WCCM V
Fifth World Congress on
Computational Mechanics
July 7–12, 2002, Vienna, Austria
Eds.: H.A. Mang, F.G. Rammerstorfer,
J. Eberhardsteiner

Verification Testing of Contact in a Finite Element Code For Quasi-Static Solid Mechanics Equilbrium

John A. Mitchell* **and Martin Heinstein**

Sandia National Laboratory
Computational Solid Mechanics/Structural Dynamics
1515 Eubank, SE
Albuquerque, NM 87185-0847
e-mail: jamitch@sandia.gov

Key words: nonlinear solid mechanics, quasi-static equilibrium, contact mechanics, testing

Abstract
ADAGIO [1] is a quasi-static nonlinear finite element program for use in analyzing the deformation of solids. It is massively parallel, built upon the SIERRA finite element framework [2], and employs the ACME library [3, 4, 5] for contact search algorithms. The mechanics and algorithms in ADAGIO closely follow those previously developed in JAC2D by Biffle and Blanford [6] as well as JAS3D by Blanford et. al [7]. ADAGIO assumes a quasi-static theory in which material point velocities are retained but time rates of velocities are neglected. Sources of nonlinearities include nonlinear stress-strain relations, large displacements, large rotations, large strains, and frictional/frictionless contact mechanics. Quasi-static equilibrium is found using a nonlinear solution strategy which includes nonlinear conjugate gradients. In this paper, contact mechanics capabilities described in a companion paper (Heinstein and Mitchell) are exercised via a set of focused regression tests. These tests highlight contact search and enforcement capabilities available in ADAGIO and include: interactions with kinematic boundary conditions, multiple contact constraints, normal smoothing at facet interfaces, capture and release mechanisms, and problems with known analytical solutions. Nevertheless, the small set of tests described here are necessary but not sufficient to insure that a code has a robust working contact enforcement capability.

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under contract DE-AC04-94AL85000.

1 Quasistatic equilibrium

Quasistatic equilibrium in Adagio is based upon the principle of virtual work in a rate form:

$$\int_{\Omega} \bar{T} : \delta\dot{\bar{\varepsilon}}\, dV - \int_{\Omega} \rho\vec{b} \cdot \delta\vec{v}\, dV - \int_{\partial\Omega} \vec{t} \cdot \delta\vec{v}\, dA = 0 \tag{1}$$

where Ω corresponds to the volume of the body in the current configuration, $\partial\Omega$ is the boundary of the body in the current configuration, $\bar{T}$ is the Cauchy stress tensor, $\vec{v}$ is the material point velocity, $\delta\dot{\bar{\varepsilon}}$ is the symmetric part of the virtual velocity gradient, $\vec{t}$ is an applied surface traction, ρ is mass density, and $\vec{b}$ is a body force vector. The extent to which equation 1 is not satisfied is a measure of the force imbalance and lack of quasistatic equilibrium. This force imbalance is called the residual and quasistatic equilibrium is defined according to how close the residual is to zero.

Adagio solves for quasistatic equilibrium over a set of time increments $\Delta t = t_{n+1} - t_n$ defined by a sequence of times $t_n \;\; n = 0, 1, 2, \ldots$. A force imbalance occurs at t_{n+1} due to loads, thermal strains, or kinematic boundary conditions that are parameterized by time. Quasistatic equilibrium is assumed to exist at t_n. The solver searches for a suitable equilibrium configuration at t_{n+1} through a sequence of trial velocities that give rise to ever decreasing residuals (force imbalance). Equilibrium is satisfied when the force imbalance reaches a user specified tolerance for convergence. The solver finds velocity vectors $\vec{v}_{n+1}$ for a load step at discrete times t_{n+1} by solving the nonlinear problem implied by the weak form 1.

