APPLICATION AND IMPLEMENTATION OF TRANSIENT ALGORITHMS IN COMPUTER PROGRAMS*

David J. Benson
University of California, Lawrence Livermore National Laboratory
Livermore, California

INTRODUCTION

This presentation gives a brief introduction to the nonlinear finite element programs developed at Lawrence Livermore National Laboratory by the Methods Development Group in the Mechanical Engineering Department. The four programs are DYNA3D and DYNA2D, which are explicit hydrocodes, and NIKE3D and NIKE2D, which are implicit programs. All of these programs were originally developed by John Hallquist in association with David Benson.

This presentation concentrates on DYNA3D with asides about the other programs. During the past year several new features were added to DYNA3D, and major improvements were made in the computational efficiency of the shell and beam elements. Most of these new features and improvements will eventually make their way into the other programs. Although the latest version of DYNA3D has not been released yet, it should be available well before the end of the year.

The emphasis in our computational mechanics effort has always been, and continues to be, efficiency. Although the supercomputers of today are almost unbelievably fast, a large nonlinear finite element analysis is still superior. To get the most out of our Cray supercomputers, we have vectorized the programs as much as possible. Vectorization is not enough, however, so we must always consider the efficiency of every algorithm we implement. The net result of our efficiency criterion is we are restricted to only the simplest elements and algorithms. All of our elements have linear shape functions. We use radial return instead of a subincremental method for our plasticity calculations. Our explicit codes use only reduced integration with viscous hourglass control. Our implicit programs use quasi-Newton methods to speed convergence.

In the remainder of the presentation, several of the more interesting capabilities of DYNA3D will be described and their impact on efficiency will be discussed. Some of the recent work on NIKE3D and NIKE2D will also be presented. In the belief that a single example is worth a thousand equations, we are skipping the theory entirely and going directly to the examples.

---

*Work performed under the auspices of the U.S. Department of Energy by the Lawrence Livermore National Laboratory under contract number W-7405-Eng-48.

ELEMENTS

DYNA3D has three elements: an eight node hexahedron, a four node shell and a two node beam. The shell and beam elements are based on the formulations of Hughes and Liu (ref. 1). All of the elements except the beam use reduced integration with viscous hourglass control. Large strains and large deformations are assumed for all of the elements. On the Cray-1, the elements require about thirty-five microseconds per integration point for a simple constitutive model, such as the standard J_2 plasticity model with isotropic and kinematic hardening. The shell and beam models were only recently vectorized. The original implementation of the shell element required 16000 microseconds per element, which made it unusable. Vectorization alone does not account for the dramatic increase in the speed of the shell element.

Suzuki provided us with the structural data for a frame member of a car chassis along with their experimental results from a 30km/hr impact into a barrier.

Our simulation of their experiment used a mesh of 1600 shell elements. The frame member is tied at each end to a rigid body. One rigid body represents the barrier, and the other represents the sled which provided the momentum to crush the frame. The constitutive model is the usual J_2 model with isotropic, linear strain hardening. Unfortunately, the experimental stress-strain data indicate that the material does not strain harden linearly, and we believe that much of the error we see in our simulation is the result of the linear strain hardening. We are going to modify the material model to take into account the nonlinearity and rerun the analysis in the near future.

The experiment lasts thirty-five milliseconds. On the Cray-XMP/48, DYNA3D uses a little over four hours of CPU to simulate the entire event. The peak deceleration, an important number to chassis designers, which occurs at only five milliseconds, can be calculated in less than half an hour of CPU.

The results of our analysis matched the peak deceleration almost exactly, but the duration of the peak was too short. We believe that the discrepancy was caused by either the previously mentioned simplification of the material model or the 2000Hz filter that Suzuki used on their data.

Based on good accuracy of our results and their reasonable cost, we believe that finite element analysis should no longer be regarded as strictly a research tool in crashworthiness design, but as a tool for the designer.

