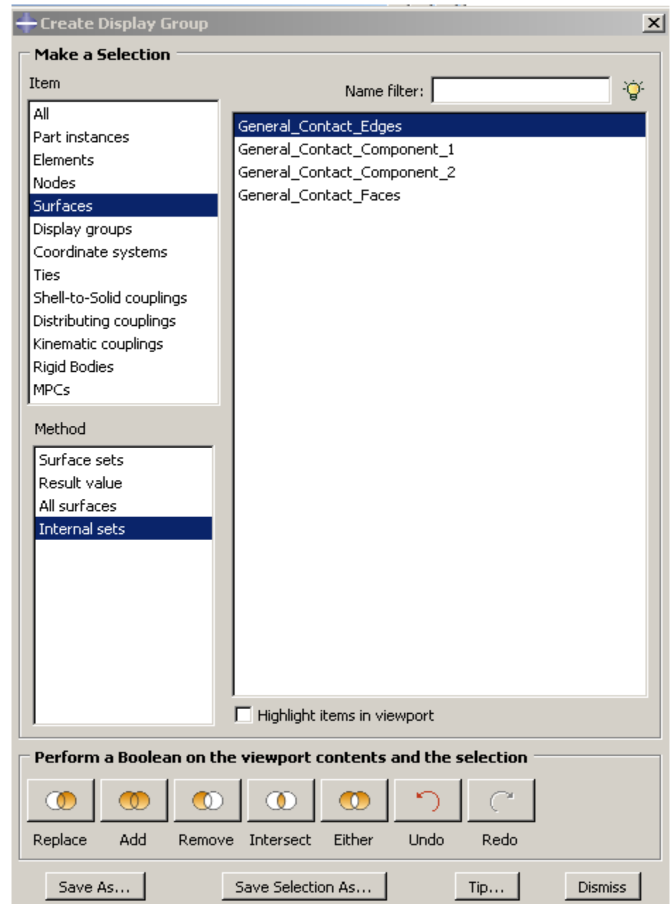


- Sign convention
  - + for exterior angles
  - - for interior angles
- θ is measured in undeformed configuration

Edge-to-surface contact

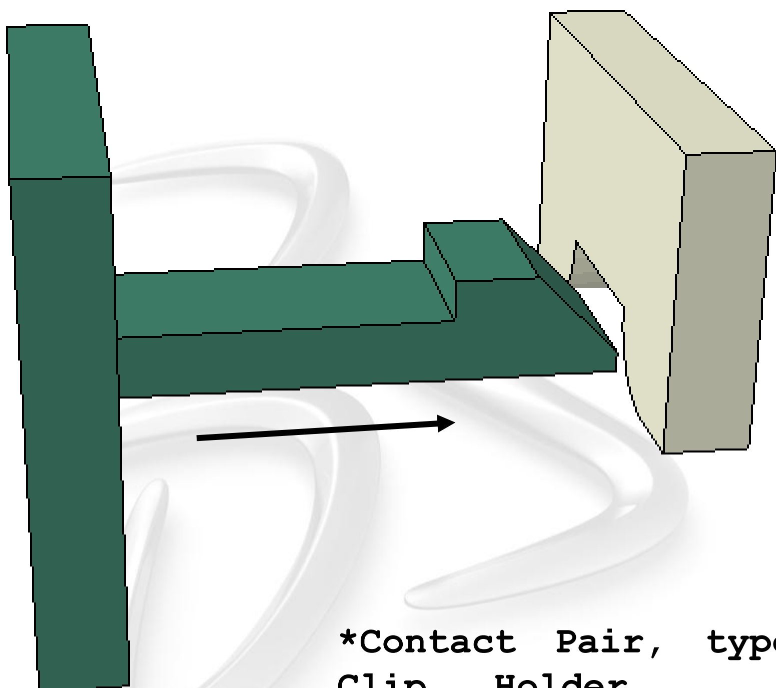
- **Internal surface “General_Contact_Edges”**
  - Edges of included surfaces that satisfy the feature edge criteria



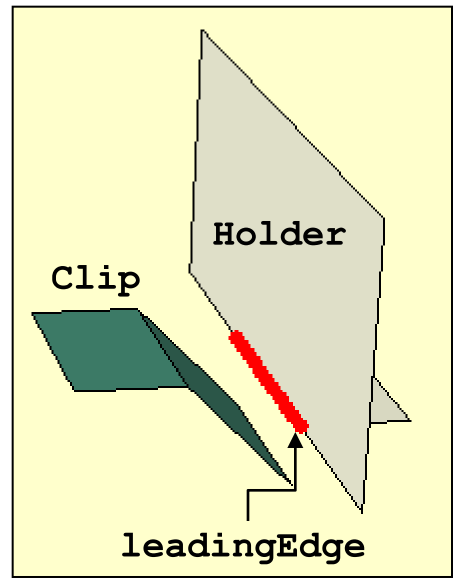


Contact Discretization

- **Modeling suggestion for *contact pair* models:**
  - Supplement surface-to-surface contact pairs with node-to-surface contact pairs involving significant feature edges



Contact surfaces




```
*Contact Pair, type=SURFACE TO SURFACE
Clip, Holder
```

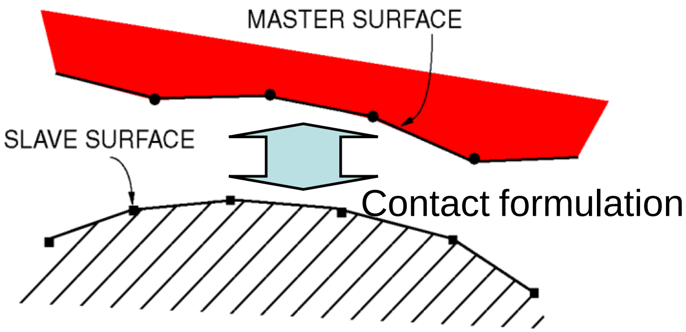
```
*Contact Pair, type=NODE TO SURFACE
leadingEdge, Clip
```

Contact Constraint Enforcement

SIMULIA

Contact Formulation Aspects

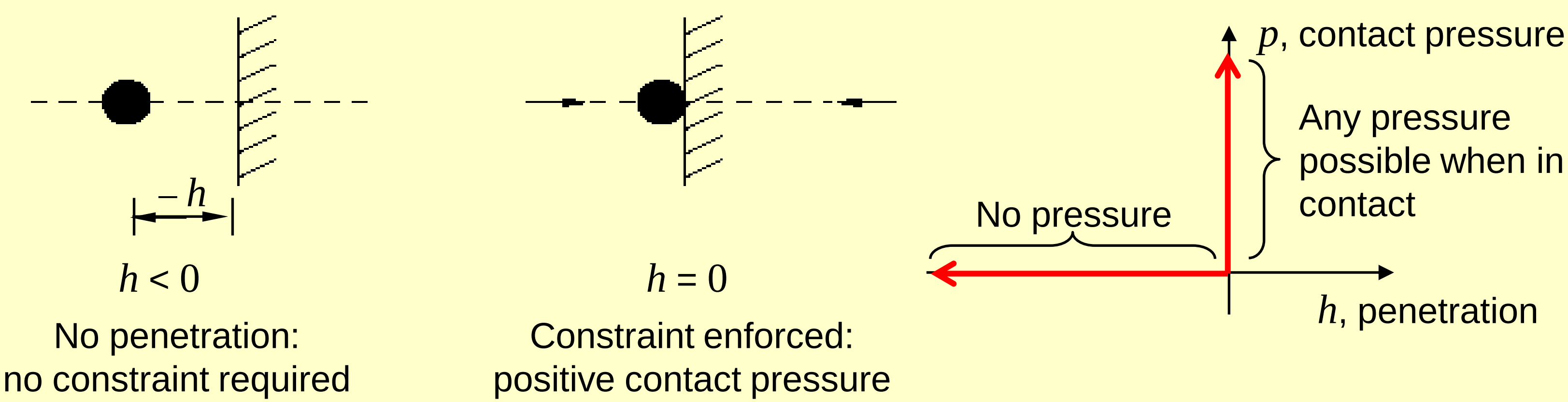
- **Discretization**
  - How are constraints **formed**?
    - For example, how to calculate gap or penetration distances from nodal positions
    - Node-to-surface or surface-to-surface
- **Enforcement**
  - How are constraints **enforced**?
    - For example, numerical method to resist penetrations
    - Direct (Lagrange multipliers) or penalty methods
- **Evolution of discretization**
  - How do constraints **evolve** upon sliding?
    - Rigorous, nonlinear evolution (“finite sliding”) vs. approximate (“small sliding”)



Constraint Enforcement

- **Strict enforcement**
  - Intuitively desirable
  - Can be achieved with Lagrange multiplier method in Abaqus/Standard
  - Drawbacks:
    - Can make it challenging for Newton iterations to converge
    - Overlapping constraints are problematic for equation solver
    - Lagrange multipliers add to equation solver cost

Physically “hard” pressure vs. penetration behavior



Constraint Enforcement

- **Direct enforcement**
  - Lagrange multiplier method
  - Constraint equations and Lagrange multipliers added to system of equations

Unconstrained system of equations

$$[K]\{u\} = \{f\}$$

Constraint equations added

$$\begin{bmatrix} K & B^T \\ C & 0 \end{bmatrix} \begin{Bmatrix} u \\ \lambda \end{Bmatrix} = \begin{Bmatrix} f \\ 0 \end{Bmatrix}$$

Vector of Lagrange multiplier degrees of freedom (constraint forces or pressures)

- **One per constraint**

Unitless distribution coefficients for constraint force

$$Ku + B^T\lambda = f$$

$$Cu = 0$$

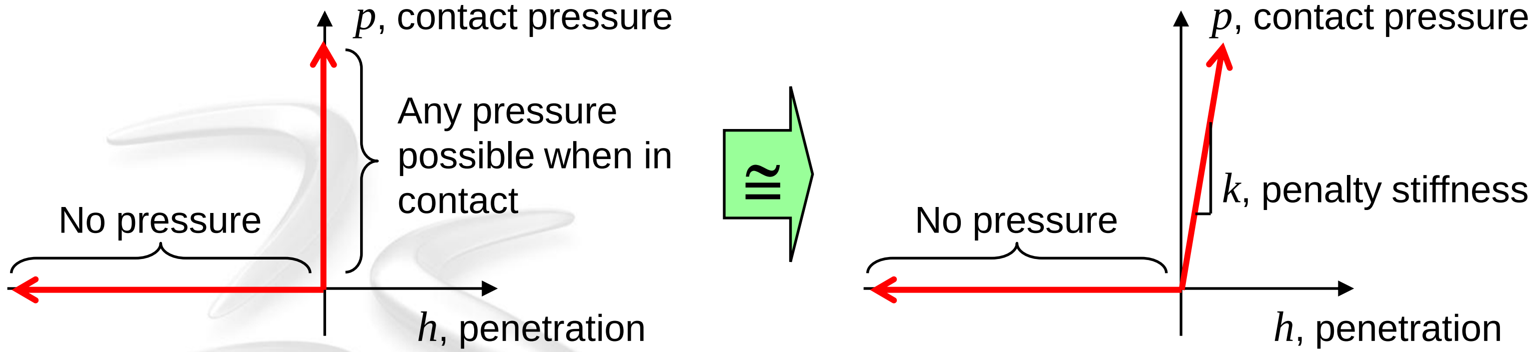
Unitless constraint coefficients

For symmetric constraints:

$$B = C$$

Constraint Enforcement

- **Penalty method**
  - Penalty method is a stiff approximation of hard contact



Strictly enforced hard contact

Penalty method approximation of hard contact

$$\left[K+K_p \right] \{ u \} = \{ f \}$$

Constraint Enforcement

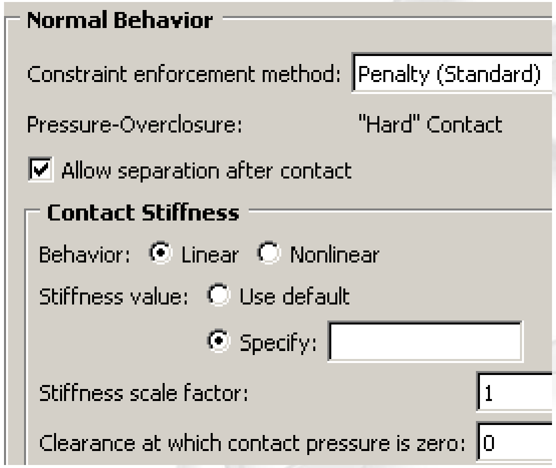
- **Pros and cons of penalty method**
  - Advantages:
    - Improved convergence rates
    - Better equation solver performance
      - No Lagrange multiplier degree of freedom unless contact stiffness is very high
    - Good treatment of overlapping constraints
  - Disadvantages:
    - Small amount of penetration
      - Typically insignificant
    - May need to adjust penalty stiffness relative to default setting in some cases

Constraint Enforcement

- **Default penalty stiffness**
  - Abaqus tries to find "happy medium" between:
    - Penalty stiffness too low:
      - Excessive penetrations
    - Penalty stiffness too high in Abaqus/Standard:
      - Convergence rates degrade
      - Lagrange multiplier degrees of freedom needed to avoid ill-conditioning
    - Penalty stiffness too high in Abaqus/Explicit:
      - Significant reduction in stable time increment
  - Default penalty stiffness is based on representative stiffness of underlying elements
    - Scale factor applied to this representative stiffness to set default penalty stiffness; magnitude higher in Abaqus/Standard than in Abaqus/Explicit

Constraint Enforcement

- **Options to scale the penalty stiffness are available:**
  - For cases in which default penalty stiffness not suitable
  - Order-of-magnitude changes recommended
  - If scale factor > 100, Abaqus will automatically invoke a variant of method that uses Lagrange multipliers to avoid ill-conditioning issues



Keyword interface

***SURFACE INTERACTION**
***SURFACE BEHAVIOR, PENALTY**
penalty stiffness, clearance offset, scale factor (all optional)
:
***STEP**
:
CONTACT CONTROLS, STIFFNESS SCALE FACTOR=*value

Step dependent (careful!)

Multiplicative!

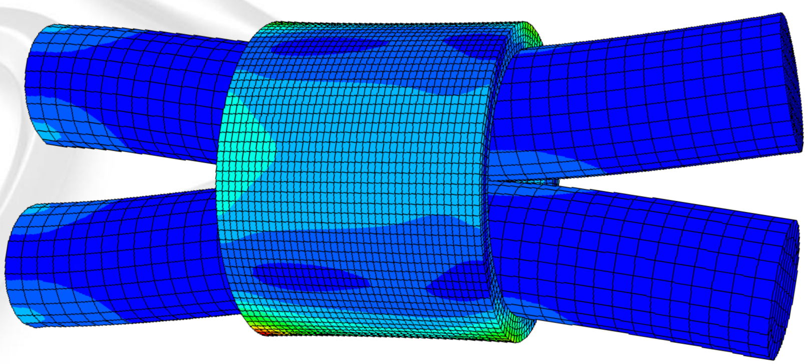
Constraint Enforcement

- **Penalty stiffness magnitude**
  - Stiff or blocky problems:
    - The default penalty stiffness generally produces results comparable in accuracy with those obtained with direct method
    - Usually requires less memory and CPU time
  - Bending-dominated problems:
    - The default penalty stiffness can often be scaled back by two orders of magnitude without any significant loss of accuracy
    - Scaling back penalty stiffness for bending-dominated problems sometimes increases convergence rate

Constraint Enforcement

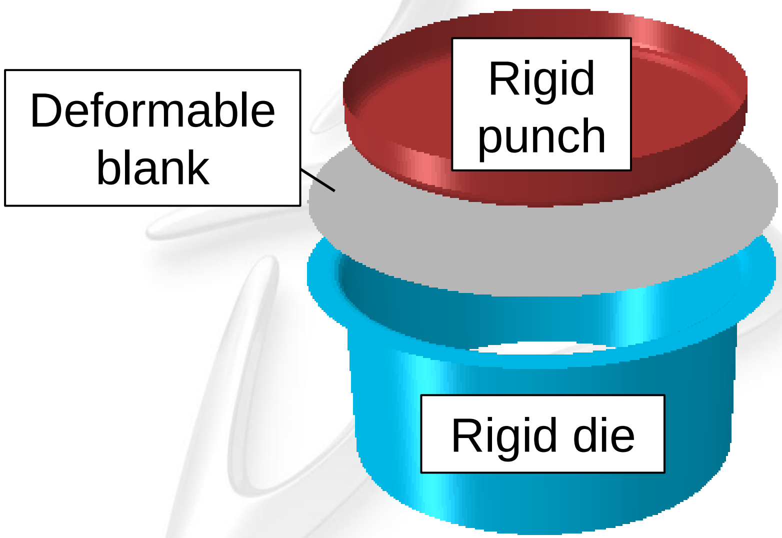
- **Example**

Constraint enforcement	Maximum penetration	Max. Mises stress	# Iters.	Solver FLOPs
Default penalty	0.4% of collar elem. dimension	6.166E4	50	2.8E10
Lagrange multiplier	0	6.173E4	57	3.6E10

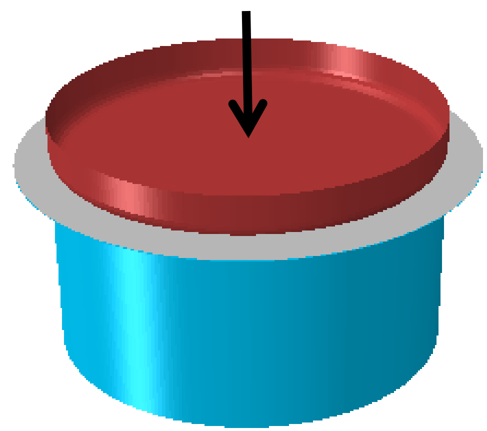


Constraint Enforcement

- **First load increment of sheet forming example**
  - Numerically challenging due to:
    - Low-energy deformation modes for flat, unstretched sheet
    - Possibility of material yielding during Newton iterations
      - Even if converged solution for increment does not yield
    - Dramatic change in contact status distribution



Components shown separated

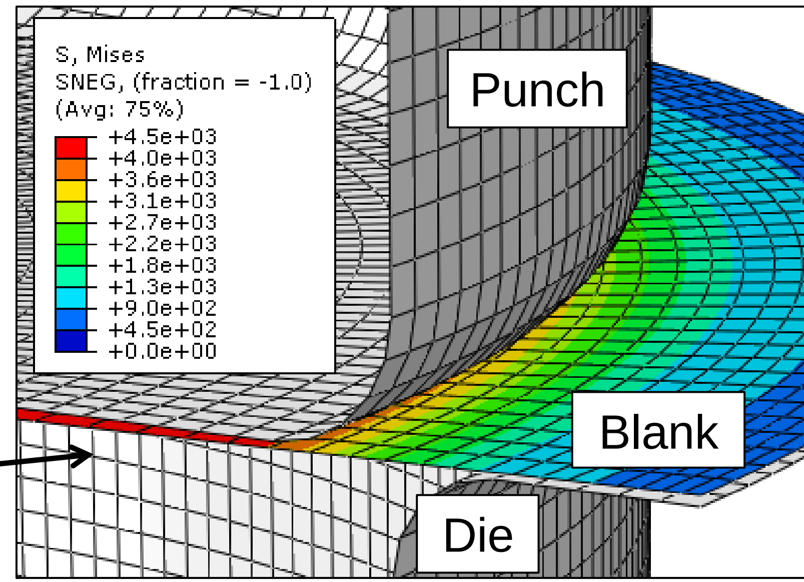


Actual initial configuration (touching)

Constraint Enforcement

- **First load increment of sheet forming example (cont.)**
  - Convergence behavior *without stabilization*

Constraint enforcement	First Increment (without stabilization)
Lagrange multiplier	Does not converge
Default penalty	Does not converge
Penalty scale factor of 10^{-5}	Converges in 5 Newton iterations



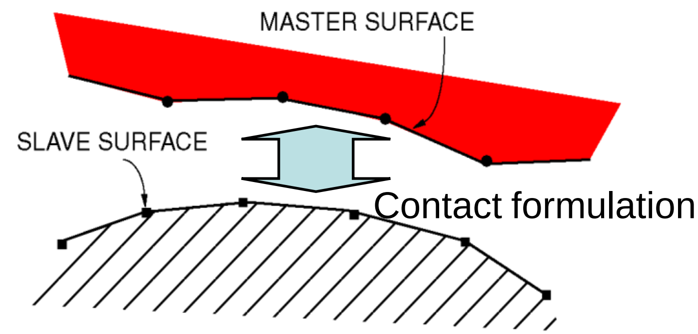
Next steps for analysis would be to:

- Increase penalty stiffness to improve accuracy
  - Easier once approximate solution is found
- Apply remaining load

Constraint Evolution upon Relative Sliding between Bodies

Contact Formulation Aspects

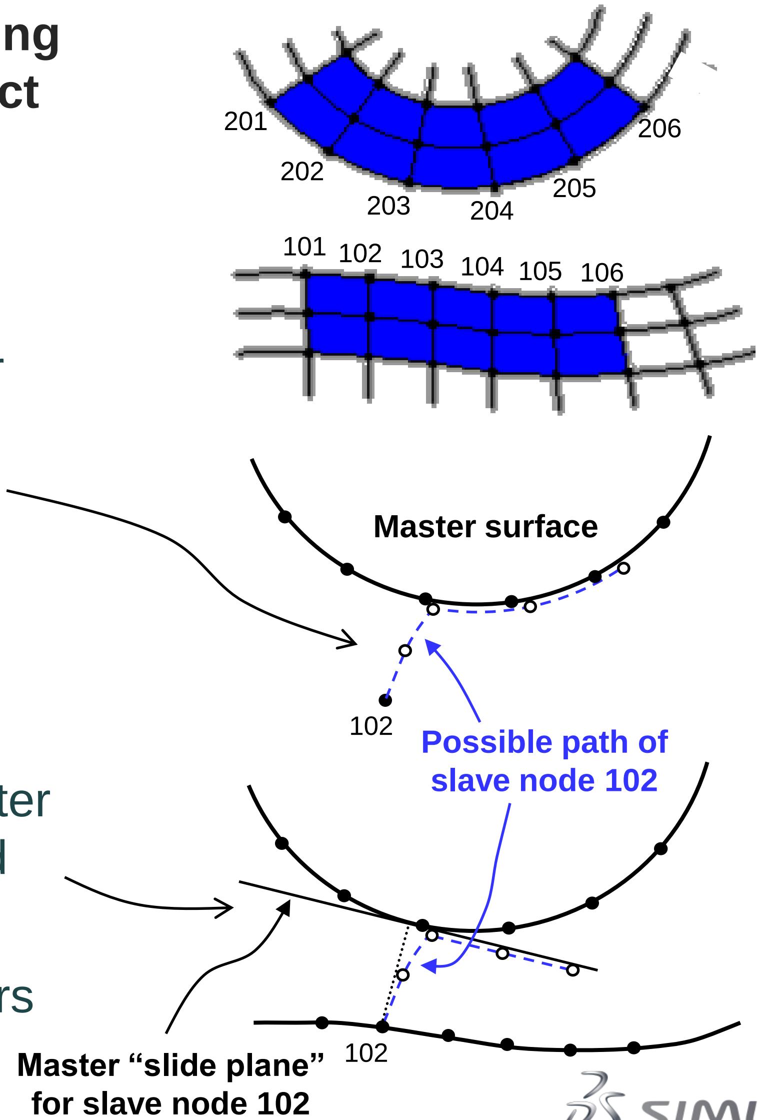
- **Discretization**
  - How are constraints **formed**?
    - For example, how to calculate gap or penetration distances from nodal positions
    - Node-to-surface or surface-to-surface



- **Enforcement**
  - How are constraints **enforced**?
    - For example, numerical method to resist penetrations
    - Direct (Lagrange multipliers) or penalty methods
- **Evolution of discretization**
  - How do constraints **evolve** upon sliding?
    - Rigorous, nonlinear evolution (“finite sliding”) vs. approximate (“small sliding”)

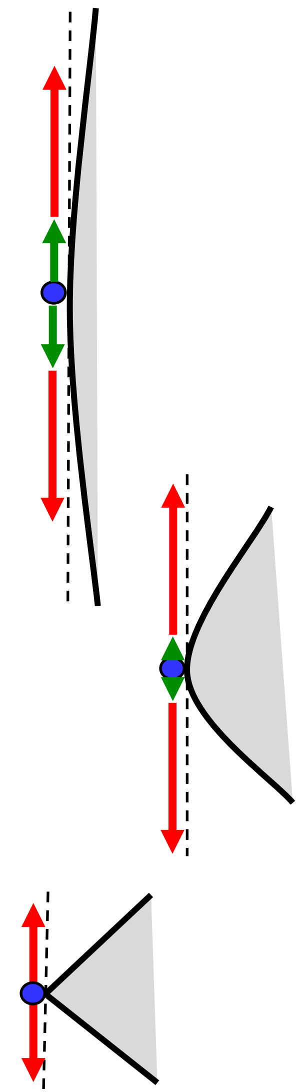
Relative Sliding between Bodies

- **Abaqus offers finite- and small-sliding versions of S-to-S and N-to-S contact formulations**
  - Finite-sliding formulation: General applicability
    - Point of interaction on master surface updated using true representation of master surface
  - Small-sliding formulation: Approximation intended to reduce solution cost; limited applicability
    - Planar representation of master surface per slave node based on initial configuration
    - Only available for contact pairs (and not self-contact or general contact)



Small-Sliding Approximation

- **Every slave node interacts with its own local slide plane**
  - In 2-D/axisymmetric it is depicted as line
  - Assumes that relative motion per slave node remains small compared to:
    - Local curvature of master surface (see diagrams)
    - Facet sizes of master surface

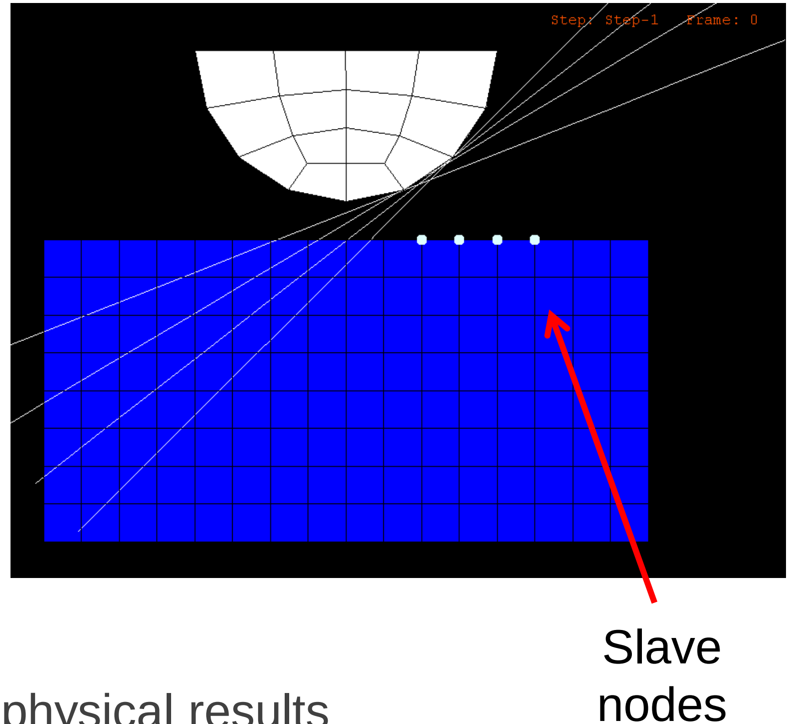


- **Advantage: Less nonlinearity**
  - Potential for reduced cost per iteration and finding a converged solution in fewer iterations

- **Disadvantage: *Results can be nonphysical* if relative tangential motion does not remain small**
  - It is the user's responsibility to ensure that the assumption is not violated

Small-Sliding Approximation

- **Example of nonphysical behavior with small-sliding formulation**
  - Approximately cylindrical surface assigned to act as **master** surface
  - Slide planes represented by white lines in animation
  - Slide planes translate with punch as it moves to the right

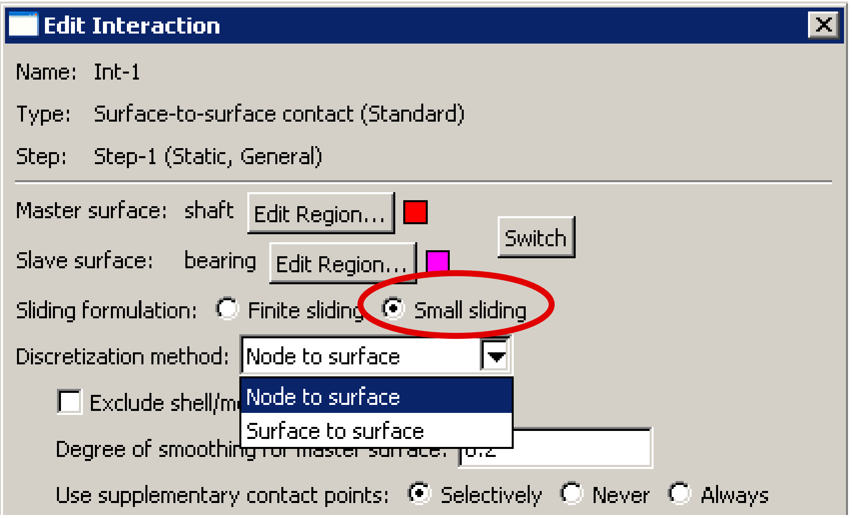


- **Key points:**
  - Small-sliding formulation can cause nonphysical results
    - Obviously incorrect response in this example
    - Not always obvious
  - Use finite-sliding formulation if you do not want to worry about whether small-sliding assumptions are appropriate!

Small-Sliding Approximation

- **Invoking small-sliding (contact pairs only):**

***CONTACT PAIR, SMALL SLIDING**



Formulation Summary

- **Good formulation characteristics (for accuracy, robustness, and generality)**
  - Accurate representation of surface geometry
    - Slave surface: Not just a collection of points — S-to-S
    - Master surface: Not approximated as flat per slave node — finite-sliding
    - Geometric corrections: Reduce discretization error — available for S-to-S
  - Distribution of nodal forces consistent with underlying element formulation
    - Ability to satisfy "patch tests" for contact — S-to-S
  - Continuity in contact forces upon sliding — S-to-S
  - Individual constraint stresses should oppose penetration (and sliding)
    - Nontrivial aspect for some quadratic element types — S-to-S
  - Avoid "over-constraints" and "under-constraints" — master-slave roles
    - Generally, number of contact constraints in an active contact region should equal number of nodes of the more refined surface in that region
  - Small amount of numerical "softening" — penalty method
  - Robust contact search algorithm to avoid missing contacts, etc. — finite-sliding
  - Special treatment of feature edges — E-to-S

Formulation Summary

- **Available formulations for general contact and contact pairs in Abaqus/Standard**

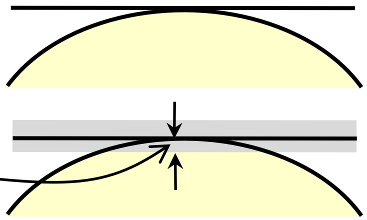
	Modeling Approach	
Formulation Aspect	**General Contact**	**Contact Pairs**
Contact Discretization	Primary: Surface-to-surface Suppl.: Edge-to-surface	Default: Node-to-surface Optional: Surface-to-surface
Contact Enforcement	Default: Penalty Optional: Direct	N-to-S default: Direct S-to-S default: Penalty
Constraint Evolution upon Sliding	Finite sliding	Default: Finite sliding Optional: Small sliding approx.

Refers to defaults for keyword input file:

- These defaults were established prior to implementation of surface-to-surface discretization and penalty methods
- These are not the defaults for contact pairs created in Abaqus/CAE based on initial proximity

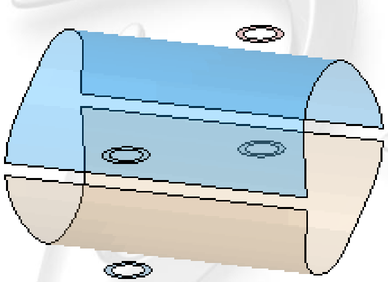
Formulation Summary

- **Common issues when converting contact pair models to general contact**
  - Most issues are related to <u>initial</u> <u>overclosures</u>
  - General contact accounts for shell/membrane thickness
    - Finite-sliding, node-to-surface contact pairs do not

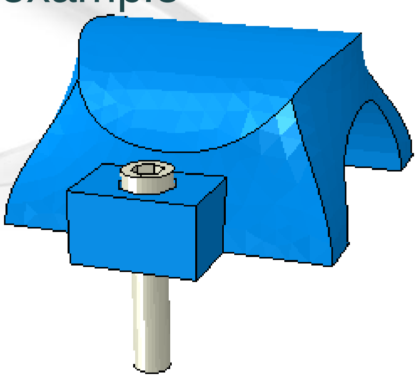


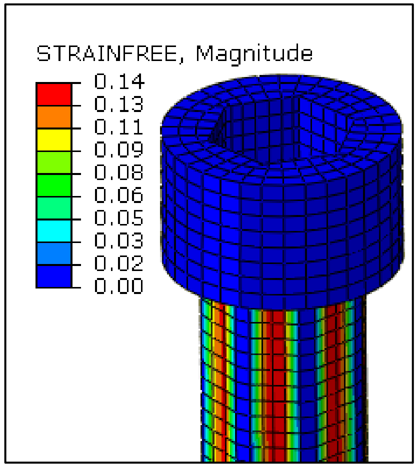
Initial penetration if shell thickness considered

  - General contact typically considers all exposed surfaces
    - Contact pairs may not be defined on some penetrated regions
    - Recall clamp example



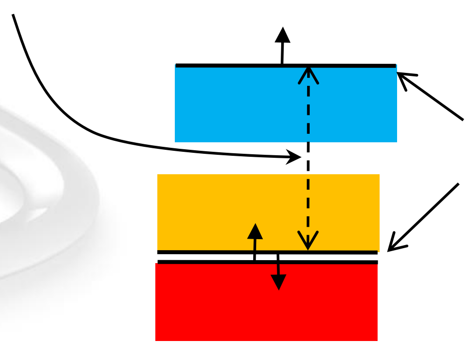
Slave surfaces of contact pairs





Formulation Summary

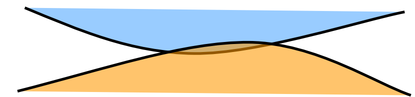
- **Common issues when converting to general contact (cont.)**
  - Different default treatment of initial overclosures
    - Contact pairs
      - Initial overclosures treated as interference fits by default
    - General contact
      - Small initial overclosures resolved with strain-free adjustments
      - Large initial overclosures assumed nonphysical/unintended



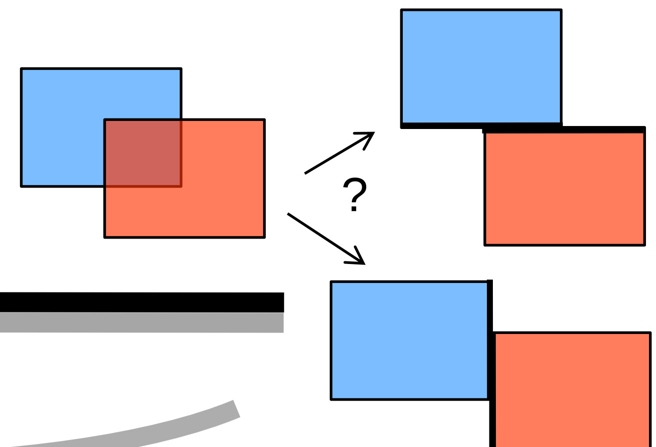
- **Further discussion on next slide**

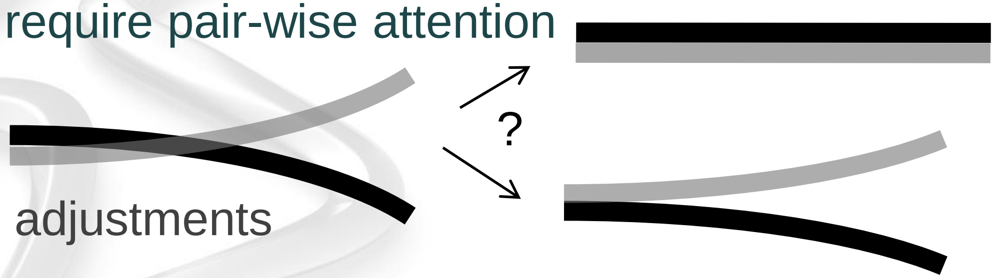
Formulation Summary

- **Comments on initial overclosures** (more comments later)
  - User responsible for directing the treatment of initial overclosures
    - Choice of whether to resolve them with or without strains requires user judgment



  - Common characteristics of interference fits
    - Overclosure distance may be large
    - Limited to specific interfaces
    - Often require pair-wise attention





  - Strain-free adjustments
    - Intended to resolve *small* overclosures (e.g., due to faceted representation of curved surfaces
    - For small overclosures, automated algorithm can determine which nodes to move and where to move them

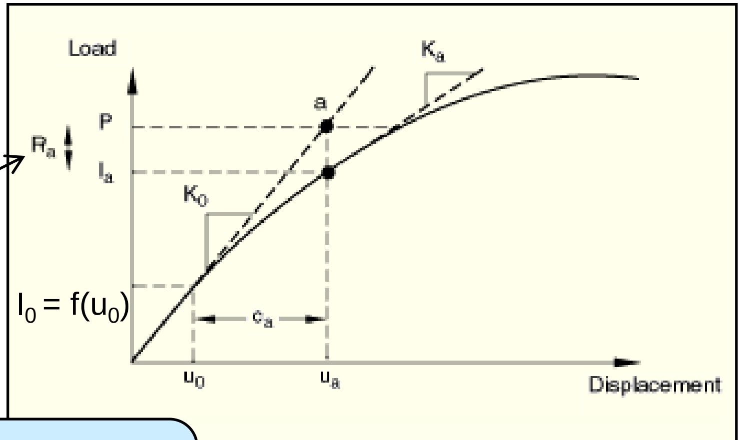
Incrementation and Newton Iterations (Abaqus/Standard)

SIMULIA

Incrementation and Newton Iterations

- **Newton method: Iterative method used to solve nonlinear problems**

“Residual force” after 1^{st} iteration



Given:

- Starting displacement, u_0
- Desired load, “P”
- Ability to evaluate f(u) and K(u)

Find:

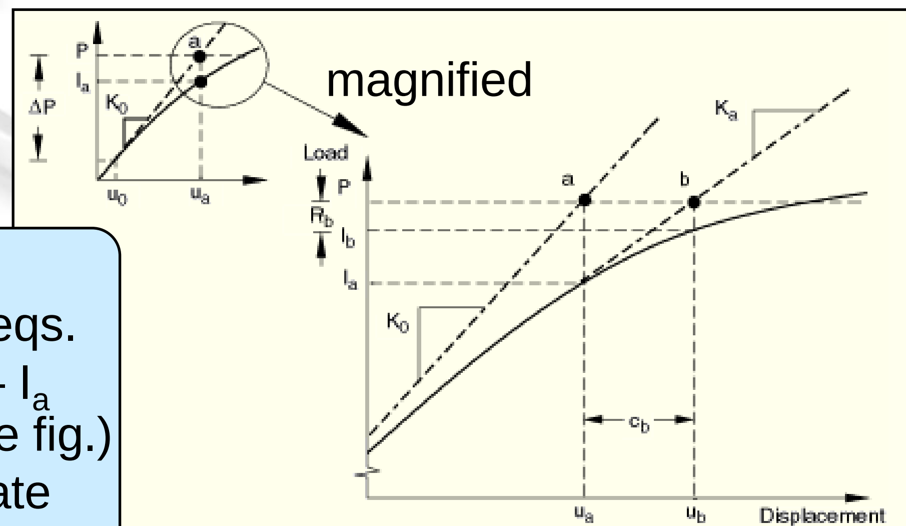
- Displacement solution, u_s, such that $f(u_s) = P$

Iteration 1

- System of eqs. $K_0 \, \Delta u = P - I_0$
- $\Delta u = c_a$ (see fig.)
- New estimate $u_a = u_0 + c_a$

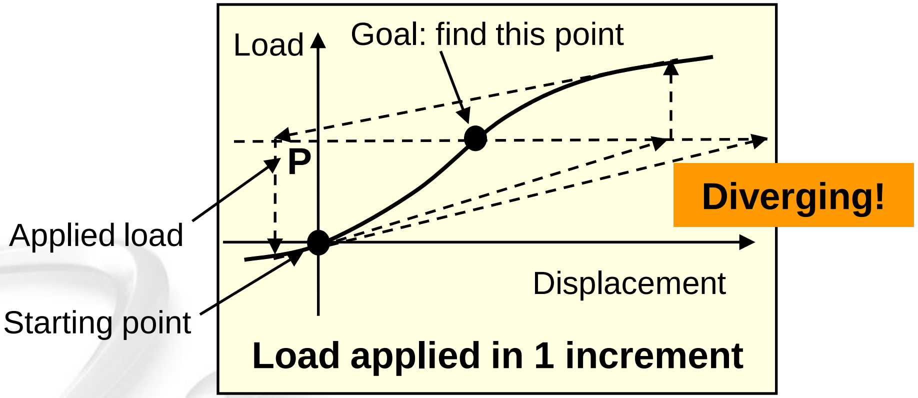
Iteration 2

- System of eqs. $K_a \, \Delta u = P - I_a$
- $\Delta u = c_b$ (see fig.)
- New estimate $u_b = u_a + c_b$

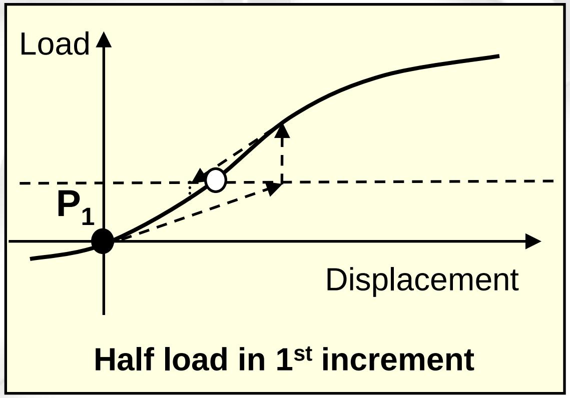


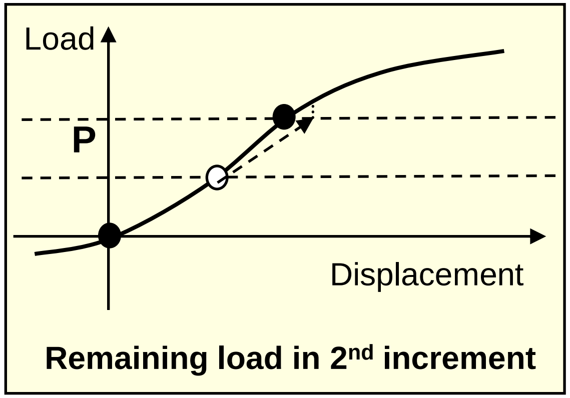
Incrementation and Newton Iterations

- **The Newton method, however, is not guaranteed to converge**
  - Example in which Newton iterations diverge:



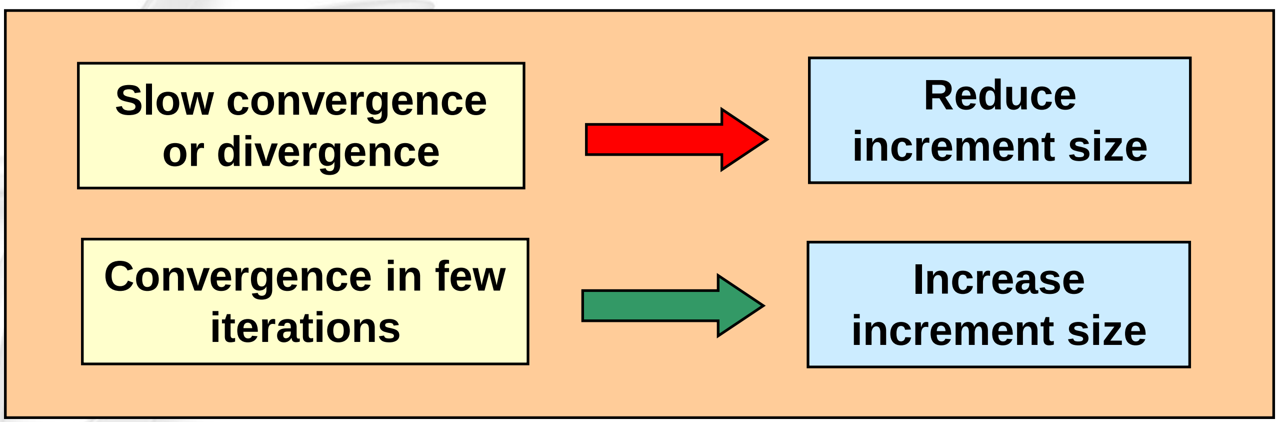
- **Increase the likelihood of convergence by decreasing load increment**
  - Use multiple load increments to achieve desired total load





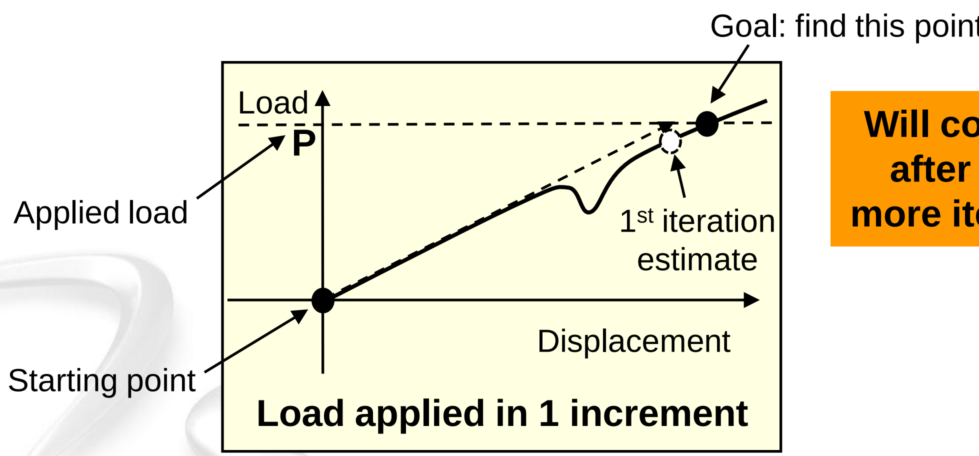
Incrementation and Newton Iterations

- **Abaqus automatically adjusts the load increment size**
  - Goal: Find converged solution robustly and efficiently with respect to the number of iterations
  - Basic idea: Track convergence rate to determine when to increase or decrease load increment size
    - User suggests increment size; Abaqus tries to optimize it



Incrementation and Newton Iterations

- **Occasionally, may "jump across" an unstable region of load-displacement curve with larger increments!**

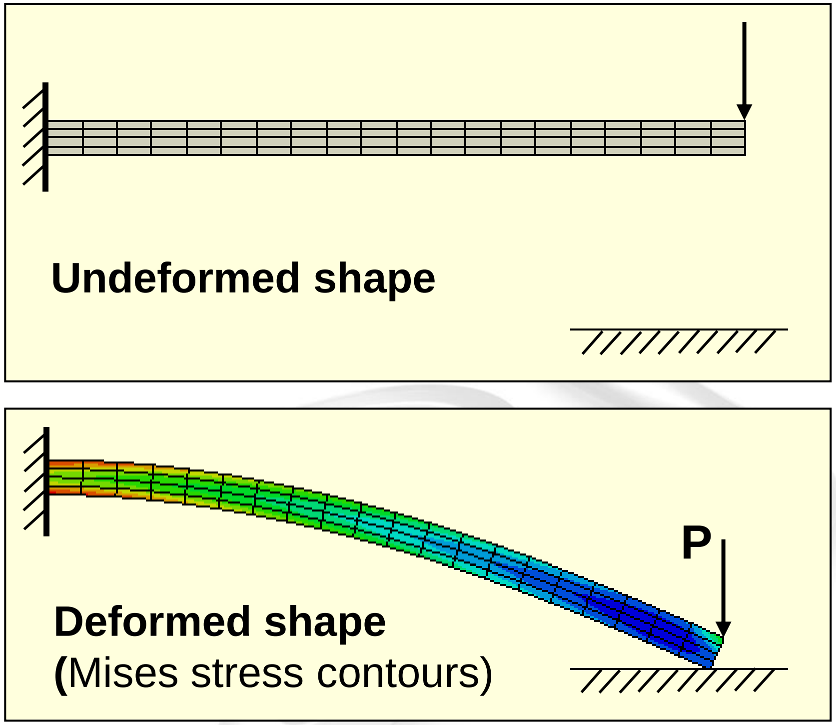


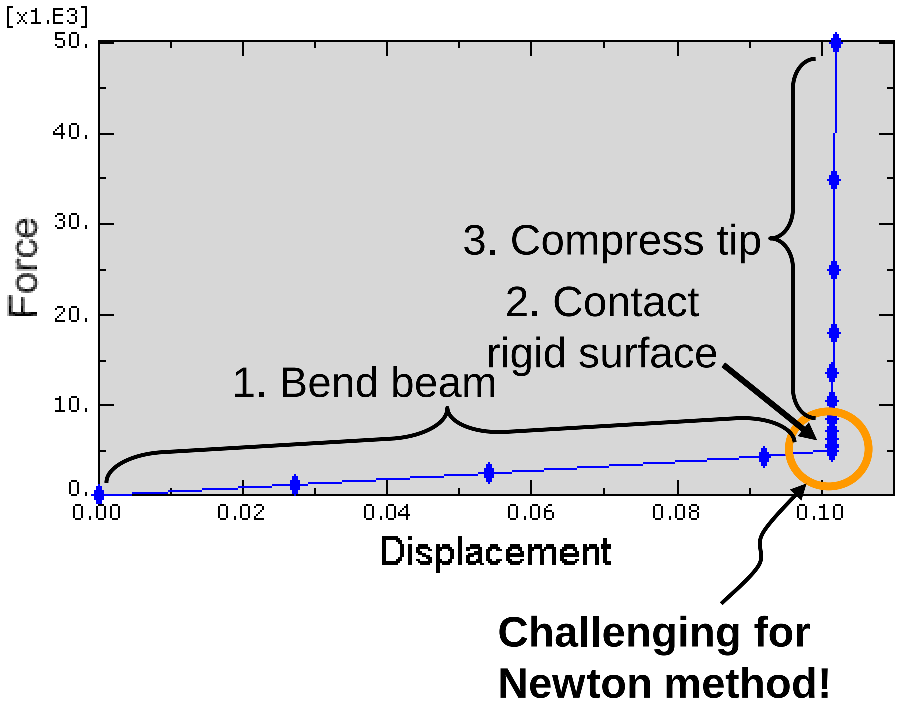
Will converge after 1 or 2 more iterations

  - Applying same total load over multiple increments would likely lead to converge failure in this example
    - Not particularly common
  - Recommendation: resolve instability rather than try to "jump past it"

Incrementation and Newton Iterations

- **Contact causes kinks in the load vs. displacement curve**
  - There is a slope discontinuity upon change in contact status
  - As a result, contact changes interrupt overall convergence rate tracking





Incrementation and Newton Iterations

- **"Severe discontinuity iterations" (SDIs)**
  - An SDI is an iteration during which contact constraints change state
    - Open/closed, stick/slip (active or inactive)
    - The logic to adjust the increment size treats SDIs separately

Status (`.sta`) file for beam contact example:

STEP	INC	ATT	SEVERE DISCON ITERS	EQUIL ITERS	TOTAL ITERS	TOTAL TIME/ FREQ	STEP TIME/LPF	INC OF TIME/LPF
1	1	1U	1	0	1	0.000	0.000	0.1000
1	1	2	0	3	3	0.0250	0.0250	0.02500
1	2	1	0	2	2	0.0500	0.0500	0.02500
1	3	1	0	4	4	0.0875	0.0875	0.03750
1	4	1U	1	0	1	0.0875	0.0875	0.05625
1	4	2	1	2	3	0.102	0.102	0.01406
1	5	1U	1	0	1	0.102	0.102	0.02109
1	5	2	0	2	2	0.107	0.107	0.005273
1	6	1	0	1	1	0.115	0.115	0.007910
1	7	1	0	1	1	0.127	0.127	0.01187
1	8	1	0	1	1	0.144	0.144	0.01780
1	9	1	0	1	1	0.171	0.171	0.02670
				1	1	0.211	0.211	0.04005
				1	1	0.271	0.271	0.06007
				1	1	0.361	0.361	0.09010
				1	1	0.496	0.496	0.1352
				1	1	0.699	0.699	0.2027
				1	1	1.00	1.00	0.3008

Separate iteration counts for SDIs and non-SDIs

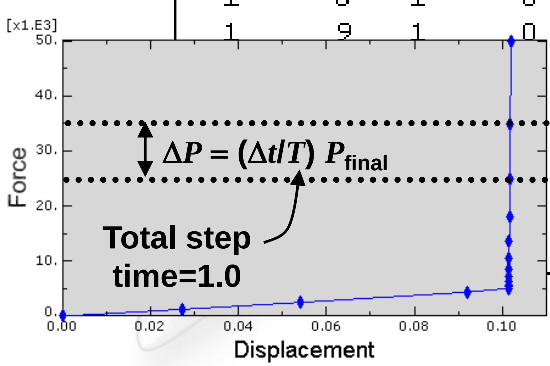
1^{st} attempt did not converge⇒reduce Δt

