AIAA 98-0935
WIND: The Production Flow Solver of the NPARC Alliance

R. H Bush
The Boeing Company
St. Louis, Missouri
G. D. Power
Sverdrup Technology, Inc., AEDC Group
Arnold Engineering Development Center
Arnold Air Force Base, Tennessee
and
C. E. Towne
NASA Lewis Research Center
Cleveland, Ohio

36th Aerospace Sciences Meeting& Exhibit
January 12 - 15, 1998 / Reno, NV

WIND: The Production Flow Solver of the NPARC Alliance*

*R. H. Bush***
The Boeing Company
St. Louis, Mo 6t3166-0516

G. D. Power[†]
Sverdrup Technology, Inc., AEDC Group
Arnold Engineering Development Center
Arnold Air Force Base, TN 37389-9013

C. E. Towne[†]
NASA Lewis Research Center
Cleveland, OH 44135

Abstract

The NPARC Alliance is dedicated to providing an applications-oriented CFD tool for government, industry, and academic research and development. To meet this challenge, the WIND code has been developed, based on combining the capabilities of three existing CFD solvers used and developed in the government and industry sectors. WIND is a structured, multi-zone, compressible flow solver with flexible chemistry and turbulence models. Zonal interfaces may be abutting or overlapped, allowing the flexibility to treat complex systems moving relative to one another. WIND is part of a user-friendly, robust flow simulation system supported, developed, and validated by NPARC Alliance members and partners. This paper describes the approach taken in managing this joint software development project, an outline of the flow solver capabilities, a brief overview of the solution procedure, and an example simulation.

Background

The NPARC Alliance

The Arnold Engineering Development Center (AEDC) and the NASA Lewis Research Center (LeRC) have formed an alliance aimed at developing, validating, and supporting a computational system for aerospace flow simulation. The NPARC (National Project for Applications-oriented Research in CFD) Alliance is supported by and responsive to an association made up of users from government, industry, and academic institutions.

The NPARC Alliance began with an inlet CFD peer review held at NASA LeRC in early 1992 in which it was recommended that redundant CFD efforts be consolidated. At the same time, management and engineers at AEDC and NASA LeRC realized that there were complementary development efforts at each Center focused on the PARC CFD code.[1] The result was the creation of the NPARC Alliance.[2] The NPARC Alliance draws on the unique talents and qualifications of its partners while at the same time soliciting the experience and insights of government, industrial, and academic users to ensure that code development proceeds in a cost-effective, customer-responsive manner.

The NPARC code[3] was a result of combining the capabilities of the NASA LeRC version of the PARC code and the AEDC version and formally putting this software under version control. The NPARC Alliance provided the framework for jointly supporting, developing, and validating the NPARC code and related software. To serve the customers of AEDC and NASA LeRC a vision was developed:

The Computational Tool of Choice for Aerospace Flow Simulation.

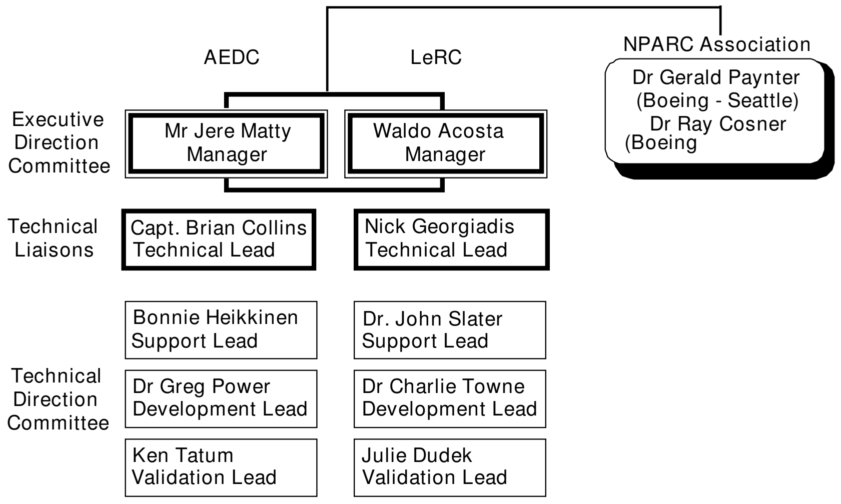
In order to achieve the vision and support the mission, the Alliance was structured to take advantage of each agency's strengths and abilities (Fig. 1).

The Executive Steering Committee consists of one manager from each Alliance partner (AEDC and LeRC). The Technical Liaisons lead the technical efforts at the two Centers supported by the Technical Direction Committee. This Committee is made up of one technical representative from each Center in the three functional areas of the alliance: Support, Development, and Validation.

* The research reported herein was performed by the Arnold Engineering Development Center (AEDC), Air Force Materiel Command. Work and analysis for this research were performed by personnel of Sverdrup Technology, Inc., AEDC Group, technical services contractor for AEDC. Further reproduction is authorized to satisfy needs of the U. S. Government.

** Associate Fellow, AIAA.

† Senior Member, AIAA.



AEDC
LeRC
NPARC Association
Dr Gerald Paynter (Boeing - Seattle) Dr Ray Cosner (Boeing
Executive Direction Committee
Mr Jere Matty Manager
Waldo Acosta Manager
Technical Liaisons
Capt. Brian Collins Technical Lead
Nick Georgiadis Technical Lead
Technical Direction Committee
Bonnie Heikkinen Support Lead
Dr. John Slater Support Lead
Dr Greg Power Development Lead
Dr Charlie Towne Development Lead
Ken Tatum Validation Lead
Julie Dudek Validation Lead

Fig. 1. NPARC Alliance organization.

The NPARC Association is a self-governing group consisting of government, industry, and academic users. Users of NPARC Alliance software are eligible to be an Association member. A Steering Committee, drawn from the NPARC Association, is chartered by the Alliance to formalize users' inputs and provide feedback to the Alliance regarding current and future code developments. The current Association Steering Committee is co-chaired by representatives from the Boeing Company. The NPARC Association plays a key role in providing user community feedback to the Alliance to ensure that NPARC software remains useable, current, and relevant.

