JOURNAL OF AEROSPACE COMPUTING, INFORMATION, AND COMMUNICATION
Vol. 1, May 2004

Accelerating CFD-Based Aeroelastic Analysis Using Distributed Processing

Anthony A. Boeckman* and Andrew S. Arena, Jr.†
Oklahoma State University, Stillwater, Oklahoma 74078

Distributed processing is applied to CFD-based aeroelastic analysis to accelerate the completion of flutter boundary predictions. The process for identifying an aeroelastic stability boundary is divided into several independent simulations that are distributed to multiple computers. The solutions are completed in parallel. This parallelism is used to accelerate the prediction of the instability boundary. This process is also applied to the generation of training data for the system identification of the unsteady aerodynamic response to motion and force on a modal structural model. The equipment used in this study is common, off-the-shelf computing hardware. This method is application ready with no required alterations to the CFD solver. Results address the extent to which this methodology is applicable to aerospace applications.

Nomenclature

$\mathbf{f}_a$	= generalized aerodynamic force vector
N	= number of data points in training set
Np	= number of data points in a parallel training set
na	= number of previous aerodynamic forces
nb	= number of previous structural displacements
np	= number of Central Processing Units
nr	= number of roots or mode shapes
q	= free stream dynamic pressure
$\mathbf{q}$	= generalized displacement vector
ρ	= free stream density

Introduction

PREDICTING instabilities in the aeroelastic behavior of aerospace structures is important in the design of modern aircraft that operate over a wide flight envelope. However, a complete aeroelastic analysis is often difficult to complete due to the complex structural, aerodynamic, and control interactions associated with even the simplest flight vehicles. To obtain the most accurate predictions for aircraft flight characteristics, contemporary research has turned toward the development of integrated computational models capable of capturing these interactions.

CFD models based on the Euler governing equations are powerful computational tools for aerodynamic analysis. Such models are often desirable for advanced aerospace applications since they make few assumptions about the characteristics of the flow field. Euler models produce reliable and accurate pressure distribution predictions, which drive the structural response. For aeroelastic analysis, one can take advantage of these benefits by coupling an unsteady Euler CFD algorithm to a time accurate structural dynamics solver. This coupled model predicts the

Received 8 August 2003; revision received 17 February 2004; accepted for publication 17 February 2004.

*Graduate Research Assistant, School of Mechanical and Aerospace Engineering, 218 Engineering North; Student Member AIAA.
†Associate Professor, School of Mechanical and Aerospace Engineering, 218 Engineering North; Senior Member AIAA.

complete aeroelastic response of an aerostructure. However, the computational time required for a CFD-based aeroelastic simulation has typically limited the use of such models in a flight test operational environment, even with recent advances in computational speeds.

This study seeks to further develop the relatively underdeveloped use of simple parallel processing techniques to speed the prediction of flutter boundaries. This is of particular interest in the flight test community, as they commonly have budget constraints on computational resources. Methods that use low cost computer clusters to reduce prediction times offer to enhance flight test productivity.

One possible method to accelerate the prediction is the use of system identification to model the aerodynamic response of the CFD solver to motion of the modal structural dynamics solver.[1] Once a model is trained, it will complete a simulation in seconds, compared to the weeks a CFD solution requires. The general form of the discrete Autoregressive Moving Average (ARMA) model is expressed as,

$$\mathbf{f}_a(k) = \sum_{i=1}^{na} [\mathbf{A}_i]\mathbf{f}_a(k-i) + \sum_{i=0}^{nb-1} [\mathbf{B}_i]\mathbf{q}(k-i) \tag{1}$$

However, the system model must be trained, and generating the training set can require just as much time as multiple CFD simulations. The relationship between the number of mode shapes and the number of data points needed to completely determine the system, N, is expressed as,

$$N = na \cdot nr + nb \cdot nr^2 \tag{2}$$

Since the number of data points needed to determine a system model is a function of the number of roots, nr, from the free vibration analysis used in the structural dynamics solver, the time needed to generate the training set increases with the complexity of the structural model. The trend is shown in Fig. 1 for the Generic Hypersonic Vehicle, GHV.[2] In this figure, a model of order 4-10, na-nb, is assumed to for all nr. The system identification training data must also be a response to a known input. The input signal used in this study is a chirp based frequency sweep. The sweep range is selected to include the natural frequencies of the structure.[1,3]