2^{nd} attempt at first incr. converges

Increase Δt due to fast convergence

Converged incr. with contact activated

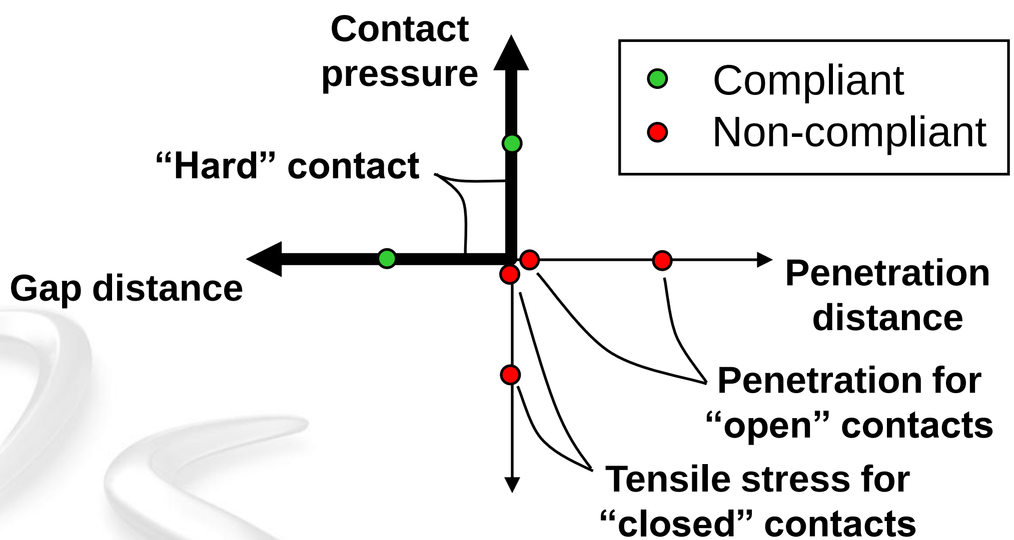
Trend toward larger Δt after contact is established



SIMULIA

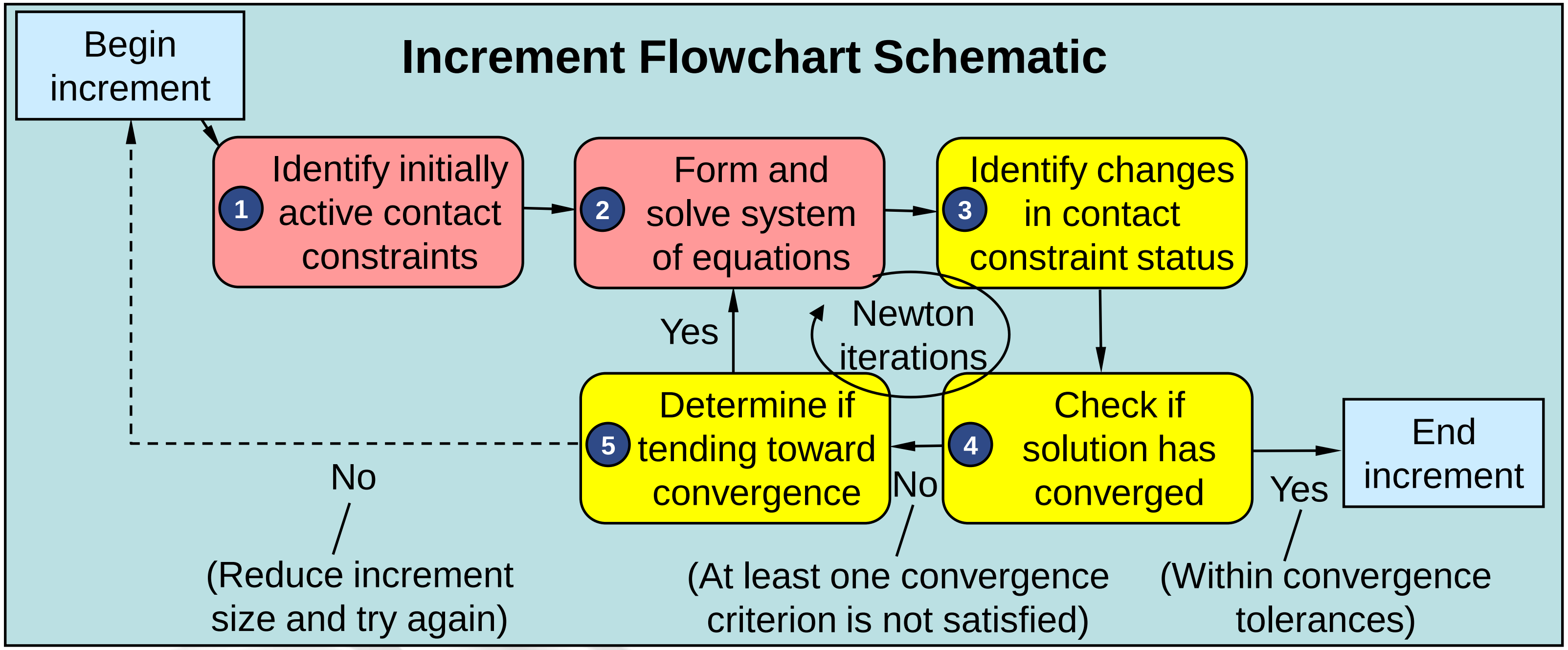
Incrementation and Newton Iterations

- “Hard” contact pressure vs. overclosure:



- Default behavior: SDIs do not block convergence
  - “Convert SDI”: Small penetrations/tensile stresses trigger contact status changes (and SDIs) but do not necessarily block convergence
- Without “Convert SDI”
  - Contact status changes block convergence
  - Some older contact controls (e.g., “Automatic Tolerances) avoid contact status changes upon small noncompliance (not recommended)

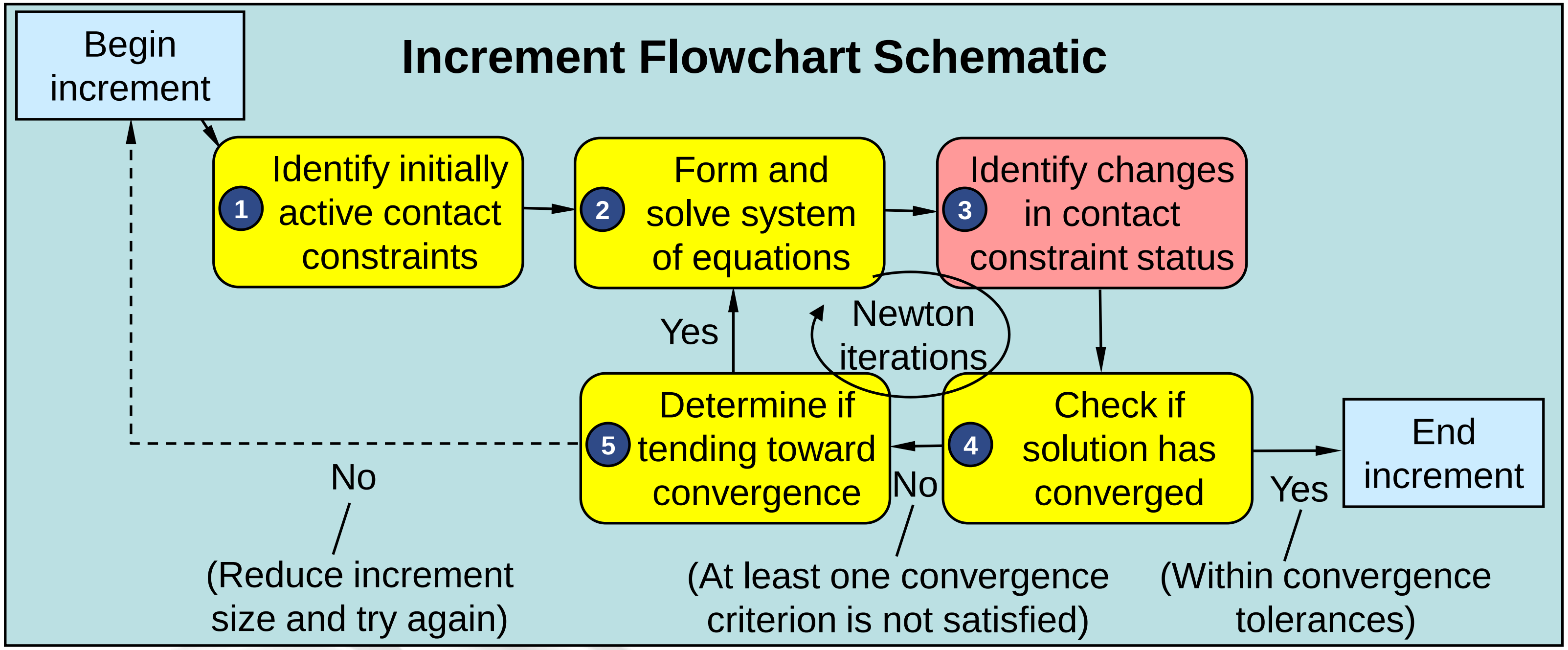
Incrementation and Newton Iterations



1. Determine the initial contact state at each point (closed or open)
   - For first increment of a step, based on initial model state
   - Otherwise, based on solution extrapolation (if any)

2. Form the system of equations with contact constraints imposed, then pass through the equation solver

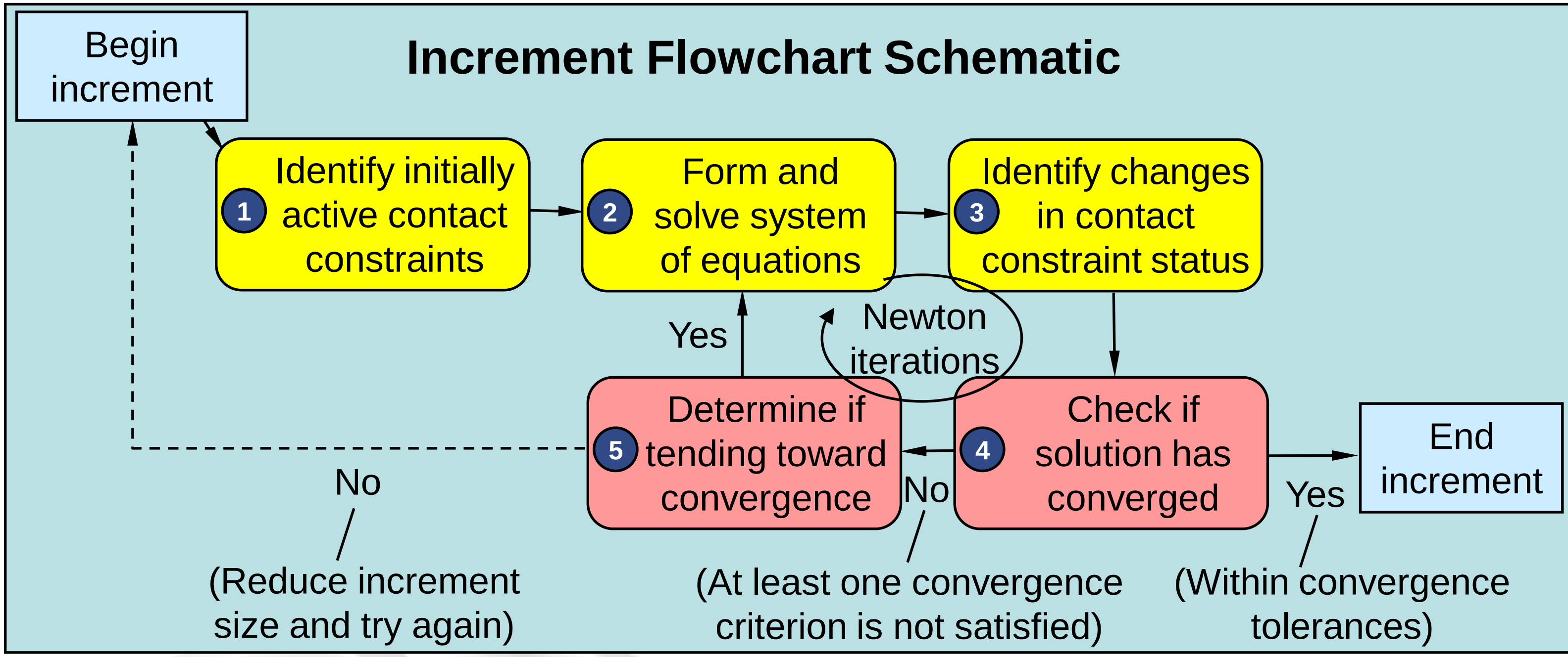
Incrementation and Newton Iterations



3 Are contact pressures, clearances, frictional stresses, and sliding increments consistent with the assumed contact state?

- Contact status changes (open/closed or stick/slip) often cause significant changes to the system of equations
- Iterations with contact status changes are flagged as severe discontinuity iterations (SDIs)

Incrementation and Newton Iterations



4 Has convergence been achieved?

- Convergence criteria ensure small force residuals, small solution corrections, and small contact incompatibilities

5 If convergence is not achieved, is it likely to be achieved?

- Abaqus determines whether to continue iterating or to reattempt the increment with a smaller load increment based on trends in recent iterations

Lecture 3 Summary

Review of Topics Discussed in this Lecture

- **Title: Numerical Methods**
- **Contact Formulation Aspects**
  - Contact Discretization
  - Contact Enforcement Methods
  - Contact Tracking
- **Incrementation and Newton Iterations**

Contact Output and Diagnostics Tools

Lecture 4

Overview

- **Output of Contact Results**
- **Contact Pressure Accuracy**
- **Contact Diagnostics (Visual)**
- **Contact Diagnostics (Text)**
  - A high-level understanding of the numerical methods that Abaqus uses for contact (subject of previous lecture) can be helpful for:
    - Understanding diagnostic output
    - Troubleshooting convergence problems
    - Overcoming solution noise
  - Tools are available in Abaqus/CAE to visualize contact output
    - Greatly simplifies the troubleshooting process

Output of Contact Results

SIMULIA

Output of Contact Results

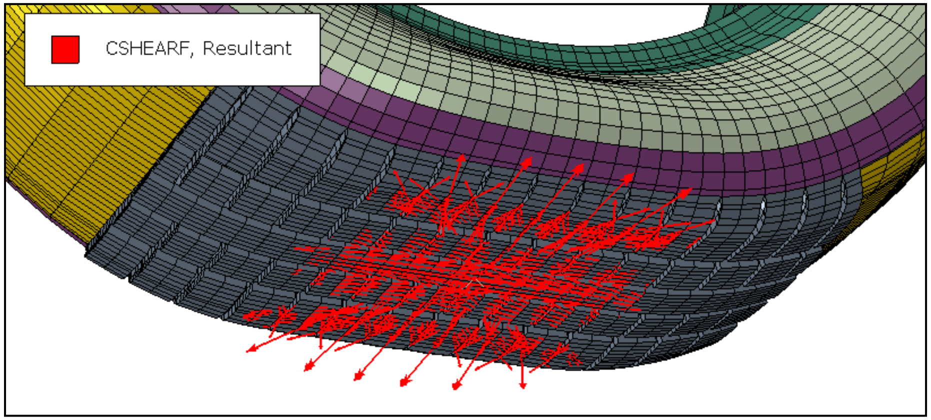
- **Output files**
  - Output database (`.odb`) file
    - Used for postprocessing with Abaqus/Viewer
    - By default, ODB output includes preselected variables
  - Data (`.dat`) file
    - Printed output; no output by default
  - Results (`.fil`) file
    - Used for postprocessing with third-party postprocessors; no output by default
- **Output variable types**
  - Nodal variables
  - Whole surface variables

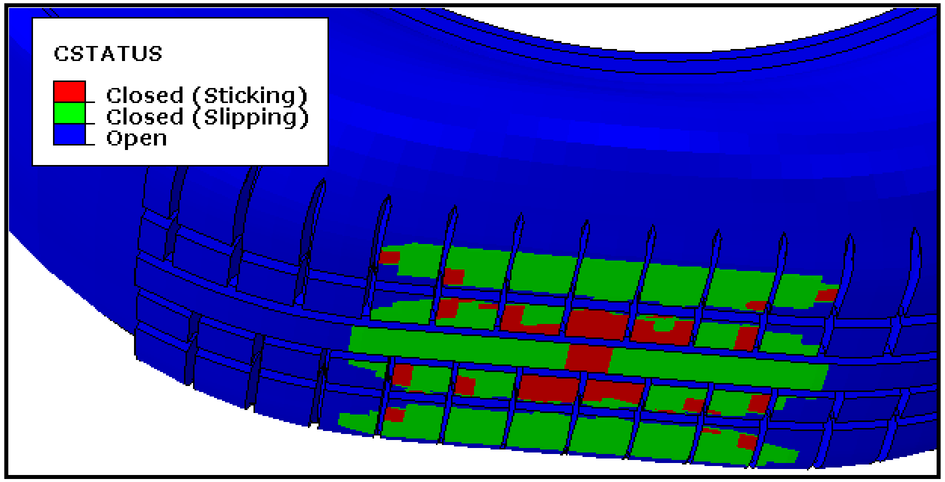
Output of Contact Results

- **Nodal output to the ODB file**
  - Default nodal contact output to ODB file includes the following variables:
    - Contact stresses (CSTRESS):
      - Contact pressure CPRESS
      - Frictional shear stresses CSHEAR1 and CSHEAR2
    - Contact displacements (CDISP):
      - Contact openings: COPEN
      - Accumulated relative tangential motions: CSLIP1, CSLIP2
  - CSHEAR2 and CSLIP2 are provided only in three-dimensional problems
  - Above output available as both field and history data

Output of Contact Results

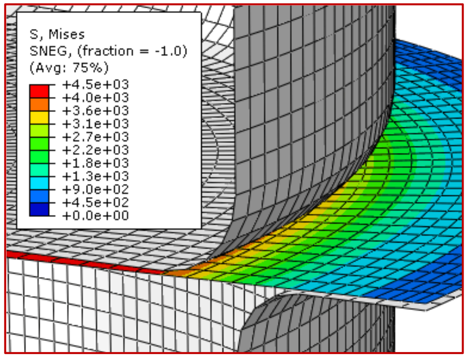
- **Additional nodal output to .odb**
  - Contact nodal force vectors (CFORCE⇒
    CNORMF & CHEARF)
  - Nodal areas associated with active contact constraints (CNAREA)
  - Contact status (CSTATUS)
    - Enables contour plots of sticking/slipping/open status

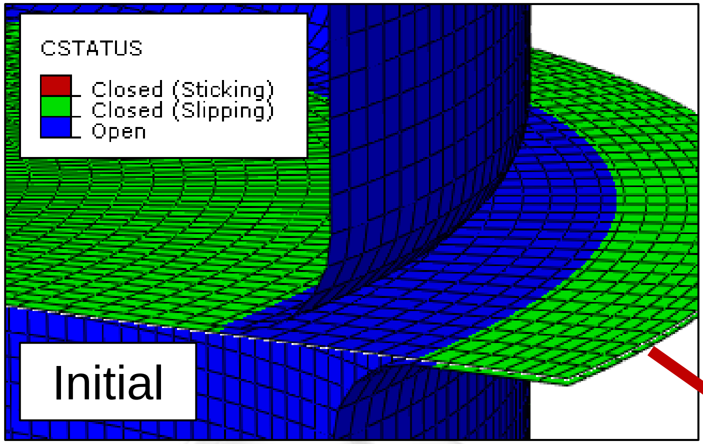




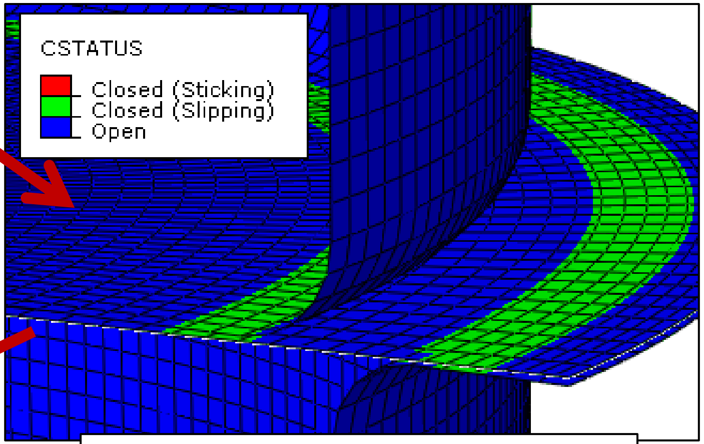
Output of Contact Results

- **CSTATUS in shell forming example discussed earlier**
  - No friction defined in this model

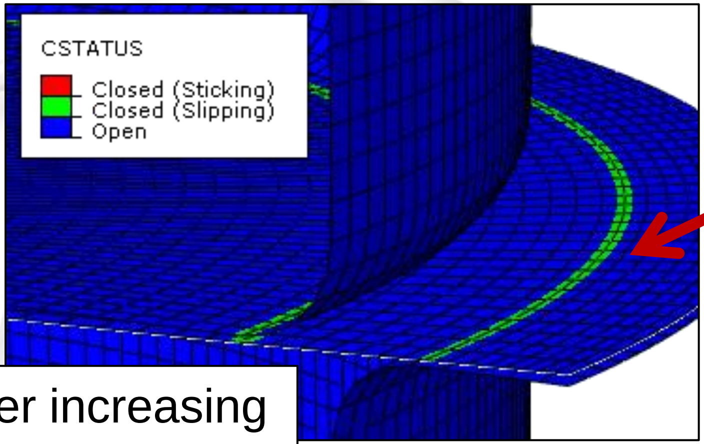




Initial



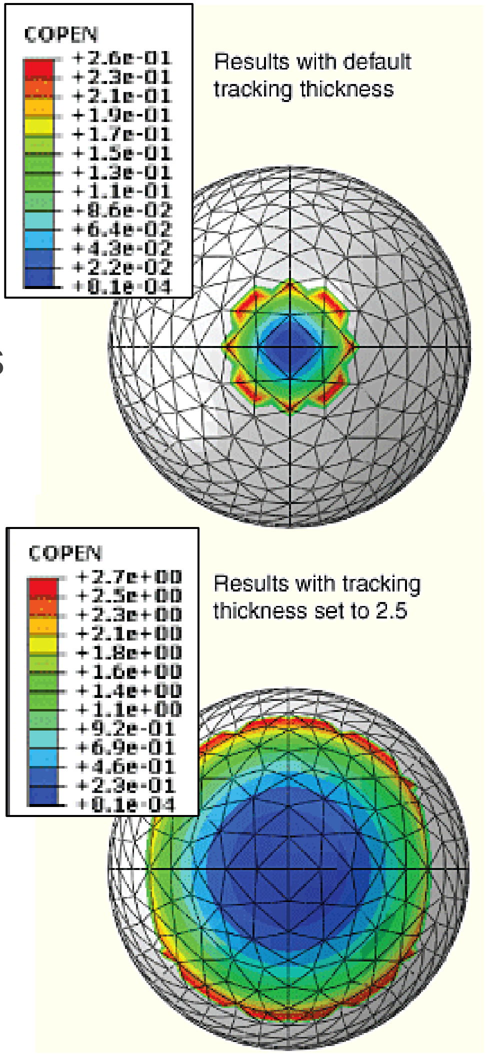
After first increment with small penalty stiffness



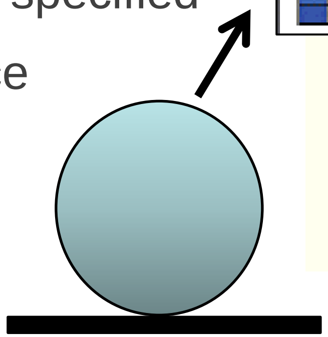
After increasing penalty stiffness

User-Defined Range for Ensuring Contact Opening Output

- **Abaqus often does not provide COPEN values for regions with a significant gap**
  - Especially in recent versions
    - Motivation: Minimize contact search time
  - Gap distance output is important in some cases
    - Previous workaround: Define an insignificant amount of contact damping over a gap range of interest
- **Abaqus 6.10: ∗Surface Interaction, Tracking Thickness=*value***
  - COPEN output at least up to value specified
  - Warning: Can degrade performance

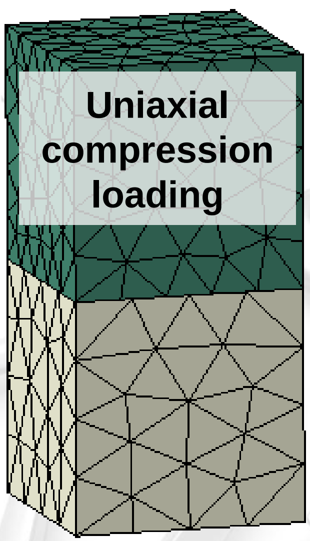


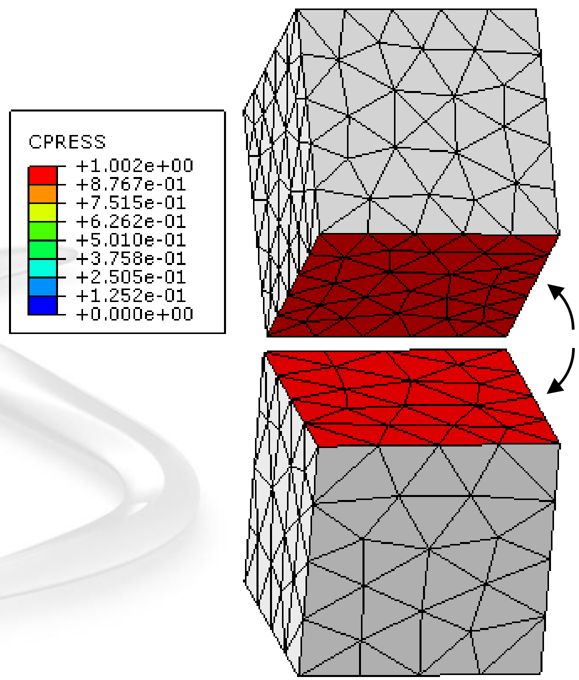
Sphere-on-plate example



Output of Contact Results

- **Most contact output is available on both slave and master surfaces**
  - Cannot view contact output on surfaces based on rigid elements types (when used as part of a contact pair) or analytical rigid surfaces



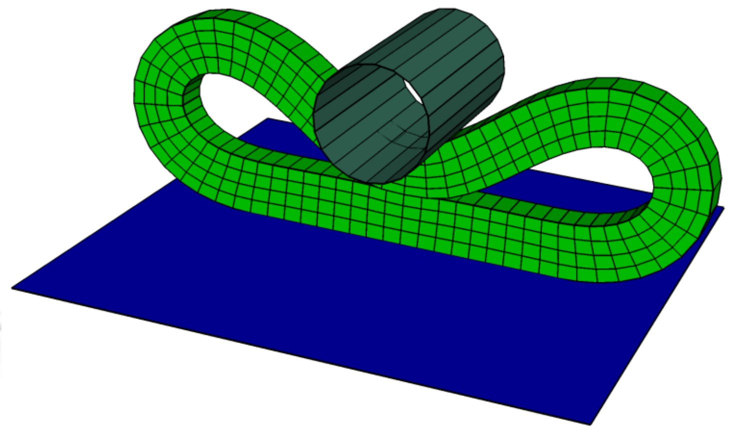


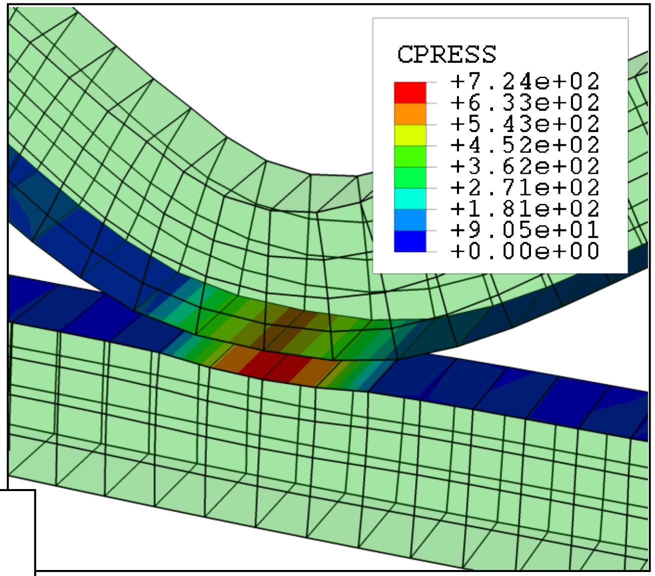
Results obtained with surface-to-surface contact formulation

Rotated to see contact pressure on both sides of contact interface

Output of Contact Results

- **Self-contact results**
  - Values of CPRESS, CSHEAR, CNORMF, CSHEARF in output database file represent net quantities
    - Contributions while a node acts as slave in some constraints and master in other constraints for a given self-contact definition





Displacement magnification factor of 0.96 to facilitate visualization

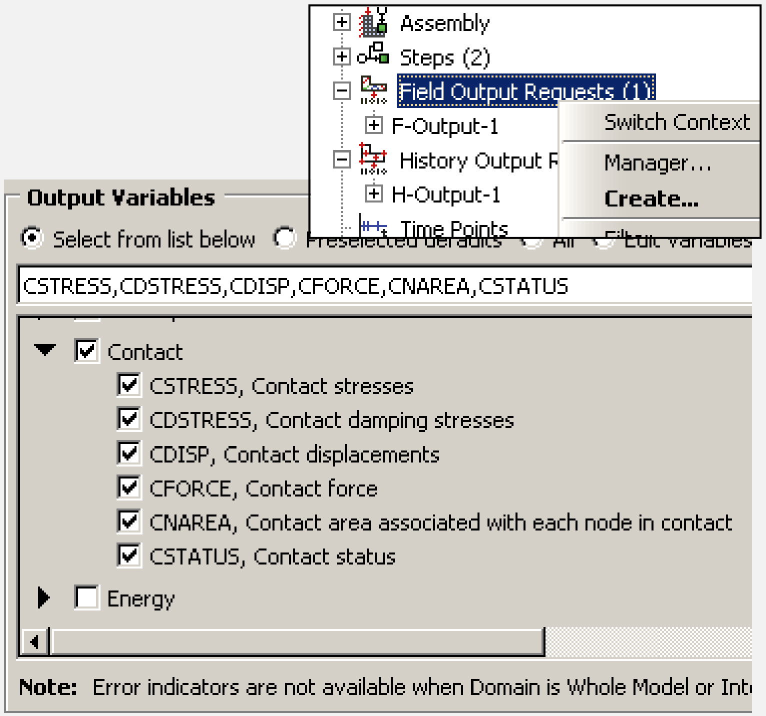
Output of Contact Results

- **Contact area**
  - Small sliding:
    - Contact area always based on **reference configuration** (regardless of whether or not geometrically nonlinear effects are considered)
  - Finite sliding:
    - Contact area always based on the **current configuration** (regardless of whether or not geometrically nonlinear effects are considered)
- **Units of contact stresses**
  - For most elements-based surfaces: Force per actual unit area (stress)
  - Beams (2-D or 3-D): Force per unit length
  - Node-based surfaces: Force per user-defined nodal area (default nodal area = 1)

Output of Contact Results

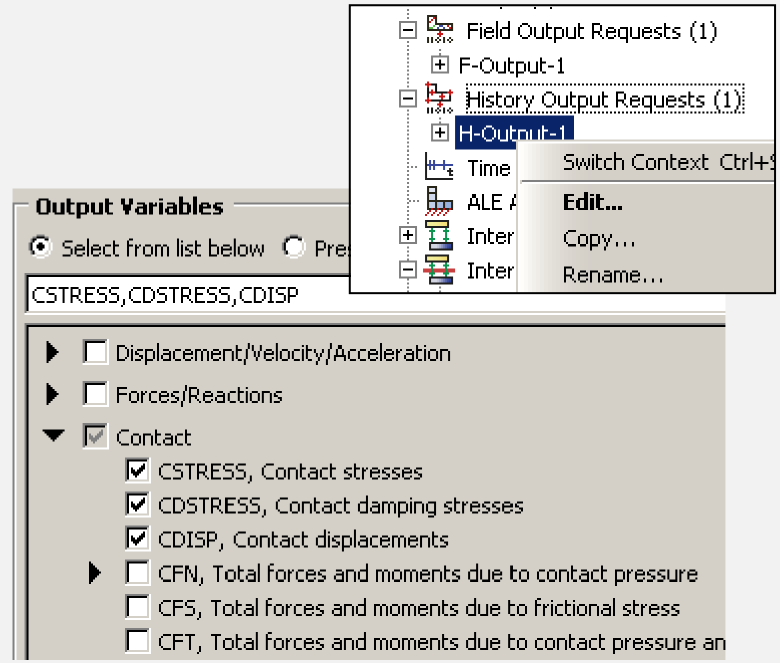
- **Nodal contact output requests**

Field




```
*OUTPUT, FIELD
*CONTACT OUTPUT
```

History




```
*OUTPUT, HISTORY
*CONTACT OUTPUT
```

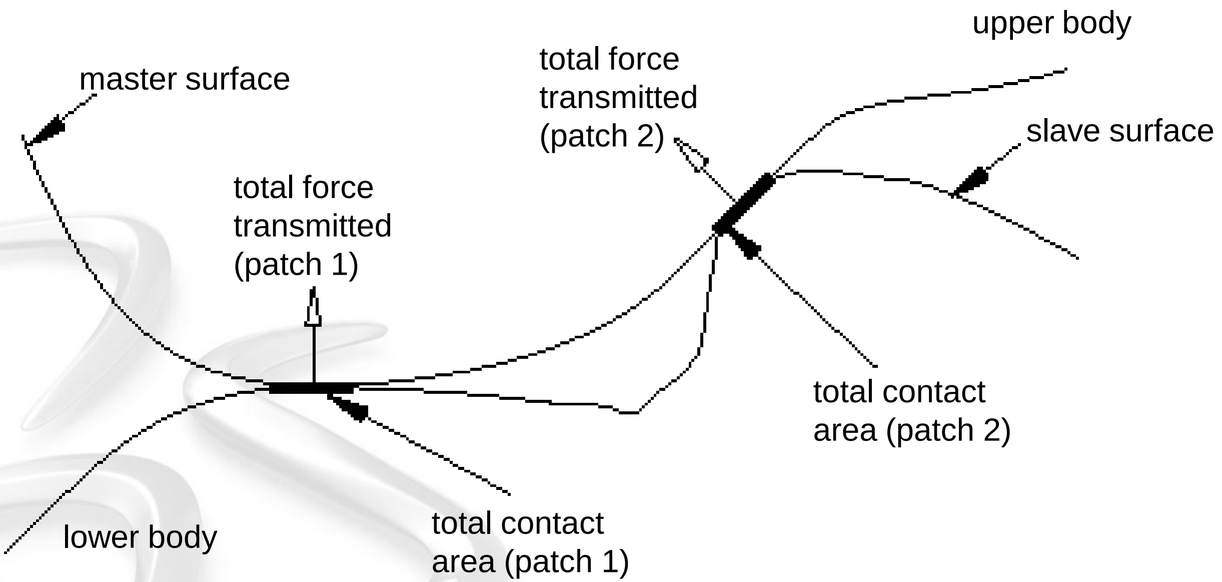
Output of Contact Results

- **Whole surface output to the ODB file**
  - History output

Output Variable	Description
CAREA	**Total area in contact**
CFN **CFS**	**Total force vector due to contact pressure and frictional shear stress, respectively**
CMN **CMS**	**Total moment vector about the origin due to contact pressure and frictional stress, respectively**
CFT	**Vector sum of CFN and CFS**
CMT	**Vector sum of CMN and CMS**
XN	**Coordinates of a point about which the total moment due to the contact pressure is equal to zero**
XS	**Coordinates of a point about which the total moment due to the frictional stress is equal to zero**
XT	**Coordinates of a point about which the total moment due to the contact pressure and frictional stress is equal to zero**

Output of Contact Results

- **Whole surface output to the ODB file**
  - Example: Two surfaces contacting at two locations



total force = total force patch 1 + total force patch 2
total area = total area patch 1 + total area patch 2
total moment = total moment patch 1 + total moment patch 2

Output of Contact Results

- **Other types of output**
  - Two options are available for generating printed output that is relevant to contact analyses

    *PREPRINT, CONTACT=YES

    - Controls output to the printed output (`.dat`) file during the preprocessing phase
    - Gives details of internally generated contact elements

    *PRINT, CONTACT=YES

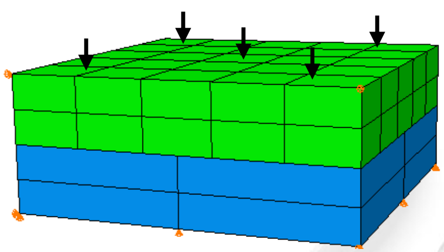
    - Controls output to the message (`.msg`) file during the analysis phase
    - Gives details of the iteration process

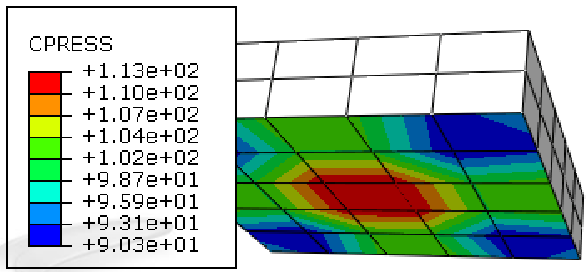
Contact Pressure Accuracy

SIMULIA

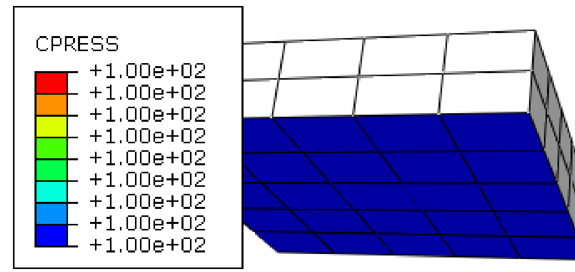
Contact Pressure Accuracy

- **Recall discussion earlier in the seminar related to this topic**
  - “Consistent force” distribution with surface-to-surface formulation
    - Results in more accurate contact pressures than with node-to-surface formulation



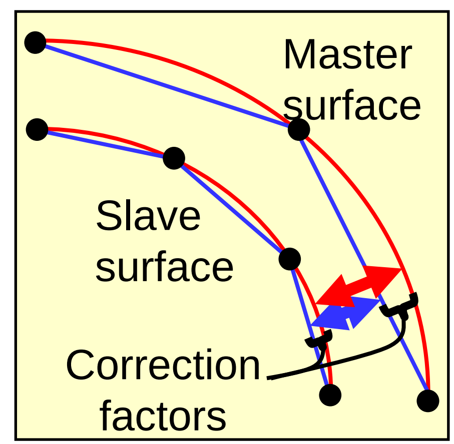


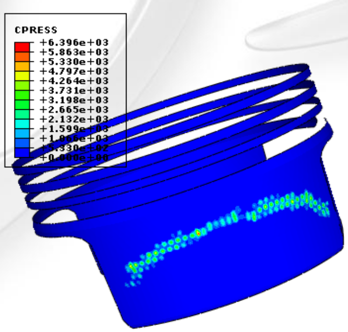
Node-to-Surface Formulation

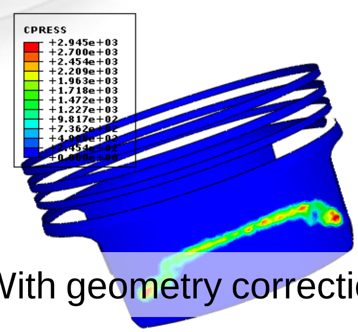


Surface-to-Surface Formulation

- Geometry corrections for curved surfaces
  - Better “input” to the contact formulation improves accuracy



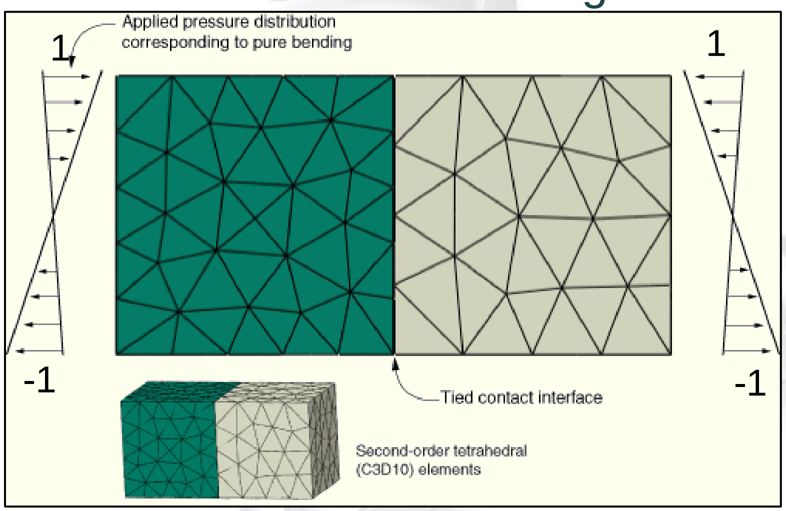


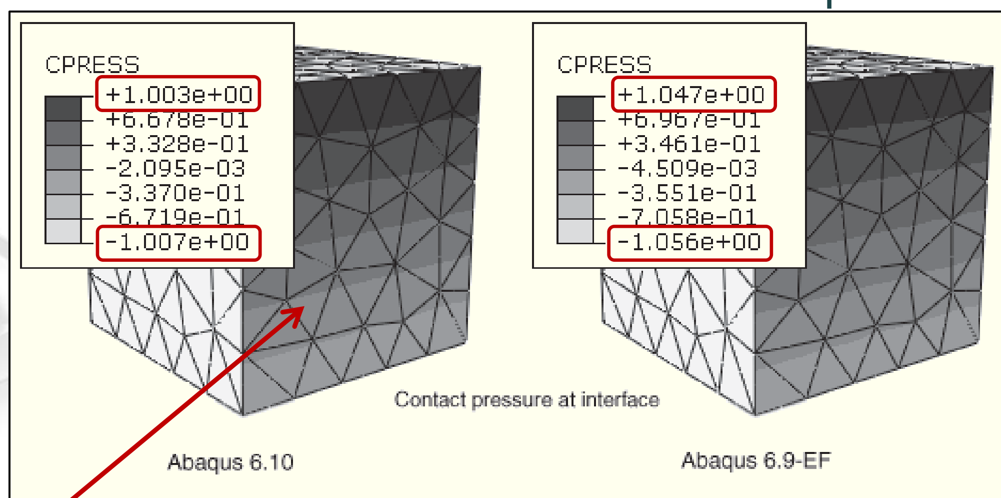


With geometry corrections

Contact Pressure Accuracy

- **Resolution of linearly varying contact pressure**
  - Enhanced in Abaqus 6.10 for models with the surface-to-surface formulation and second-order elements
  - Demonstrated in a pure bending example below
    - Tied contact interface; C3D10 elements
    - Order of magnitude reduction in CPRESS noise in this example

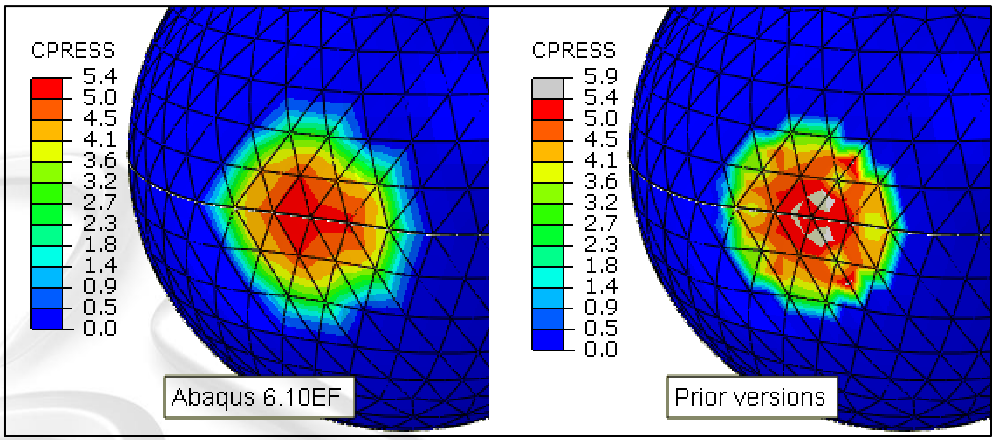


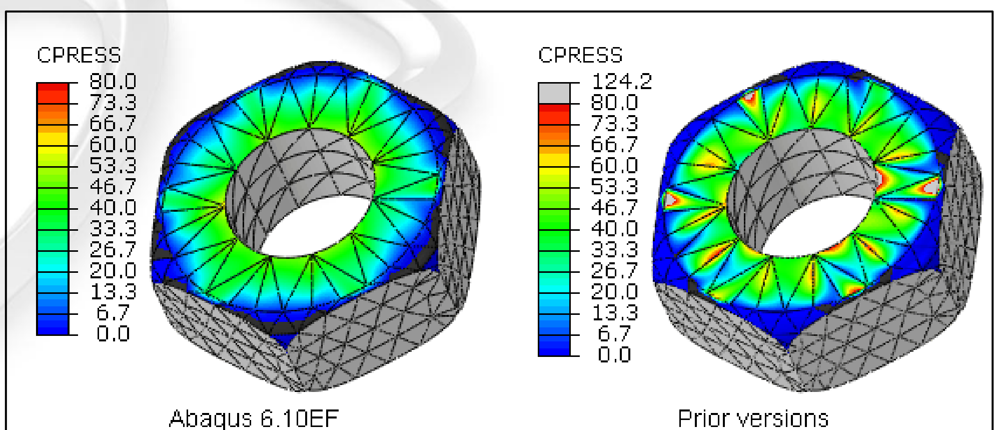


CPRESS noise ≈ 2% of variation in CPRESS over individual facets (for linearly varying pressure with similar C3D10 meshes)

Contact Pressure Accuracy

- **New filtering in Abaqus 6.10EF**
  - Applies to surface-to-surface and node-to-surface formulations
  - Generally, nice effect on solutions

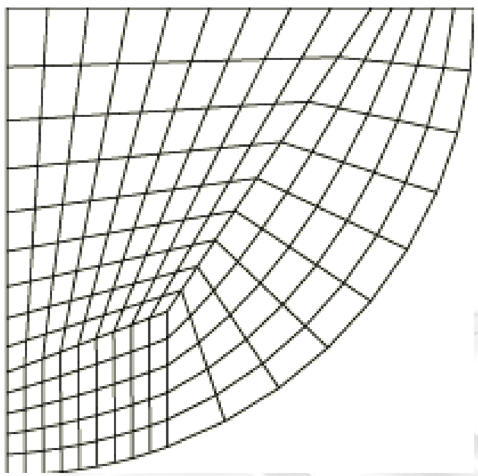


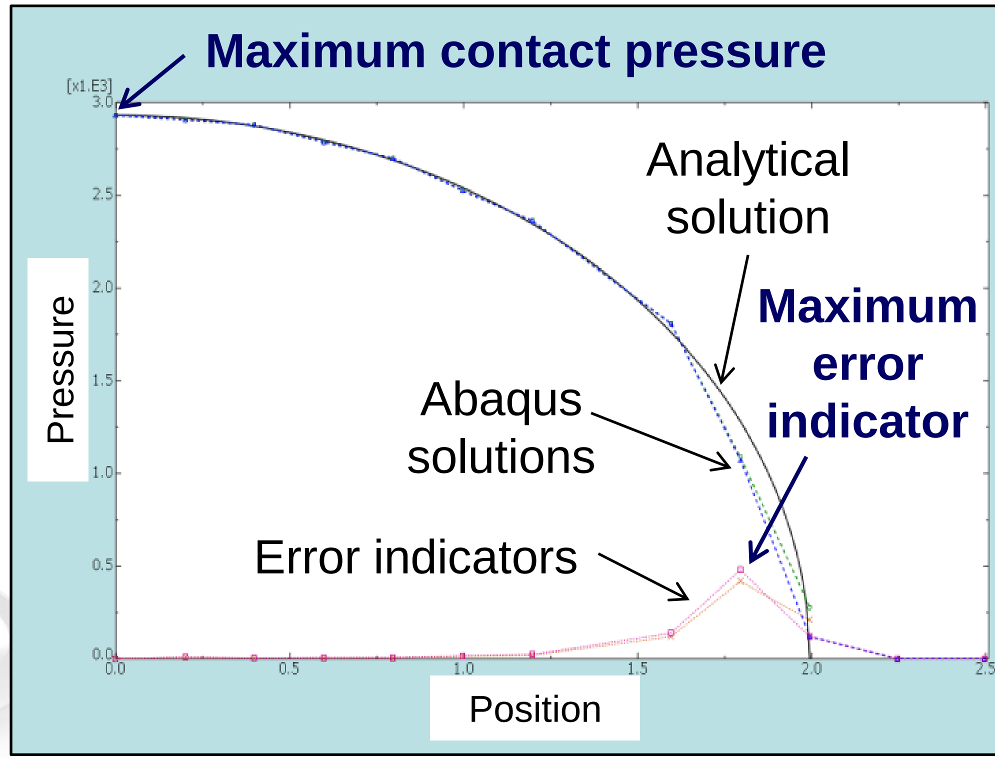


SIMULIA

Contact Pressure Accuracy

- **Contact stress error indicators added for Abaqus/Standard 6.11**
  - Hertz contact example

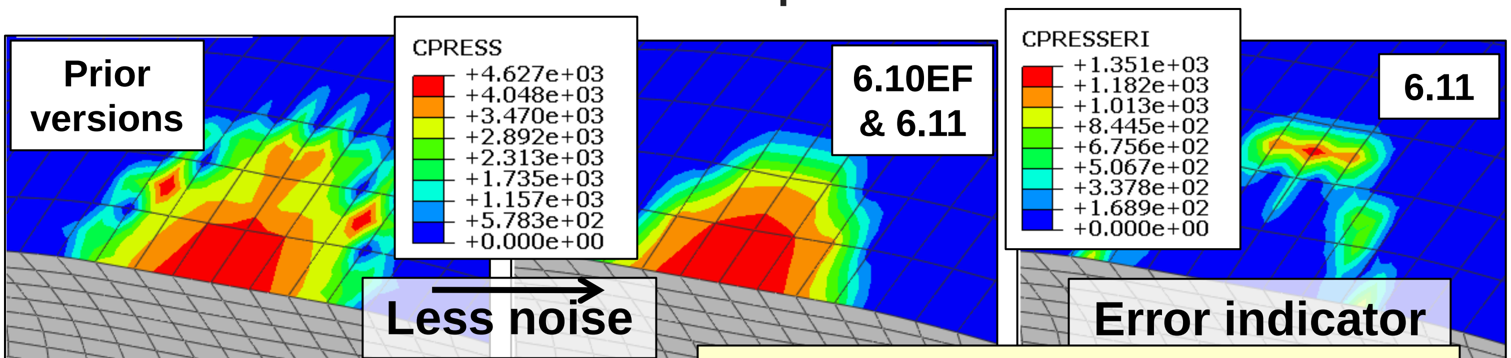




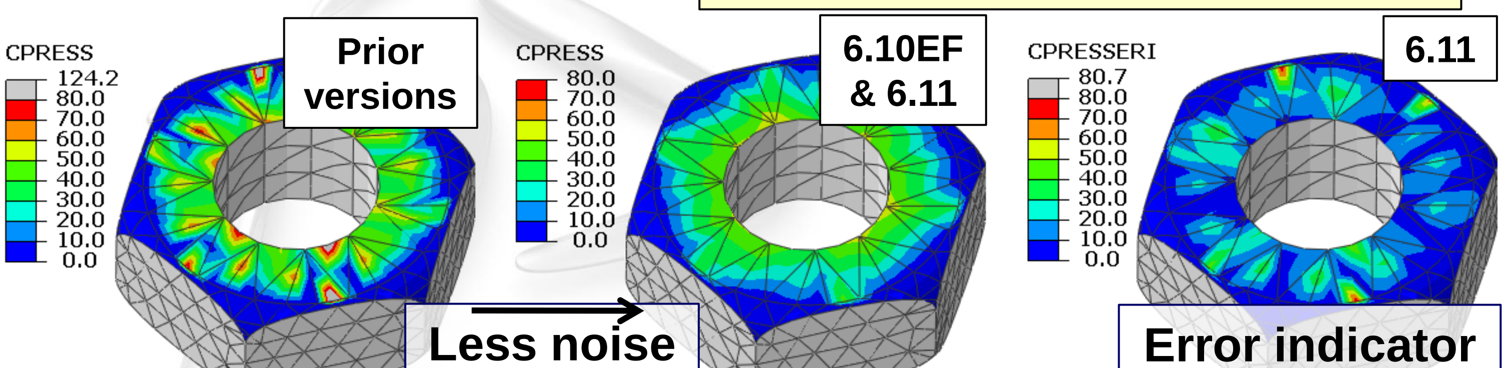
- **Points to remember for error indicators:**
  - Tend to be large where local variation of base variable is more complex than what can be captured by the mesh
  - *Not normalized*; same units as base variable
  - *Not conservative* or precise estimates of error

Contact stress error indicators

- Consider error indicators for examples shown earlier:



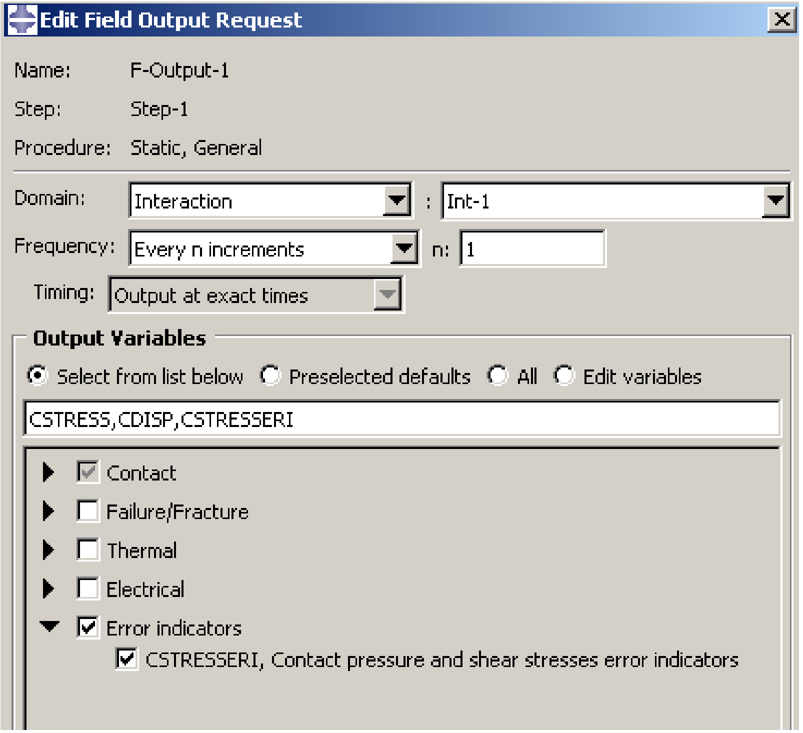
Interpretation:
- Accurate prediction of maximum CPRESS
- Some uncertainty where gradient is large but pressure is low



Interpretation:
- Need finer mesh to predict maximum contact pressure and characterize local “hot spots”

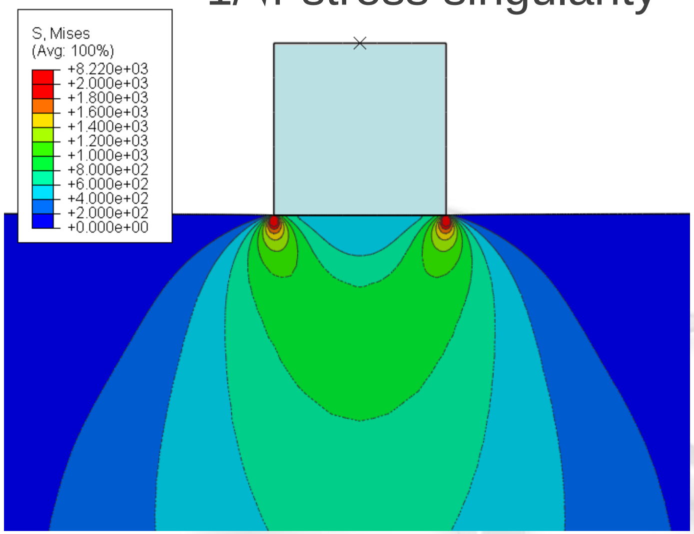
Contact Pressure Accuracy

- **Contact stress error indicators**
  - Nodal variables
    - Similar to CSTRESS
  - Request CSTRESSERI under *Contact Output
    - Output of CPRESSERI, CSHEAR1ERI, CSHEAR2ERI
  - Supported by /CAE
  - Field variable output to .odb
  - Not part of Variable=Preselect
  - *Cannot be used to drive adaptive remeshing*

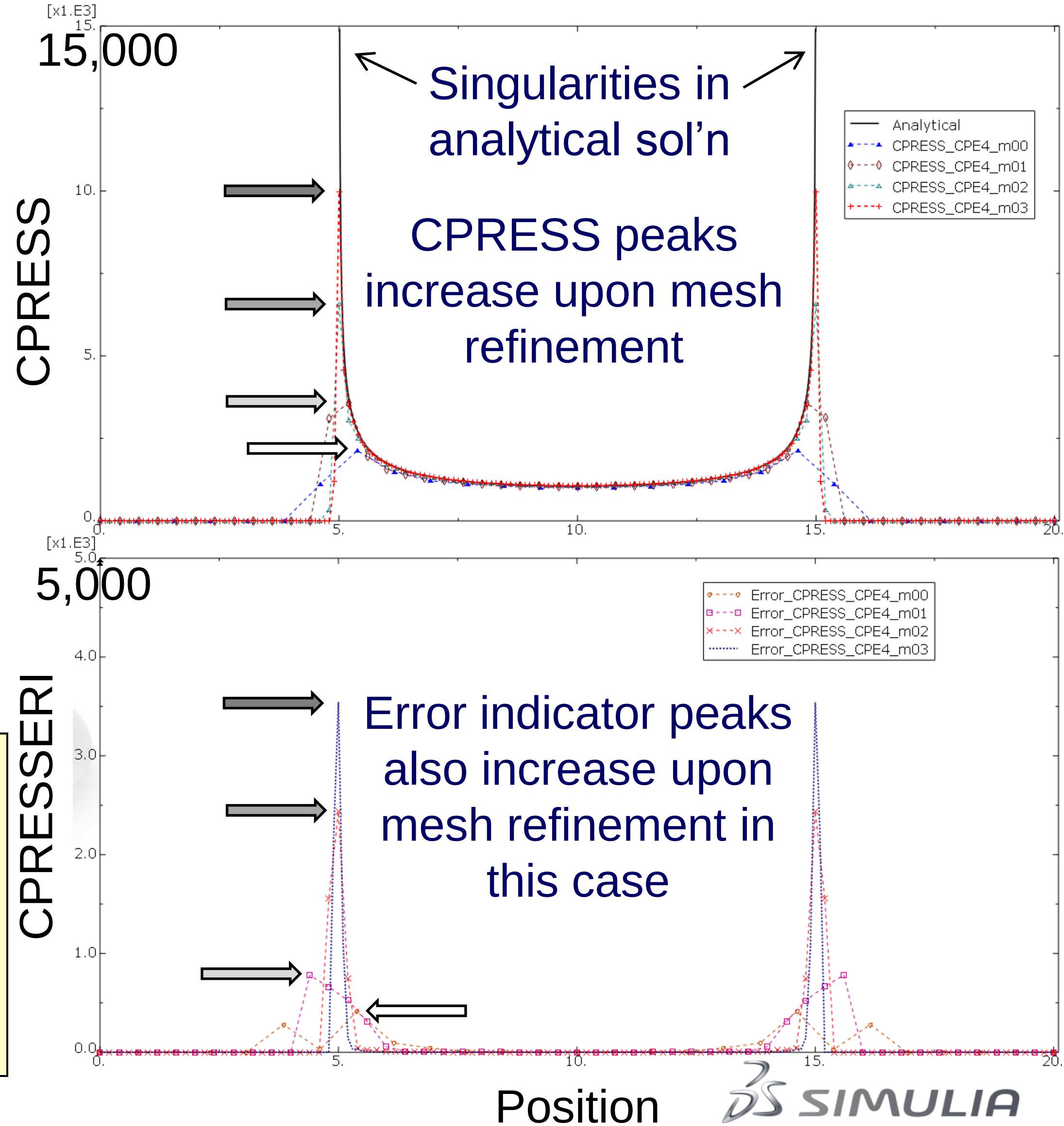


Contact Pressure Accuracy

- **Rigid punch example**
  - Analytical solution has $1/\sqrt{r}$ stress singularity

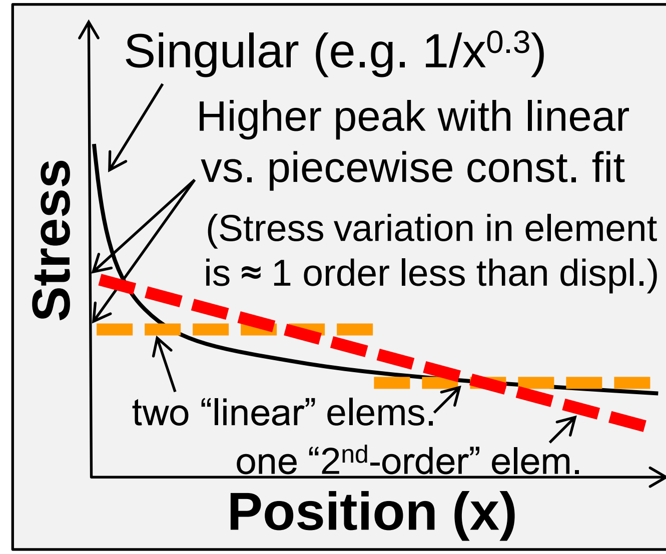


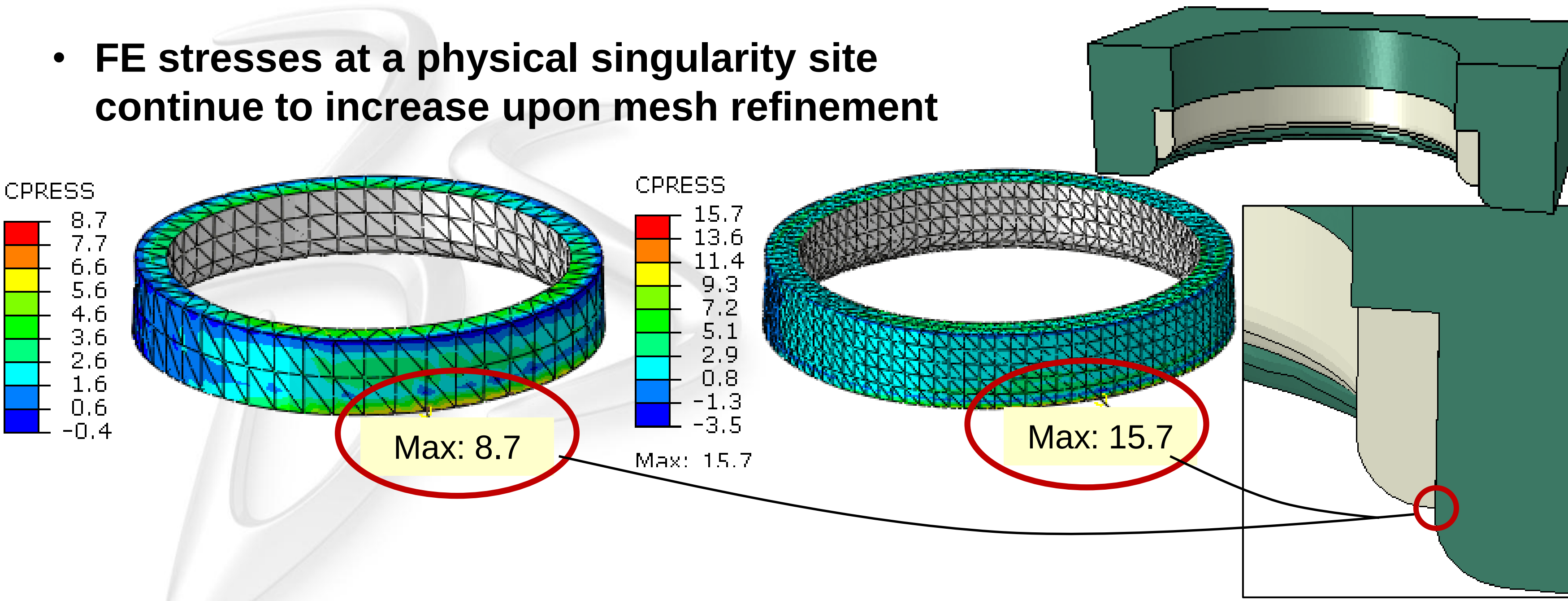
- Singularity order would be $1/r^{0.23}$ for deformable bodies of like material (frictionless)
- Corner contact singularities are *common* and their presence is often *not intuitive*



Contact Pressure Accuracy

- **2^{nd}-order elements (with S-to-S contact) tend to be more sensitive to localized effects**
  - Increases in local stress peaks with this modeling approach are often misinterpreted as numerical noise (unaware of possibility of physical singularity)
- **FE stresses at a physical singularity site continue to increase upon mesh refinement**

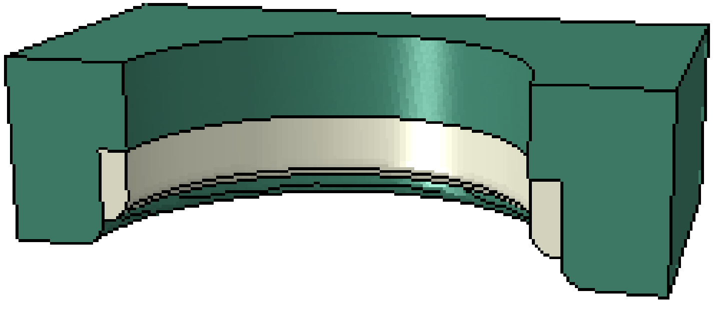




Contact Pressure Accuracy

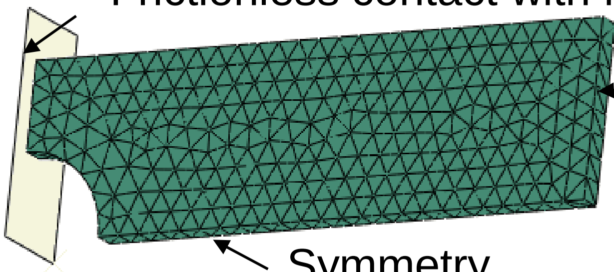
- **In actual mechanical systems:**
  - Slight rounding of corners and localized yielding (not included in model description) may reduce significance of these effects
    - But extra wear, etc. at these locations is likely
- **Consider fillets or local yielding with a *sub-modeling* approach**
  - May be impractical to model these details in a full assembly model
    - Extra degrees of freedom & iterations
  - More effective to use results from a **global model** as boundary conditions for a **more detailed local model**

Relatively small region of a power train analysis:

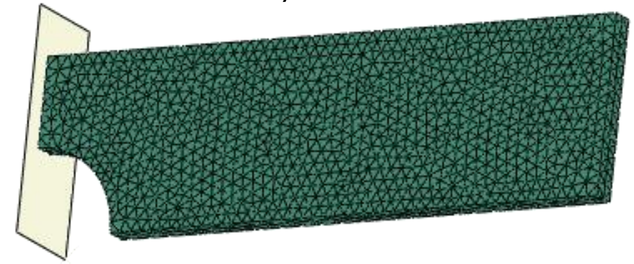


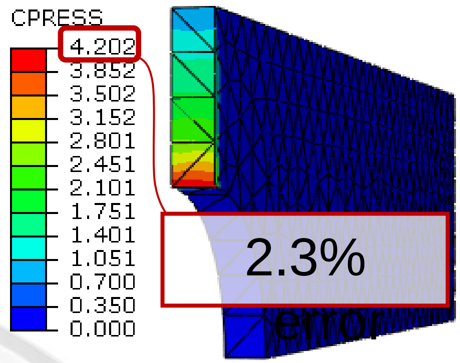
Contact Pressure Accuracy

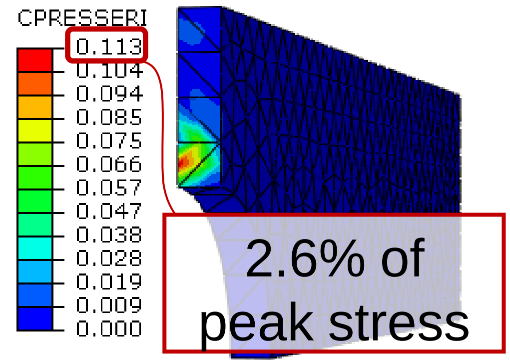
- **Stress concentration example**



Second, finer mesh

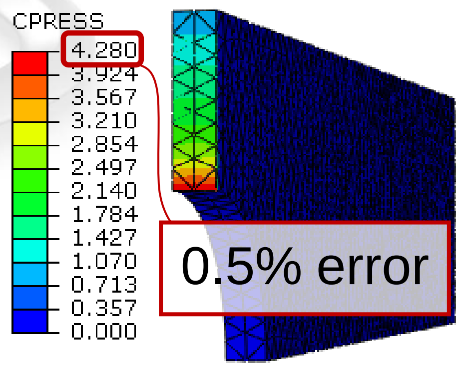


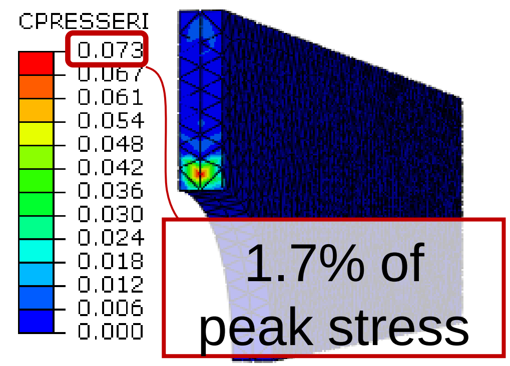




Reference solution:

- Peak stress=4.3





Contact Stress Error Indicators

- **“Art” of interpreting error indicators**
  - Documentation excerpt:

    Warning: *Error indicator output variables are approximate and do not represent an accurate or conservative estimate of your solution error. The quality of an error indicator can be particularly poor if your mesh is coarse. The error indicator quality improves as you refine the mesh; however, you should never interpret these variables as indicating what the value of a solution variable would be upon further refinement of the mesh.*

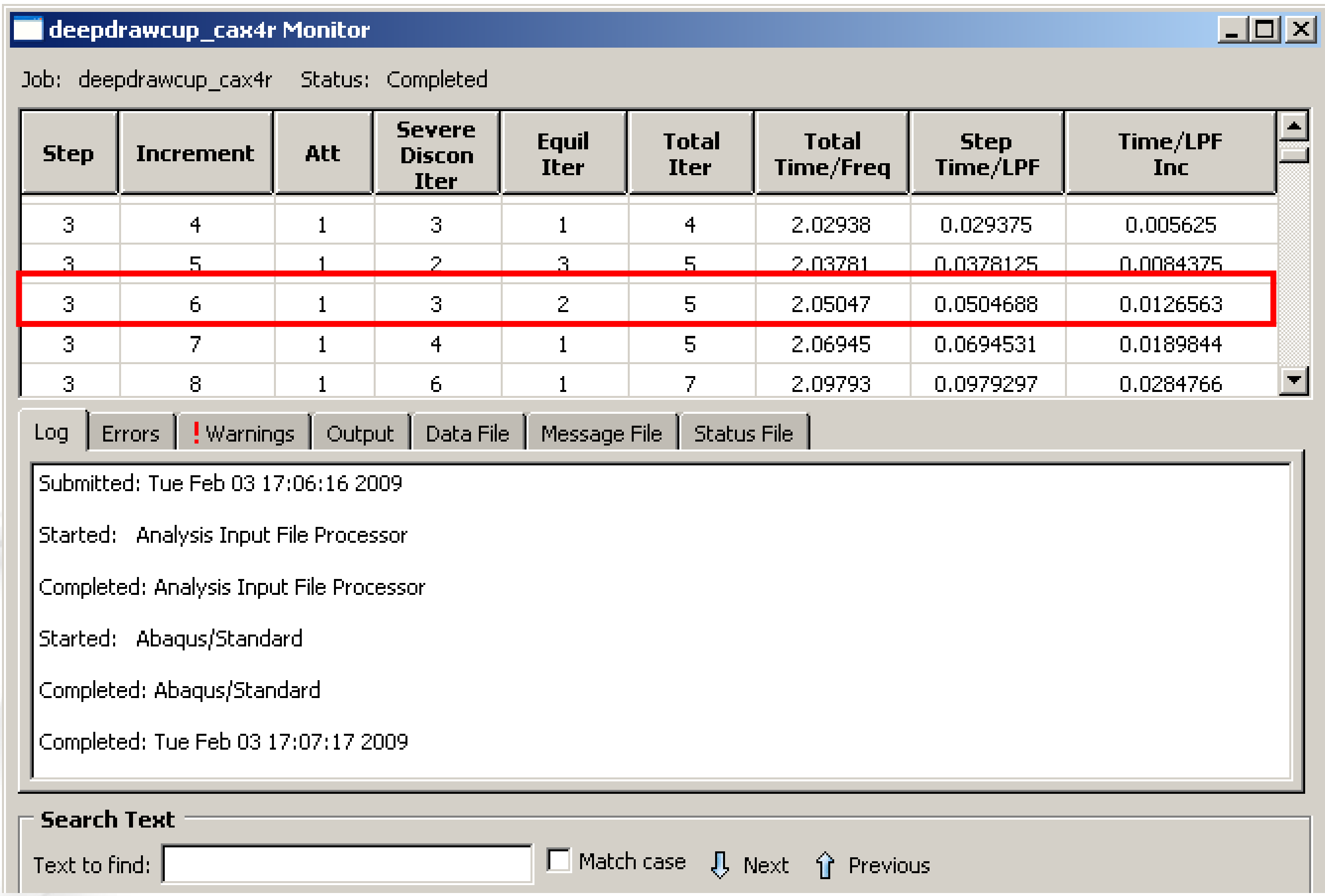
Error indicators do not replace need for:
- Mesh refinement studies
- Other ways that analysts gain confidence in modeling practices

Contact Diagnostics (Visual)

SIMULIA

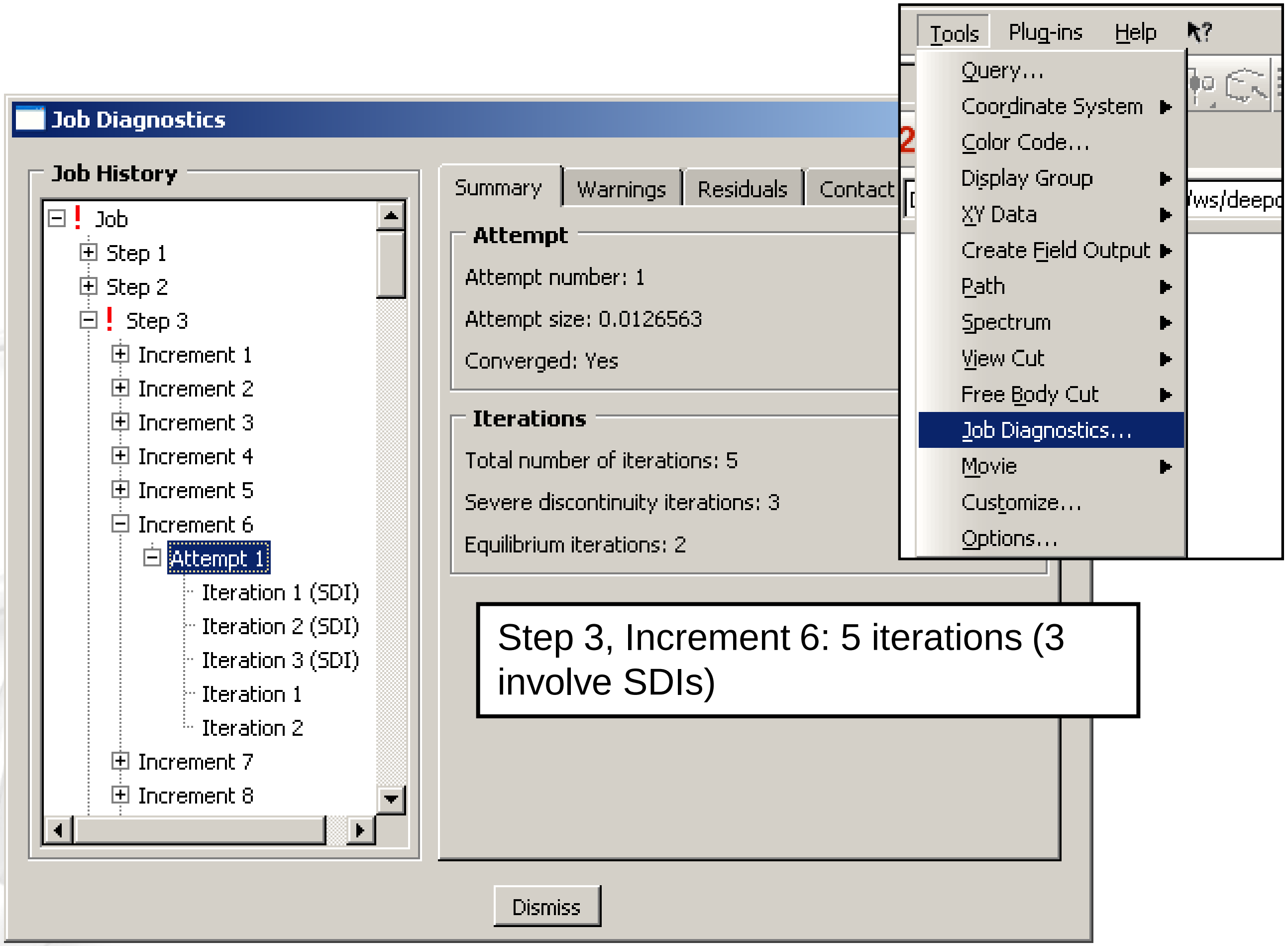
Contact Diagnostics (Visual)

- **Contact diagnostics example using Abaqus/CAE**
  - Reference: Example Problem 1.3.4, *Deep drawing of a cylindrical cup*

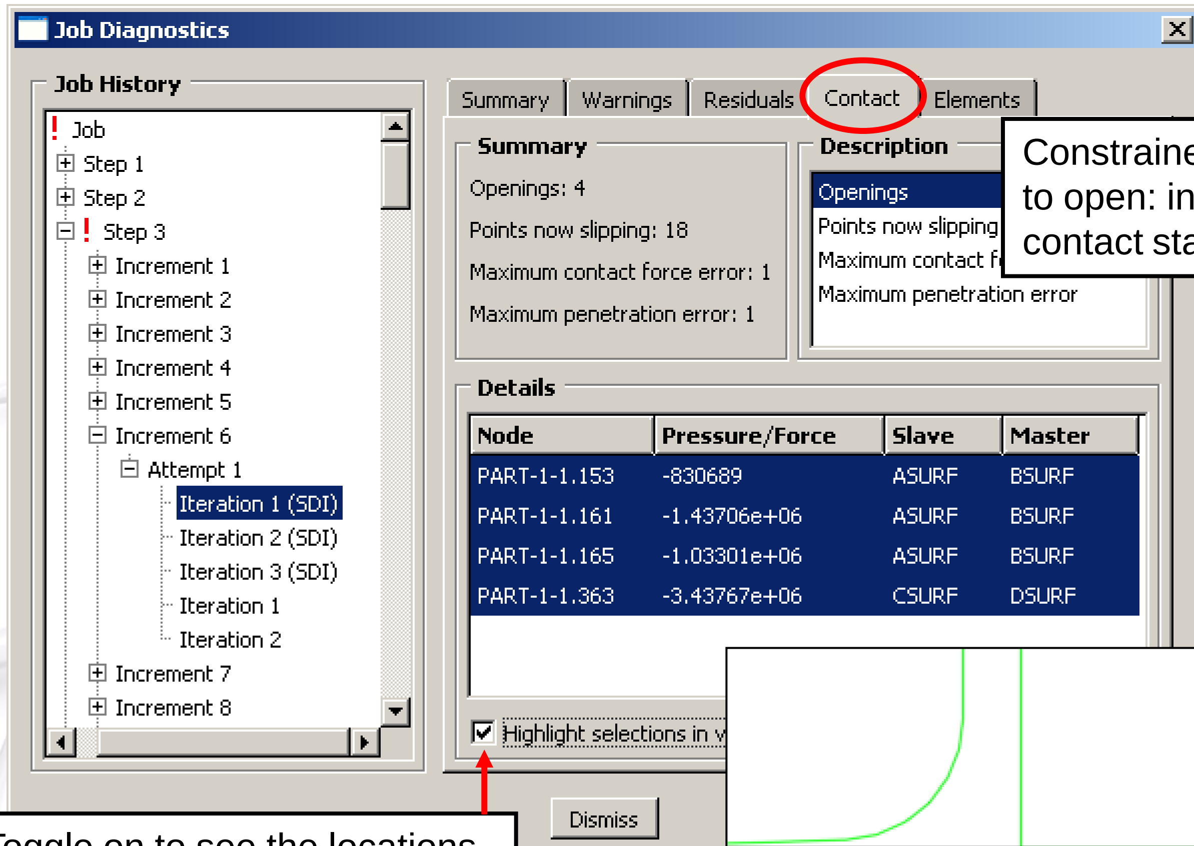


Contact Diagnostics (Visual)

- **Visual diagnostics available in the Visualization module of Abaqus/CAE**

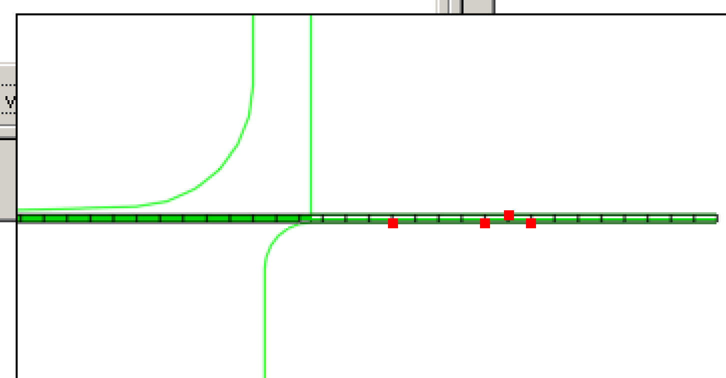


Contact Diagnostics (Visual)



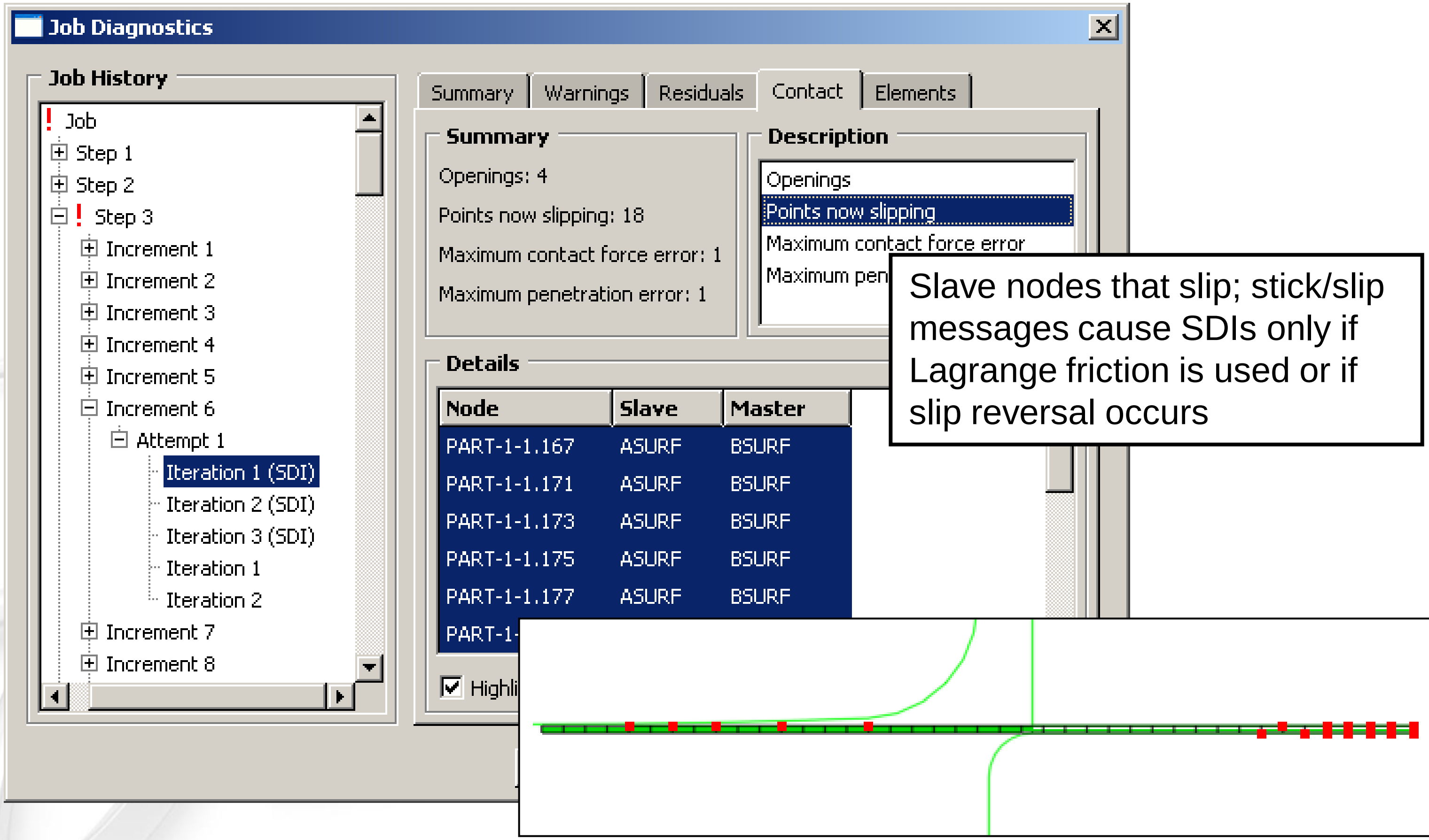
Constrained nodes want to open: incompatible contact state

Toggle on to see the locations in the model where the contact state is changing



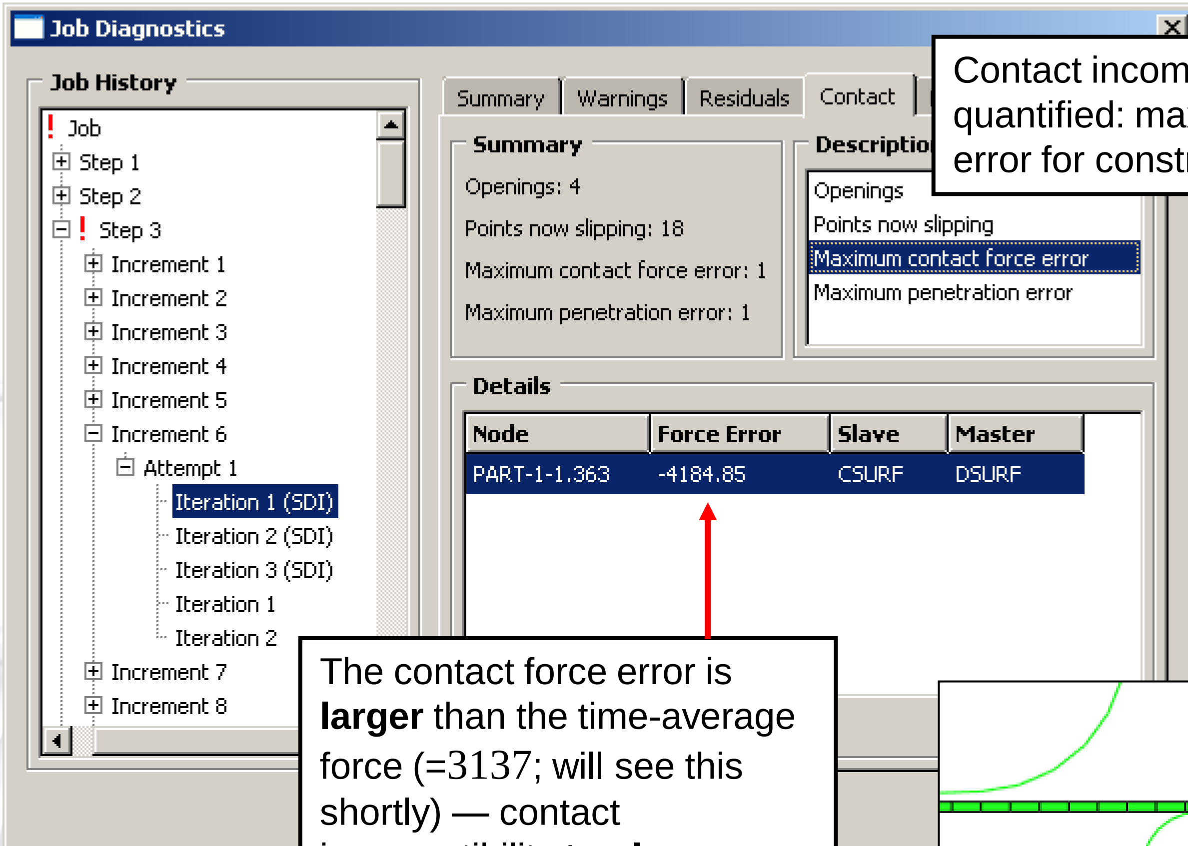
SIMULIA

Contact Diagnostics (Visual)



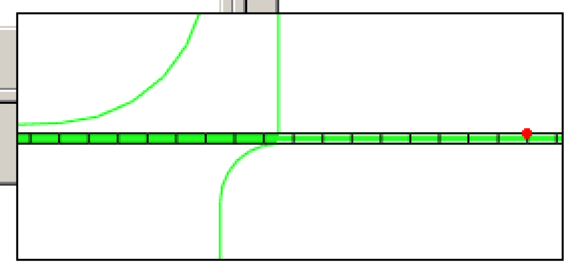
Slave nodes that slip; stick/slip messages cause SDIs only if Lagrange friction is used or if slip reversal occurs

Contact Diagnostics (Visual)

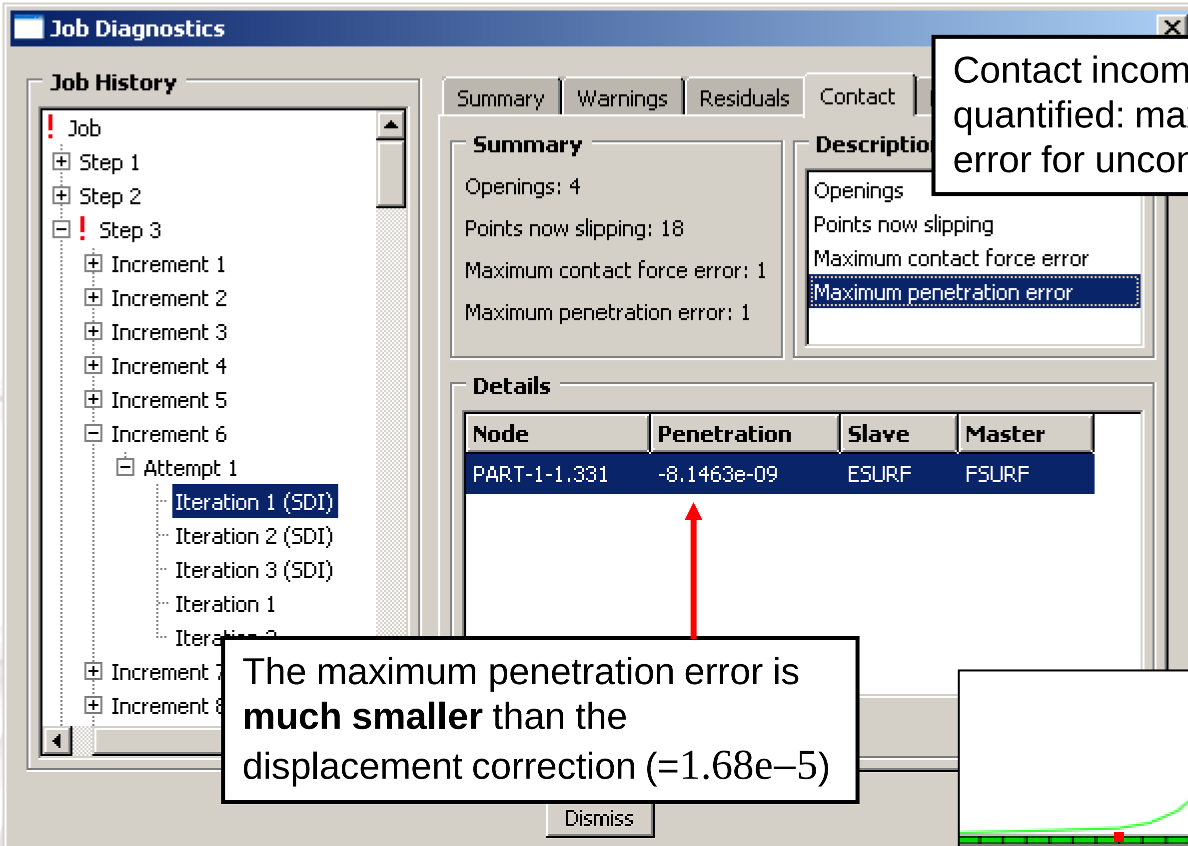


Contact incompatibilities are quantified: max **force** error for constrained nodes

The contact force error is **larger** than the time-average force (=3137; will see this shortly) — contact incompatibility **too large**

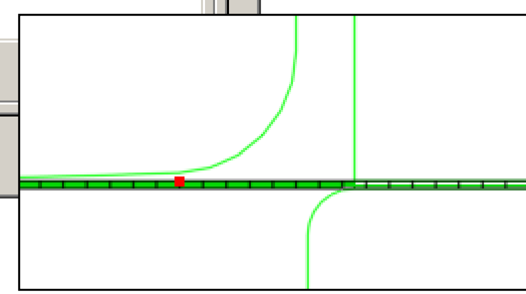


Contact Diagnostics (Visual)

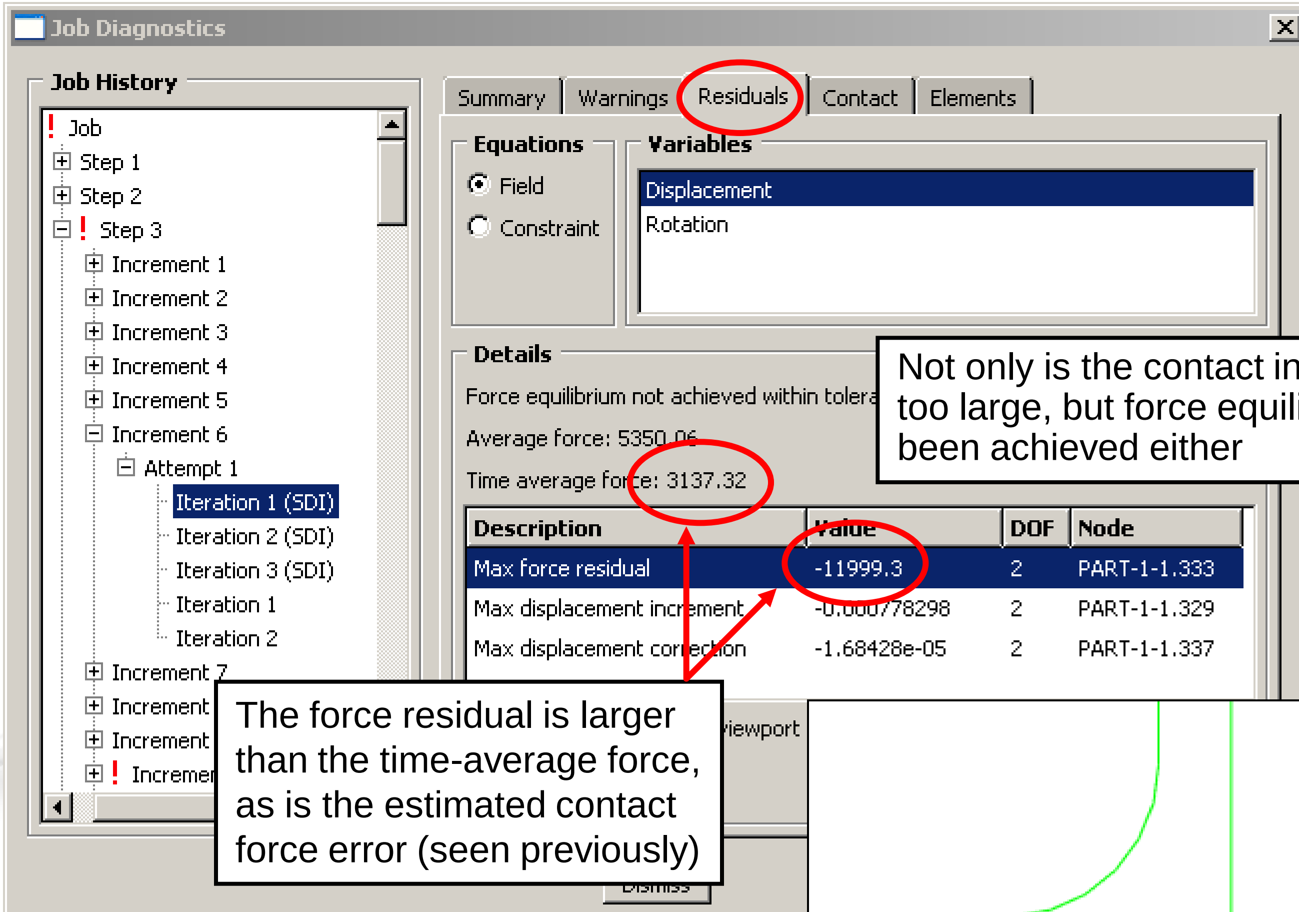


Contact incompatibilities are quantified: max **penetration** error for unconstrained nodes

The maximum penetration error is **much smaller** than the displacement correction (=1.68e–5)

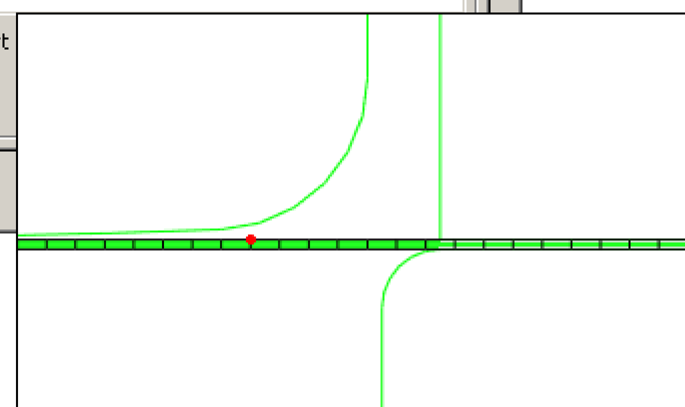


Contact Diagnostics (Visual)



Not only is the contact incompatibility too large, but force equilibrium has not been achieved either

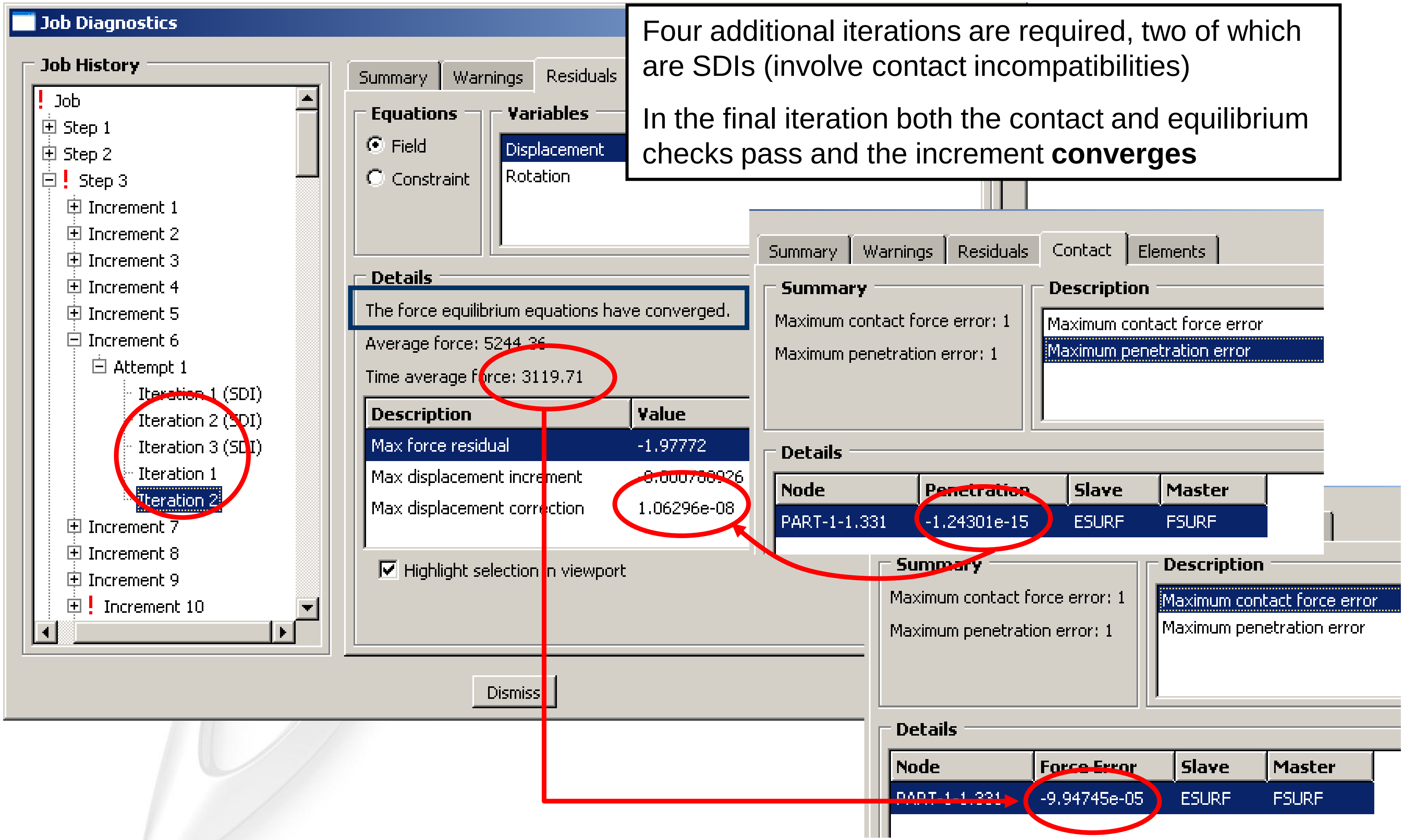
The force residual is larger than the time-average force, as is the estimated contact force error (seen previously)



Contact Diagnostics (Visual)

Four additional iterations are required, two of which are SDIs (involve contact incompatibilities)

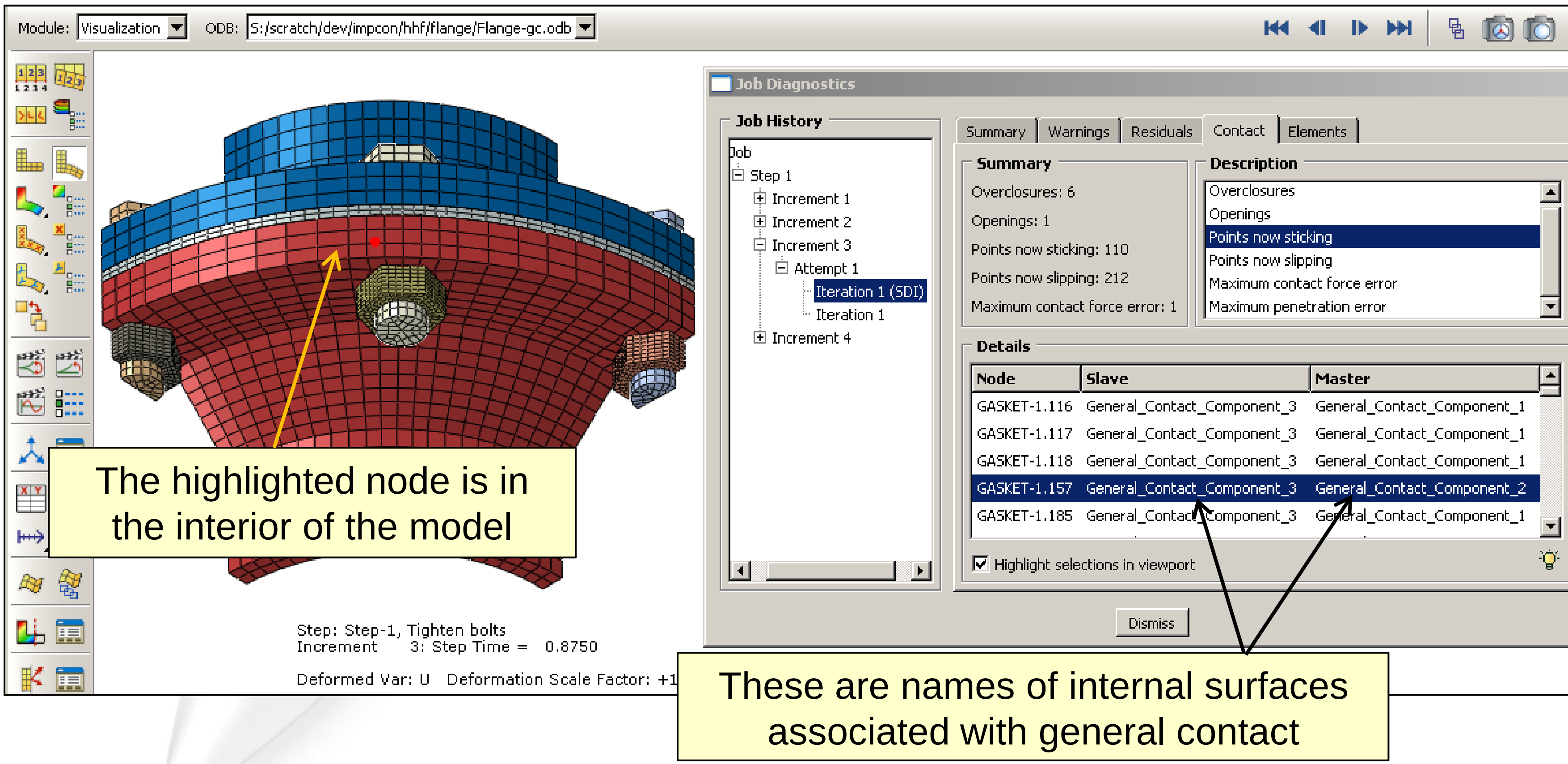
In the final iteration both the contact and equilibrium checks pass and the increment **converges**