2 Solution control for contact

A multilevel solution control scheme is used for contact and is referred to here as control contact. The multilevel concept is based on linearizing contact constraints within a nonlinear core solver such as conjugate gradients (CG). After establishing a list of nodes that are in contact (constraint set), a model problem is solved using CG with this set held constant. After model problem convergence, the constraint set is updated to reflect changing contact conditions and this necessarily gives rise to a force imbalance and to another model problem. Changing or updating the constraint set is referred to as a contact update and multilevel convergence typically requires multiple contact updates before equilibrium is achieved. The contact update consists of a gap removal step, an equilrium query, and finally a slip calculation if equilibrium is not satisfied. Gap removal detects contact conditions between surfaces through a combination of proximity searches and contact forces. Slip calculations are performed on slave nodes in order to relieve contact forces that have necessarily built up due to fixing the slave nodes to the master surface during the model problem.

3 Contact Search and Enforcement

The contact algorithms in Adagio currently use the master/slave approach which may be logically separated into two parts: search and enforcement. The search is used to detect potential interactions between the master and slave surfaces. Associated with this pair of surfaces are two search tolerances: normal, and tangential. These tolerances are used to define the notion of contact between the two surfaces that are in proximity to each other. Interaction of a slave surface node with a master surface face is returned by the search algorithm if the slave node is in proximity to the master surface as specified by the search tolerances. For purposes of discussion, we define the contact point as the closest point projection of the slave node onto the master surface. Associated with this contact point we identify a face id, local element coordinates on the face, and a surface normal. All of this information is returned by the search algorithm and is refered to as an interaction. In addition, a push back vector along with its magnitude is returned. The sign of the magnitude is used to communicate whether the node penetrates the master surface or not. The push back vector is a unit vector pointing from the slave node location to the contact point and is used along with the magnitude by the enforcement algorithm to enforce the zero penetration constraint or possibly pull the node to the surface.

Enforcement of contact interactions in Adagio uses an explicit iterative algorithm [8] which consists of two pieces: gap removal, and slip calculations. Given an interaction, it is necessary to either pull or push the node to the surface, and we refer to this as the gap removal phase. As mentioned above, if the push back magnitude is positive, then the node does not penetrate and therefore a decision must be made as to whether the node should be ignored or whether it should be pulled to the surface and considered as a constraint. For nodes that do not penetrate a capture tolerance is used to determine whether the node should be considered further. The capture tolerance is less than or equal to the search tolerance. On the other hand, if the node is penetrated, then the push back magnitude is negative and the node is pushed to the surface. When pushing or pulling a node to the surface we are incrementally adjusting the velocity vector on the slave node which necessarily gives rise to an imbalance of forces and thus we must perform a residual calculation. If the imbalance exceeds the user specified equilibrium tolerance, then slip calculations are performed. In the case of Coulomb friction, we determine whether or not the friction capacity at the node is above or below the residual force imbalance on the node. If the friction capacity is less then the force imbalance, then the node is slipping and we calculate a slip increment based upon the difference of the residual force with the friction capacity. If on the other hand the friction force capacity exceeds the residual force imbalance on the node, then the node is sticking. In either case, slave node contact forces are then assembled onto the master surface.

4 Tests

In this section, a set of contact tests are provided and discussed. These may be useful in the development of a contact capability for a finite element code. They have a certain practical aspect to them in that they exercise combinations of boundary conditions and geometries that are certain to be exercised in practice. Nevertheless, the small and simple set of tests discussed here are necessary, but not sufficient, to insure that a code has a robust working contact enforcement capability.

