View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by NASA Technical Reports Server

3 1176 01308 5777

NASA Technical Memorandum 86774

NASA-TM-86774 19860003828

Computational Aspects of Zonal Algorithms for Solving the Compressible Navier-Stokes Equations in Three Dimensions

Terry L. Holst, Scott D. Thomas, Unver Kaynak, Karen L. Gundy, Jolen Flores, and Neal M. Chaderjian

FOR REFERENCE

NOT TO BE TAKEN FROM THIS ROOM

LIBRARY COPY

NOV 25 1985

LANGLEY RESEARCH CENTER
LIBRARY, NASA
HAMPTON, VIRGINIA

October 1985

NF00037

NASA
National Aeronautics and
Space Administration

NASA Technical Memorandum 86774

Computational Aspects of Zonal Algorithms for Solving the Compressible Navier-Stokes Equations in Three Dimensions

Terry L. Holst, Ames Research Center, Moffett Field, California
Scott D. Thomas, Informatics General Corporation, Palo Alto, California
Unver Kaynak, Stanford University, Stanford, California
Karen L. Gundy,
Jolen Flores,
Neal M. Chaderjian, Ames Research Center, Moffett Field, California

October 1985

NASA
National Aeronautics and
Space Administration

Ames Research Center
Moffett Field, California 94035

#1
N86-13296

COMPUTATIONAL ASPECTS OF ZONAL ALGORITHMS FOR SOLVING THE COMPRESSIBLE NAVIER-STOKES EQUATIONS IN THREE DIMENSIONS

Terry L. Holst*
NASA Ames Research Center, Moffett Field, California 94035

Scott D. Thomas*
Informatics General Corporation, Palo Alto, California 94303

Unver Kaynak**
Stanford University, Stanford, California 94305

Karen L. Gundy,* Jolen Flores* and Neal M. Chaderjian*
NASA Ames Research Center, Moffett Field, California 94035

Abstract

Transonic flow fields about wing geometries are computed using an Euler/Navier-Stokes approach in which the flow field is divided into several zones. The flow field immediately adjacent to the wing surface is resolved with fine grid zones and solved using a Navier-Stokes algorithm. Flow field regions removed from the wing are resolved with less-finely clustered grid zones and are solved with an Euler algorithm. Computational issues associated with this zonal approach, including data base management aspects, are discussed. Solutions are obtained that are in good agreement with experiment, including cases with significant wind tunnel wall effects. Additional cases with significant shock-induced separation on the upper wing surface are also presented.

Introduction

The literature indicates that research with the time-accurate Navier-Stokes formulation in three spatial dimensions has been limited. From about the mid-1970s, three-dimensional Navier-Stokes applications have been primarily applied to simple-geometry, high-speed flows. Some of these include the blunt-body application of Holst et al.,[1] the three-dimensional corner flow of Shang and Hankey,[2] the three-dimensional compression corner of Hung and MacCormack,[3] the swept-shock/boundary-layer interaction of Hung and MacCormack,[4] the hemisphere-cylinder application of Pulliam and Steger,[5] the flared-cylinder application of Hung,[6] and the oblique-shock/circular-cylinder flow of Hung.[7] More recently, a few applications involving more sophisticated geometries have appeared, including the transonic fuselage and forebody flows of Cosner,[8] the hypersonic wing-fuselage interaction of Shang,[9] the supersonic blunt-fin/wall interaction of Hung and Kordulla,[10] the high-subsonic turret flow simulation of Purohit et al.,[11] the transonic wing flow of Mansour,[12] the transonic afterbody flows of Deiwert and Rothmund[13] and Deiwert et al.,[14] the transonic wing flows of Agarwal and Deese,[15] the transonic forward-fuselage flow of Chaussee et al.,[16] the high-subsonic delta wing and low-supersonic, shuttle-like flows of Fujii and Kutler,[17] and the shuttle flow of Rizk and Shmuel.[18]

Most of these calculations utilized relatively coarse grids and/or required large amounts of computer time on even the fastest vector computers, such as the Cray X-MP or the Cyber 205. Nevertheless, the advent of these first pioneering calculations has set the stage for the aggressive assault on the problem of solving the Navier-Stokes equations about a complete aircraft.

* Research Scientist

** Graduate Student

Another area of investigation in the present study is the use of zonal techniques. Generating a single. well-behaved grid about a reasonably complete aircraft is indeed a difficult task. With the additional complications that the grid should contain appropriate clustering normal to all no-slip surfaces, as well as enough generality to allow application to more than one configuration, the task takes on enormous proportions. This fact has probably been the single biggest reason for slow development of three-dimensional CFD applications about complex geometries (Chapman[19]). One method that could improve this situation is the use of zonal grid techniques. With this approach. grid zones can be added or deleted as the configuration changes. coarse or fine as the flow field dictates. or viscous or inviscid as required. In a sense. the grid can be adapted to the expected solution. In addition. zonal schemes can be used to improve the data base organization. One grid zone including grid metrics and dependent flow variables can be processed in main memory while the majority of the data base resides on more-plentiful extended memory. Thus. larger grids can be supported with less main memory. Another big advantage. largely unexplored to date, is the use of "multitasking" within the context of a zonal scheme. Many new supercomputers involve the use of multiple processors. With a zonal scheme, one processor could operate on the first zone while another processor operates on the second zone.

Recently, the use of zonal or blocked grid approaches has become quite popular. Atta[20] and Atta and Vadyak[21] have used the blocked grid concept for full potential calculations. Benek et al.,[22] Hessenius and Pulliam,[23] Rai,[24] and Hessenius and Rai[25] have used different types of zonal grid approaches for solving the Euler equations. With these approaches, topologically complicated geometries have been solved with several grid zones which by themselves are not complicated. Therein lies the power of the zonal grid approach.

In this paper, Navier-Stokes solutions for the transonic flow over an isolated-wing geometry with both free-air and wind tunnel wall-boundary conditions are presented. This work represents a first step toward solving the flow about a complete aircraft and is the direction of the present research. The current computer code, called TNS (Transonic Navier-Stokes), utilizes a zonal approach with a total of four grid zones. The Euler equations are solved in the two outer zones. and the thin-layer Navier-Stokes equations are solved in the two inner zones near the wing surface. An H-mesh topology is used in both directions (chordwise and spanwise). Data base management aspects of the TNS zonal grid scheme are given special attention.

