A CRAY RESEARCH, INC. PUBLICATION

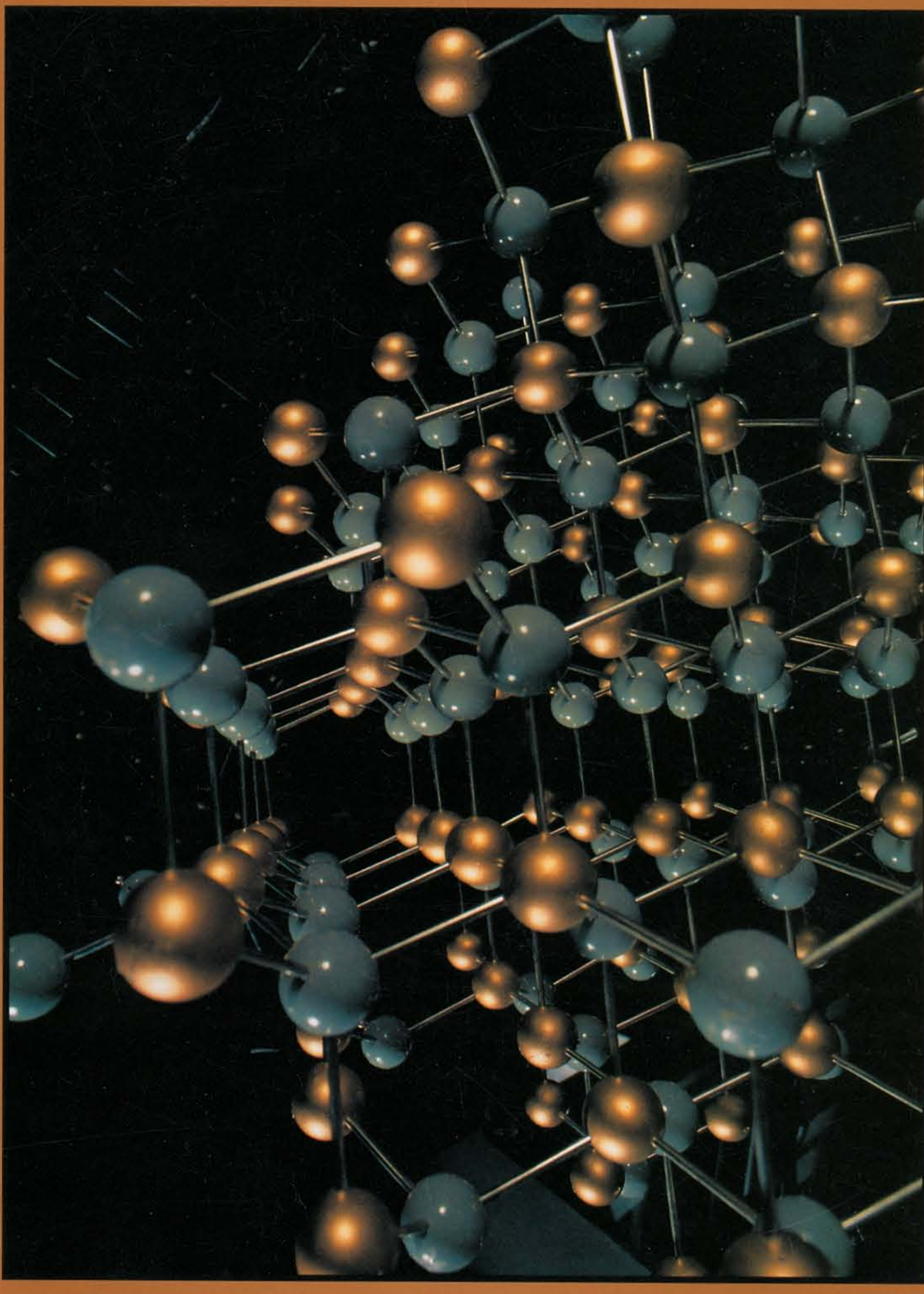
CRAY CHANNELS

Summer 1986

ANNOUNCEMENT!
Introducing the CFT77 Fortran compiler

FEATURE ARTICLES:

DEPARTMENTS:

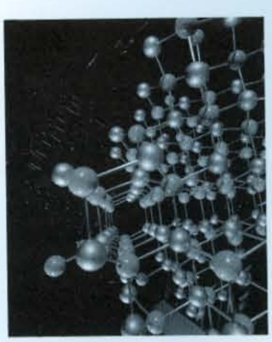


IN THIS ISSUE

Thanks to supercomputers, we've finally caught up with the nineteenth century. That's one way to look at recent developments in computational fluid dynamics (CFD), where aerodynamicists are designing modern aircraft using equations perfected in the nineteenth century. Cray computers make it practical to solve the complex equations that describe fluid flow and apply them to the design of aircraft components, and even entire aircraft. The same techniques are also being used to design aircraft propulsion systems, where the complexity of the flows involved had previously made modeling impossible. In this issue of CRAY CHANNELS we examine these and other developments in this rapidly progressing field.

In addition, we introduce new Fortran, Pascal, and C compilers. Our regular departments include reports on new CFD codes, a high-tech rock video, and electronic circuit design at AT&T Bell Laboratories.

Although trial and error brought the Wright brothers to their ultimate success, today's aircraft designers need more efficient design strategies. Numerical simulation is an approach better suited to an aerospace industry interested in everything from passenger carriers and fighter planes to spacecraft. As this methodology exerts a growing influence, Cray computer systems will be called on increasingly for their fast and efficient problem solving capabilities

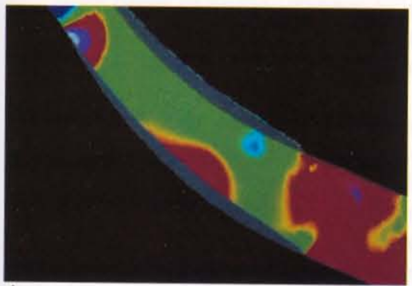


On the cover is a physical model of a gallium arsenide crystal. Because of its special electrical properties, gallium arsenide technology is being developed for use in future Cray computer systems. The material provides computer designers the opportunity to house more and faster processors and larger memories in smaller frames. Such miniaturization allows for shorter component interconnections, further improving a system's overall performance.

CRAY CHANNELS

A Cray Research, Inc. publication

Summer 1986



6

18

FEATURES

DEPARTMENTS

CRAY CHANNELS is a quarterly publication of Cray Research, Inc. intended for users of Cray computer systems and others interested in the company and its products. Please mail subscription requests, feature story ideas and news items to CRAY CHANNELS at Cray Research, Inc., 608 Second Avenue South, Minneapolis, MN 55402

Volume 8, Number 2
Summer 1986

Computational methods and aerodynamics

Tim Baker and Antony Jameson, Princeton University, Princeton, New Jersey

Aircraft design relies heavily on an understanding of aerodynamics. Solving the mathematical equations that govern air flow poses great difficulties, however, and the important insights of aerodynamics generally have been achieved by making judicious simplifications. Computational methods now offer the opportunity to remove most such simplifications and to extend the scope of aerodynamic predictions over a greater range of situations.

The challenge defined

In the aircraft industry there is often a very narrow margin between success and failure. In the past two decades the development of new commercial aircraft successful enough to make a profit for their manufacturer has often been a fickle and elusive goal. The economics of aircraft operation are such that even a small improvement in efficiency can translate into substantial savings in operational costs. Therefore, the operating efficiency of an airplane is a major consideration of potential buyers. This provides manufacturers with a compelling incentive to design more efficient aircraft. Three main opportunities exist for improving operating efficiency. They are: the use of composite materials, the improvement of engine efficiency, and the improvement of aerodynamic efficiency. Following is a description of the challenges involved in, and a promising approach to, addressing the third area—improving aerodynamic efficiency.

The physical process

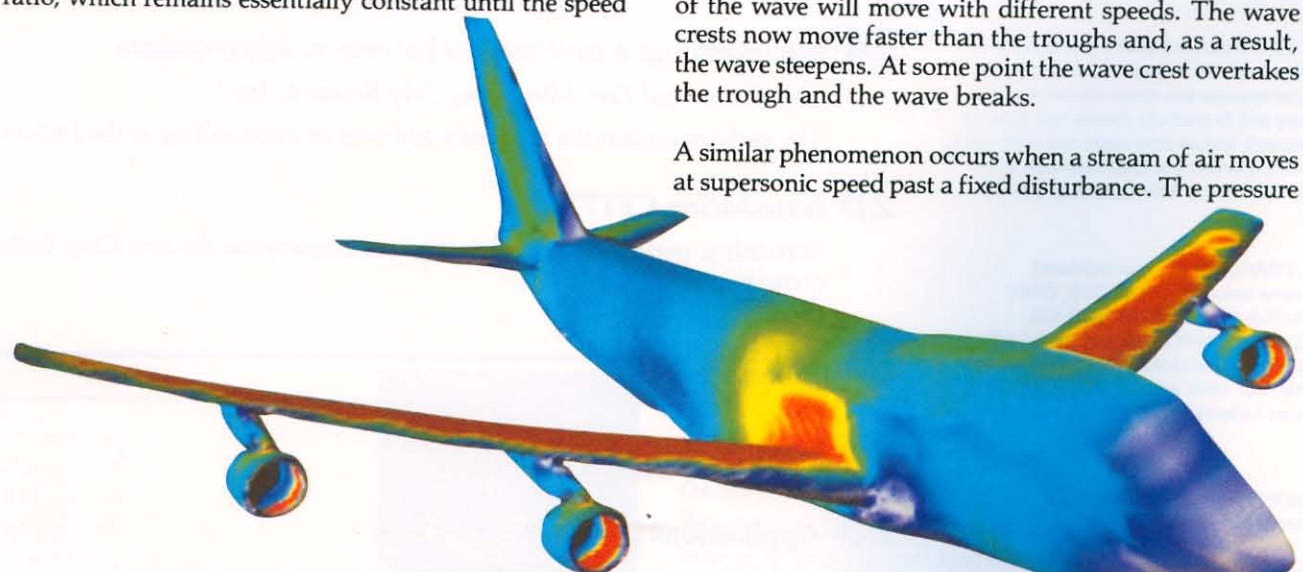
Aerodynamic efficiency is proportional to the lift-to-drag ratio, which remains essentially constant until the speed of the oncoming air is near the speed of sound. The ratio of flow speed to sound speed is the *Mach number*. At the critical Mach number value corresponding to rapid drag rise, the lift-to-drag ratio suddenly decreases with a corresponding decrease in aerodynamic efficiency.

The loss of aerodynamic efficiency occurs because the flow over the aircraft becomes locally supersonic. Although the major part of the flowfield is subsonic, it contains pockets of supersonic flow on the wings and other parts of the aircraft. The transition from supersonic to subsonic flow almost always occurs through a *shock wave*, a condition that gives rise to a sudden deceleration and increase in pressure.

Dissipative effects inside a shock wave produce a resistive force. This force, known as *wave drag*, causes the rapid loss in efficiency. Although wave drag is not the only source of aerodynamic resistance, it effectively limits the speed at which an aircraft can fly efficiently. Indeed, the main improvements in the aerodynamic efficiency of commercial aircraft over the last 25 years have been a direct result of raising the drag rise Mach number to achieve a high cruising speed without incurring any extra drag penalty. Because drag rise occurs when the flowfield contains regions of both subsonic and supersonic flow, it is not surprising that this so-called transonic regime is of particular interest and importance.

The appearance of a shock wave can be compared with the breaking of a wave on the shore. When the wave height becomes comparable with the water depth, different parts of the wave will move with different speeds. The wave crests now move faster than the troughs and, as a result, the wave steepens. At some point the wave crest overtakes the trough and the wave breaks.

A similar phenomenon occurs when a stream of air moves at supersonic speed past a fixed disturbance. The pressure



Color-coded pressure contours over the Boeing 747-200 calculated by the authors' new transonic flow code, AIRPLANE. Blue represents high pressure and low velocity. Red represents low pressure and high velocity.

wave created by the disturbance will travel downstream at a speed equal to the sum of the flow speed and the speed of sound. The sound speed varies with the flow speed, but if the disturbance is weak (that is, just a small perturbation about the mean flow) the wave will travel essentially undistorted like the small-amplitude waves on shallow water. For a strong disturbance there will be a steepening effect, because the speed of sound in the faster moving air is greater than that in the slower moving air. Air is very slightly viscous, so that eventually the extreme velocity gradient in a steepened wave generates significant viscous stresses that counteract the steepening process and prevent further steepening. A shock wave thus represents a balance between the steepening effect due to flow nonlinearity and the weakening effect of viscosity.

As the Mach number approaches zero, the flow may be regarded as incompressible or, put another way, as having infinite signal speed. If one assumes that the flow is incompressible and furthermore considers a flow that is steady, inviscid, and at a uniform speed far from the aircraft, then the mathematical equations that describe the flow are enormously simplified. In particular, the governing equation is now linear, which means that two or more elementary solutions can be added together to give a more complicated solution still satisfying the governing equation. Thus, it is possible to represent the incompressible flowfield around a complicated shape as a weighted sum of elementary solutions, where the weights are chosen such that there is no flow across any solid surface. This is the approach used in *panel methods*, which were developed during the 1960s and early 1970s and are now widely used by the aerodynamic departments of aircraft companies for low-speed predictions.

If the flow is almost entirely supersonic, which is typical for slender aircraft traveling at very high speeds, then the governing flow equations can again be reduced to a single linear equation. Once again, it is possible to develop a solution for a complicated shape by adding together a number of elementary solutions. The character of supersonic flow, however, is quite different from subsonic flow. A disturbance will travel at the speed of sound; and if the oncoming flow is subsonic, this disturbance can move upstream and hence spread in all directions. In a supersonic flow, a disturbance cannot move upstream because the oncoming flow is moving faster than the speed of sound. The region of influence of the disturbance is therefore restricted to a cone whose vertex is at the point from which the disturbance emanates. Moreover, supersonic flow has a wavelike character and hence even small changes in surface shape can create waves that travel far from the aircraft body. In subsonic flow, small disturbances usually attenuate quickly, even though they move in all directions.

The importance of nonlinearity

Transonic flow is characterized by the presence of both subsonic and supersonic regions of flow and the formation of shock waves. Because of its essentially nonlinear behavior, a complete solution cannot be obtained by adding together elementary solutions. As in many other fields of physics, nonlinearity gives rise to interesting and remarkable features. Unfortunately, although a highly developed body of theory for dealing with linear problems exists, mathematicians can muster only a relatively meager selection of ad hoc methods to deal with nonlinearities.

The importance of flow nonlinearity prevents omission of the nonlinear terms from the governing equations, and makes a numerical solution of the problem unavoidable. Furthermore, even though viscosity may be negligible for much of the flow, it can have a significant effect on the global flow pattern. Aside from its appearance as part of the shock wave mechanism, viscosity assumes a dominant role in the boundary layer. This usually thin layer occurs along solid surfaces and arises because a fluid (in this case air) always sticks to a solid surface. Because the fluid immediately adjacent to a solid surface does not slip past, the streamwise velocity increases rapidly as one moves away from the surface. The boundary layer on a wing, for example, is usually thin compared with the dimensions of the wing, and at a cruise condition it will remain attached until the trailing edge, where it leaves forming a thin wake. Under more extreme conditions, such as setting the wing at a moderately large angle of attack, the boundary layer will separate, causing a radical change in the flow including a sudden loss of lift leading to a stall.

The computational challenge

In principle, it is possible to solve numerically the full equations of fluid flow including viscous effects. In practice, considerable difficulties exist in representing the viscous terms accurately and in modeling turbulence. There are sound reasons for examining the inviscid equations of fluid flow separately. First, the boundary layer for an aircraft at cruise should have a relatively small effect on the overall flow. The drag component due to viscous effects does not vary significantly with increasing speed and the rapid rise in drag near the speed of sound is caused by the appearance of shock waves. Although viscosity affects the internal structure of a shock wave, the change in flow properties through a shock wave and the wave drag generated are not dependent on the amount of viscosity. It is thus possible to regard a shock wave as an inviscid flow phenomenon and to calculate transonic flow containing shock waves by solving the inviscid flow equations. For the design of aircraft at cruise, inviscid flow calculations can therefore provide a great deal of useful information.

Second, many of the concerns in the development of a numerical method, such as mesh generation for complex shapes and algorithm efficiency, carry over with little change from inviscid to viscous flow calculations. A reliable and accurate inviscid flow code is therefore an excellent stepping-stone for the development of a fully viscous method.

How then does one set about solving the inviscid flow equations? Stated in the briefest terms, the recipe sounds astoundingly simple: place a network of cells around the aircraft, construct a system of equations that represents the conservation of mass, momentum, and energy for each cell, then solve the equations by an iterative technique until the fluxes of mass, momentum, and energy for each cell are exactly conserved. In developing a computational method, one is therefore concerned with mesh generation, algorithm efficiency, and the accuracy of the resulting numerical solution. The three aspects impinge on each other. Too coarse a density of mesh cells or a low-order discretization of the flow equations can result in a loss of solution accuracy. A rapid change in cell size or a high-order discretization can adversely affect algorithm efficiency, and so on.

The first flow codes were developed for simple geometric shapes, in particular such two-dimensional shapes as airfoils and engine inlet sections. These are still useful examples for testing new algorithms and concepts in numerical analysis. Indeed, algorithm developers have made significant strides over the past few years. Methods that used to take several thousand cycles to reach a dubious steady state have now been replaced by algorithms capable of reaching a converged result after 20 or so cycles.

Toward greater efficiency

The most efficient of today's algorithms are based on the *multigrid technique*. With this technique, information is spread rapidly around the flowfield using a sequence of coarser meshes to enhance the evolving solution on the original set of mesh cells. Like many ideas, multigrid has been around for some time, but its successful application has required considerable effort and extensive research into the complexities of the algorithm mechanism.

A specific concern about solution accuracy relates to the captured shock wave. To obtain a numerical solution of transonic flow that contains a shock wave, one must introduce some artificial dissipation to prevent uncontrolled oscillations in the neighborhood of the shock wave. Artificial dissipation has an effect similar to that of viscosity in a real fluid and effectively smears the flow discontinuity over a few mesh cells. Insufficient dissipation will lead to numerical instability, whereas too much dissipation will generate a badly smeared shock and an inaccurate solution. How to introduce artificial dissipation in the most effective manner has been an area of intense exploration in its own right. Fairly sophisticated techniques are now available to inject just the right amount of dissipation to capture a shock wave over no more than two mesh cells without causing any pre- or post-shock oscillations.

Tailoring the mesh

Mesh generation is another area that poses particular difficulties. For example, to model an airfoil one must create a network of cells that covers an extensive region of the surrounding space. To keep the number of mesh cells manageable, the cells should be densest where rapid flow changes occur (such as near the airfoil surface) and least dense where one would expect smaller flow variations. The flow is assumed to be at its freesteam condition at the outer boundary of the mesh. A recent development is the use of adaptive meshing procedures that allow the mesh to evolve with the solution by monitoring some characteristic of the solution and refining the mesh where necessary.

A desirable constraint on the mesh is the requirement that the airfoil surface coincide with cell sides. This facilitates the application of solid surface boundary conditions that would be very complicated to apply if the airfoil profile passed through mesh cells. An obvious choice is a set of rectilinear cells, where the mesh points defining the cells are generated by the intersection of two families of coordinate lines. Typically, one family would correspond to the airfoil profile at one extreme and the outer boundary at the other extreme, with the second family of lines connecting the airfoil to the outer boundary. This imposes two natural directions on the mesh corresponding to the two families of coordinate lines. Many of the solution algorithms that have been developed exploit this structure.

The application of mesh generation techniques to more complicated geometries becomes increasingly difficult when one tackles three-dimensional problems. Considerable ingenuity is required to form a network of cells that are not too distorted and yet meet all the conditions previously mentioned. Successful mesh generation methods that use cube-like cells have been developed for wing-body combinations and for a combination of wing, body, and tail. However, it becomes increasingly difficult to keep boundary surfaces aligned with cell faces when one uses a regular structure of rectilinear cells. This difficulty has hindered the development of flowfield computational methods to treat a complete aircraft including engine nacelles and struts.

A novel approach

An alternative to the mesh formed by an array of rectilinear cells is the use of triangular elements in two dimensions or tetrahedra in three dimensions. With this cell type there is no longer any need to retain structure in the mesh. Indeed the lack of any natural coordinate direction or need for structure becomes a virtue because it is always possible to connect a set of points to form a covering of triangles in two dimensions or tetrahedra in three dimensions. The authors have explored this alternative approach and have recently developed a method for calculating inviscid transonic flow over a complete aircraft based on an unstructured mesh of tetrahedra.

Development of the computer code was carried out using the CRAY X-MP/48 and CRAY X-MP/216 computer systems at Cray Research's Mendota Heights, Minnesota, facility. The present version requires eight million words and takes about one hour to run. Approximately one-third of this time is consumed by the triangulation procedure that automatically connects the points to form the tetrahedra. The remaining time is taken up by the solution algorithm that calculates the flowfield. The present mesh contains 20,000 points and is made up of 100,000 tetrahedra. This is not sufficient for an accurate solution and realistic calculations will require many more points than are now used. We, the authors, expect that the memory requirement for an acceptably accurate result will be about 50 million words—a computation that is certainly feasible on the CRAY-2 computer system or a CRAY X-MP computer system with a large SSD storage device.

Our method, which is based on an unstructured mesh of cells, differs from most fluid dynamics codes currently in use or under development, and is more closely related to finite element methods used in structural analysis. It poses a variety of novel problems, both in the triangulation process and in the design of a suitable algorithm. Methods based on a structured network of rectilinear cells usually lend themselves readily to vectorization. For example, the wing and the wing-body-tail codes developed by the authors can sustain a rate of about 70 million floating-point operations per second (MFLOPS) on a CRAY X-MP computer system. For the unstructured aircraft code the MFLOP rate drops to about one-tenth of this speed unless steps are taken to achieve a vectorizable algorithm.

Suppose that the cells are labled by the parameter

L = 1,NCELL.