4.1 Interactions with boundary conditions

It is very common to have multiple constraints at a node and this can lead to the lack of convergence if constraint interactions are not handled in a consistent way. This test is constructed so that a slave node is also part of a Dirichlet boundary condition that is not orthogonal to the sliding friction(less) constraint. A schematic of the test along with an algorithm that can be used for the slave node residual in an explicit treatment of the contact constraint is shown in Figure 1. A similar treatment is required for the slip calculation. At the time of this writing, we are not aware of an analytical solution for this problem except for the simple case in which the coefficient of friction $\mu > tan\phi$, where ϕ is the angle of the contact interface plane. In this case, there is no slip and a simple one dimensional bar solution can be used to verify the calculation. However, this solution may only be valid for small deformations

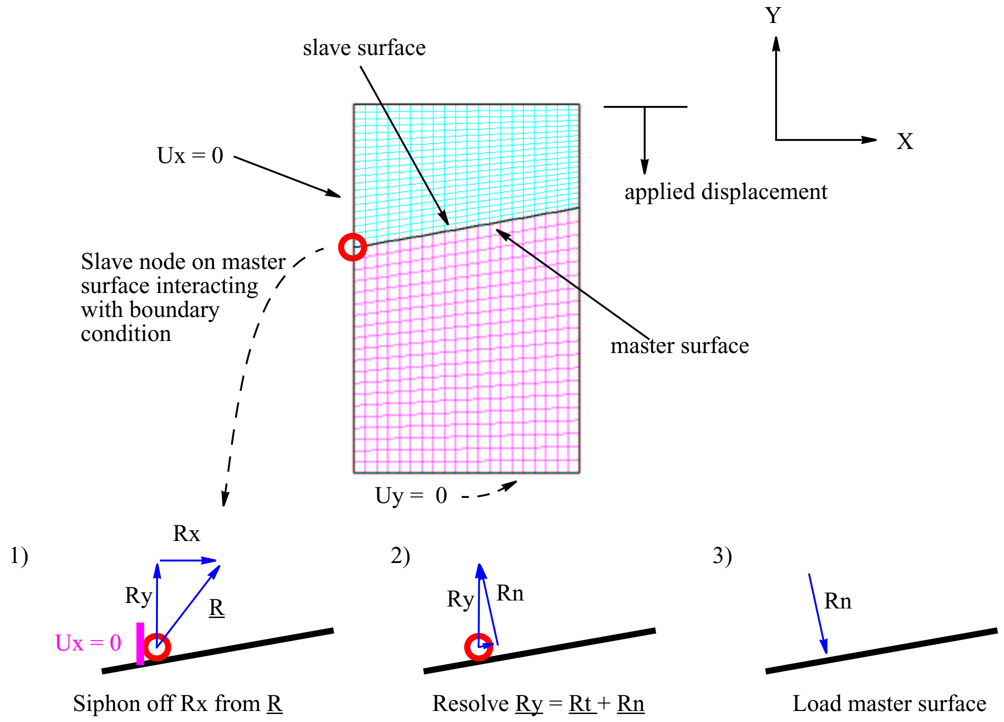


Figure 1: Test: Contact BC Interaction

for which the angle ϕ does not change significantly. Correctly detecting the onset of slip by adjusting ϕ or by running a series of calculations varying μ may also be useful. Results for the x-component of the displacement along the slave node interface are shown in Figure 2 where we have assumed that the interface is frictionless. Since we do not have an analytical solution, we have also given the results calculated by JAS3D.

4.2 Normal smoothing

This is a basic test of the "normal smoothing" capability provided by the search as well as the contact enforcement procedures described above. Although not the main focus of this test, boundary conditions also co-exist with contact at certain nodes. A schematic of the mesh, node numbers, boundary conditions, and contact surfaces is shown in Figure 3. The top two elements are compressed against the lower two elements via a prescibed y-displacement and frictionless contact is enforced on the interface seperating the upper elements from the lower elements. Due to the symmetry, slave nodes 15 and 16 should only move in the y-direction. However, the deformation of the lower two elements forms a numerical instability which causes the slave node to oscillate between two different master surface facets that have different surface normals. As shown in Figure 3, the deformation of the lower two elements is such that a "V" is formed in the master surface. The problem solution is such that nodes 15 and 16 move only in the y-direction and lie directly at the bottom of the "V". However, depending on which facet is used as the contact constraint, the x-component of the master surface normal changes sign. This discontinuity in the surface normal creates an the iterative instability which is the subject of this test. Here, normal smoothing allows for the iterative solution to converge quickly.