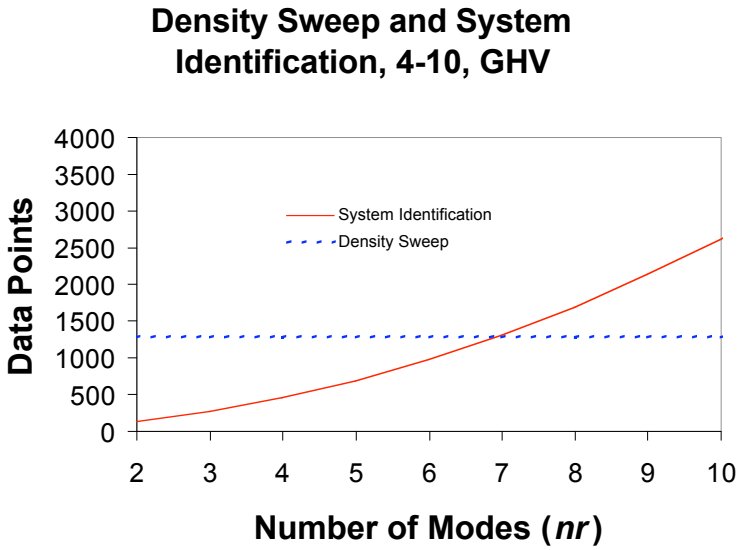

Fig. 1 Number of data points, or time steps, required for a dynamic pressure sweep and a system identification search.

Recent work has focused in reducing the time to complete a single discrete CFD time step. Several researchers have done extensive work on the acceleration of CFD speeds by applying parallel processing.[4,5] These methods used domain decomposition to divide the flow field into smaller zones, which separate processors then simulate. This method requires that the processors communicate the boundary values to other processors. The speed of that

communication affects the efficiency of the solver. The method also requires that the flow solver have the capability to handle the decomposition.

Another method to accelerate the completion of an aeroelastic instability boundary is distributed parallel processing, in particular the technique of distributed batch processing.[6] In this method, the problem of predicting an aeroelastic flutter boundary is reduced into several independent simulations, which are then sent to separate computers. A central computer controls which computers handle individual simulations. The individual computers, often called nodes, perform the CFD simulations then report the results to the central node. This technique can be applied to the system identification method as well. The generation of the training data set can be divided into a group of independent responses to input signals.

A methodology for the implementation of distributed batch processing is presented here, with results from two typical aeroelastic test cases.

Computational Tools

Computational analysis for this study is performed using the aeroelastic capabilities of the Euler3d (Ref. 7) code, part of the STARS[8] suite of programs. STARS is a highly integrated, finite element based software suite for multidisciplinary analysis of flight vehicles, including static and dynamic structural analysis, computational fluid dynamics, heat transfer, and aeroservoelastic capabilities. STARS is developed at NASA's Dryden Flight Research Center.

Structural analysis in STARS is accomplished using the finite element method to compute eigenvectors and eigenvalues, which describe the elastic modes for a structure. Arbitrary motions of the structure can then be represented by multiplying each eigenvector by a generalized displacement and applying modal superposition. Euler3d uses a time-marched, finite element approach to solve the unsteady Euler equations. The CFD solution is performed on a mesh consisting of unstructured tetrahedra using the transpiration method[9] to simulate structural deformations.

A complete aeroelastic analysis is accomplished by coupling the structural dynamics solver, using the modal vectors, with the unsteady CFD solver, which computes the generalized aerodynamic forces acting on the structure. The coupled solution is then a time marched methodology for solving

$$[\mathbf{M}]\ddot{\mathbf{q}} + [\mathbf{C}]\dot{\mathbf{q}} + [\mathbf{K}]\mathbf{q} = \mathbf{f}_a(t) \tag{3}$$

the matrix equation of motion for an arbitrary structure using generalized coordinates.