CONTACT, IMPACT, AND FRICTION

The contact and impact algorithms have long been among the strongest points of DYNA3D. The penalty approach is used in both the two and three dimensional versions of DYNA. A distributed parameter approach is also available in DYNA2D based on the algorithms developed by others for HEMP, TENSOR, and TOODY. Aside from the obvious simplicity of the penalty approach in comparison with the distributed parameter approach, the major advantages of the penalty method are that it is symmetric and that it does not excite hourglassing modes as much as the distributed parameter approaches. The surface stiffness for the penalty method is automatically calculated based on the

material properties, instead of being input by the user, which we believe accounts for the excellent reliability of the method.

We have two fundamentally different algorithms for surface contact. The original one assumes there are two different surfaces which may come into contact. They are designated the master and slave surfaces. The limitation to this approach is a surface cannot buckle and collapse onto itself. Our second method eliminates this problem, but it is slower. Most of the CPU time in the algorithms is used in the search for the regions in contact, and we have not been able to vectorize this section of code to any significant extent.

The Coulomb friction model blends the transition from the static to the dynamic coefficient of friction with an exponential decay based on the relative velocities of the contact surfaces. Several calculations performed with DYNA2D show good agreement with experimental results using this model.

EXAMPLE: Metal forming

Shearbanding was studied in this analysis. The problem is neither planar nor axisymmetric, and therefore had to be analyzed in three dimensions. A cylindrical blank of 304L stainless steel is high energy rate forged (HERF) at 1850F with shearbanding resulting in one plane. The initial velocity of the ram is 600cm/sec.

The die is modeled as a rigid body in the analysis so that a very fine mesh can be used to define the curved surfaces of the die without incurring a computational penalty from either the large number of elements or the Courant stability limit on the integration time step. The ram is also modeled as a rigid body with enough mass to give it the proper momentum. The cylindrical blank was modeled with J_2 elastoplasticity.

The analysis was run both with and without friction between the blank and the walls of the die. Shearbanding only occurred when friction was included. The contours of plastic strain correlate quite well in a qualitative way with the shearbands of the acid-etched forging.

Roughly 5 CPU hours on the Cray-1 were needed for the calculation. Higher ram velocities would require proportionally less CPU time, and lower velocities would require longer.

EXAMPLE: Pipe whip

The damage caused by one pipe hitting another is an area of interest to the nuclear power industry. In this example, a free segment of pipe collides with another section of pipe fixed rigidly at both ends. This model uses shell elements with the two surface contact algorithm. It runs in only four minutes on the Cray-1.

EXAMPLE: Axial buckling of a cylinder

This example demonstrates the use of our single surface contact algorithm for problems with buckling. The analyst does not know a priori where the cylinder will fold and therefore cannot divide the surface into a series of master and slave segments. Two contact surfaces were defined, one being the

exterior surface, the other, the interior experience. The length of the cylinder is 440mm, the diameter is 100mm, and the wall thickness is 1.5mm. The mesh consists of 1980 shell elements with five integration points through the thickness. A little under thirteen CPU hours on the Cray were used in this analysis, most of which was used in the contact routine.

RIGID-BODY DYNAMICS

A recent addition to DYNA3D is material type number 20, the rigid-body material. In many crash analyses, the plastic deformation is localized to a rather small region, but the entire structure must be modeled in order to include the correct amount of momentum and inertia in the calculation. To reduce the cost of such a calculation, we replace the regions far from the impact with rigid bodies. Rigid bodies cost only one microsecond per zone, which, when compared to a constitutive evaluation, is almost free. The fact that a rigid body is defined by a material type makes this feature almost transparent to the analyst. Regions of a structure are easily frozen by switching them to material type 20. Several materials are easily merged into a single body by adding merge cards to the data file. All of the contact algorithms and most of the boundary conditions worked with rigid bodies with only minor changes to the code. In addition to the standard boundary conditions, we have implemented joint constraints. DYNA3D is probably the only hydrocode that has universal and ball joint models in it.