Introduced in Abaqus 6.9-EF

Contact Diagnostics (Visual)

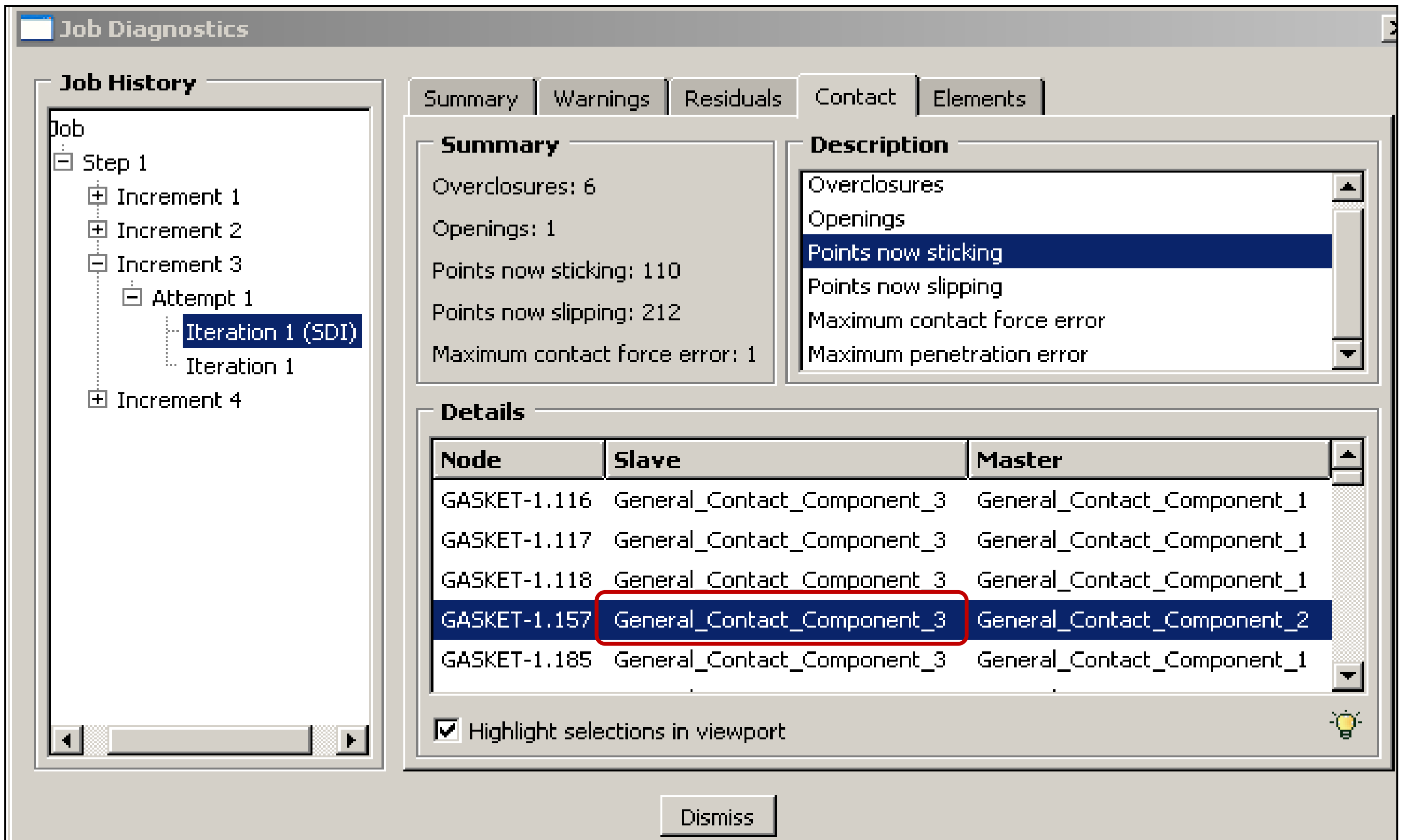
- **Internal "component surface" names appear in diagnostic messages associated with general contact**
  - Previously these messages referred to the overall general contact surface



SIMULIA

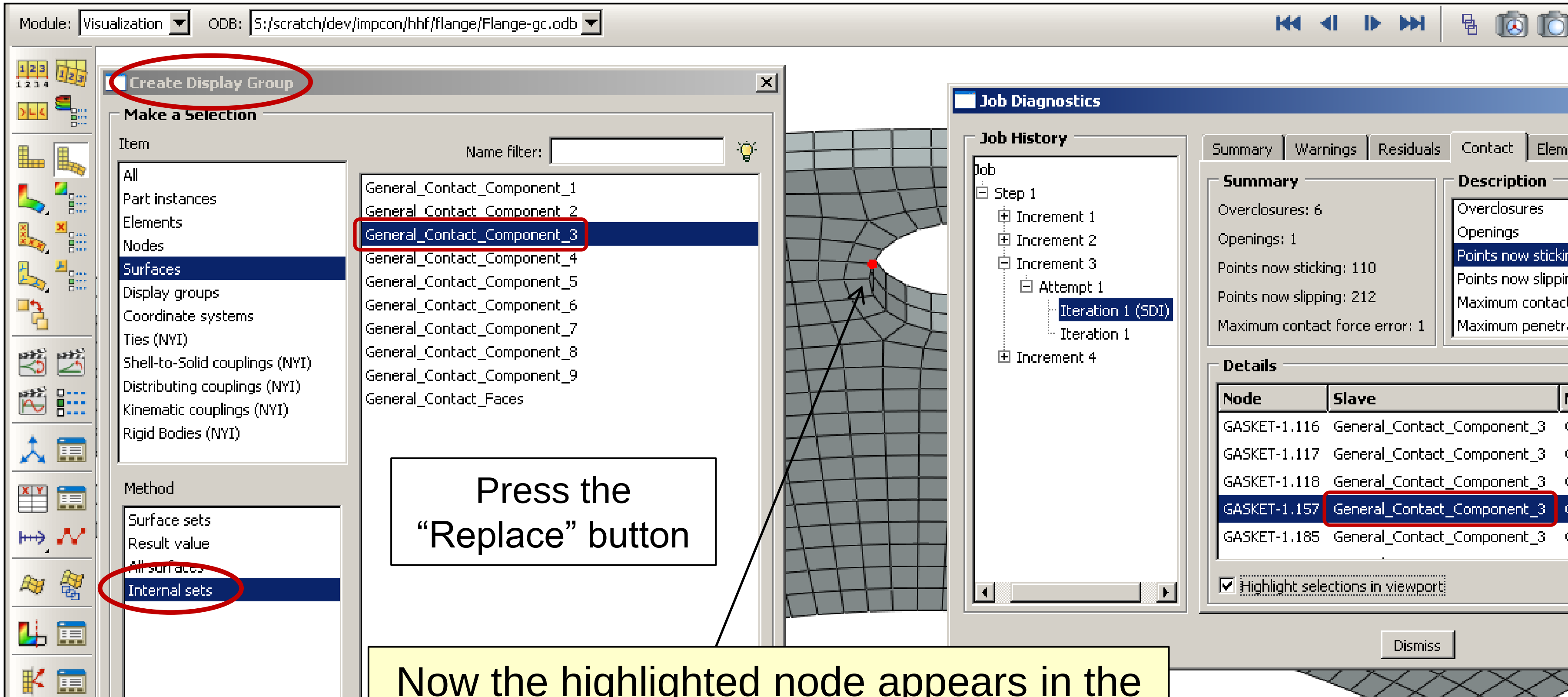
Contact Diagnostics (Visual)

- **To facilitate visualization**
  - Limit what appears in Abaqus/Viewer to the slave and/or master surface mentioned in a diagnostic message



Contact Diagnostics (Visual)

- **Use the "Create Display Group" dialog box**
  - Set "Method" to "Internal sets" in this case



Now the highlighted node appears in the context of the slave surface configuration

Contact Diagnostics (Text)

SIMULIA

Contact Diagnostics (Text)

- **Contact diagnostics example using the message (`.msg`) file**
  - Reference: Example Problem 1.3.4, "Deep drawing of a cylindrical cup"
  - Status (`.sta`) file:

```
SUMMARY OF JOB INFORMATION:
 MONITOR NODE:      200  DOF:   2
 STEP    INC ATT SEVERE EQUIL TOTAL   TOTAL       STEP        INC OF        DOF     IF
                 DISCON ITERS ITERS   TIME/     TIME/LPF     TIME/LPF     MONITOR RIKS
                 ITERS                FREQ
   1      1   1      1     1     2  1.00        1.00        1.000         0.000
   2      1   1      0     1     1  2.00        1.00        1.000         0.000
   3      1   1     10     0    10  2.01        0.0100      0.01000      -0.000600
   3      2   1      7     1     8  2.02        0.0200      0.01000      -0.00120
   3      3   1U     9     0     9  2.02        0.0200      0.01500      -0.00120
   3      3   2      5     0     5  2.02        0.0238      0.003750     -0.00142
   3      4   1      3     1     4  2.03        0.0294      0.005625     -0.00176
   3      5   1      2     3     5  2.04        0.0378      0.008438     -0.00227
   3      6   1      3     2     5  2.05        0.0505      0.01266      -0.00303
   3      7   1      4     1     5  2.07        0.0695      0.01898      -0.00417
   3      8   1      6     1     7  2.10        0.0979      0.02848      -0.00588
   3      9   1      3     4     7  2.14        0.141       0.04271      -0.00844
   3     10   1U     4     0     4  2.14        0.141       0.06407      -0.00844
   3     10   2      7     1     8  2.16        0.157       0.01602      -0.00940
   3     11   1      3     2     5  2.18        0.181       0.02403      -0.0108
   .
   .
   .
```

Contact Diagnostics (Text)

- **Message file, Step 3, Increment 6:**

```
INCREMENT     6 STARTS. ATTEMPT NUMBER  1, TIME INCREMENT  1.266E-02
 CONTACT PAIR (ASURF,BSURF) NODE 167 IS NOW SLIPPING.
 CONTACT PAIR (ASURF,BSURF) NODE 171 IS NOW SLIPPING.
 :
 :
 :
 :
 CONTACT PAIR (ASURF,BSURF) NODE 153 OPENS. CONTACT PRESSURE/FORCE IS -830689..
 CONTACT PAIR (ASURF,BSURF) NODE 161 OPENS. CONTACT PRESSURE/FORCE IS -1.43706E+006.
 CONTACT PAIR (ASURF,BSURF) NODE 165 OPENS. CONTACT PRESSURE/FORCE IS -1.03301E+006.
 CONTACT PAIR (CSURF,DSURF) NODE 363 OPENS. CONTACT PRESSURE/FORCE IS -3.43767E+006.
 CONTACT PAIR (ESURF,FSURF) NODE 309 IS NOW SLIPPING.
                  5 SEVERE DISCONTINUITIES OCCURRED DURING THIS ITERATION.
                  4 POINTS CHANGED FROM CLOSED TO OPEN
                  1 POINTS CHANGED FROM STICKING TO SLIPPING
```

Slave nodes that slip; stick/slip messages cause SDIs only if Lagrange friction is used or if slip reversal occurs

***PRINT, CONTACT=YES** causes this detailed printout.

(Useful for troubleshooting)

Incompatibilities detected in the assumed contact state → SDI

Due to slip reversal →

Contact Diagnostics (Text)

- Message file, Step 3, Increment 6 (cont'd):

```
        CONVERGENCE CHECKS FOR SEVERE DISCONTINUITY ITERATION     1

MAX. PENETRATION ERROR -8.1463E-009 AT NODE 331 OF CONTACT PAIR (ESURF,FSURF)
MAX. CONTACT FORCE ERROR -4184.86 AT NODE 363 OF CONTACT PAIR (CSURF,DSURF)
       THE ESTIMATED CONTACT FORCE ERROR IS LARGER THAN THE TIME-AVERAGED FORCE.

AVERAGE FORCE                   5.350E+03   TIME AVG. FORCE       3.137E+03
LARGEST RESIDUAL FORCE         -1.200E+04   AT NODE         333   DOF   2
LARGEST INCREMENT OF DISP.     -7.783E-04   AT NODE         329   DOF   2
LARGEST CORRECTION TO DISP.    -1.684E-05   AT NODE         337   DOF   2
         FORCE     EQUILIBRIUM NOT ACHIEVED WITHIN TOLERANCE.

AVERAGE MOMENT                   110.       TIME AVG. MOMENT
ALL MOMENT    RESIDUALS ARE ZERO
LARGEST INCREMENT OF ROTATION   1.847E-33   AT NODE         100   DOF   6
LARGEST CORRECTION TO ROTATION  6.454E-34   AT NODE         300   DOF   6
         THE MOMENT    EQUILIBRIUM EQUATIONS HAVE CONVERGED
```

Convergence checks for contact state

Convergence checks for equilibrium

Not only is the contact incompatibility too large, but force equilibrium has not been achieved either

Contact Diagnostics (Text)

- Four additional iterations are required; the first two are SDIs (involve contact incompatibilities).
- In the final iteration both the contact and equilibrium checks pass and the increment converges

```
      CONVERGENCE CHECKS FOR SEVERE DISCONTINUITY ITERATION     2 ...
      CONVERGENCE CHECKS FOR SEVERE DISCONTINUITY ITERATION     3 ...
      CONVERGENCE CHECKS FOR EQUILIBRIUM ITERATION     1
      CONVERGENCE CHECKS FOR EQUILIBRIUM ITERATION     2

 MAX. PENETRATION ERROR -1.24301E-015 AT NODE 331 OF CONTACT PAIR (ESURF,FSURF)
 MAX. CONTACT FORCE ERROR -9.94745E-005 AT NODE 331 OF CONTACT PAIR (ESURF,FSURF)
         THE CONTACT CONSTRAINTS HAVE CONVERGED.

AVERAGE FORCE                        5.244E+03   TIME AVG. FORCE       3.120E+03
LARGEST RESIDUAL FORCE               -1.98       AT NODE        135    DOF  1
LARGEST INCREMENT OF DISP.          -7.809E-04   AT NODE        129    DOF  2
LARGEST CORRECTION TO DISP.          1.063E-08   AT NODE        135    DOF  2
         THE FORCE     EQUILIBRIUM EQUATIONS HAVE CONVERGED

AVERAGE MOMENT                        109.       TIME AVG. MOMENT       88.8
ALL MOMENT    RESIDUALS ARE ZERO
LARGEST INCREMENT OF ROTATION        1.925E-33   AT NODE        100    DOF  6
LARGEST CORRECTION TO ROTATION      -6.933E-38   AT NODE        100    DOF  6
         THE MOMENT    EQUILIBRIUM EQUATIONS HAVE CONVERGED
```

No SDIs in these iterations

Lecture 4 Summary

Review of Topics Discussed in this Lecture

- **Output of Contact Results**
- **Contact Pressure Accuracy**
- **Contact Diagnostics (Visual)**
- **Contact Diagnostics (Text)**
  - Keys to obtaining accurate results
    - Adequate mesh refinement
    - Ability of formulations to accurately pass “patch tests”
  - Troubleshooting problems in an analysis is facilitated by:
    - Having a high-level understanding of numerical methods that Abaqus uses for contact (subject of previous lecture)
    - Using diagnostic output
    - Having perspective on common sources of convergence difficulty (further discussion in next lecture)

Convergence Topics

Lecture 5

Overview

- **Review previous discussions related to convergence**
- **Static instabilities**
  - Unconstrained rigid body motion and negative eigenvalues
  - Regularization methods
- **Overconstraints**
- **Best practices for treating initial over closures**
- **Discouraging semi-obsolete features**

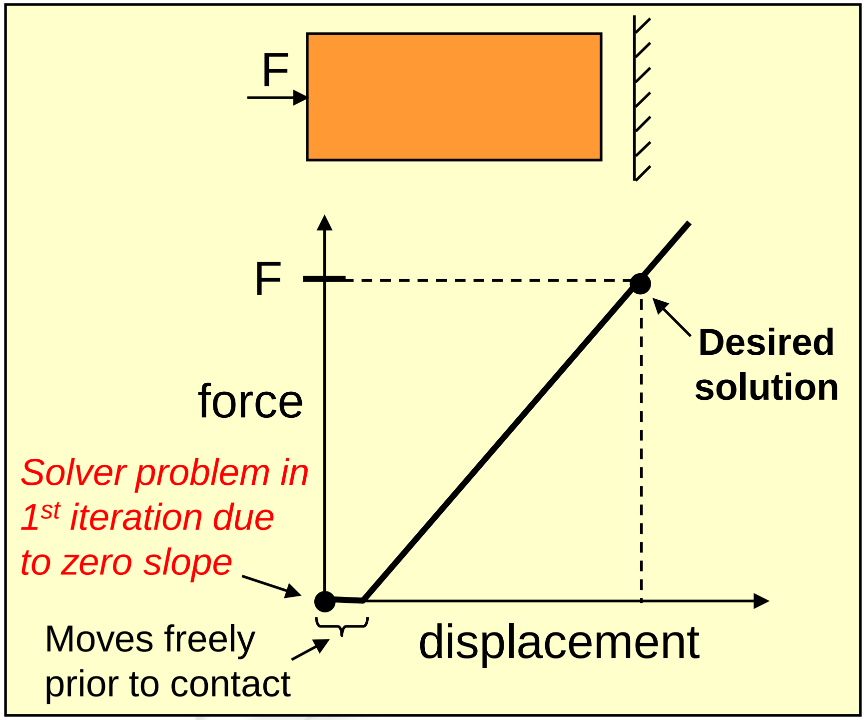
Already Discussed

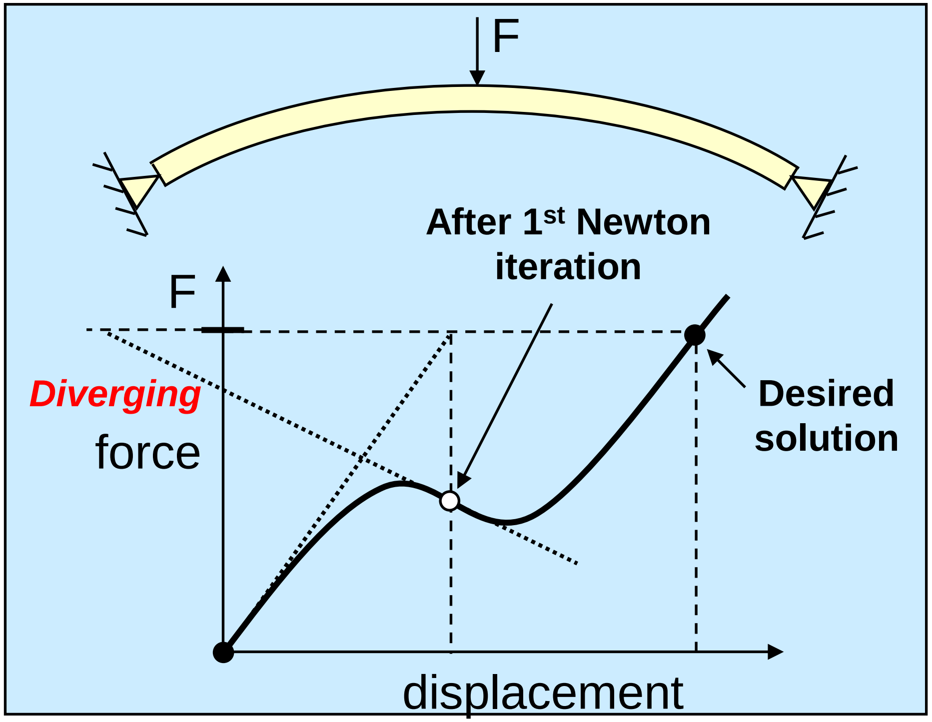
- **Newton iterations, radius of convergence, and incrementation**
- **Diagnostics output**
  - Helpful for determining location and cause of convergence problems
- **Changes in contact status (open/closed and slip/stick) are characterized as severe discontinuities by iteration control algorithm**
  - Strict enforcement: Change from no contact stiffness to ∞ stiffness
  - Penalty enforcement: Change from no contact stiffness to finite stiffness
    - Less severe
- **“Smooth” contact formulation characteristics enhance convergence**
  - E.g., continuity in nodal contact forces upon sliding
  - Surface-to-surface contact discretization is smoother than node-to-surface contact discretization
- **Also helpful for convergence:**
  - Smooth (and more accurate) representation of curved surfaces
  - Accounting for nonsymmetric stiffness terms in equation solver

Static Instabilities

Static Instabilities

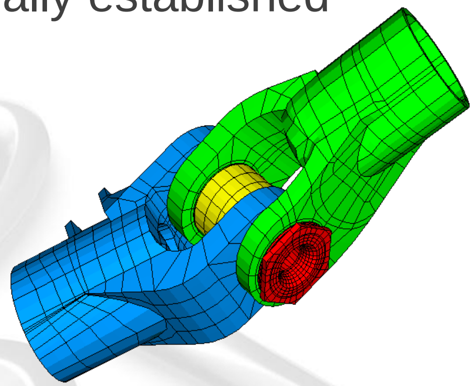
- **Types of instabilities**
  - Unconstrained rigid body modes
  - Geometric instabilities (snap through, etc.)
  - Material instabilities (softening)





Static Instabilities

- **Unconstrained rigid body motion**
  - Many mechanical assemblies rely on contact between bodies to prevent unconstrained rigid body motion
  - Often it is impractical or impossible to model such systems with contact initially established



Example with initial "play" between pin and other components

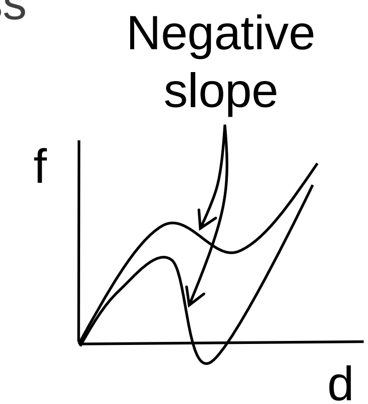
  - Without user intervention, Abaqus may report solver singularities in the message (**.msg**) file :

```
***WARNING: SOLVER PROBLEM.  NUMERICAL SINGULARITY WHEN
PROCESSING NODE 17
          D.O.F. 2 RATIO = 3.93046E+16
```

    - Often leads to slow or no convergence

Static Instabilities

- **"Negative eigenvalues"**
  - Nonlinear systems often experience temporary instabilities associated with a negative tangent stiffness for a particular incremental deformation mode
    - Geometric instability (snap through)
    - Material instability (softening)
  - Without intervention, Abaqus will report negative eigenvalues in the message (.msg) file
    - Often leads to slow or no convergence



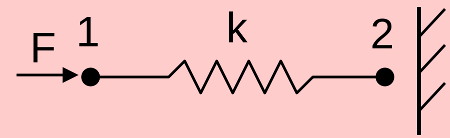
Static Instabilities

- **Intervention approaches**
  - Add boundary conditions (e.g., displacement-controlled loading)
  - Adjust initial contact state
  - Add stabilization stiffness (damping)
  - Consider inertia effects (dynamic analysis)

Unconstrained Rigid Body Motion during Static Analysis

- **Singular system of equations prior to establishing contact**

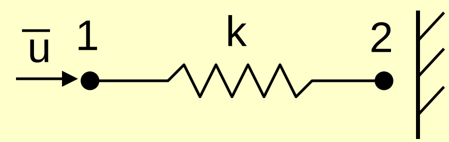
1-D representation



$$\begin{bmatrix} k & -k \\ -k & k \end{bmatrix} \begin{Bmatrix} u_1 \\ u_2 \end{Bmatrix} = \begin{Bmatrix} F \\ 0 \end{Bmatrix}$$

Determinant is 0 *(singular)*

- **Displacement-controlled loading prior to establishing contact avoids the singularity**



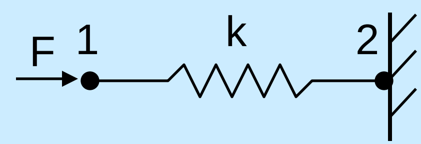
$$\begin{bmatrix} k \end{bmatrix} \begin{Bmatrix} u_2 \end{Bmatrix} = \begin{Bmatrix} ku_1 \end{Bmatrix}$$

Nonsingular,
sol'n: u2=u1=$\bar{u}$

- **Once contact is established, the system of equations is also stable for force-controlled loading**
  - Also true with penalty enforcement

$$\begin{bmatrix} k & -k \\ -k & k+k_p \end{bmatrix} \begin{Bmatrix} u_1 \\ u_2 \end{Bmatrix} = \begin{Bmatrix} F \\ 0 \end{Bmatrix}$$

Sol'n: u_1=F/k+F/k_p, u_2=F/k_p



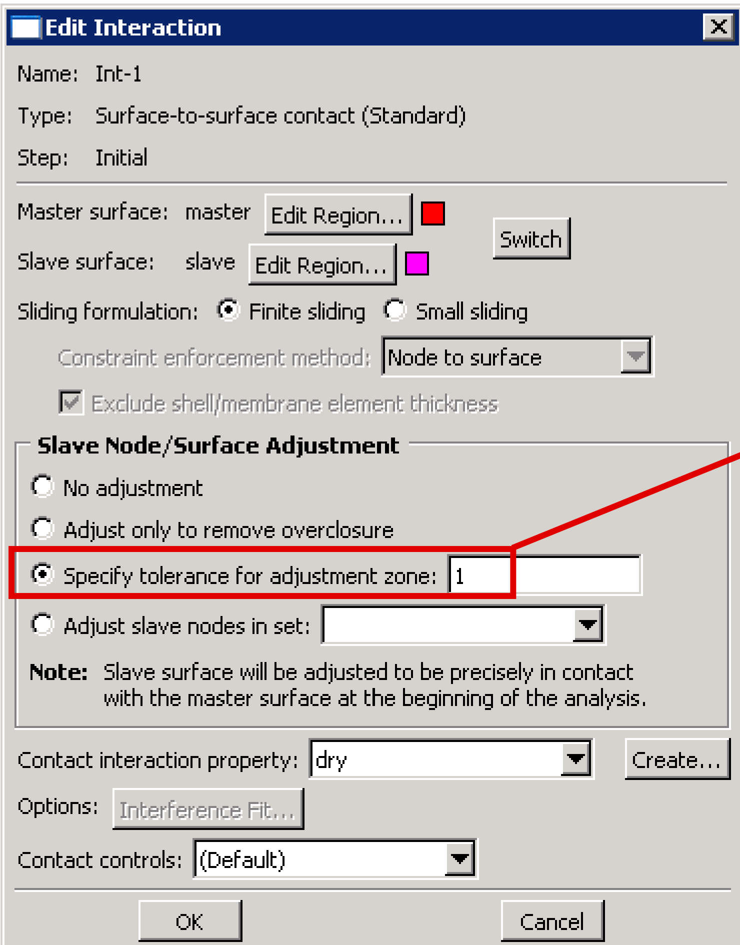
$$\begin{bmatrix} k & -k & 0 \\ -k & k & 1 \\ 0 & 1 & 0 \end{bmatrix} \begin{Bmatrix} u_1 \\ u_2 \\ \lambda \end{Bmatrix} = \begin{Bmatrix} F \\ 0 \\ 0 \end{Bmatrix}$$

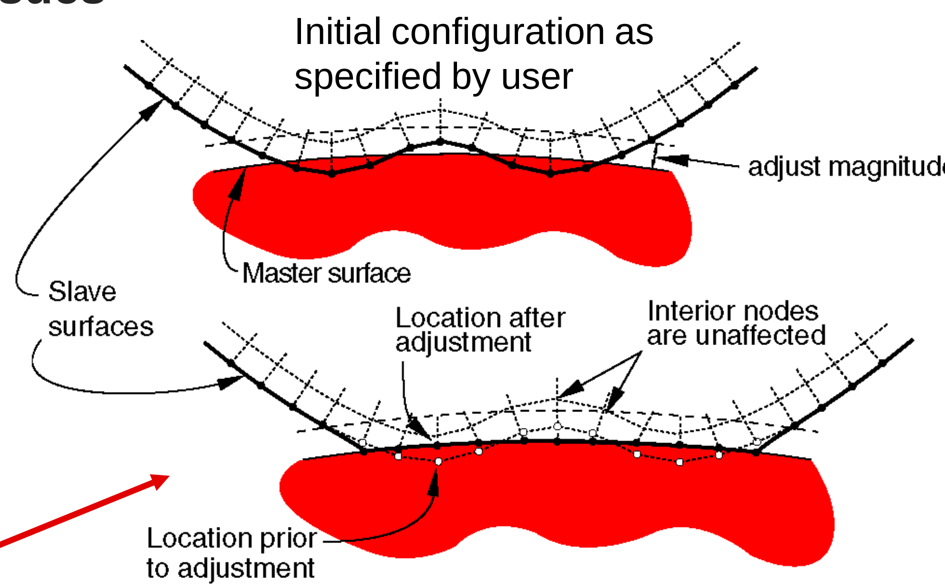
Nonsingular

Sol'n: u_1=F/k, u_2=0, λ=F

Avoiding Unintended Initial Gaps (adjustment zone)

- **Avoids some rigid-body-mode issues**
- **User interface for contact pairs:**



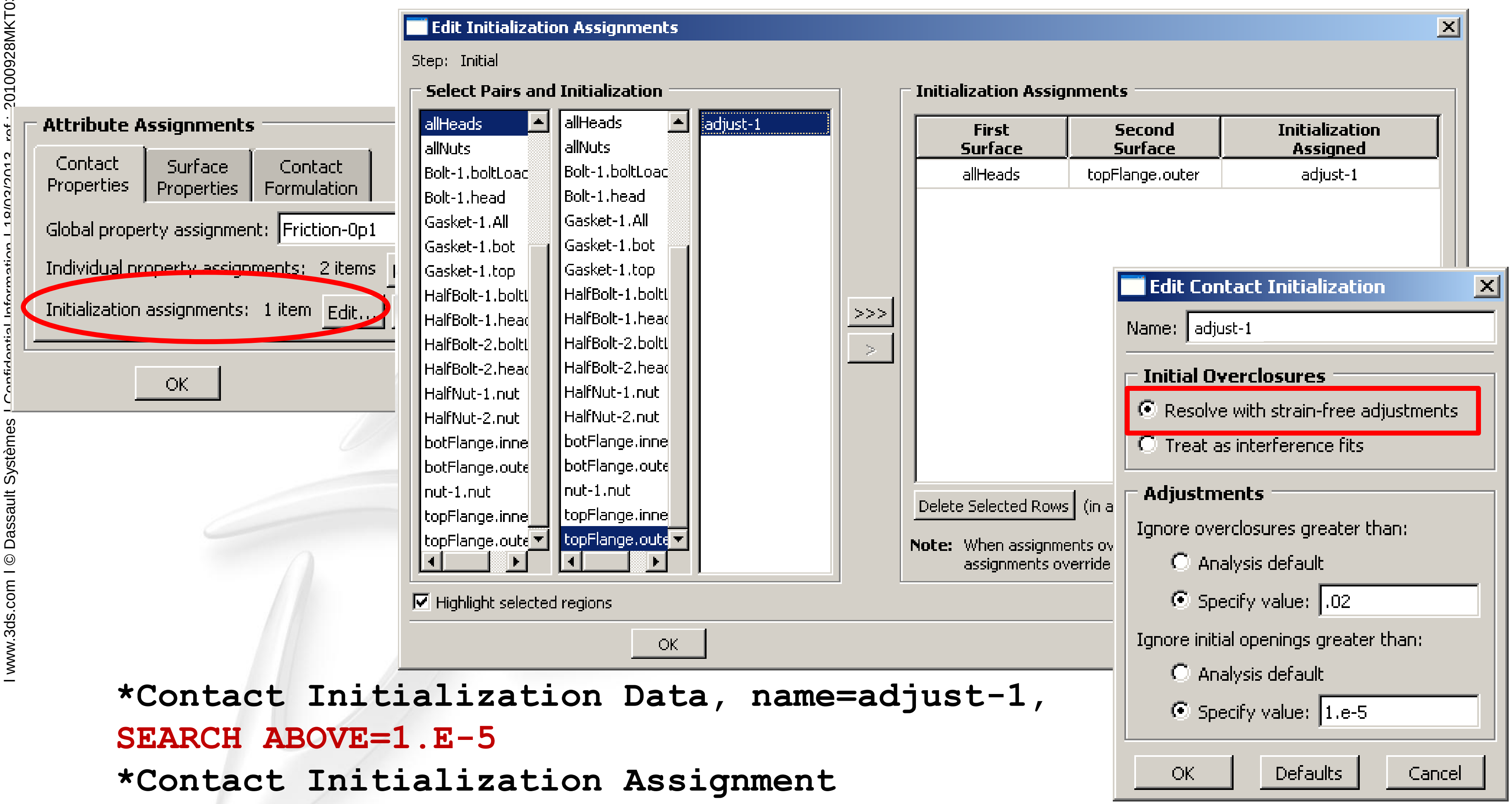


Configuration after adjustment and prior to start of analysis: slave nodes outside adjust bands are unaffected (some exceptions for S-to-S formulation)

***CONTACT PAIR, INTERACTION=DRY,**
ADJUST=*a*

Avoiding Unintended Initial Gaps (Adjustment Zone)

- User interface for general contact in Abaqus/Standard



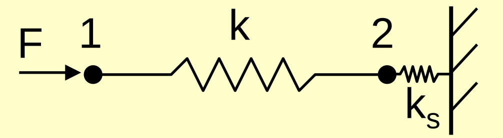

```
*Contact Initialization Data, name=adjust-1,
SEARCH ABOVE=1.E-5
*Contact Initialization Assignment
allHeads , topFlange.outer , adjust-1
```

Stabilization Methods

- **Artificial stiffness (“damping”)**
- **Preferred approaches**
  - Contact-based stabilization
    - Small resistance to relative motion between nearby surfaces while contact constraints are inactive
    - Quite effective for stabilizing initial rigid body modes prior to establishing contact
  - Volume-based stabilization
    - Adaptive stabilization throughout bodies
    - Quite effective for overcoming temporary instabilities that sometimes occur mid-analysis

Contact-Based Stabilization

- **Primarily targets cases with small initial "play" between surfaces**
- **Small resistance to incremental relative motion between *nearby* contact surfaces**
  - Resistance (stiffness) is a small fraction of the underlying element stiffness
  - Resistance is ramped to zero at end of step by default
  - Resistance is inversely proportional to the increment size ("damping")
- **Typically, minimal effect on results**
  - Energy dissipated by normal stabilization is nearly always insignificant
  - Energy dissipated by tangential stabilization can become large if large sliding occurs
- **User interface shown on next slide**



$$\begin{bmatrix} k & -k \\ -k & k+k_s \end{bmatrix} \begin{Bmatrix} u_1 \\ u_2 \end{Bmatrix} = \begin{Bmatrix} F \\ 0 \end{Bmatrix}$$

Nonsingular

After 1st *iteration*:
$u_1 = F/k + F/k_s$, $u_2 = F/k_s$

Likely to trigger a contact status change for the next iteration

Contact-Based Stabilization

- **User controls**

Use the default damping coefficient:

```
*CONTACT CONTROLS, STABILIZE
```

Scale the default damping coefficient:

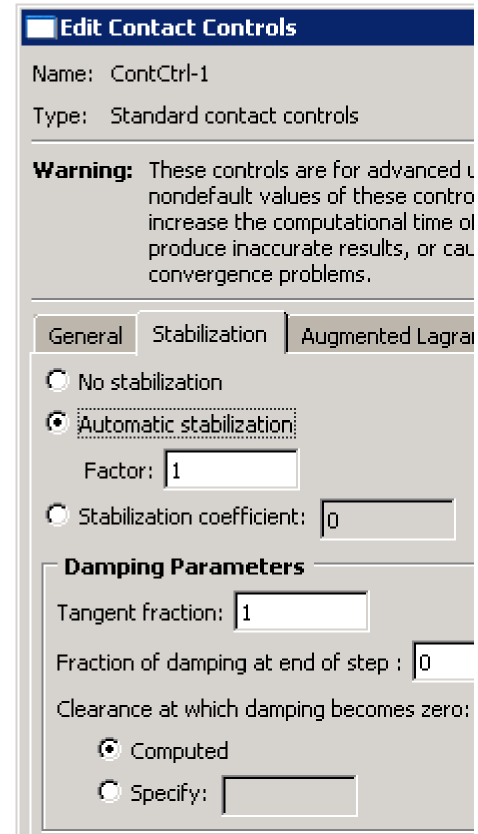
```
*CONTACT CONTROLS, STABILIZE=<factor>
```

Specify the damping coefficient directly:

```
*CONTACT CONTROLS, STABILIZE
 <damping factor>
```

Specify a nondefault ramp-down factor:

```
*CONTACT CONTROLS, STABILIZE
 , <ramp-down factor>
```



Decrease or increase the tangential damping or set it to zero:

```
*CONTACT CONTROLS, STABILIZE, TANGENT FRACTION=<value>
```

Contact-Based Stabilization

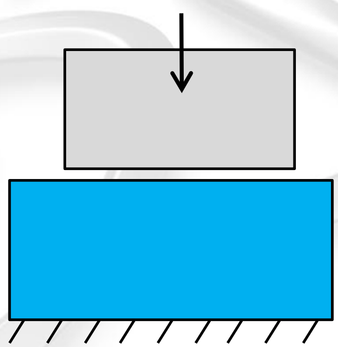
- **User controls (cont.)**

  New keyword interface for general contact added in Abaqus 6.10

  ***CONTACT STABILIZATION**

  Not yet supported in Abaqus/CAE

  - Specify local or global contact stabilization controls
  - First step-dependent suboption of *CONTACT for Abaqus/Standard
  - Not active by default (with one exception to be discussed); but when activated, the “built-in” settings target temporary, initial unconstrained rigid body modes



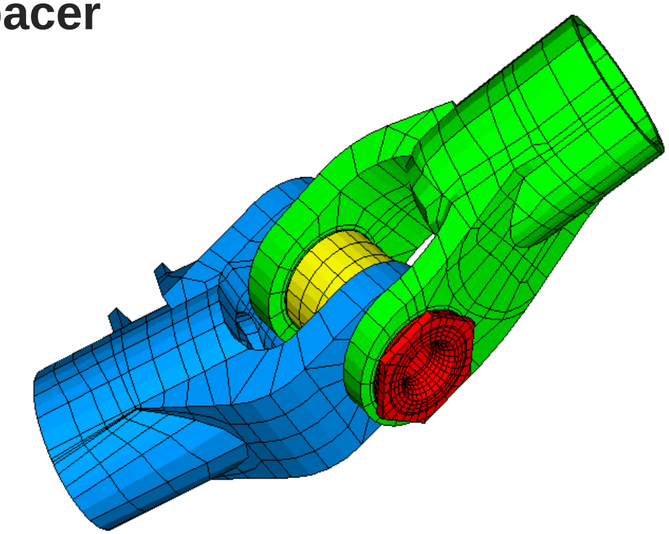
Comments on “built-in” settings:

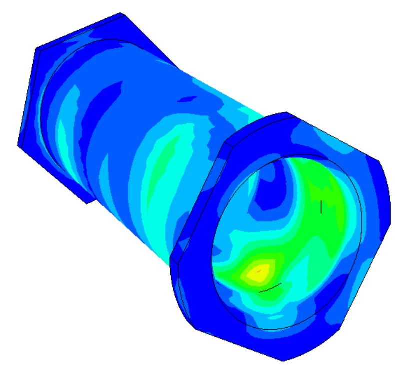
- No tangential stabilization
- Stabilization is aggressively ramped down over increments

Contact-Based Stabilization

- **Contact Pair Example: Joint with pin and spacer**
  - 105K degrees of freedom
  - Four bodies, connected by contact pairs

Contact Stabilization	No	Yes
Wallclock time (min)	226	53
# Increments	25	18
# Iterations	145	29

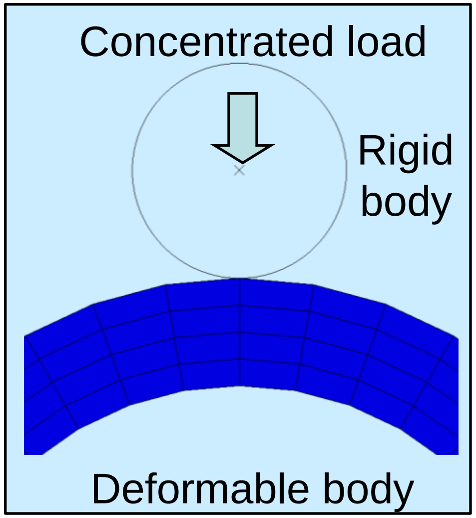




Mises stress in pin

Contact-Based Stabilization

- **Special case: Initially touching surfaces for surface-to-surface discretization**
  - Consider the case shown where the average gap > 0 for each slave node; thus:
    - Surface-to-surface contact constraints are initially *inactive*
    - Initial system of equations would have no resistance to the applied load



  - Stabilization stiffness **automatically added** for such cases (even if the point of touching does not correspond to a node)
    - Similar to the nondefault contact stabilization just discussed: Stabilization stiffness is zero by the end of the step and is inversely proportional to the increment size
    - Some differences: Activated automatically, acts only in the normal direction, and is more aggressively ramped off in early increments

Contact-Based Stabilization

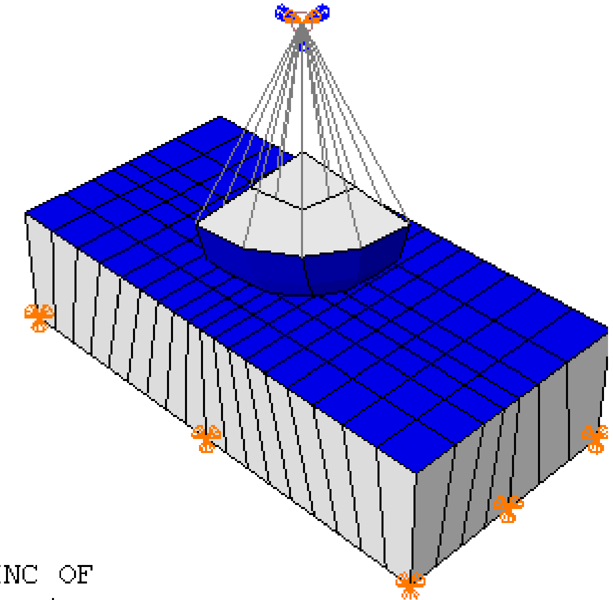
- **This special form of automatic stabilization is on by default for the finite-sliding, surface-to-surface formulation**
  - Cannot be applied to other formulations
- **Keyword interface**
  - Contact pairs

```
*CONTACT PAIR, TYPE=SURFACE TO SURFACE,
MINIMUM DISTANCE = [YES(DEFAULT)/NO]
```

  - General contact

```
*CONTACT INITIALIZATION DATA, NAME=xyz,
MINIMUM DISTANCE = [YES(DEFAULT)/NO]

*CONTACT

*CONTACT INCLUSIONS

*CONTACT INITIALIZATION ASSIGNMENT

 , , xyz
```

Contact-Based Stabilization



- **Example**
  - Information from status (**.sta**) file for this example is shown below

STEP	INC	ATT	SEVERE DISCON ITERS	EQUIL ITERS	TOTAL ITERS	TOTAL TIME/ FREQ	STEP TIME/LPF	INC OF TIME/LPF
1	1	1U	0	1	1	0.000	0.000	1.000
1	1	2	0	1	1	0.250	0.250	0.2500
1	2	1U	5	0	5	0.250	0.250	0.2500
1	2	2	1	1	2	0.313	0.313	0.06250
1	3	1	0	1	1	0.406	0.406	0.09375
1	4	1	0	1	1	0.547	0.547	0.1406
1	5	1	0	1	1	0.758	0.758	0.2109
1	6	1	0	1	1	1.00	1.00	0.2422

THE ANALYSIS HAS COMPLETED SUCCESSFULLY

Stabilization is too low (zero)

Adequate stabilization after cutback

Stabilization is ramped too low

Adequate stabilization after cutback; a contact constraint is now active

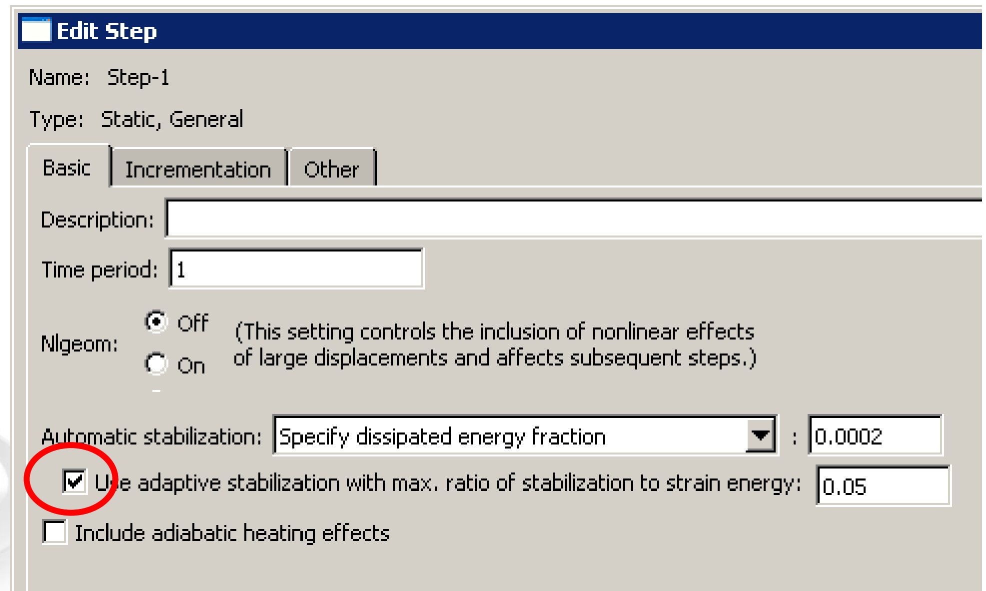
Good convergence behavior despite aggressive ramping down of stabilization stiffness

Note: This analysis does not run to completion with:
- ∗CONTACT CONTROLS, STABILIZE: Different ramp-down of stabilization stiffness
- Node-to-surface contact: Closest point does not correspond to a slave node

Volume-Based Stabilization

- **Also referred to as "static stabilization"**
  - Volume proportional "damping" targeting local dynamic instabilities
- **User interface** (see documentation for details)

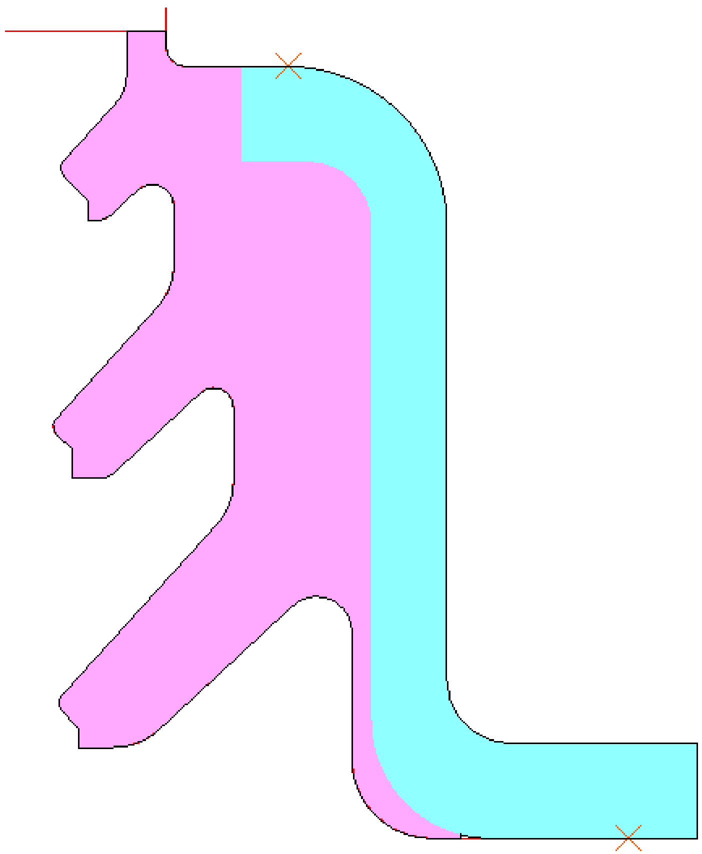
  *STATIC, STABILIZE



- **Applicable to the following quasi-static procedures:**
  - Static
  - Visco
  - Coupled Temperature-Displacement
  - Soils, Consolidation

Volume-Based Stabilization

- **Example of a static analysis using static stabilization**



Volume-Based Stabilization

- **Damping term in equilibrium equation:**

$$c\boldsymbol{M}^*\dot{\boldsymbol{u}} + \boldsymbol{I}(\boldsymbol{u}) = \boldsymbol{P},$$

quasi-velocity

mass matrix with unit density

damping factor (discussed on next page)

- Effect on equations solved in each Newton-Raphson iteration

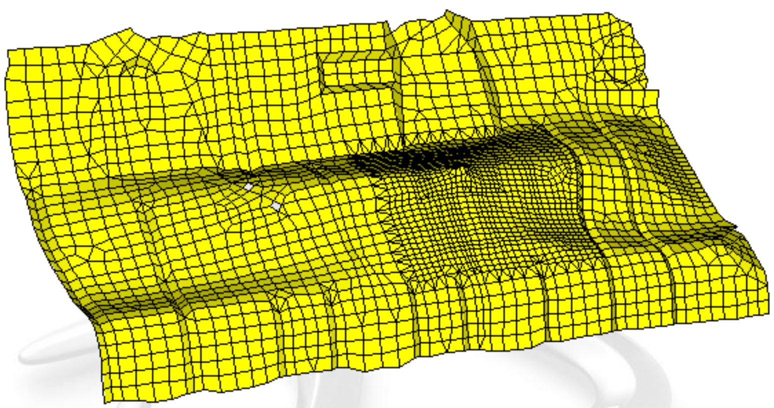
$$\left(\boldsymbol{K}_t + \frac{c}{\Delta t}\boldsymbol{M}^*\right)d\boldsymbol{u} = \boldsymbol{R} - c\boldsymbol{M}^*\frac{\Delta \boldsymbol{u}}{\Delta t}$$

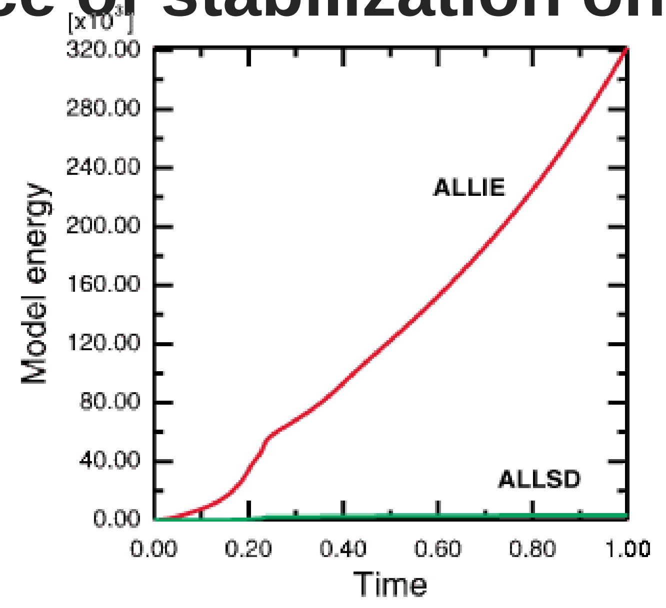
Volume-Based Stabilization

- **Automatic selection of the damping factor**
  - Abaqus automatically calculates the damping factor c
    - Varies in space and with time
    - Adaptive based on convergence history and ratio of energy dissipated by viscous damping to the total energy
  - Initial damping factor is based on the following premises:
    - The model's response in the first increment of a step to which damping is applied is stable
      - Not particularly effective for stabilizing unconstrained rigid body modes at the beginning of an analysis
    - Under stable circumstances the amount of dissipated energy should be very small

Volume-Based Stabilization

- **The amount of energy dissipation associated with the stabilization usually provides a good indication of the significance of stabilization on results**

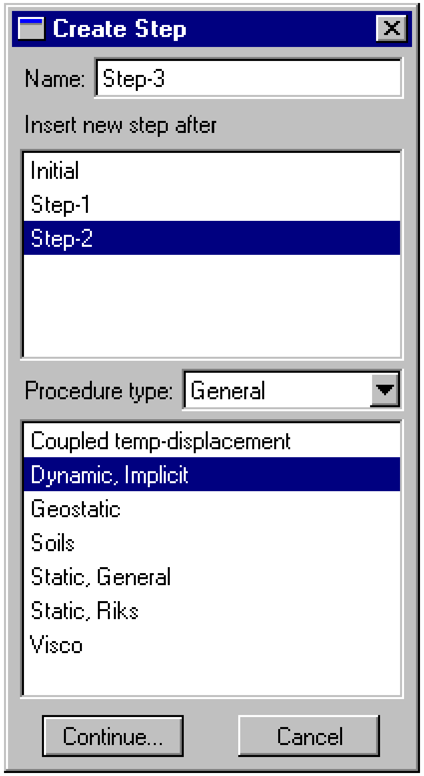




Here, the total energy dissipated due to stabilization is very small compared to the total energies involved in deformation

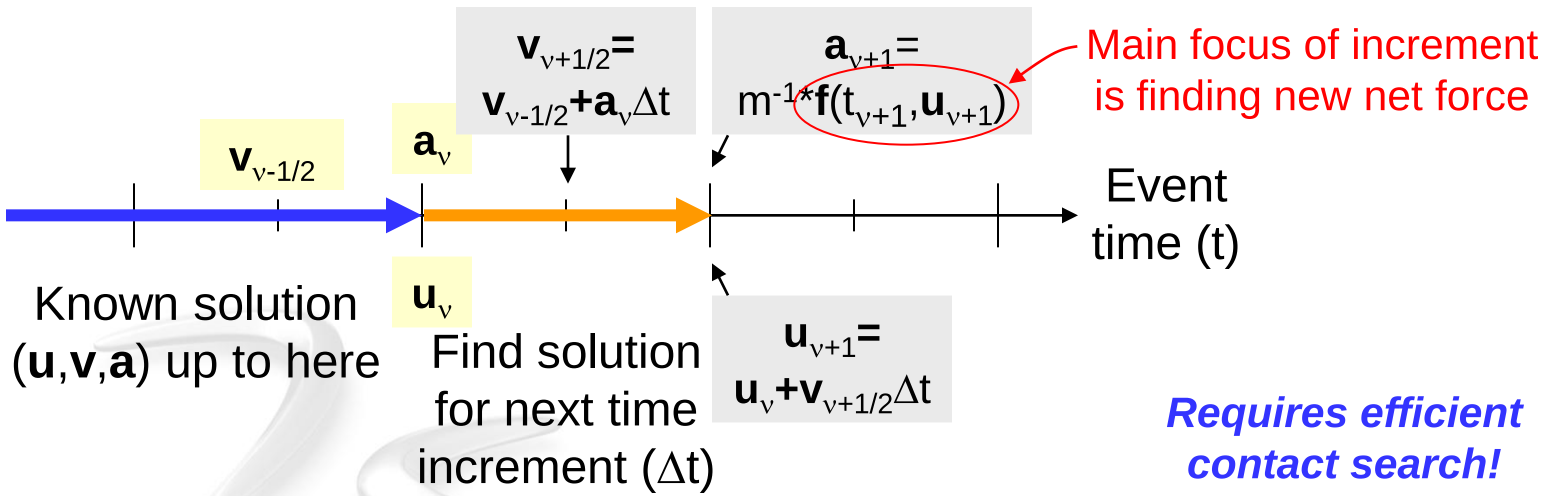
Dynamics

- **Another approach for overcoming static instabilities is to use a dynamic procedure**
  - Inertia is inherently stabilizing
  - Equation of motion: $\boldsymbol{M}\ddot{\boldsymbol{u}} + \boldsymbol{C}\dot{\boldsymbol{u}} + \boldsymbol{I}(\boldsymbol{u}) = \boldsymbol{P}.$
  - Abaqus provides implicit and explicit dynamics procedures
  - Implicit dynamics was enhanced in Abaqus 6.9-EF



Explicit Dynamics Time integration

- **March forward in time using the central difference method**



Requires efficient contact search!