Governing Equations and Numerical Algorithm

The basic governing equations and numerical algorithm utilized by the TNS computer code, including the turbulence model. have been taken from the Pulliam-Steger ARC3D computer code.[5,26] Thus, the TNS code solves the Reynolds-averaged Navier-Stokes equations written in strong conservation law form. These equations are simplified by using the standard thin-layer approximation for the viscous terms. For zones in which viscous effects are not important, the Euler equations are used. The governing equations are transformed to the computational domain (ξ, η, ς) so as to preserve the strong conservation law form of the equations.

Two numerical algorithms have been investigated with the TNS computer code, an Alternating Direction Implicit (ADI) algorithm which solves block-tridiagonal matrices along each coordinate direction, and a diagonalized algorithm which solves a set of five scalar pentadiagonal matrices along each coordinate direction. The first algorithm is a variation of the Beam-Warming ADI scheme[27] due to Pulliam and Steger,[5] and the second is an extension of this scheme due to Pulliam and Chaussee.[28] In addition. several variations on the smoothing algorithm and the spatial time-step variation have been studied. Detailed information about these aspects and how they are affected by the present zonal scheme are discussed by Flores.[29]

All of the results presented in this paper have been computed using the diagonal algorithm with fourth-order implicit smoothing and a Jacobian-scaled time-step variation. This algorithm combination seemed to be the most computationally efficient of all the variations tested. The turbulence model used in the TNS computer program is the Baldwin-Lomax algebraic model.[30] This model is used because of the ease with which it can be implemented. Additional results with more sophisticated turbulence models are planned for the future.

Zonal Approach

Grid Generation

The grid-generation process begins with the generation of a single-zone base grid which includes the entire flow field. This grid contains no viscous clustering and has an H-mesh topology in both the spanwise and chordwise directions. It can be generated from either of two approaches: the elliptic solver approach of Sorenson[31] or the parabolic solver approach of Edwards.[32] Both of these approaches solve partial differential equations to generate smooth, well-behaved finite-difference meshes. They have the capability of generating suitable grids about isolated wing geometries with either free-air or wind tunnel wall outer boundaries.

Once the base grid is generated, a zoning algorithm is used to divide the grid into separate zones. The first grid zone (grid 1) is the base grid itself, with a small block of grid points near the wing removed. The second grid zone (grid 2) basically occupies the space left open by the block of points removed from grid 1 with a small region of overlap included (usually one or two grid cells on all boundaries). Grid 2 is constructed so as to contain twice as many grid points in each spatial direction as the original base grid. This refinement of grid 2 relative to the base grid is accomplished using cubic-spline interpolation. Similar to grid 1, grid 2 has a small block of points removed near the wing.

The final two grid zones (3 and 4) occupy the space left open by the block of points removed from grid 2, again with a small region of overlap. Grids 3 and 4 are constructed so as to contain the same number of points in both the spanwise and chordwise directions as grid 2. However, the grid points in the normal direction are highly clustered in order to capture viscous effects on the wing surfaces. Grid 3 is designed to capture the upper-wing-surface viscous effects and grid 4 the lower-wing-surface viscous effects. The two outer inviscid grid zones are topologically represented in the computational domain as cubes, with smaller cubes removed from the middle. The third and fourth viscous zones are topologically represented as simple cubes in the computational domain. This grid topology can be more adequately explained through an example.

Figure 1 shows a typical TNS grid plotted in perspective. Grid detail on the upper and lower free-stream boundaries, the inflow and outflow planes, and the wing symmetry plane are all visible. This grid, which is generated directly by the parabolic grid-generation approach, becomes the outer, coarse grid zone (grid 1). The grid detail near the wing/symmetry-plane juncture has been removed. It is in this region that grid zones 2, 3, and 4 are located. A blowup of the grid in the vicinity of the wing showing detail of grid zones 2 and 3 is displayed in Fig. 2. The wing geometry used in this case is composed of NACA 0012 cross sections, has a taper ratio of 1.0, 20° of leading edge sweep, and an aspect ratio of 3.0. This wing does not have any twist or dihedral. Note that the grid immediately adjacent to the wing surface (grid zone 3) is highly clustered in the normal direction and therefore is appropriate for a Navier-Stokes flow solver. Also note that this grid expands in thickness from the leading edge toward the trailing edge to better capture the growing boundary layer. Grid zone 4 (not shown) is designed to capture the viscous effects on the lower wing surface and is a mirror image of grid zone 3. The total number of grid points used in the grid of Figs. 1-2 is 166,621. The individual grid point breakdown for each zone is as follows: grid 1, 65 x 26 x 25 = 42,250; grid 2. 69 x 29 x 21 = 42,021; grid 3. 61 x 27 x 25 = 41.175: and grid 4, 61 x 27 x 25 = 41,175.

Data Base Management

Once the grid is generated and divided into the proper zones, the flow solver is initiated. The iteration procedure starts in the outer Euler block (grid 1), proceeds to the inner Euler block (grid 2), and ends with the two Navier-Stokes blocks: first the upper block (grid 3) and then the lower block (grid 4). Only one iteration using a spatially varying time step is completed in each grid zone before passing to the next. Only the flow-field solution (Q-arrays), transformation Jacobian (J), and the turbulence model arrays (when appropriate) associated with a single block reside in main memory of the Cray X-MP at one time. The information associated with the other blocks resides in extended storage. On the Cray X-MP this device is called the Solid State Device or SSD. The SSD is utilized functionally in the same manner as standard-rotating-disk extended storage, but is physically composed of much faster semiconductor memory. Use of the SSD instead of disk greatly reduces input/output (I/O) wait time, and for jobs that are normally I/O bound this is a significant advantage. The data base management system is outlined in Fig. 3.

The SSD installed with the Ames Cray X-MP has 16 million 64-bit words of memory. This can easily be extended to 32 million words if half-precision (32 bits) is used. The current version of the TNS code, with grid dimensions as outlined above, requires 5.8 million words of SSD storage. All arrays on the SSD are stored in 64-bit precision, with the exception of the metric arrays, which are stored with 32-bit precision. To allow more space in main memory, the metrics are shuffled into main memory from the SSD in two-dimensional planes as needed. This allows the maximum grid size of each zone to be about 50,000 points. Because the flow-solver algorithm used in TNS is an ADI-type algorithm with implicit sweeps in all three directions, the metrics must be transferred into main memory with three different orientations: x-y planes, x-z planes, and y-z planes. Thus, there are three different copies of the metrics stored on the SSD corresponding to the three different metric orientations. Because of the availability of so much storage on the SSD, this causes no problem and makes the overall memory management more efficient.