EXAMPLE: Earth penetrator

This calculation was run to determine the effect of a collision with a tree trunk on the trajectory of an Earth penetrator. The original calculation, which took four CPU hours on a Cray, modeled the tree trunk with an elasto-plastic material and the penetrator was elastic. We replaced the elastic material model with the rigid-body material and reduced the cost to six CPU minutes. The large reduction in cost resulted not only from the large reduction in the number of constitutive evaluations, but also from the Courant stability limit. In the original calculation, stability was determined by a small element in the projectile, but with the rigid-body projectile, stability was determined by the comparatively coarse zoning of the tree trunk. The results of the two calculations agreed almost exactly.

EXAMPLE: Cylinder impact

This test was run several years ago by Sandia National Laboratories. A steel cylinder was gripped "rigidly" at both ends by an apparatus that slammed it into a steel rail at a velocity of 1676cm/sec. The number of interest is the depth of the dent in the side of the cylinder, which is known to be 3.64cm from the experiment. This problem was one of the first successes for DYNA3D. The original calculation used an elasto-plastic model for the cylinder. An extremely dense, elastic material was used for the two rings representing the apparatus gripping the ends of the cylinder. Both the cylinder and rings were modeled with solid elements. Fifty-five CPU minutes are required for the original model, which predicts a dent 3.886cm in depth. The cost of the analysis drops to thirty-one minutes if the rings are made into rigid bodies, but the dent is only 3.048 cm. Our conclusion is that the apparatus was not nearly as stiff as everyone had assumed. We recently ran the problem modeled

with shell elements, and it took only sixteen minutes; however the deformation was too large.

INTERACTIVE REZONING

Interactive rezoning has been available for several years in DYNA2D and was recently added to NIKE2D. Rezoning allows the user to eliminate or smooth sections of a mesh with thin or highly distorted elements. This increases the computational efficiency of the programs by allowing a larger time step in DYNA2D and by improving the convergence rate in NIKE2D.

This NIKE2D example shows a thick-walled cup being formed by back extrusion. The mesh was rezoned several times during the analysis. Only a few minutes of CPU was required. The same analysis with DYNA2D would take several hours because of the low speed of the forming process.

ITERATIVE SOLUTION OF EQUATIONS

One area in which we are currently supporting research is the iterative solution of linear algebraic equations. The cost of factoring a cube with N elements on each edge is proportional to N^7. An iterative solution method is faster than a direct solution for even fairly small problems of this type. For example, with eight elements on an edge, a direct solution takes 1.08 minutes of CPU while the iterative solution takes .53 minutes. With twenty-four elements on an edge, the direct solution takes 3380 seconds for the 46875 equations as compared to the 125 seconds for the iterative method.

Improvements to the element-by-element preconditioner, developed by Hughes, for the conjugate gradient method are being developed (Robert Ferencz, Lawrence Livermore National Laboratory, unpublished data). The major difficulty with iterative methods is their lack of robustness -- problems that have a wide range of eigenvalues, caused, for example, by structural elements or a penalty contact algorithm, converge slowly (if at all) with these methods. The goal of this research is the development of a preconditioner that will improve the robustness of the conjugate gradient method.

In this example, a bar hitting a rigid wall is modeled with 2700 solid elements and two planes of symmetry, giving 9196 equations. The direct method requires a little over 2.5 million words of storage, while the iterative method requires 1.7 million words. The solution cost with the direct method required 2249 seconds on the Cray XMP with 18 seconds of I/O to the solid state disk. With a standard disk, almost 1800 seconds of I/O would be required. In contrast, the element-by-element method required only 654 seconds of CPU and solved the problem without using the disk.

Although the range of eigenvalues for this problem is not as large as for a problem with beam elements, the problem is elastoplastic. This problem demonstrates that iterative solution methods are improving in robustness. As iterative solution methods improve and larger computers become available, many problems that would now be solved using explicit finite-element programs out

of necessity will be solved more efficiently by implicit programs in the future.