In two dimensions each cell is a triangle with three vertices. Each vertex has a position defined by its x and y coordinates. Let the nth point have coordinates $x(n,1)$ and $x(n,2)$. A typical part of the flow algorithm might require a loop over the cells in which the coordinates of the cell vertices are required in some further computation. Thus, we may suppose that we have an array, say NDC(L,I) where L refers to the cell number and I = 1,2,3, refers to the three vertices. Thus, a typical loop might have the form

```
      DO 100 L = 1, NCELL
      N1 = NDC(L,1)
      N2 = NDC(L,2)
      N3 = NDC(L,3)
      X1 = X(N1,1)
      Y1 = X(N1,2)
      X2 = X(N2,1)
      Y2 = X(N2,2)
      X3 = X(N3,1)
      Y3 = X(N3,2)
       .
       .
       .
100   CONTINUE
```

The positions of the triangle vertices are thus defined by the coordinate pairs $(X1,Y1)$, $(X2,Y2)$, and $(X3,Y3)$. However, the use of indirect addressing mandated by the unstructured nature of the mesh means that each point will appear at least three times because it will be referenced as a vertex of at least three different triangles. The possibility of a vector dependency will inhibit vectorization by either Cray Fortran compiler. However, if one first sorts the cells into groups so that no vertex is referenced more than once in each group, one can override the compiler and force vectorization confident that vector dependency will not occur. Hardware gather and scatter operations then ensure that the vectorized algorithm for an unstructured mesh will have a processing rate comparable to that achieved by the traditional algorithms which do not require indirect addressing. The loop now takes the form

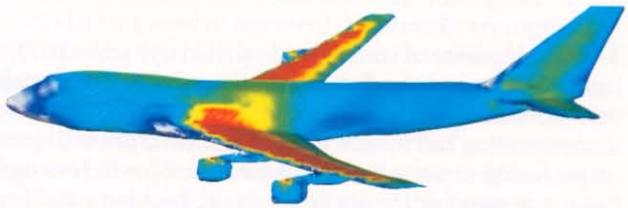
```
      DO 110 K = 1, KGRP
      L1 = LGRP(K)
      L2 = LGRP(K+1)-1
      DO 100 L = L1,L2
       .
       .
       .
100   CONTINUE
110   CONTINUE
```

The sorting of a triangulated region into disjoint groups of triangles such that no vertex occurs more than once in each group is reminiscent of map coloring problems. One might therefore expect that ingenious sorting methods would generate the smallest number of possible groupings. A naive sorting algorithm is already in use and can achieve a fivefold improvement over a straight scalar computation, resulting in a sustained processing rate of 20 to 40 MFLOPS. The variation in processing rate is caused by the variation in the number of cell groups, which depends on the mesh. One might expect improvements in sorting algorithms to lead to improved MFLOP rates.

Future development

Although the AIRPLANE code is still in an early stage of development, we expect that the use of unstructured meshes and the consequent need for indirect addressing will remain in future versions and will become a feature of fluid dynamics codes developed by others. There is a clear need for research into the adaptation and design of vector and parallel processing architectures to suit unstructured algorithms.

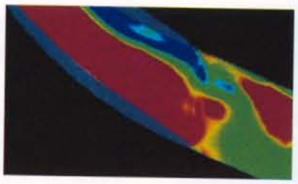
The advent of computer codes capable of calculating transonic flow around highly complicated geometries entails large amounts of memory. The CRAY X-MP computer system with a large SSD storage device and the CRAY-2 computer system are well-suited to this requirement. The introduction of viscous terms into the equations of fluid flow is the next logical step in transonic flow code development. The emergence of computers of the CRAY-2 class comes at a time when computational methods for flowfield calculation are poised to exploit their capabilities. □



About the authors

Tim Baker was born and educated in England. He received his bachelor of arts and Ph.D. degrees from Cambridge University. He first worked for the Plessey Company in Romsey, England, on problems involving communication theory and signal processing. After moving to the Aircraft Research Association in Bedford, England, he worked for several years on computational methods for calculating transonic flow. In 1982 he accepted a research appointment at Princeton University.

Antony Jameson was born and educated in England, receiving his bachelor of arts and Ph.D. degrees from Cambridge University. He then worked as an economist for the Trades Union Congress in London and as a mathematician for Hawker Siddeley Dynamics. He came to the United States in 1966 and spent six years in the aerodynamics department of the Grumman Aerospace Corporation before joining the Courant Institute of Mathematical Sciences, where he was a professor of computer science. He came to Princeton in 1980 and two years later was named the James S. McDonnell Distinguished University Professor of Aerospace Sciences.



Numerical simulation of time-dependent viscous flows in aerospace propulsion systems

James N. Scott, University of Dayton, Dayton, Ohio

The development of computational fluid dynamics (CFD) has paid tremendous dividends to researchers analyzing complex flow phenomena and continues to provide new understanding and insight into fundamental physical processes found in various flow situations. This technology has made possible the study of classic problems such as shock wave-boundary layer interaction, unsteady and separated flows, and vortex production and interaction. Furthermore, only within the past year has the technology been extended to permit the computation of the three-dimensional viscous external flow field about an entire vehicle configuration.[1]

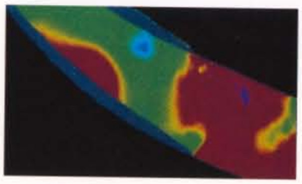
The single most important factor permitting these achievements has been the development of supercomputers. The rapid growth of core memory and computing speed has allowed order-of-magnitude increases in the number of grid points used to resolve time-dependent and three-dimensional flow phenomena. The analysis of such flows requires the numerical solution of the complete compressible time-dependent Navier-Stokes equations governing fluid motion. Most of the computations involving these equations have been performed for external flow, for which the payoff has been extremely high. While the number of successful computations for internal flows using the Navier-Stokes equations has been relatively small, indications are that the potential payoff in internal flows in propulsion systems may be much greater than for external flows. Hence, the challenge lies in extending this capability to analyze extremely complex flow situations encountered in advanced aerospace propulsion systems.

The flow in modern propulsion systems is complicated by unsteadiness and/or separation in virtually every component as a result of either forced or self-excited oscillations. In addition to the time-dependent features, the flow may include particulates through dust ingestion and/or surface erosion, two-phase flow, chemical reactions associated with combustion, very high temperatures with unsteady heat transfer, and jet mixing.

Turbomachinery, combustors, and jet mixing are three particular areas in advanced aerospace propulsion systems in which CFD is a key element in advancing the state of the art. Accurate experimental data have been extremely difficult or impossible to obtain in these areas, either because of unsteady flow or severe environments, or because it has not been possible to experimentally simulate the actual flight environment.

Turbomachinery

In the case of turbomachinery, unsteady flow occurs as a result of both periodic and nonperiodic disturbances. The nonperiodic disturbances may be caused by, for example, the ingestion of turbulent air or boundary layer separation.



The periodic unsteady flow results from wakes from upstream blade rows impinging on blades in downstream blade rows at a frequency known as the *blade passage frequency*. This is the classic rotor-stator interaction. These unsteady flow effects are important in both the compressor and the turbine, but for different reasons.

In the turbine the unsteady flow produced by the rotor-stator interaction causes unsteady surface pressure on the blades. As a result, the blades experience fluctuating lift and drag forces that translate into fluctuating normal and shear stresses. The contribution of the shear stresses to the aerodynamic losses through the turbine is important when assessing the turbine performance. Another important aspect of the flow through the turbine is the heat transfer. The temperatures in the turbine are extremely high and are, in fact, the primary limiting factor in turbojet engine design because of material temperature limitations. It has been found that the total temperature can fluctuate substantially in an unsteady flow, thereby resulting in local increases in temperatures that are already near upper limits. Other unsteady viscous and three-dimensional flow phenomena that are important in turbine components include secondary flows, tip bleed, and horseshoe vortex formations. For most of these phenomena, little or no time-accurate experimental data are available for analysis. Consequently, the numerical solution of the time-dependent compressible Navier-Stokes equations can provide significant insight into these unsteady flow problems. The computational results can be used to help define experiments for obtaining time-accurate data.

Compressor

The influence of unsteady flow in a compressor is different than that encountered in the turbine. Most modern compressor rotors operate in the transonic flow regime, in which the relative Mach number of the flow entering the rotor is supersonic while its axial component is subsonic. The performance of such compressors depends to a great degree upon the mass flow rate, which is controlled primarily by the axial velocity component. As a result of the supersonic relative inflow, an intricate system of shock and expansion waves exists in the blade passage. Consequently, a unique relationship exists between the Mach number and incidence angle of the incoming flow, which establishes a particular mass flow through the blade passage. Small deviations in the mass flow rate produce incorrect angles of attack that result in off-design performance penalties.

Another feature associated with this system of waves can cause fluctuations in the mass flow rate. In particular, the interaction of shock waves with the boundary layer on the suction surface results in a region of separated flow along that surface. This separation causes vortex shedding at the blade trailing edge, which in turn produces oscillations in the separation region. The separated flow acts as a blockage in the blade passage, thereby causing a reduction in the mass flow rate.

During actual operation these flow features are further complicated by wakes from upstream blade rows via the rotor-stator interaction. These wakes occur in the form of a velocity defect that causes fluctuations in both the magnitude and direction of the incoming flow. As a result of these fluctuations, the mass flow and the wave structure in the blade passage oscillate periodically. Consequently, in the case of unsteady flow the meaning of the unique incidence condition becomes somewhat obscured by the flow's oscillatory features.

The flow in this type of compressor is currently being investigated by solving the complete compressible time-dependent Navier-Stokes equations.[2,3] One of the major challenges in achieving such solutions lies in the development and implementation of boundary conditions that accurately represent unsteady flow. For the particular case currently being investigated, the unsteady inflow boundary conditions are developed using an experimentally obtained stagnation pressure profile at the leading edge of the rotor blade. In this stagnation pressure profile (Figure 1) the deep trough at 4° represents the wake from the upstream stator while the more shallow troughs at 1° and 6.3° represent the wake from the inlet guide vane. These data are used to represent the periodic entropy wave and velocity defect from the upstream wakes. Upon implementing this condition along with the remaining boundary conditions, computation of the flowfield between two adjacent rotor blades was carried out for several thousand time steps to ensure that the starting transients had been completely removed from the computational domain as well as to assess the periodic aspects of the unsteady flow.

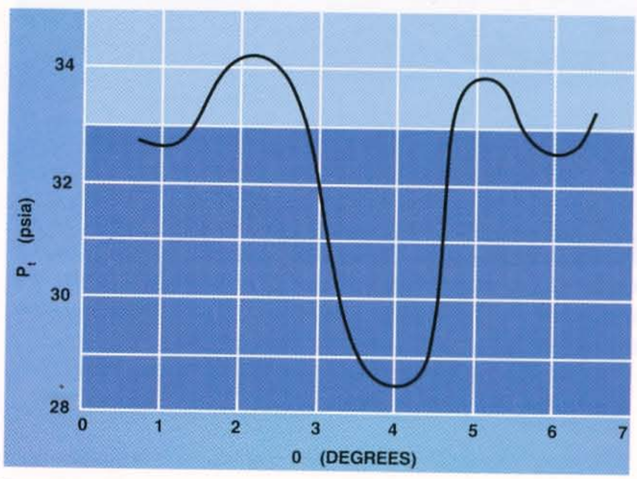
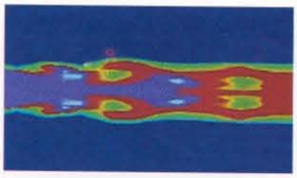


Figure 1. Time-dependent stagnation pressure profile at inflow plane of compressor rotor, obtained from experimental data.



bulence model is the substantial reduction of the flow separation region on the suction surface. This reduces the size of the vortex structures, which are ultimately shed. Some of these differences can be seen in the color graphics representation of the stagnation pressure computation with the turbulence model included (Figure 5). Particularly obvious in this figure are the reduced separation region and the stronger dissipation of the upstream wakes as they enter the rotor blade passage.

While the results obtained with the turbulence model generally exhibit the correct trends, additional analysis and comparison with time-accurate experimental data are required to validate the results. Many unresolved questions remain regarding the use of turbulence models with time-dependent computations. Further grid refinements can help in assessing these issues. In addition, such grid refinements are necessary to obtain more accurate simulations of flow details in the vicinity of the shock at the blade leading edge and in the region of the shock wave-boundary layer interaction.

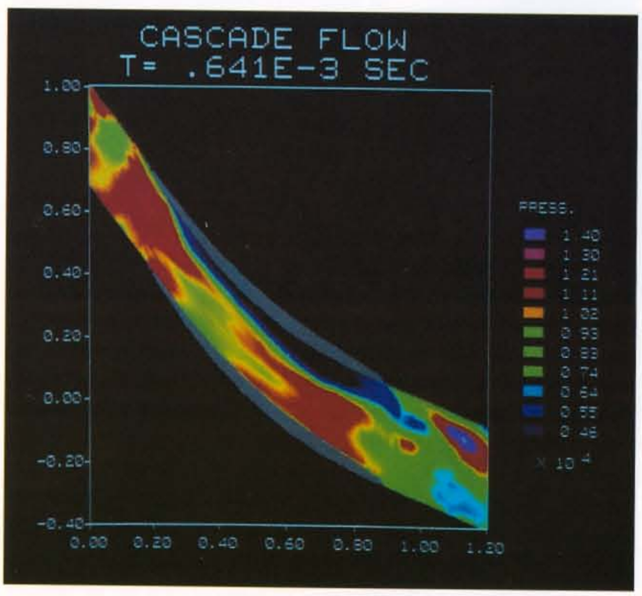
Efforts are currently under way to address these issues. In addition, the computational domain has been expanded to model the flow through four adjacent compressor blades to simulate more accurately the periodic inflow behavior. These computations are being performed on the CRAY X-MP computer system at Wright-Patterson Air Force Base in Ohio. The large memories and rapid processing of supercomputers now make it feasible to extend this capability to multiple blade rows and three-dimensional configurations.

Unsteady shear layers

Another area of primary interest in unsteady flows is the mixing that takes place in a shear layer. This phenomenon is important in jets, nozzles, ejectors, and combustors. These flows possess inherently unstable velocity profiles and thereby produce vortex rings that interact with one another and ultimately break up into turbulence. Noise, fuel-oxidizer mixing, and IR signatures are greatly influenced by the mixing of adjacent streams.

To investigate the mixing process, computations have been performed to simulate the flow in an axisymmetric confined jet configuration.[4] The results of these computations revealed the nature of the shedding of vortex rings and their

Figure 4. Color representation of computed stagnation pressure variation for time-dependent flow in a transonic compressor rotor with no turbulence model in the computation.



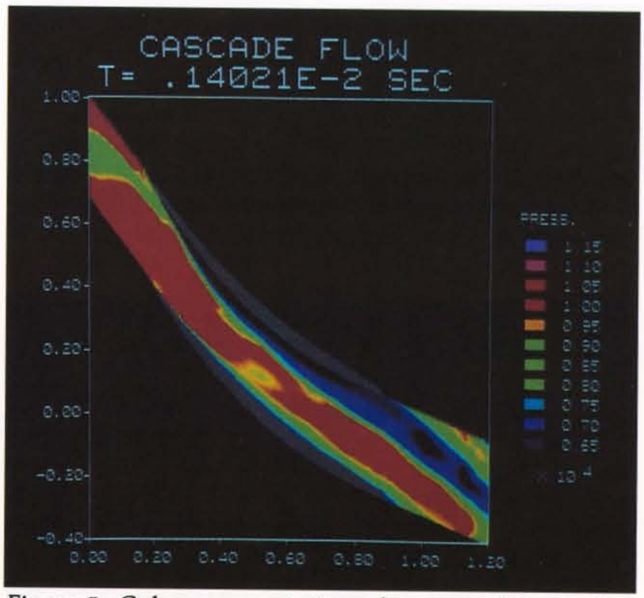
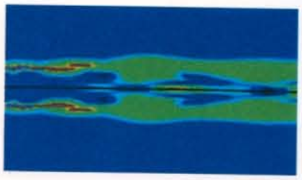


Figure 5. Color representation of computed stagnation pressure variation for time-dependent flow in a transonic compressor rotor. An algebraic eddy viscosity turbulence model was incorporated into the computational procedure.



Color graphics are used to show the results of the computation for Mach number, static pressure, and stagnation pressure. The Mach number variation (Figure 2) shows the general character of the shock at the blade leading edge and how it interacts with the boundary layer on the blade suction surface to produce the flow separation (dark blue) and the vortex shedding.

The static pressure variation (Figure 3) more accurately represents the details of the shock structure. In particular, the sharp bend or kink in the shock near the suction surface is more apparent from the static pressure data. Hence the shock exhibits the character of a classic lambda shock associated with shock wave-boundary layer interaction.

Analysis of the time-dependent flow features must account for two frequencies. Most apparent is the periodic fluctuation associated with the blade passage frequency. This is evident in the variation of the leading edge shock structure as well as the mass flow rate. A secondary oscillation of much lower frequency occurs in the flow separation as a result of the vortex shedding. This secondary frequency appears primarily in the mass flow rate but to a lesser degree in the shock structure.

The stagnation pressure variation (Figure 4) reveals some different flow features. Most notably the region of green near the leading edge represents the low stagnation pressure corresponding to the deep trough due to the upstream stator wake (in Figure 1). The green regions farther down in the blade passage show how the wakes merge and dissipate as they move through the blade passage. This phenomenon can be regarded as the migration of entropy waves through the blade passage. The extent of the flow separation and vortex shedding are also shown by the stagnation pressure variation.

The computational results shown in Figures 2 through 4 were obtained using a laminar formulation. To examine the influence of turbulence modeling, a turbulence model was incorporated into the computation and a number of differences were observed between the laminar and turbulent results. In general, the unsteady flow features are substantially suppressed by the turbulence model. The most dramatic example of this appears in the mass flow rate in which the amplitude of the fluctuations due to the upstream wakes was reduced significantly. This damping effect is also observed in the behavior of the leading edge shock oscillation. Another very significant effect of the tur-

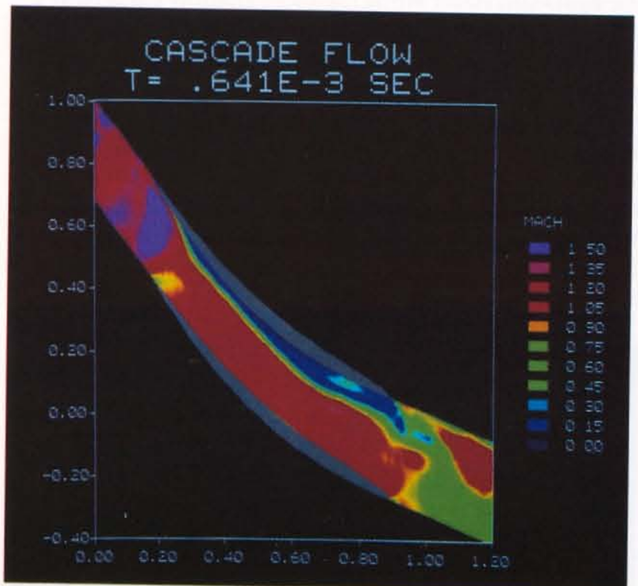
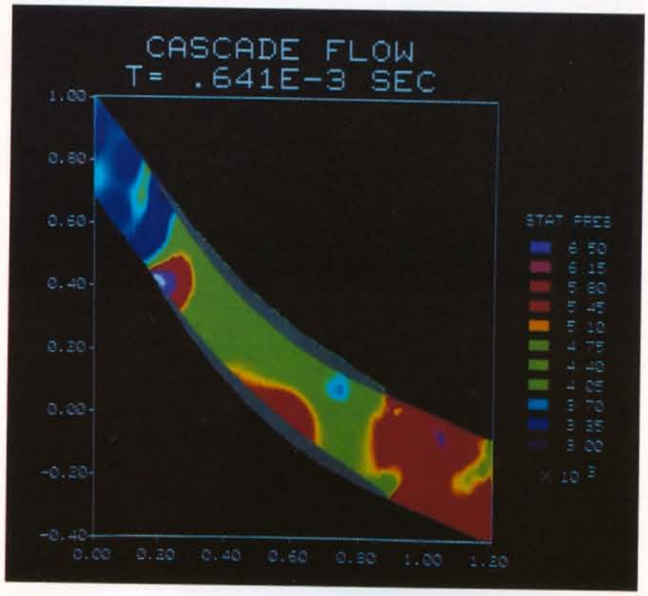
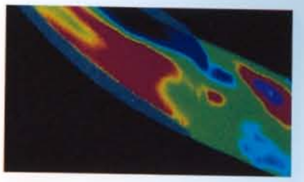


Figure 2. Color representation of computed Mach number variation for time-dependent flow in a transonic compressor rotor.

Figure 3. Color representation of computed static pressure variation for time-dependent flow in a transonic compressor rotor.





subsequent interaction. The computed vorticity and temperature fields are depicted using color representations (Figures 6 and 7) in which the merging of adjacent vortices can be observed. The presence of temperature extremes in these vortex rings (Figure 7) indicates the significant influence of the unsteady flow behavior in the distribution of stagnation enthalpy.

Methods of exciting the axisymmetric shear layer to enhance the mixing are currently under investigation through numerical simulation. These computations are being performed on the CRAY X-MP computer system at NASA Lewis Research Center, Cleveland, Ohio. Here again, the numerical simulation will now provide not only insight into the mixing process but also act as a tool that can be used for parametric studies to help optimize the design process.

Combustion

Combustion in modern aerospace propulsion systems can involve either unsteady or steady flow. Some typical combustor configurations contain subsonic, separated, and unsteady flow. A recent study was conducted to analyze the unsteady behavior of cold nonreacting flow in a bluff centerbody combustor in an effort to qualitatively compare the computed unsteady flow features with those observed in the frame.[5] The results (Figure 8) indicate that the numerical approach may have significant potential for analyzing such flows if the chemical reactions are incorporated into the computational procedure. This figure shows that large scale unsteady features in the form of vortex rings occur in both the cold flow simulation and in the combustion experiment.

Another challenging combustion problem in which CFD can play a vital role is associated with the propulsion system of the proposed aerospace plane. The propulsion system for this vehicle will consist of a supersonic combustion ramjet (SCRAMJET) engine. This type of engine requires the injection of hydrogen fuel into a supersonic stream. The subsequent combustion then requires that the flame front propagate at a supersonic speed relative to the incoming flow. This requires extremely high temperatures and high mass flow rates for which experimental simulation of the flight environment is not feasible. Hence the use of numerical simulation is necessary to bridge the gap between practical experiments and the actual flight condi-

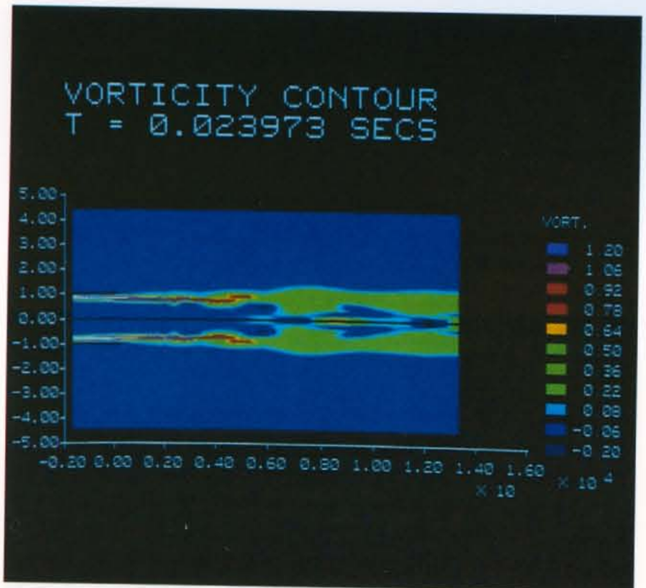
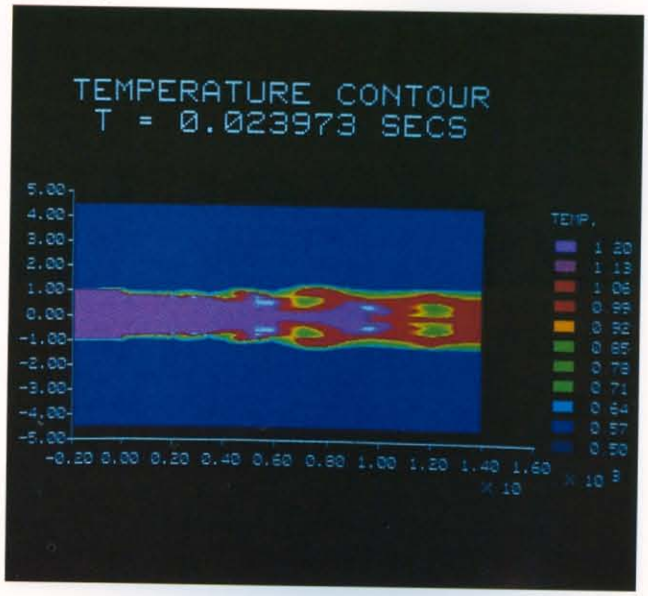
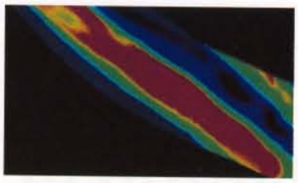


Figure 6. Color representation of computed vorticity for time-dependent flow in an axisymmetric shear layer.