A primary method for internal communication is the Annual Plans and Policy Document. This document outlines the current Vision and Mission Statements as well as the current organizational structure. The real substance of the document is the schedules for each of the three areas: development, validation, and support. These schedules are negotiated each year and define the expectations from each of the Alliance partners for the upcoming two years.

Further information on the NPARC Alliance is also available through the World Wide Web at: **http://info.arnold.af.mil/nparc**, or by e-mailing a request for information to *nparc-support@ info.arnold.af.mil*.

Joint Development – WIND

Prior to the merger of the McDonnell Douglas Corporation (MDC) and the Boeing Company, MDC in St. Louis offered to the NPARC Alliance the CFD technology in their primary flow solver NASTD[4] and associated software. In exchange, the software would be maintained and supported by the Alliance and MDC would benefit from leveraging future development resources. At nearly the same time, a modification in the operating contract at AEDC resulted in the merger of two separate CFD groups, propulsion CFD, co-developers of the NPARC code; and aeromechanics CFD, developers of the NXAIR code.

At the annual planning meeting of the NPARC Alliance, attended by government, industry, and academic representatives, the Alliance decided to take advantage of the capability and resources that had become available. A design committee, consisting of representatives of each organization, was formed to develop an approach to merging the capabilities of these three CFD tools. The committee decided to begin with one of the codes as a baseline and merge the capabilities of the other two codes. The NASTD code was chosen as the baseline code based on an estimation of resource requirements.

A list of capability requirements was developed based on the current and near-term foreseeable capabilities of each of the codes. The design team developed a plan to attain 90 percent capability merger within one year and 100% within two years, starting with about 70- to 80-percent capability within NASTD. Each organization agreed to expend its own development resources to attain this goal.

A joint development effort began with responsibilities and resources shared by the three principal participants, AEDC, NASA LeRC, and MDC (now

Boeing). This effort was bolstered by the selection of the NPARC Alliance software suite as a project in the Common HPC Software Support Initiative (CHSSI) of the High Performance Computing Modernization Program (HPCMP). As part of the CHSSI program, the Air Force Research Laboratory (AFRL) is contributing to the development effort. In the following section, the program management and the software management of this joint development project are described in more detail.

Development Environment

Program Management

The design team identified the major development efforts required to complete the capability merger with NASTD as the baseline. The responsibilities for accomplishing the development of the new WIND code are shared equally by the three principal organizations. The primary tasks and the associated responsible organization are:

- Program Management (AEDC)
- Software Management (Boeing)
- Documentation and Validation (NASA LeRC)
- Store Separation Integration (AEDC)
- Zonal Interface (Boeing)
- Turbulence Models (NASA LeRC)
- Framework (AEDC)
- Algorithm (Boeing)
- I/O Systems (NASA LeRC)
- Parallel Scalability (AFRL)

Each of the responsible organizations provides feedback to the program manager, who maintains a central database of progress and problems. Each of the organizations has a development lead who keeps in contact with the task managers responsible for each of the tasks above. The development leads from each organization have a face-to-face meeting quarterly and video teleconferences monthly to work out difficulties and assess the status of the project. The program manager provides monthly feedback to participants and management at each organization.

The key elements in a joint development project, particularly with organizations and individuals located at dispersed geographic locations, are planning, version control, standards, and communication. Of these, communication is arguably the most important. The Internet is used extensively to transmit information, including bug reports, suggestions, data sets, and documentation. A majordomo email system was established early in the program to facilitate global information exchange, and a WWW site was established for hyperlinked documentation that could be updated almost instantly.

A set of standards documents was developed as one of the first tasks. A document describing programming guidelines provides a set of required programming practices, as well as suggested practices for less critical items, such as formatting. A documentation standards document provides the developers guidelines on what information is required to document new features and a description of all documentation provided with NPARC Alliance software. Finally, the testing standards provide those doing validation and functional testing of Alliance software guidance on procedure, documentation, and archiving.

Software Management

The WIND development and release system was designed to provide support for multi-site, multi-platform software distribution and development from a central location. Files are provided to users and developers through four different “release sets” – application, tools, development, and build releases. WIND users may download the latest releases from a protected FTP site, and will soon be able to access the software from a dedicated site on the World Wide Web.

The application release includes run scripts and executables for specified platforms, providing only those files necessary to run WIND. The tools release provides auxiliary software supporting grid generation, boundary condition definition, and post-processing. The development release defines a structure in which developers may modify, compile, and link their own version of WIND prior to incorporating their changes into the primary version of the code. Finally, the build release is simply a collection of source code for WIND and all its related libraries, designed to provide a starting point for porting the code to unsupported platforms.

Conflicts may arise when developers at different sites make simultaneous changes to the primary source code. This problem is mitigated through the use of the Revision Control System (RCS), a GNU utility that provides locking and history tracking of text files. A number of special scripts incorporate RCS functions to provide custom revision control for WIND. Source code must be "locked" before changes may be made, which prevents multiple users from making simultaneous updates, or at least notifies them that others are working on the routines to be changed. A History file maintains a log of all changes to the code, and each RCS-controlled routine maintains a history of changes through all versions, allowing developers to easily retrieve a specified version number and build previous versions of WIND.

Self-Documenting Database

The Boeing Common File Format (CFF) is a third-generation self-documenting file format designed to be compact, quickly accessible, and machine portable. The internal format uses an N-tree hierarchy consisting of nodes. Each node then has pointers to other nodes or variables, which are identified by user-supplied character names. Since the file format contains character tags, it is also self-documenting.

Common File versions 1 and 2 use a FORTRAN binary direct access file to minimize the file size and to allow quick, direct access to nodes or variables. The current version, version 3, uses the ADF core developed by the Complex Grid Navier-Stokes (CGNS) project which was sponsored by NASA. The file is accessed through the CFF library, which contains all required functionality. Using this file format, the node and variable names were defined for CFD as well as which variables were contained in the grid and solution files. For more information see the Common File Programmers Guide accessible through the NPARC Web site.

The Common File is currently used throughout the CFD process from grid generation to flow solver to post processing. It allows better integration of tools within the flow simulation system and avoids unnecessary file conversions. Since CFF is self-documenting, smarter codes have been developed that can automatically build from existing data stored in the file, like the Common file post-processor, CFPOST. A wide variety of utilities have been developed for the common file in support of CFD, like CFSPLIT, which allows the user to break up zones into smaller ones modifying, the boundary conditions and zone connectivity data so that the split grid is ready to run.