FRACTURE AND FAILURE

Another area of research for us is fracture. Last year we implemented a tie-breaking slideline in our version of NIKE2D based on modifications to the program at South Carolina. The program was used to study the formation of chips in machining processes. We have also installed nodal constraints based on the same ideas in DYNA3D to study the petalling of sheet metal as projectiles penetrate it. In both cases, plastic strain is the fracture criterion. A smeared crack fracture model is also available in DYNA3D. An element fails when the maximum principal stresses exceed a specified fracture stress with this model. Our work in this area is very preliminary; we have concentrated more on the methods of implementing failure criteria efficiently rather than a sophisticated fracture criteria. As we gain experience and experimental results, we hope to improve the fracture criteria.

The computational overhead associated with the failure models is small. The smeared crack material model is only about twenty percent more expensive than our standard elasto-plastic material model, and the tie-breaking nodal constraints are completely vectorized.

A steel plate, .1 inches thick, is struck by a 3 inch diameter rigid sphere with a velocity of 6000in/sec. The data were chosen rather arbitrarily, with the only goal of the problem to see whether or not petalling would occur. We plan to run better calculations in the future and compare them to experimental results.

CONCLUSIONS

Given the increasing size and speed of computers and the increasing efficiency and robustness of finite-element algorithms, we believe that problems regarded by most as impossible today will be possible, if expensive, tomorrow. On the Cray-2, which is the technology of today, a multi-tasked version of DYNA3D could solve problems with more than a million zones and ten thousand time steps in less than ten hours.

REFERENCE

1. T. J. R. Hughes and W. K. Liu: Nonlinear Finite Element Analysis of Shells: Part I. Three-Dimensional Shells. J. Comp. Meth. Appl. Mech., vol. 27, 1981, pp. 331-362.

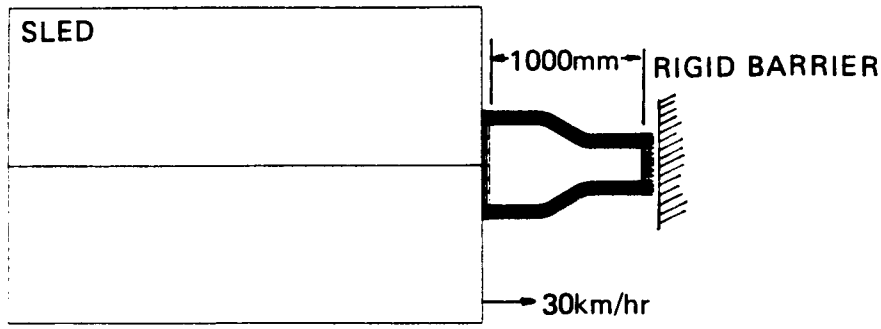


Figure 1. Finite-element model of Suzuki sled test.