Figure 7. Color representation of computed temperature variation for time-dependent flow in an axisymmetric shear layer.





tions. A program is currently in progress to compute the flow in both subsonic and supersonic ramjets with the chemistry of the combustion process included in the numerical formulation. These computations are being performed on a CRAY X-MP computer system at Wright-Patterson Air Force Base. Here too, the computational approach will provide a much better understanding of fundamental flow processes as well as a tool for achieving improved design capabilities.

Future promise

Several examples of numerical simulation of complex unsteady flows in propulsion systems have been shown. The efforts described here are preliminary and much more work is required to develop this approach into a reliable tool for accurately predicting the details of these types of flow. However, these early results are encouraging. Continued development of supercomputers such as the CRAY X-MP and CRAY-2 computer systems, along with this numerical approach, provides great promise for analyzing complex flow features in propulsion systems. Ultimately this computational capability will provide a means for assessing key performance parameters affecting design changes and will yield improved performance. Hence, the application of CFD to the design of propulsion systems promises significant payoffs. □

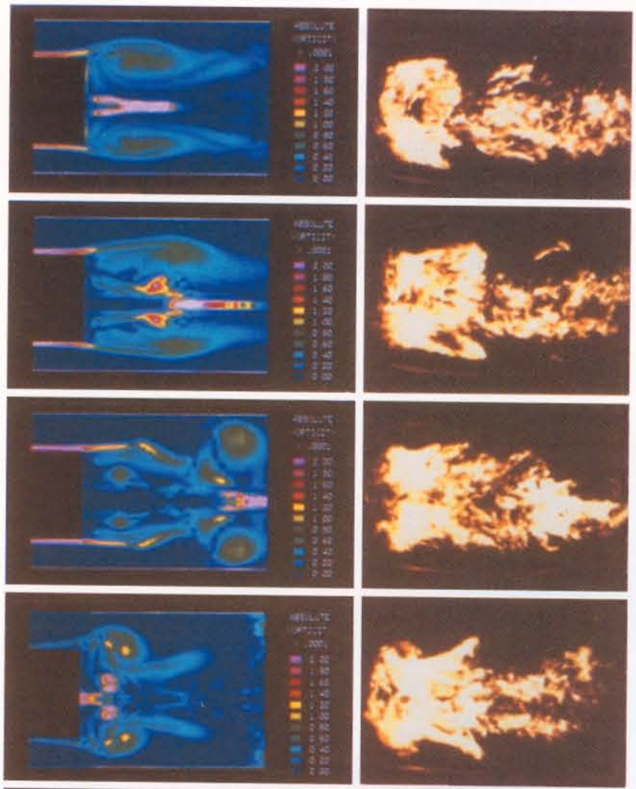


Figure 8. Comparison of computer-simulated vorticity in time-dependent cold flow and combustion experimental results in an axisymmetric centerbody configuration.

Acknowledgments

The compressor work described here is being supported by the Turbine Engine Division of the Aero Propulsion Laboratory of the Air Force Wright Aeronautical Laboratories, Wright-Patterson Air Force Base, Ohio. The ramjet combustion work is being supported by the Ramjet Division of the Aero Propulsion Laboratory. The unsteady shear layer computations are being supported by the Aerodynamics Branch at NASA Lewis Research Center, Cleveland, Ohio.

About the author

James N. Scott received his Ph.D. degree in aeronautical and astronautical engineering from Ohio State University in 1977. He spent six years with NASA Lewis Research Center working in the area of aerodynamic noise, and two years at NASA Goddard Space Flight Center studying acoustic launch environments. He has spent the last six years at the University of Dayton, where he is an associate professor, developing a research program in computational fluid dynamics emphasizing Navier-Stokes solutions for unsteady flow problems. His current research addresses turbomachinery, shear layers, and combustor flows.

References

1. Shang, J. S. and S. J. Scherr, "Navier-Stokes Solution of the Flow Field Around a Complete Aircraft," AIAA 85-1509, Cincinnati, Ohio, July 1957.

2. Scott, J. N. and W. L. Hankey, Jr., "Boundary Conditions for Navier-Stokes Solutions of Unsteady Flow in a Compressor Rotor," Three-Dimensional Flow Phenomena in Fluid Machinery, Fluids Engineering Technical Session, FED-Vol. 32, ASME 1985 Winter Annual Meeting, Miami Beach, Florida, November 1985.

3. Scott, J. N. and W. L. Hankey, Jr., "Navier-Stokes Solutions of Unsteady Flow in a Compressor Rotor," ASME Paper 86-GT-226, presented at the 31st ASME International Gas Turbine Conference and Exhibit, Dusseldorf, West Germany, June 8-12, 1986.

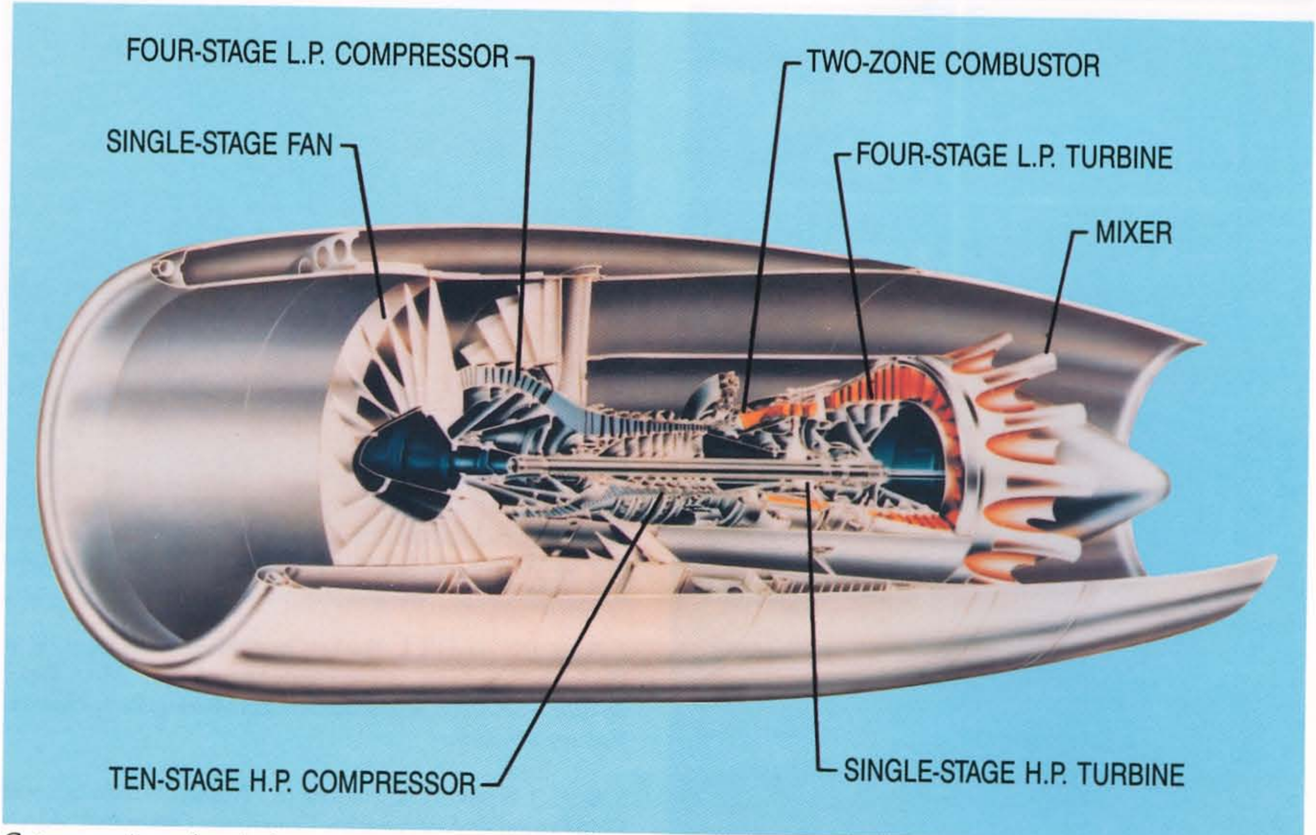
4. Scott, J. N. and W. L. Hankey, Jr., "Numerical Simulation of Unsteady Flow in an Axisymmetric Shear Layer," AIAA Paper 86-0202, AIAA 24th Aerospace Sciences Meeting, Reno, Nevada, January 6-9, 1986.

5. Scott, J. N. and W. L. Hankey, Jr., "Numerical Simulation of Cold Flow in an Axisymmetric Centerbody Combustor," AIAA Journal, Vol. 23, (5), May 1985, pp. 641-649.

Numerical simulation of multiblade row turbomachinery flows

John J. Adamczyk and Robert W. Graham
NASA Lewis Research Center, Cleveland, Ohio

Over the past 40 years, the design of the compressor and turbine used in gas turbine aircraft propulsion systems has progressed remarkably. This progress, however, has not been achieved without considerable difficulty; it has often been an arduous, frustrating experience marked by failure as well as achievement. Today, numerical modeling and supercomputers promise to greatly improve the efficiency of new design efforts.



Cutaway view of turbofan engine showing compressors and turbines.

During most of the evolutionary history of turbomachinery components, designers found themselves dealing with technical decisions beyond the state of available technological knowledge or experience. They were compelled to take risks and to extrapolate technical practice to meet the operating demands of the engine. In the early days, few analytical tools existed to bridge the gap between established practice and the design demands for these components. Trial and error was one of the practical ways of overcoming the gap. This process matured into a development testing program in which new concepts were tried and modified until they worked and proved themselves. Although the procedure was reasonably successful, it became increasingly apparent that the iteration process between fabrication and test was becoming prohibitively costly and time-consuming.

In conjunction with experimental development, increased fundamental research provided a foundation of information or data for improved design correlations or analysis. This research encompassed a broad spectrum of disciplines including fluid mechanics, aerodynamics, heat transfer, thermodynamics, structural mechanics, and materials. Engine manufacturers, universities, and government laboratories became joint participants in this research endeavor. In addition, the advent of the electronic computer and electronic acquisition and recording of data significantly influenced the use of fundamental research information in the design of compressors and turbines. There is a definite movement toward reliance on computational methods to establish the design of these components, with mechanical calculators and slide rules giving way to a new computational tool — the supercomputer.

It is still too early to predict what this new tool will make possible. To date, it has led to a deeper understanding of the flow field within a blade row and to the design parameters that influence it. Hopefully, further insight will be gained through higher-resolution simulations made possible by today's advanced supercomputers. This understanding should lead to innovative design concepts, improve the certainty of predictions, and minimize costly design changes.

This article traces the design practices for multiblade row turbomachinery dating back to the 1950s, with particular emphasis on computational methods. With such a background and a picture of the current situation, we will assess the future direction of computational fluid mechanics as applied to aircraft gas turbine turbomachinery components.

Early design approaches

The earliest U.S. compressor used in an aircraft gas turbine was centrifugal. For larger engines, this type of compressor was quickly replaced with the axial-flow type because of its smaller frontal profile and multistaging possibilities. The axial-flow machine has remained the preferred compressor configuration for the large engines used in civil transports and most military aircraft except helicopters. The turbine has traditionally been of the axial-flow type as well. Radial or mixed flow turbines have generally been restricted to small gas turbine applications.

In early designs, airfoil-wing theory was applied to the airfoil shapes of the blades and vanes of the axial-flow machine. The *blade element* design approach resulted from this association. In this design approach (shown schematically in Figure 1), each blade row was treated as a finite number of radially stacked elements. Velocity diagrams of the leading and trailing edges of each element were selected to give the proper flow turning and pass the required quantity of air. A blockage correction was applied to the flow area to account for the presence of the blades. An airfoil shape was selected that would accommodate the turning and velocity diagram specifications. Demands for increased performance per stage in more advanced machines called for maximizing loading or turning in each blade row. This demand necessitated loading and loss information obtained in experiments with stationary cascades of blades that simulated either a compressor or turbine blade row.

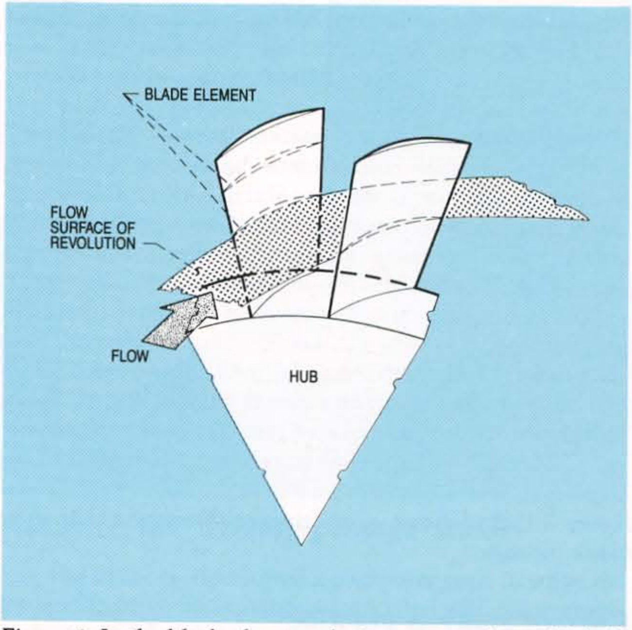


Figure 1. In the blade element design approach, each blade row is treated as a finite number of radially stacked elements.

In addition to the cascade approach, the performance of single-stage configurations was used to ascertain loading limits and losses. Analyses of these types of data from both cascades and single-stage machines were used in establishing empirical guides for loading limits and losses for individual blade rows. The early multistage design procedures amounted to extensions of the single-stage procedure.

Early in the design history, the major limitations of the practice were identified. Analytical approaches as well as experimentally-founded empiricisms were initiated to overcome these limitations. Some of the analytical approaches predate the general availability of electronic computers. For example, a three-dimensional approach to analyzing isolated blade rows appeared in the early 1950s.[1] This reference was a major contributor to the quasi-three-dimensional models that followed in the 1960s and 1970s in which "through-flow" or "flow-path" computational methods were developed.[2,3,4]

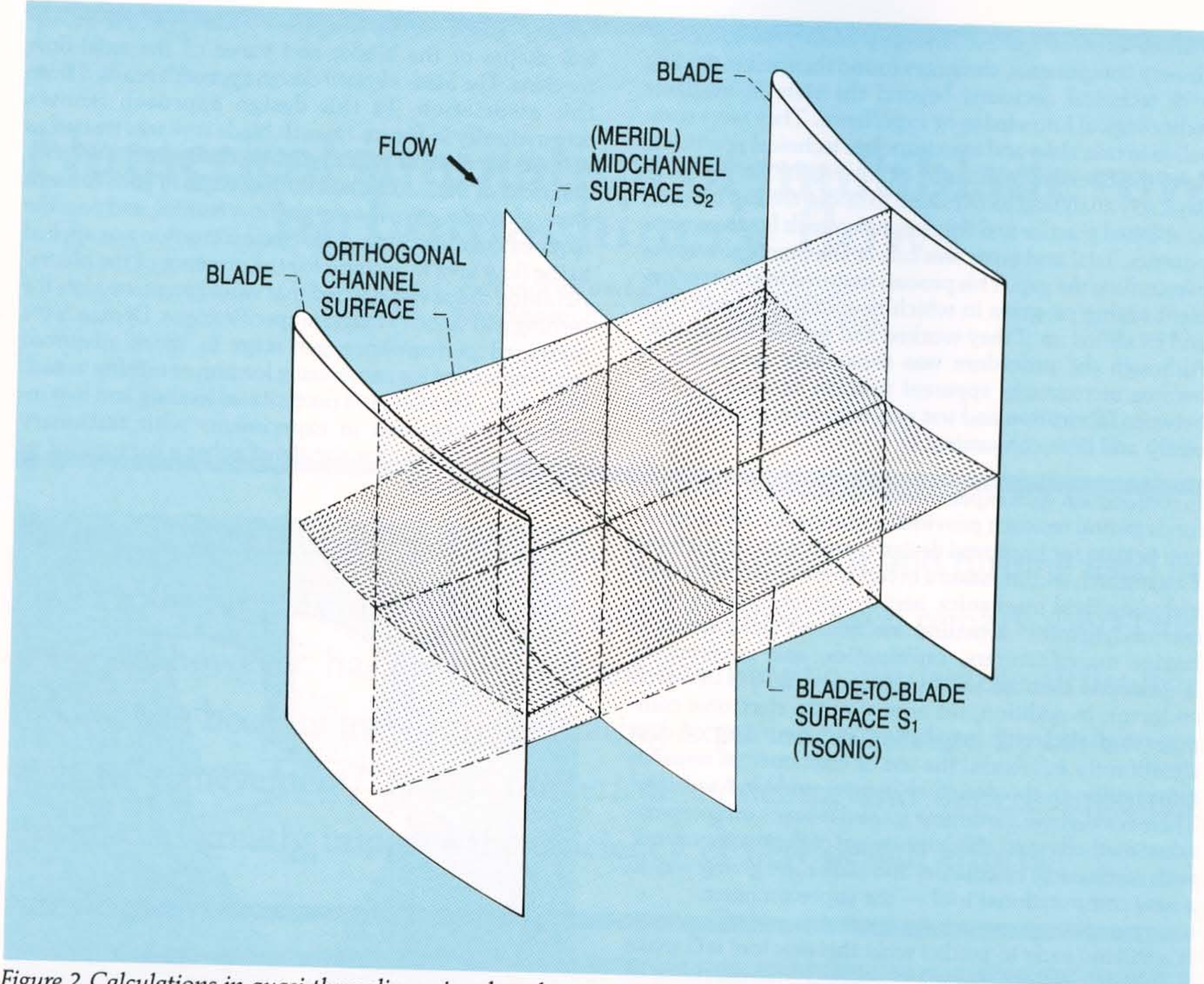


Figure 2. Calculations in quasi-three-dimensional analyses are carried out on two orthogonal surfaces within a typical blade passage.

Current analysis practice

Currently, the prevalent analytical methods used by industry in analyzing turbomachinery flow are of the quasi-three-dimensional type, in which the calculations are carried out on two orthogonal surfaces within a typical blade passage (Figure 2). The surface denoted as S_1 is an axisymmetric surface of revolution whose intersection with a blade row defines a cascade geometry. The relative flow field through the cascade is assumed to be steady in time. In practice, a finite number of S_1 surfaces are chosen to define a series of cascade flows from hub to shroud. The S_2 surface represents a meridional through-flow surface. The flow depicted on this surface is an axisymmetric representation of the flow field through the machine. This flow field is also assumed to be steady in time.

Upon deriving the flow equations for the S_1 and S_2 surfaces, one finds that they form a coupled system. The field equations for the S_1 surface (that is, cascade plan) require the geometry of the S_1 surface to be specified. This information is provided by the S_2 or meridional through-flow solution. In turn, the equations associated with the S_2 surface require input of the force exerted by the blading on the gas stream. This force is calculated from the S_1 cascade solutions. The effects of unsteadiness, turbulence, and end-wall flows (that is, clearance flows) are introduced through empirical correlations.

To date, the major effort in computational fluid dynamics as applied to turbomachinery has been the development of efficient and accurate numerical procedures for solving the quasi-three-dimensional flow model equations. The literature on this activity is very extensive and includes grid generation, algorithm development, transonic irrotational and rotational flow simulation, and viscous cascade flow simulation. Stimulated by the capability of high-speed computers, this activity has steadily shifted the methodology for turbomachinery design toward an analytical design system based on numerical simulation. Information once obtained from cascade performance tests is routinely provided today by numerical cascade flow simulations. These simulations, as a matter of course, provide detailed flow information far in excess of that obtained from cascade performance tests. Shock waves are identified and their strengths assessed, blade-surface pressure distribution is established, and its effect on boundary-layer growth is determined. Today, cascade tests are used primarily to validate and calibrate computer codes and in experimental studies of fluid dynamic phenomena that establish empirical relationships utilized in cascade flow simulation.

Future design approaches

The advances in turbomachinery technology brought about by use of the quasi-three-dimensional flow models are noteworthy. However, there are instances, such as off-design conditions and nonconventional geometries, that require extrapolation of the empirical data base for analysis, particularly where the results obtained from these models have been less than satisfactory. This generally occurs whenever there are significant radial variations in the flow across the pitch of a blade passage. Such flow variations can be brought about by in-passage shock waves, separated boundary layers, and end-wall flows. To better resolve these important flow features, analysts have begun to turn to three-dimensional flow codes.

In general, these three-dimensional flow codes analyze the flow through an isolated blade row. They are based on either the inviscid or Reynolds-averaged form of the Navier-Stokes equation of motion, written with respect to an observer fixed in the frame of reference of the blade row. The inviscid codes have been extremely useful in identifying the three-dimensional shock structure within high-speed compressor rotors and high-aspect-ratio stream turbine blading.[5,6] The viscous codes, because of the unavailability of sufficient computer capacity, have been limited to qualitative flow information. This, however, is changing rapidly as these codes become operational on supercomputers like the CRAY X-MP computer system with an SSD storage device or the CRAY-2 computer system. The challenge currently facing turbomachinery analysts is the extension of the three-dimensional isolated blade row codes to multiblade row configurations. A proposed scheme for meeting this challenge is outlined in the next section.

A direct extension of either the three-dimensional viscous or inviscid isolated blade-row codes noted previously to multiblade row configurations would in general entail resolving the time-dependent flow field within each blade passage. For all but the simplest of configurations, such an undertaking is impractical, even on today's most advanced supercomputers. It is also by no means obvious that answers to many questions related to blade-row performance and durability require this degree of flow resolution.