- No matrix inversion (lumped mass) ⇒ *Each increment is fast*

  E.g., 1 second analysis time/increment for a 2 million element model

- Conditional stability (small Δt) ⇒ *Lots of increments*

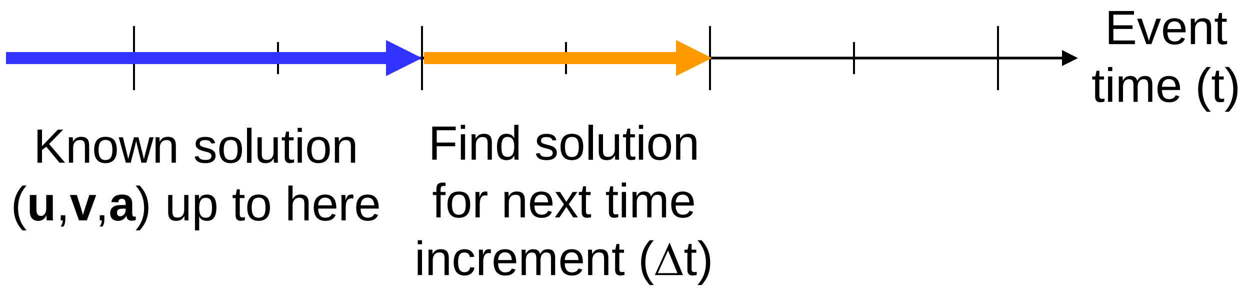
  E.g.,100,000 increments for a 0.1 second event

≈ 1 day analysis time

Small incremental motion simplifies update of contact conditions

Implicit Dynamics Time integration

- **March forward in time with implicit time integration**



- Solve nonlinear implicit system of equations each time increment
  - Equation solver and Newton iterations (like statics)
- The time integrators used by Abaqus/Standard are unconditional stability
  - Time increment size is governed by convergence rate and accuracy
- Compared to explicit time integration:
  - Higher cost per increment, but fewer increments (larger Dt)
  - Possibility of lack of convergence
    - Convergence criteria are very similar to statics
- Inertia has a stabilizing effect (for rigid body modes, etc.)

Abaqus 6.9-EF Enhancements to Implicit Dynamics

- **Prior to Abaqus 6.9-EF the direct-integration dynamics procedure typically used very small time increments for contact simulations**
  - Often not a viable approach
  - Example excerpt from status (`.sta`) file:

Increment number

Time increment size

1. **Not accepted due to contact status change**
2. **Cut back to average time of impact**
3. **"Impact" calculation**

1	11	1U	0	2	2	1.28e-005	1.28e-005	1.000e-005
1	11	2	0	2	2	1.79e-005	1.79e-005	5.088e-006
1	12	1	0	2	2	1.79e-005	1.79e-005	1.000e-011
1	13	1U	0	2	2	1.79e-005	1.79e-005	1.000e-005
1	13	2	0	2	2	2.37e-005	2.37e-005	5.808e-006
1	14	1	0	2	2	2.37e-005	2.37e-005	1.000e-011
1	15	1U	0	2	2	2.37e-005	2.37e-005	1.000e-005
1	15	2	0	2	2	3.02e-005	3.02e-005	6.556e-006
1	16	1	0	2	2	3.02e-005	3.02e-005	1.000e-011
1	17	1U	0	2	2	3.02e-005	3.02e-005	1.000e-005

Repeated pattern

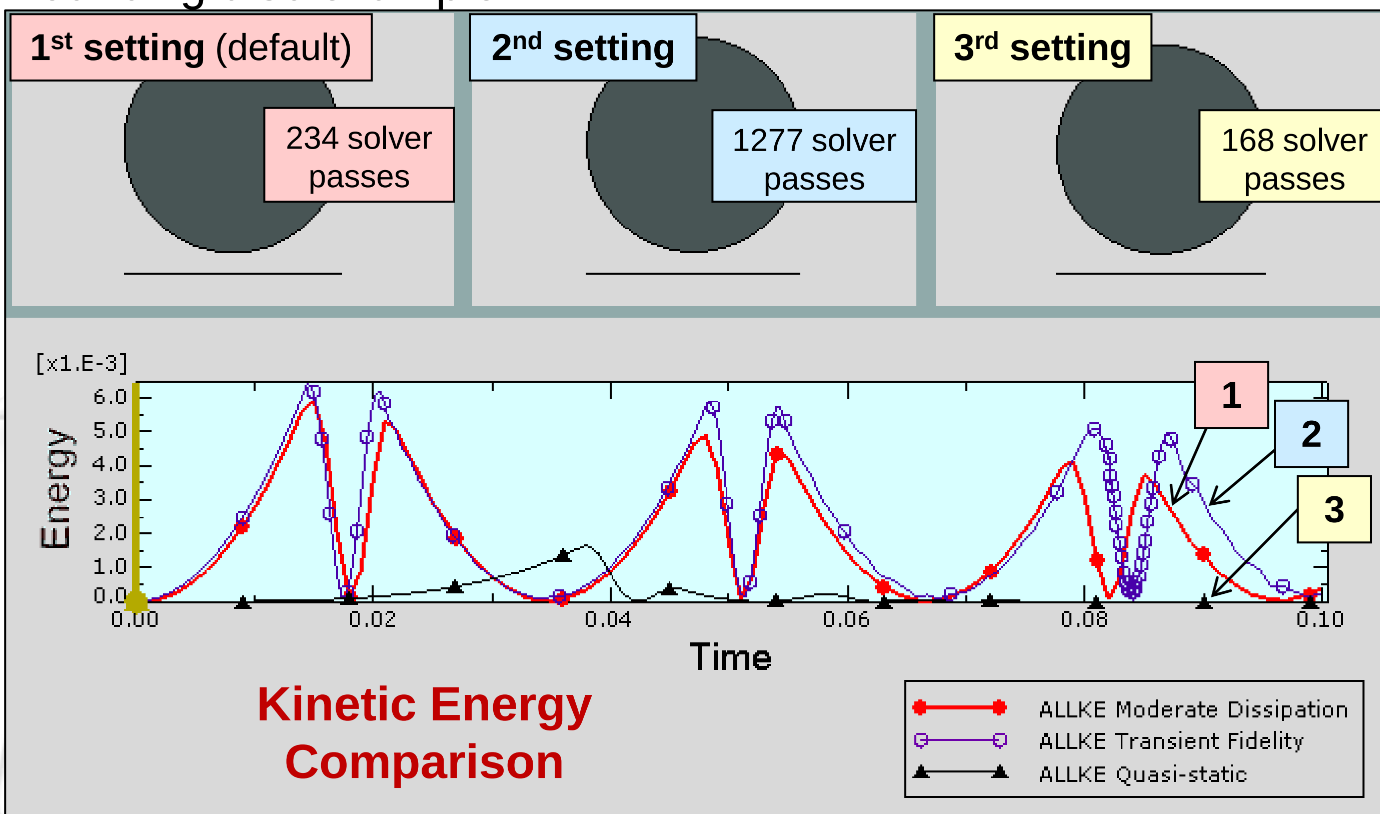
- **Time incrementation strategies first available in Abaqus 6.9-EF are better suited for contact analyses**

Abaqus 6.9-EF Enhancements to Implicit Dynamics

- **High-level parameter:**

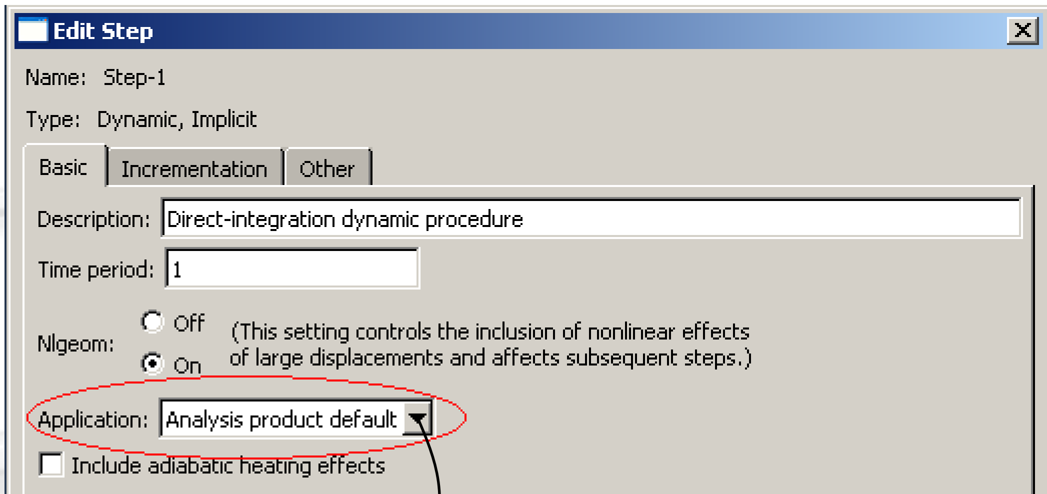
*Dynamics, Application =
- Moderate Dissipation ← **Default for contact models**
- Transient Fidelity ← **Similar behavior to Abaqus 6.9; remains default for noncontact models**
- Quasi-static ← **Intended for quasi-static modeling**

Bouncing disc example:



Implicit Dynamics Enhancements in Abaqus/CAE

- **Key implicit dynamics enhancements supported in Abaqus/CAE 6.10**
  - See Abaqus 6.10 Release Notes entry 6.2



Other choices:
Moderate Dissipation
Transient Fidelity
Quasi-static

Abaqus 6.9-EF Enhancements to Implicit Dynamics

- **"Moderate Dissipation" setting** (vs. "Transient Fidelity" setting)
  - Some additional numerical dissipation
  - Better convergence behavior for contact applications
  - Fewer solver passes
  - Reasons:
    1. No direct enforcement of velocity and acceleration compatibility across contact interfaces
    2. No half-increment residual tolerance
    3. Different parameter settings for the HHT time integrator

HHT time integrator

$$\mathbf{u}_{t+\Delta t} = \mathbf{u}_t + \Delta t \mathbf{v}_t + \Delta t^2 \left[\left(\tfrac{1}{2} - \beta \right) \mathbf{a}_t + \beta \mathbf{a}_{t+\Delta t} \right]$$

$$\mathbf{v}_{t+\Delta t} = \mathbf{v}_t + \Delta t \left[(1-\gamma) \mathbf{a}_t + \gamma \mathbf{a}_{t+\Delta t} \right]$$

$$-\mathbf{R}_{t+\Delta t} = \mathbf{M}\mathbf{a}_{t+\Delta t} + (1+\alpha)(\mathbf{I}-\mathbf{P})_{t+\Delta t} - \alpha (\mathbf{I}-\mathbf{P})_t$$

$$-\tfrac{1}{2} \le \alpha \le 0 \qquad \beta = \tfrac{1}{4}(1-\alpha)^2 \qquad \gamma = \tfrac{1}{2} - \alpha$$

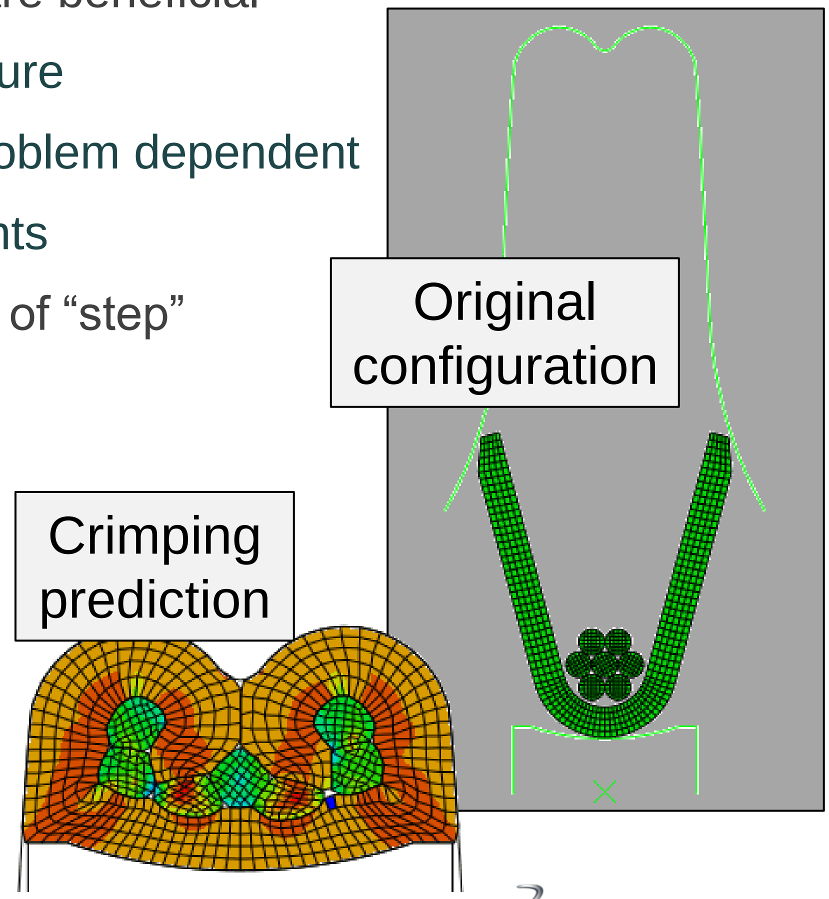
Application setting	HHT parameters		
	α	β	γ
Moderate dissipation	≈-0.41	0.5	≈0.91
Transient fidelity	-0.05	≈0.28	0.55

Abaqus 6.9-EF Enhancements to Implicit Dynamics

- **Comments on Application = Quasi-static**
  - Mainly intended for cases in which a static solution is desired but stabilizing effects of inertia are beneficial
    - Unable to converge with static procedure
    - Performance vs. Abaqus/Explicit is problem dependent
    - Also applicable to some dynamic events
  - Default amplitude type is “ramp” instead of “step”
    - Like the general static procedure
  - High numerical dissipation
    - Backward Euler time integrator

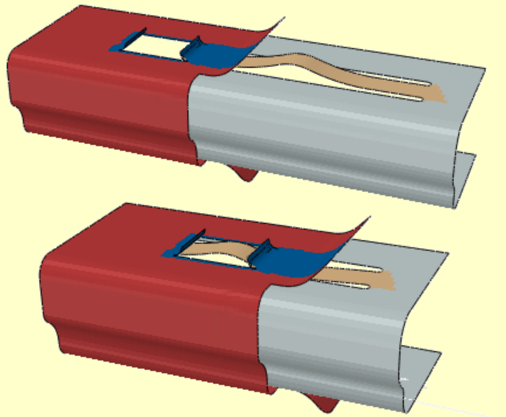
$$\mathbf{u}_{t+\Delta t} = \mathbf{u}_t + \Delta t \mathbf{v}_{t+\Delta t}$$
$$\mathbf{v}_{t+\Delta t} = \mathbf{v}_t + \Delta t \mathbf{a}_{t+\Delta t}$$
$$-\mathbf{R}_{t+\Delta t} = \mathbf{M}\mathbf{a}_{t+\Delta t} + \left(\mathbf{I} - \mathbf{P}\right)_{t+\Delta t}$$

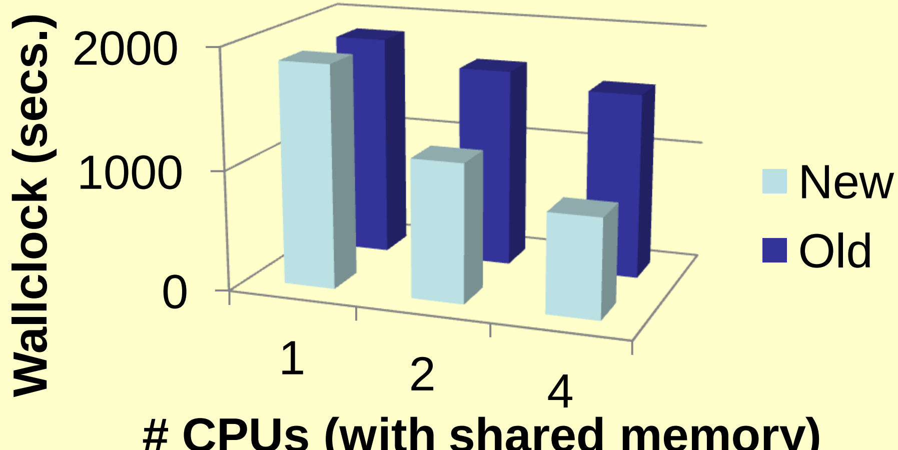
Wire crimping example



Thread-Parallel Elements and Contact Search for Dynamics

- **Parallel performance enhancement in Abaqus/Standard 6.10**
  - Removed restrictions on thread parallelization
  - Affects most contact analyses run in parallel
  - Influence on run-time can be quite dramatic for moderate-sized models with many increments





Perspectives on Implicit Dynamics (Direct Integration)

- **Each Newton iteration considers a system of equations of the form**

  $\mathbf{K'}\Delta\mathbf{U} = \mathbf{R'}$

  - **K´** and **R´** incorporate *static terms **plus** inertia & damping terms*
- **For trapezoidal rule of time integration (α=0, β=1/4, γ=1/2):**

  $\mathbf{K'} = \mathbf{K} + (4/\Delta t^2)\,\mathbf{M}$ (similar for other time integrators)

  - Some singular modes of **K** (static stiffness) are not singular for **K´**
    - Key example: Unconstrained rigid body modes
  - Stabilizing effects of inertia increase after a cut-back in the increment size (note Δt^2 in denominator)
    - Inertia effects should stabilize a negative eigenvalue of **K** *if the time increment is small enough*
    - Typical entries of **M** are typically orders of magnitude smaller than those of **K**
- **Use of other stabilization methods can enable larger Δt for dynamics**

Perspectives on Implicit Dynamics (Direct Integration)

- **Stiffness proportional (beta) Rayleigh damping in the material often improves convergence behavior without significantly affecting results**
  - Stabilizes high-frequencies
    - Whereas inertia effect on **K´** has most effect on low frequencies
  - Not active by default

```
*Material
...
*Damping, Beta=βR
```

Abaqus/CAE:
Property module: material editor: **Mechanical Damping**: **Beta**: β_R

This is a different “beta” than the “beta” associated with HHT and Newmark time integrators!

Perspectives on Implicit Dynamics (Direct Integration)

- **Comparison to statics**
  - Pure static analysis is usually more efficient than quasi-static analysis with the dynamic procedure if a model is statically stable
  - Quasi-static analysis with the dynamic procedure should be more robust
    - But good to supplement with other stabilization methods
- **Comparison to explicit dynamics**
  - Cost of increments/iterations vs. number of increments/iterations
    - Relative overall performance is problem dependent
  - Satisfaction of residual tolerances in implicit only
  - Effects of “mass scaling” (the only way to scale the mass in Abaqus/Standard is to adjust the density):
    - Increases stable time increment in Abaqus/Explicit
    - Increases inertia effects in both

Static Instabilities (Summary)

- **Have discussed several ways to address static instabilities**
  - Boundary conditions
  - Avoiding unintended initial gaps
  - Contact-based stabilization
  - Volume-based (static) stabilization
  - Dynamic analysis (accounting for inertia effects)
- **Abaqus Analysis User's Manual contains more information on these *and other methods***
  - Automatic stabilization of unstable problems
  - Automatic stabilization of rigid body motions in contact problems
  - The Riks method
  - Viscous regularization
  - Contact damping
  - Spring elements
  - Dashpot elements

Overconstraints

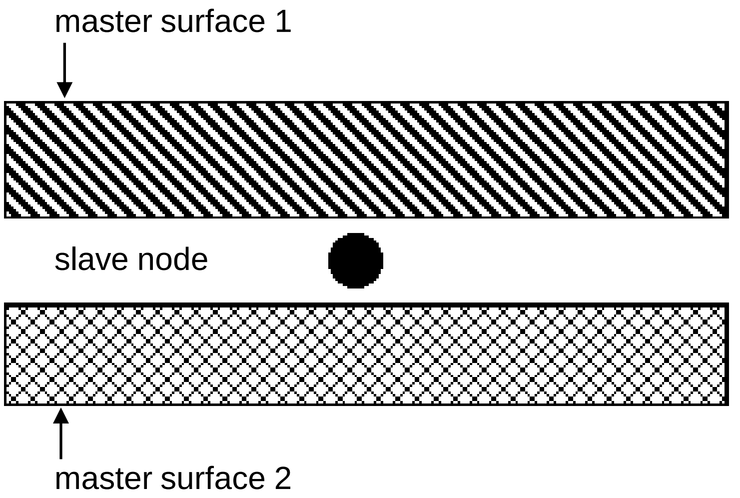
SIMULIA

Overview

- **Review previous discussions related to convergence**
  - Newton iterations
  - Severe discontinuities
  - Desirable formulation characteristics
- **Static instabilities**
  - Unconstrained rigid body motion and negative eigenvalues
  - Regularization methods
- **Overconstraints**
- **Best practices for treating initial overclosures**

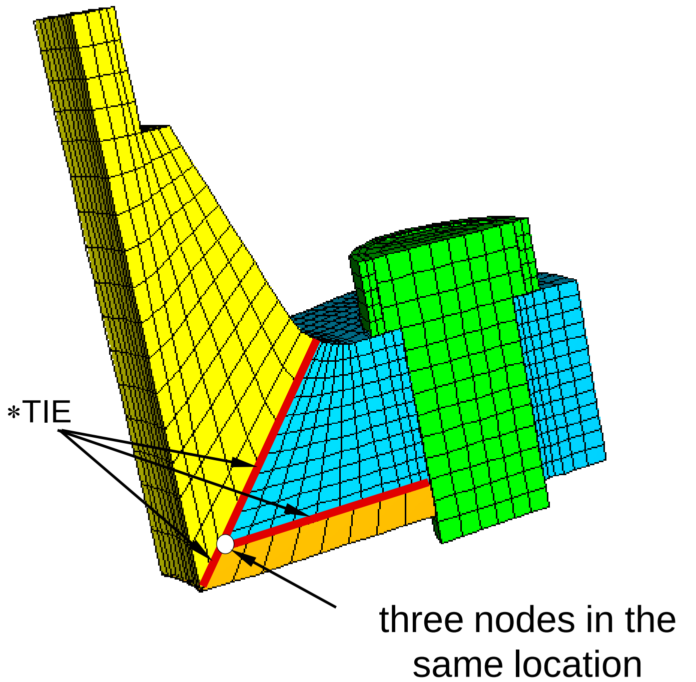
Overconstraints

- **Overconstraining the model**
  - Lagrange multipliers that impose contact constraints are indeterminate when node is overconstrained
    - Analyses will typically fail in such cases
  - This situation occurs when multiple kinematic (boundary condition, contact, or MPC) constraints act in same direction on same node
    - May be caused by single slave node interacting with a number of different master surfaces from different contact pairs



Overconstraints

- **Abaqus automatically resolves limited set of consistent overconstraints**
  - Overconstraints resolved before analysis involve intersections of boundary conditions, rigid bodies, and tie constraints
  - Overconstraints resolved during analysis involve intersections of contact interactions with boundary conditions and tie constraints



Overconstraints

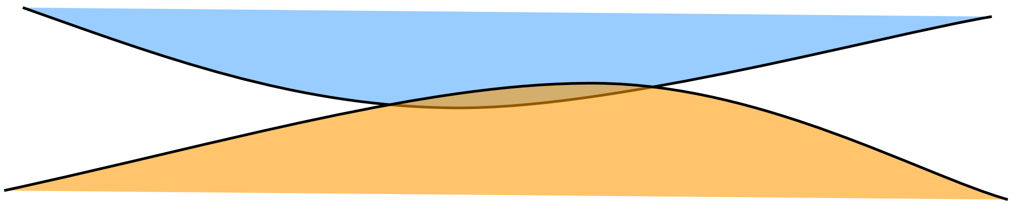
- **If overconstraint cannot be resolved automatically by Abaqus:**
  - A zero pivot warning message will typically be reported to the message (`.msg`) file (by the equation solver)
  - You will need to:
    - Identify and remove the overconstraint manually, or
    - Switch to a penalty form of constraint enforcement

- **Comments on overlapping constraints enforced with a penalty method**
  - Usually not catastrophic
  - But can degrade convergence (still try to avoid)
  - Tends to become more of an issue if the penalty stiffness is greater than the default

Best Practices for Treating Initial Overclosures

SIMULIA

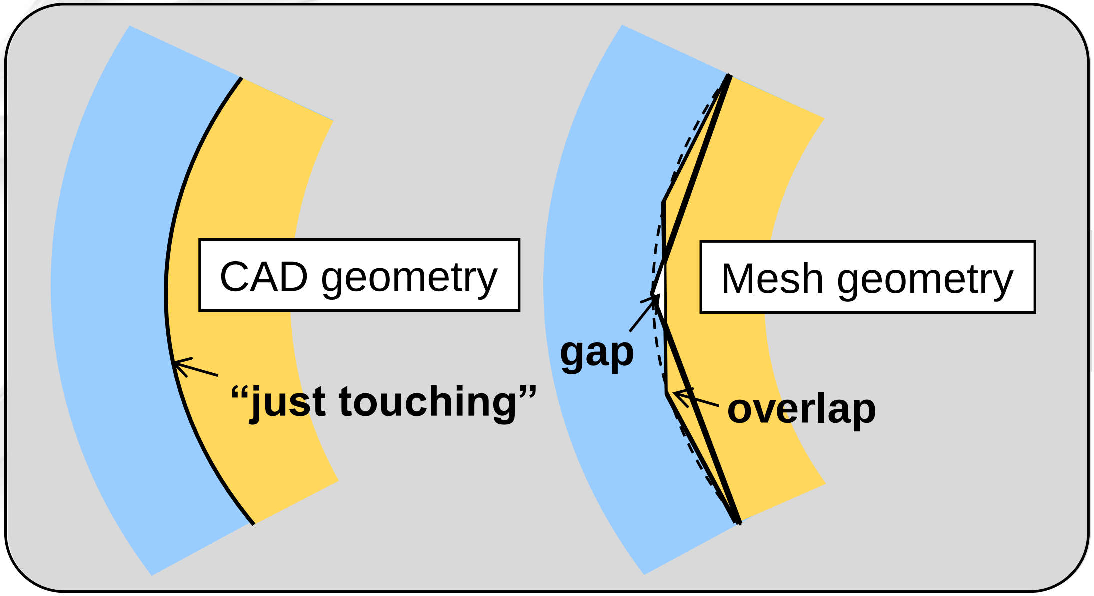
Initial Overclosure



- **Key question: Are the initial overclosures intended as interference fits or unintended?**
  - It’s really up to the user to provide the answer to this question

Initial Overclosure

- **Common causes of initial overclosure**
  - Intended
    - Modeling interference fit in Abaqus/Standard
  - Unintended
    - Shell thickness not accounted for in preprocessor
    - Preprocessor error
    - Discretization of curved surfaces (without geometry corrections)

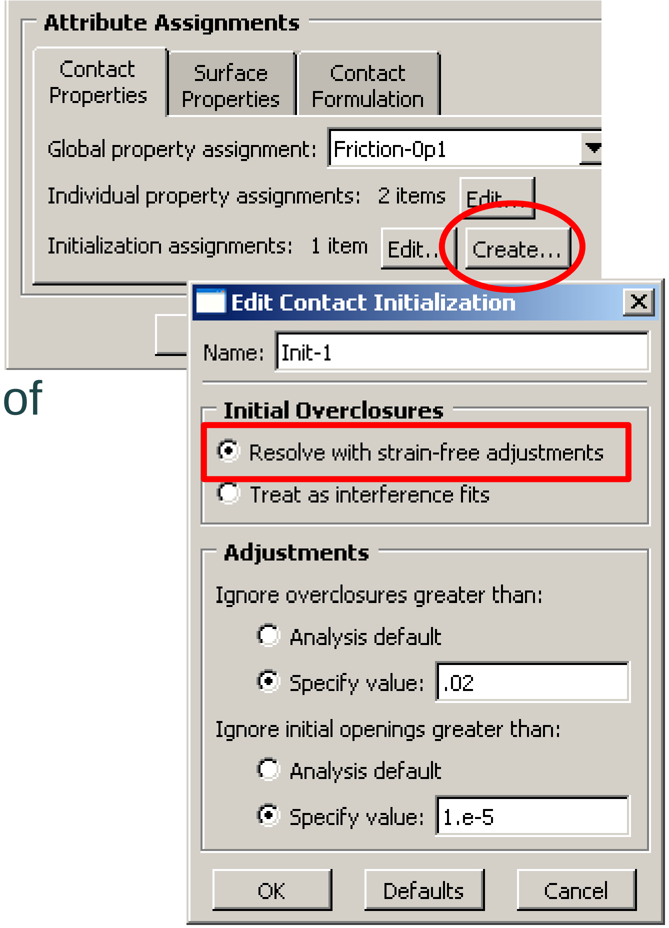


Initial Overclosure

- **Default treatment**
  - General contact in Abaqus/Standard and Abaqus/Explicit
    - Treats initial overclosures (within a given tolerance) with strain-free adjustments by default
    - Overclosures greater than specified tolerance ignored
    - Alternatively, in Abaqus/Standard overclosures can be treated as interference fits that are gradually resolved over the first step
  - For contact pairs in Abaqus/Standard
    - Treat initial overclosures as interference fits by default
    - Resolve all interference in the first (i.e., a **single**) **increment**
      - **Can cause convergence difficulty because the "loading" does not scale with the increment size**
    - Alternatively, overclosures can be resolved gradually or via strain-free adjustments

Strain-Free Adjustments

- **General contact in Abaqus/Standard**
  - By default, contact initialization removes small initial overclosures via stain-free adjustments
    - Default tolerance based on size of underlying element facets
  - Initial gaps remain unchanged by default adjustments
  - Optionally, large initial overclosures and initial gaps can also be adjusted
    - Specify search distances above and below surfaces
    - Search above to close gaps (discuss previously)
    - Search below to increase default overclosure tolerance

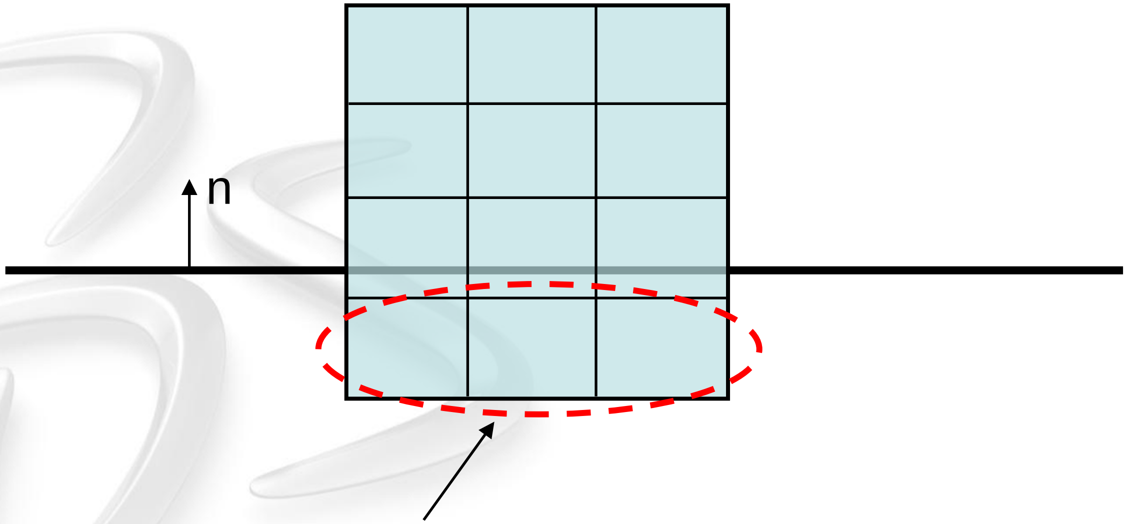



```
*Contact Initialization Data,
  name=Init-1,
  SEARCH ABOVE=distance,
  SEARCH BELOW=distance
*Contact Initialization Assignment
 , , Init-1
```

SIMULIA

Strain-Free Adjustments

- **Warning: Only slave surface nodes are relocated**
  - Gross (large) adjustments can severely distort initial element shapes
  - You should rely only on strain-free adjustments to resolve small initial overclosures (relative to element dimensions)



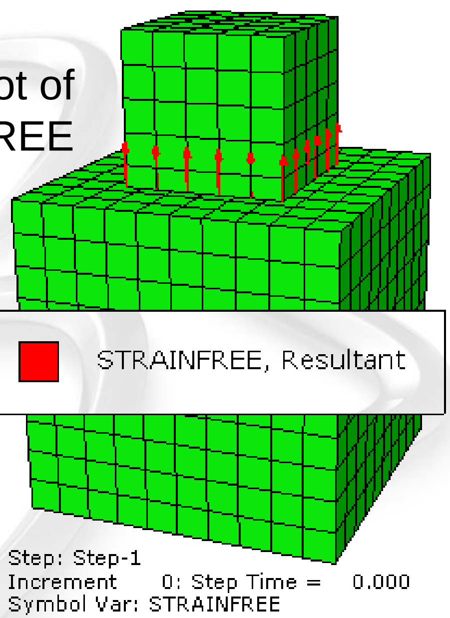
Element inversion (negative volume) will occur after strain-free adjustments

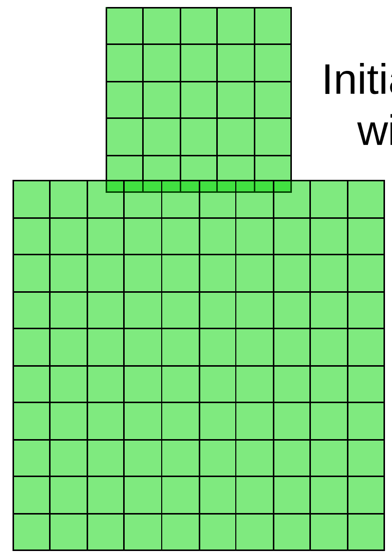
Introduced in Abaqus 6.9-EF

Visualizing Strain-Free Adjustments

- ***Nodal* output variable called "STRAINFREE" provided to visualize strain-free adjustments in Abaqus/Standard**
  - Output variable written by default if any initial strain-free adjustments are made
  - Variable *available only in the initial output frame at t=0*

Symbol plot of STRAINFREE





Initial configuration without contact

Visualizing Strain-Free Adjustments

- **The following inconsistency exists between Abaqus/Standard and Abaqus/Explicit with respect to strain-free adjustments:**

$$\mathbf{x} = \mathbf{x_o} + \mathbf{u}$$

Explicit adjusts **u**

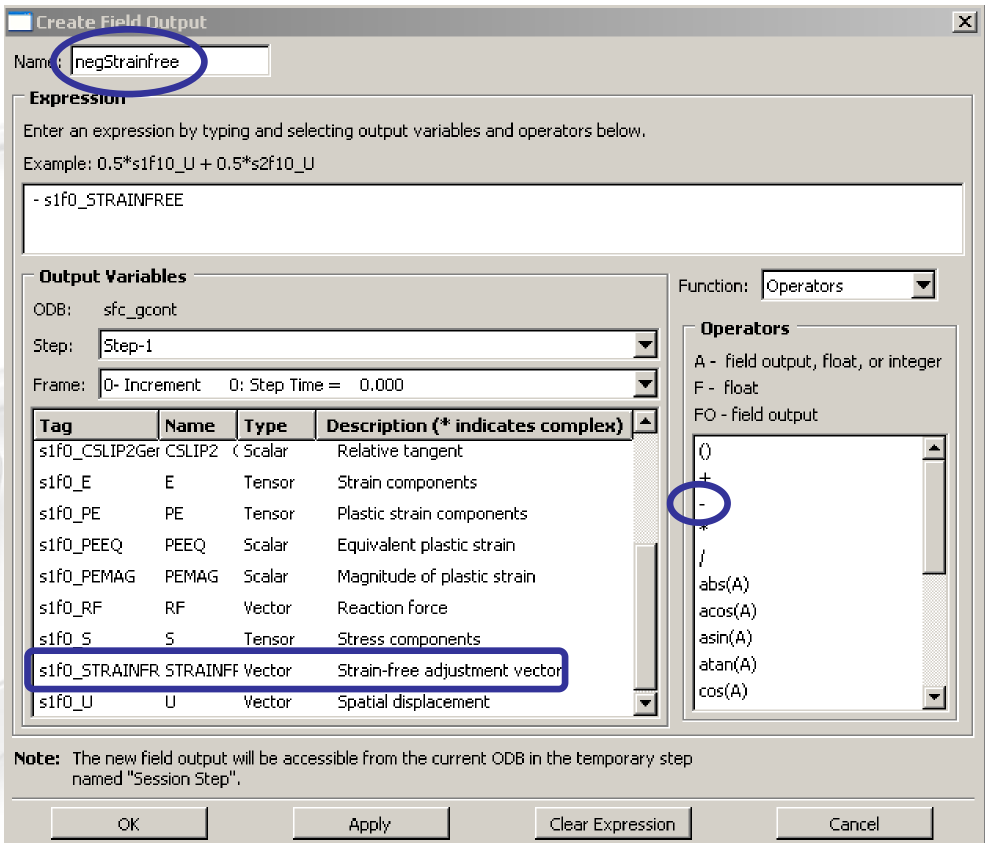
Standard adjusts $\mathbf{x_o}$

Desired aspect to visualize	**Technique in Abaqus/Viewer**	
	Abaqus/Standard model	Abaqus/Explicit model
Nodal adjustment vectors	Symbol plot of STRAINFREE at t=0	Symbol plot of U at t=0
Nodal adjustment magnitudes	Contour plot of STRAINFREE at t=0	Contour plot of U at t=0
Adjusted configuration	Undeformed shape or deformed shape at t=0	Deformed shape at t=0
Configuration prior to adjustments	Substitute -STRAINFREE for U in deformed plot (t=0)	Undeformed shape

Visualizing Configuration Prior to Adjustments for Abaqus/Standard

1. Create a field output variable equal to –STRAINFREE

- Abaqus/Viewer: Tools→Create Field Output→From Fields
- Choose a name for the new variable (“negStrainfree” in this example)
- Choose “-” operator and STRAINFREE output variable



Visualizing Configuration Prior to Adjustments for Abaqus/Standard

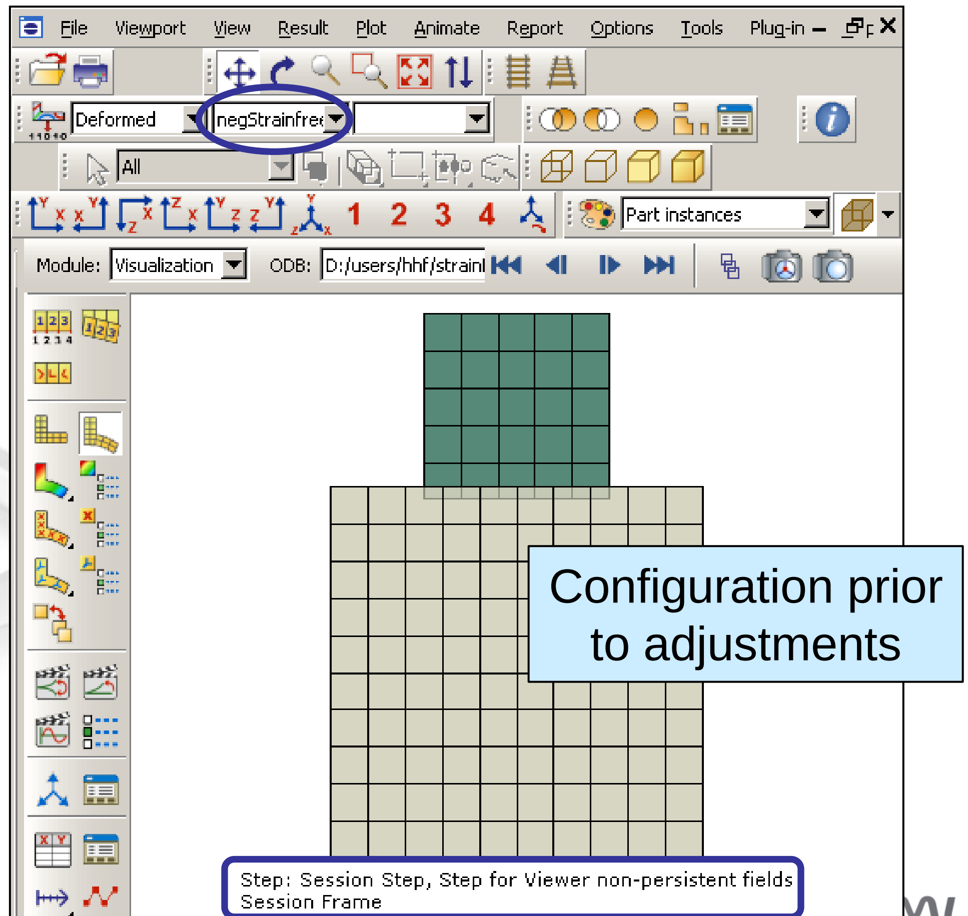
2. **View deformed plot based on this variable instead of U**
   - Abaqus/Viewer: Result→Step/Frame→Choose the "Session Step"
   - Make a deformed plot with the new variable driving the "displacements"

Configuration that appears in the plot

$$x = x_o + \text{negStrainfree}$$

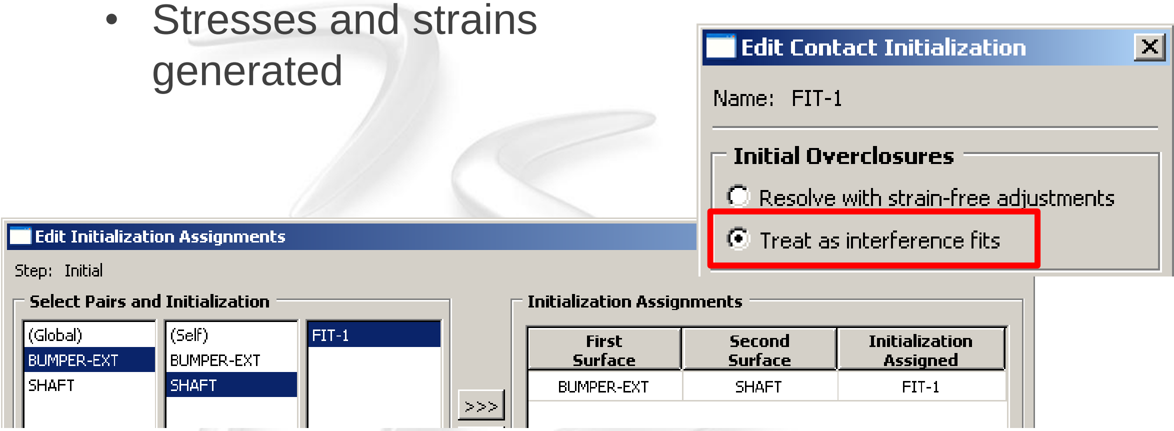
Configuration with strain-free adjustments

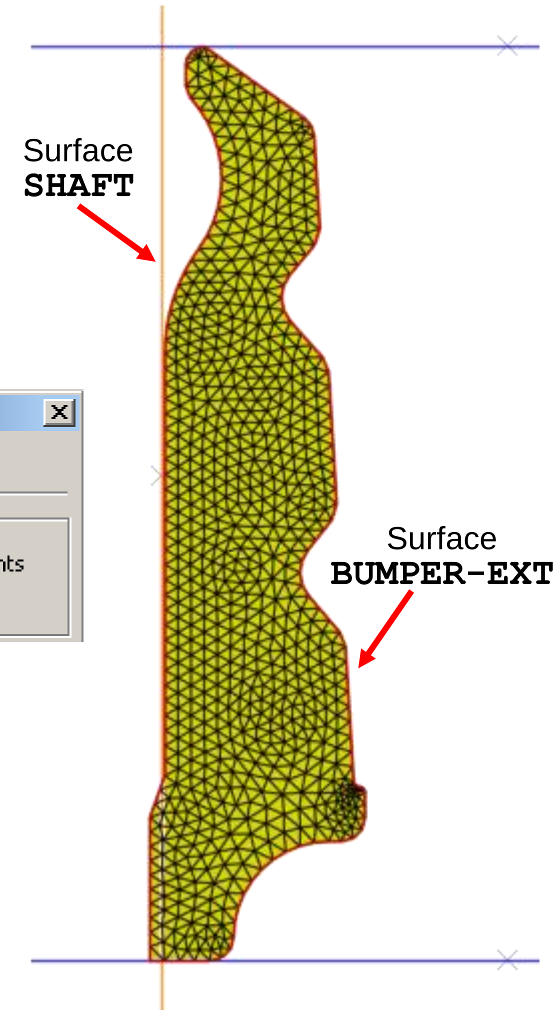
Net effect is to subtract strain-free adjustments



Interference Fit

- **General contact in Abaqus/Standard**
  - General contact algorithm can treat initial overclosures as interference fits
  - Uses a shrink-fit method to resolve the interference gradually over the course of the first analysis step
  - Stresses and strains generated





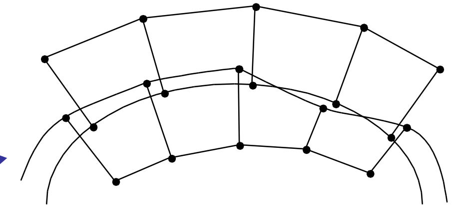

```
*Contact Initialization Data, name=Fit-1, INTERFERENCE FIT
*Contact initialization Assignment
 BUMPER-EXT, SHAFT, Fit-1
```

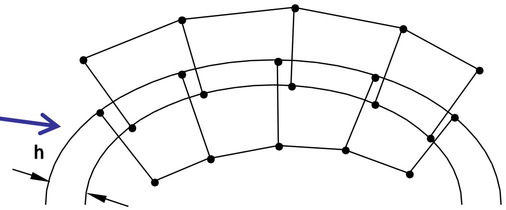
User-Specified Interference and Clearance Distance for General Contact in Abaqus/Standard

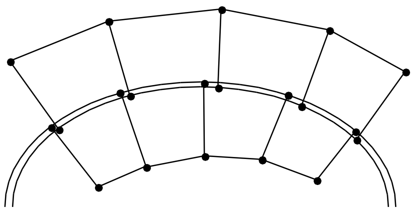
New to Abaqus 6.10

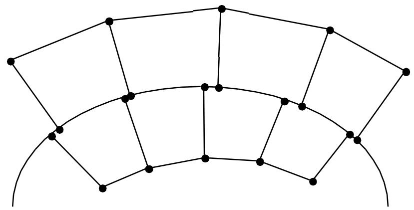
- **High-level description:**
  1. The original mesh need not reflect desired interference or clearance distance
  2. Strain-free adjustments used to achieve user-specified interference/clearance distance
     - Large adjustments may cause element distortion problems
  3. Followed by shrink fit during first step to resolve interference
     - Generating stress and strain
  4. Surfaces that had interference fit will appear compliant at end of first step (aside from penalty penetration)

No equivalent to this for contact pairs









User-Specified Interference and Clearance Distance for General Contact in Abaqus/Standard

New to Abaqus 6.10

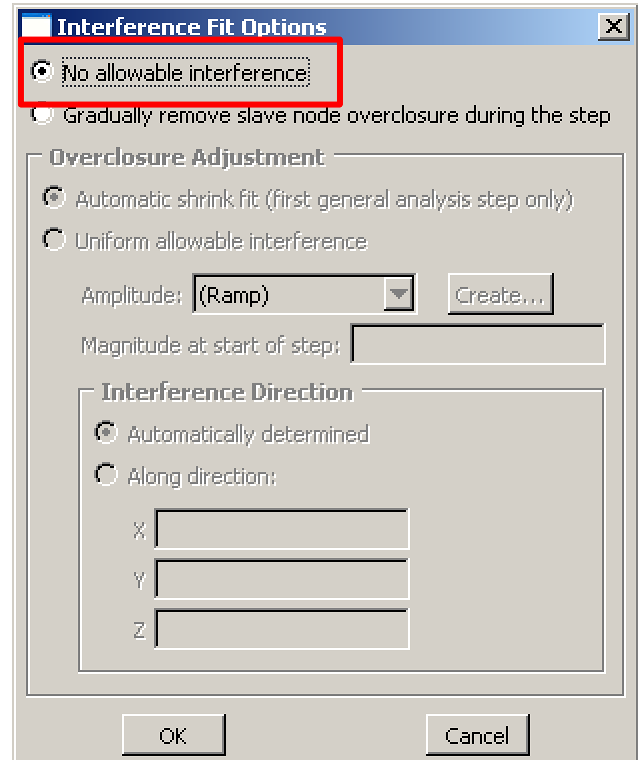
- **Keyword interface**
  - See Section 32.2.4 in Abaqus Analysis User's Manual for details
  - Assign a contact initialization method using the *CONTACT INITIALIZATION ASSIGNMENT option
  - Specify the clearance or interference distance with the *CONTACT INITIALIZATION DATA option
    - Clearance

      *CONTACT INITIALIZATION DATA, INITIAL CLEARANCE=*value*
    - Interference

      *CONTACT INITIALIZATION DATA, INTERFERENCE FIT=*value*
    - In both cases the SEARCH ABOVE and SEARCH BELOW parameters can override the default "capture zone"
- **Not yet supported in Abaqus/CAE**

Interference Fit

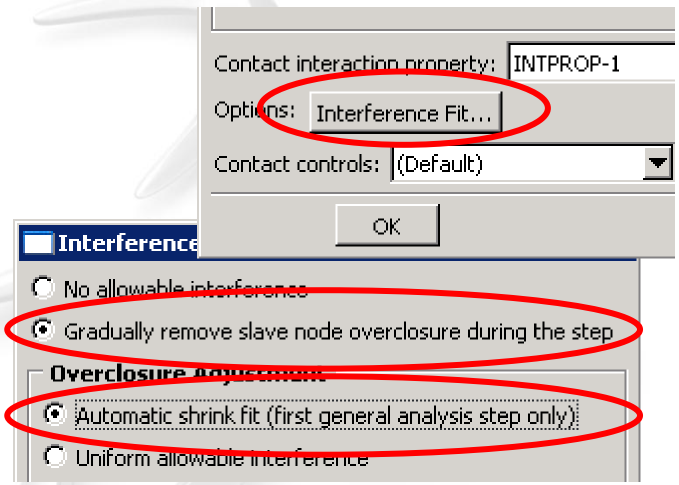
- **By default, Abaqus/Standard contact pairs treat initial overclosures as interference fits to be resolved in the first increment of the analysis**
  - However, with this approach the amount of "interference fit load" applied in this first increment is independent of the increment size relative to the step duration
    - The full interference fit load is applied in the first increment
  - The full interference fit load is sometimes large enough to cause the Newton method to diverge
    - Highly nonlinear response



