ADVANCES IN COMPUTING FOR PROBLEMS INVOLVING INTENSE IMPULSIVE LOADING

Gordon R. Johnson

Network Computing Services, Inc.
1200 Washington Ave. S.
Minneapolis, MN 55415
gordon.johnson@netaspx.com

This paper presents the author's perspective on changes that have occurred in one area of DoD computing during the past 35 years. The specific class of problems to be addressed is that of intense impulsive loading, which often results from high-velocity impact or explosive detonation. Although there are many DoD applications for Computational Solid Mechanics (CMS), the area of intense impulsive loading has many unique features. Often these problems involve large strains, high strain rates, high temperatures, high pressures and failure. These conditions require complex material models, accurate sliding/contact interfaces and a general robustness to handle the large distortions that often occur.

While it is clear to all that there have been significant advances in computer hardware during the past several decades, there have also been corresponding changes in the computing environment. In the early days preprocessors and postprocessors were very basic, and the analyst would focus on simply getting the problem to run and to understand the results. Although the hardware and software were very limited compared to today's standards, the intense effort required to run these problems often resulted in a clear understanding of the dynamic response. In contrast, now that many of the codes have been made very easy to use, it is possible for the analyst to generate better results, but perhaps not achieve a better understanding.

This paper will look at the transition of finite differences to finite elements, and more recently to meshless particle methods. Also, the development of more accurate and robust interface algorithms has done much to increase the range of problems that can be addressed today. The history of material model development will also be discussed. The early models consisted of a dynamic flow stress, and now there are complex models available which can require up to a hundred constants.

The significant rise in the complexity of the current numerical algorithms will require changes in how software is developed for the codes of the future. As advances continue to be made, researchers will continue to produce technical publications, but they will also be required to deliver modular software that is accurate, efficient, robust and well documented.