For this study, a cluster of eight personal computers was used for the computation of the aeroelastic responses. The individual nodes of the cluster were all Intel Pentium 4 2.53 GHz processors on Intel 845ge chipset motherboards with one gigabyte of RAM. Due to the nature of distributed batch processing, the master node does not require any special equipment. The computers are connected through a fast ethernet switch.

Distributed Processing Methodology

The standard method of searching for a flutter boundary with a CFD-based model of an aeroelastic system is the determination of the damping of several responses at various dynamic pressures. Using a curve fit, the dynamic pressure of the flutter boundary is estimated. This method is well suited to distributed batch processing. The geometry and structural dynamics are sent to each computing node. Each node is then assigned a dynamic pressure to simulate. Once all the simulations are complete, the aeroelastic investigator performs the analysis of the damping. Using batch processing, an entire flight envelope can be swept in the same time required to simulate a single response, assuming that enough computer nodes are available. In general, the speed advantage of this method is difficult to predict, as it is not known how many responses must be simulated before encountering the flutter boundary. However, assuming that all the computers in a cluster, *np*, are used for each sweep, then the speed increase is a factor of *np*.

Distributed batch processing can be applied to CFD system identification by dividing up the generation of training data. The training data can be separated by dividing the individual mode shapes into *nr* independent signals, where *nr* is the number of eigenvectors or mode shapes. The number of data points required for each mode shape is N_p, expressed as,

$$N_p = na + nb \cdot nr \tag{4}$$

This is the number of points needed to determine the parameters for the effect of previous forces and motions on the current force on the mode.

The time required to complete a training set for the entire structural system can be determined by multiplying the number of data points to be calculated on a machine by the time the machine takes to complete a single data point. For a heterogeneous cluster, machines of varying computational speed, the time to finish a training set can be expressed as,

$$Time = \max(Z_i \cdot dt_i) \cdot Np \tag{5}$$

where i refers to the index of a node, dt is the time the machine requires to finish a data point, and Z is the number of mode shapes assigned to the machine. The maximum value of this set of times is used, so that a true estimate of the time until completion of the training set is found. Since the combination of Z and dt is the determining factor in the time to completion, this value should be minimized to increase the efficiency.

For most cases, the machines within a cluster will have nearly identical speeds. This allows the replacement of the dt_i term with a constant dt. This is expressed as,

$$Time = \max(Z_i \cdot Np) \cdot dt \tag{6}$$

In the common case of np equal to nr, the Z term goes to unity. The resulting relationship between time and nr is expressed as,

$$Time = Np \cdot dt = (na + nb \cdot nr) \cdot dt \tag{7}$$

Since the serial, one computer, training data generation requires $N \cdot dt$, the ratio of speed increase can be expressed as,

$$\frac{N \cdot dt}{Np \cdot dt} = \frac{na \cdot nr + nb \cdot nr^2}{na + nb \cdot nr} = nr \tag{8}$$

This indicates that for a sufficiently large cluster, the time to complete a training data set is reduced by a factor of nr. So, for the average full airplane simulation with about 15 mode shapes the time is reduced by 93%. Note that nr is still in Eq. (7). So artificially increasing nr will not help. The effect of nr on time has been changed to a linear function Eq. (7) from a quadratic Eq. (2).

Results

Distributed batch processing is applied to the two methods of locating a flutter boundary for several geometries over a range of flight regimes. One such geometry is the AGARD445.6. The AGARD445.6 is a standard aeroelastic test case, which was investigated experimentally at NASA's Langley Research Center.[10] The planform view of the geometry is shown in Fig. 2.

For this example, the AGARD445.6 is modeled with two structural modes. These modes represent first bending and first torsion. The CFD mesh has 69,630 nodes and 373,798 tetrahedral elements. The Mach number is 0.96.

Using two computing nodes of the cluster, the response to a forced motion of each mode is simulated. The forced motion includes a frequency sweep through a range surrounding the structure's natural frequencies. Using the method of state space representation[11] for the combined structural and aerodynamic systems, the eigenvalues of the system as a function of the dynamic pressure are plotted. Since the system is a discrete time model the eigenvalues with an absolute value greater than one represent instability in the system. The first dynamic pressure to yield an unstable eigenvalue is considered the flutter boundary.