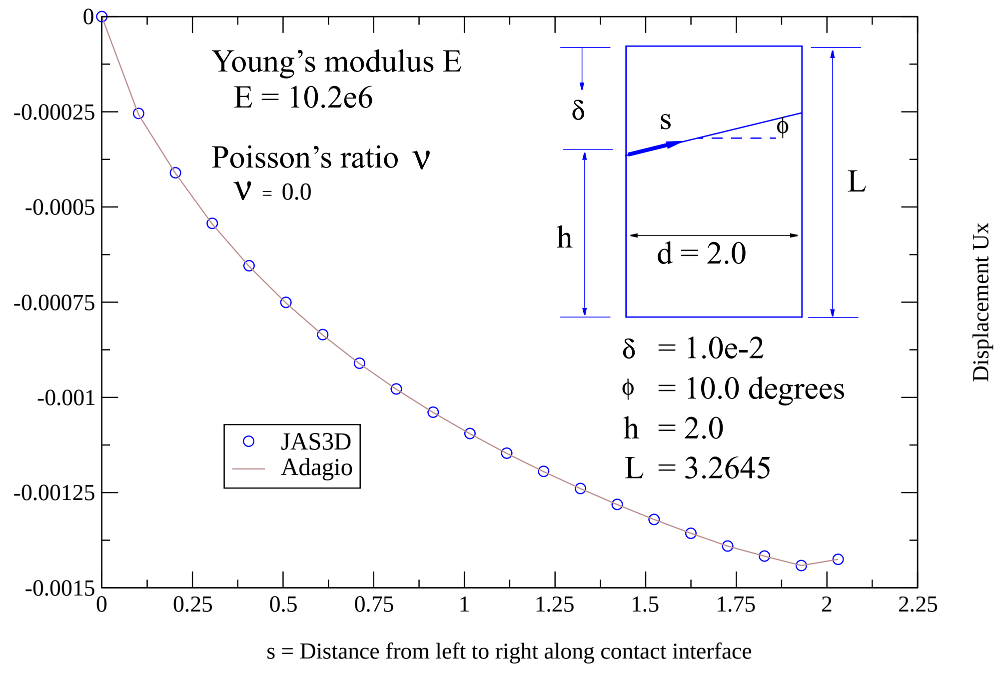


Figure 2: Test: Contact BC Interaction; Displacement $U_x(s)$

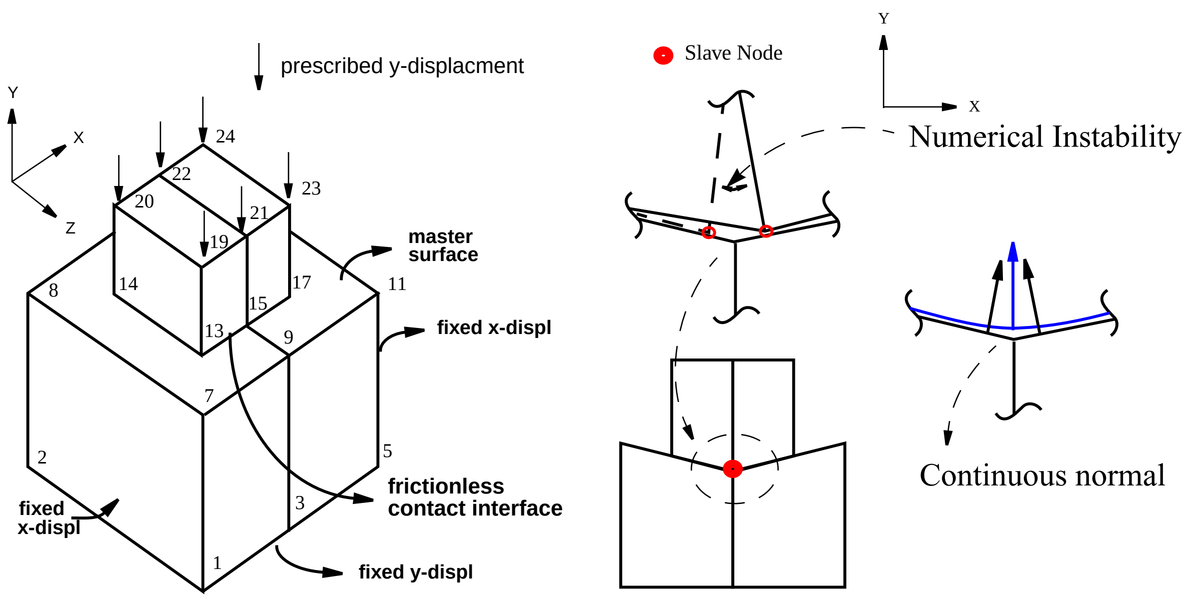


Figure 3: Test: Normal Smoothing Schematic

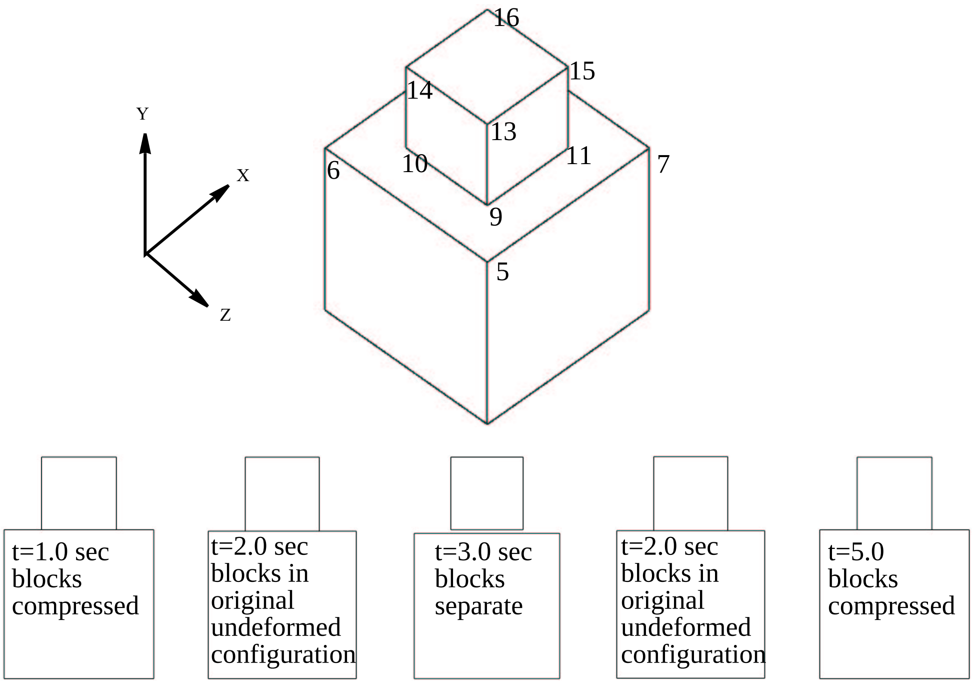


Figure 4: Test: Catch and release

4.3 Capture and release

In this test, the applied kinematics are such that nodes come in and out of contact thus exercising the mechanisms that are used to detect constraints as well as the mechanisms used to release constraints. This is accomplished by prescribing a displacement along the y-direction of nodes 13, 14, 15, and 16 as indicated in Figure 4. Figure 5 shows the prescribed motion of these nodes as a function of time. Note that in the first segment [0.0,1.0], the y-displacement is negative so that contact enforcement is required between the top and bottom element. In the next time segment [1.0,2.0], the prescribed motion returns the nodes to their original undeformed configuration so that there should be zero loads across the contact interface although both surfaces are just touching. In the third time segment [2.0,3.0], nodes 