Each of the metric arrays is required in main memory several times for each grid zone during each iteration. This places extreme demands on I/O requirements. Nevertheless, the SSD handles these I/O requirements without problem. The computational statistics displayed in Table 1 demonstrate this for two cases ($M_\infty = 0.8$ and $M_\infty = 0.95$). The major emphasis in this table is on I/O statistics. The grid used for both calculations is the same as that presented in Figs. 1 and 2 and consists of four zones with a total of 166,621 grid points. As seen from Table 1, the higher Mach number case converges more slowly than the lower Mach number case. This is not surprising, since the $M_\infty = 0.80$ case is only moderately separated and the $M_\infty = 0.95$ case is massively separated. To advance a typical interior grid point in the TNS program one time level requires approximately 2030 floating-point operations. Thus, for the cases displayed in Table 1, the TNS program execution rate on the Cray X-MP computer is about 63 MFLOPS (million floating point operations per second).

From the statistics in Table 1, the efficiency level associated with the SSD I/O is quantitatively established. During the $M_\infty = 0.95$ case, the program transfers almost 24 billion 64-bit words between the SSD and the main memory using about a quarter of a million read/write requests. The I/O time charge for this case is just over 2 min. The estimated I/O time charge for this case, assuming a disk is used to replace the SSD, is over 24 hrs. (Because the I/O utilized in the TNS program is asynchronous, these I/O time charges represent only that portion of the I/O time that was not "covered up" by the CPU during execution.)

Table 1. Computational Statistics from the TNS Program
(NACA 0012 wing, $Re_c = 8 \times 10^7$, AR = 3.0, TR = 1.0, $\Lambda = 20°$)

Quantity	$M_\infty = 0.80$	$M_\infty = 0.95$
Iterations*	825	1890
CPU Time (Hrs)	1.25	2.82
Read/Write Requests (Memory/SSD)	110550	253260
64-Bit Words Transferred (MWDS)	10391	23806
SSD I/O Time (Sec)	61.5	140.8
Estimated Disk I/O Time (Hrs) (If disk had been used in place of SSD)	10.7	24.4

*Avg residual reduced three orders of magnitude (all grid zones).

Obtaining this level of I/O performance required an extensive investigation of the different types of I/O available. Such an investigation is summarized in Tables 2 and 3. Each column corresponds to a different mode of I/O defined as follows: (1) memory to memory (which does not actually involve extended-storage-to-memory I/O, but is included for the sake of comparison since it represents the theoretical optimum); (2) SSD to memory (unblocked); (3) SSD to memory (unblocked, using special routines called SHRINK and EXPAND which allows 32-bit word transfers); (4) SSD to memory (blocked); (5) disk to memory (unblocked); (6) SSD to memory (blocked, synchronous); (7) SSD to memory (blocked, synchronous); (8) disk to memory (blocked, synchronous); and (9) disk to memory (blocked, synchronous, formated). All I/O modes are asynchronous and unformatted unless otherwise specified. Note that all I/O modes investigated are for input only. The I/O times for output are similar to those for input and are not included for the sake of simplicity.

Table 2. Efficiencies for several I/O strategies, measurements in terms of CPU time normalized by the CPU time required for asynchronous (BUFFER IN), unblocked, SSD-to-memory I/O (case 2).

Transfer Size	Case number								
N	1	2	3	4	5	6	7	8	9
1	.15	1.0	1.4	.9	1.0	34	11	34	250
5	.63	1.0	1.7	.9	1.0	38	11	38	1240
10	1.2	1.0	2.3	.9	1.0	45	11	45	2500
25	2.9	1.0	4.0	.9	1.0	63	11	63	6250
50	5.9	1.0	7.1	.9	1.0	91	10	91	12200
75	8.3	1.0	10	.9	1.0	123	10	122	18200
100	12	1.0	13	.9	1.0	154	10	152	24400
250	29	1.0	31	.9	1.0	333	10	333	62500
500	59	1.0	63	.9	1.0	625	10	625	120000
750	83	1.0	91	.9	1.0	910	10	910	181000

Synchronous I/O is implemented with a FORTRAN "READ/WRITE" statement and keeps the CPU busy for the duration of the operation. Asynchronous I/O is implemented with the "BUFFER IN/BUFFER OUT" statement (at least on the Cray installed at NASA Ames) and runs in the background, freeing the CPU for other work. In each case the same amount of wall time is spent performing I/O, but the CPU time can be drastically different.

Table 3. Efficiencies for several I/O strategies. measurements in terms of wall-clock time normalized by the wall-clock time required for asynchronous (BUFFER IN), unblocked. SSD-to-memory I/O (case 2).

Transfer Size	Case number								
N/512	1	2	3	4	5	6	7	8	9
1	.016	1.0	1.04	180	29	227	1790	2170	45
5	.067	1.0	1.06	180	29	233	1790	2170	145
10	.13	1.0	1.12	180	29	227	1890	2170	192
25	.19	1.0	.67	108	34	139	1140	1300	400
50	.39	1.0	.83	101	50	132	1050	1220	1000
75	.45	1.0	.83	83	50	110	910	1020	833
100	.52	1.0	.91	71	71	100	714	833	2330
250	.71	1.0	1.0	42	110	59	417	476	227
500	.83	1.0	1.0	24	130	37	233	270	1640
750	.83	1.0	1.0	17	130	29	161	185	1520

Data are arranged in blocks of 512 words on extended-storage devices on the Cray computer. Blocked I/O involves the transfer of these blocks between the extended storage device and a buffer in main memory. An additional memory-to-memory transfer to move the data between the main-memory buffer and the actual location in main memory where the data are needed is required with blocked I/O. Unblocked I/O eliminates the need for the main-memory buffer and the additional memory-to-memory transfer. Therefore, unblocked I/O is preferred and should be used whenever possible. Care should be taken, however, since unblocked I/O must involve arrays with an integer multiple of 512 words.