A sweep of the dynamic pressures from zero past the flutter boundary is run. After this initial sweep, the increment of the density is refined to the necessary tolerance. The flutter pressure for the AGARD445.6 at Mach 0.96 is found to be 0.40 psi using the system identification technique. This agrees well with experimental[10] and other computational[11,12] results. An eigenspace plot of the system stability for various dynamic pressures near the flutter boundary is shown in Figure 3.

In general, the prediction of the system identification model is confirmed with a sweep of the density in the full CFD model. For the AGARD445.6 at Mach 0.96, the flutter prediction of the system identification model was used

as a starting point. Five dynamic pressures were chosen, and are listed in Table 1. The damping of the first mode is listed as well. A 4th order curve is fit to the five data points relating the damping and the dynamic pressure. This is shown in Figure 4.

Table 1 Dynamic pressures in the density sweep of the AGARD445.6 at Mach 0.96

Dynamic Pressure, q	CPU hours	Damping on Mode 1, ξ
0.32 psi	22.6	0.01276
0.36 psi	22.4	0.01077
0.40 psi	22.5	0.00251
0.44 psi	22.6	-0.00141
0.48 psi	23.1	-0.01274

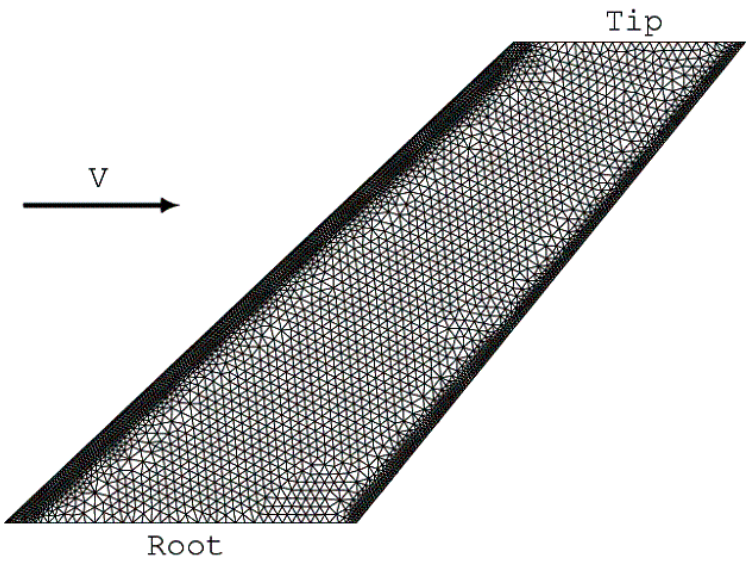

Fig. 2 AGARD445.6 test wing geometry and surface discretization.

The damping trend predicts a flutter point at 0.422 psi. This is an excellent match with experimental data.[10] The density sweep requires 23.1 CPU hours on five nodes of the cluster. In contrast, using one node of the cluster would require 112 CPU hours, assuming five times the fastest node of 22.4 CPU hours. The distribution of the free response onto multiple computers saved 88.9 CPU hours, a 79% savings in time.

The time to finish a density sweep and the time to finish the development of a system identification procedure is compared using the time to complete one time step on an average node of the cluster. Figure 5 shows the relationship between the time for a sweep of dynamic pressures and system identification as a function of the number of mode shapes used in the structural dynamics solver. Note that the pressure sweep free response technique is not an explicit function of the number of mode shapes, *nr*. The steps within the trends for system identification with a set number of nodes is due to the number of nodes required to complete more than one simulation before finishing. The density sweep is assumed to require only four free response time histories to make a flutter prediction.

Another interesting geometry is the Generic Hypersonic Vehicle, GHV.[2] The GHV is a test case developed by NASA to test the possible aeroelastic effects that would be observed on a hypersonic vehicle. Figure 6 shows the geometry and surface mesh for the GHV. The grid has 58,511 nodes and 321,755 tetrahedral elements. The GHV is modeled at Mach 2.2 for this example. Structurally the GHV is more complicated than the AGARD445.6. In this example the mode shapes associated with the nine lowest eigenvalues are used. As shown in Fig. 6, it is possible for the sweep of free responses at varying dynamic pressures to finish more quickly than the system identification technique.