13, 14, 15, and 16 are given a positive y-displacement so that the top element moves away from the bottom element. In this case, the nodes are released and there are no contact constraints. In the next time segment, [3.0,4.0], the prescribed nodes are returned to their original underformed configuration. Here again, the contact forces should be zero and there should be zero stresses in both elements. In the last time segment [4.0,5.0], the nodes are given a negative displacement, and the elements are again compressed against each other. The solution for this time segment should be the same as the solution for the first time segment [0.0,1.0].

4.4 Stick/slip

One of the most fundamental aspects of modeling Coulomb friction is managing the stick/slip phenomenon. In this one-dimensional test we have an analytical solution and we focus in on a stick slip front that propagates according to the loading. A schematic of the test and results are shown in Figure 6. As shown in the figure, there is a pressure loading phase during the time interval $t \in [1.0, 2.0]$, and we

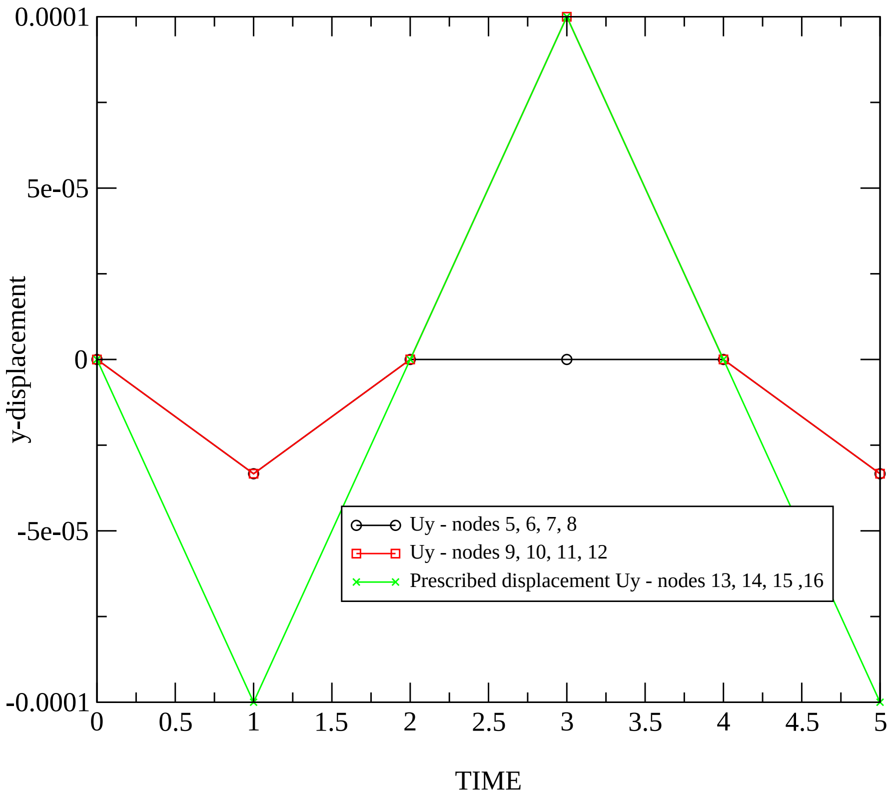


Figure 5: Test: Catch and release; Motion history

have an analytical solution for this phase. The analytical solution is given as $u(x) = \frac{\mu \delta x^2}{h^2}$ where μ is the coefficient of friction and $s = L - x$. Here we define L to be the length of the bar that is slipping and is given as $L = \frac{ph^2}{2\mu \delta E}$ where E is youngs modulus. For the results shown, $E = 10.0e6$, $\mu = .5$, $h = .1$, and the remaining parameters are given in the figure. The coordinate variable x is zero at the stick slip front and increases to the left until it reaches the left end at $x = L$. For purposes of plotting and visualization we have used s in the graphs. During the unloading portion of the test $t \in [2.0, 3.0]$ we do not have an analytical solution. The unloading portion can be difficult to compute if the finite element code has a loadstep and or slip predictor which utilizes the computed solution from the last load step. As shown in the figure, on onloading, most of the nodes stick and only the nodes near $s = 0.0$ slip after the unloading has been active for a time.