The entries in Tables 2 and 3 are normalized by unblocked, SSD-to-memory I/O (case 2) and are displayed over a range of transfer sizes N ranging from 512 to 384,000 words. The timings associated with Tables 2 and 3 are in terms of CPU time and wall-clock time, respectively. Good performance in both timing categories is necessary for efficient program execution, but the wall-clock times are probably more important because they dramatically affect job turnaround time. Charging algorithms, which vary from installation to installation, can drastically affect this point. All these statistics were obtained on the Cray X-MP Computer installed at NASA Ames Research Center operating in stand-alone mode. That is, the multijob environment normally maintained by the operating system was suspended so that only the timing program that produced the results shown in Tables 2 and 3 was in execution. Thus, the unpredictable behavior associated with the multijob mode was removed. This generally produced smoother and more reliable timing trends. Note that the case 9 results (formatted I/O) were run in the multijob mode. This is because timings for this I/O mode were too expensive in the stand-alone mode. The wall-clock times for this case are particularly "noisy" and should not be considered to be very accurate.

The columns are arranged (approximately) in the order of performance in terms of wall-clock time, with the first column representing the most efficient type of transfer and the last column representing the least efficient type. Note the wide range in performance. The last four columns are extremely inefficient, and should never be used for a large amount of I/O in an iterative mode. The I/O algorithm used in TNS is asynchronous, unblocked, SSD-to-memory/memory-to-SSD I/O (case 2). This algorithm is used for all unit numbers except the metric units where asynchronous, unblocked, SSD-to-memory I/O using EXPAND (32-bit precision, case 3) is used.

Zonal Interface Scheme

Communication between the blocks is achieved by means of an interpolation procedure. The grid zones are automatically (but carefully) constructed to overlap by a specific number of cells, usually one or two cells. Then, information required at the boundary of one zone is interpolated from the interior of another zone. Because the grid zones are carefully constructed from a base grid such that surfaces requiring interpolation are coincident, the interpolation is greatly simplified. The most complicated zonal interface boundary condition involves only a series of one-dimensional linear interpolations. At the beginning of each iteration for grid-zone N, boundary values for all grid-zone N interface planes are updated from an SSD-array called BCBUF (see Fig. 3). After the interface planes and any other standard boundary conditions are updated, the interior solution is updated by one iteration. Then, the appropriate interface planes in BCBUF are updated and stored back on SSD. This completes the iteration for grid-zone N and the algorithm proceeds to grid-zone N − 1. This interpolation procedure is described in more detail in Ref. 33.

Computed Results

Lifting Supercritical Case

The first case involves a transonic flow field about a NACA 0012 wing with an aspect ratio of 3.0, a taper ratio of 1.0, 20° of sweep, a Reynolds number based on chord of 8 million, 2° angle of attack, and a free-stream Mach number of 0.826. This wing has no twist or dihedral. The grid used for this calculation is very similar to the grid previously presented in Figs. 1 and 2 (except that wind tunnel walls corresponding to the experiment of Lockman and Seegmiller[34] have been modeled for the present case, see Ref. 33 for more detail on this wind tunnel wall grid). Pressure coefficient distributions at two spanwise locations ($2y/b = 0.25$ and 0.78) are compared in Fig. 4 with experimental data from Ref. 34. Results for the free-air calculation are also included. The shock position for the case with walls is in good agreement with the experimental shock position while the free-air shock position is too far upstream, as one would expect. Thus, modeling the wind tunnel walls for this calculation was quite important. Generally, the agreement between the wind tunnel wall case and experiment is better inboard ($2y/b = 0.25$) than it is outboard ($2y/b = 0.78$). In particular, the computed upper-surface shock strength at the outboard station is larger than that of the experiment. This is caused by a large boundary-layer separation in the experimental results at this semispan location that is not accurately reproduced by the computed results.

This situation is shown in Figs. 5 and 6 where computed wing-surface particle paths and the experimental oil flow of Lockman and Seegmiller[34] are displayed. Note that both results indicate a moderate amount of shock-induced separation at about midchord. The spanwise extent of the experimental separation is reasonably predicted by the computation, but the streamwise extent is underpredicted. Despite the difference in the size of the separation zone, the overall comparison is quite encouraging. The computed position of the separation line is in reasonably good agreement, as are many other other qualitative details of the flow field. It is interesting to note that if the wind tunnel walls are not simulated in this calculation, the computed flow field remains attached. Thus, the importance of simulating the wind tunnel walls is emphasized.

Massive Shock-Induced Boundary-Layer Separation Cases

The last set of solutions presented consists of a series of calculations with shock-induced boundary-layer separation. These cases were computed to ascertain the degree of robustness of the present algorithm and the ability of the present algorithm to compute changes in the upper-wing surface flow-field pattern. The wing geometry and surrounding grid used for this series of calculations are the same as those presented in Figs. 1 and 2. The angle of attack and Reynolds number have been arbitrarly chosen to be 5° and 80 million, respectively. Four cases corresponding to the free-stream Mach numbers of 0.80, 0.85, 0.90, and 0.95 have been computed. It is interesting to note that utilization of the wind tunnel wall boundaries, as described in the previous case, produced a "choked" solution, with a shock wave spanning the tunnel (at least for the higher Mach number cases). After several hundred iterations and a moderately converged calculation, the solution (as expected) diverged. This, of course, was a consequence of the "fixed" upstream boundary conditions forcing more mass flow through the tunnel than the choked condition would allow. Thus, all the results displayed from this series of calculations have been computed using free-air boundary conditions.

Computed particle paths on the upper wing surface for each of the four calculations are displayed in Fig. 7. Note that the separation region increases in size and that the upper-surface flow pattern changes dramatically as the free-stream Mach number is increased. At the lowest Mach number, the separation is relatively small. Along the separation line there exists a nodal critical point. At the next highest Mach number ($M_{\infty} = 0.85$), the separation line has grown and is terminated on the inboard side by a spiral node and on the outboard side by a standard node. A saddle point exists about halfway between these two nodes on the separation line.

The separation region for the cases involving the next two highest Mach numbers ($M_{\infty} = 0.9$ and 0.95) extends over a still larger portion of the wing upper surface. Again, these solutions contain several interesting flow-field features as indicated in Fig. 7c and 7d. The $M_{\infty} = 0.9$ case has a node critical point at the inboard side of the separation line and a spiral node at the outboard side. The $M_{\infty} = 0.95$ case has node-like critical points at both ends of the separation line. For these two higher Mach number solutions, it is expected that unsteadiness effects may be important; however, none were observed. Since the scheme used for these calculations assumed a steady solution: that is, it uses a spatially varying time step, unsteadiness may have been unphysically supressed. This aspect will be investigated in the future.