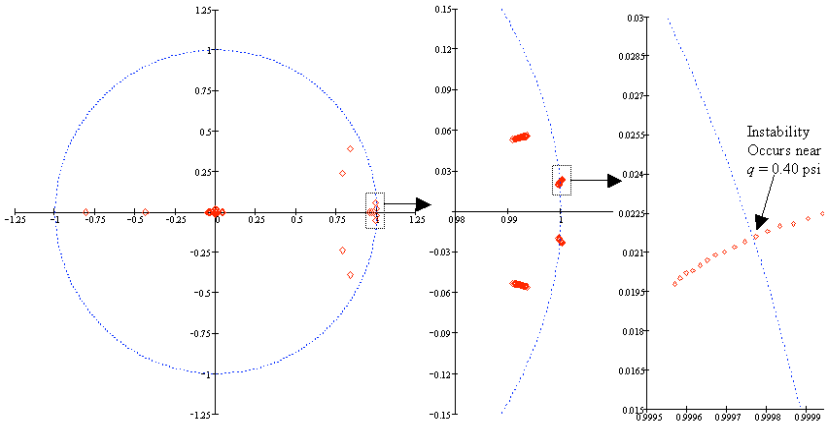

Fig. 3 Complete root locus for AGARD445.6 discrete-time aeroelastic system at Mach 0.96 including close-up of stability crossover point.

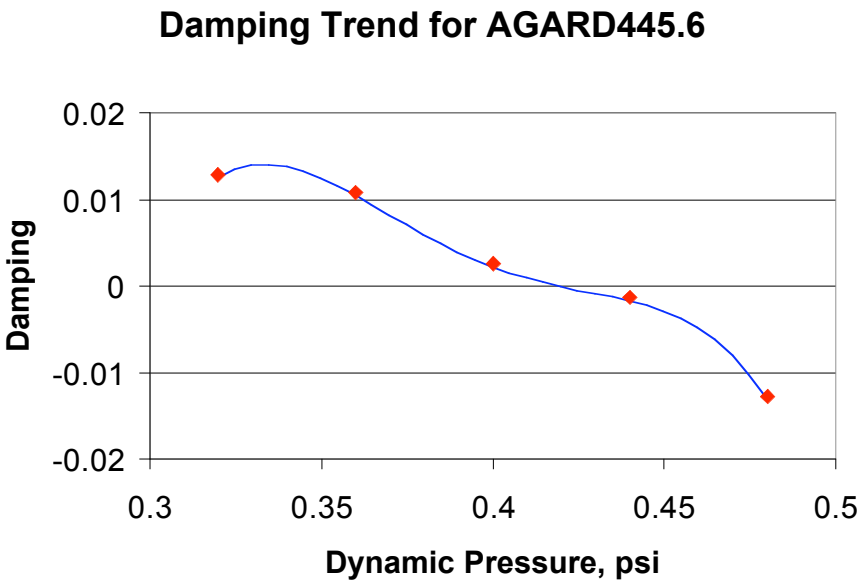

Fig. 4 Damping trend for the AGARD445.6.

A range of dynamic pressures is selected for a pressure sweep. The pressures are listed in Table 2. The free response analysis takes 2.31 CPU hours to complete on eight nodes of the cluster. As shown in the table the damping on Mode 2 is an indicator of the overall stability of the system. The dynamic pressure of 154.8 psi has a damping of very near zero.

In order to further refine the flutter boundary estimate, a sweep is run between 141.9 and 167.7 psi. The results are shown in Table 3.

The initial estimate of 154.8 psi is a valid flutter boundary prediction. The total time for the two dynamic pressure sweeps is 4.60 CPU hours using all eight nodes of the cluster. The total time using only the fastest node of the cluster is 35.56 CPU hours.