Flows frequently have a very wide spectrum of length and time scales that make direct numerical simulation impractical. Historically, aerodynamicists confronted with analyzing such flows have resorted to simulations based on flow models for which resolution is restricted to focus directly onto the features of the flow judged to be of most importance. The flow models used in the analysis of high-Reynolds-number flows about airfoils, wings, and bodies of revolution are classic examples of this approach. These models describe the deterministic flow field and not the random field associated with turbulence. Such models are developed to be compatible with the computer resources and with the diagnostic instrumentation at the researchers' disposal, while simultaneously providing a degree of flow resolution that enhances the understanding of the flow.

In keeping with the spirit of this approach, an analysis was undertaken to develop a flow model describing the three-dimensional, deterministic, time-averaged flow field within a typical blade-row passage of a multiblade row configuration.[7] The equations governing such a flow are referred to as the "average-passage" equation system. This flow model is a logical extension of the quasi-three-dimensional model to a three-dimensional one. For an isolated blade row, the model describes the deterministic flow field within a blade passage as governed by the Reynolds-averaged form of the Navier-Stokes equations.

The "average-passage" equation system can be derived by first forming the ensemble average of the Navier-Stokes equations, which yield the familiar Reynolds-averaged form. Next the equations are time-averaged everywhere in space with respect to a frame of reference fixed to each blade row. Finally, the time-averaged flow field of each blade row is phase-locked averaged with respect to the tangential direction. This last operation yields a flow field that is spatially periodic from blade passage to blade passage within a given blade row.

Through the "average-passage" model, each blade row is physically associated with a three-dimensional time-averaged flow field that is periodic from blade passage to blade passage. One finds that these flow fields are coupled to one another through a system of body forces, energy sources, and momentum and energy correlations, all of which require some degree of empiricism to model. It is through these terms that the effects of neighboring blade rows on the flow field are introduced. The modeling of the body forces, energy sources, and momentum and energy correlations forms the closure problem associated with the average-passage model.

Applying the "average-passage" model

We, the authors, developed a computer code to solve the inviscid form of the equations associated with the average-passage model to examine the model's potential usefulness. The algorithm chosen to solve the governing equations was in part based on the finite volume multistage Runge-Kutta integration procedure developed by Jameson, Schmidt, and Turkel.[8] This code was written to take advantage of the multiprocessor architecture of the CRAY X-MP computer system by assigning a processor to each blade row. To date, only configurations with two blade rows have been simulated. The geometries of these configurations have included an aircraft gas-turbine-engine high-speed fan stage, a rocket engine fuel-pump turbine stage, and high-speed, counter-rotating propellers. Of these, the simulations of the high-speed, counter-rotating propeller designed by General Electric most clearly show the resolution potential of the "average-passage" model.

The General Electric counter-rotating propellers are designed to operate at high subsonic flight Mach numbers and at high aerodynamic disc loads. To operate efficiently at these conditions, the propellers must be swept to reduce the effective Mach number of the approaching flow. The combination of operating condition and geometry leads to a flow field that is truly three-dimensional. It features a complex three-dimensional shock pattern and regions of cyclic flow produced by the tip vortex shed from each blade.

Because of the importance of these flow characteristics to propeller performance, there is a strong desire to be able to resolve them analytically. The "average-passage" model is fully capable of this task.

In a joint analysis project with General Electric, the shock-wave structure was analyzed in a preliminary design of a counter-rotating propeller. Previous attempts at resolving the structure of the shock waves using a quasi-three-dimensional flow model and an isolated three-dimensional blade-row model were unsatisfactory. To bring out the structure of the shock waves, we used color to code contour plots of the pressure field projected onto cross channel and blade-to-blade (that is, S_1 surfaces) planes.

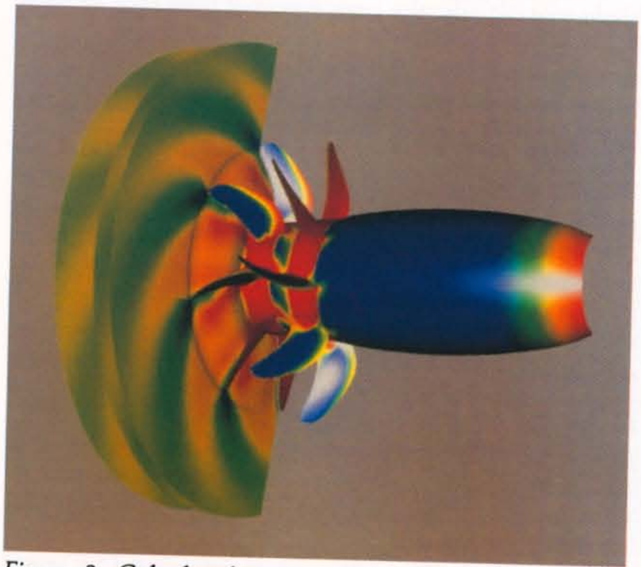


Figure 3. Calculated pressure field radiated by the aft propeller.

Figure 3 shows the pressure field radiated by the aft propeller. The pressure distribution is color-coded with a spectrum ranging from blue to green to yellow to red, with blue representing low pressure and red representing high pressure. The range of the color bar was chosen to highlight the radial and tangential variations of the pressure distribution displayed on three cross-channel planes. The first plane is located slightly forward of the trailing edge of the aft propeller. The remaining two are located downstream of the aft propeller. In this figure, the forward propeller rotates clockwise while the aft rotates counterclockwise. The green region is at a lower pressure than the yellow-orange region. The sharply defined boundary between these two colors is the footprint of the aft propeller trailing-edge shock. On the first cross-channel plane, the base of the shock lies at approximately three-quarters of the span. From this point it appears to spiral outward beyond the top of each blade. This shock appears to be weakened by an expansion wave emitting from the blade tips. This expansion wave is also responsible for inducing a cyclic motion in the flow which ultimately becomes the tip vortex.

Figure 4 shows the blade-to-blade projection of the pressure field of the forward and aft propeller. This figure was generated by matching the average-passage pressure field of the forward propeller to that of the aft propeller at a cross channel plane slightly downstream of the forward propeller. The color bar shown on the left of the figure again ranges from blue to red, with blue representing low pressure and red representing high pressure. The blade-to-blade plane on which the pressure field is projected is approximately at three-quarters of the span of the forward propeller. The sense of rotation of the forward propeller is toward the bottom of the figure and that of the aft is toward the top. The flow is from left to right. As the flow approaches the forward propeller, it expands about the suction, or upper blade surface, to supersonic speed, while on the pressure, or lower blade surface, it appears to be compressed. The trailing-edge shock of the first propeller

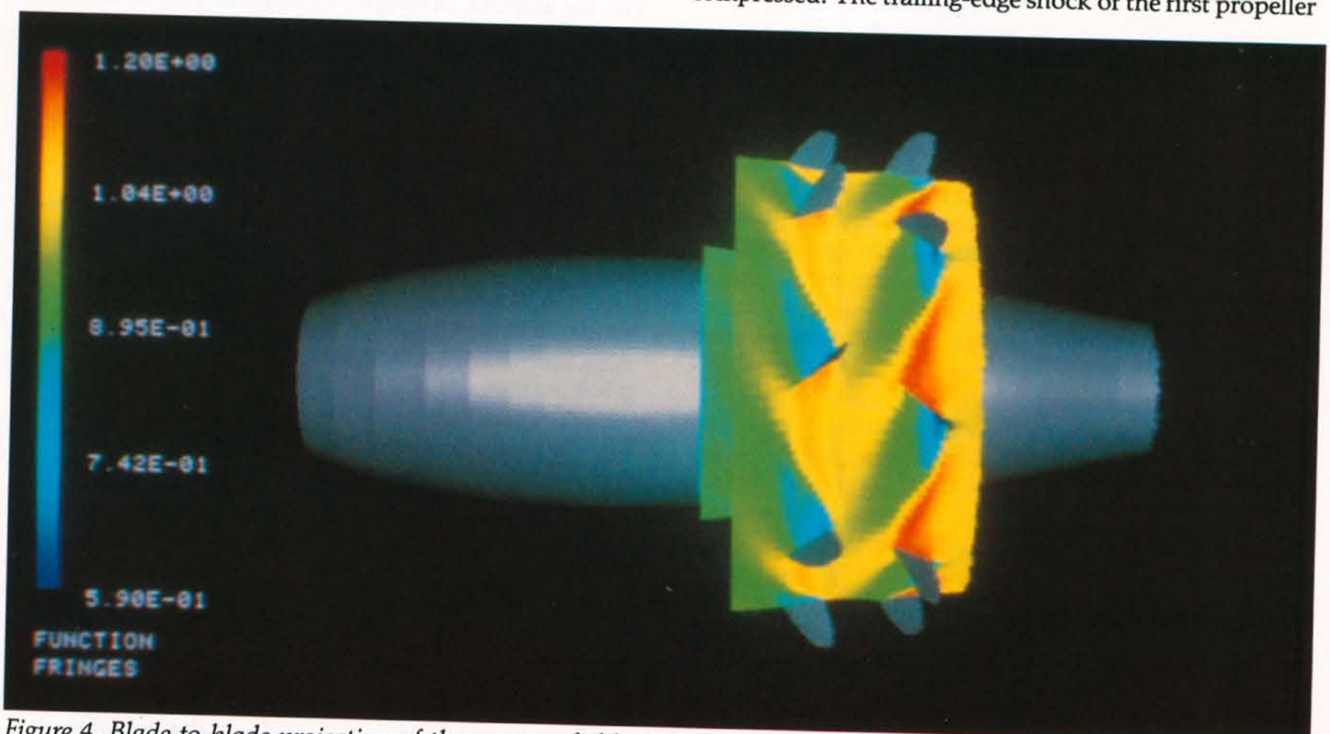


Figure 4. Blade-to-blade projection of the pressure field of the forward and aft propeller.

is defined by the sharp transition between blue-green and yellow. Because the axial or flight Mach number is subsonic, this shock propagates upstream of the forward propeller. This figure indicates that this shock significantly affects the flow over the entire pressure surface. The pressure distribution of the aft propeller is similar in character to that of the forward propeller. The only significant difference appears to be a stronger trailing-edge shock. The increased shock strength is caused in part by the higher approach Mach number as a result of the flow being acted upon by the forward propeller.

In a very subtle way, these figures illustrate the aerodynamic coupling that exists between the two propellers. The forward propeller induces a swirl in the oncoming flow which the aft propeller is designed to remove. The aerodynamic design of the aft propeller must thus be compatible with the flow discharged by the forward propeller. The forward propeller in turn must be designed to accommodate the higher mass flow induced by the aft propeller. Designers must be aware of the magnitude of this coupling if they are to design efficient sections for both propellers. Clearly one would have considerable difficulty assessing the magnitude of the coupling from isolated propeller simulations. The results of the "average-passage" model, as applied to a counter-rotating propeller, clearly show the capability of the approach. A similar capability exists for conventional turbomachinery configurations.

Simulation and design

The digital computers of the late 1950s and early 1960s revolutionized the methodology of analyzing turbomachinery flows by making numerical simulation of the axisymmetric through-flow practical in a design environment. Today we have supercomputers with sufficient computing power to once again bring about a revolution in turbomachinery flow analysis by making three-dimensional simulation practical. In this article we have proposed a model that could form the basis of such simulation. We have also demonstrated its ability to resolve the three-dimensional flow field associated with a high-speed counter-rotating propeller.

This initial application yielded very promising results which can be expected to improve with time. The full potential of three-dimensional numerical simulation of multiblade row flows can only be realized through a well-coordinated research effort involving grid generation, algorithm and code development, closure modeling, and code validation. The ultimate impact of this activity on turbomachinery design is unknown. One would hope its impact will equal or exceed that made by axisymmetric through-flow simulation. We are assured, however, that by using today's supercomputers and numerical simulation we can come closer to realizing a design system that optimally evolves configurations to meet specified constraints than we could by relying on empirical development programs.

Today's extensive test programs will be replaced by experimental research focused on addressing phenomenological issues related to flow physics. This represents a major redirection in turbomachinery aerodynamic research as we know it today and in part is the result of the advances in computer hardware made over the last ten years. □

Acknowledgments

The authors wish to express their appreciation to Cray Research for providing the computer resources to conduct this research activity and to Kent Misegades at Cray Research for his use of color to illustrate the output of this research.

About the authors

John J. Adamczyk graduated from the University of Connecticut with a bachelor of science degree in 1965 and a master of science degree in 1966. He received his Ph.D. in applied mechanics from the same institution in 1971. Adamcyzk joined the staff of NASA's Lewis Research Center in 1975. Prior to that he was employed by Westinghouse Research and the United Technologies Research Center. His research activity has focused on fluid mechanics associated with turbomachinery, with emphasis on analytical and numerical simulation.

Robert W. Graham graduated from Case-Western Reserve University with a bachelor of science degree in mechanical engineering in 1948. He received his master of science degree in 1950 and his Ph.D. in 1953, both from Purdue University, where he also taught from 1948 to 1951. Graham joined the staff of NASA's Lewis Research Center in 1948. He has specialized in the fluid mechanics of turbomachinery, heat transfer research, and studies of alternative energy sources.

References

1. Wu, Chung-Hua, "A General Theory of Three-Dimensional Flow in Subsonic and Supersonic Turbomachines of Axial-, Radial-, and Mixed-Flow Types," NASA TN-2604, 1952.

2. Smith, L. H., Jr., "The Radial-Equilibrium Equation of Turbomachinery," *Journal of Engineering for Power,* Trans. ASME, Ser. A, 88, pp.1-12, 1966. (Discussion in 88, 1966: 282-283).

3. Novak, R. A., "Streamline Curvature Computing Procedures for Fluid-Flow Problems," *Journal of Engineering for Power,* Trans. ASME, Ser. A, 89, pp. 478-490, 1967.

4. Katsanis, T., "Fortran Program for Calculating Transonic Velocities in a Blade-to-Blade Stream Surface of a Turbomachine," NASA TN D-5427, 1969.

5. Thompkins, W. T., Jr. and D. A. Oliver, "Three-Dimensional Flow Calculations for a Transonic Compressor Rotor," in *Through-Flow Calculations in Axial Turbomachinery,* AGARD-CP-195, 1976.

6. Denton, J. D., and U. K. Singh, "Time Marching Methods for Turbomachinery Flow Calculation," *Application of Numerical Methods to Flow Calculations in Turbomachines,* Von Karman Institute for Fluid Dynamics Lecture Series, 1979.

7. Adamczyk, J. J., "Model Equation for Simulating Flows in Multistage Turbomachinery," NASA TM-86869 and ASME 85-GT-226, November 1984.

8. Jameson, A., W. Schmidt, and E. Turkel, "Numerical Solutions of the Euler Equations by Finite Volume Methods Using Runge-Kutta Time-Stepping Schemes," AIAA-81-1259, June 1981.

CFD gets down to Earth

Michael Engelman, Fluid Dynamics International, Evanston, Illinois

When aerodynamicists use computational modeling, the process tends to be highly application-specific and often relies on proprietary codes. Consequently, it can be of limited use to others also interested in modeling fluid dynamics, such as engineers designing pumps, manufacturers casting metals, or brewers concerned that their bottle sterilizers do a thorough job. The Fluid Dynamics Analysis Package (FIDAP) is a CFD program developed to address a variety of industrial fluid-flow modeling needs. FIDAP was developed by Fluid Dynamics International (FDI) and is marketed jointly by FDI and Boeing Computer Services as a software package and as an application on Boeing's CRAY X-MP computer system.

Thanks largely to CFD's aerodynamics heritage, finite difference methods have been the traditional basis for CFD application codes. But borrowing a technique first applied in structural analysis, FIDAP uses the model building and computational techniques of the finite element method (FEM). The code is a generalized package for analyzing low-speed incompressible flows and thermal effects. It is the only commercially available finite element package that performs steady-state or transient analysis for two-dimensional axisymmetric and three-dimensional geometries. Applications have included the bottle sterilization problem mentioned above, fluid flow in carburetors, the behavior of molten aluminum during smelting, and the relative pressures in two different types of artificial heart valves.

FEM for CFD

As with structural analysis techniques, the first step in a finite element CFD analysis involves generating a computational mesh. Unlike finite difference codes, which use a mesh of squares resembling a sheet of graph paper and where curves or angles greater or smaller than 90° result in definite "steps," FIDAP's finite element selections include linear and quadratic isoparametric quadrilaterals for two-dimensional analyses. Selections for three-dimensional problems include linear and quadratic bricks, tetrahedrons, and wedges. The code's automatic mesh generator can tailor a mesh to virtually any geometry and can increase mesh density for greater detail by adding elements.

Beyond the use of a computational mesh, CFD differs from structural analysis in fundamental ways. Unlike a structural mesh, which models frames and solid structures, a CFD mesh models the spaces between solid structures. Structural analyses also address forces that change the shape of the elements and their relationships, whereas fluid dynamics accounts for changes in the velocity, temperature, and pressure of the fluid throughout the flow domain while leaving the mesh itself unchanged.

CFD and structural analysis also differ in relative complexity. The numerical model of a fluid problem, like that of a structural analysis problem, is a set of partial differential equations, but its mathematical complexity may be several orders of magnitude greater than a corresponding structural model. Three closely-coupled equations are the basis for fluid mechanics. Thermal effects add another equation; turbulence adds two more. Viscoelasticity can add several. All these equations are highly nonlinear.

Both structural and fluid equations were developed by British mathematicians in the 1840s. Though the structural calculations were beginning to see practical application in the early 1950s, fluid equations had to wait another 20 years for computers capable of solving anything but the most trivial problems.

To benefit fully from using any CFD code, engineers need a good background in fluid dynamics. Much important fluid behavior is only partly understood and the algorithms needed to solve the relevant equations only partly defined. Turbulence is a good example of such behavior. It is treated in FIDAP with a K-epsilon model, one that is state-of-the-art but nonetheless an approximation. In addition, fluid problems extend through time and may have more than one steady state, so taking a snapshot look at the solution may be misleading. Boundary layers also play a critical role in flow analysis. It often takes considerable insight to know where to concentrate the computational mesh. Focusing on too small an area within the problem will leave determining factors outside that area undetected.

Some applications

The FIDAP finite element CFD code was used recently to model a particularly demanding fluid flow problem: a continuous plate glass production process. The physical process involved unmelted chopped glass being poured steadily at one end of a furnace with molten glass flowing out the other end. The melting glass produced a moving front where the substance being studied changed state, behavior, volume, and temperature along a constantly shifting boundary. The furnace was an open chamber containing a lake of molten glass, and because temperature was not uniform throughout the lake, natural convection played a dominant role in the physics of the process.

Another commercial application, which is being explored by the Alcoa Corporation, involves modeling the physics of an electrolytic smelting cell to increase thermal and electrical efficiency. Smelting aluminum involves complicated fluid flows and powerful electrical forces, but the real barrier to physical testing is the 1000°C temperature and the closed nature of the system. Another high-temperature study is underway at the Westinghouse Research and Development Center, which is investigating gallium arsenide crystal growth to find an economic way to mass-produce computer chip substrate.

New applications are being investigated continually for possible future use. One such application involves finding circuit board cooling strategies for environments where convection currents cannot be exploited. Applying such strategies may result in less noise from cooling fans in personal computers and office equipment. Outer space and the topsy-turvy world inside attack fighter aircraft during aerobatics are also potential application environments, since FIDAP can accommodate the presence or absence of gravity and other forces.

A growing market

Although research and development are proceeding, the basic equations for FIDAP are already in place. Future additions will include improved turbulence modeling, a viscoelastic model that would make FIDAP ideal for studying plastic injection molding, and restructured algorithms to take better advantage of vector and parallel processing. With the availability of supercomputers such as Cray computer systems, finite element CFD will continue to move from the laboratory into wider practical application. □

About the author

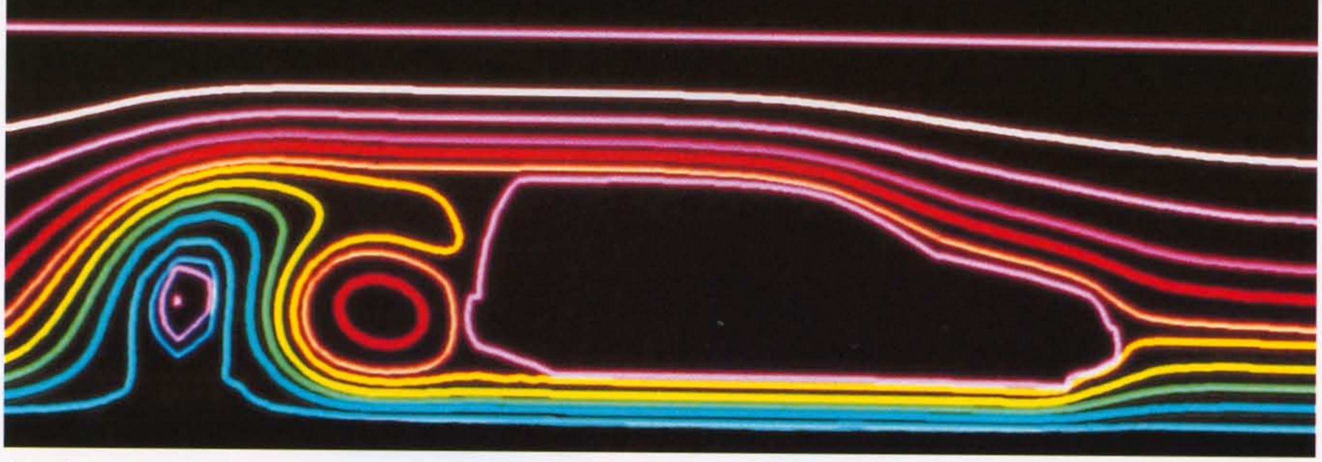
Michael Engelman is president of Fluid Dynamics International, a company that consults on a variety of fluid dynamics questions and supports and enhances FIDAP. Engelman is also an adjunct professor at the Illinois Institute of Technology.

The car problem

When the Los Angeles-based computer graphics company Digital Productions recently sought graphics footage to demonstrate their capabilities to scientific and technical audiences, they got what they needed running FIDAP on a Cray computer system. The production house, whose credits include computer-generated sequences used in the motion pictures *2010* and *The Last Starfighter*, wanted to create a technically accurate yet visibly exciting fluid analysis model of a wind tunnel test based on Fluid Dynamics International's two-dimensional definition of an automobile.

The final model had 2566 nodes and 2406 elements and was developed and edited within two days. Solving the problem took 1 hour of CPU time on a CRAY X-MP computer system with 1.5 hours of postprocessing. The problem matrix alone required over one-half million words of memory.

After viewing vector plots, streamline plots, and pressure distribution plots, Digital Productions chose the streamline plot of the air flow for its visual impact. The final output was produced on a CRAY X-MP computer system using the FIDAP postprocessor FIPOST with output to a camera connected directly to an I/O channel. Each frame was exposed directly onto the film. To create the appearance of slowed motion in the final footage, Digital Productions shot the film at the rate of 24 frames per second — the standard speed that prevents an image from flickering — and duplicated each frame once. Each frame took 1½ seconds to generate and 12 seconds to expose.



Aerodynamic automobile analysis performed using FIDAP on a CRAY X-MP computer system.