The position of the separation region relative to the zonal interface boundary is best displayed by plotting particle paths constrained to lie in spanwise, cross-sectional planes. Such a plot is displayed in Fig. 8 for the $M_{\infty} = 0.9$ case taken from a span station of $2y/b = 0.66$. This span station approximately corresponds to a cut through the separation saddle point ($k = 13$). At this station the separation region is large and easily extends above the zonal boundary from the Navier-Stokes region into the Euler region. The particle paths pass smoothly across the interface boundary with no function or slope discontinuities. Thus, despite the existence of strong gradients across the explicitly updated interface boundary, the present approach is capable of prediciting a stable solution that is reasonably free from interface boundary influence.

Concluding Remarks

Transonic flow fields about wing geometries have been computed using an Euler/Navier-Stokes formulation. In the present approach, the flow domain is divided into several zones or blocks. This zonal approach offers several distinct advantages over nonzonal techniques. The flow field can be broken into grid zones that tend to adapt to the flow field in a beneficial way. That is, regions of the flow field with significant gradients can be resolved with dense grid zones, while regions in which the flow has small gradients can be resolved with coarse grid zones. In addition, use of the present zonal grid technique represents an effective data management scheme. This is especially effective for large application problems implemented on computers with relatively small main memories. One grid zone is processed in main memory while the other grid zones remain on extended storage. Finally, several I/O strategies for use with the present zonal scheme were presented and discussed. Having an efficient data base management scheme with fast I/O is imperative for the present zonal scheme. This approach will be used for simulating the flow about complicated configurations such as full fighter aircraft in the near future.

Acknowledgments

The authors would like to thank Drs. Thomas Pulliam and Joseph Steger of NASA Ames for numerous discussions during the course of this work. The second author would like to acknowledge support from NASA Ames through Contract Number NCA2-OR745-309.

References

[1] Holst. T. L.. Tannehill. J. C.. and Rakich. J. V., NASA SP-347. 1975, pp. 1457-1471.

[2] Shang. J. S. and Hankey. W. L.. AIAA Paper 77-169. Jan. 1977.

[3] Hung. C. M. and MacCormack. R. W.. AIAA Paper 77-694. June 1977.

[4] Hung. C. M. and MacCormack, R. W., AIAA J., Vol. 16. No. 10, Oct. 1978, pp. 1090-1096.

[5] Pulliam. T. H. and Steger. J. L., AIAA J., Vol. 18, No. 2, Feb. 1980, pp. 159-167.

[6] Hung. C. M.. AIAA Paper 79-1547, July 1979.

[7] Hung. C. M., AIAA Paper 82-0025, Jan. 1982.

[8] Cosner, R. R., AIAA Paper 82-0252. Jan. 1982.

[9] Shang. J. S., AIAA Paper 83-0225, Jan. 1983.

[10] Hung. C. M. and Kordulla, W.. AIAA Paper 83-1957. July 1983.

[11] Purohit. S. C., Shang, J. S., and Hankey, W. L., AIAA Paper 83-1738. July 1983.

[12] Mansour. N. N., AIAA Paper 84-0522, Jan. 1984.

[13] Deiwert, G. S. and Rothmund. H., AIAA Paper 83-1709. July 1983.

[14] Deiwert. G. S.. Andrews, A. E., and Nakahashi. K., AIAA Paper 84-1524. June 1984.

[15] Agarwal. R. K. and Deese, J. E., AIAA Paper 84-1551. June 1984.

[16] Chaussee, D. S., Buning, P. G., and Kirk, D. B., AIAA Paper 83-1785, July 1983.

[17] Fujii. K. and Kutler, P., AIAA Paper 84-1550. June 1984.

[18] Rizk, Y. and Shmuel, S.. AIAA Paper 85-0168, Jan. 1985.

[19] Chapman, D. R., AIAA J.. Vol. 17, No. 12, Dec. 1979, pp. 1293-1313.

[20] Atta, E. H., AIAA Paper 81-0382, Jan. 1981.

[21] Atta. E. H. and Vadyak. J. A.. AIAA Paper 82-1017, 1982.

[22] Benek, J. A.. Steger, J. L., and Dougherty, F. C., AIAA Paper 83-1944, July 1983.

[23] Hessenius, K. A. and Pulliam. T. H., AIAA Paper 82-0969. June 1982.

[24] Rai. M. M., AIAA Paper 84-0164, Jan. 1984.

[25] Hessenius, K. A. and Rai. M. M., AIAA Paper 84-1532, June 1984.

[26] Pulliam, T. H., Workshop, The University of Tennessee Space Institute, Tullahoma, Tenn., Mar. 12-16, 1984.

[27] Beam, R. and Warming, R. F., J. of Comp. Phys., Vol. 22, Sept. 1976, pp. 87-110.

[28] Pulliam. T. H. and Chaussee, D. S., J. of Comp. Phys., Vol. 39, Feb. 1981, pp. 347-363.

[29] Flores, J., AIAA Paper 85-1495, July 1985.

[30] Baldwin, B. S. and Lomax. H., AIAA Paper 78-257, Jan. 1978.

[31] Sorenson, R. L. and Steger. J., ASME Advances in Grid Generation, Vol. 5, June 1983, pp. 181-187.

[32] Edwards, T. A., AIAA Paper 85-0485, Jan. 1985.

[33] Holst, T. L., Kaynak. U., Gundy, K. L., Thomas, S. D., Flores, J., and Chaderjian, N. M.. AIAA Paper 85-1640, July 1985.

[34] Lockman. W. K. and Seegmiller. H. L., NASA TM-84367, June 1983.