4.5 Multiple contact constraints

Multiple contact constraints occur frequently in practice, and this is an entry level test for developing the capability. A contact patch test is given to validate the correctness of the treatment for multiple contact constraints at a slave node as well as the search algorithms used to detect/compete possible constraints. Shown in Figure 7 is an assembly of blocks that are subjected to a uniformly distributed pressure on the top surface with motion constrained to zero along the normal of the remaining surfaces. In this test, the center most slave node in the shaded block interacts with 3 master surfaces that are unconnected. Assuming that all blocks consist of the same material, the analytic solution is a uniform stress field associated with the axis of the applied loading.

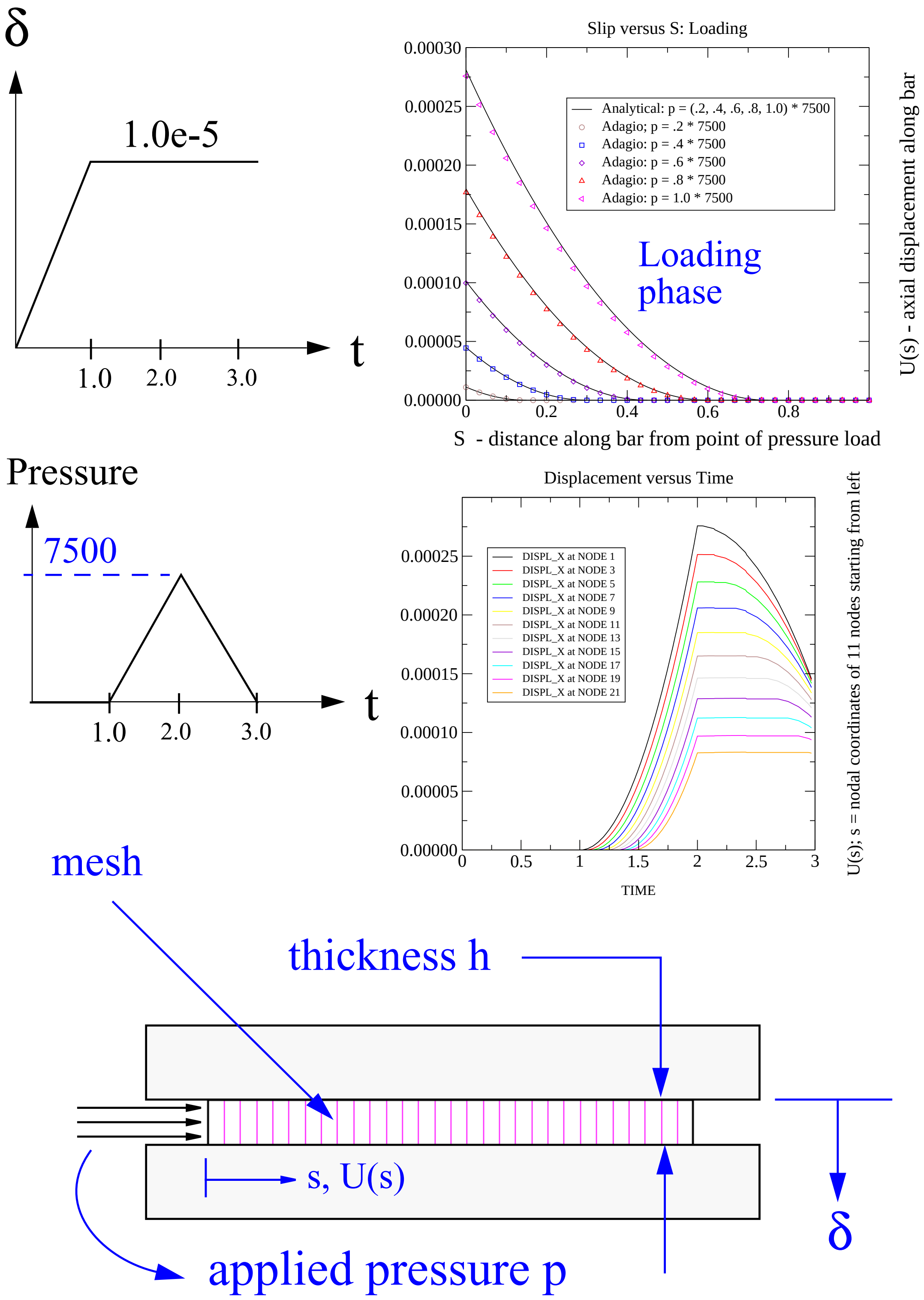
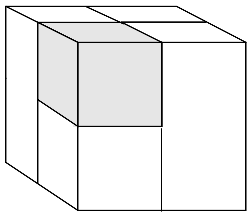