Table 2 Dynamic pressures in the density sweep of the GHV at Mach 2.2

Dynamic Pressure, q	CPU hours	Damping on Mode 2, ξ
103.2 psi	2.26	0.03392
116.1 psi	2.24	0.03746
129.0 psi	2.30	0.03696
141.9 psi	2.26	0.05131
154.8 psi	2.31	-0.6177E-04
167.7 psi	2.22	-0.04274
180.6 psi	2.28	-0.05290
193.5 psi	2.27	-0.07259

Table 3 Dynamic pressures in the second density sweep of the GHV at Mach 2.2

Dynamic Pressure, q	CPU hours	Damping on Mode 2, ξ
141.9 psi	2.26	0.05131
144.8 psi	2.24	0.04624
147.7 psi	2.23	0.05485
150.6 psi	2.29	0.06916
153.5 psi	2.25	0.001982
154.8 psi	2.31	-0.6177E-04
156.3 psi	2.29	0.00090
159.2 psi	2.23	-0.01632
162.1 psi	2.26	-0.03071
165.0 psi	2.28	-0.02903
167.7 psi	2.22	-0.04274

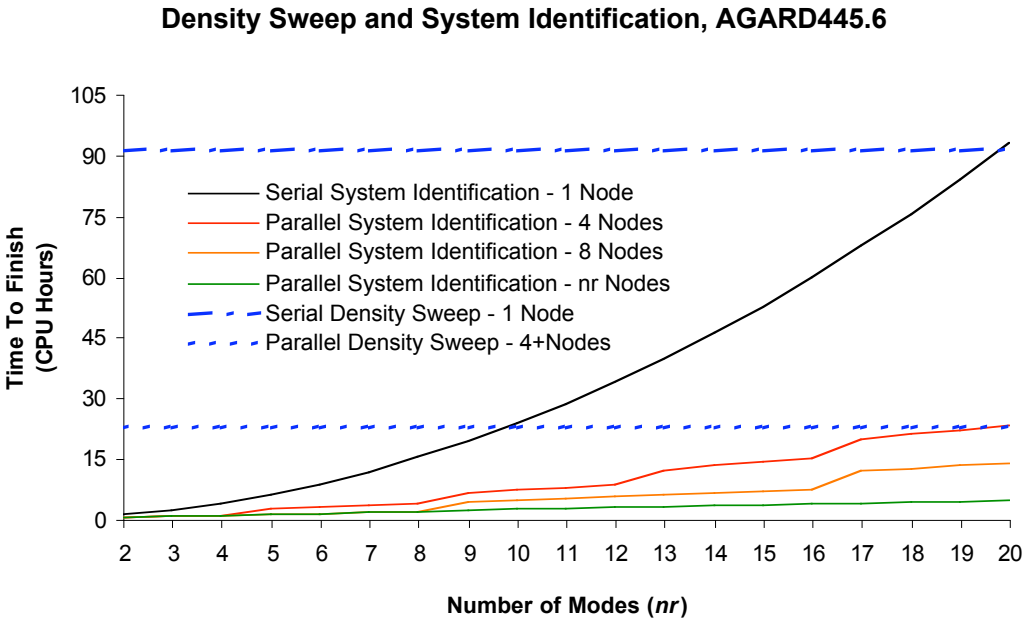

Fig. 5 The density sweep and system identification trends for the AGARD445.6 at Mach 0.96.

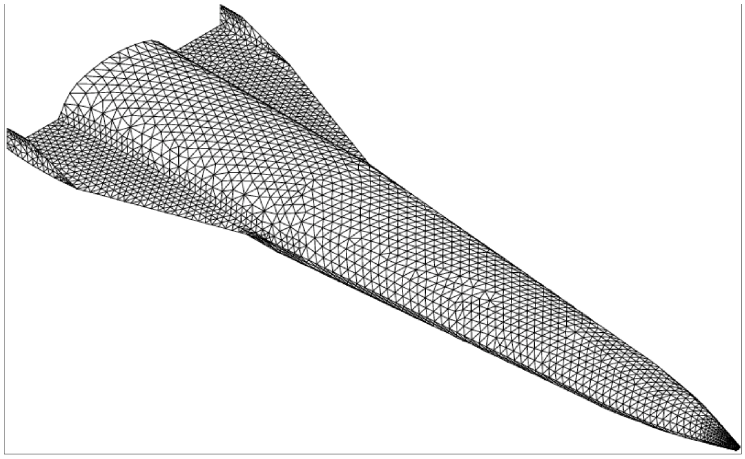

Fig. 6 The generic hypersonic vehicle (GHV) geometry and surface mesh.