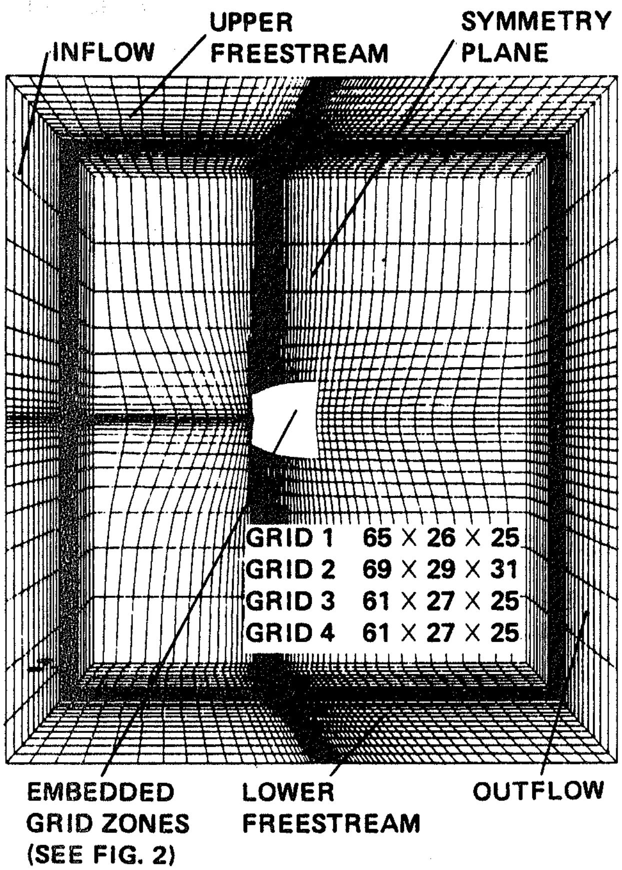


Fig. 1 Global finite-difference grid showing detail on outer boundaries.

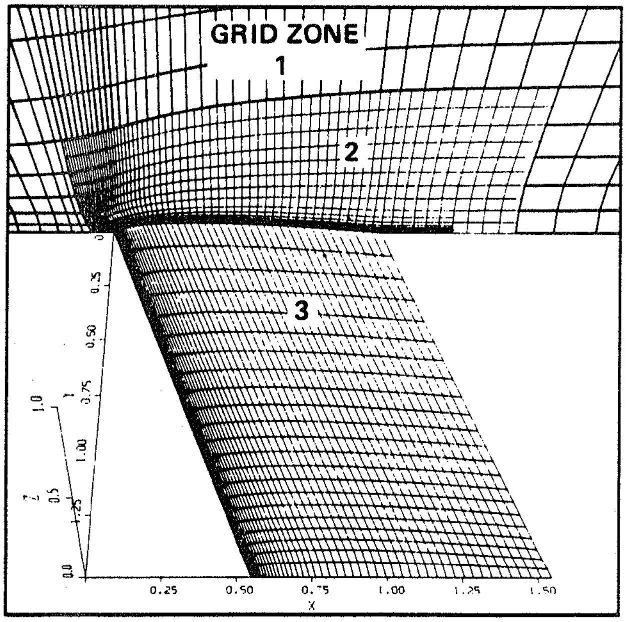


Fig. 2 Perspective view of embedded grid with upper symmetry plane ($y = 0, z > 0$) and wing surface highlighted.

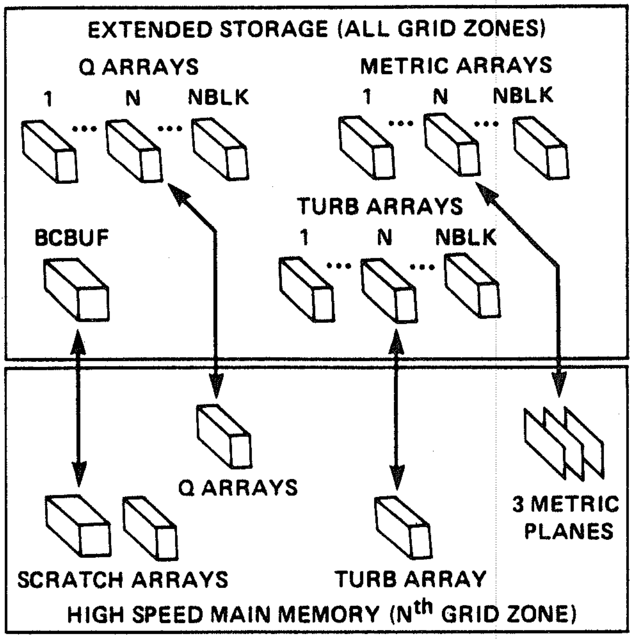


Fig. 3 Data base organization scheme for the TNS program.

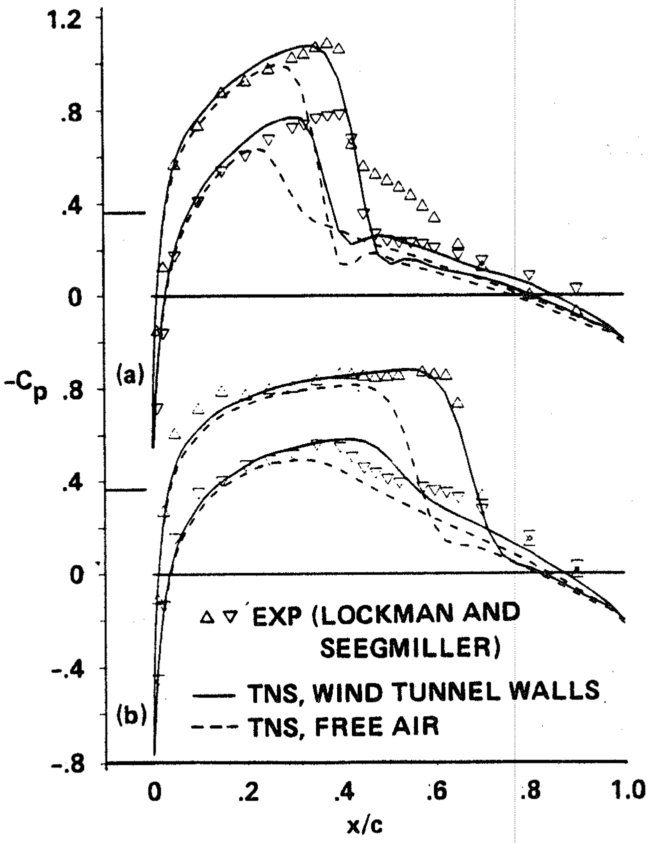


Fig. 4 Pressure coefficient comparisons: NACA 0012 airfoil section, $AR = 3.0$, $\Lambda_{LE} = 20°$, $TR = 1.0$, $M_\infty = 0.826$, $\alpha = 2°$, $Re = 8x10^6$, a) $2y/b = 0.25$; b) $2y/b = 0.78$.