Parallel Environment

The WIND code can run in parallel on a network of workstations or a large multiprocessor machine like the Silicon Graphics ORIGIN-2000 or the HP Exemplar. The parallel model is a master-worker implementation with PVM being used for communication and spawning of worker processes. When running in parallel, each zone is assigned to a CPU based on an algorithm in which the largest zone waiting to run is spawned to the first available processor. This algorithm accounts for differences in CPU speed and zone sizes.

There are two parallel modes available; the default spawns one process per zone, and the other spawns one process per CPU and assigns multiple zones for that process to run. The first mode is the most efficient, but requires enough memory to hold all of a worker's assigned zones in that system's core memory plus swap space (currently 1GB memory is required for 5 million grid points). The second mode minimizes the memory but adds the overhead of storing zones in a local scratch file as the CPU cycles through its assigned zones.

WIND normally only passes boundary data each cycle; the flow data are passed back to the master to save at user-specified checkpoint intervals. In order to make the code somewhat fault tolerant, if a worker dies, the master has the ability to restart the job by reassigning the process to another CPU and telling the other processes to go back to the last checkpoint. Returning to the last checkpoint ensures that the solutions in all zones are in equivalent states of evolution, minimizing the chances that the overall solution would diverge.

To run in parallel, the user creates a multi-processor control file (MPC) containing a list of hosts on which to run and parallel directives like checkpoint interval and parallel run mode. The user may specify the checkpoint interval by clock time or by number of cycles. The WIND run script and associated scripts set up the entire PVM virtual machine environment and run the job transparent to the user. The scripts check the remote machine to see if it is overloaded before adding it to the virtual machine so that the job will not be held up waiting for the overloaded machine to complete its tasks. The load limit can be set by a keyword in the MPC file on a machine-by-machine basis.

Documentation

Documentation for WIND is generally available in both PostScript (generated using LaTeX 2e and dvips, and intended for printing) and HTML form (intended for interactive use). Much of the WIND documentation is based on documentation originally developed at McDonnell Douglas Aerospace Company (now Boeing) for the NASTD code. The WIND documentation is available on the World Wide Web, through the NPARC Web site.

There are two levels of WIND documentation - one aimed at users and one at developers. The user-level documentation is what a typical user will probably refer to most often. It includes information that a user, from beginner to expert, will need to successfully apply the program to real-world problems. The developer-level documentation contains detailed reference information that a developer can refer to when modifying or extending the program.

User-Level Documentation – The principal user-level documentation is the WIND User' s Guide. It describes the operation and use of the WIND code, including a basic tutorial; the physical and numerical models that are used; the boundary conditions; convergence monitoring; diagnostic messages that may be printed; the files that are read and/or written; execution scripts and parallel operation; a complete list of input keywords and test options; and where to go for additional help.

Separate user' sguides are also available for GMAN and CFPOST. GMAN is an interactive menu-driven pre-processor that is used to specify boundary condition types and zonal connectivity in multi-zone grids. CFPOST is a post-processor that may be used to examine the contents of the Common Flow (.cfl) file created by WIND. It includes options to list and plot results, generate reports, and produce files for use by other post-processors.

User-level documentation is also available for several smaller utilities distributed with the WIND code. A separate "Guide for Translating Between NPARC and WIND Boundary Conditions" is available to help current NPARC users in transitioning to the new WIND code.

Developer-Level Documentation – The principal developer-level documentation for WIND is the WIND Programmer' sReference. A preliminary and incomplete version of this documentation is currently available, with the complete version scheduled for release with Version 2 of the WIND code in January 1999. The current version includes information about the program structures in the form of both a high-level conceptual calling tree and a detailed subprogram calling tree; lists of all the FORTRAN parameters and common variables, with definitions for some of the more important ones; and a list of all the subprograms, including a one-line summary of the subprogram' spurpose, the argument list, the name of the file the subprogram is in, a list of the subprograms that it calls and that call it, and a list of the common blocks that it includes. It does not yet include detailed descriptions of the work done in all the subprograms.

A Common File Programmer' s Guide that describes the common file structure and the library routines used to access and store information in common files is also available.

Code Capabilities

The three CFD simulation systems chosen to contribute to the NPARC Alliance system are very similar, but have evolved in different environments leading to unique and complementary capabilities. The original NPARC code was used extensively for internal flows such as inlets, nozzles, and test facilities, with an emphasis on usability and robustness. The NASTD code was developed in an aircraft manufacturer design environment with more emphasis on the entire system, rather than just the flow solver. The NXAIR code has been a primary workhorse for moving body simulations, particularly store separation, with requirements for time accuracy and fast turn-around.

The original design team developed a comprehensive list of required capabilities by focusing on the goal of 100-percent capability merger. Independent of the simulation capabilities, general requirements were established including standards documents, program and software management, and documentation, as discussed above.

Flow Simulation System

While the flow solver itself is the major component of a flow simulation system, the NPARC Alliance acknowledges the necessity to provide a complete system for its members and other users. The past strategy has been to provide a description of the flow solver's input and output capabilities and require that the user provide preprocessing, e.g. grid generation, and postprocessing, e.g. visualization, consistent with these capabilities.

To allow more flexibility and provide a more complete system, several requirements have been established. A self-documenting database (Common File) is the primary interface to all pre- and postprocessing tools. These tools are consistent with the flow solver models, allowing printing and viewing of primitive and derived variables, including loads, flow rates, and turbulence information. The flexibility and user accessibility to the Common File will continue to be improved in later versions.

A preprocessor for Grid MANagement (GMAN) is provided which includes a graphical user interface (GUI). GMAN has some grid generation capability, but the major use is for interactive setting of boundary conditions, hole cutting, and block interface connectivity. For backward compatibility to NPARC 3.0, a stand-alone program is provided to translate between NPARC files (restart and input) and the grid and solution Common Files.