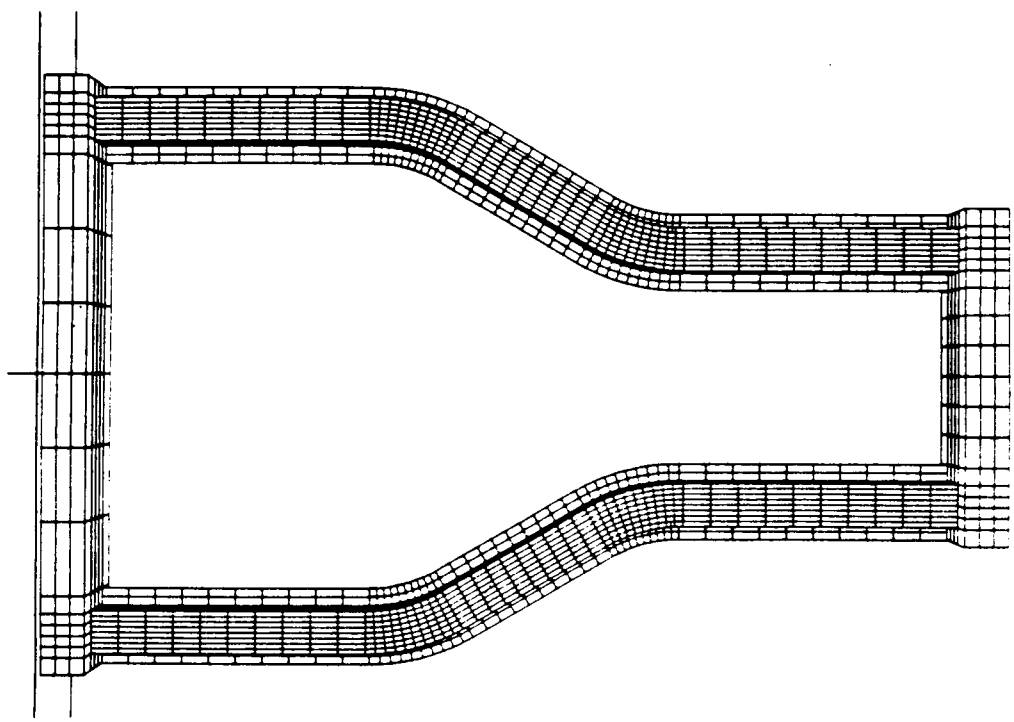


Figure 2. Close-up view of the finite-element mesh. The mesh contains 1600 shell elements.

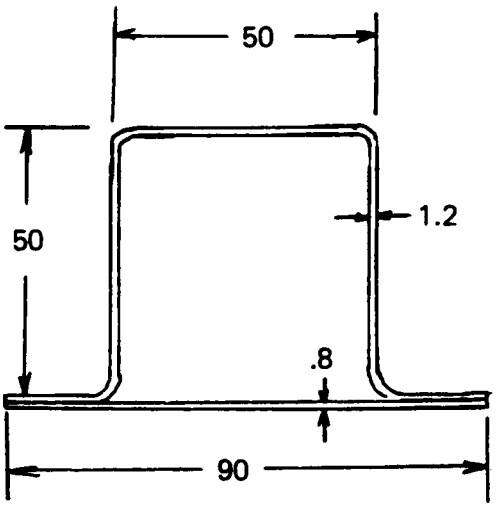


Figure 3. Cross section of frame member. All dimensions are in millimeters.

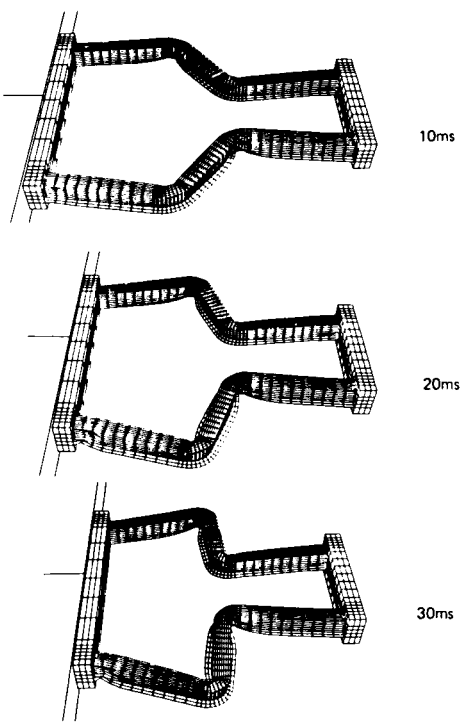
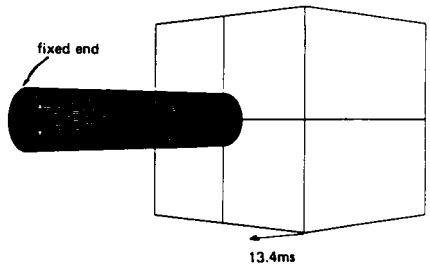
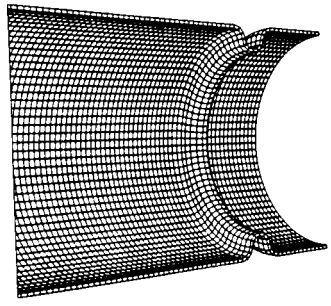


Figure 4. Deformed shapes at 10 ms output intervals.



(a) Initial configuration of cylinder.



(b) Close-up view showing cross section: One quarter of the cylinder and mass were modeled with symmetric boundary conditions.