Supercomputers, the America's Cup, and winning

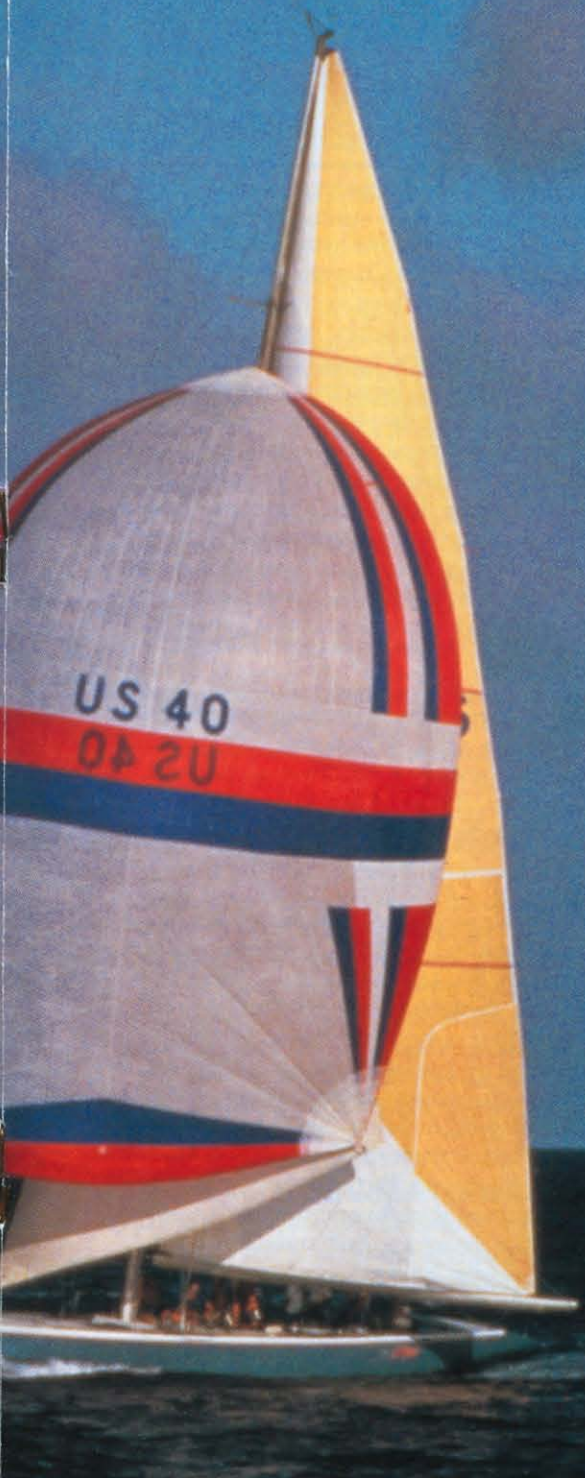
For 132 years, the gleaming symbol of yachting preeminence had been bolted to a table in the trophy room of the New York Yacht Club. The longest winning streak in sports history had kept sailing's most coveted prize — the America's Cup — in the country that gave the Cup its name. That is, until 1983, when an Australian syndicate with an innovative and controversial winglet-keeled yacht sailed away from the American defenders and took the once permanent Cup down under.

When *Australia II* defeated the U.S. yacht *Liberty* in the seventh and final race of the 1983 Cup defense, the Australians accomplished something that many in the 12-meter racing community thought to be impossible: they beat the American team with superior technology. Forget America's long and prestigious sailing history, dating back to the clipper ships and the great whaling fleets of the 19th century. This was a technology race, and the loss of the America's Cup was a stunning defeat to an American public that is unaccustomed to being outsmarted.

Within months of the failed 1983 defense, syndicates from around the world began developing plans for the 1987 challenge matches in Australia. The groups knew that they would need a significant leap forward from "traditional" 12-meter yacht design if they were to beat the Australians in their home waters. U.S. syndicates summoned some of the country's premier naval architects and 12-meter design specialists, as well as experts in aerodynamics,

hydrodynamics, and computer science. Knowing that the design of a truly superior contender would require considerable testing and analysis of new hull-and-keel configurations, designers from two of the leading U.S. syndicates also sought the help of supercomputers to try to gain the winning edge.

The St. Francis Golden Gate Challenge, a San Francisco-based syndicate, and the Sail America Foundation, San Diego, are both using a CRAY X-MP/48 computer system in an effort to predict the performance of new 12-meter designs. Cray Research has provided computer time at its Mendota Heights, Minnesota computer center for the testing.



The techniques being applied by the two design teams are fairly similar, but both believe firmly that their results will be markedly different. In fact, the critical design elements are so revolutionary that none of the designers are talking about their findings. The entire effort has been shrouded in secrecy.

Space age technologies aid an ancient craft

Charles Boppe, a technical specialist in aerodynamics for Grumman Aerospace Corporation, has been devoting his nights and weekends to the Sail America Foundation. With the after-hours help of two other Grumman engineers, Bruce Rosen and Joe Laiosa, Boppe has concentrated on the engineering analyses required for configuration development. The team has also worked toward defining trends and absolute levels for five critical variables: wave drag, "lift" or side force, lift-induced drag, viscous drag, and flow separation.

The designers were fortunate to have diagrams of *Australia II* that became available shortly after the 1983 races. Using *Australia II's* performance characteristics as a yardstick, they began the meticulous process of simulating hundreds of subtly different hull-and-keel configurations.

A yacht traveling through the water transfers energy to the water in the form of waves, creating drag. Wave drag predictions on an America's Cup 12-meter attempt to find a form that cuts through the water with a minimum of turbulence and drag. Panel method calculations determine the flow of water around the hull and simultaneously calculate the characteristics of the "flat" surface produced by the boat as it sits in the water. With the results of the panel method calculations, engineers are able to predict the side force exerted by the boat upon the water at various wind speeds. Side force, in turn, creates lift-induced drag which is similarly calculated and combined with the wave drag and side force predictions.

These elements yield a potential flow, or inviscid, solution. The next step is to determine the streamline, essentially following the path of a molecule of water as it flows around the hull and keel. This involves boundary layer calculations that predict viscous drag and any regions of separated flow. Bruce Rosen of Sail America notes that some degree of viscous drag — the tendency of fluid such as water to cling to a surface — is unavoidable. Flow separation, however, can be reduced or eliminated through the design of proper hydrodynamic shapes. Flow separation occurs at the trailing edge of a boat hull or airfoil, defining an area where the fluid flows past the widest part of the surface and leaves a drag-inducing vacuum directly behind the trailing edge. It is also difficult to predict what a separated flow will look like, further complicating the simulation of viscous effects. When flow separation can be eliminated, yacht performance can be predicted much more precisely.

Applying supercomputers to America's Cup yacht design is not new to Heiner Meldner, chief scientist for the St. Francis Golden Gate Challenge syndicate. As a researcher in the defense sciences department at Lawrence Livermore National Laboratories, he has worked with Cray systems for nearly a decade. Meldner offered his expertise for sail and keel design of a 1977 Cup defender.

"We looked at winglet keels back in 1977," Meldner recalls, "but our interpretation of the international rules at that time led us to believe winglets would be a violation." The *Australia II* syndicate received approval for its design before the 1983 Cup challenge.

Meldner characterizes the St. Francis design effort as a process of elimination. He and a development team of more than a dozen scientists and engineers calculated many multi-parameter sets of shapes, examining everything from overall boat length to the aspect ratio of winglets on the keel. Armed with this volume of shapes and possibilities, they then executed what he calls "relatively simple performance prediction programs." The team input probability curves for variables such as wind strength and wave height to predict the likely performance of various shapes in given conditions. They then scrambled the variables somewhat with "Monte Carlo Noise" and ran simulated races to test the relative performance of the designs described by the previous analysis.

The "winning" shapes that emerged from this analysis were then run through computer codes that attempt to simulate a large-scale tank test. While the earlier analysis required a few minutes of supercomputer time, these large-scale simulations required several hours of processing. According to Meldner, the codes have been more than 14 man-years in development.

Resolution for Meldner's large-scale simulations was 100 points in each dimension, or roughly one cell for every six square inches of hull. "It's a fairly coarse model," Meldner notes, "but that level of resolution is adequate for simulating the performance of a yacht that is more than 65 feet long." Calculating the three dimensions of length, height, and width (100 x 100 x 100 points) requires 1 million points to describe the geometry of one shape. Adding the variables of orientation to the water ("heel" in sailing jargon), three components of velocity, three components of force, boundary layer conditions, relationships of water to air and water to hull, results in up to 100 million calculations. Each of these must be time-stepped, or updated, 1000 times to gain accurate resolution in time-dependent Navier-Stokes calculations.

Meldner was reluctant to reveal just how many iterations of hull-and-keel geometries he put through simulation, knowing that competing syndicates would like to discover such details about his calculations. He admitted, however, that the number of iterations was "large."

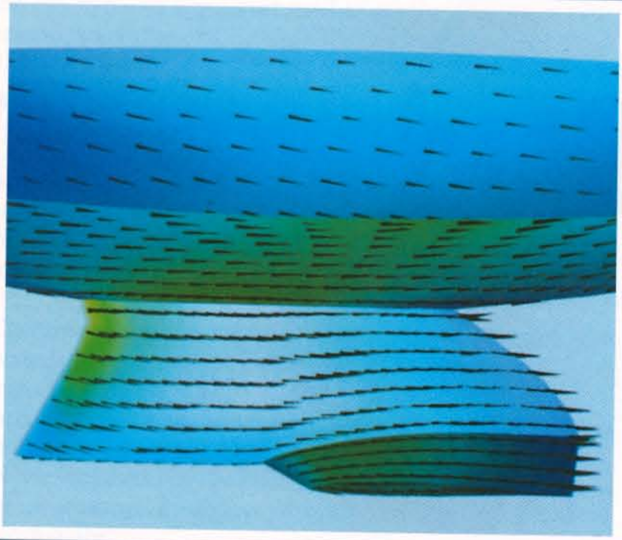
The computing demands were so great, in fact, that Meldner put a supercomputer at the top of his list of bare necessities. Because of the long run times, he noted that system reliability was of key importance. "The Cray system's high mean-time-to-interrupt and software availability make these kind of projects workable," he said. Meldner added to that list capable code development people who work hand-in-hand with the designers. "Designers can fall victim to wishful thinking," he says. "The parameters can be so weighted that the computer simulations will yield unrealistically favorable performance results."

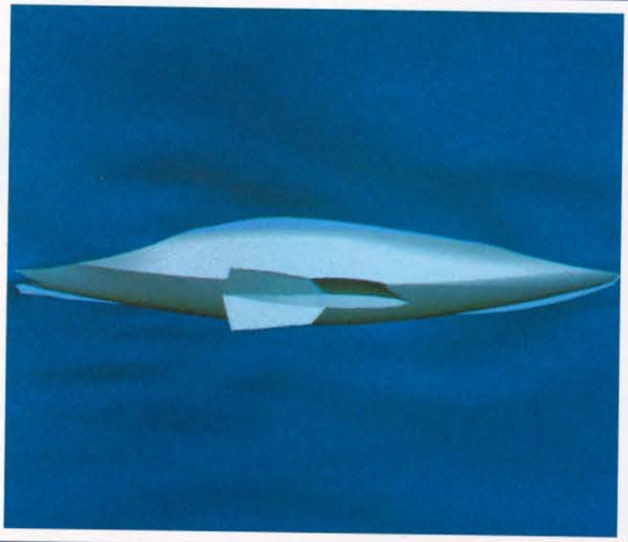
New conclusions bring new questions

Dr. Nils Salvesen, a manager in marine hydrodynamics at Science Applications International Corporation, is working with Charles Boppe and his colleagues on behalf of the Sail America Foundation. Like Meldner, he sees the difficulty of using computer models to predict performance. "It is one thing to discover the hydrodynamic forces on the hull and keel, but it is quite another to translate that information into actual performance," he says. "We can calculate the forces on the hull together with the forces on the sail. We can even correlate that with codes for sailing conditions off the coast of Perth (the home course for the 1987 Cup races) for the last ten years. While we are confident that we will put the best boat in the water for given conditions, we cannot predict what the sea will do the day of the race."

Meldner adds caution as well. "Perth typically has pretty strong winds, so it makes sense to optimize a boat for heavy air," he says, "but the early elimination races are in light air. Whichever conditions you design the boat for, you could simultaneously be creating your own Achille's heel."

Ironically, John Bertrand, who skippered *Australia II* to victory with supposedly superior technology, makes the point even stronger. In his account of the 1983 Cup challenge, *Born To Win*, he states that the Australians did not win the race with their radical keel design. Bertrand notes that although the winged keel of his yacht was predicted by computer models and extrapolation of test tank data to travel four percent faster than any conventional hull design, in fact it performed poorly under a variety of actual racing conditions. By the end of the fourth race he trailed the Americans three to one, and had to win three straight races to take the match. Bertrand stresses that it was no one secret that gave his boat an advantage, but rather a combination of factors including thorough crew training, skillful helmsmanship, and even sports psychology. Technology detractors should note, however, that the Australians did their tank testing in smooth water. They also did not have computer models that could cope with the complex hydrodynamic effects of ocean waves.





Two views of the ***Australia II****: side view showing keel and winglets. Green tinted areas indicate regions of low water pressure, blue regions indicate high water pressure (left). The keel and winglets as seen from the bottom (right).*

Both syndicates say that their new hull designs outperform the *Australia II* in computer simulation and tank testing. But neither group is relying on superior design alone. Dennis Conner, previous America's Cup winner and skipper of the losing 1983 defender *Liberty*, has been training with the Sail America crew in the Hawaiian Islands for several months. The St. Francis syndicate, led by veteran skipper Tom Blackaller, began sailing trials and practice matches in San Francisco Bay in March. Far from being arrogant, the two leading American challengers are setting to their tasks with a cold determination.

Lessons for future cup efforts

Just as computer simulation has not outweighed the time-honored need for a first-class skipper and a disciplined crew, computer techniques also have not replaced the basic intuitive artistry that is part of yacht design. The scientific teams took direction from veteran 12-meter design experts who still use drafting pens in concert with CAD workstations. They also relied on the graphics capabilities of Cray software and hardware to give them a visual representation of their mathematical designs. The software shows the distribution of water pressures and other critical information as gradations of color on the graphic image. With an actual form to look at and study, designers were able to exercise their intuitive talents in a sophisticated, interactive environment. Graphics will play an increasingly important role in the design of future America's Cup 12-meter yachts.

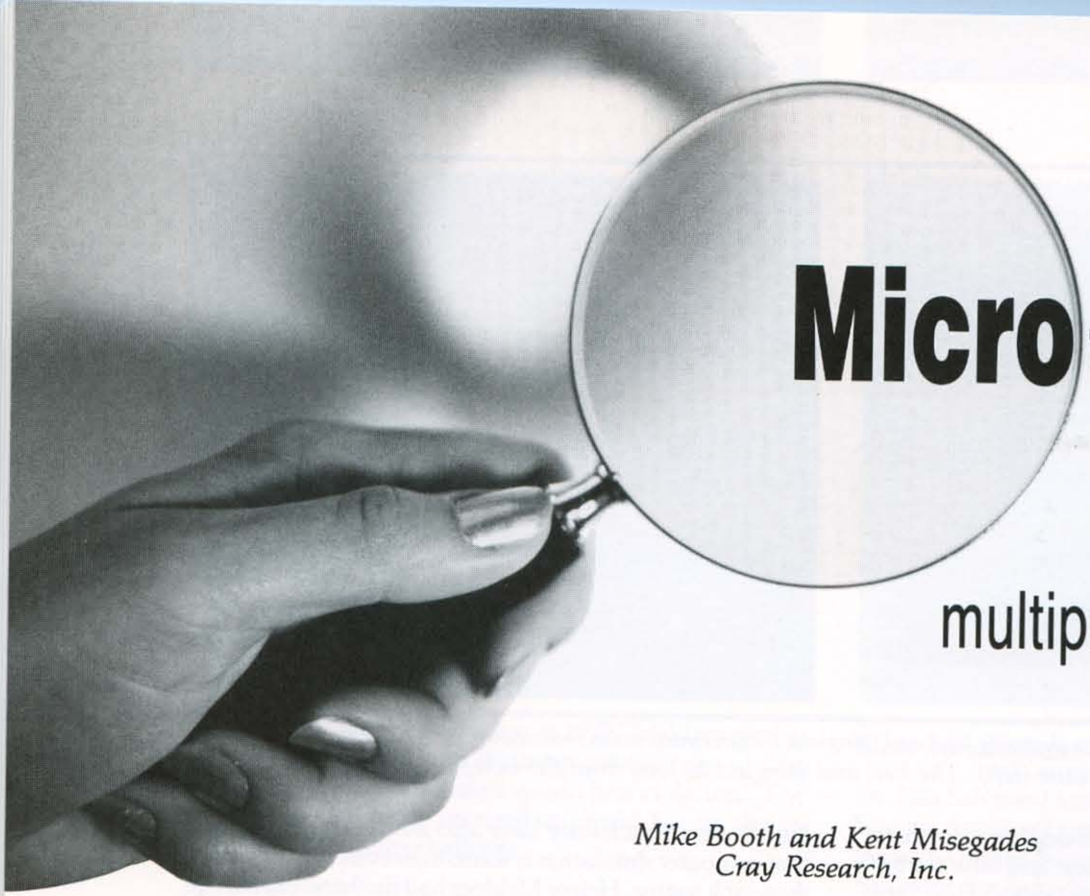
Both teams also valued the time savings provided by supercomputers. With only three to four years to prepare a boat and crew for a race, time is critical. The development of a one-third-scale model takes more than a month and costs more than $40,000. Syndicates lack the time and budget to produce more than a few models for tank testing. Supercomputers allow them to test radical ideas and refine concepts before committing them to a scale model. As Bruce Rosen puts it, "Computers allow us to weed-out novel designs that might be dogs in the water."

Hardware and software have also advanced to the point that computer simulation is sometimes even more accurate than tank testing. Heiner Meldner had firsthand experience with the inconsistency of tank tests. "We gathered data from tank testing of a scale model that indicated we should take a different design direction than the computer simulations had indicated. To double-check, we built a model of the design that the CRAY's output had recommended. As it turned out, the CRAY was right — by a wide margin." Tank testing is still a critical phase in the design process, but both teams agree that supercomputers will exert a growing influence on yacht design.

Despite the advances utilized and in some cases pioneered by the design teams, there is still room for improvement in their techniques. The simulation process is piecemeal, examining each variable independently and then correlating results for other variables. "The next America's Cup design effort will probably draw upon totally integrated methods capable of simultaneous solution of the five variables we examined," Charles Boppe said. "State-of-the-art computational fluid dynamics that includes the viscosity of the fluid in calculations is also not yet mature, but we're getting there. To be successful, designers will need to use these techniques."

May the best boat win

The two syndicates, as well as other interested groups in San Francisco and San Diego, are hoping that the next design effort will be for a Cup defender rather than a challenger. Four other U.S. syndicates, including the New York Yacht Club's *America II*, are hoping the same. The winning syndicate earns the right to host the next America's Cup matches, which means plenty of prestige, new jobs, and several hundred million dollars for the host city. Eight other challengers representing Canada, France, Italy, New Zealand, and Great Britain will be competing for the same right in trials that begin October 5 off Perth. The winning challenger will meet the Australian champion—most likely *Australia III*—in the finals that begin January 31, 1987. □



Microtasking: a new way to harness multiprocessors

Mike Booth and Kent Misegades
Cray Research, Inc.

Shortly after the CRAY X-MP multiprocessor computer system was introduced in 1982, Cray Research began to provide software tools allowing users to harness multiprocessors to operate on a single program. This multitasking technique is implemented at the subroutine level and is very effective at speeding up program turnaround for certain types of application programs.

By late 1984, Mike Booth and Jeff Nicholson of Cray Research's benchmarking and software development departments were experimenting with alternative ways of using multiple processors on a single program. Their multitasking implementation, called microtasking, is implemented at the DO-loop level where the task size, or granularity, may be small.

To distinguish between the two approaches to multitasking, Cray Research has dubbed the former implementation "macrotasking," while the latter continues to be called "microtasking." This article takes a close look at microtasking, discusses why and how it is used, provides a general example of how to microtask a code, and illustrates how a CFD code has been effectively microtasked and executed on a CRAY X-MP/48 computer system.

Cray users are accustomed to using vectorization to optimize their programs on Cray supercomputers. Vectorization employs special vector hardware in a single CPU to rapidly perform operations found in innermost DO-loops that have no data dependencies. In fact, most vectorization is done automatically by the Cray Fortran compilers, CFT and CFT77. Once vector optimization is complete, multitasking techniques can further optimize a program for execution on multiprocessor systems. A user may choose to employ macro- or microtasking implementations, or under certain conditions, may use both macro- and microtasking in a single program.

The incentive to consider multitasking is highest for programs that can be characterized as long-running, using most or all of memory, or requiring a dedicated environment. Codes with large granularity or with tasks that can run independently with few synchronization points, are good candidates for macrotasking. Cray Research has developed a set of Fortran subroutines using a basic set of primitive macrotasking functions. See CRAY CHANNELS Vol. 7, No. 2, Summer 1985 for more information about macrotasking.

Using microtasking

Microtasking makes codes with small task size and frequently synchronized parallel tasks good candidates for multitasking. This is because microtasking can partition codes at both large and small levels of granularity.

The fact that microtasking handles small granularity is helpful because independent small tasks can be identified quickly. In addition, it is not necessary to rewrite large portions of the program. Microtasking is specified by a small number of user-supplied directives that appear as comment lines. A preprocessor, called PREMULT, interprets the directives and inserts the appropriate microtasking code. The processed program is then compiled and executed in the normal way. The addition of the microtasking directives does not reduce the portability of the original program.

Because of its low overhead, microtasking is very efficient. There is little degradation in performance for a microtasked job in comparison to the original code using only one processor. In addition, microtasking works well when the number of processors available to a job is unknown or varies during the program's execution. This becomes very important in batch environments, where processors may become free briefly and sporadically. The microtasked job can dynamically adjust itself to use the number of processors available.

Microtasking methodology

Converting a program for microtasking does not involve programming individual processors. Rather, microtasking is a method of controlling the accessibility of available processors to the data (variables of a program) and the code of the program.

Controlling processor access to both the variables and the code is a coordinated procedure. First, the scope of the variables is determined. Next, independent portions of code are identified, and the appropriate control structures are selected to manage the parallelism. Finally, the scope of the variables may be changed to accommodate the need of processors to share variables or for independent workspace.

Determining the scope of variables

The ability of processors to access a variable is determined by the variable's scope. The scope of a variable is either global or local with respect to a subroutine. Scope is an attribute of the variable in the same sense as the attributes INTEGER, REAL, LOGICAL, or CHARACTER.

Global variables appear in COMMON blocks, SAVE statements, DATA statements, or in a subroutine's argument list. Variables with global scope have the same address for each processor entering a subroutine. The result of an operation performed on a global variable by one processor is known to all processors. Global variables are usually large data arrays, or read-only variables.