The Common File POSTprocessing (CFPOST) package allows the user to manipulate, integrate, print, and plot results stored in the Common File database. While plotting capability is rudimentary, CFPOST can generate PLOT3D files containing user-specified variables in user-specified units, in addition to the standard conservation variables. The RESPLT program, in conjunction with CFPOST, provides the capability to plot convergence history, including residuals and integrated data.

The WIND Flow Simulator

The basic requirement for the NPARC Alliance flow solver is that it be applications-oriented, that is, it must be easy to use, flexible, and robust, with the capability to modify and add capability as application requirements warrant. The classes of applications currently being addressed by Alliance members and partners drive the overall code requirements. The types of applications which are analyzed include air-breathing engine inlets and nozzles, liquid and solid-propellant rocket propulsion systems, full aircraft systems, store separation, missile control systems, test cell and wind tunnel aerodynamics, and hypersonic vehicles.

The code's general capabilities include a single source code to treat two-dimensional, axisymmetric, variable- width two-dimensional and three-dimensional simulations. Efficient steady-state and time-accurate simulations are possible for all of these geometric approximations, through advanced algorithms, such as the global Newton approach and coarse-grain parallel processing.

Input/Output – In addition to the GUI preprocessor, the user interface is easy to use and intuitive through an English keyword parser. Flow-field conditions are currently input in English units, but plans are to allow user-specified units in the future. For tracking the solution, convergence monitoring includes not only the residuals, but also integrated data such as loads and flow rates. Most other output is accomplished through the CFPOST package.

Grid Zones – WIND uses a grid block or zonal approach with structured grids within each zone. Each zone is solved independently of all other zones, except for information exchange at block boundaries. The block boundaries can be either abutting, with point matching not required, or overlapping. Chimera-type overlapped zones can also be treated with hole-cutting to preclude solution within solid volumes. Zonal interfaces can also communicate within a single zone through rotationally symmetric interpolation.

Zonal boundaries can also represent modeled physical systems such as screens or actuator disks. Information is propagated to the boundary interface through the standard characteristic approach, then manipulated to represent the effect of the physical system on the flow-field quantities.

Numerical Schemes – To allow time-accurate simulation of moving bodies resulting in rotation, translation, or deformation of the grids, the unsteady grid metrics are calculated and included in the equations of motion. Time-accurate simulations are possible for a range of time scales. For high- frequency response, an explicit Runge-Kutta solver is available. For large time scales, a Global Newton algorithm has proven to provide time accuracy for large CFL numbers.[5]

The global Newton algorithm is an approach which stabilizes the solution and improves time accuracy by placing the entire unsteady transport equations on the right-hand side of the matrix solver and iterating within a time step over all of the zones. Thus, the interface boundaries are brought up to the new time level, along with the interior flow field, resulting in an essentially implicit treatment of the boundaries.

In addition to fast, time-accurate simulations, the global Newton algorithm has been shown to improve steady-state convergence.[6] Other methods are also available to improve steady-state convergence, including grid sequencing and local time stepping. The interaction of the global Newton algorithm with convergence acceleration techniques and all of the solver options has not been fully explored. The current recommendation is to use the point Jacobi matrix inversion method with the global Newton for constant time step and without any other acceleration technique.

Equations and Discretization – WIND includes the flexibility to solve various approximations of the full Navier-Stokes equations, including parabolized Navier-Stokes (PNS), Euler, and thin-layer Navier-Stokes (TLNS). The multi-species transport equations may also be solved. The additional species continuity equations are solved, fully coupled with the Navier-Stokes equations. Several turbulence model equations can be solved in an uncoupled mode. In general, the modular structure of the code is such that adding an additional or different set of equations to solve is fairly straightforward.

The equations in WIND are written in delta form using a node-centered finite-volume approach. Specification of the discretization of the equations of motion on the right-hand side is modular and flexible. The user may specify central difference, Coakley[7] upwind, standard Roe[8] upwind, or a physical Roe upwind for stretched grids.[9] Some of these options allow the user to adjust the spatial accuracy, through fifth order. Artificial viscosity can be imposed for any discretization scheme, if desired.

Boundary Conditions – Boundary condition types are specified using GMAN and are stored in the grid Common File. Flow conditions associated with a particular boundary condition, e.g., free-stream inflow/outflow, are specified in the input data file. All boundary conditions can be imposed explicitly, and surface boundary conditions can be imposed implicitly. As mentioned above, the global Newton algorithm provides a method for pseudo-implicit imposition of all boundary conditions through an iterative update. Boundary conditions may be imposed on any full or partial grid plane within a zone.

At free boundaries, i.e., inflow or outflow, characteristic information is imposed consistent with the Roe discretization scheme. Inflow conditions may be imposed uniformly or may vary across the boundary. For downstream, internal flow boundaries, the static pressure, mass-averaged Mach number, or mass flow rate may be imposed.

At solid boundaries, the standard slip or no-slip boundary conditions may be imposed. For no-slip boundaries, the user may specify adiabatic, constant temperature or a temperature variation.

Specialized boundary conditions may also be imposed to simulate the effects of a screen, an actuator disk, a hole with bleed or blowing, and a compressor face.

Physical Models – Turbulence models are solved uncoupled from the flow equations to allow maximum modularity. Several models are available. An algebraic model combines the Baldwin-Lomax[10] model near a solid surface and the P. D. Thomas[11] model for free shear layers. Two one-equation turbulence models are available: Baldwin-Barth[12] and Spalart-Allmaras.[13] Finally, the user may select one of two, two-equation turbulence models: a k-ε model or the SST model.[14,15] The SST model is a hybrid model which blends the solution of the k-ω model near a solid surface and the k-ε model elsewhere.

In addition to perfect gas simulations, WIND can predict real-gas effects using either finite-rate chemistry or a frozen chemistry approximation. Several standard chemistry databases are provided, or the user may supply a database of species properties and reaction rates. The species mass fraction transport equations are solved coupled with the flow equations.

Solution Process

The solution process using any conventional time-marching Navier-Stokes code is basically the same, and may be divided into the following steps:

- Create a grid file
- Set boundary conditions
- Set initial conditions
- Set program control parameters
- Run the code
- Monitor convergence
- Examine the results

The mechanics of doing each of these steps may vary from code to code, however. The following sections briefly describe how these steps are typically accomplished when running the WIND code. Additional details may be found in the WIND User' s Guide.