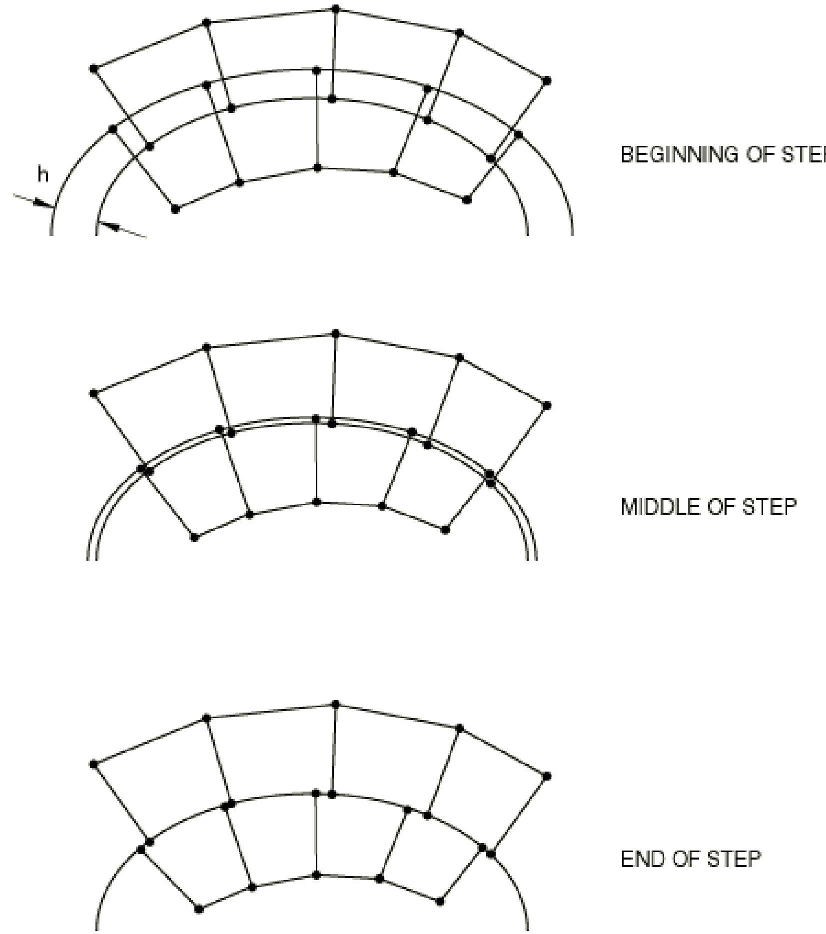
Default behavior: Abaqus/Standard attempts to remove entire interference fit for contact pairs in a single increment

Interference Fit

- **To model interference fits robustly when using contact pairs in Abaqus/Standard**
  - Generally recommended that you specify the shrink fit option such that the interference fit can be resolved over multiple increments within the first step

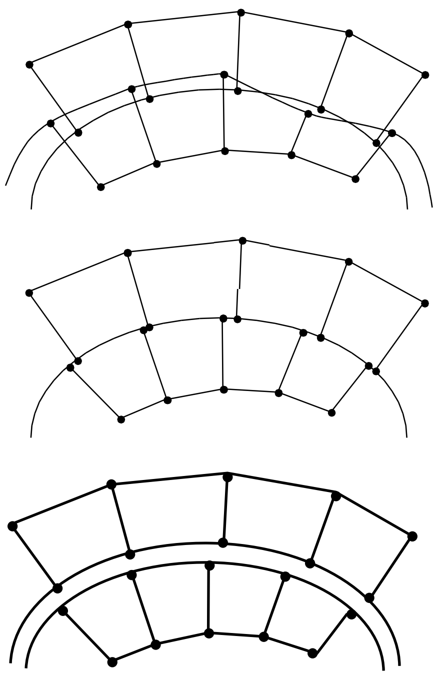



```
*CONTACT INTERFERENCE, SHRINK
 slave, master
```



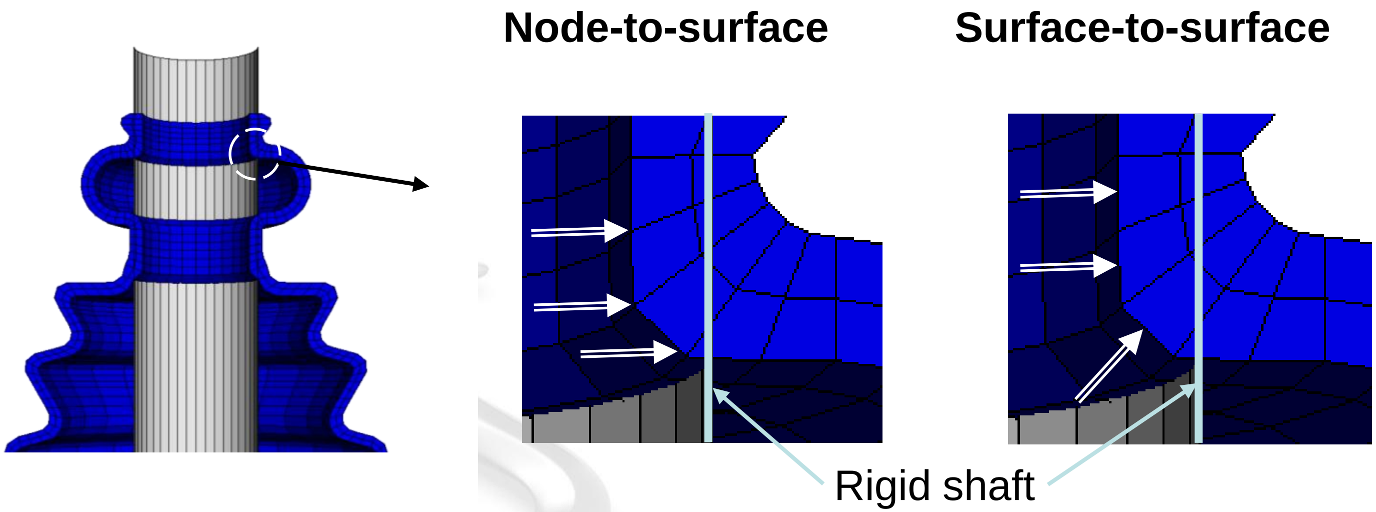
Interference Fit

- **Modeling an interference distance that differs from the initial mesh overclosure with contact pairs**
  - Tricky combination of options
    - Awkward, confusing, and not as accurate compared to new method for general contact
  - Process:
    - Strain-free adjustments to *zero* penetration
      - Using ADJUST parameter
    - Ramp allowed interference from 0.0 to –*h* in the first step
      - *h* is the desired interference fit distance
      - Using contact interference option
      - Will appear as if a gap of distance *h* exists between surfaces at end of first step (even though contact constraints are active)



Interference Fit

- **Interference fits and the surface-to-surface contact discretization**
  - Normal constraints applied along directions of slave surface normals
  - Example: Boot seal contact-interference fit problem



- For node-to-surface interference tends to be resolved along the ***master*** facet normals
- For surface-to-surface interference tends to be resolved along the ***slave*** facet normals; may cause undesirable tangential motions

If penetration is deeper than the element size, you may need to use the node-to-surface formulation

Discouraging Semi-Obsolete Features

(Semi-) obsolete contact features

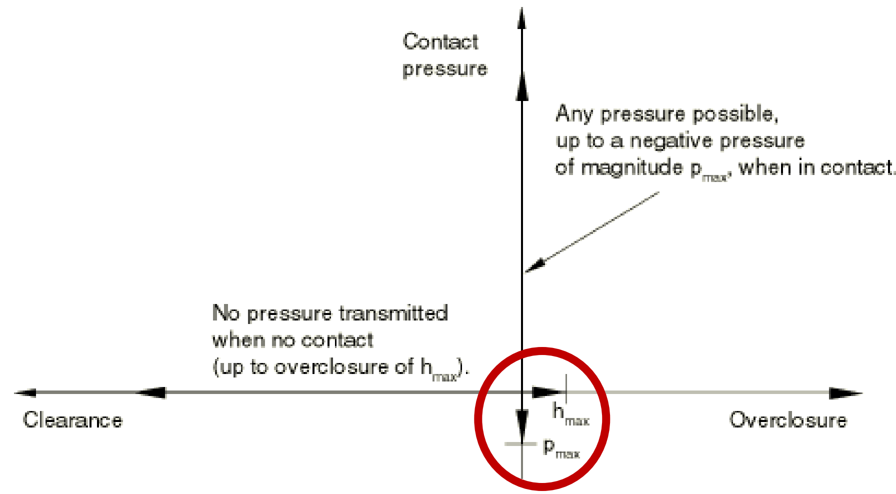
- **Changes in Abaqus 6.11 to discourage use of some features**
  - Objectives:
    - Encourage best modeling practices
    - Simplify Abaqus/CAE interface and primary documentation
    - Facilitate code maintenance and development
    - Mitigate customer frustration over disappearing features
  - Summary of changes:
    - Retire “contact iterations” solution technique
      - Limited effectiveness, difficult to maintain
    - De-emphasize many contact controls
    - Disallow problematic combination of features
      - Node-to-surface, direct enforcement & C3D10 elements underlying slave surface

De-emphasize many contact controls

- **Affected parameters of *Contact Controls option**
  - Approach, Automatic Tolerances, Friction Onset, Lagrange Multiplier, MAXCHP, PERRMX, UERRMX
- **Implications of being de-emphasized**
  - Removed from Abaqus/CAE dialog boxes and input file reader
  - Documentation for them moved to .pdf files accessed through Abaqus/Answer 4605
    - Format of respective sections same as Analysis User's Manual, Keywords Manual, and Verification Manual
  - Trigger warning messages during datacheck
  - Continue to support these features (QC testing, etc.)
- **Release Notes entry 11.8**

Review of de-emphasized contact controls

- **Automatic Tolerances, MAXCHP, PERRMX, UERRMX**
  - All related to avoiding contact chattering
  - Pre-date “Convert SDI”, which has similar intent and is typically superior



- Automatic Tolerances is popular among some users
- Often no longer needed
  - Especially if other nondefault controls are removed
- Sometimes covering up fundamental modeling issues or bugs
- Often helpful to add contact stabilization in normal direction
  - Which is unlikely to affect results

Review of de-emphasized contact controls

- **Approach**
  - Purpose is to stabilize initial rigid body modes
  - Pre-dates the “Stabilize” parameter, which is recommended
    - May need to adjust gap distance over which “Stabilize” acts
    - Recommend setting Tangent Fraction=0.0

- **Friction Onset**
  - Allows user to specify that friction can be neglected for increment in which contact is newly established
  - Non-default Friction Onset=Delayed setting is likely to degrade accuracy

Review of de-emphasized contact controls

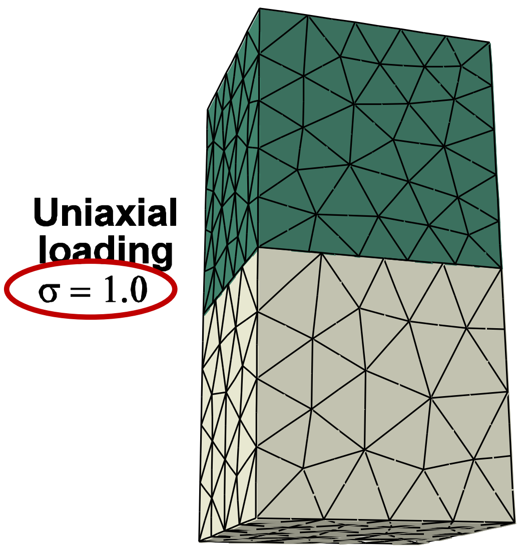
- **Lagrange Multiplier**
  - Controls whether Lagrange multipliers are exposed to the equation solver (in some cases)
  - Default algorithm controlling this choice is robust

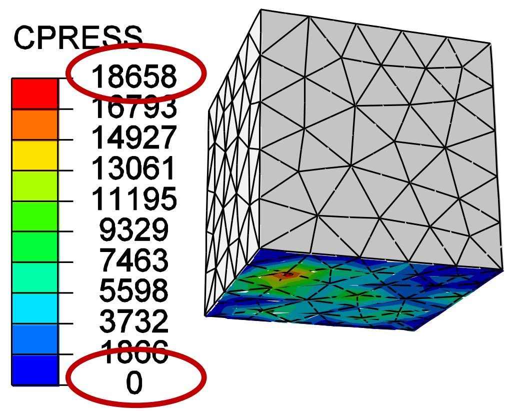
Disallow problematic combination of features

- **Disallowed combination:**
  - Node-to-surface contact formulation
  - Direct enforcement of contact constraints
  - 2^{nd}-order triangular slave faces
- **This combination has historically caused:**
  - Convergence problems
  - Extremely noisy contact stress output
- **Release Notes entry 11.9**

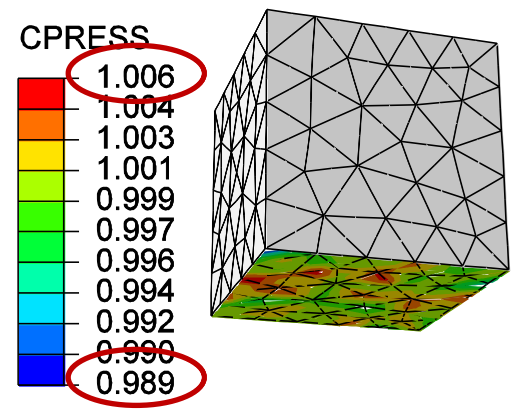
Disallow problematic combination of features

- Uniaxial compression example





C3D10, Node-to-Surface, Strict enforcement



C3D10, Surface-to-Surface, Penalty enforcement

Contact discretization	Element type	Constraint enforcement method	Maximum error in CPRESS
Node-to-surface	C3D10	Direct	4 orders of magnitude
Node-to-surface	C3D10M	Direct	35.4%
Node-to-surface	C3D10	Penalty (default stiffness)	21.2%
Surface-to-surface	C3D10	Direct	1.9%
Surface-to-surface	C3D10	Penalty (default stiffness)	1.1%

Disallow problematic combination of features

- **Unintentionally having this bad combination of features is quite common**

- **Avoiding this combination**
  - Current recommendation
    - Surface-to-surface enforcement and penalty method are generally recommended
    - Somewhat neutral on element type recommendation, but for example C3D10(I) gives a more accurate representation of curved surfaces than C3D10M
  - Years ago
    - We focused on C3D10M as an alternative to C3D10

Lecture 5 Summary

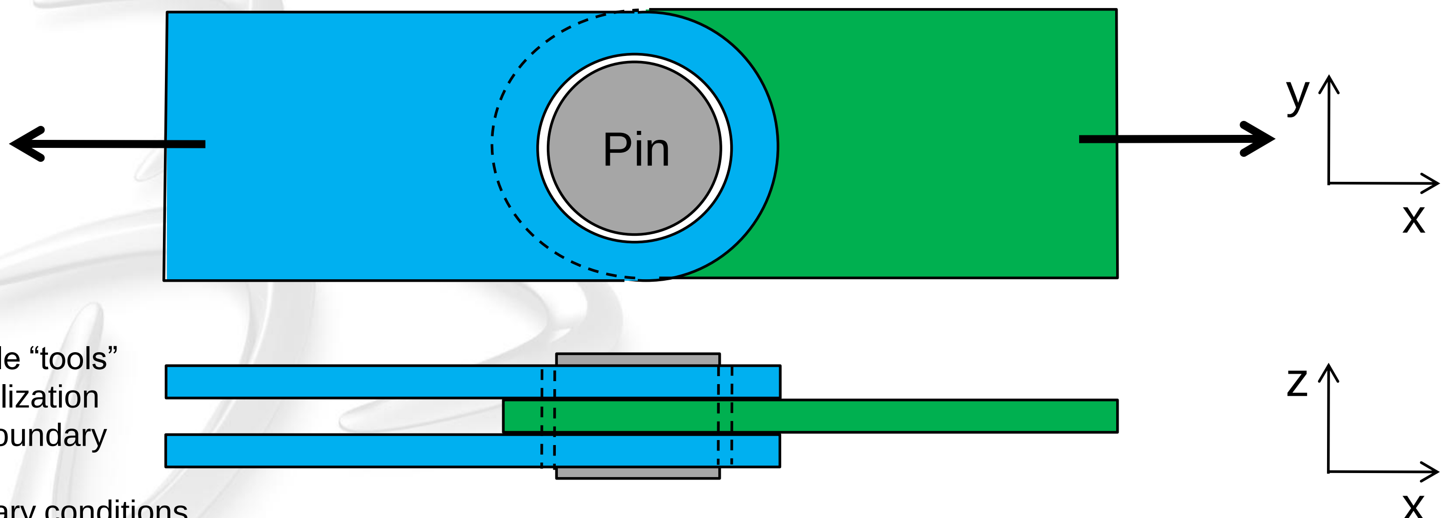
SIMULIA

Review of Topics Discussed in this Lecture

- **Title: Convergence Topics**
- **Static Instabilities**
  - Unconstrained rigid body motion and negative eigenvalues
  - Regularization methods
- **Overconstraints**
- **Best Practices for Treating Initial Overclosures**
- **Discouraging semi-obsolete features**

Discussion (Virtual Workshop)

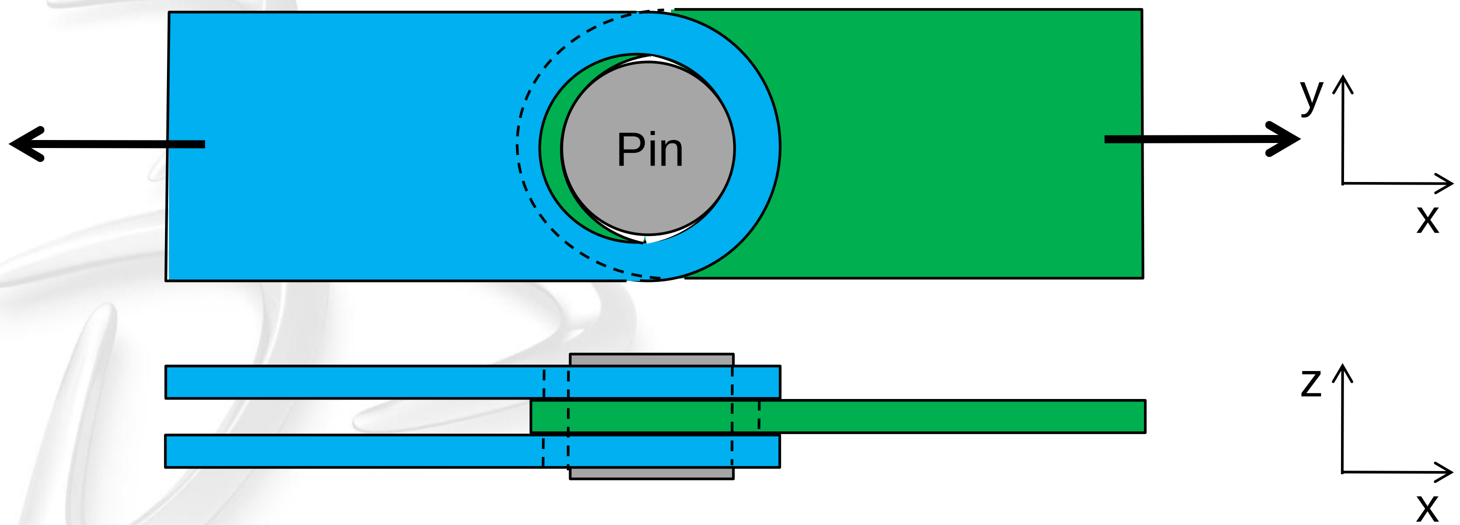
- **Pin connection example**
  - All three components are deformable and modeled with elements
  - Small radial gap around pin initially
  - Discuss how to control rigid body modes of pin



Various possible "tools"
- Contact stabilization
- Symmetric boundary conditions
- Other boundary conditions
- Friction
- Distributing coupling
- …

Discussion (Virtual Workshop)

- **Pin connection example**
  - Shown here with contact established
  - Without friction there may be little or no resistance to rotation of the pin even after contact is established



General Contact in Abaqus/Explicit

Lecture 6

Overview

- **Not providing as comprehensive an overview of Abaqus/Explicit contact as we have for Abaqus/Standard contact in this seminar**
- **Some discussion of Abaqus/Explicit contact in previous "Lectures"**
- **Topics in this lecture include:**
  - Historical perspective on general contact
  - Examples
  - Unique aspects of general contact in Abaqus/Explicit

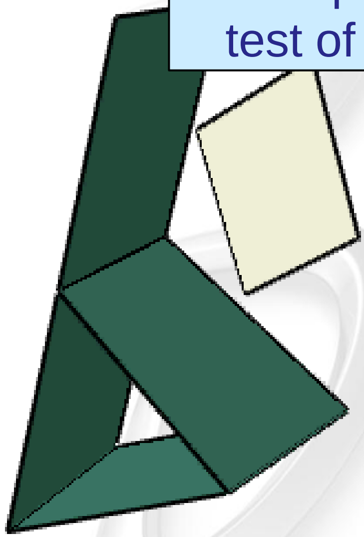
General Contact in Abaqus/Explicit

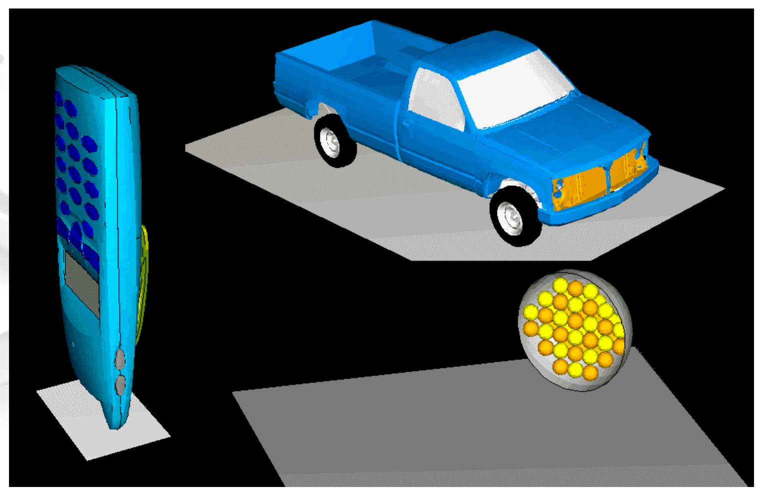
Timeline of initial implementation

- Start of project
- Abaqus 6.3 release

1999 | **2000** | **2001** | **2002**

- Named G.C.
- First car crash test with G.C. prototype
- "AUC"
- First prototype test of G.C.





- Most /Explicit models now use G.C. instead of contact pairs
- GC in Abaqus/Standard released in 2008 (6.8EF)

Contact in Abaqus/Explicit

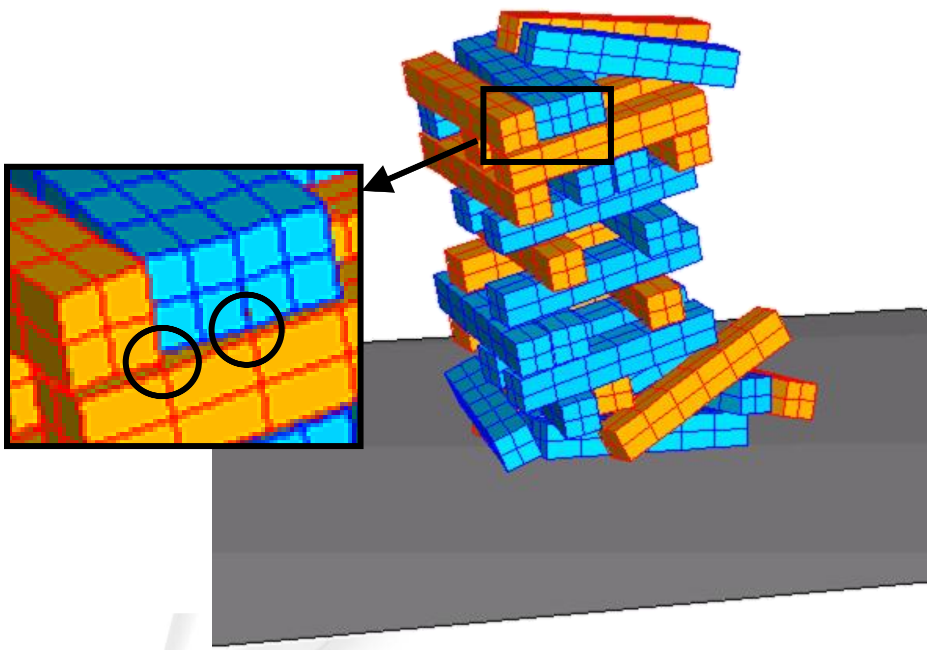
- **Explicit integration method efficiently solves extremely discontinuous events**
  - Possible to solve complicated, very general, three-dimensional contact problems with deformable bodies in Abaqus/Explicit

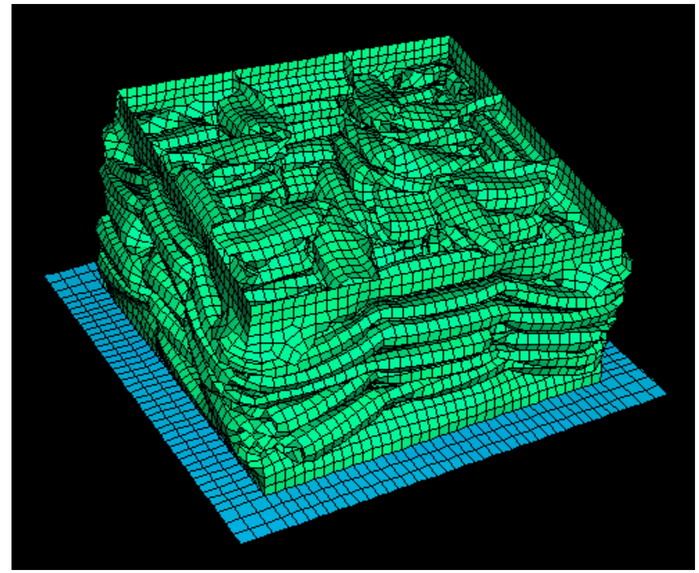
Courtesy of BMW*

* Gholami, T., J. Lescheticky, and R. Paßmann, "Crashworthiness Simulation of Automobiles with Abaqus/Explicit," ABAQUS Users' Conference, Munich, 2003

Examples with multiple contact per node

- **Crushing of aluminum extrusion**
  - Pinched shell layers
- **Falling stack of blocks**
  - Corners

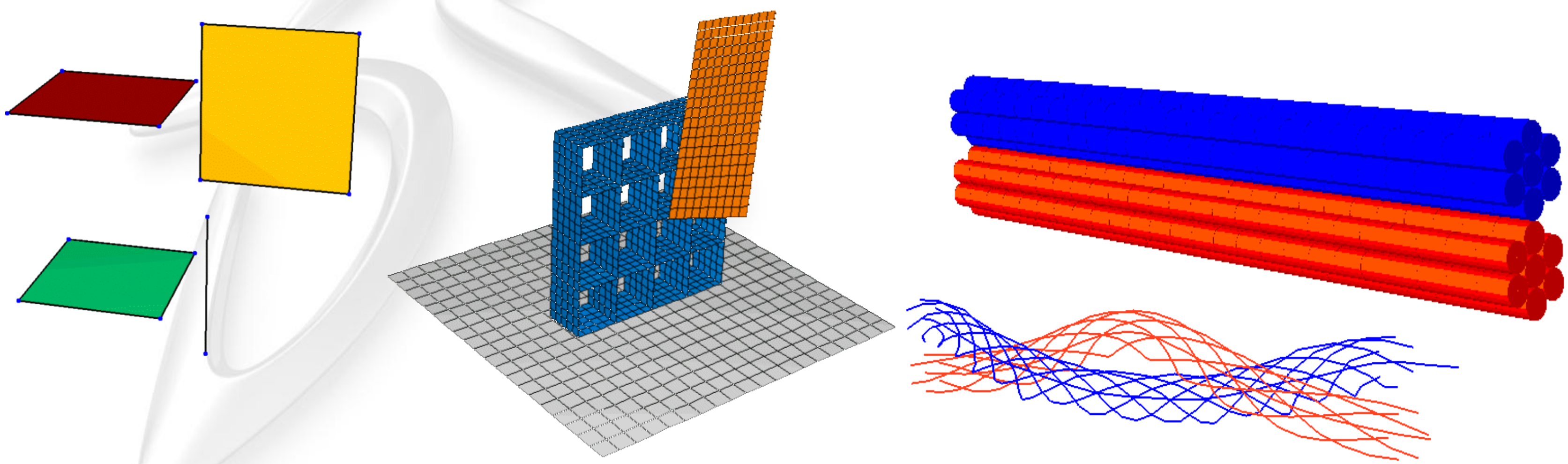




Courtesy of *Alcan Mass Transportation Systems*, Zürich

High-level comparison of G.C. in /Explicit and /Standard

- **Very similar, highly-automated user interfaces**
  - Mostly same keywords and dialog boxes
  - More options are step-dependent in Abaqus/Explicit
- **Underlying contact formulations**
  - /Standard: Surface-to-surface (master-slave) plus edge-to-surface
  - /Explicit: Node-to-surface (balanced) plus edge-to-edge
    - Edge-to-edge examples that /Standard can't yet model:



Repeat slide from Lecture 2

High-level comparison of G.C. in /Explicit and /Standard

- **Examples of differences between general contact in Abaqus/Explicit and Abaqus/Standard**

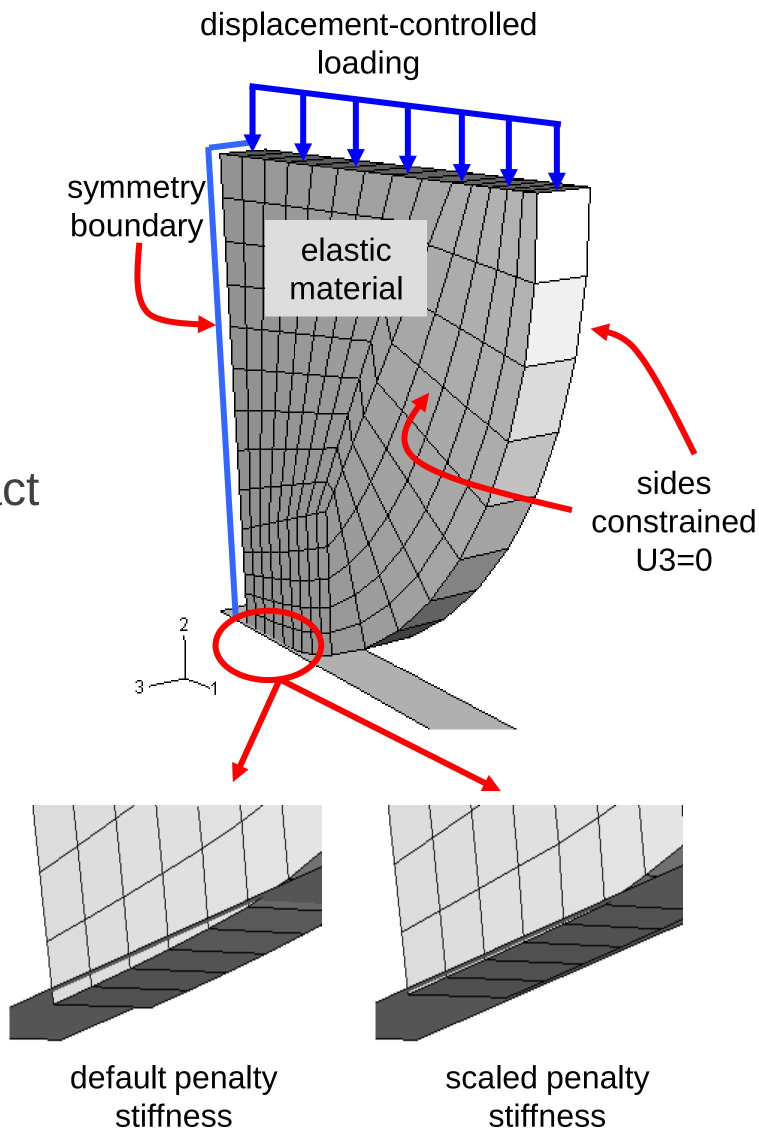
Characteristic	Abaqus/Explicit	Abaqus/Standard
Primary formulation	Node-to-surface	Surface-to-surface
Master-slave roles	Balanced master-slave	Pure master-slave
Secondary formulation	Edge-to-edge	Edge-to-surface
2-D and axisymmetric	Not available	Available
Most aspects of contact definition	Step-dependent	Model data

Contact constraint enforcement

- **Penalty method is used by default for general contact in /Std & /Exp**
  - Only /Std has an alternative penalty enforcement method
    - Lagrange multiplier method
  - Default penalty stiffness is factor of 10 to 100 higher in /Std
    - Increasing penalty stiffness tends to reduce time increment size in /Exp
    - Increasing penalty stiffness tends to degrade convergence behavior in /Std
  - Can scale penalty stiffness in /Std & /Exp
    - Further discussion on next slide

Penalty stiffness

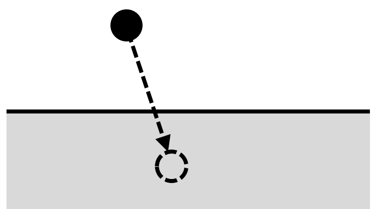
- **For rare cases in which contact penetration becomes significant, penalty stiffness can be increased**
  - Increase could have negative effect on stable time increment
  - Factors that can lead to increased contact penetrations:
    - Displacement-controlled loading
    - Highly confined regions
    - Coarse meshes
    - Purely elastic response



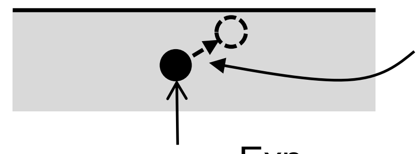
Hertz contact problem: Benchmark 1.1.11

Penalty stiffness

- **Penalty contact forces *react* to penetrations of previous increment in Abaqus/Explicit**



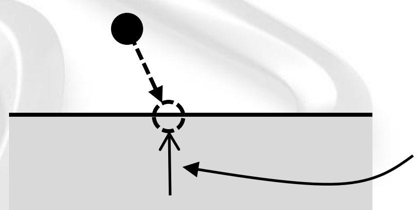
No contact force acting this increment



$$f_{cont}=k_{pen}^{Exp}h$$

Contact normal force acting *throughout* this increment is proportional to penetration at *beginning* of increment

- **Contact is treated "implicitly" in Abaqus/Standard**

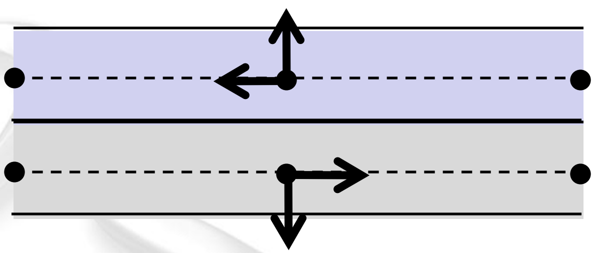


$$f_{cont}=k_{pen}^{Std}h$$

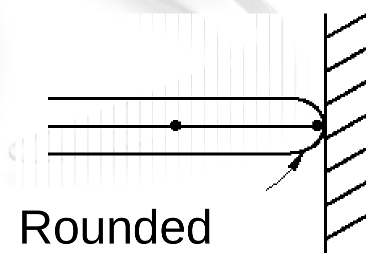
Contact normal force for increment is proportional to penetration at *converged* configuration of increment

Shell thickness and offsets

- **Considered during penetration/gap calculations in /Std & /Exp**
- **Limited thickness-to-facet-dimension ratio for /Exp**
    - Further discussion on next slide
- **/Exp does not account for moment due to friction frictional forces when surface nodes are offset from point of contact**



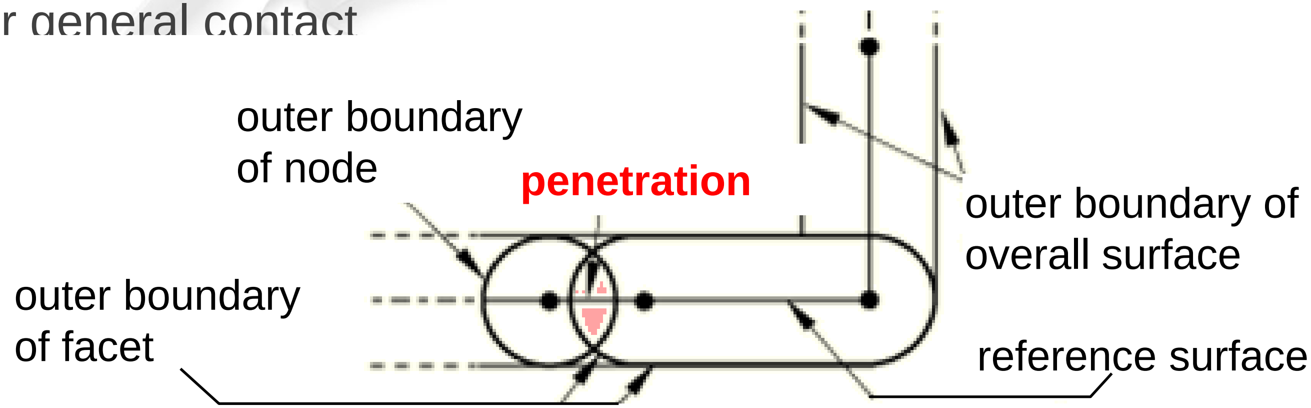
- **No bull-nose at shell perimeters**



Rounded perimeter in /Exp

Shell thickness

- **Surface thickness reductions**
  - Abaqus may automatically reduce contact thickness associated with structural elements to avoid issues of self-intersection
    - If thickness is reduced, a warning is produced in the status file along with element set *WarnElemGContThickReduce*
  - Reducing the contact thickness of a surface may mean that contact occurs later than expected—think of a pinched shell
  - Use output variable CTHICK to contour the actual shell thickness used for general contact

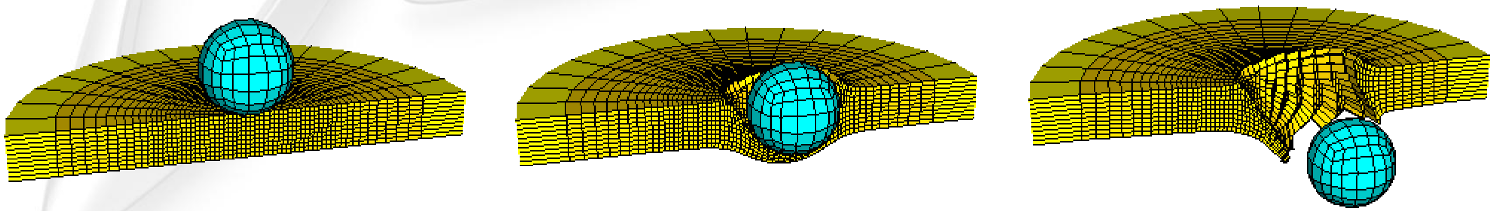


Penetration when the contact thickness exceeds the surface facet edge length

Surface erosion

- **Available in Abaqus/Explicit (but not in /Std)**
- **Both surfaces involved in contact can erode**
  - Abaqus/Examples Manual Section 2.1.4: "Eroding projectile impacting eroding plate"

- **Usage discussed in next slides in context of:**
  - Abaqus/Examples Manual Section 2.1.3: "Rigid projectile impacting eroding plate"



Surface erosion

- **Defining contact inclusions example: Projectile impacting eroding plate**

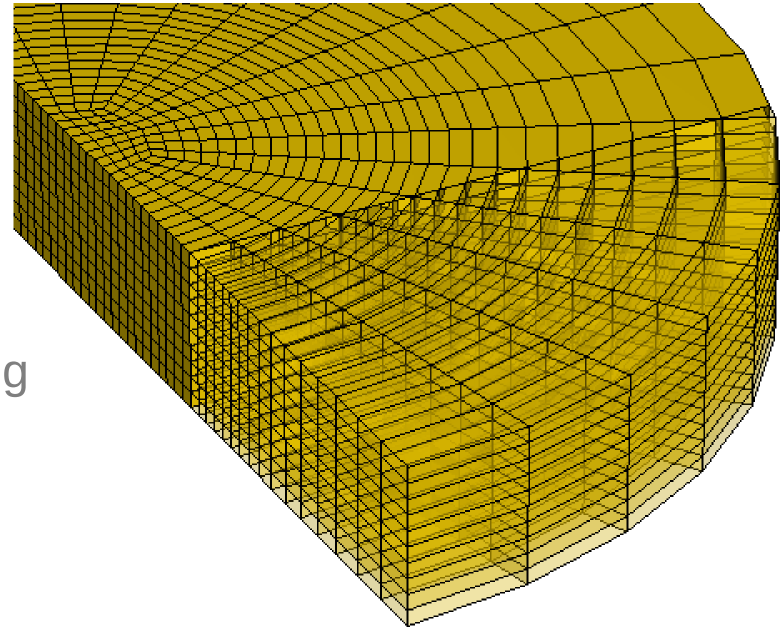
1. Define an element-based surface that includes exterior and interior faces of eroding plate

automatic free surface generation

```
*SURFACE, NAME=ERODE
PLATE,
PLATE, INTERIOR
```

automatic interior surface generation

- Here **PLATE** is an element set containing all plate continuum elements
- Interior surfaces not yet supported in Abaqus/CAE
  - Create model with exterior surface and plate element set
  - Then, modify resulting input file



Surface ERODE

Surface erosion

- **Example (cont'd): Projectile impacting eroding plate**

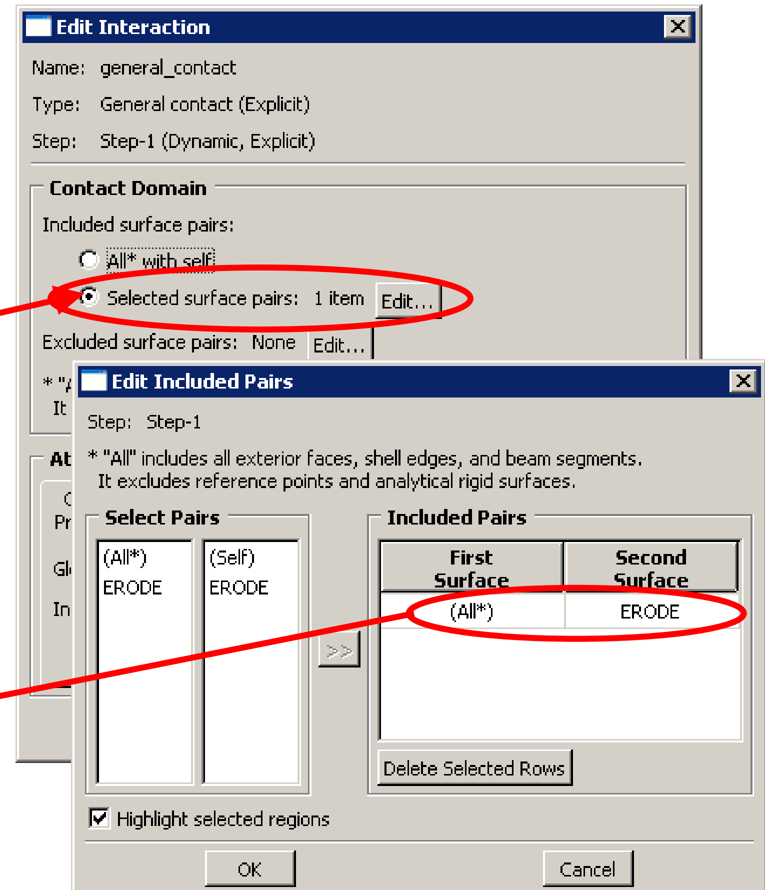
2 Include general contact between projectile and "interior" surface **ERODE**

  - Surface topology will evolve to match exterior of elements that have not failed

```
*CONTACT
*CONTACT INCLUSIONS
  ,ERODE
```

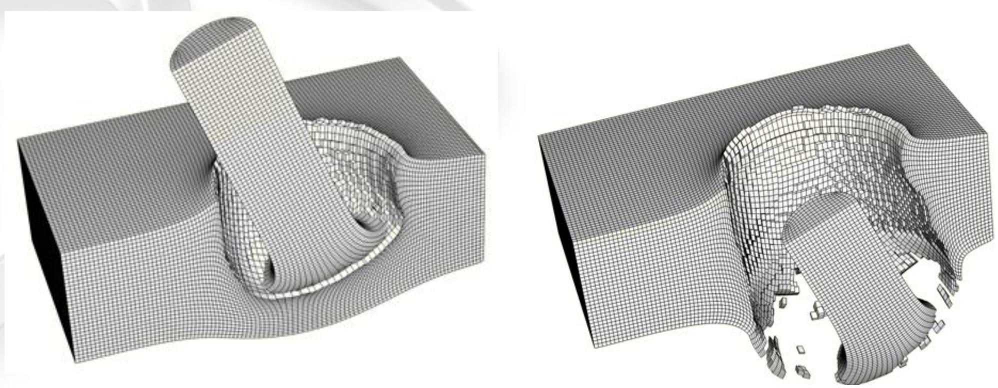
Contact between default all-inclusive element-based surface and **ERODE**

- Self-contact of "interior" surface not included



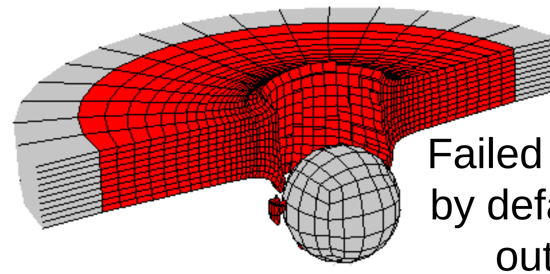
Surface erosion

- **Nodes attached only to eroded elements**
  - By default, treated as point masses that can experience contact with intact facets
    - Some additional momentum transfer
    - Do not interact with other such nodes
  - Alternatively, can specify *CONTACT CONTROLS ASSIGNMENT, NODAL EROSION=YES
    - In this case, excluded from contact
    - See documentation for details

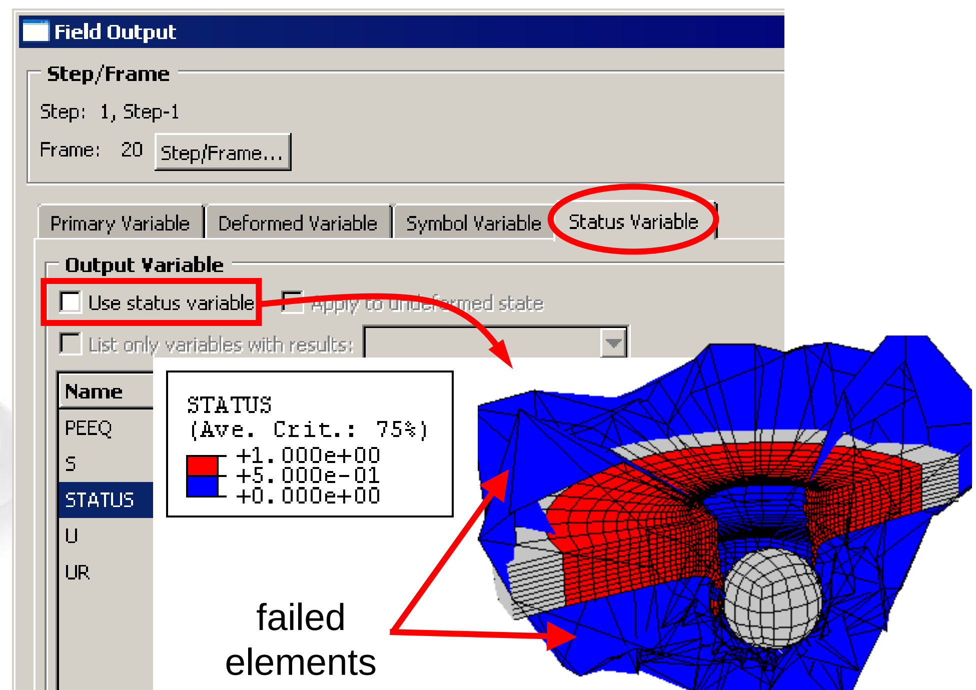


Surface erosion

- **Output variable STATUS indicates whether or not an element has failed**
  - STATUS = 0 for failed elements
  - STATUS = 1 for active elements
- **Abaqus/Viewer will automatically remove failed elements when output database file includes STATUS**



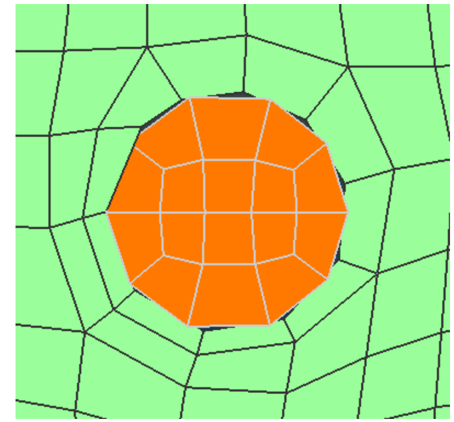
Failed elements removed by default when STATUS output is available



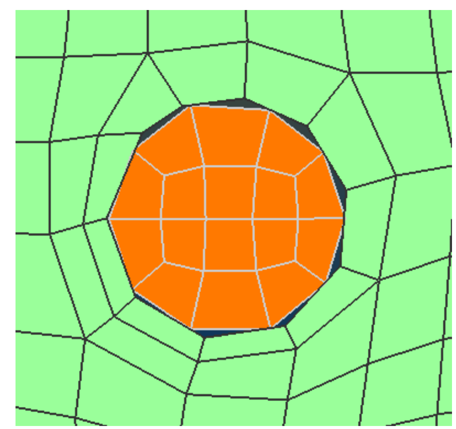
Deactivate status variable to view failed elements

Initial overclosures

- **/Explicit is not well-suited for modeling interference fits**
  - Better to model with /Standard
- **Contact overclosures present in the first step are resolved with strain-free adjustments by default**
  - Adjustments are to nodal *displacements* in /Explicit
- **In subsequent steps, no special action taken to remove initial penetrations for newly introduced contacts**
  - Penalty contact forces applied or penetrations or, in some cases, penetrations may be ignored



Defined mesh with overclosures

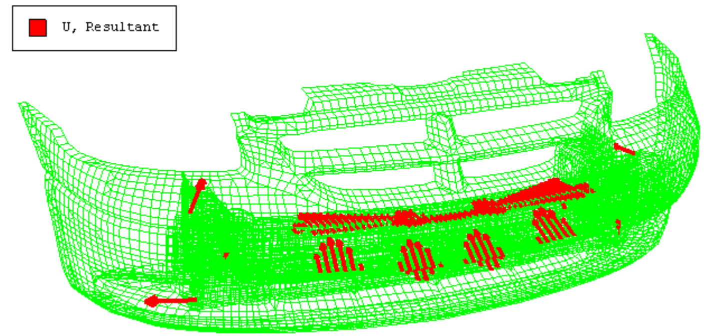


Initial increment with overclosures resolved

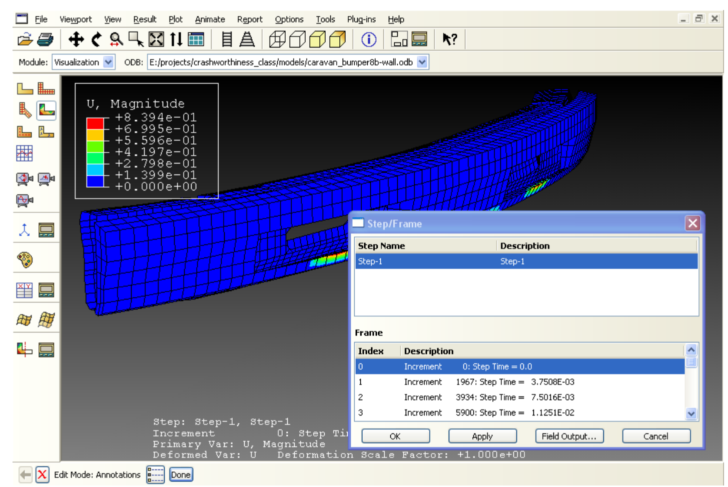
Section of a bolt in a bolt hole

Diagnostics

- **Feedback on resolution of initial overclosures**
  - Symbol (vector) plots of displacements (U) at time=0.0
  - Contour plots of displacements (U) at time=0.0
  - Automatically generated node sets
    - Adjusted nodes: node set *InfoNodeOverclosureAdjust*
    - Nodes with unresolved initial overclosures: node set *InfoNodeUnresolvInitOver*
  - Examine status and message file for additional information