All other variables are local variables and can be referenced only within a subroutine. A separate and private storage location for each local variable exists on the stack for each processor entering a microtasked subroutine. Thus, the result of an operation on a local variable is known only to the processor that performed it. It is not accessible to other processors. Local variables are usually temporary workspace.

Control structures

Whether or not a processor can access the code of a microtasked subroutine is determined by user-defined control structures. Without control structures, each processor entering a microtasked subroutine will attempt to redundantly execute all of the code. The user inserts control structures to partition the code into processes and to coordinate their execution. A process is the smallest amount of code that can be assigned to a processor. The user also determines the number of processes in a control structure. This allows for multiple concurrent processes or, if only one process is specified in a control structure, a single process not done in parallel.

Control structures are defined by the user with directives in the Fortran source code. Since these directives begin with a *C* in column 1, they are ignored by compilers, and are recognized only by the Cray microtasking preprocessor, PREMULT. PREMULT reads the source code and replaces the directives with Fortran code and with calls to subroutines in the microtasking library. The list in Figure 1 describes the common preprocessor directives.

CMIC$ GETCPUS	Acquires processors for use in microtasked code
CMIC$ MICRO	Identifies a subroutine that contains directives
CMIC$ DO GLOBAL	Defines the bounds of a control structure in which each iteration of a DO-loop is treated as an independent process
CMIC$ PROCESS	Defines the start of a control structure specifying a list of processes that can be executed in parallel
CMIC$ ALSO PROCESS	Defines the beginning of an additional independent process
CMIC$ END PROCESS	Defines the end of the PROCESS block control structure
CMIC$ GUARD	Defines the start of a segment of code within a control structure that can be executed by only one processor at a time
CMIC$ END GUARD	Defines the end of a GUARD segment
CMIC$ CONTINUE	For the next subroutine call, all processors continue parallel processing at the lower level

Figure 1. Microtasking directives

Using the directives

A typical microtasked program begins with the directive GETCPUS, which readies the program for parallel processing. Subroutines that contain directives are preceded by the MICRO directive. A call to such a subroutine enables any available processors to join in the execution of that subroutine. The actual number of processors that participate is nondeterministic. When the microtasked subroutine finishes, one processor continues program execution until another microtasked subroutine is called.

Inside a microtasked subroutine, each of the available processors executes the code sequentially. Control structures exist in the code to coordinate this parallel execution. One control structure is specified by the DO GLOBAL directive. This directive appears before a DO-loop that has independent iterations. Each iteration of the loop becomes a process, and each available processor executes an unprocessed iteration until all the iterations have been completed. No processor leaves the control structure until the loop is finished.

Another control structure uses the PROCESS and END PROCESS directives in the simplest form. These directives surround a block of code which can be executed by only one processor. The block of code becomes a process. Any processor that arrives at an occupied PROCESS block waits until the block is completed before resuming execution after the PROCESS block.

A compound PROCESS block defines several processes. The PROCESS directive marks the beginning of the first process. The ALSO PROCESS directive is used to mark the end of one process and the beginning of another. The last process in the compound PROCESS block is followed by the END PROCESS directive. Each process defined in a compound PROCESS block may be executed in parallel.

Within any microtaskable code, the GUARD and END GUARD directives surround a segment of code that can be executed in parallel but not simultaneously. The CONTINUE directive is used in a microtasked subroutine to direct processors to a lower level microtasked subroutine where parallel processing may continue. In this case the behavior is the same as if the lower level routine were in-line. Any code not in a control structure is redundantly executed by each available processor.

Microtasking a code is illustrated by the following example.

An FFT example

The two-dimensional Fast Fourier Transform (FFT) subroutine illustrated in Figure 2a consists of four parts. First, a call to FFTINIT initializes a portion of a workspace, WORK, for an FFT of length N. Next, a call to FFTCOL is made to compute the FFT of each of the M columns of A. Third, a call to FFTINIT reinitializes the workspace for an FFT of length M. Finally, a call to FFTROW is made to compute the FFT of each of the N transformed rows of A.

Of the six variables in the subroutine, A, N, M, and WORK are global, while J and I are local. For each value of J, the computation of the FFT of the Jth column of A is independent of any other column, and can be viewed as a process. Each process uses WORK as temporary workspace.

The situation is similar for the row computations in the I loop. The J loop and the I loop should become control structures so that each processor entering the subroutine computes only a fraction of the total loop code. The calls to FFTINIT should not be in a control structure because each available processor must initialize its own private workspace. The scope of the variable WORK is not in agreement with the use of the variable as workspace in the

```
          SUBROUTINE FFT2D(A,N,M)
          DIMENSION A(N,M)
          COMMON / WSPACE / WORK(2048)
          CALL FFTINIT(N,WORK)
          DO 100 J = 1, M
100       CALL FFTCOL(A(1,J),N,1,WORK)
          CALL FFTINIT(M,WORK)
          DO 200 I = 1, N
200       CALL FFTROW(A(I,1),M,N,WORK)
          RETURN
          END
```

Figure 2a. An FFT subroutine

independent processes. The scope of WORK must be changed to accommodate this use. This is accomplished by removing WORK from its COMMON block, and dimensioning it as a local variable in the subroutine.

The microtasked version of the FFT2D subroutine in Figure 2b is produced by modifying the scope of WORK and adding directives.

```
CMIC$     MICRO
          SUBROUTINE FFT2D(A,N,M)
          DIMENSION A(N,M),WORK(2048)
          CALL FFTINIT(N,WORK)
CMIC$     DO GLOBAL
          DO 100 J = 1, M
100       CALL FFTCOL(A(1,J),N,1,WORK)
          CALL FFTINIT(M,WORK)
CMIC$     DO GLOBAL
          DO 200 I = 1, N
200       CALL FFTROW(A(I,1),M,N,WORK)
          RETURN
          END
```

Figure 2b. The microtasked FFT subroutine

A fluid dynamics example

Microtasking has been used effectively with CFD software running on a CRAY X-MP. A recent example is a three-dimensional Navier-Stokes code developed by Ramesh Agarwal and Jerry Deese of McDonnell Douglas Research Laboratories, St. Louis, Missouri. The code is used to simulate the airflow about wings and will eventually be capable of complete aircraft solutions.

The entire code was vectorized already, achieving a processing rate of 23.0 microseconds (µsec)/mesh-point/iteration on one processor of a CRAY X-MP/48 supercomputer. By using the Cray software performance tool, SPY, it was found that essentially all of the computational time was spent in five subroutines: EULER, FILTER, TSL, TSL1, and EMUTURB. Only these subroutines were microtasked.

Microtasking possibilities

A quick analysis of the five subroutines showed that the major computation consisted of a series of triply-nested DO-loops. With the exception of five singly-dimensioned workspace arrays used in subroutine EULER, none of the variables presented any scope problems at the outermost DO-loop level, where microtasking was actually performed.

The five workspace arrays in EULER hindered microtasking since they were contained in a COMMON block. This type of data structure would have been global to all processors, while the intent was to have a copy for each processor in the subroutine. The scope of these arrays was changed from global to local by simply using a DIMENSION statement instead of COMMON. Since the COMMON block appeared only in this subroutine, the use of DIMENSION instead of COMMON had no effect on other parts of the program.

```
C       MAIN PROGRAM

        DO 10 N = 1, NEND
          CALL EULER
10      CONTINUE
        STOP
        END

        SUBROUTINE EULER

        DO 10 K = 1, KL
          DO 10 J = 1, JL
            DO 10 I = 1, IL
C             MANY COMPUTATIONS
10      CONTINUE

        CALL FILTER

        CALL TSL

        CALL BC

        RETURN
        END

        SUBROUTINE FILTER

        DO 10 K = 1, KL
          DO 10 J = 1, JL
            DO 10 I = 1, IL
C             MANY COMPUTATIONS
10      CONTINUE
        RETURN
        END

        SUBROUTINE TSL

        CALL TSL1

        CALL EMUTURB

        DO 10 K = 1, KL
          DO 10 J = 1, JL
            DO 10 I = 1, IL
C             MANY COMPUTATIONS
10      CONTINUE
        RETURN
        END

        SUBROUTINE TSL1

        DO 10 K = 1, KL
          DO 10 J = 1, JL
            DO 10 I = 1, IL
C             MANY COMPUTATIONS
10      CONTINUE
        RETURN
        END

        SUBROUTINE EMUTURB

        DO 10 K = 1, KL
          DO 10 J = 1, JL
            DO 10 I = 1, IL
C             MANY COMPUTATIONS
10      CONTINUE
        RETURN
        END
```

```
C       MAIN PROGRAM
CMIC$   GETCPUS
        DO 10 N = 1, NEND
          CALL EULER
10      CONTINUE
        STOP
        END

CMIC$   MICRO
        SUBROUTINE EULER
CMIC$   DO GLOBAL
        DO 10 K = 1, KL
          DO 10 J = 1, JL
            DO 10 I = 1, IL
C             MANY COMPUTATIONS
10      CONTINUE
CMIC$   CONTINUE
        CALL FILTER
CMIC$   CONTINUE
        CALL TSL
CMIC$   PROCESS
        CALL BC
CMIC$   END PROCESS
        RETURN
        END

CMIC$   MICRO
        SUBROUTINE FILTER
CMIC$   DO GLOBAL
        DO 10 K = 1, KL
          DO 10 J = 1, JL
            DO 10 I = 1, IL
C             MANY COMPUTATIONS
10      CONTINUE
        RETURN
        END

CMIC$   MICRO
        SUBROUTINE TSL
CMIC$   CONTINUE
        CALL TSL1
CMIC$   CONTINUE
        CALL EMUTURB
CMIC$   DO GLOBAL
        DO 10 K = 1, KL
          DO 10 J = 1, JL
            DO 10 I = 1, IL
C             MANY COMPUTATIONS
10      CONTINUE
        RETURN
        END

CMIC$   MICRO
        SUBROUTINE TSL1
CMIC$   DO GLOBAL
        DO 10 K = 1, KL
          DO 10 J = 1, JL
            DO 10 I = 1, IL
C             MANY COMPUTATIONS
10      CONTINUE
        RETURN
        END

CMIC$   MICRO
        SUBROUTINE EMUTURB
CMIC$   DO GLOBAL
        DO 10 K = 1, KL
          DO 10 J = 1, JL
            DO 10 I = 1, IL
C             MANY COMPUTATIONS
10      CONTINUE
        RETURN
        END
```

Figure 3. Original program (left) and microtasked program (right).

The basic structure of the original code is given to the left in Figure 3. The microtasked code with preprocessor directives appears to the right. In total, 26 preprocessor directives were added to the program.

Performance results

The performance of this microtasked code was evaluated by running the code alone on the CRAY X-MP/48, where it required approximately 7.5 million words of main memory. The actual speedup achieved using four processors was 3.73 which is the ratio of wall-clock time for one CPU to wall-clock time for four CPUs. This slight difference from the maximum speed-up is due in part to the fact that several small sections of code in subroutine EULER were not done in parallel. In addition, a slight imbalance in processor workload could not be avoided. The microtasked execution produced an effective processing rate of 6.3 sec/mesh-point/iteration.

The total time required to analyze the code, add preprocessor directives, and generate a new executable program was approximately 30 minutes. Needless to say, a familiarity with the CFD code was an asset to this microtasking exercise. The fact, however, that microtasking can exploit small granularity parallelism, i.e., at the DO-loop level, makes the job of modifying many codes an easy one.

Cray's microtasking support

Microtasking is available on CRAY X-MP multiprocessor systems running level 1.14 or later versions of the COS operating system. The preprocessor, PREMULT, is available from Cray Software Development in a prerelease version. A chapter of the *Cray Multitasking User Guide Revision B* addresses the details of microtasking and provides guidance in converting programs to make use of this new optimization alternative. Those with additional questions about microtasking may contact Mike Booth at Cray Research, Inc., 400 East Las Colinas Boulevard, Suite 580 LB 56, Irving, Texas 75062; telephone: (214) 869-1676. □

About the authors

Mike Booth has been the district analyst manager for Cray Research in Dallas, Texas since early 1986. He was a sales analyst in Dallas and from 1980 until 1985, worked in Cray Research's benchmarking department. Before joining Cray Research, Booth was a research engineer with Arnold Engineering and Development Center in Tullahoma, Tennessee for a year. He received a bachelor of science degree in mechanical engineering from Memphis State University.

Kent Misegades is a senior applications analyst for Cray Research in Mendota Heights, Minnesota. Since joining the company in July 1984, he has been responsible for CFD code development. Misegades was previously an aerodynamicist at Dornier GmbH, West Germany, where he worked in the theoretical aerodynamics department from 1980 to 1984. He is a graduate of the Diploma Course in Fluid Dynamics of the von Karman Institute for Fluid Dynamics, Brussels, Belgium. He received a bachelor of science degree in mechanical engineering from Auburn University in 1979.

I N T R O D U C I N G

CFT77

Cray Research sets a new standard in Fortran compiler technology

Continuing Cray Research's parallel commitment to state-of-the-art hardware and software, the company is proud to announce the availability of its new generation Fortran compiler, CFT77.

Performance and application portability are top priorities for users of Cray systems, and CFT77 provides both. Applying the latest concepts in compiler design, CFT77 delivers highly vectorized and optimized code in the tradition of excellence established by CFT, Cray Research's initial Fortran compiler. In fact, CFT77 is designed to allow full portability of codes written for CFT. The initial release can be used on CRAY X-MP computer systems and releases scheduled for later this year will support CRAY-1 and CRAY-2 computer systems.

CFT77 conforms to the American National Standards Institute (ANSI) standard X3.9-1978, often called Fortran 77. It also contains a number of extensions to the standard, including those already supported by CFT. Some of the extensions add helpful features that make Fortran richer and more versatile. Others enhance portability by reflecting features developed by other manufacturers.

High performance

The performance delivered by CFT77 is multifaceted, taking full advantage of the unique architecture of the Cray computer systems. CFT77 delivers high performance on Fortran programs by utilizing:

- Vectorization. CFT77 automatically generates code that uses the vector registers and functional units of the Cray system. Speed improvements on the order of 10:1 are common when comparing vector processing to scalar processing. The programmer need not be concerned with vectorization — CFT77 vectorizes loop constructs automatically.
- Multitasking. CFT77 permits the partitioning of a program among multiple processors, enabling different sections to execute at the same time. Teamed with vectorization, multitasking is a powerful performance advantage.
- Scalar optimization. When operations are being performed serially, CFT77 provides extensive code optimizations to increase execution speed. The result: scalar code that approaches optimum performance on Cray computer systems.

One compiler for all Cray systems

CFT77 takes a two-track approach to portability. It will run on the CRAY-1, CRAY X-MP, and CRAY-2 computer systems and will allow for easy "rehosting" on future machines as they are developed. In addition to its portability across different types of Cray systems, the compiler adapts to differing software environments. It runs efficiently under both the COS and UNICOS operating systems.

Existing Cray Research customers will be pleased to know that changing from CFT to CFT77 is also easy. Routines that compile and execute correctly with CFT will do the same with CFT77.

The library partnership

Irrespective of the model of Cray computer system or the operating system being used, a wide variety of library routines are callable from CFT77. They include mathematical routines intrinsic to Fortran, routines that implement extensions to Fortran 77, scientific application routines, performance monitoring routines, and various I/O and utility routines.

With the Cray system libraries, users have ready access to sort utilities, a random number generator, Fourier analysis routines, and sparse matrix manipulation routines. The library routines are coded for maximum efficiency and are fine-tuned for each type of Cray system in use today.

Extensive supporting software

CFT77 includes all the features that Cray system users have come to expect. A symbolic debugging package complements the extensive compile-time checking performed by CFT77. The package lets the user set breakpoints, examine variable values, and make corrections while the code is executing. System and library routines dump job memory and provide listings of variable names and values. An interactive utility lets the user browse through the dumps using Fortran names and values.

When keeping track of nonstandard code is important, an option exists for flagging Fortran constructs that do not meet the Fortran 77 standard. In addition, a number of options are available for listing control, including the ability to set the level of warning messages issued by the compiler. CFT77 also has an extensive cross-reference facility that can list the addresses, references, and definitions of variables, statement labels, and subprograms.

CFT77 is compatible with all other Cray Research language processors. Routines compiled with CFT77 can call and be called by routines compiled or assembled using Cray Research's Pascal, C, or CFT compilers or the CAL assembler. SEGLDR, a segmenting loader, allows control over memory use at run time. This is particularly useful for large codes with several distinct sections, such as initialization, computation, and output.

Vectorization with CFT77

CFT77 combines the practical knowledge gained in Cray Research's decade of vectorization experience with the findings of research performed at several universities. CFT77 vectorizes loops containing nested IF statements, loops that use indirect (gather/scatter) addressing, loops that search for particular conditions, and a variety of other types of loops.

CFT77 provides an extensive set of vectorization diagnostics, telling the user what was vectorized and what was not (and why). Often simple code changes or compiler directives can help the compiler fully vectorize any remaining scalar program sections. In the future, CFT77 will divide individual DO-loops into vectorizable and nonvectorizable parts, as well as vectorize loops at different nesting levels.

Features and extensions

CFT77 supports all of the features contained in the Fortran 77 standard plus numerous important extensions to the language implemented by Cray Research, including:

- □ Array processing, permitting operations on whole arrays or array sections
- □ Arrays with flexible bounds (automatic arrays)
- □ Recursive subprograms
- □ Pointer data type
- □ Hollerith, Boolean, and hexadecimal constants
- □ Variable names of up to 31 characters and external and common block names containing up to eight characters
- □ Comments appended to a line of source code
- □ Compiler directives for vectorization control, dynamic common blocks, array bounds checking, and listing control
- □ A choice of static or stack memory allocation methods
- □ TASK COMMON storage for multitasking
- □ On the CRAY-2 computer system, common blocks allocated to local memory, permitting faster access to variables
- □ Asynchronous I/O, permitting input or output operations to execute at the same time as other program statements
- □ NAMELIST I/O
- □ Extra edit descriptors, including those for right justification and octal or hexadecimal output

Global optimization

CFT77 optimization analyzes an entire compilation unit to determine the execution and data flow within the subprogram. This information then provides the framework for applying optimizations that transform the internal representation of the Fortran program into a more efficient but functionally equivalent program. This is achieved by simplifying expressions and by detecting and eliminating redundant operations. Many different techniques, which apply to both scalar and vector code, are used to perform the optimization, including:

- □ Common subexpression elimination
- □ Forward propagation of constants and expressions
- □ Extraction of invariant expressions from loops
- □ Strength reductions
- □ Movement of stores out of loops
- □ Store elimination
- □ Dead code elimination
- □ Arithmetic simplification
- □ Short-circuiting of logical expressions
- □ Constant expression evaluation
- □ Bottom-loading of loops
- □ Instruction scheduling
- □ Global register assignment

A bright future

"CFT77 is part of a new era for Cray software," said Margaret Loftus, Cray Research's vice president of software. "Its vectorization and optimization capabilities have been carefully designed to provide users with ready access to the power of Cray hardware."

In addition to its performance, the compiler will serve as a bridge when customers consider upgrading to new, more powerful Cray systems. "Because the compiler will be available on current and future Cray computer systems, customer investment in software will be protected well into the future," Loftus said.

CFT77 delivers state-of-the-art performance. For additional information on CFT77 contact the nearest Cray Research sales office. □

CORPORATE REGISTER

More new customers for Cray computer systems

Recent Cray computer system installations and orders show Cray Research's continued penetration of commercial markets as well as continued recognition of Cray systems as the standard computational tool for the scientific community.

Lockheed Advanced Aeronautics Company (LAAC) installed a CRAY X-MP/24 computer system in January. The system is located at Lockheed's Kelly Johnson Research Center in Rye Canyon, California. LAAC is one of five companies in the Lockheed Corporation Advanced Systems Group (ASG), which also includes the Lockheed Georgia and Lockheed California Companies. LAAC oversees all aeronautical research and development for Lockheed. The company is now using the CRAY X-MP system to process scientific and design information for the five ASG companies, as well as the Lockheed Missiles and Space Company in Sunnyvale, California. According to LAAC scientific services director Douglas A. Ford, the Cray system will increase the unit's productivity by processing scientific codes more than 15 times faster than their largest general-purpose computer.

McDonnell Douglas Corporation ordered a CRAY X-MP/14 computer system and a 32-million-word SSD storage device, Cray Research announced in May. The system is scheduled for installation the second quarter of 1986 at the company's St. Louis, Missouri, facility. McDonnell Douglas will use the system to support the activities of several of the corporation's divisional companies in the design and production of aerospace products.

In March, Cray Research announced the order of a CRAY X-MP/48 computer system with a 128-million-word SSD solid-state storage device by the University Corporation for Atmospheric Research (UCAR). The system will be installed in the third quarter of this year at the National Center for Atmospheric Research (NCAR) in Boulder, Colorado. The new eight-million-word CRAY X-MP system will replace a one-million-word CRAY-1 computer system that has been installed at NCAR since 1983. NCAR is also home for the second CRAY-1 system ever produced, in operation at the center since 1977.

UCAR will use the new Cray system to study climate processes, atmospheric chemistry, weather prediction potential, solar and oceanic processes, and relationships among the atmosphere, the oceans, and the sun. "These studies are directly related to concerns of the nation on such topics as possible climate change from the release of carbon dioxide, acid rain, and predictions and warnings of severe weather," said Wilmot N. Hess, director of NCAR. "This new capability is essential to the advancement of our science to serve the national interest." UCAR is a consortium of 55 North American universities with graduate programs in atmospheric and related sciences. The consortium operates NCAR under contract with the National Science Foundation.

Merlin Profilers Limited of London has ordered a CRAY X-MP computer system due for installation in the first quarter of 1987 at the company's processing center in Woking, England. Merlin Profilers is a leading European seismic contractor that offers seismic services to the oil and gas exploration industry worldwide. Merlin acquired its first Cray system in 1984. The addition of a dual-processor CRAY X-MP computer system will considerably enhance the company's data processing throughput. As Merlin enters the growing three-dimensional seismic market in the North Sea and elsewhere in the world, group managing director Paul Blundell expects Merlin clients to demand even more rapid turnaround of high quality data and interpretation. "It is my view that with Merlin Profilers' high caliber staff and seismic software combined with the CRAY X-MP computer system, we have a unique opportunity to meet this challenge," Blundell said.

Cray Research announces 3.0 Pascal

Cray Research recently announced the availability of release 3.0 of the Cray Pascal Compiler. Users can now access the complete range of powerful Cray hardware while continuing to enjoy the benefits of structured programming.

Designed for portability, Cray Pascal runs on the CRAY-2, CRAY X-MP, and CRAY-1 computer systems. It executes under both the COS and UNICOS operating systems. The new compiler adheres to the ISO Level 1 standard with very minor exceptions, meaning that standard programs developed on other computer systems can easily be moved to Cray computer systems. In addition, a number of the extensions to the standard implemented in Cray Pascal match those of other Pascal compilers, enhancing portability even further.

Other Cray extensions are intended specifically to take advantage of Cray hardware. Vectorization and multitasking, for example, are both supported by 3.0 Pascal.