Create a Grid File

WIND uses externally generated, structured computational grids. The grids for all the zones must therefore be created before running the WIND code, using any convenient grid generation code. WIND expects the grids to be stored in a Common Grid (.cgd) file which is in Common File format described earlier. Since most grid generation codes do not produce .cgd files directly, a separate utility called cfcnvt is included with WIND that may be used to convert a variety of file formats, including PLOT3D files, to Common Files. A typical procedure is thus to first store the grid file as a PLOT3D xyz file, which is an available option in most general-purpose grid generation codes, and convert it to a .cgd file using cfcnvt.

Another utility included with WIND, called GMAN, may be used to examine the .cgd file, assessing grid quality and listing information about the points and zones in the grid. GMAN may also be used to generate the flow-field (i.e., interior) grid itself, given the grids on the zonal boundaries. The GMAN User' sGuide contains detailed descriptions of these and others capabilities.

Set Boundary Conditions

With most CFD codes, boundary conditions are completely specified in an input data file. With WIND, however, setting boundary conditions may be thought of as a two-step process. The first step in setting boundary conditions is to label each boundary of each zone with the type of boundary condition to use, such as "viscous wall," "confined outflow," or "coupled." This is done using GMAN, and the information is stored in the Common Grid file. Boundary condition types may be specified for all or part of a boundary, allowing multiple boundary condition types on a single boundary.

Zonal interface boundaries do not have to be explicitly labeled by the user. GMAN can automatically examine the grid to find them and determine the zones involved, compute the geometric interpolation factors, and store the information in the .cgd file. GMAN is also used to cut holes and generate interpolation coefficients for overlapping (chimera) boundaries. The process is currently not completely automated for chimera boundaries.

The second step in setting boundary conditions is to define any values needed for a particular boundary condition, such as an exit pressure or a bleed rate. This information is specified in the Input Data (.dat) file.

The Input Data file is a user-created ASCII file containing information about the flow problem and how the WIND code is to be run. With many CFD codes, including NPARC, this information is specified using FORTRAN namelist and/or formatted input. With WIND, the input is specified using descriptive keywords. In addition to boundary condition information, the .dat file specifies the procedure for defining initial conditions and sets various control parameters, as discussed in the following two sections.

Set Initial Conditions

The usual procedure with WIND is to start a new problem by initializing the flow conditions at each grid point to the values specified by the user in the Input Data file via the FREESTREAM keyword. Other options allow different values to be used in different zones, a boundary layer to be added along a specified surface in a zone, and reinitialization of the flow in specified zones after a restart.

Set Program Control Parameters

Several keywords may be specified in the Input Data file to control the physical and numerical models to be used when running the code. Some of the options available are:

- Dimensionality (3D, 2D, axisymmetric, quasi-3D)
- Flow equations (Euler, Navier-Stokes, thin-layer Navier-Stokes, parabolized Navier-Stokes)
- Turbulence model (algebraic, one-equation, two-equation)
- Gas model and chemistry (perfect gas, frozen chemistry, equilibrium air, finite-rate chemistry)
- Implicit operator (explicit, scalar implicit, block implicit, explicit or implicit boundary conditions)
- Explicit operator (central, Coakley upwind, Roe upwind; Physical; 1st to 5th order)
- Damping schemes (2nd/4th order, boundary damping, TVD)
- Time-stepping (iterations and cycles, CFL#, Runge-Kutta)
- Convergence acceleration (grid sequencing, local CFL#, ramped CFL#)
- Integrated convergence parameters (forces, moments, mass flow)

Run the Code

WIND is invoked using a Unix script which links appropriate files and either starts WIND interactively or sets up a batch job. WIND can also be run in parallel mode, simultaneously using multiple systems connected via a network as though they were a single computer. These systems are typically workstation-class machines and need not be all from the same vendor.

In parallel mode WIND uses a master-worker approach. The user specifies the names of the participating worker systems via a multi-processing control (.mpc) file. (Note that the master may also be a worker.) The user must, of course, have accounts on the master and worker systems, and remsh/rsh access to the workers must be allowed via an /etc/hosts.equiv or .rhosts file. The PVM software needed for parallel operation, and the WIND code itself, is copied from the master to temporary directories on the workers. Thus, the worker systems need not have any of the required software installed.

There are a couple of very convenient features built into the script used to run WIND. The first allows a WIND run to be stopped at (or more exactly, shortly after) a pre-determined time through the use of an NDSTOP file. This is useful when an overnight run must be stopped before morning, when the workstations being used will be needed for interactive work. The second allows the user to break a long run into "sub-runs" by writing a script called wind_post containing tasks to perform between each run. This is useful, for example, when the complete solution is to be saved at various time intervals in an unsteady problem. This can also now be done via the spawn command, thus avoiding losing one's place in a queue. Details on the use of these features are in the WIND User' s Guide.

Monitor Convergence

Monitoring and properly assessing convergence levels during a WIND run are critical to obtaining meaningful, useful results. WIND users may track convergence by following residuals and/ or integrated forces, moments, and mass flow. For engineering applications, the recommended convergence monitoring method is the tracking of integrated quantities of interest. For example, if a wing/ body geometry is being modeled to determine drag, the integrated drag should be monitored, and some reasonable bounds on drag oscillations should be used as the convergence criterion.

The solution residuals are included in the List Output (.lis) file. For each iteration and zone, WIND prints the zone number, cycle number, location of the maximum residual, equation number for which the maximum residual occurred, the value of the maximum residual, and the L2-norm of all the residuals for all the equations over all the points in the zone. The integrated parameters that are chosen in the Input Data file will also be listed in the .lis file. The integration may be done over a number of specified three-dimensional regions and/or two-dimensional areas of a computational surface.

A time history tracking capability is also built into WIND, in which computed values at specified grid points in specified zones may be periodically written to a separate Time History (.cth) file. Currently, the only values that may be tracked with this option are Mach number, static pressure, static temperature, and the three Cartesian velocity components.

Utilities are included with the WIND code that allow plotting of the residuals and/or integrated quantities in the .lis file, and the values stored in the .cth file.