Figure 5. Axial buckling of cylinder.

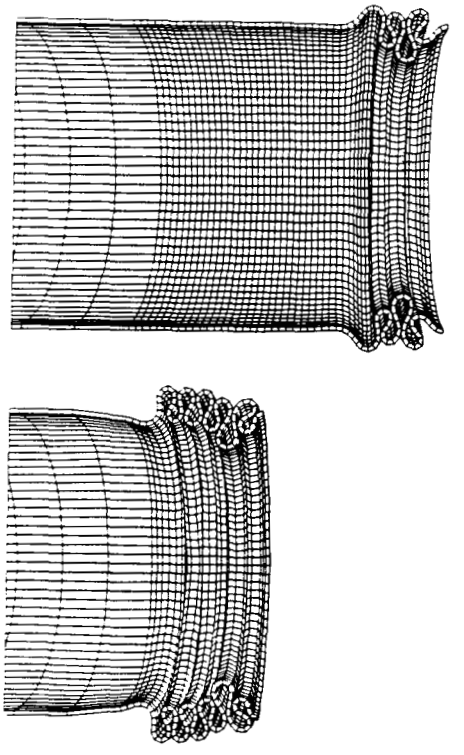


Figure 6. Deformed cross sections at 5 and 10 ms.

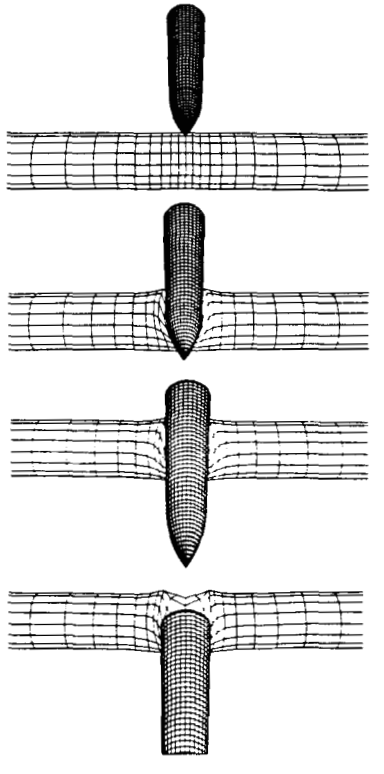


Figure 7. Earth penetrator.

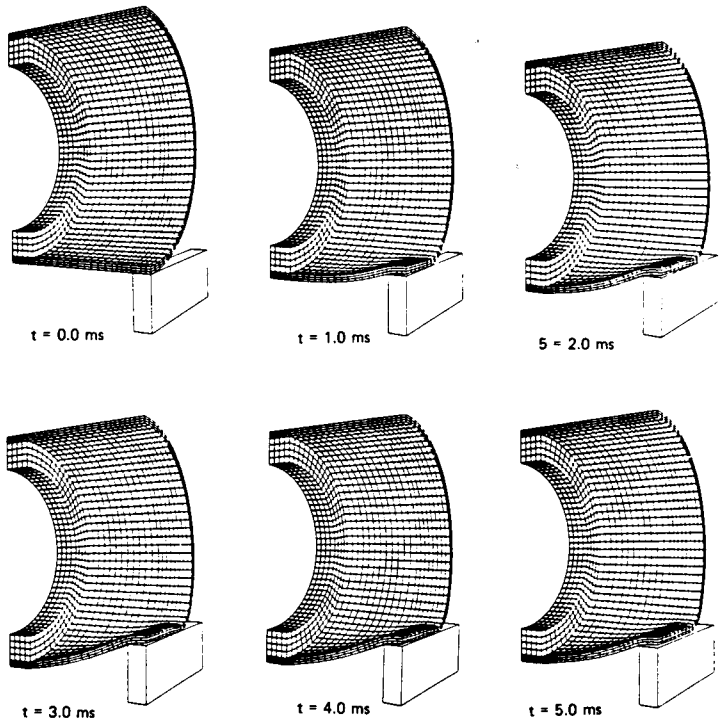


Figure 8. Cylinder impact.