Pascal will vectorize FOR loops and special array processing constructs that do not contain dependencies. The following FOR loops vectorize on all Cray computer systems:

```
FOR i : = 1 TO n DO BEGIN
a[i] : = 0.0;
b[i] : = SQR(b[i]) + 1
END;

FOR i : = 1 TO n DO
d[i].fb[1] : = d[i].g;
```

Another FOR loop involves indirect indexing and takes advantage of gather/scatter hardware on a CRAY X-MP computer system to vectorize:

```
FOR i : = 1 TO 10 DO
a[c[i]] : = b[c[i]];
```

When all elements of an array are involved in an operation, array processing allows the user to specify a statement that will vectorize in the briefest of notations. The following example multiplies all elements in array b with the corresponding elements in array c and assigns the results to array a:

```
a : = b * c;
```

A Pascal program using the 3.0 compiler can be multitasked. In other words, the work of a single program can be broken up into several tasks, each of which is then processed by a different central processing unit (CPU). On Cray computer systems with multiple CPUs, the combination of vectorization and multitasking can significantly reduce the execution time of user programs.

Multitasking is implemented the same way in Pascal as in Fortran — through calls to routines in the multitasking library. All of the routines are declared as external procedures in a Pascal program. The user places calls to these routines in the program to indicate what sections of the code can be executed in parallel.

Tools are available to help debug and tune programs that use multitasking.

Other new features:

- ☐ CPU targeting, to permit a program developed on one Cray computer system to execute on another within the CRAY X-MP and CRAY-1 families
- ☐ Additional debugging and listing options
- ☐ Expressions in constant definitions, to allow greater flexibility in defining constants
- ☐ The ability to share data in Fortran TASK COMMON blocks
- ☐ A 32-bit integer data type, to allow faster operations on a CRAY-2 computer system
- ☐ Conditional expressions, to permit an IF-THEN-ELSE structure to be part of an expression

Cray Research announces release 1.0 of the C compiler

Release 1.0 of the Cray C Compiler is the most recent development in Cray Research's continuing effort to provide the highest performance systems available. The Cray C Compiler is based on the Portable C Compiler from AT&T Bell Laboratories.

The C language was originally designed as a high-level systems programming language. Most of the UNICOS operating system kernel code and utilities are written in C. Since C is a structured and highly efficient language, its potential has also been realized in programming applications other than operating system code. C offers a large standard library of functions and an ever-expanding base of software application programs.

The Cray C Compiler is available for use on the CRAY-2 computer system running the Cray operating system UNICOS and the CRAY X-MP and CRAY-1 computer systems running either the UNICOS or COS operating system.

The compiler translates C language statements into assembler instructions that make effective use of the target Cray computer system. Local variables, for example, use the high performance local memory on the CRAY-2 computer system and the B and T registers on the CRAY X-MP and CRAY-1 computer systems.

Features of the Cray C Compiler include:

- Declaration of a large number of register variables
- Support for the following data types: character, integer, short and long integers, float, and double. (Float and double, because of the hardware considerations, are both considered one 64-bit Cray word.)
- Shared common frame package and stack handling mechanisms, which allow access to Fortran and other languages
- Support for enumeration and recent C language additions announced by AT&T, including unique structure member names
- Ability to pass variable arguments between program modules on the CRAY-2 computer system

The C preprocessor, cpp, is included as a part of the Cray C Compiler. Cpp allows macro substitution, conditional compilation, and the inclusion of named files in the compilation process.

Use of the Cray C Compiler requires licensing. Contact the nearest Cray Research sales office for additional information.

Workshop explores issues and applications

In conjunction with the February 1986 opening of the National Center for Supercomputing Applications (NCSA), the University of Illinois at Urbana-Champaign hosted a four-day workshop for scientists and engineers working with high-speed computers. The workshop, "Scientific Applications and Algorithm Design for High-Speed Computing," drew over 100 researchers from private industry, education, and U.S. national laboratories. The event was sponsored jointly by the campus' two supercomputing centers and Cray Research. The centers are the NCSA, under the direction of Larry Smarr, and the Center for Supercomputing Research and Development (CSRD), directed by David Kuck.

The NCSA, established as part of the National Science Foundation's supercomputing initiative, is designed around the university's CRAY X-MP/24 computer system and offers supercomputing services to qualified researchers on site and via a high-speed communication network. The CSRD was established to research technical issues relating to large-scale computers. Its focus is the design and construction of Cedar, a parallel computer system for general purpose applications.

Attendees at the workshop were treated to detailed presentations on numerical techniques in scientific problem solving and to lively discussions of supercomputer performance and utilization issues.

The opening round of presentations and panel discussions featured Smarr, Kuck, and Steve Chen, senior vice president at Cray Research and chief designer of the CRAY X-MP series of computer systems. They addressed the topics of supercomputing architectures and environments.

In his overview of the NCSA, its facilities and philosophy, Smarr stressed a need for interdisciplinary research and linkages among computer technologies, particularly supercomputers and personal computers. "Seymour Cray was developing the CRAY-1 computer at about the same time Steve Jobs and Steve Wozniak were working on the first Apple computer," Smarr said. "A major thrust of the NCSA will be to bring supercomputers and personal computers together; the two can be integrated very naturally. Our overall intent is to be a kind of combination laboratory and theoretical research center — a place where scientists from the academic and industrial worlds can come together, share ideas, and try them out."

In his presentation, David Kuck explained the Cedar project. Its goal is the construction of a high-speed computer by linking clusters of eight processors, with a goal of doubling performance annually and linking 128 processors in five years. Kuck presented this approach as a middle ground between that of massive parallelism — using up to several hundred relatively slow processors — and lower-level parallelism using relatively few very fast processors. "Using massive parallelism probably is nice for a supermini, but I'm not sure it's the way to build a supercomputer at this moment," Kuck said. "There are problems with switching and with the massiveness of the parallelism itself." Low level parallelism, which characterizes the CRAY X-MP series, also presents difficulties, said Kuck, such as the engineering difficulties involved in cooling.

Chen noted during his talk that supercomputers have been defined and characterized in many ways over the years. "They are sometimes called the fastest computers available at any given time, and this is true. They are also said to be used primarily for scientific and engineering applications, which is also true. Other people say they are one generation behind what is needed to solve the latest problems of interest, and this is true. Now I would like to add to the confusion, and suggest that supercomputers can also be characterized as being one generation ahead of the latest user application programming techniques."

Chen's comment was in part a response to interest in even greater degrees of hardware parallelism, as voiced by several attendees at the workshop. Other issues raised during the course of the workshop included concerns over a percieved lag in graphics capabilities and a need for at least limited symbolic processing availability. "Input from our users is always helpful," Chen said, "and a workshop like this gives us an opportunity to hear user concerns firsthand."

The technical presentations covered a range of supercomputer applications, including fluid dynamics, theoretical chemistry, device and circuit simulation, structural dynamics, petroleum reservoir simulation, and seismic data processing.

APPLICATIONS IN DEPTH

VSAERO: a practical engineering tool

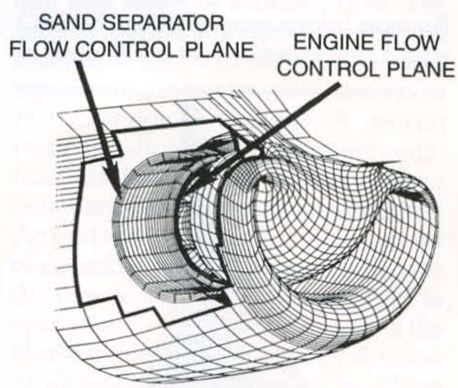
When panel methods for aerodynamic calculations were introduced about 25 years ago, they offered the first effective alternative to wind tunnel testing. The methods continue to find application as engineering tools today, despite advances in finite difference methodology that have engaged much of the research community.

VSAERO model of Grumman 698-411 with nacelles tilted 60°. Red represents low pressure; blue represents high pressure. The powered nacelle is represented with twin coaxial jet tubes modeling the fan and core exhausts. The fan inlet face mass flow was modeled to obtain correct spillage effects. Complete analysis took about 10 minutes on a CRAY-1/S computer system.

Interest in panel methods has recently been fueled by their expanded capability to model real flow features on a wide range of shapes. The ease with which this can be accomplished is exemplified by the program VSAERO, developed by Analytical Methods, Inc. of Redmond, Washington, with support from the NASA Ames Research Center. VSAERO is currently used in many applications covering complete aircraft, including V/STOL configurations, and helicopters, marine craft, and automobiles.

VSAERO is a low-order panel method; it employs flat panels with uniform source and doublet singularities, as opposed to quadratic or cubic singularity distributions on the panels (as are used in high-order methods). The low-order approach has several advantages:

- □ The construction of the panel model is simplified because there is no



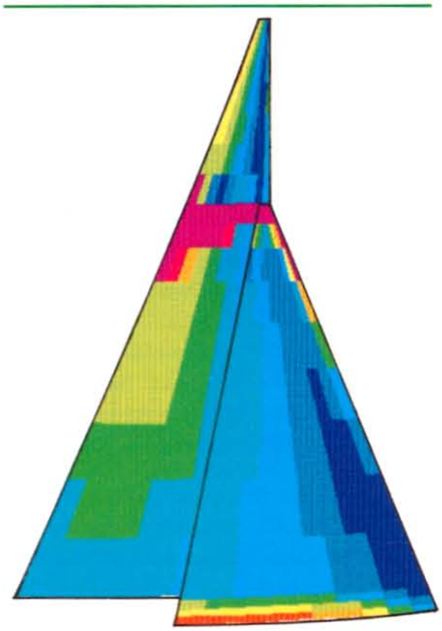
Model of nacelle and inlet for tilt-rotor vertical lift aircraft. The design was analyzed for the Boeing Military Airplane Company using VSAERO on a CRAY-1/S computer system. The aircraft inlet has to operate over a wide range of conditions as the craft moves from hover, through transition with the nacelles still at 90° relative to the direction of motion, and finally to forward flight with nacelles aligned with the flow. On the basis of VSAERO solutions, an inlet design that worked over the above range of conditions was obtained. A typical case involving 2500 panels took 350 seconds on a CRAY-1/S computer system.

requirement to maintain singularity slope continuity across network boundaries, substantially reducing the effort needed to prepare input.

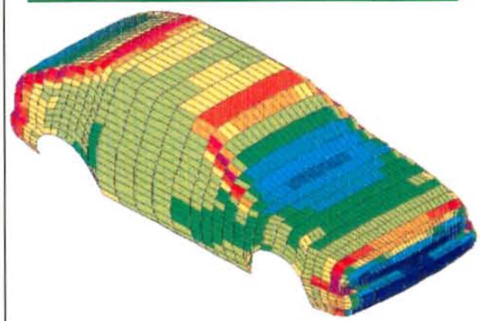
- The manipulation of interpolation parameters within the program is avoided, resulting in significant savings in I/O time and storage.
- The speed of VSAERO, as verified by independent investigators, is typically 20 to 30 times faster than that of higher-order codes (see Margason, R.J., Kjelgaard, S.W., Sellers, W.L. and Morris, C.E.K., "Subsonic Panel Methods — A Comparison of Several Production Codes," AIAA-85-0280, January 1985).

Brian Maskew, vice president of research at Analytical Methods, reports, "Large cases involving more than 2000 panels (unknowns), are best performed on a supercomputer such as a CRAY X-MP system. Typically, a complex case with 3000 panels takes less than eight minutes on such a computer."

For more information on using VSAERO with Cray systems, contact Analytical Methods, Inc., 2133 152nd Avenue N.E., Redmond, WA 98052; telephone: (206) 643-9090.



Panel methods offer high performance yacht designers the ability to analyze sail combinations and novel keel designs. Shown here is a typical spectrum plot calculated with VSAERO illustrating the loading on a fractional rig sail plan.



VSAERO potential flow results for a hatchback automobile. Red represents low pressure; dark blue represents high pressure. The calculations were performed for Isuzu Motors, Ltd. using VSAERO on a CRAY X-MP computer system. The analysis indicated a high pressure region at the grille and base of the windshield. The possibility of flow separation approaching this region was indicated by the boundary layer calculations. Peak suction at the sides of the windshield lead to the formation of the A-pillar vortex.

Amtec offers CFD codes for Cray computers

Concurrent with the increased use of CFD in aircraft design has been the emergence of a new industry — the design and marketing of commercial CFD software. As a participant in this expanding field, Amtec Engineering, Inc., of Bellevue, Washington, recently made available five new CFD codes developed for use on Cray computers. Principal applications for the codes are in the aerospace industry, but they are suitable for a wide range of flow problems. The following descriptions list the codes' key features.

PNOZ/2D

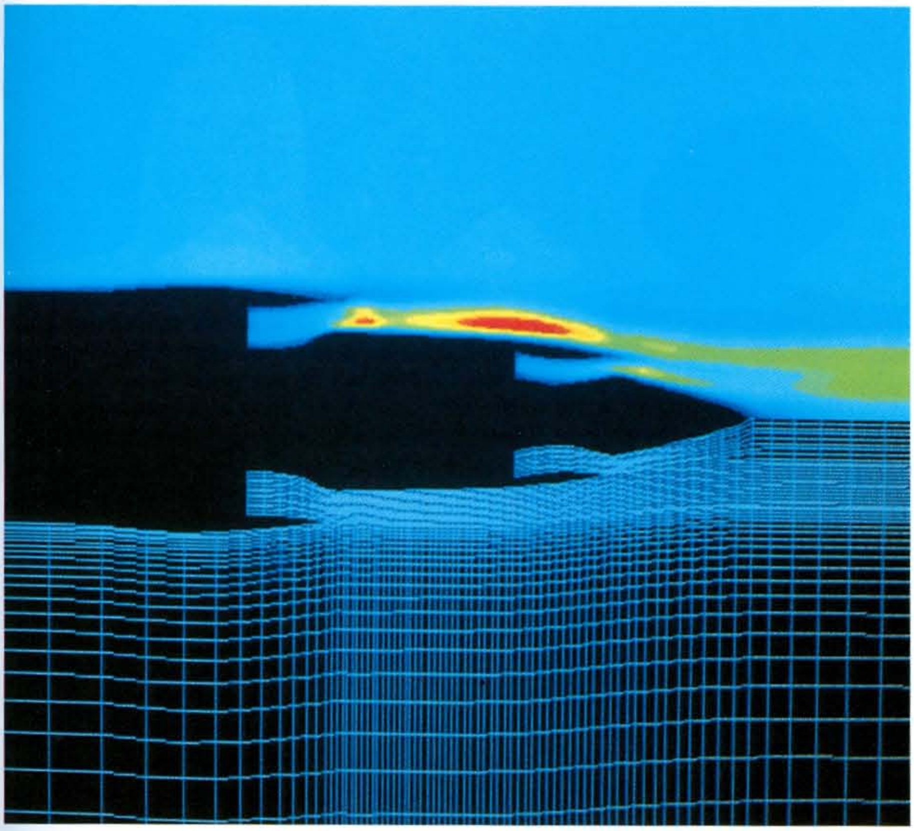
The PNOZ/2D code solves the two-dimensional/axisymmetric Navier-Stokes equations with swirl velocity. It also includes a turbulence model and several options for gas properties. The primary application is the simulation of multistream nozzle/afterbody/plume flow fields. PNOZ/2D is applicable to other flow problems as well, such as inlets, ejectors, and diffusers. An efficient fully implicit finite-volume solution procedure is especially tailored to solving the Navier-Stokes equations in a stacked multizone computational mesh. Each mesh zone is a body-fitted finite-volume grid. In the example problem each zone encompasses a separate fluid stream — primary exhaust, fan stream, and external flow stream. "The several choices of boundary conditions enable users to compute subsonic, transonic, and supersonic flows with mixing layers, boundary layers, shock waves, and flow separations on the CRAY X-MP system in a few minutes," said Mike Peery, president of Amtec Engineering.

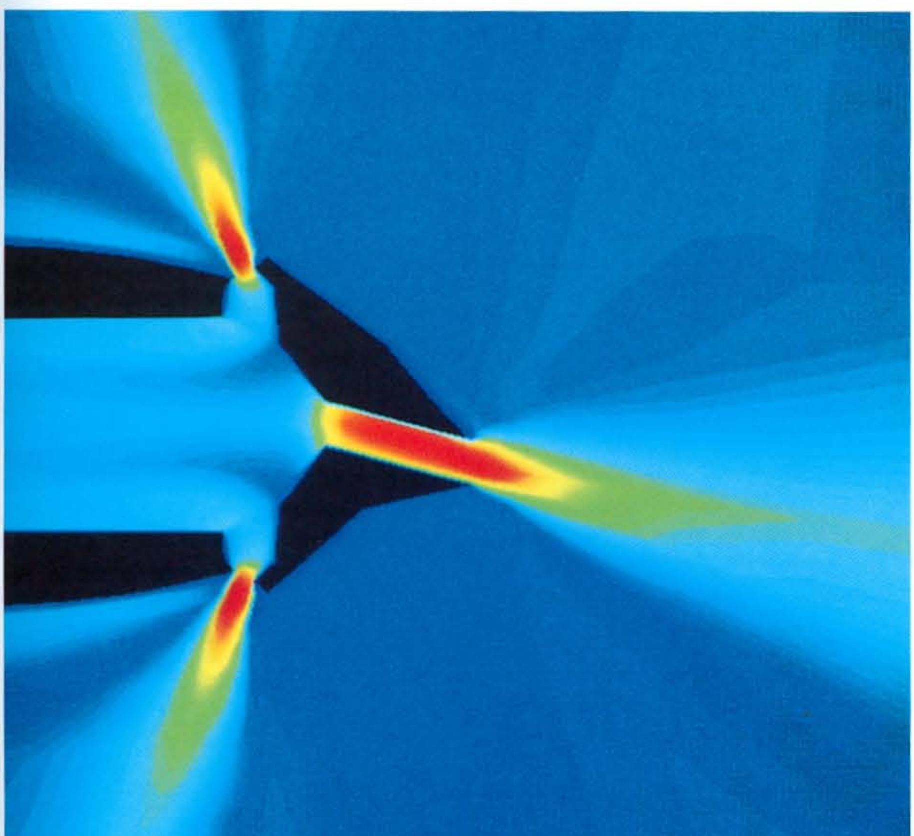
INCA/3D

A general three-dimensional extension of the multizone solution procedure of PNOZ/2D is used in the INCA/3D code to solve the full three-dimensional Navier-Stokes equations. Unlike the PNOZ/2D code, the mesh zones in INCA/3D are not restricted to any particular orientation or mutual connectivity. With custom "DRIVERS" that provide the necessary zonal connectivity, the INCA/3D code can be used to solve many complex two- and three-dimensional problems efficiently with a minimum requirement of input data. Standard DRIVERS are available for many flow problems from conic bodies to three-dimensional nozzles. A data base manager in INCA/3D automatically controls the storage of zone field variable data — whether it is located in RAM, disk storage, or Cray Research's SSD storage device. Multizone finite-volume mesh construction allows efficient finite-difference solution procedures to be used for problems with very complex geometries.

ZEUS/2D

Unsteady inviscid gas dynamic flows about objects having complex geometries can be simulated with the ZEUS/2D code. An explicit time-integration procedure is used to solve the two-dimensional/axisymmetric Euler equation with options for accelerating solution convergence to steady state. Flow fields containing shock waves, expansions, contact surfaces, and embedded vorticity can be calculated.





APPL/3D

The three-dimensional compressible parabolic/hyperbolic Navier-Stokes equations, including a two-equation turbulence model and transport equations for chemical species are solved by the APPL/3D code for simulating three-dimensional plumes, jets, and wakes. The equations have been transformed to allow the use of general curvilinear boundary-fitted computational meshes. An efficient space-marching procedure is used to advance the solution in the direction of the plume centerline. APPL/3D is applicable to subsonic and supersonic plumes in a subsonic freestream and supersonic plumes in a supersonic freestream. Plumes at an angle of attack relative to the freestream can also be modeled.

SPEAR/3D

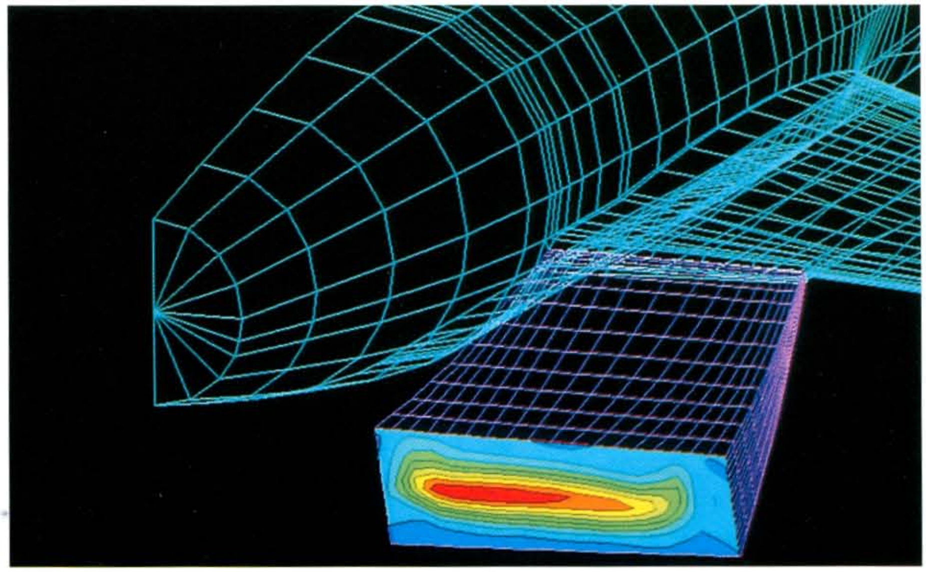
The SPEAR/3D code solves the three-dimensional compressible parabolized Navier-Stokes equations for external and internal supersonic viscous flows with thin wall layers of subsonic flow. The code is principally applicable to the analysis of two- and three-dimensional supersonic and hypersonic aircraft inlet and forebody flow fields. A general transformation of the governing equation allows the use of curvilinear boundary-fitted computational meshes. An option is available to couple this set of equations with an elliptic relation for the three-dimensional pressure field to yield steady-state elliptic solutions of the Navier-Stokes equations. The solution procedure iterates between marching passes through the solution domain and

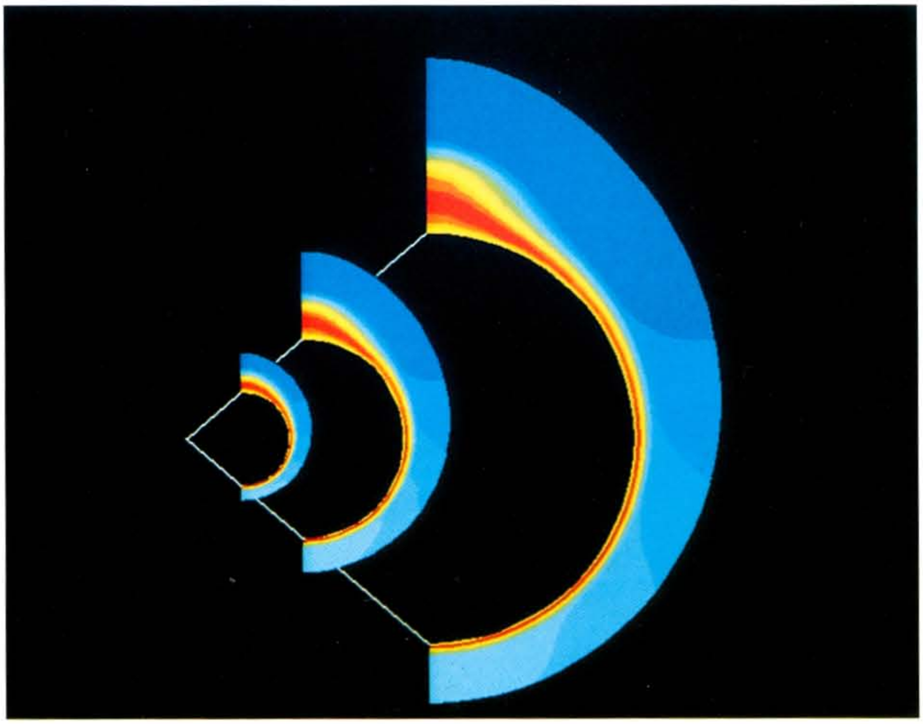
Top, calculated Mach number contours and computational mesh for an axisymmetric scale-model turbofan exhaust nozzle computed with PNOZ/2D. The freestream Mach number is 0.85, fan and primary nozzle pressure ratios are approximately 2.5.