Examine the Results

All flow-field results computed by WIND, including the mean flow variables, turbulence model variables, and chemistry variables, are written into a Common File called a Common Flow (.cfl) file. The CFPOST utility included with the WIND distribution is a post-processing tool for examining the contents of the .cfl file. With CFPOST, a wide variety of variables and integrated values may be computed. Listings may be sent to the screen or to a file, and PLOT3D files may be created for other plotting packages and post-processors. CFPOST can also be used to create x-y, contour, and vector plots directly, with PostScript output. Commands are available to precisely specify the information of interest, the domain of interest, and the units in which the results are to be presented. Detailed information may be found in the CFPOST User' s Guide.

Summary

The steps in the generalized solution process listed earlier may be restated specifically for the WIND code as follows:

- Create a grid file using any convenient grid generation software, saving the file in PLOT3D xyz format.
- Convert the PLOT3D xyz file to a Common Grid (.cgd) file using cfcnvt.
- Store the boundary condition types and zonal connectivity data in the .cgd file using GMAN.
- Prepare the Input Data (.dat) file, defining boundary condition values, initial conditions, program control parameters, and integrated parameters for monitoring convergence.
- For parallel execution, prepare the multi-processing control (.mpc) file.
- Run the WIND code using the run script supplied with the code.
- Monitor convergence by examining the residuals and integrated values in the List Output (.lis) file, and the values in the Time History (.cth) file if applicable.
- Examine the computed results in the Common Flow (.cfl) file using CFPOST, creating PLOT3D files for other post-processing packages if desired.

Example Cases

The three codes on which WIND is based have been in use for up to 10 years and have been applied to a wide variety of complex configurations and flow-fields. Towne and Jones[16] recently published validation examples using NPARC, and numerous references can be found on the NPARC Technical Report Server at:

http://info.arnold.af.mil/nparc/NPARC_TRS/NPARC_TRS.html.

Recent application of the NXAIR code to unsteady and moving body simulations by Nichols and Tramel[6] indicate that good agreement with data can be obtained for time-accurate store separation with reasonable turnaround time. NASTD results published by Mani, et al.[15] validate and compare the turbulence models in NASTD for a wide range of flow-fields. In addition, Barber, et al.[17] have recently compared the computational results of five codes, including NPARC and NASTD, for the prediction of three-dimensional, supersonic mixing layers.

An extensive validation effort is currently underway for the WIND code. Initial comparison of the WIND code with NPARC and NXAIR are presented here for transonic turbulent flow in a two-dimensional converging-diverging duct, using the WIND, NPARC, and NXAIR codes. The geometry is shown in Fig. 2. The throat height h_{thr} = 0.14435 ft. This is one of the example cases in the NPARC validation archive, accessible via the WWW at:

http://info.arnold.af.mil/nparc/Archive_information.html.

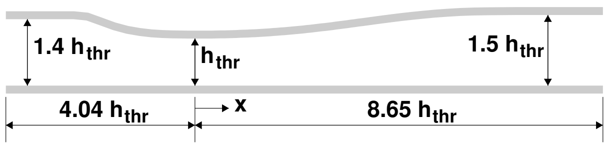


Fig. 2. Geometric configuration for converging-diverging duct.

Extensive experimental data are available for this geometry, at a variety of flow conditions.[18-22] For this example case, flow enters the duct at about M = 0.46, accelerates to just under M = 1.3 slightly downstream of the throat, shocks down to about M = 0.78, then decelerates and leaves the duct at about M = 0.51. The total pressure and temperature at the inflow are 19.58 psi and 525.6^o R, respectively, and the backpressure is 16.055 psi.

An 81 $\times$ 51 body-fitted computational mesh was generated algebraically, and is shown in Fig. 3, with every other grid line removed for clarity. The same mesh was used with all three codes, with $y^+ <$ 4 for the first point off of the wall.

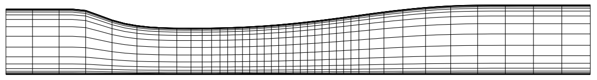


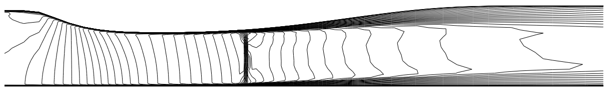
Fig. 3. Computational mesh for converging-diverging duct.

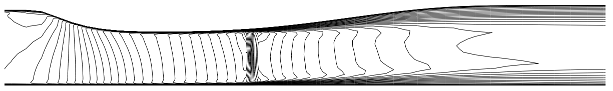
Adiabatic no-slip boundary conditions were used on both the upper and lower walls. The inlet boundary conditions corresponded to uniform flow at M = 0.46. A constant static pressure was specified at the exit boundary. For the first part of the calculation (2,000 iterations for the WIND and NPARC codes), the exit pressure was set low enough to establish supersonic flow throughout the diverging portion of the duct. The exit pressure was then raised to the value used in the experiment. For NXAIR, the flow was intialized to inflow conditions, then run with the specified backpressure for 150 time steps with 3 global Newton iterations per step and 5 iterations of the Gauss-Seidel matrix solver per Newton.

The NPARC calculation used the Baldwin-Lomax turbulence model,[10] and the WIND and NXAIR calculations used the Spalart-Allmaras turbulence model.[13] For the most part, each of the three codes was run using default values for the various input parameters. No attempt was made to use the same numerical schemes, smoothing methods, etc., for the three calculations. The results of these calculations should therefore not be used to draw definitive conclusions about the relative performance of the three codes.

The computed Mach number contours from the three codes are shown in Figs. 4a-c. The WIND and NXAIR results show a slightly more well-defined normal shock than the NPARC results, primarily due to the Roe-like algorithms used in both

WIND and NXAIR, as opposed to the central difference algorithm in NPARC. The shock position and boundary-layer growth, which is critical in determining shock postion, are nearly identical in all three simulations.


a. WIND


b. NPARC

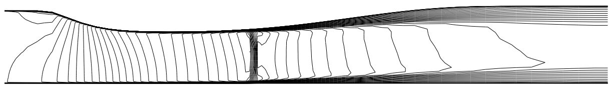

c. NXAIR

Fig. 4. Computed Mach contours for converging-diverging duct.