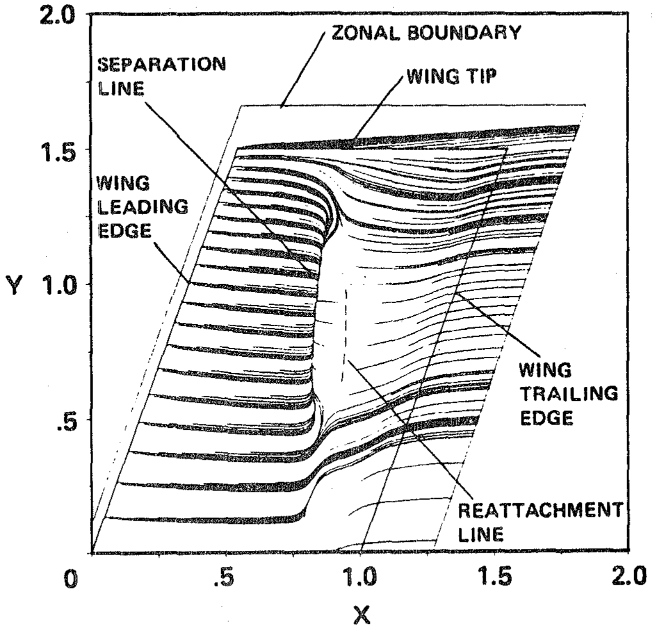


Fig. 5 Computed particle paths on the upper wing surface: NACA 0012 airfoil sections. $AR = 3.0$, $\Lambda_{LE} = 20^\circ$, $TR = 1.0$, $M_\infty = 0.826$, $\alpha = 2^\circ$, $Re = 8x10^6$.

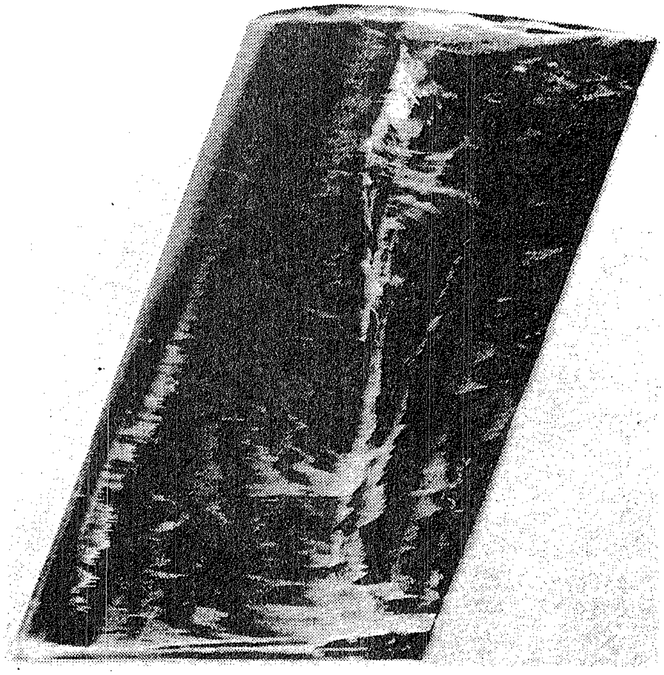
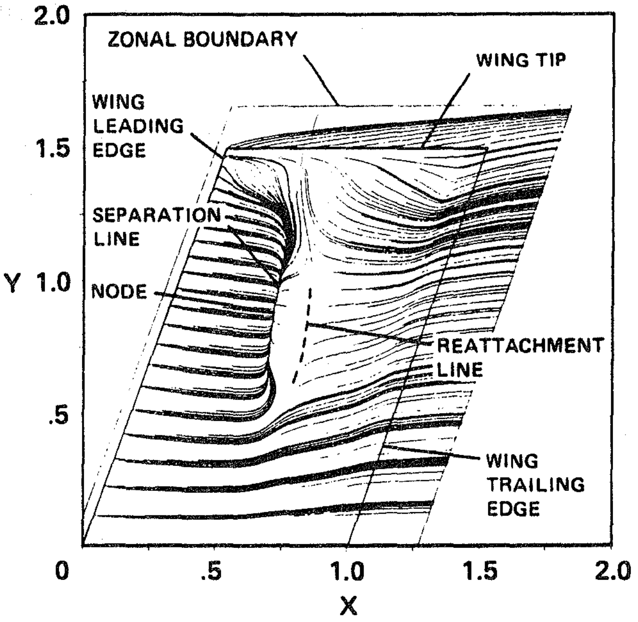
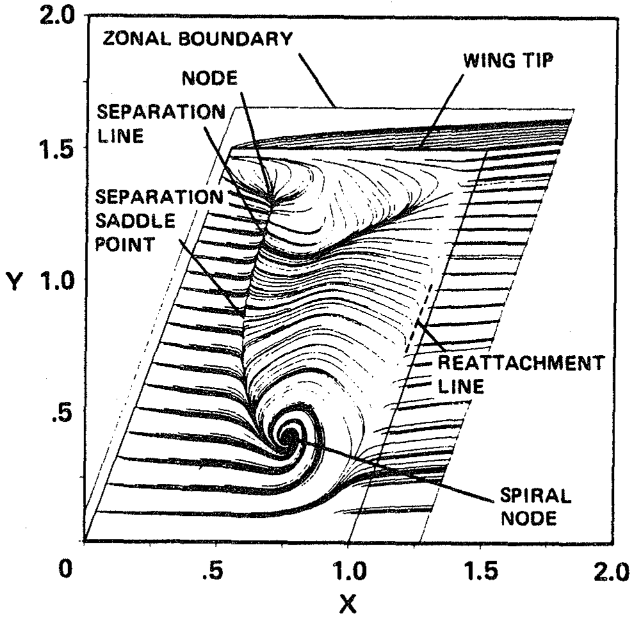


Fig. 6 Oil-flow pattern on upper wing surface: NACA 0012 airfoil section, $AR = 3.0$, $\Lambda_{LE} = 20^\circ$, $TR = 1.0$, $M_\infty = 0.826$, $\alpha = 2^\circ$, $Re = 8x10^6$. from Lockman and Seegmiller.[34]