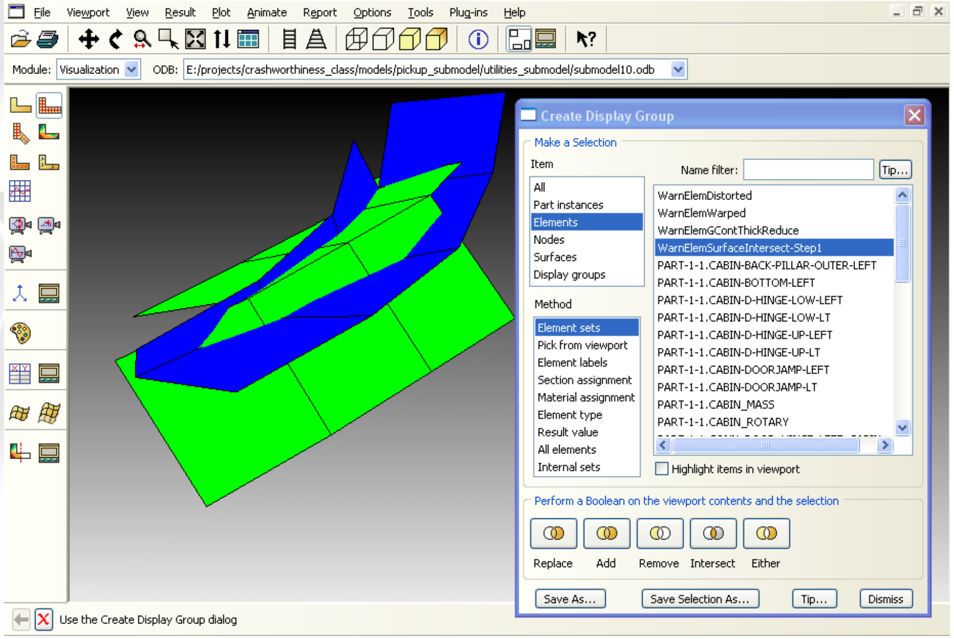
Symbol plot of surface adjustments



Contour plot of surface adjustments

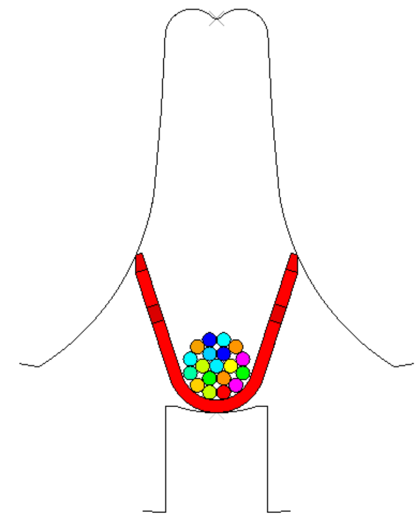
Diagnostics

- **Initially crossed-crossed surfaces generally indicate geometry is wrong**
  - Diagnostic output provided:
    - View element set *WarnElemSurfaceIntersect* using Display Group dialog box
  - Should be manually avoided
    - Otherwise the surfaces will remain “locked” together for duration of analysis

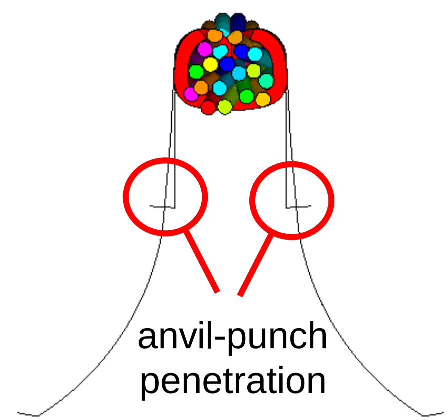


Wire crimping example

- **For choice of model set up, “requires contact exclusions”**
  - Results of wire crimping analysis with default all-inclusive general contact domain shown
  - Comparing results with modeling intent:
    - Goal to capture behavior of deformable bodies (grip and wires)
      - Rigid bodies fully constrained
    - Away from deformable bodies, rigid body geometries are approximated
      - Contact between rigid bodes not intended
      - However, rigid body contact is enforced when it occurs because both rigid bodies are included in default contact domain
        - Resulting model overconstrained



Undeformed shape



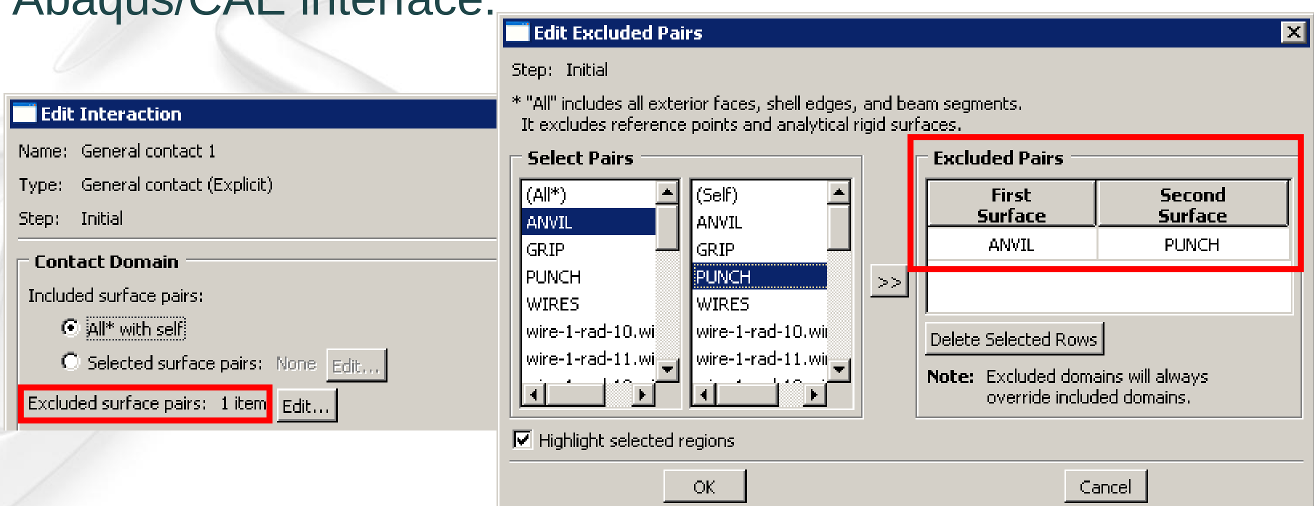
Final deformed shape

Defining General Contact

- **Example (cont'd): Wire crimping**
  - Crimping example with contact excluded between anvil and punch:
    - Keywords interface:

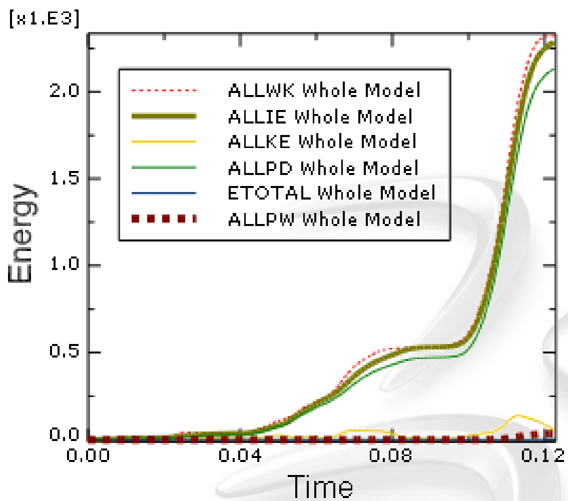
```
*CONTACT
*CONTACT INCLUSIONS, ALL EXTERIOR
*CONTACT EXCLUSIONS
 ANVIL, PUNCH
```

    - Abaqus/CAE interface:

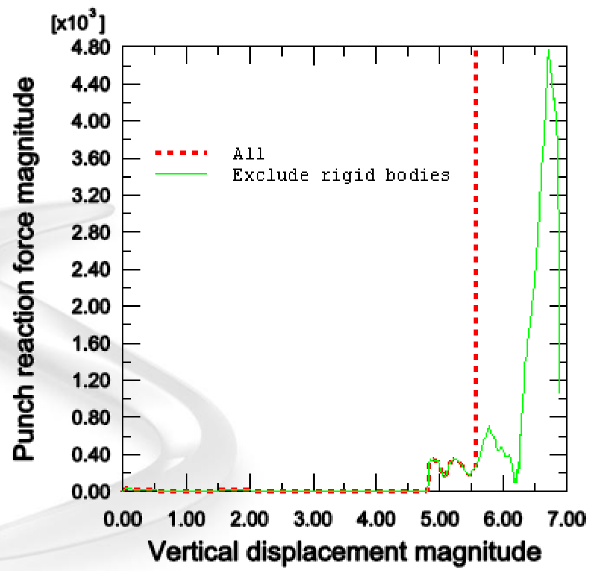


Defining General Contact

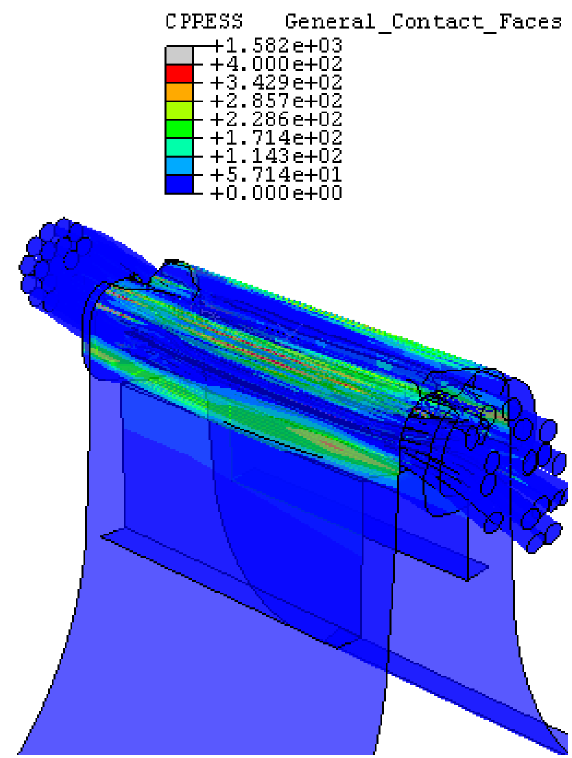
- Valid results produced for wire crimping problem when contact between rigid bodies excluded



Energy history with rigid body contact excluded



Force displacement comparison

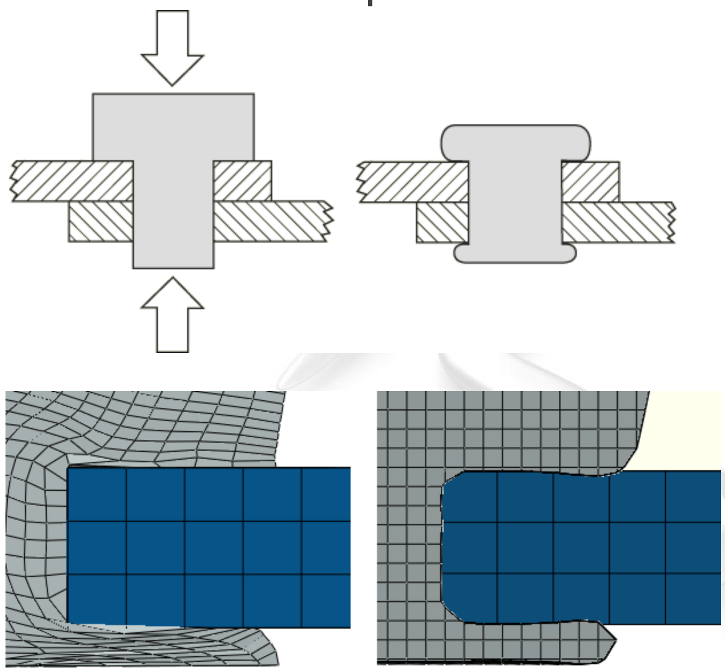


Contact pressure at end of analysis with rigid body contact excluded

SIMULIA

General Contact for Coupled Eulerian-Lagrangian

- Same general contact user interface for CEL
- Not covering CEL in this seminar
- Nice examples:



Section 2.3.1, “Rivet forming,” Abaqus 6.11 Examples Manual

Section 2.3.2, “Impact of a water-filled bottle,” Abaqus 6.11 Examples Manual

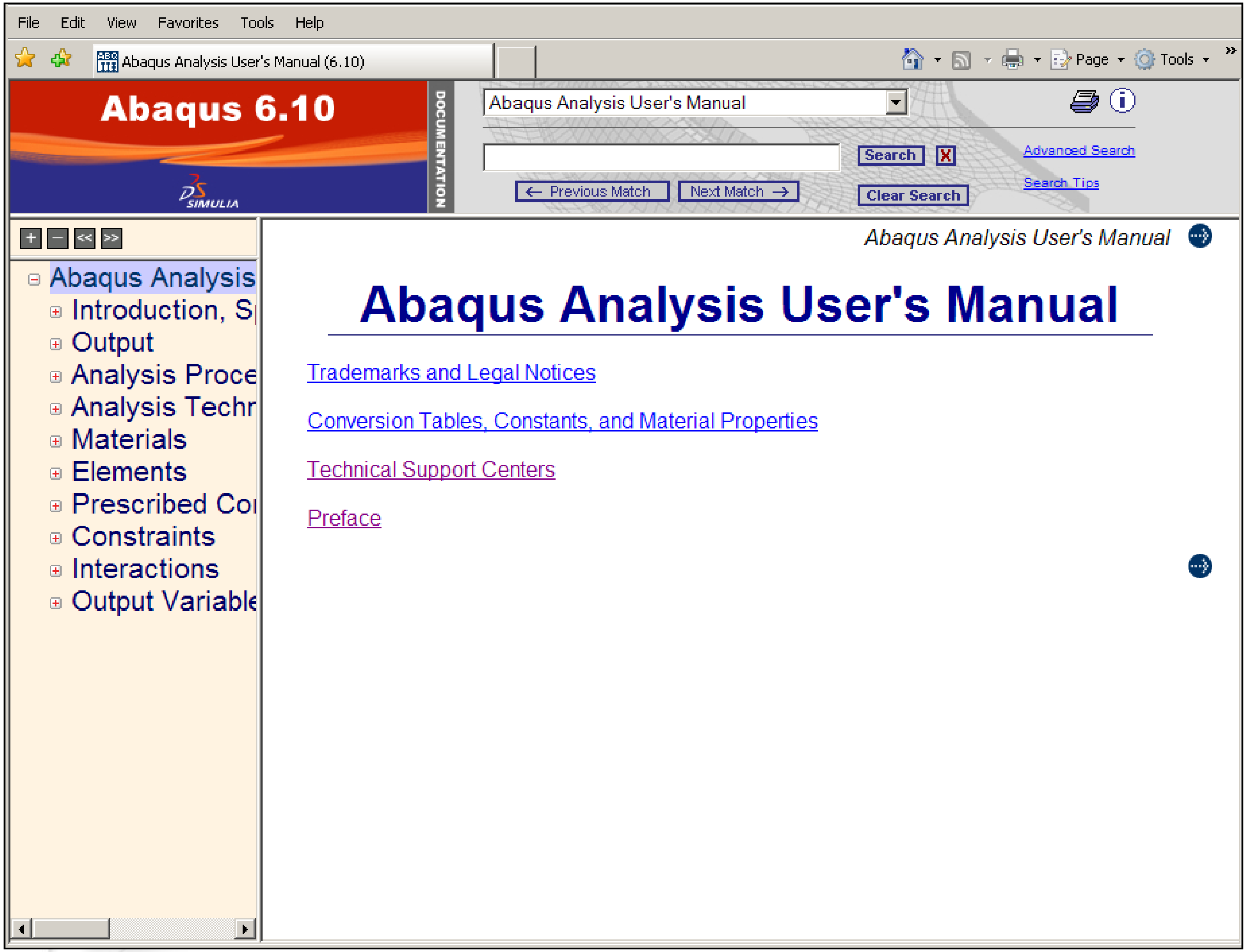
More Features

Lecture 7

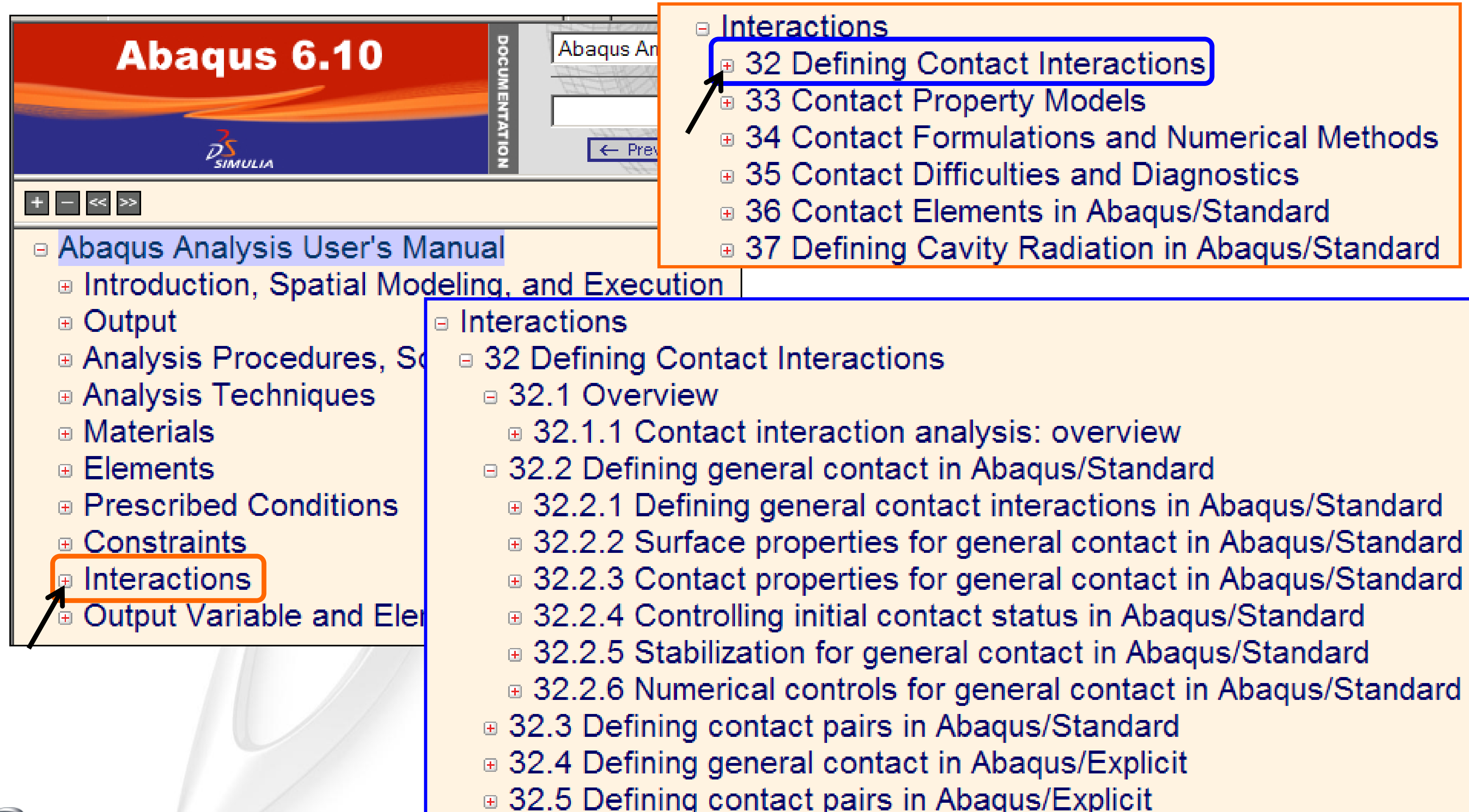
Overview

- **Finding more information about contact features and formulations (once you get back to work)**
  - Abaqus Analysis User's Manual
  - Input files demonstrating features
- **Contact constitutive models**
  - Pressure vs. overclosure
  - Friction
- **Cohesive contact, cracks, and related features**
  - High-level overview
- **Indirectly modeling pressurized fluid working its way between contact surfaces**
  - Pressure-penetration loading
- **Other features related to contact**
  - Rigid bodies, tie constraints, interaction involving other physics

Abaqus Analysis User’s Manual



Abaqus Analysis User's Manual



SIMULIA

Abaqus Analysis User's Manual

Abaqus 6.10

SIMULIA

DOCUMENTATION

Abaqus

- Abaqus Analysis User's Manual
  - Introduction, Spatial Modeling, and Execution
  - Output
  - Analysis Procedures, Solut
  - Analysis Techniques
  - Materials
  - Elements
  - Prescribed Conditions
  - Constraints
  - Interactions
  - Output Variable and Eleme

- Interactions
  - 32 Defining Contact Interactions
  - 33 Contact Property Models
  - 34 Contact Formulations and Numerical Methods
  - 35 Contact Difficulties and Diagnostics
  - 36 Contact Elements in Abaqus/Standard
  - 37 Defining Cavity Radiation in Abaqus/Standard

- 33 Contact Property Models
  - 33.1 Mechanical contact properties
    - 33.1.1 Mechanical contact properties: overview
    - 33.1.2 Contact pressure-overclosure relationships
    - 33.1.3 Contact damping
    - 33.1.4 Contact blockage
    - 33.1.5 Frictional behavior
    - 33.1.6 User-defined interfacial constitutive behavior
    - 33.1.7 Pressure penetration loading
    - 33.1.8 Interaction of debonded surfaces
    - 33.1.9 Breakable bonds
    - 33.1.10 Surface-based cohesive behavior
  - 33.2 Thermal contact properties
  - 33.3 Electrical contact properties
  - 33.4 Pore fluid contact properties

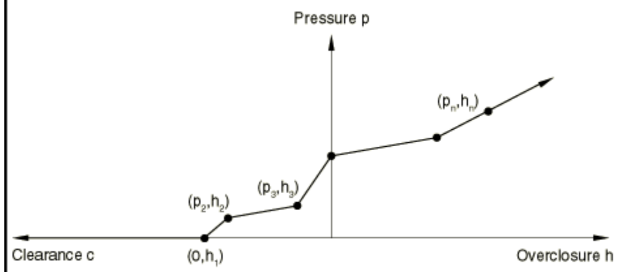
SIMULIA

Abaqus Analysis User's Manual

"Softened" contact defined in tabular form

To define a piecewise-linear pressure-overclosure relationship in tabular form, as shown in Figure 33.1.2–3, you specify data pairs (p_i, h_i) of pressure versus overclosure (where overclosure corresponds to negative clearance). You must specify the data as an increasing function of pressure and overclosure. In this relationship the surfaces transmit contact pressure when the overclosure between them, measured in the contact (normal) direction, is greater than h_1, where h_1 is the overclosure at zero pressure. For the general contact algorithm in Abaqus/Explicit h_1 must be zero. For overclosures greater than h_n the pressure-overclosure relationship is extrapolated based on the last slope computed from the user-specified data (see Figure 33.1.2–3).

Figure 33.1.2–3 "Softened" pressure-overclosure relationship defined in tabular form.



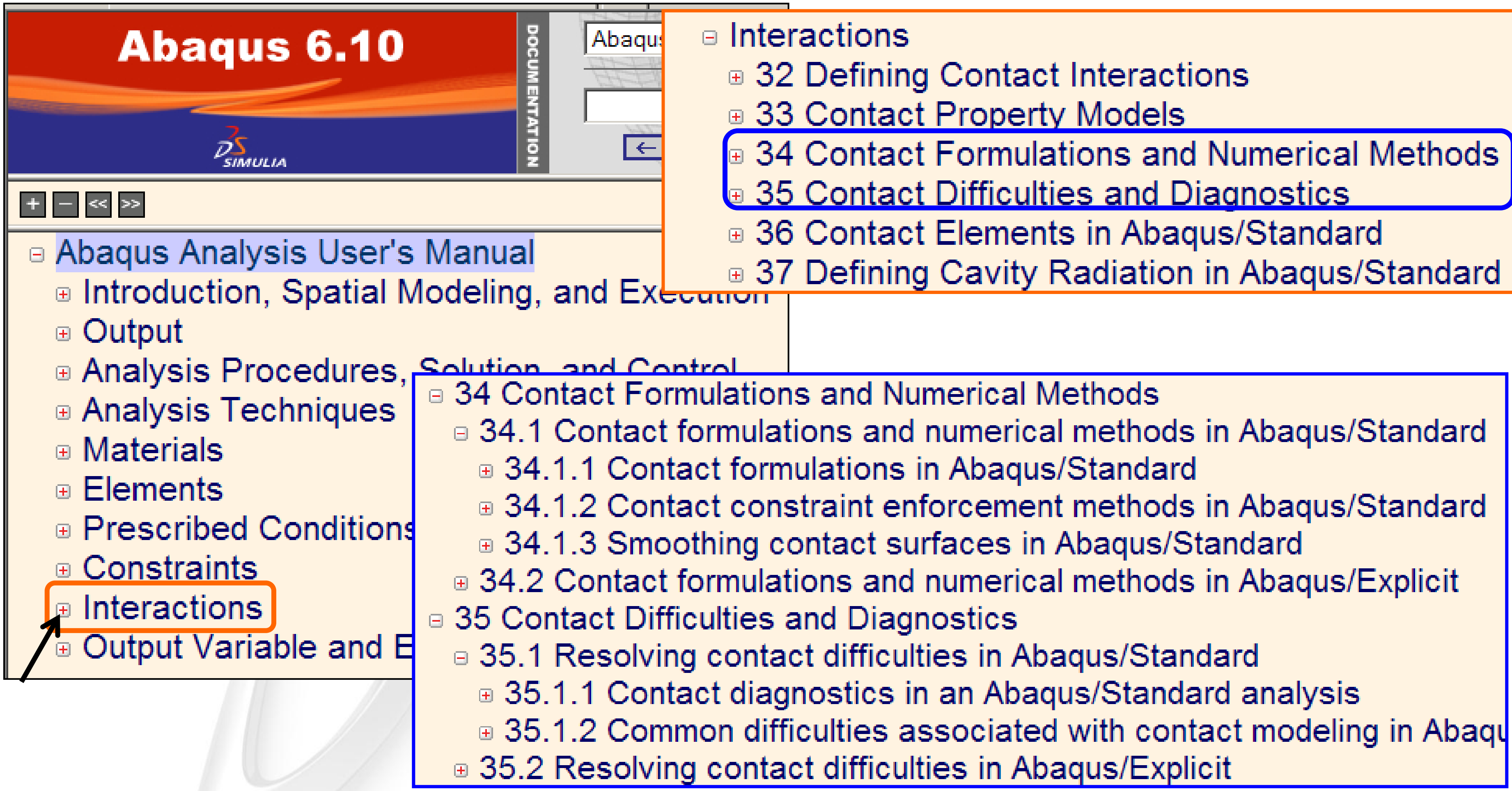
Input File Usage: *SURFACE BEHAVIOR, PRESSURE-OVERCLOSURE=TABULAR

Abaqus/CAE Usage: Interaction module: contact property editor: **Mechanical→Normal Behavior**: **Constraint enforcement method: Default**: **Pressure-Overclosure: Tabular**

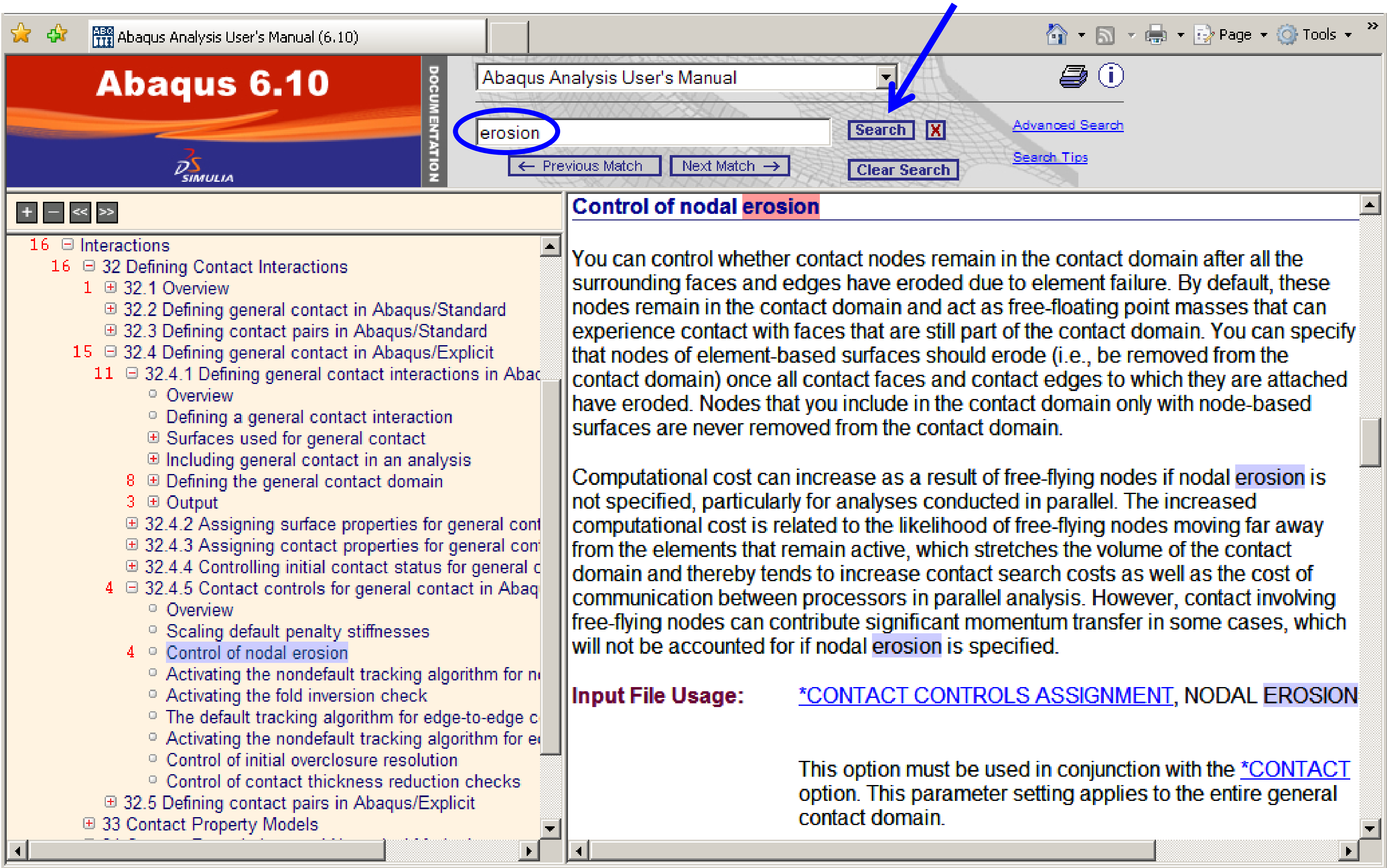
To find input files demonstrating features:

```
d:\Users\hhf>abq findkey
*surface behavior, pressure-overclosure=tabular
*static
*
```

Abaqus Analysis User’s Manual



Abaqus Analysis User's Manual



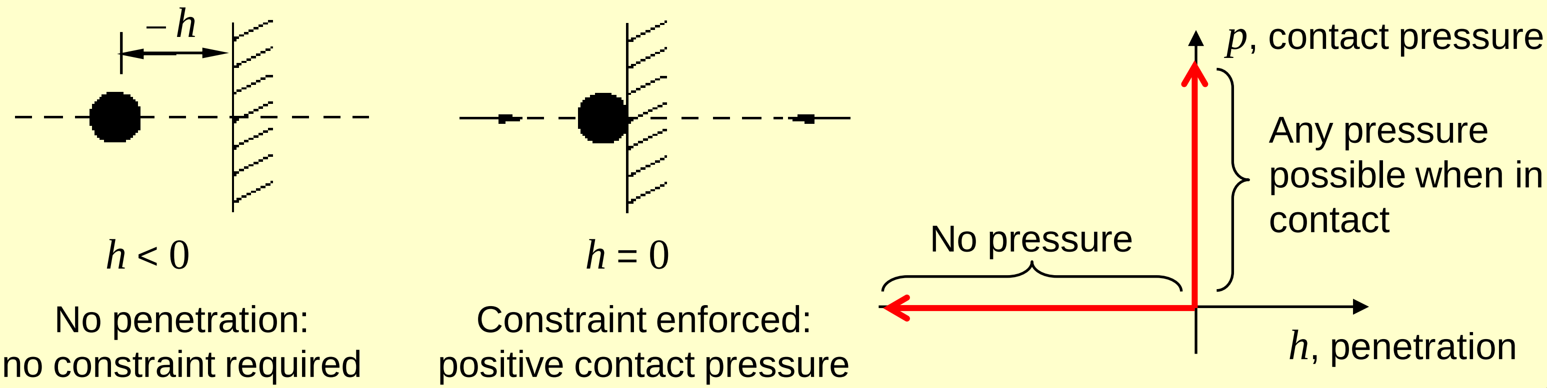
SIMULIA

Pressure-Overclosure Models

Pressure-Overclosure Models

- **Default *physical* pressure-overclosure model is "hard" contact**
  - Although the idealized "hard" model is not always strictly enforced in the numerical solution due to:
    - Softening in the numerical *constraint method*
      - Example: Penalty method (finite rather than ∞ constraint stiffness)
    - Convergence tolerances for Newton iterations
      - Example: Accept as converged despite very small negative contact pressure

Idealized "hard" pressure vs. penetration behavior



Pressure-Overclosure Models

- **Abaqus provides alternative physical pressure-overclosure models**
  - Softened contact
    - Exponential
    - Linear
    - Tabular
  - Contact without separation

Motivation for usage may be:
- Physically based: surface coatings
- Numerically based: improve converge
  - These models were available prior to penalty enforcement

- **Other features influencing overall contact constitutive behavior**
  - Breakable bonds, surface-based cohesive behavior, and crack propagation along a contact interface
    - Also influences tangential behavior
  - User-defined behavior with user subroutine **`UINTER`**
    - Also controls tangential behavior
    - Not discussed here

Exponential Pressure-Overclosure

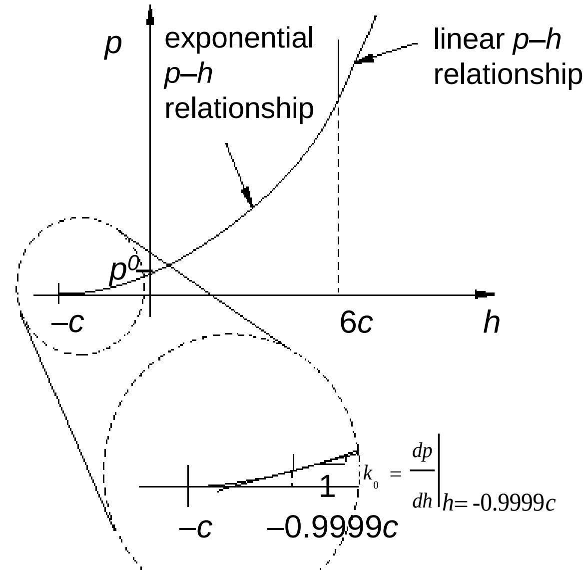
- **Contact pressure increases exponentially for penetrations in range –c to 6c**

$$p = \frac{p^o}{e-1}\frac{c+h}{c}\left(e^{\frac{c+h}{c}} - 1\right) \text{ for } -c < h \leq 6c.$$

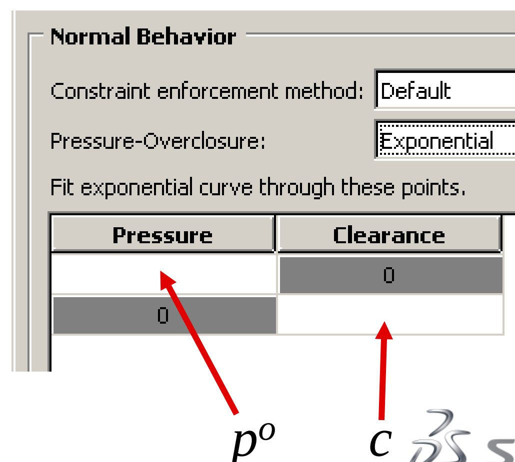
  - **Surfaces come into contact when gap distance is still slightly positive**
  - **Positive contact pressure (and contact stiffness) when surfaces are just touching**
  - Special treatment very close to $h=c$ to avoid numerical issues with very small stiffness
  - Pressure-overclosure relationship is linear for larger penetrations (to avoid numerical issues with very large stiffness)

- **Both c and p^o must be positive**

```
*SURFACE INTERACTION
*SURFACE BEHAVIOR,
PRESSURE-OVERCLOSURE=exponential
c, p^o
```

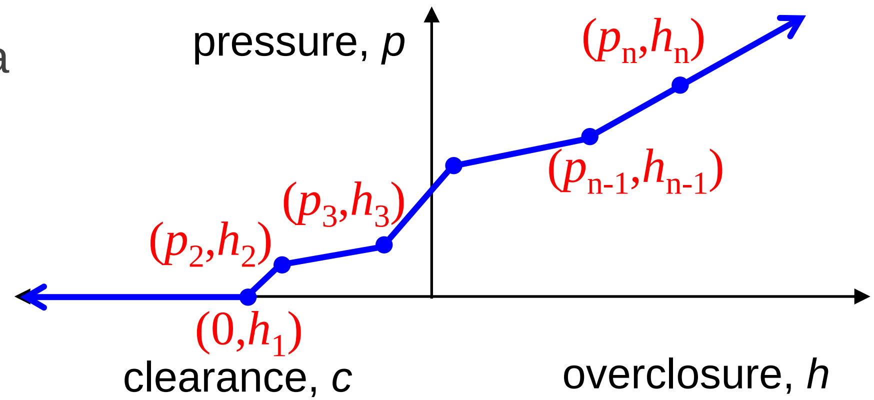


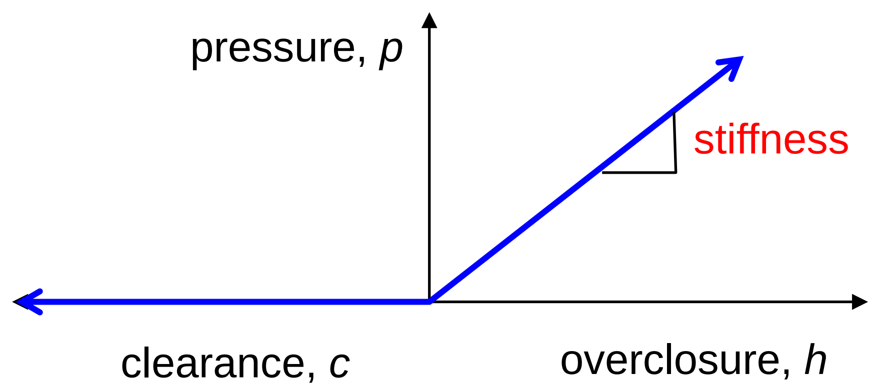
Sometimes used as a "trick" to avoid unconstrained-rigid-body-mode issues



Tabular and Linear Pressure-Overclosure

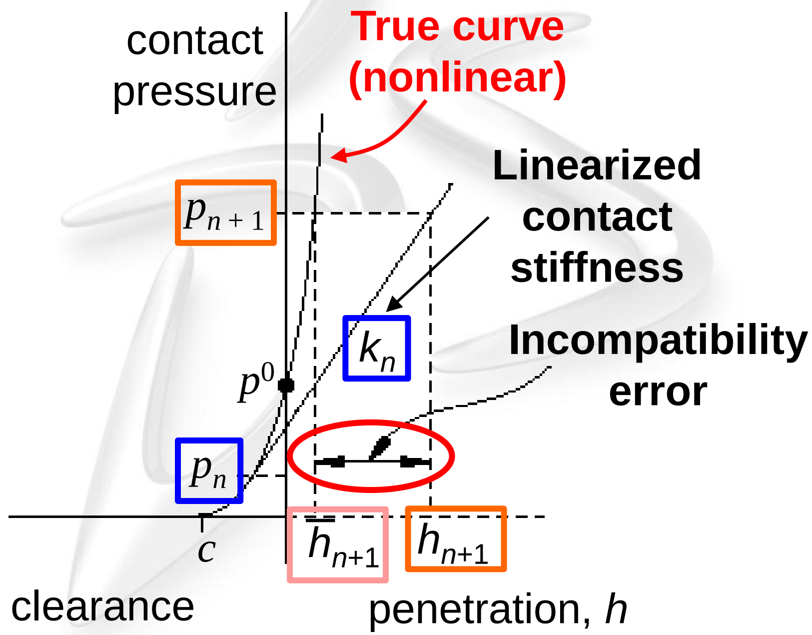
- **Tabular**
  - Input data points (p_i, h_i) to define a piecewise linear relationship between pressure and overclosure
    - First data point is $(0, h_1)$
    - Zero slope before first data point
    - Monotonic increase in successive data points: $h_{i+1}>h_i$, $p_{i+1}>p_i$
    - Constant slope after second-to-last data point
- **Linear**
  - Input single contact stiffness value
  - Similar to penalty method





Softened Contact Nonlinearity

- **Numerical treatment**
  - Linearized contact stiffness used for each Newton iteration
  - Tolerance enforced on deviation from true pressure vs. overclosure curve in convergence check
  - Except in cases in which the slope of the pressure vs. overclosure curve is very large, this contact stiffness is enforced without exposing Lagrange multipliers to equation solver

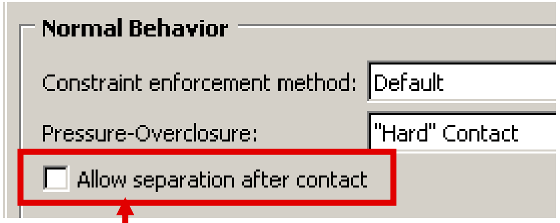


1. For current p_n, find k_n
2. Solve system of eqns., resulting in p_{n+1}, h_{n+1}
3. For current p_{n+1}, find k_{n+1} and $\bar{h}_{n+1}$ assoc. with pressure vs. overclosure curve
4. Magnitude of $h_{n+1} - \bar{h}_{n+1}$ considered in convergence check
5. Continue iterations, as necessary (new linearization)

Contact-Without-Separation Model

- **Useful for modeling adhesives**
- **This feature causes surfaces to be bonded for duration of analysis once contact is established**
  - Only normal contact is affected—relative sliding still allowed
  - Often used with the rough friction option (no sliding either)
- **Usage sometimes numerically motivated (improve convergence)**
- **Syntax:**

```
*SURFACE INTERACTION
*SURFACE BEHAVIOR, NO SEPARATION
```



Toggled **off** to invoke NO SEPARATION

Friction Models

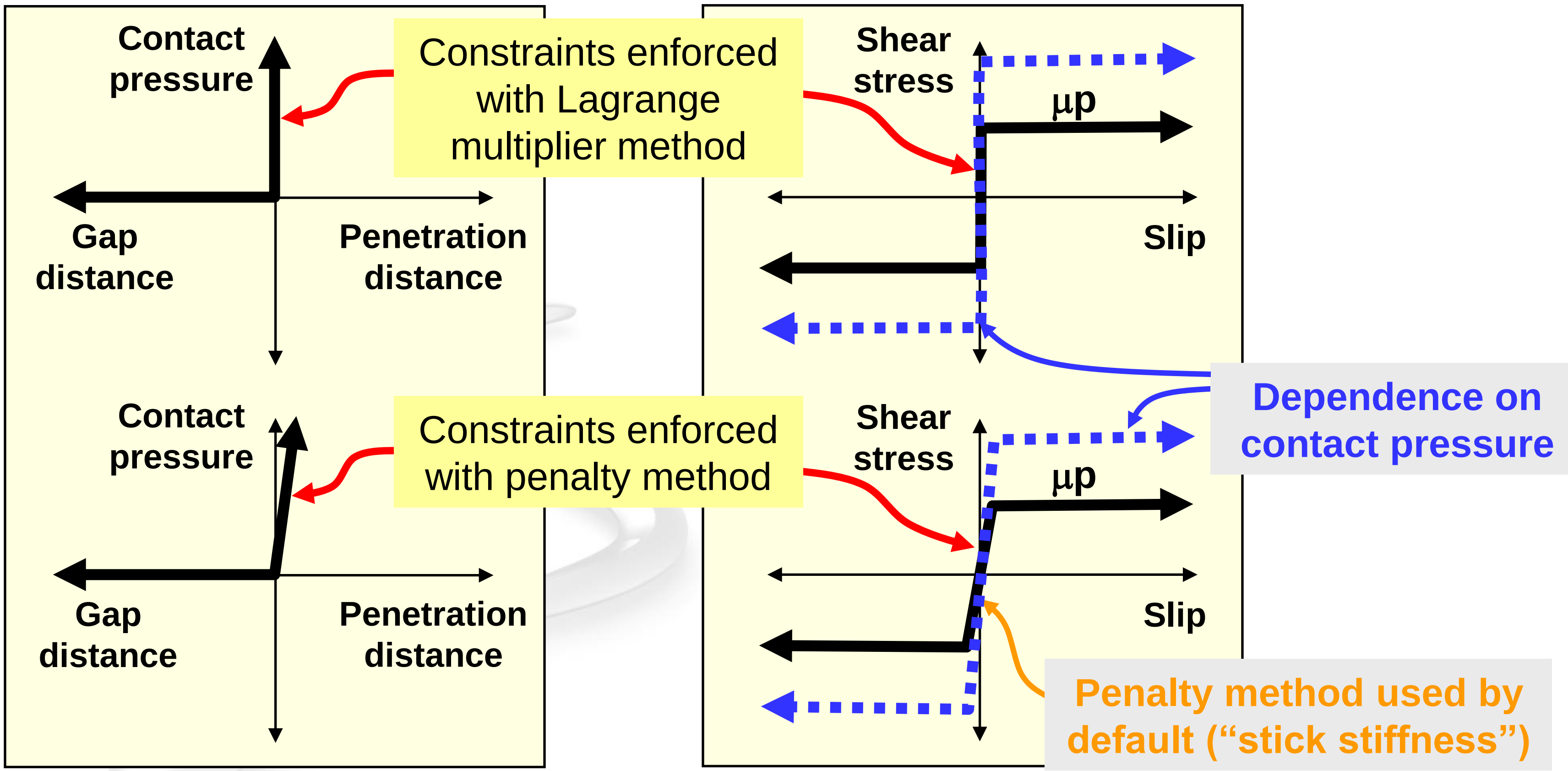
SIMULIA

Friction

- **Available friction models in Abaqus:**
  - **Coulomb** friction
    - Isotropic or anisotropic
    - Optional friction coefficient dependence on slip rate, pressure, temperature, and field variables
      - Linear interpolation of tabular data
      - Exponential dependence on slip rate
      - User subroutine **`FRIC_COEF`**
    - Optional upper bound on shear stress
  - **“Rough”** friction
    - Sticking regardless of contact pressure as long as normal contact constraint is active
  - **User-defined** (through user subroutine **`FRIC`** or **`UINTER`**)

Friction

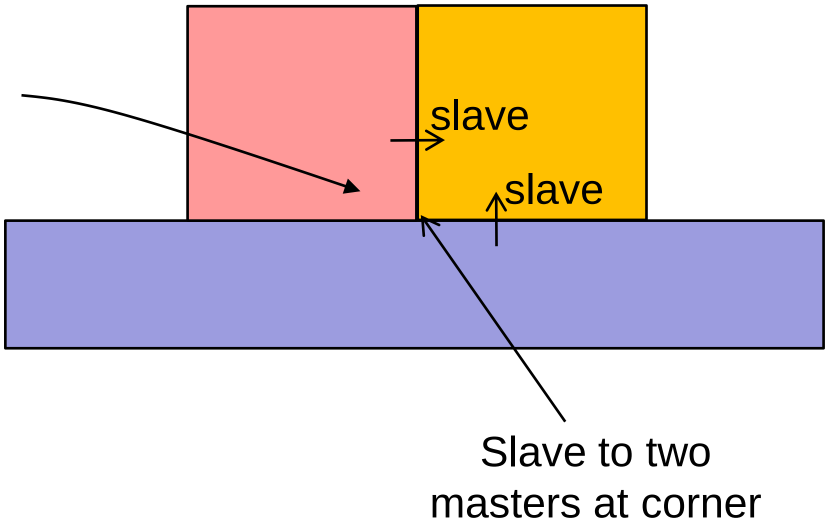
- **Stick/slip discontinuity for friction is similar to open/closed discontinuity in normal direction**

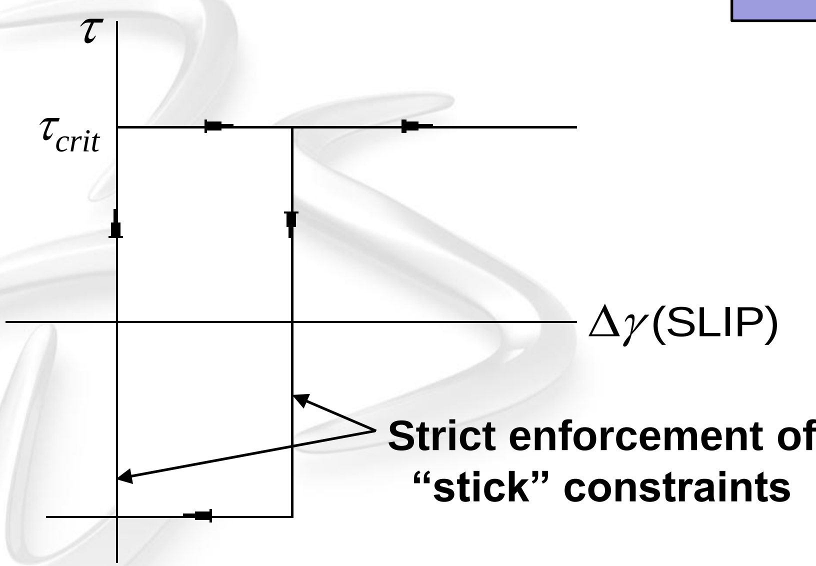


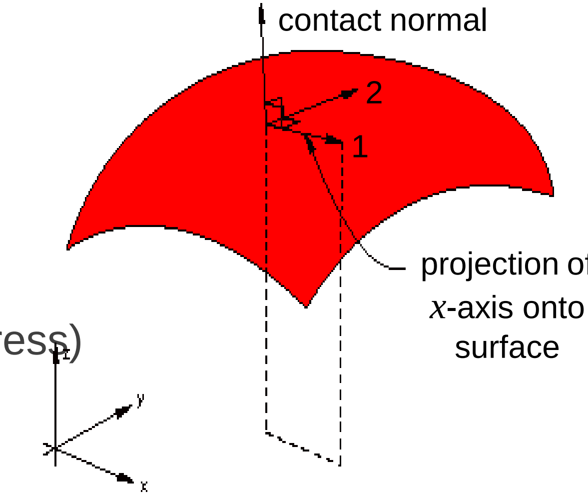
Normal direction behavior **Tangential behavior**

‘Stick’ Constraint Enforcement

- **Lagrange multiplier method can cause overconstraint problems**
  - Such as at junctions like shown here
  - Overlapping, strict constraints cause problems for equation solver

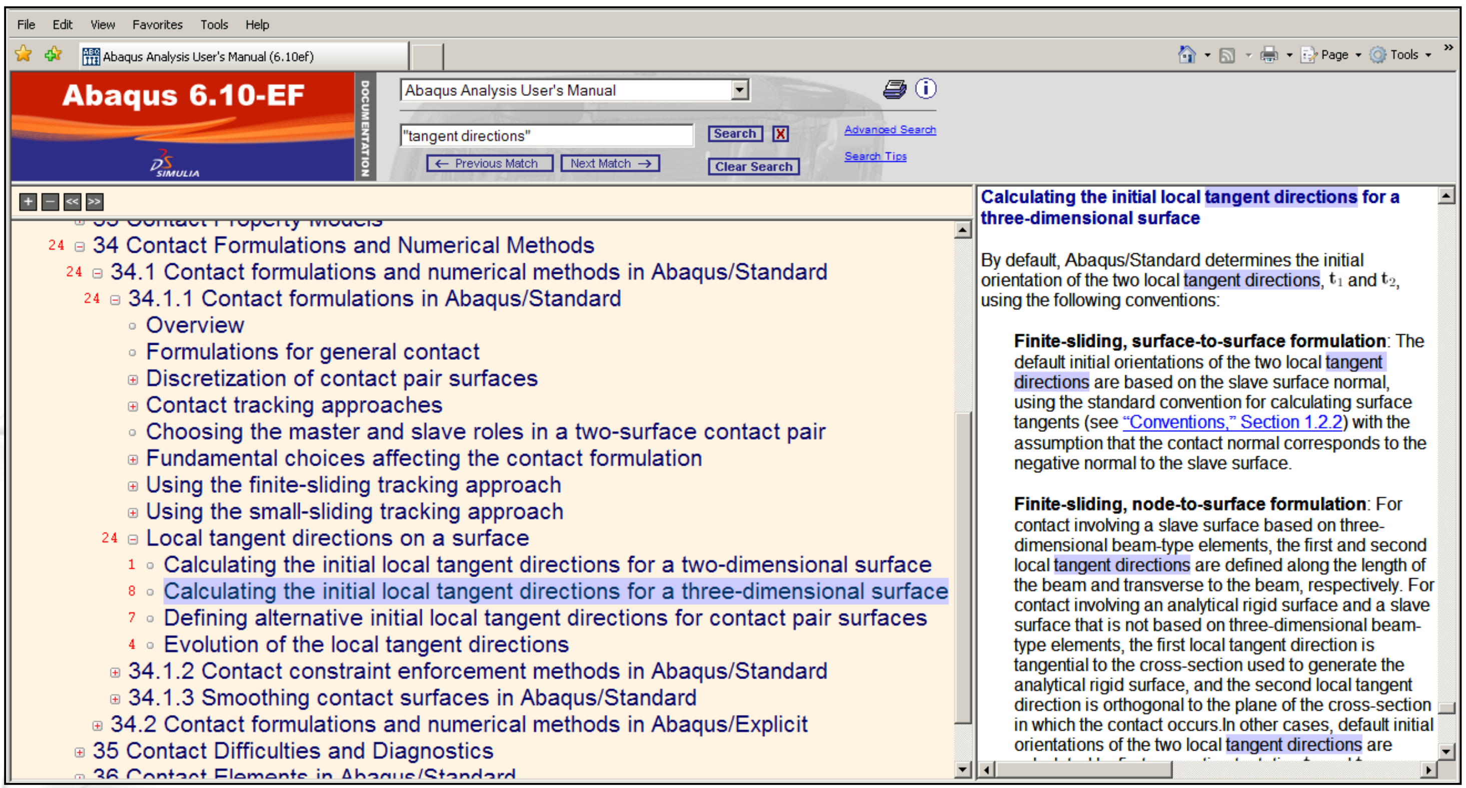






Local Tangent Directions

- **Are used for:**
  - Contact output (e.g., components of slip & shear stress)
  - Anisotripic friction (different μ_1 and μ_2)
- **Conventions** (see Manual)