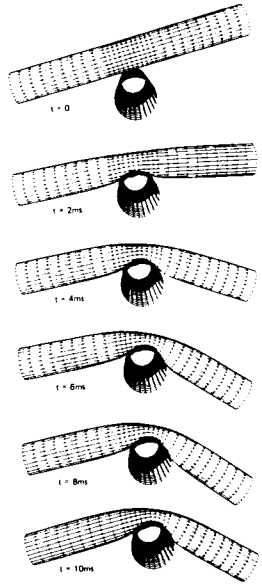


Figure 9. Sequence of deformed shapes in pipe wip analysis.

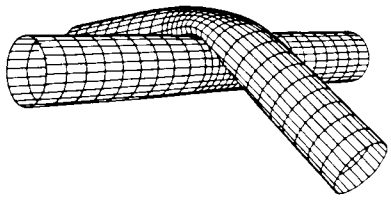


Figure 10. Rotated view of final configuration.

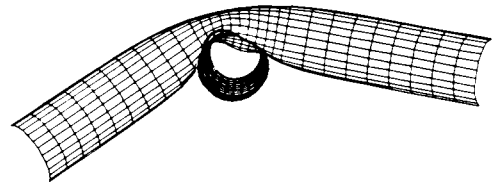


Figure 11. Deformed cross section.

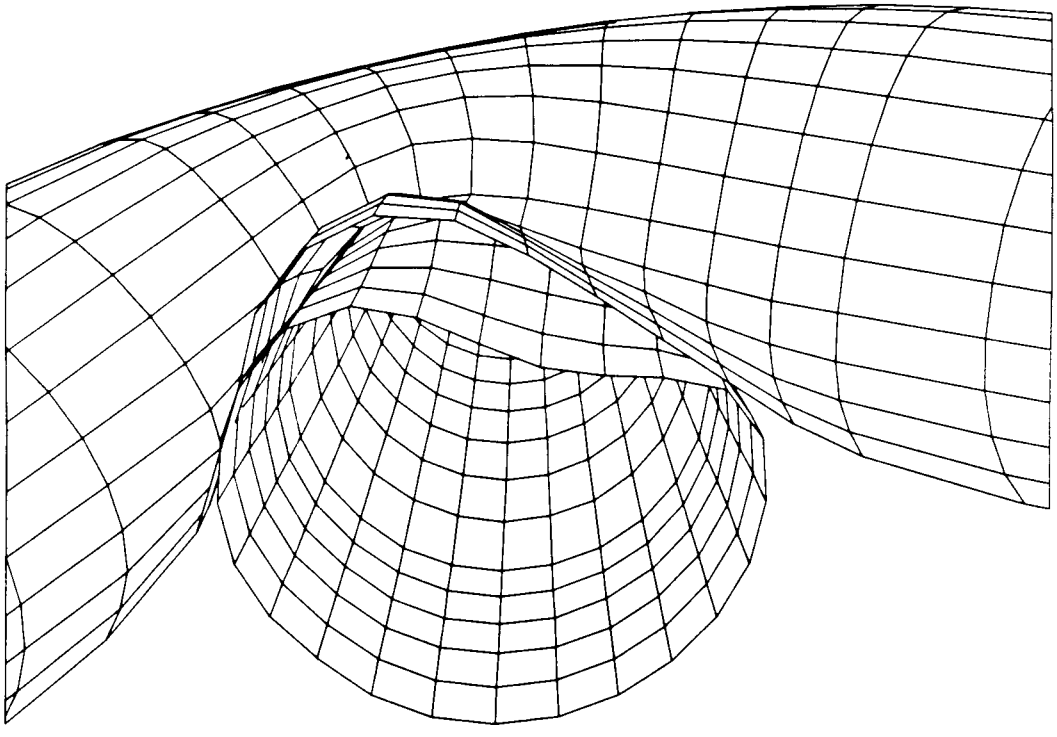


Figure 12. Close-up of deformed cross section.