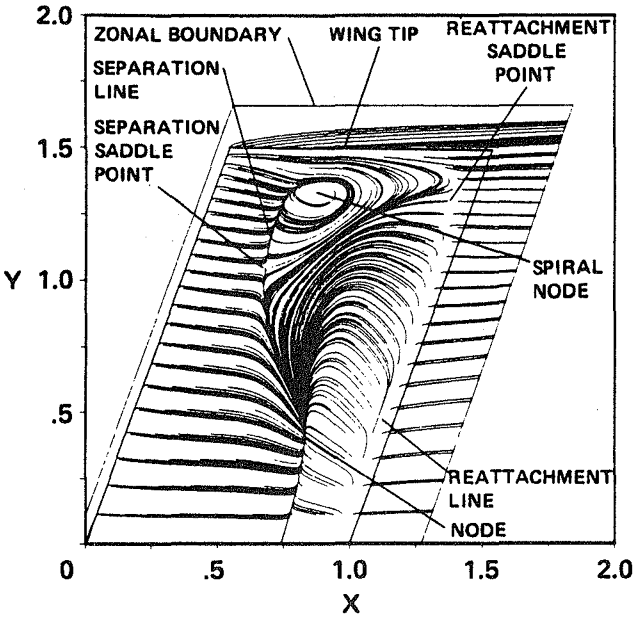
a) $M_\infty = 0.80$

Fig. 7 Computed particle paths on the upper wing surface: NACA 0012 airfoil sections, $AR = 3.0$, $\Lambda_{LE} = 20^\circ$, $TR = 1.0$, $\alpha = 5^\circ$, $Re = 8x10^6$.



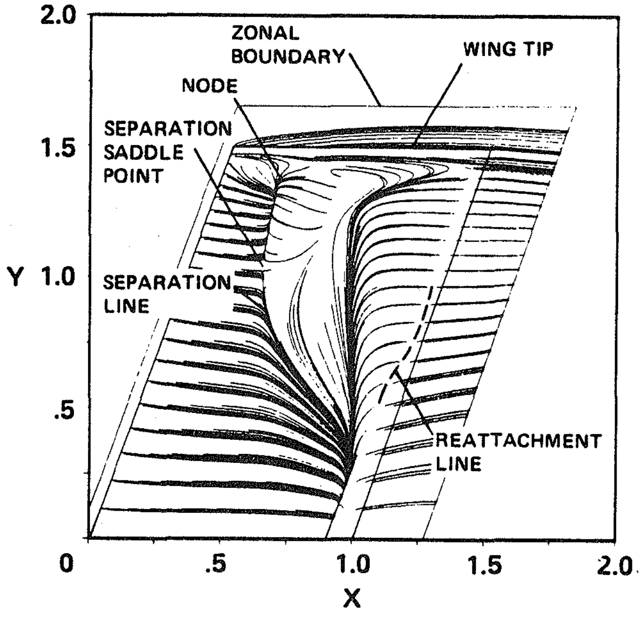
b) $M_\infty = 0.85$

Fig. 7 continued.



c) $M_\infty = 0.90$

Fig. 7 continued.



d) $M_\infty = 0.95$

Fig. 7 concluded.

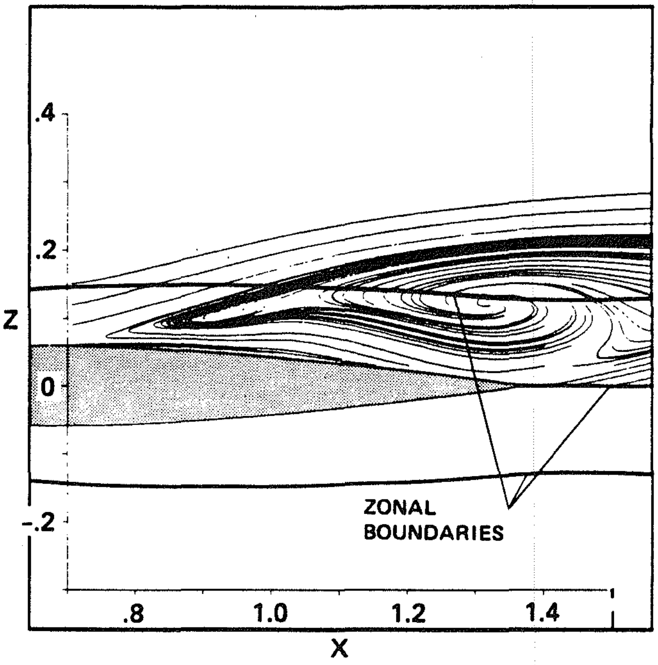


Fig. 8 Computed cross-sectional particle paths: NACA 0012 airfoil section, $AR = 3.0$, $\Lambda_{LE} = 20°$. $TR = 1.0$. $M_\infty = 0.90$, $\alpha = 5°$, $Re = 8x10^6$. $2y/b = 0.66$.

1. Report No.
NASA TM 86774

2. Government Accession No.

3. Recipient's Catalog No.

4. Title and Subtitle
Computational Aspects of Zonal Algorithms for Solving the Compressible Navier-Stokes Equations in Three Dimensions

5. Report Date
October 1985

6. Performing Organization Code

7. Author(s)
Terry L. Holst*, Scott D. Thomas+, Unver Kaynak**, Karen L. Gundy*, Jolen Flores*, and Neal Chaderjian*

8. Performing Organization Report No.
85340

9. Performing Organization Name and Address
*Ames Research Center, Moffett Field, California
+Informatics General Corp., Palo Alto, California
**Stanford University, Stanford, California

10. Work Unit No.

11. Contract or Grant No.

12. Sponsoring Agency Name and Address
National Aeronautics and Space Administration
Washington, DC, 20546

13. Type of Report and Period Covered
Technical Memorandum

14. Sponsoring Agency Code
505-31-01

15. Supplementary Notes

Point of contact: Terry L. Holst, Ames Research Center, MS 202A-14,
Moffett Field, CA 94035; (415)694-6329 or FTS 448 6239

16. Abstract

Transonic flow fields about wing geometries are computed using an Euler/Navier-Stokes approach in which the flow field is divided into several zones. The flow field immediately adjacent to the wing surface is resolved with fine grid zones and solved using a Navier-Stokes algorithm. Flow field regions removed from the wing are resolved with less-finely clustered grid zones and are solved with an Euler algorithm. Computational issues associated with this zonal approach, including data base management aspects, are discussed. Solutions are obtained that are in good agreement with experiment, including cases with significant wind tunnel wall effects. Additional cases with significant shock-induced separation on the upper wing surface are also presented.

17. Key Words (Suggested by Author(s))
Computational fluid dynamics
Compressible Navier-Stokes equations
Zonal algorithms
Euler/Navier-Stokes approach

18. Distribution Statement
Unlimited

Subject category: 02

19. Security Classif. (of this report)
Uncl.

20. Security Classif. (of this page)
Uncl.

21. No. of Pages
14

22. Price*
A02

*For sale by the National Technical Information Service, Springfield, Virginia 22161

End of Document