The computed static pressure distribution along the top and bottom walls is shown in Figs. 5a-b, along with the experimental data of Hsieh, Wardlaw, Bogar, and Coakley.[23] All three codes give essentially the same results upstream and downstream of the shock. The WIND and NXAIR results agree a little better with the experimental data in the diverging part of the duct.

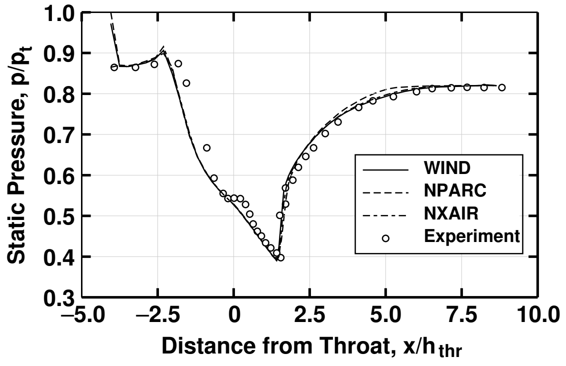
The computed u-velocity profiles at four locations are compared with the experimental data in Figs. 6a - d. The data were taken at x/h_{thr} = 1.73, 2.88, 4.61, and 6.34. The computed results are shown at i = 37, 52, 67, and 73, corresponding to x/h_{thr} = 1.73, 2.90, 4.58, and 6.20. These are the grid indices closest to the experimental data locations. Overall, all the results are similar, with the WIND and NXAIR results perhaps in slightly better agreement with the experimental data, at least in the core region.

Conclusion

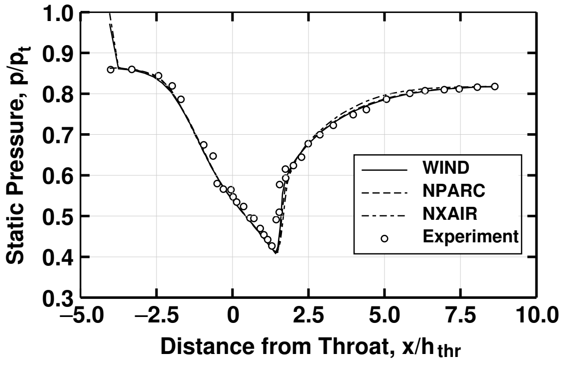
A joint project, with partners in government, industry, and academia, has been undertaken under the auspices of the NPARC Alliance to combine the capabilities of three existing application-oriented CFD solution systems: NPARC (NASA LeRC and AEDC), NASTD (Boeing), and NXAIR (AEDC). This software development project has presented many challenges in the area of program management, software management, and communication. The result is a suite of codes centered on the WIND flow solver. This system is user-friendly and flexible, with a GUI interface for boundary condition setting. The WIND code solves the compressible, Navier-Stokes equations with or without real-gas effects on structured, blocked grid zones with general zonal interfaces. An example demonstrating some of the current capabilities of this system was presented. WIND code results indicate good agreement with both NPARC and NXAIR results. Several capabilities will be added within the next year to achieve a complete merger of code capabilities, resulting in significant resource leveraging for future CFD development.

Acknowledgments

The authors would like to acknowledge the leadership and vision of management at AEDC,

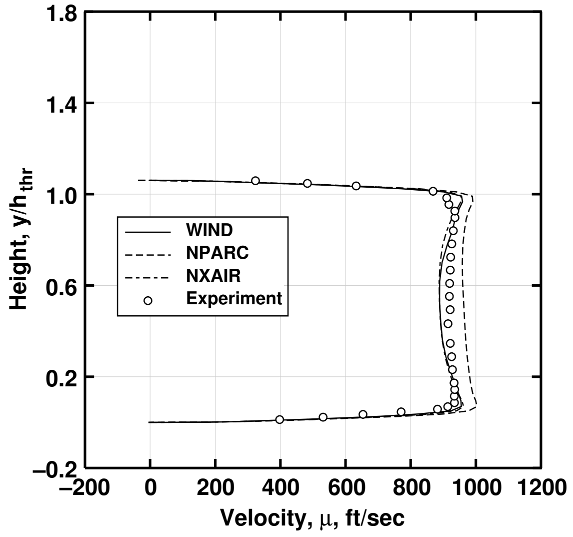


a. Top wall

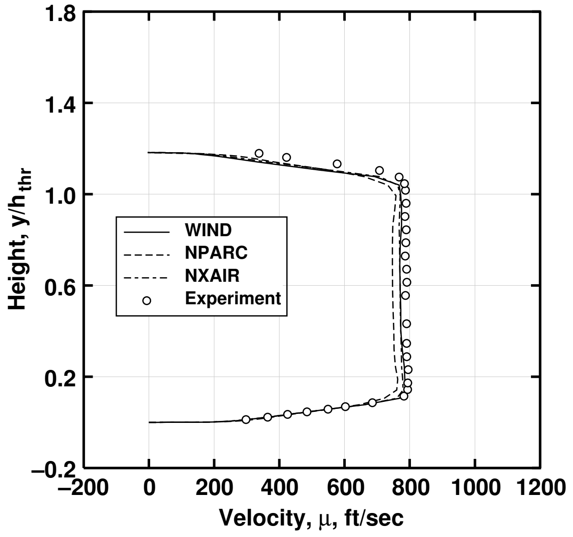


b. Bottom wall

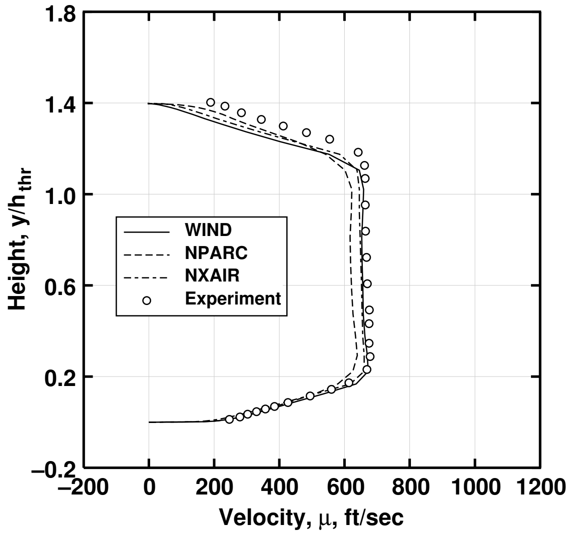
Fig. 5. Static pressure distribution for converging-diverging duct.