Nonlinear Friction Coefficient

- **Friction coefficients can be functions of:**
  - Equivalent slip velocity, $\dot{\gamma}_{eq} = \sqrt{\dot{\gamma}_1^2 + \dot{\gamma}_2^2}$
  - Contact pressure, *p*
  - Average surface temperature, $\overline{\theta} = \dfrac{\theta^A + \theta^B}{2}$
  - Average field variable value, $\overline{f}_i$
- **For linear interpolation of tabular data:**
  - If μ is a function of field variables, the DEPENDENCIES parameter must be used on the *FRICTION option to specify the number of field variable dependencies

Nonlinear Friction Coefficient

- **User subroutine `FRIC_COEF` (and `VFRIC_COEF`)**
  - Allows you to specify an expression for the friction coefficient
  - For Abaqus/Standard, also provide expressions for derivatives
- **Example:** $\mu = A\,(1 + B\,\dot{\gamma} + C\,\dot{\gamma}^2)\,(1 + D\,p)$

```
*SURFACE INTERACTION, NAME=name
*FRICTION, USER=COEFFICIENT,
  PROPERTIES=4
 A, B, C, D (substitute real numbers)
```

```
      subroutine fric_coef ( fCoef, fCoefDeriv,
     *   nBlock, nProps, nTemp, nFields, jFlags, rData,
     *   surfInt, surfSlv, surfMst, props, slipRate, pressure, tempAvg, fieldAvg )

      include 'aba_param.inc'
      dimension fCoefDeriv(3)
      parameter ( one = 1.d0, two=2.d0 )

      fs = one + props(2)*slipRate + props(3)*slipRate**2
      fp = one + props(4)*pressure

      fCoef = props(1) * fs * fp

      fCoefDeriv(1) = props(1) * (props(2) + two*props(3)*slipRate) * fp
      fCoefDeriv(2) = props(1) * fs * props(4)
      fCoefDeriv(3) = zero

      return
      end
```

μ → fCoef

$\frac{\partial \mu}{\partial \dot{\gamma}}$ → fCoefDeriv(1)

$\frac{\partial \mu}{\partial p}$ → fCoefDeriv(2)

$\frac{\partial \mu}{\partial \theta}$ → fCoefDeriv(3)

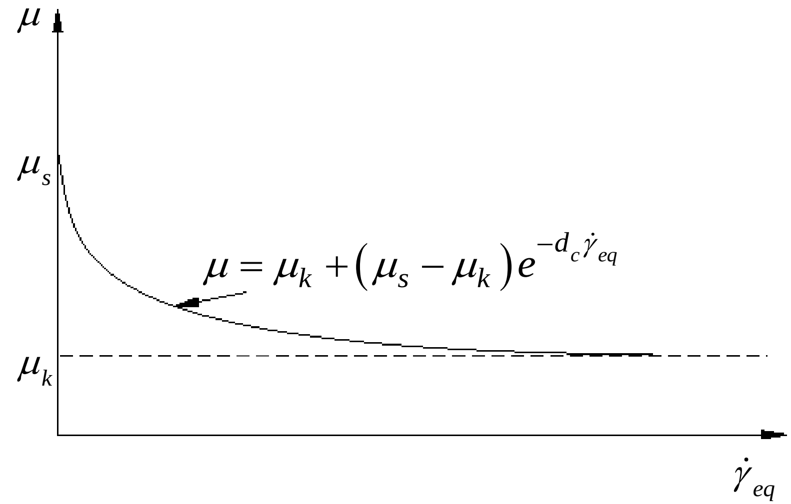
SIMULIA

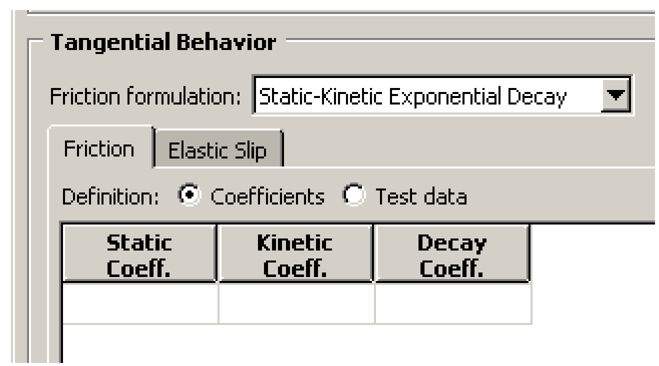
Nonlinear Friction Coefficient

- **Kinetic friction model: Specific form of friction coefficient vs. slip rate**
  - Exponential transition from a static friction coefficient (μ_s) to a kinetic friction coefficient (μ_k)

    $$\mu = \mu_k + (\mu_s - \mu_k)e^{-d_c\dot{\gamma}_{eq}},$$

    where d_c is the decay coefficient
  - Two methods for defining this model:
    - Provide the static, kinetic, and decay coefficients directly
    - Use test data to fit the exponential model



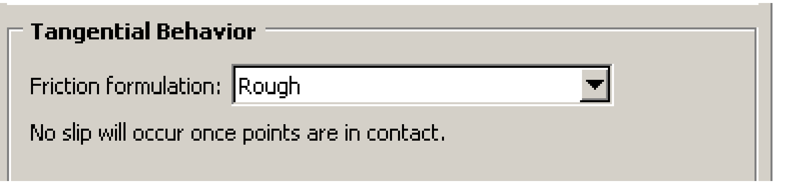



```
*SURFACE INTERACTION
*FRICTION, EXPONENTIAL DECAY
```

“Rough” friction

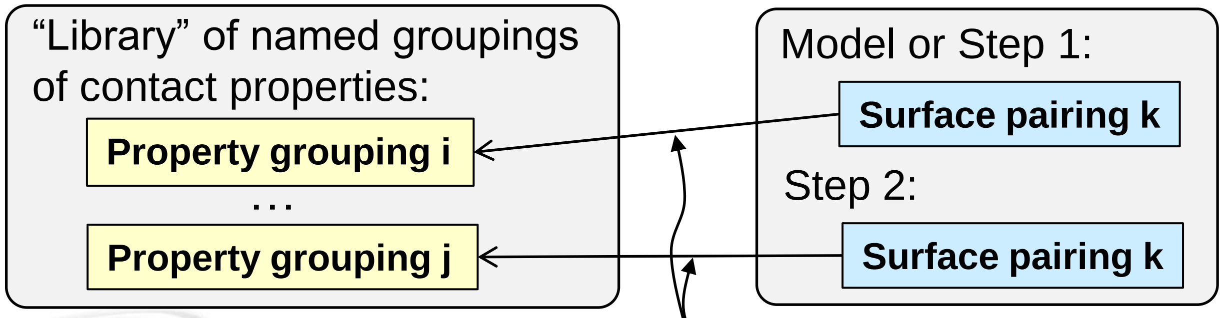
- **Optional behavior in which sticking conditions are always enforced while surfaces are in contact (i.e., while normal constraints are active)**
  - Similar to Coulomb friction with $\mu = \infty$
    - But if “NO SEPARATION” behavior is also specified, resist relative motion even if normal contact forces are tensile
  - Idealized model has zero slip while in contact
    - But small amount of slipping may occur due to numerical softening (for penalty enforcement of sticking condition)
  - Motivation for using rough friction may be physical or numerical (avoid convergence problems)

```
*SURFACE INTERACTION, NAME=name
*FRICTION, ROUGH
```



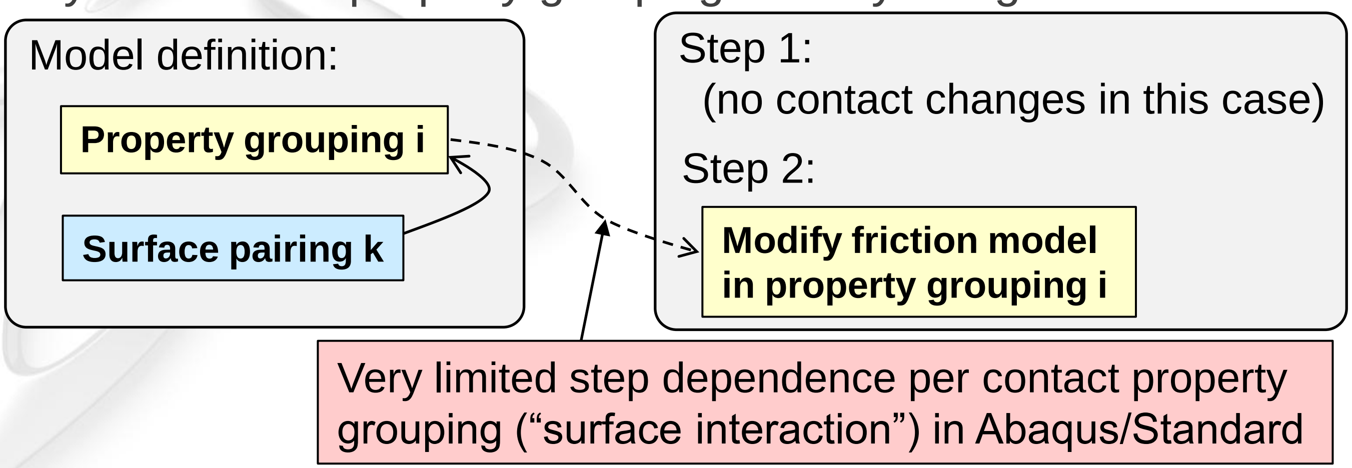
Changing Friction Properties during an Analysis

- **Abaqus/Explicit:** Assign a different named "grouping" of contact properties
  - Friction model is one part of a contact property grouping



Assignment of contact property grouping (or "surface interaction") is step dependent in Abaqus/Explicit

- **Abaqus/Standard**
  - Modify the contact property grouping already assigned



Changing Friction Properties for Abaqus/Standard

- **Keyword interface:**

  ∗CHANGE FRICTION, INTERACTION=*name*

  ∗FRICTION

- **Examples of what can be changed:**
  - Friction coefficient (most common)
    - Gradually ramped from old value to new value over increments of step for most step types
  - Slip tolerance associated with penalty enforcement of stick conditions (uncommon)
    - Starting in Abaqus 6.10, slip tolerance transition uses same ramping behavior as friction coefficient transition in most cases
    - Previously any change was suddenly applied

$$\mu(t) = \mu_{initial} + (\mu_{final} - \mu_{initial}) \times A(t)$$

$$F_f(t) = F_{f\ initial} + (F_{f\ final} - F_{f\ initial}) \times A(t)$$

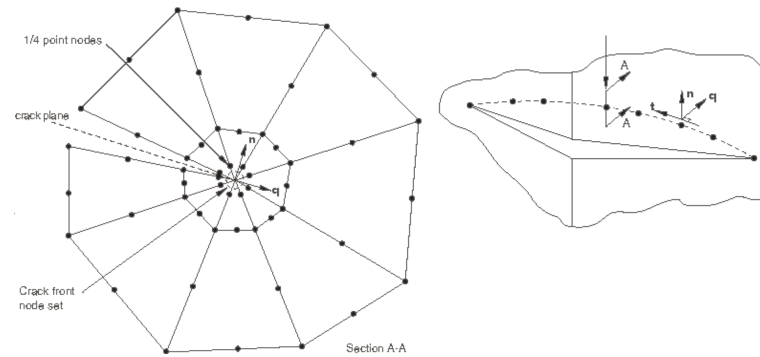
Cohesive contact, cracks, and related features

High-level overview

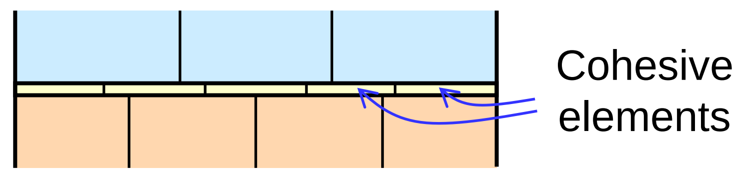
SIMULIA

Stress Intensity Factors, Crack Growth, Delamination, etc.

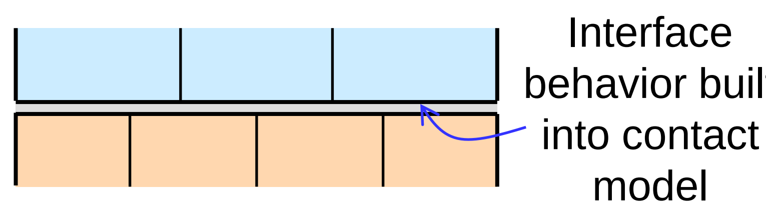
- **Meshing options**
  - Focused mesh around crack tip
    - Traditional approach for evaluating



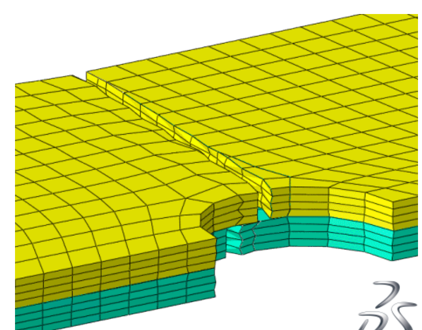
  - Cohesive elements
    - Special elements with nodes on both sides of an interface



  - Surface-based cohesive behavior
    - Contact constitutive model may include adhesive behavior and possibility of failure

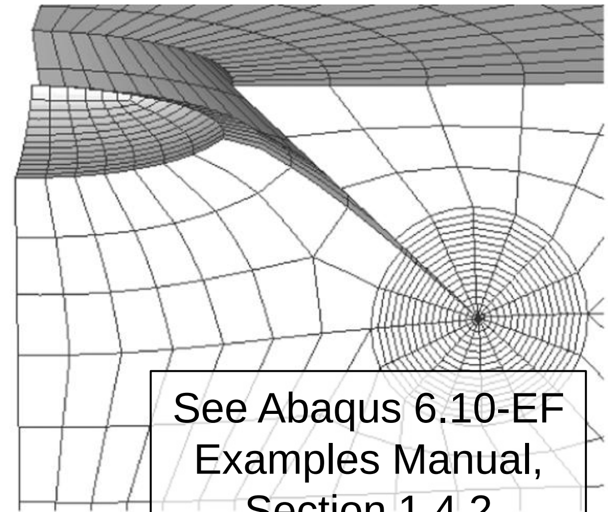


  - XFEM
    - Discontinuities (e.g., cracks) *within* elements
    - Arbitrary, solution-dependent crack path (without re-meshing)

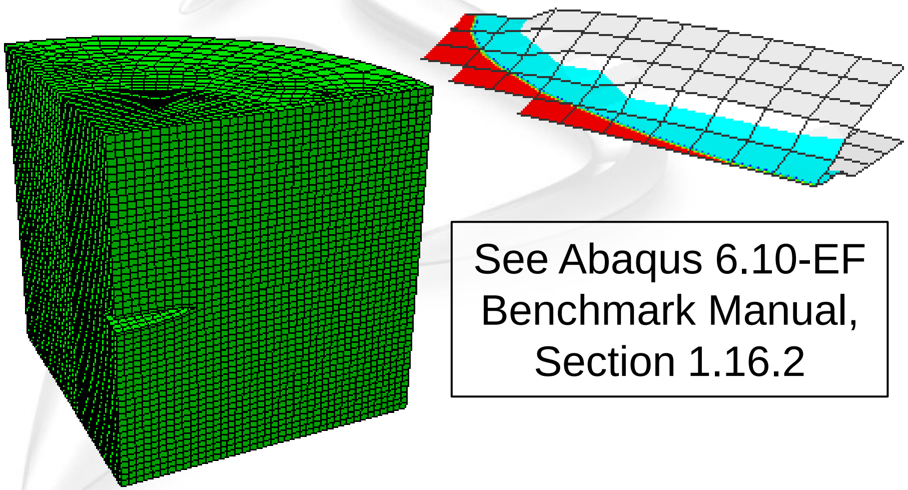


Stress Intensity Factor, Crack Growth, Delamination, etc.

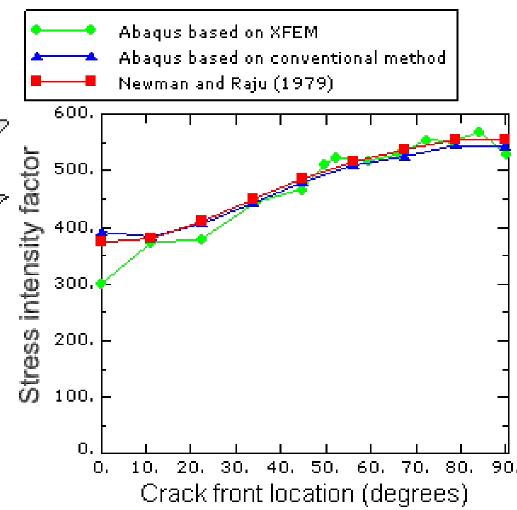
- **Methods to evaluate SIF for stationary cracks**
  - Focused mesh of traditional elements
    - Tried and true method, although somewhat tedious meshing
  - Extended finite element method (XFEM)
    - Create mesh without consideration of crack geometry
    - Then introduce crack



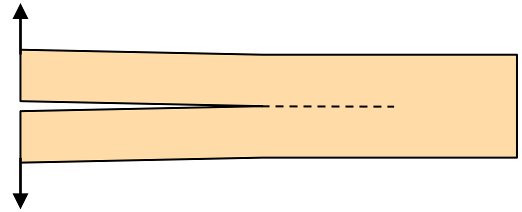
See Abaqus 6.10-EF Examples Manual, Section 1.4.2



See Abaqus 6.10-EF Benchmark Manual, Section 1.16.2



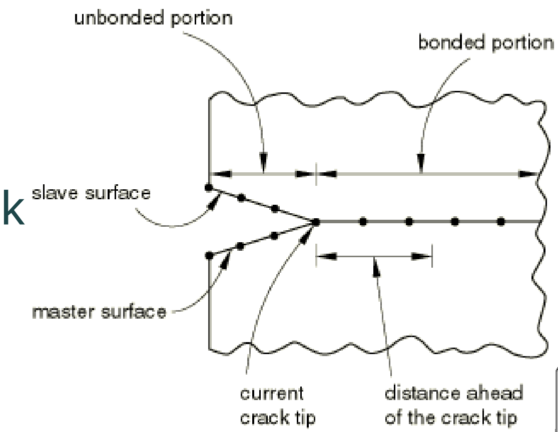
Delamination, Crack Gowth, etc.

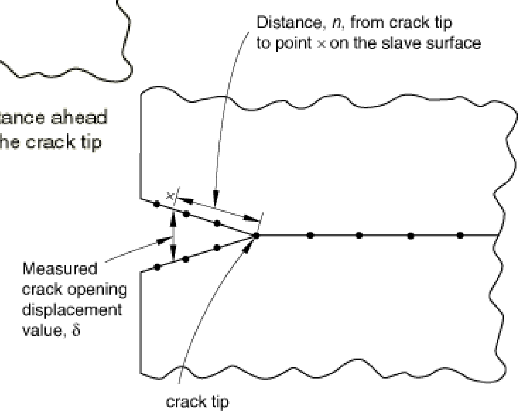


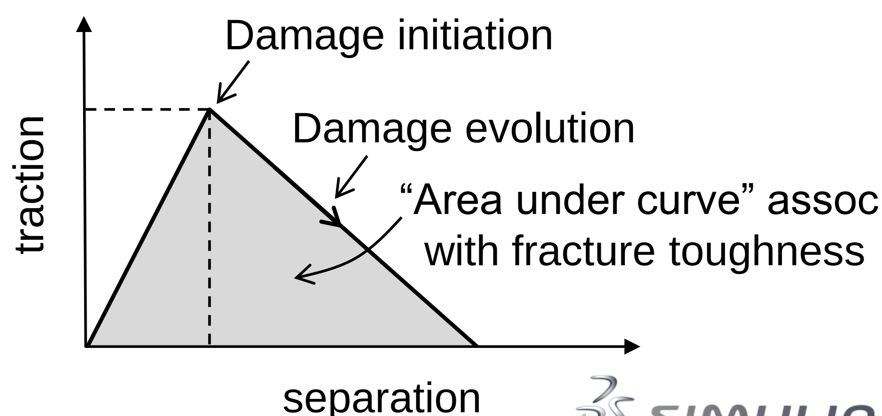
- **Fracture/failure models**
  - Crack propagation criteria
    - Critical stress ahead of crack
    - Critical crack opening displacement
    - VCCT (virtual crack closure technique)
    - Crack length vs. time
    - Low-cycle fatigue based on Paris law
  - Traction-separation model
    - Built into constitutive model
    - No need for an initial crack

Use with cohesive elements or contact

Use with XFEM, cohesive elements, or contact



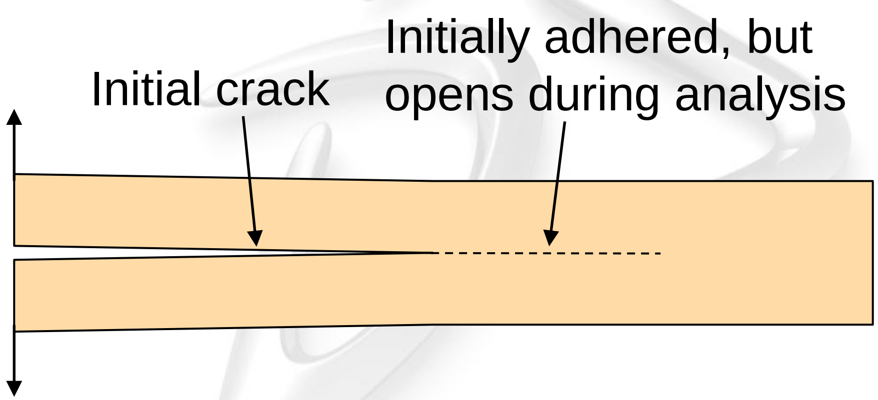


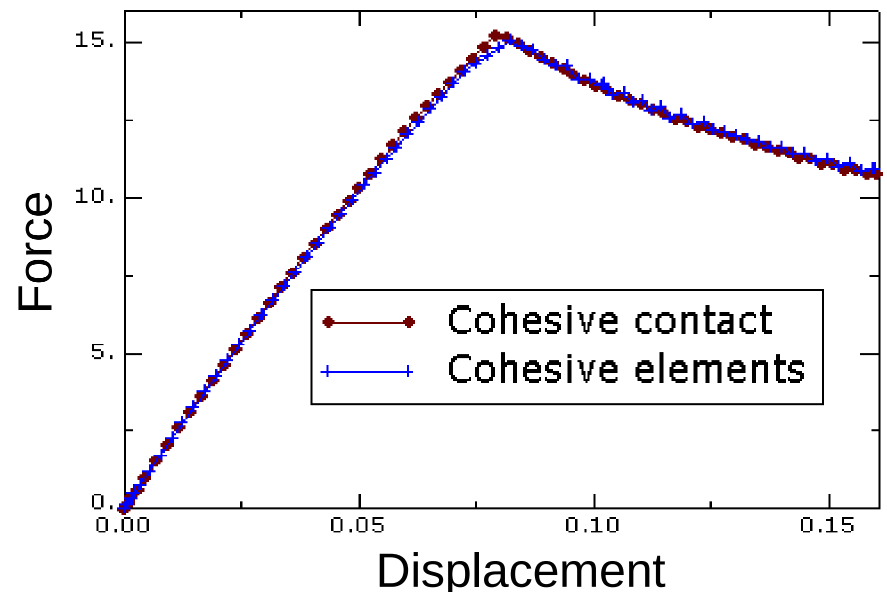


Cohesive Contact vs. Cohesive Elements

- **Cohesive contact avoids the following aspects when creating model**
  - No separate mesh for the adhesive
  - Not required to specify the undamaged traction-separation behavior
  - No density associated with the adhesive (for dynamic procedures)
- **Consistent specification of damage behavior**
- **Results often in close agreement**

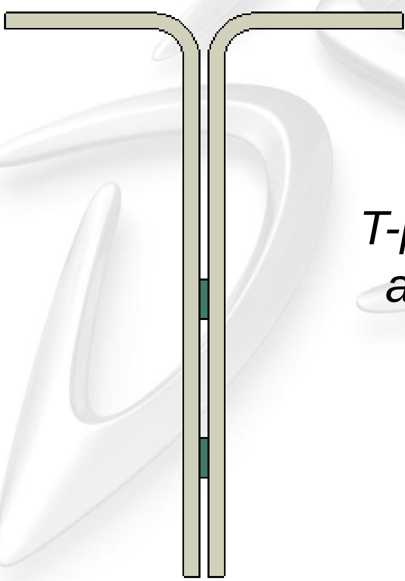
Crack growth example

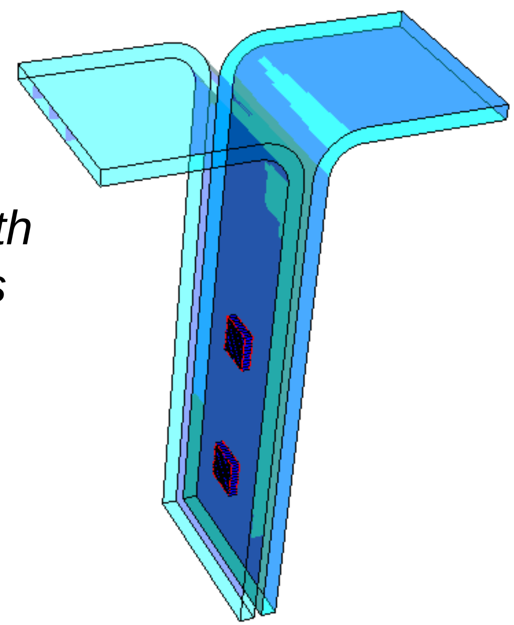




Cohesive contact vs. cohesive elements

- **Usability simplifications imply some applicability limitations**
  - Circumstances in which cohesive elements are recommended:
    - Mesh for adherents is not adequately refined to capture adhesive behavior
    - Undamaged behavior other than “traction-separation” needed
    - Normal directions of contact surfaces significantly deviate from being “directly opposed,” while the cohesive remains active

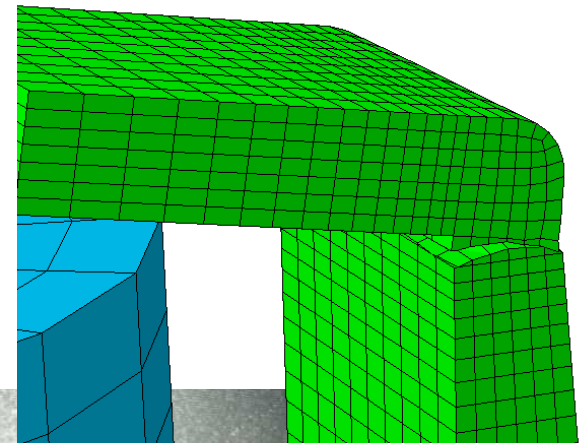


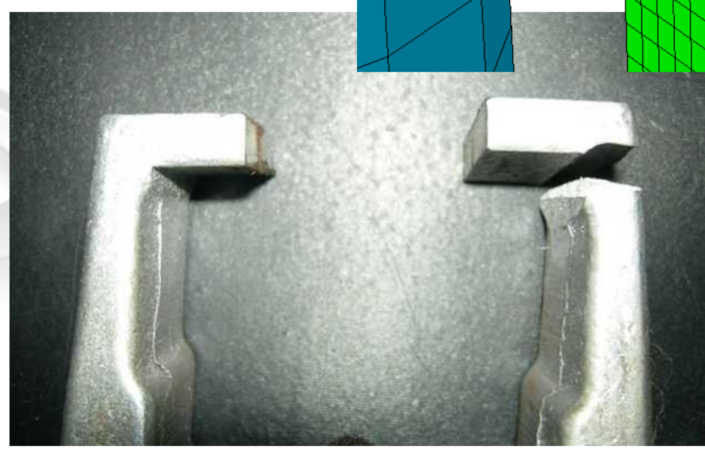


T-peel example with adhesive patches

Cohesive elements/contact approaches vs. XFEM

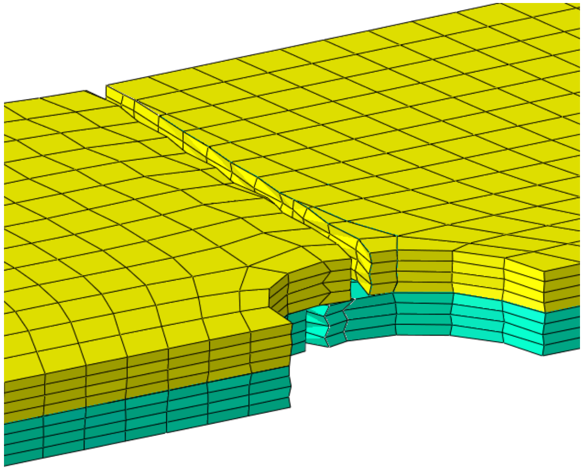
- **Cohesive elements/contact are applicable to situations in which location of delamination or cracking is pre-determined**
  - For example, adhered interfaces
- **XFEM**
  - Crack path is not pre-determined





Contact Involving Surfaces Formed During XFEM Analysis

- **Limited to:**
  - Resisting penetration upon re-closing (of cracked region) with a small-sliding contact formulation using a penalty method
    - Only if a contact property is referred to in the XFEM “enrichment” specification (by the user)
- **What isn’t modeled (yet)**
  - Contact with other surfaces
  - Finite-sliding contact of re-closed region
  - Friction of re-closed region



Stabilization of Implicit Models With Cracking/Delamination

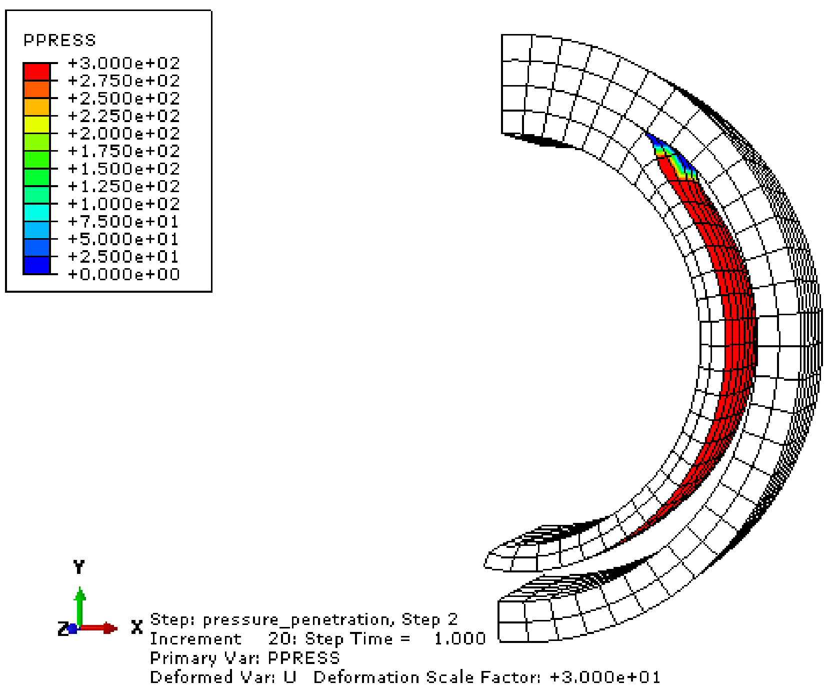
- **Stiffness degradation associated with interface failure is likely to cause convergence difficulties in Abaqus/Standard**
  - Search for "viscous regularization" in the Abaqus Anlaysis User's Manual
    - Discussed in several sections
    - Another tool to help mitigate these problems
  - Inertia effects of dynamic analyses have stabilizing characteristics
    - For example, XFEM applicable to implicit dynamic procedure starting in Abaqus 6.10

Pressure-Penetration Loading

SIMULIA

Pressure-penetration loading

- **Models effects of pressurized fluid penetrating between contact surfaces**
  - Without directly modeling the fluid (no fluid elements)
  - Similar to “DLOAD,” but with an algorithm to control where the load is applied over time
    - Contact pressure threshold governs expansion of “wetted region”
  - Available in 3D starting in Abaqus 6.10EF



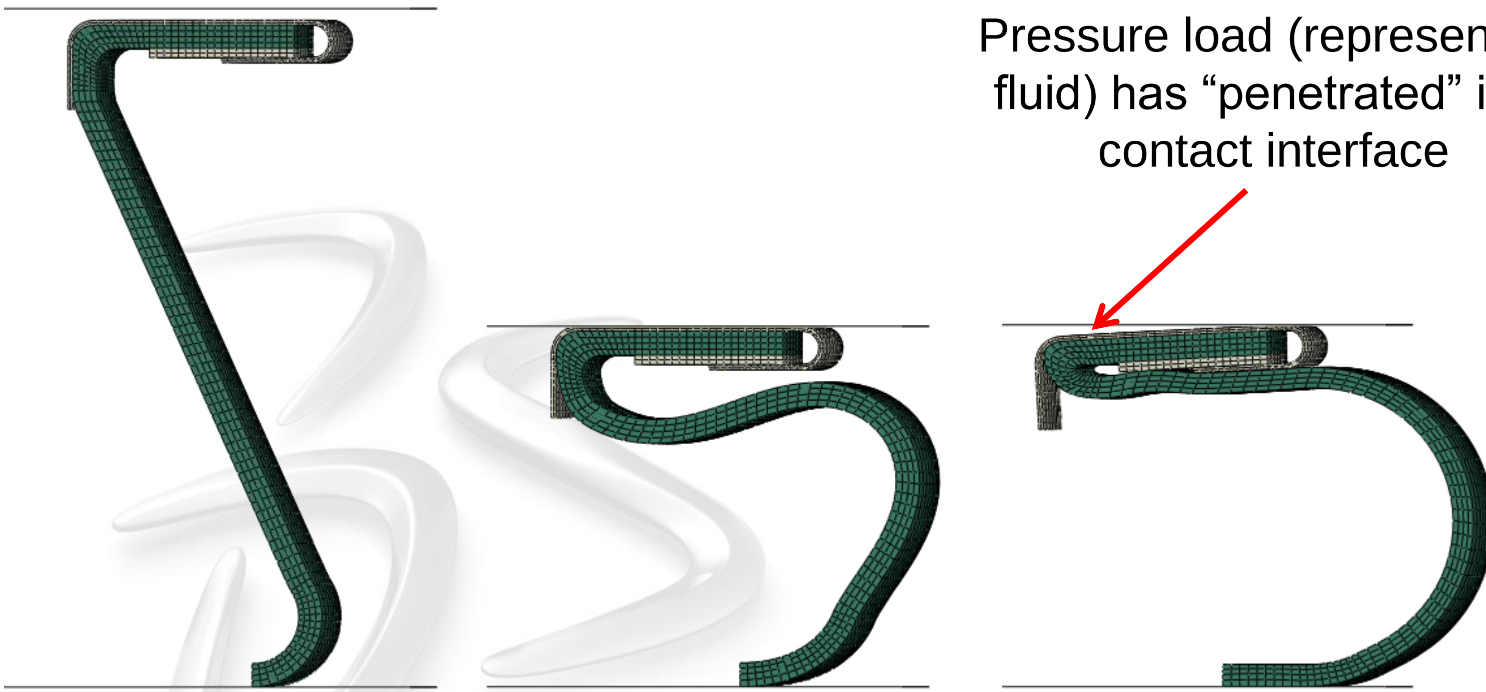
Pressure-penetration loading

- **Use with contact pairs**
  - Refer to slave and master surfaces of contact pair
  - Identify at least one slave node initially exposed to fluid
  - Not yet supported with general contact
- **Expansion of the "wetted region" is not instantaneous once the pressure-penetration criterion is reached**
  - Current fluid pressure is ramped on over 0.001 of step time by default
    - Can control magnitude of fluid pressure vs. time with an amplitude definition
  - Results may depend on time increment size
    - Recommend controlling maximum time increment size to obtain accurate results
- **Wetted region does not shrink**
  - Even if contact pressure returns above threshold

Pressure-penetration loading

- **Air duct seal example**
  - Section1.1.16 of Abaqus Example Problems Manual

Pressure load (representing fluid) has “penetrated” into contact interface



Undeformed configuration

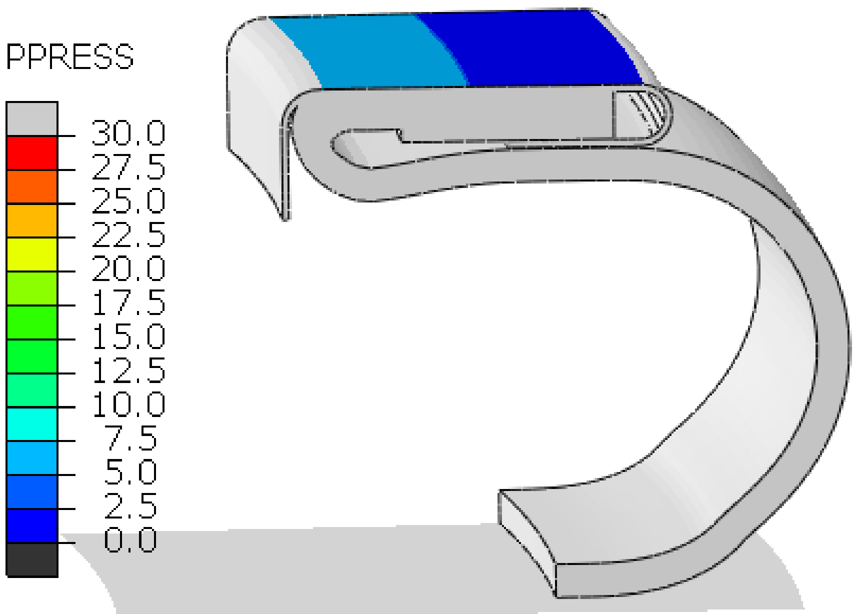
After moving rigid surfaces closer together

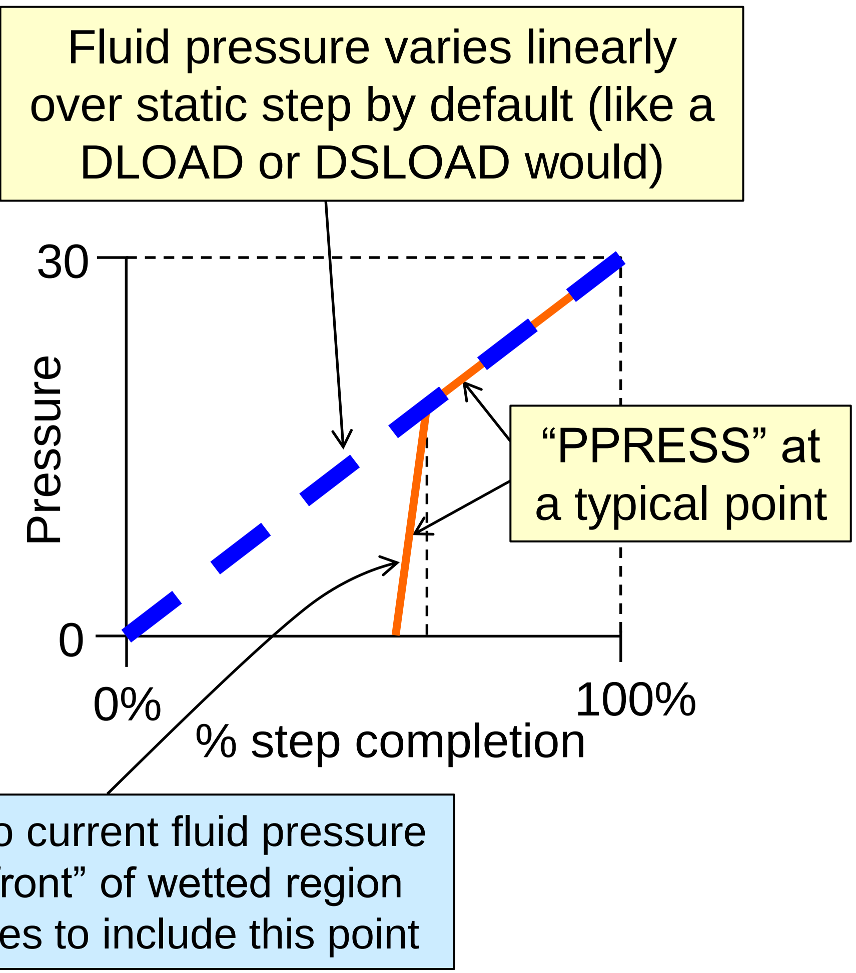
Response to fluid pressure loading

Animation on next slide

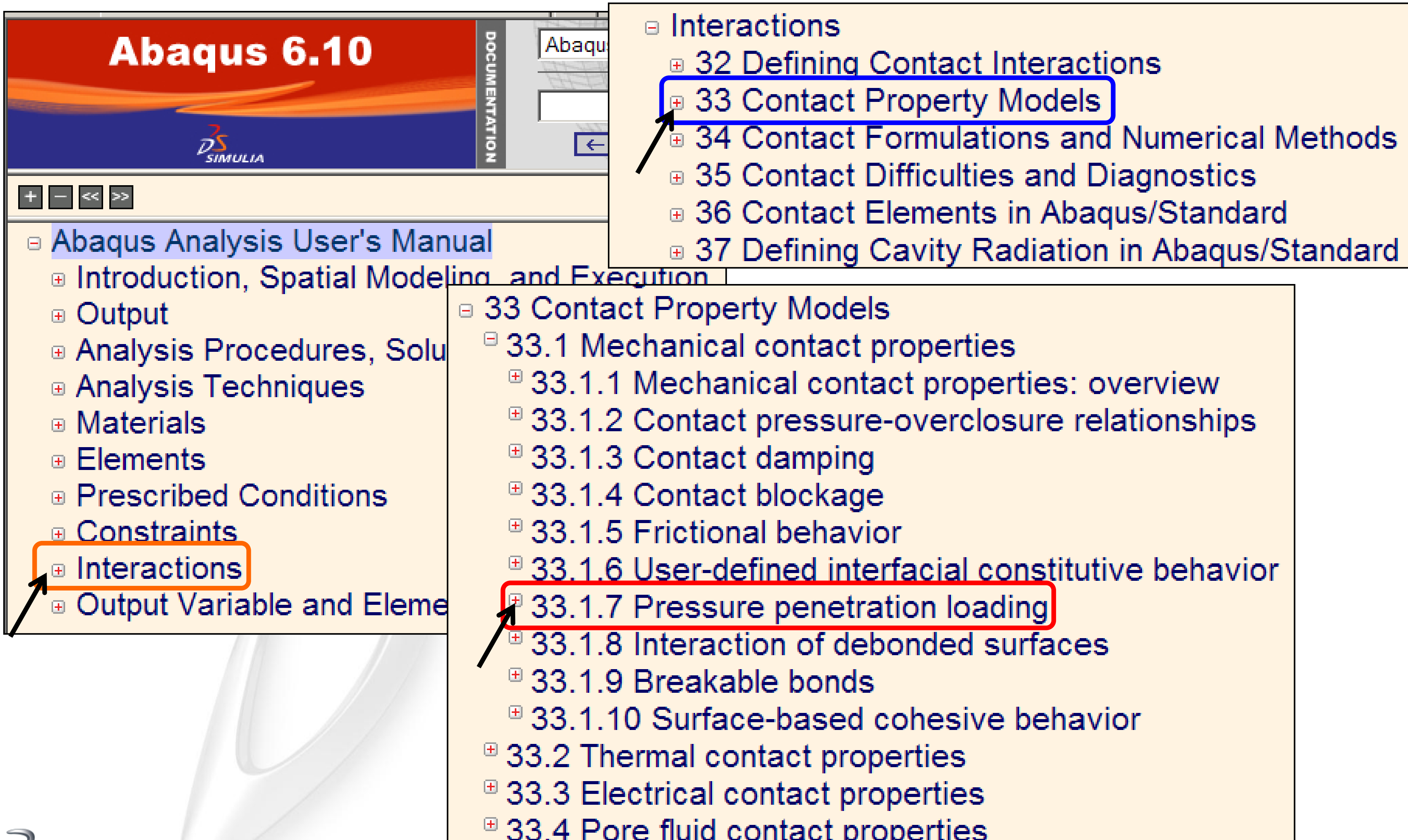
Pressure-penetration loading

- **Air duct seal example**



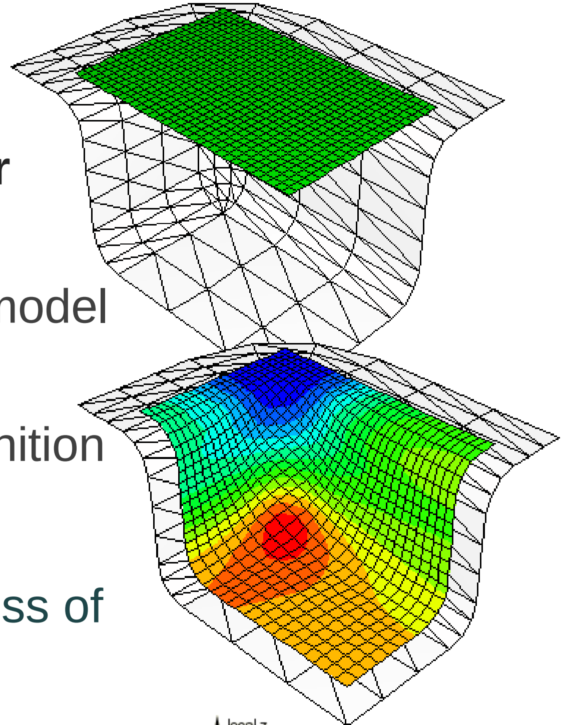


For more information…

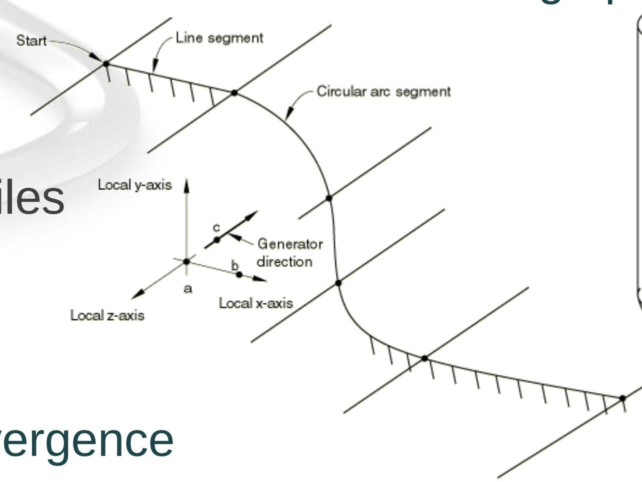


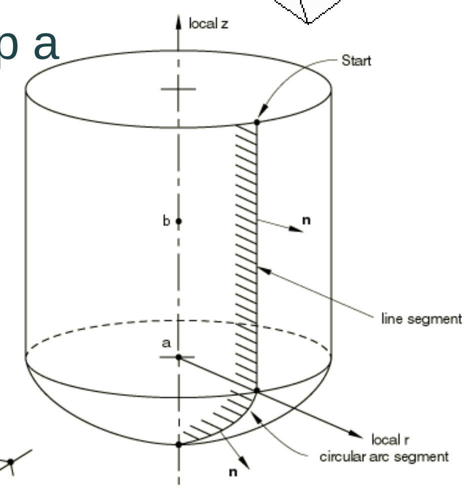
Rigid bodies

Rigid Bodies and Contact



- **Model a body as rigid if it is much stiffer than other bodies with which it will come in contact**
  - For example, rigid bodies are commonly used to model dies in metal forming simulations
  - Include set of (regular) elements in rigid body definition
  - Saves computations
    - 6 degrees of freedom per rigid body (regardless of number of nodes included in the rigid body)
    - No element calculations for elements making up a rigid body
- **Analytical rigid surfaces**
  - For cases with 2D profiles
  - Exact geometry
  - Smooth
    - Beneficial for convergence

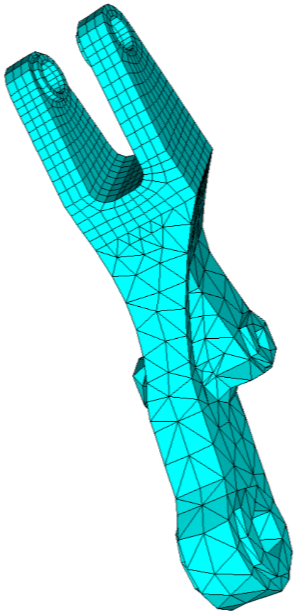


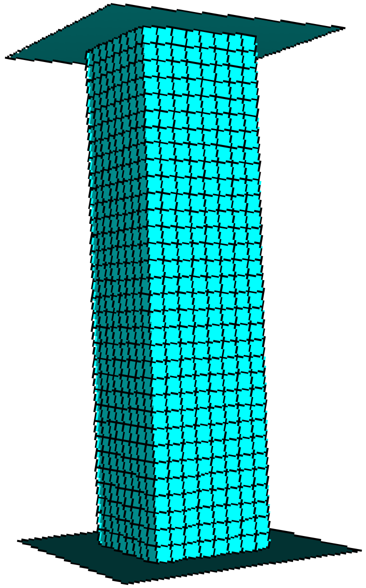


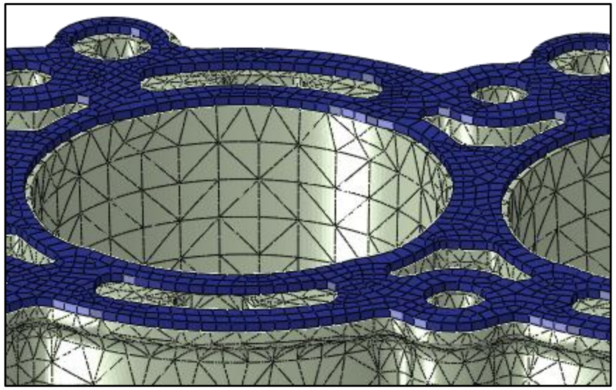
Ties

Surface-Based Tie Constraints

- **Potential applications**
  - Mesh-refinement transitions
  - Two parts that are permanently attached together (no chance of debonding)
  - Approximation of contact interface where user expects separation and sliding to be nonexistent or insignificant
    - Nonphysical results if such assumptions are not valid! (User's responsibility)
- **Initialization aspects**
  - Position tolerances govern what regions are actually tied
  - Strain-free adjustments to achieve compliance







Surface-Based Tie Constraints

- **Two keyword interfaces** (!)
  - *Tie
    - Constraints are enforced by eliminating slave degrees of freedom prior to equation solver
      - Slave node tied to multiple master surfaces is problematic
    - Cannot view constraint stresses
  - *Contact Pair, Tied
    - Constraints are enforced either with a Lagrange multiplier method or a penalty method
      - Slave DOF (and any Lagrange multipliers) are exposed to equation solver
      - Overconstraints are not as problematic with a penalty method
    - Can view constraint stress (CPRESS & CSHEAR)
  - Some other differences exist in details of these two implementations

Other physics

Other physics

- **Interactions may also involve thermal, electrical, and pore fluid fields**
  - (If underlying elements involve these fields)
  - Specify contact conduction, etc. properties

- 33 Contact Property Models
  - 33.1 Mechanical contact properties
  - 33.2 Thermal contact properties
    - 33.2.1 Thermal contact properties
      - Overview
      - Including thermal properties in a contact property definition
      - Thermal contact considerations in Abaqus/Explicit
      - Modeling conductance between surfaces
      - Modeling radiation between surfaces when the gap is small
      - Modeling heat generated by nonthermal surface interactions
      - Surface-based interaction variables for thermal contact property models
      - Thermal interaction variables for thermal gap elements
      - Thermal interactions involving rigid bodies
      - Modeling thermal interactions with node-based surfaces
      - Thermal interactions between surfaces with nodes containing multiple temperature degrees of freedom
  - 33.3 Electrical contact properties
  - 33.4 Pore fluid contact properties

Lecture 7 Summary

Review of Topics Discussed in this Lecture

- **Abaqus Analysis User's Manual**
- **Contact constitutive models**
  - Pressure-Overclosure
  - Friction
- **Cohesive contact, cracks, etc.**
- **Pressure-penetration loading**
- **Rigid bodies**
- **Tie constraints**
- **Other physics**