Bottom, calculated Mach number contours for a partially deployed two-dimensional thrust reversing and thrust vectoring nozzle computed with INCA/3D. The freestream conditions are quiescent and the nozzle pressure ratio is approximately 2.0.

updates to the three-dimensional pressure field. Normal and oblique shocks are captured as part of the steady-state solution. Internal regions may change from supersonic to subsonic, boundary layer bleed may be imposed, ideal gas or real gas properties may be specified, and external bow shock fitting may be performed.

The understanding and interpretation of computed flow fields typically requires some form of graphic display. The images shown here were created using Amtec's interactive graphics program TECPLOT. For more information on using these CFD codes with Cray computer systems, contact Amtec Engineering, Inc., 10001 N.E. Fourth Street, Bellevue, WA 98004; telephone (206) 454-8060.





Top, calculated density contours 70 microseconds after rocket-assisted projectile leaves gun barrel. The calculation, made with ZEUS/2D, is time-dependent and axisymmetric. At time zero, the projectile leaves the barrel at near ambient sonic speed. A strong shock wave bellows out behind the projectile caused by the high-pressure gas (200 atmospheres) in the gun barrel. Two muzzle brakes outside the gun muzzle reduce recoil and muzzle flash.

Middle, a three-dimensional vectored exhaust plume from a high aspect ratio nozzle integrated into an airframe modeled with APPL/3D. The inviscid flow about the airplane and around the viscous plume was simultaneously computed using a panel method. The resulting calculation shows the plume trajectory and "roll up" and provides the influence of the exhaust plume on the airplane aerodynamics. This calculation was computed on a CRAY X-MP computer system using the SSD storage device and required approximately one CPU hour. The panel code required 98 percent of that time.

Bottom, calculated temperature contours (in cross planes) about an 8° conic body traveling at Mach 8 in laminar flow computed with SPEAR/3D. The computational mesh is fit to the bow shock and the surface is cooled.

USER NEWS

"Hard Woman" has no hard edges

No, that's not a neon apparition stalking your television set. It's the latest generation of graphics wizardry created by Digital Productions for Mick Jagger's "Hard Woman" music video.

If you've ever tuned in to MTV, Friday Night Videos, or any other popular music video programs, you may have seen the innovative images in this new production. In the video, a computer-generated "Vector Woman" dances with Mick through an elaborate montage of simulated plants, furniture, and leering skeletons.

According to John Whitney, Jr., president of Digital Productions, the problem with combining live action (such as Jagger's) with simulated figures is that the action of computer images appears cold and erratic next to the fluid movement of human characters. The "Hard Woman" video, by contrast, features simulated images that move smoothly — creating the illusion of near-human movement on the screen.

"We didn't film or directly digitize human behavior," Whitney said. "The fluid movement of the characters in the video was really a result of the sensitivity of the techniques employed by the designers. Their artistry, and the interactive capabilities of our software, actually allowed the creation of three-dimensional images that modeled human movement." Whitney added that the sheer power of Digital's CRAY X-MP computer system makes that kind of creativity possible.

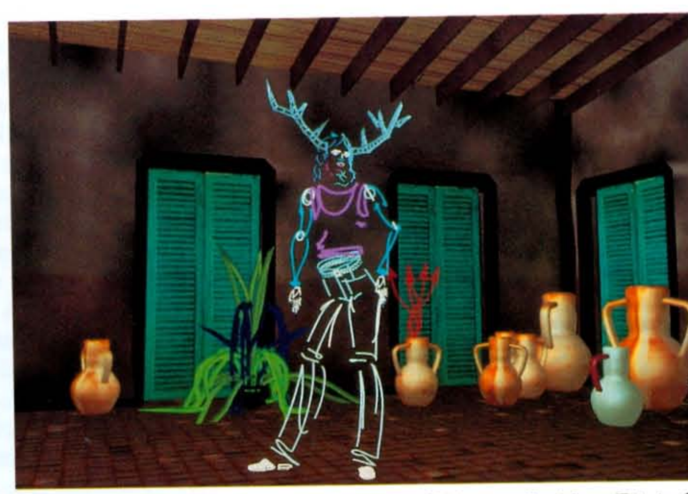
Digital's designers created forty-nine scenes, each involving several million calculations to establish each point of light, or pixel, in the image. Making the figures move with any degree of realism required constant recalculation at speeds that would be unimaginable without a supercomputer. Once Whitney's team was satisfied with the simulations, Digital Film Printing software was used to scan the simulated footage and merge it with live images from computer memory. The resulting composite frames were then refined and dubbed with a soundtrack to create the final "Hard Woman" video.

"We're beginning to change people's perceptions about computer graphics," Whitney concluded. "The old stereotype was one of hard-edged, cold imagery — but you won't find any hard edges in this piece."

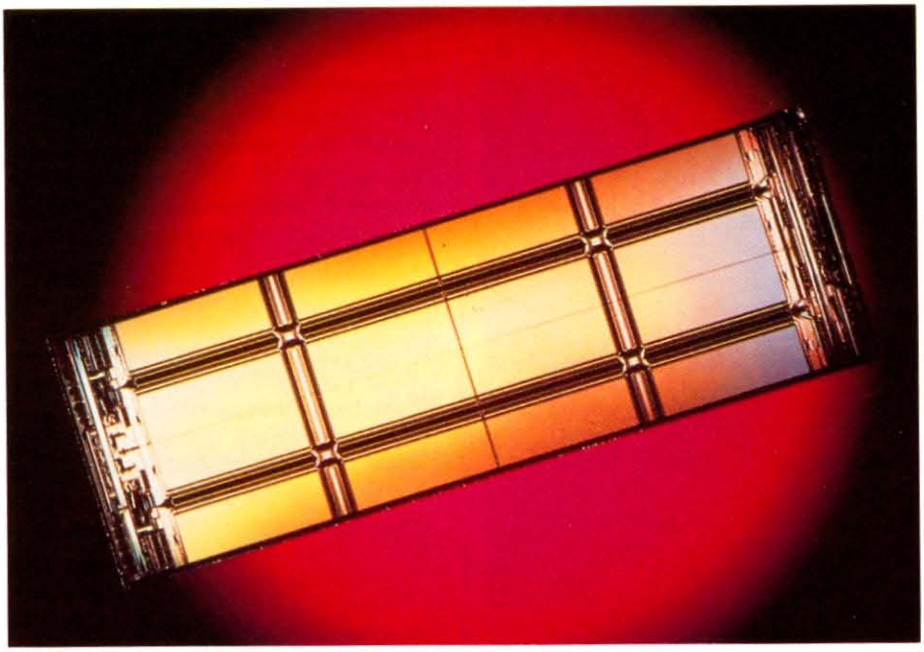
Cray systems help Bell Labs ring in success

Late in 1985 Cray Research announced the installation of a CRAY-1/S computer system at a customer site dedicated to semiconductor research and development. Early this year Cray Research announced the sale of a CRAY X-MP computer system to a manufacturer of personal computers. These events signaled an exciting expansion of Cray computer systems into electronic design applications.

However, one of Cray Research's earliest customers has applied the computing power of Cray systems to the design of electronic components since 1979. AT&T Bell Laboratories, in Murray Hill, New Jersey, received its first Cray system, a CRAY-1 computer system, in that year and has since built much of its computer-



Characters computed by Digital Productions' CRAY X-MP computer system star in Mick Jagger's "Hard Woman" video. Digital Scene Simulation (sm) by Digital Productions, Los Angeles, CA for HARD WOMAN/Promotone, B.V. © Copyright, 1985. All rights reserved.



AT&T's new megabit computer memory chip, which can store over one million digital bits of information.

aided design (CAD) work around the CRAY-1 and a newly acquired CRAY X-MP computer system.

"CAD is vital to our work; there is really no other way to design the complex devices we're working with today," explains David Beecham, head of Bell Labs' memory design department. Beecham notes that the labs' CRAY-1 computer system was an important tool in designing a recent breakthough, the megabit chip, as well as the preceding 256K DRAM. "As far back as the 64K DRAM, the CRAY-1 computer system has been invaluable," Beecham comments.

The process of computer-aided circuit design has three components, each of which involves a discrete computer modeling application. The first component, *process simulation*, models the diffusion of dopant ions through a substrate material, typically silicon. Dopant ions diffuse through a substrate at varying rates that can be controlled with temperature adjustments. The diffusion creates new compounds with desirable electrical properties within the substrate material. Although the underlying physical mechanisms of this process are not entirely understood, process simulation can provide enough information to be useful to engineers.

The output from a process simulation is a profile of the dopant distribution through the substrate. This information is used in the second step of the circuit design process, *device simulation*, in which the process model is modified to include metal contacts, and the electrical properties of the resulting device are computed.

"The semiconductor problem is a challenge because of the differences in scale we deal with," explains R. Kent Smith of the advanced VLSI development laboratory at Bell Labs. "Although the overall device might be on the scale of microns, we have to resolve features down to 100 angstroms. Furthermore, the model might have to accommodate a change in carrier density of 20 orders of magnitude over a few grid points. The Cray computers' large memories are a necessity for this kind of work."

Traditionally, devices were so large that one-dimensional models of electron travel paths provided engineers with sufficient information about a device's performance. But as engineers designed smaller circuits, two-dimensional electrical effects in devices had to be addressed, spurring development of two-dimensional modeling codes. "Ideally, we would like to isolate the devices," Smith comments, "but the need for smaller circuits requires that they be packed close together. The latest circuits require such close packing that we have to be concerned with parasitic effects, that is, how one device's operation can affect its neighbors."

Smith adds, however, that three-dimensional effects are already becoming significant and future models will have to address them as well. "This is going to require some novel approaches to software because we can't simply extrapolate what we're doing now to three dimensions," he says. "It is also going to require even more powerful computing hardware. We currently are solving large sets of partial differential equations that are very CPU- and memory-intensive. For three-dimensional codes we will have to create new numerical techniques and we will need about 200 million words of memory. The math required is quite sophisticated; the equations are very nonlinear and strongly coupled."

But the accelerating miniaturization of devices leads to even greater design challenges than the need to address higher dimensions. Ultimately it leads to the murky realm of quantum mechanics. "Classical physics works fine for the time being, but in future devices quantum effects become important. We're right on the edge of that now," Smith concedes. "We really have to start rethinking the whole thing."

The third tier of the circuit design process involves modeling many interconnected devices and observing the behavior of the resulting circuit. Such a simulation requires evaluating the proposed circuit in terms of its inductance, resistance, and capacitance, then calculating the ways in which these properties affect the circuit's performance. All design proposals must be evaluated with an eye on customer needs and current manufacturing capabilities. As an additional constraint, integrated circuits have to withstand a range of use and abuse conditions in the real world.

Beecham notes that such constraints are incorporated into Bell Labs' design simulations.

Unlike simulation of logic devices, memory circuit simulation is a transient analysis carried out on a very large matrix with many thousands of nodes. The complexity of today's electronic circuits poses problems of precision for circuit designers that are staggering by conventional standards. "It's extremely difficult to get a small signal, with a charge on the order of 30-35 femtofarads into a cell and later to locate and retrieve it. The work is very exacting." Beecham explained. (A femtofarad is one quadrillionth — 10^{-15} — of a farad, a standard unit of capacitance.) Bell Labs uses an advanced proprietary version of the circuit simulation program SPICE for circuit design modeling.

"The Cray computer provides an extremely cost-effective way to do this work," Beecham continued. "Not only can it handle the size of our problems, but it's also fast enough to provide a real cost-cutting advantage — no one has to kill time waiting for an answer. Using the Cray system might save us three to six months on a design project, which is quite significant for a one-year program. It gives us a tremendous advantage in the marketplace."

Computer modeling has become an integral tool for leading-edge electronic design efforts, such as those at AT&T Bell Laboratories, and Cray computer systems provide the fast turnaround and detailed solutions these modeling efforts require. The work at Bell Labs demonstrates that Cray computer systems can accelerate each step in the circuit design process. In addition to advances made at Bell Labs, work in electronic design using Cray computer systems is proceeding at other customer sites and at Cray Research, where Cray computer systems are helping design future generations of supercomputers.

CRAY-2 computer system takes a slice out of pi

It wasn't done as a lark or a publicity stunt. There were good reasons to calculate 29,360,000 digits of the mathematical constant pi using the CRAY-2 computer system recently installed at the NASA Ames Research Center in Mountain View, California. "Our main purpose in performing this computation was to thoroughly test the computer's reliability," explains Sterling Software's David H. Bailey, a consultant with the Numerical Aerodynamic Simulation program at NASA Ames. The record-breaking calculation not only established the system's reliability but also piqued the interest of mathematicians and, of course, added a paragraph to the history books.

For those rusty in geometry, pi is the ratio of a circle's circumference to its diameter. It appears not only in geometry calculations but also in many other branches of mathematics. The precise value of pi has been the subject of calculations ever since Archimedes first established its value to four decimal places in the third century B.C. It is now known that pi is irrational and transcendental; its decimal expansion never repeats, although it can be calculated to as many places as desired by means of various algorithms. For most applications, knowing pi to 10 or 15 digits is more than enough, with the knowledge of millions of digits having no immediate practical use.

But as a test of computer reliabilty, the pi calculation is hard to beat. "The calculation actually was performed twice using two different algorithms," Bailey notes. "I compared the results of the two calculations and they were in exact agreement for all but the last 24 digits, a normal round-off error. This implies that both computations, with a total of over 30 trillion arithmetic operations, were performed entirely without error."

"All facets of the computer's performance were tested," Bailey adds. "The airflow codes for which the system will primarily be used tend to be self-correcting, so that an error in execution might just slow the convergence to the correct result. But the pi calculation is almost completely unforgiving of error, so that if a single error occurred, the result would be completely wrong after an initial correct section. Since the results were in agreement, it follows that the CRAY-2 Fortran compiler correctly processed over 2000 lines of source code. In addition, the UNICOS operating system handled files over 100 million bytes long without error, and over 100 million words of main memory data were stored and fetched without corruption for nearly 30 hours."

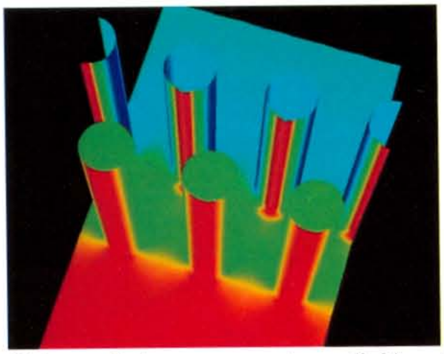
A secondary benefit of the calculation was the opportunity to test a new algorithm for pi. One of Bailey's two pi runs was performed with a new algorithm that had been devised just six months earlier by J. M. Borwein and P. B. Borwein, mathematicians at Dalhousie University in Halifax, Nova Scotia. Bailey's computation was the first implementation of this algorithm. The multi-precision routines used also employed some advanced algorithms never before implemented.

Another notable byproduct of the computation is a statistical analysis of the expansion of pi that it permitted. The analysis is being used to explore the unresolved question of the randomness of pi's digits. "Mathematicians are genuinely interested in this calculation because the ultimate nature of pi is a lingering mathematical problem," Bailey says. "The conjecture that the digits of pi are statistically random has so far resisted proof." Bailey's analysis has not yet revealed any statistical irregularities, so it appears the conjecture is sound. (And so is the CRAY-2 computer system.)

CRAY-2 computer system aids shuttle engine design

Engineers working to redesign the space shuttle orbiter's main engine will be referring to numerical studies performed on the CRAY-2 computer system at NASA's Numerical Aerodynamic Simulation (NAS) facility at Moffett Field, California. The studies involved modeling the flow field in the liquid oxygen post areas of the main injector in the orbiter's main engine. Previous runs on this problem had been made on the NAS CRAY X-MP/12 computer system

USER NEWS




Color-coded pressure contours (left) and particle traces (right) around space shuttle orbiter's main engine LOX posts. The solutions were calculated on the CRAY-2 computer system at NASA's Numerical Aerodynamic Simulation facility.

but were limited by the system's two-million-word memory. The most recent runs used over five million words of the CRAY-2 computer system's 256-million-word memory. (The engine being studied is contained within the orbiter; the studies are unrelated to January's shuttle loss.)

The physical problem modeled is the flow of hot pressurized hydrogen past hollow liquid oxygen (LOX) posts, which carry liquid oxygen to the main engines. NASA engineers want to understand the hydrogen flow field so they can redesign the engines to operate at higher power levels than current limits allow, enabling the orbiter to carry heavier payloads. The numerical studies were performed as part of a joint CFD effort between NASA Ames and Rocketdyne to ensure that the flow at the desired power levels does not cause severe fluid dynamic loading on the LOX posts. The studies, which were performed by Sterling Software's Stuart Rogers, a research scientist and consultant at the NASA facility, will be used in analyzing the flow around the posts.

The calculations modeled flow over a flat plate-post junction, with no-slip boundary conditions at the base and on the post. The top region had symmetry boundary conditions, and the sides had periodic boundary conditions. The grid was generated with an elliptic generator widely used by the CFD community at the NASA Ames Research Center. "The CRAY-2 computer was very easy to use for these applications, partly due to the UNICOS operating system, which allowed me to work closely with the machine," Rogers commented.

The studies were conducted using the program INS3D, a three-dimensional incompressible Navier-Stokes code. Further numerical modeling of the LOX post flow field will continue in the future, and Rogers is using the intervening time to refine the model. However, similar CFD studies under the NASA Ames-Rocketdyne joint program have already resulted in several design improvements in the orbiter's main engines. Additional improvements to be applied in upcoming years include decreasing the number of fuel transfer tubes running to the main engine from three to two, a direct result of INS3D calculations. This change is expected to contribute significantly to achieving the goal of a nine percent thrust increase over that of the original orbiter engine.

Los Alamos lends a hand to NSF centers

The Los Alamos National Laboratory has a long history of working with supercomputers. The Lab has often applied its computing capability — perhaps the most powerful in the world — to the toughest problems of science and technology. Over the years, Los Alamos has actually designed and built computers, developed networks that link computers to handle large amounts of complex information, and written software for sophisticated applications.

But while Los Alamos and other organizations forged ahead, university researchers were forced to get by with more limited computing power. "Most university computer systems were so slow that they discouraged university researchers from tackling the highly complex problems at the frontiers of science," said Los Alamos Lab Fellow Jack Worlton of Computing and Communications (C) division. As a result, a vast potential resource of theoretical knowledge and scientific creativity has gone largely untapped.

The National Science Foundation (NSF) hopes to change that. To date, the NSF has established five supercomputer centers at major universities across the country. Los Alamos is offering its expertise to help bring the supercomputers on line at Cornell and the Universities of Illinois, Minnesota, and California at San Diego.

Much of the work so far has been with the University of Illinois and the University of California at San Diego. Staff from C division at Los Alamos are providing the universities with two types of software: the Los Alamos version of the Cray Time-Sharing System (CTSS), and the Los Alamos Common File System, designed for efficient mass storage of data. The division copies its software tapes and sends them to the universities. Staff in C division are also collaborating with the staff at the University of Illinois to link workstations to the supercomputer, creating a facility similar to the Lab's Integrated Computing Network.

Los Alamos, in turn, hopes to gain new colleagues from many diverse scientific disciplines. "Our work with the NSF centers can give us a rapport with people who can share their knowledge and inject different ideas into Los Alamos research," said Bill Buzbee, deputy C division leader. All of the help they have provided may pay other dividends in years to come. "In the long run, I expect a certain number of graduate students (who have benefited from the supercomputing centers) to come to work at Los Alamos," Buzbee said. With university researchers already booking time on the computers, the collaboration between Los Alamos and the NSF could indeed bear much fruit.

WORLDWIDE BUSINESS CENTERS

Domestic

Corporate Headquarters
608 Second Avenue South
Minneapolis, MN 55402
(612) 333-5889, telex 290789

Government Marketing/ Government Relations Office
National Place
1331 Pennsylvania Avenue, Suite 1331 N
Washington, DC 20004
(202) 638-6000

Central Region Office
5350 Manhattan Circle
Boulder, CO 80303
(303) 499-3055
Albuquerque
Chicago
Detroit
Minneapolis/St. Paul
St. Louis

Eastern Region Office
11710 Beltsville Drive, Suite 500
Beltsville, MD 20705
(301) 595-5100
Atlanta
Boston
Huntsville
Laurel
Pittsburgh
Rochester
Tampa

Petroleum Region Office
5847 San Felipe, Suite 3000
Houston, TX 77057
(713) 975-8998
Dallas
Tulsa

Western Region Office
5776 Stoneridge Mall Road, Suite 350
Pleasanton, CA 94566
(415) 463-2800
Los Angeles
Seattle

International

Cray Canada, Inc.
4211 Yonge Street, Suite 210
Willowdale, Ontario
Canada M2P 2A9
(416) 229-2729, telex 2296238

Cray Research France S.A.
7 Rue de Tilsitt
75017 Paris, France
(01)-4766-01-55, telex 660568

Cray Research GmbH
Perhamerstrasse 31
8000 Munich 21, West Germany
(089)-56014-0, telex 5213211

Cray Research Japan Ltd
W 1154, Shin Aoyama Building
1-1 Minami-Aoyama 1-chome
Minato-Ku, Tokyo 107, Japan
(03)-403-3471, telex 28893

Cray Research S.R.L.
Via Vivaio, 15
20122 Milan, Italy
(02)-70-98-27, telex 315403

Cray Research (UK) Ltd.
Cray House
London Road, Bracknell
Berkshire RG12 2SY
United Kingdom
(0344)-485971, telex 848841

CRAY
RESEARCH, INC.