Figure 6: Test: Stick/slip bar



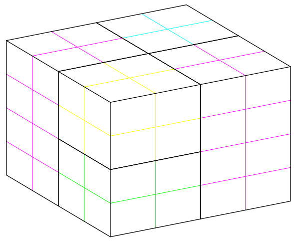


Figure 7: Test: Multiple contact constraints, contact patch test

References

[1] J. A. Mitchell, A. S. Gullerud, W. M. Scherzinger, R. Koteras, V. L. Porter, *Adagio: non-linear quasi-static structural response using the sierra framework*, in K. Bathe, ed., *First MIT Conference on Computational Fluid and Solid Mechanics*, Elsevier Science (2001).

[2] C. Edwards, J. R. Stewart, *SIERRA: A software environment for developing complex multi-physics applications*, in K. Bathe, ed., *First MIT Conference on Computational Fluid and Solid Mechanics*, Elsevier Science (2001).

[3] K. H. Brown, R. M. Summers, M. W. Glass, A. S. Gullerud, M. Heinstein, R. E. Jones, *ACME – algorithms for contact in a multiphysics environment, API version 1.0*, Tech. Rep. SAND2001-3318, Sandia National Laboratories, Albuquerque, NM (2001).

[4] K. Brown, M. Glass, A. Gullerud, M. Heinstein, R. Jones, R. Summers, *ACME, a parallel library of algorithms for contact in a multi-physics environment*, in K. Bathe, ed., *First MIT Conference on Computational Fluid and Solid Mechanics*, Elsevier Science (2001).

[5] M. Heinstein, S. Attaway, J. Swegle, F. Mello, *A general purpose contact detection algorithm for nonlinear structural analysis codes*, Tech. Rep. SAND92-2141, Sandia National Laboratories, Albuquerque, NM (1993).

[6] J. Biffle, M. Blanford, *JAC2D – a two-dimensional finite element computer program for the nonlinear quasi-static response of solids with the conjugate gradient method*, Tech. Rep. SAND93-1891, Sandia National Laboratories, Albuquerque, NM (1994).

[7] M. L. Blanford, M. Heinstein, S. W. Key, *JAS3D – a multi-strategy iterative code for solid mechanics analysis, users' instructions, release 1.6*, Tech. Rep. to be published as a SAND report, Sandia National Laboratories, Albuquerque, NM (2000).

[8] M. W. Heinstein, T. A. Laursen, *An algorithm for the matrix-free solution of quasistatic frictional contact problems*, International Journal For Numerical Methods In Engineering, 44, (1999), 1205–1226.