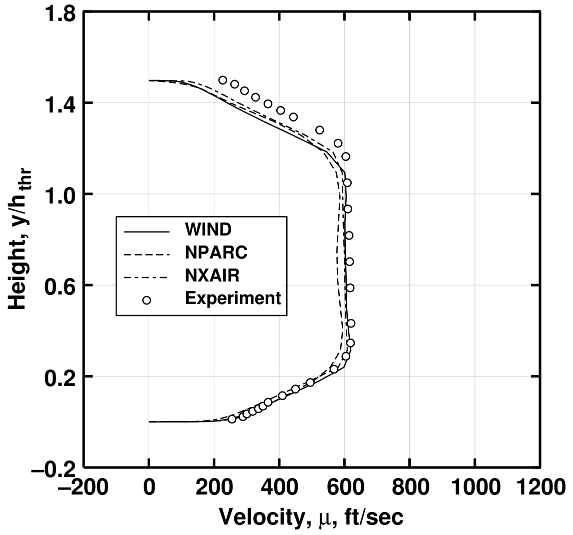
a. $x/h_{thr} = 1.73$



b. $x/h_{thr} = 2.88$



c. $x/h_{thr} = 4.61$



d. $x/h_{thr} = 6.34$

Fig. 6. u-velocity profiles for converging-diverging duct.

NASA LeRC, and the Boeing Company, especially Jere Matty, Dr. Joe Shaw, and Dr. Ray Cosner. We would also like to acknowledge the considerable technical contribution from the many individuals who have been involved in this project and the NPARC Alliance. Finally, we would like to thank the HPCMP CHSSI program for ongoing support of this effort relative to scalable parallel processing.

References

1. Cooper, G. K. and Sirbaugh, J. R., "The PARC Distinction: A Practical Flow Simulator," AIAA Paper 90-2002, 1990.

2. Matty, J. J and Shin, J. "The NPARC Alliance: A Progress Report." AIAA Paper 97-3353, 1997.

3. Power, G. D., Cooper, G. K., and Sirbaugh, J. R., "NPARC 2.2 – Features and Capabilities," AIAA Paper 95-2609, 1995.

4. Bush, R. H., "A Three Dimensional Zonal Navier-Stokes Code for Subsonic Through Hypersonic Propulsion Flowfields," AIAA Paper 88-2830, 1988.

5. Tramel, R. W. and Nichols, R. H., “ A Highly-Efficient Numerical Method for Overset-Mesh Moving-Body Problems,” AIAA Paper 97-2040, 1997.

6. Nichols, R. H. and Tramel, R. W., “ Application of a Highly Efficient Numerical Method for Overset-Mesh Moving Body Problems,” AIAA Paper 97-2255, 1997.

7. Coakley, T. J., “ Implicit Upwind Methods for the Compressible Navier-Stokes Equations,” NASA TM-84364, 1983.

8. Roe, P. L., “ Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes,” *Journal of Computational Physics*, Vol. 43, pp. 357-372.

9. Cain, A. B. and Bush, R. H., “ Numerical Wave Propagation Analysis for Stretched Grids,” AIAA Paper 94-0172, 1994.

10. Baldwin, B. S., and Lomax, H., "Thin Layer Approximation and Algebraic Model for Separated Turbulent Flows," AIAA Paper 78-257, 1978.

11. Thomas, P. D., “ Numerical Method for Predicting Flow Characteristics and Performance of Nonaxisymmetric Nozzles-Theory,” Langley Research Center, NASA CR 3147, 1979.

12. Baldwin, B. S. and Barth, T. J., “ A One-Equation Turbulence Transport Model for High Reynolds Number Wall-Bounded Flows,” NASA TM-102847, 1990.

13. Spalart, P. R., and Allmaras, S. R., "A One-Equation Turbulence Model for Aerodynamic Flows," AIAA Paper 92-0439, 1992.

14. Menter, F. R., “ Zonal Two Equation κ-w Turbulence Models for Aerodynamic Flows,” AIAA Paper 93-2906, 1993.

15. Mani, M., Ladd, J. A., Cain, A. B. and Bush, R. H. “ An Assessment of One-and Two-Equation Turbulence Models for Internal and External Flows.” AIAA Paper 97-2010, 1997.

16. Towne, C. E. and Jones, R. R., III., “ Results and Current Status of the NPARC Alliance Validation Effort,” AIAA Paper 96-0387, 1996.

17. Barber, T. J., Chiappetta, L. M., DeBonis, J. R., Georgiadis, N. J., and Yoder, D. A., "An Assessment of Parameters Influencing the Prediction of Shear Layer Mixing," AIAA Paper 97-2639, 1997.

18. Chen, C. P., Sajben, M., and Kroutil, J. C., "Shock Wave Oscillations in a Transonic Diffuser Flow," *AIAA Journal*, Vol. 17, No. 10, pp. 1076-1083.

19. Bogar, T. J., Sajben, M., and Kroutil, J. C., "Characteristic Frequencies of Transonic Diffuser Flow Oscillations," *AIAA Journal*, Vol. 21, No. 9, pp. 1232-1240.

20. Salmon, J. T., Bogar, T. J., and Sajben, M., "Laser Doppler Velocimeter Measurements in Unsteady, Separated Transonic Diffuser Flows," *AIAA Journal*, Vol. 21, No. 12, pp. 1690-1697.

21. Sajben, M., Bogar, T. J., and Kroutil, J. C., "Forced Oscillation Experiments in Supercritical Diffuser Flows," *AIAA Journal*, Vol. 22, No. 4, pp. 465-474.

22. Bogar, T. J., "Structure of Self-Excited Oscillations in Transonic Diffuser Flows," *AIAA Journal*, Vol. 24, No. 1, pp. 54-61.

23. Hsieh, T., Wardlaw A. B. Jr., Bogar, T. J., and Coakley, T. J., "Numerical Investigation of Unsteady Inlet Flowfields," *AIAA Journal*, Vol. 25, No. 1, pp. 75-81.