The system identification technique is used to predict the flutter point of the GHV as well. The prediction is 133.5 psi. This is a significant difference from the density sweep prediction. However, the model does predict a boundary below the actual flutter pressure. Since the cluster has only eight nodes, one node of the cluster is required to simulate the response of two mode shapes. This results in a near doubling of the time needed to finish the system identification training set. The set takes 6.42 CPU hours to complete. The set could finish in 3.26 CPU hours with nine computer nodes.

The time for the density sweep and the system identification procedures to finish are shown in Figure 7. Notice that the system identification requires more time than the density sweep in most cases in the figure. This is a reversal of the transonic AGARD445.6. This is most likely due to the larger time steps that can be used in the supersonic regime were shocks are well established and less prone to motion compared to the transonic regime.

One interesting statistic of the distributed processing method is the efficiency of the examples listed here. The efficiency is defined by

$$\eta = \frac{Time_{FastestSerial}}{Time_{Distributed}} \tag{9}$$

where $Time_{FastestSerial}$ is the fastest time for a node to complete a simulation and $Time_{Distributed}$ is the time to finish using multiple computers. Efficiency is best described as how well the computing power of the cluster is utilized when processing a batch.

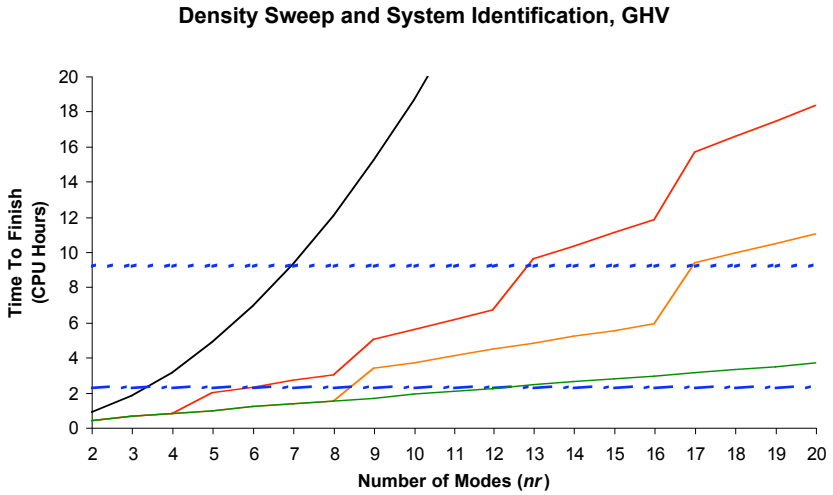

Fig. 7 The density sweep and system identification trends for the GHV at Mach 2.2.

For the AGARD445.6, the fastest time for the pressure sweep is 22.4 CPU hours (Ref. 13). The parallel computing takes 23.1 CPU hours. This yields 97% efficiency. Only three percent of the possible computations are wasted during the density sweep batch processing. The GHV shows a better efficiency with a 98.4% for the dynamic pressure sweep.

The system identification method for the GHV does show the effect of running nine simulations on eight computers. The fastest time for a simulation is 3.20 CPU hours. The distributed batch processing takes 6.42 hours. This leads to a poor efficiency of 49.8%. However, it should be noted that the computing power of the seven nodes not used to simulate the ninth mode shape training are available for use on other simulations because the computer resources are released once their assigned simulations are complete. If nine computers were used to run the batch processing, the time would require 3.33 CPU hours. This would yield an efficiency of 96.1%.

Conclusions

The distributed processing technique demonstrated here is shown to be an efficient method for the acceleration of aeroelastic flutter prediction for both the dynamic pressure sweep or system identification methodology. This method is applicable to any CFD flow solver. The method is shown to work in any flow regime including the transonic region.

The distributed processing methodology has one additional benefit. No changes to the CFD solver are required. Distributed processing libraries can be used to add automation features to CFD solver; however, the basic methodology does not require it. Furthermore, the efficiencies found in the presented examples are shown to compare well to other types of parallel processing.[12]

References

[1]Cowan, T. J., Arena, A. S., and Gupta, K. K., "Accelerating CFD-Based Aeroelastic Predictions Using System Identification," AIAA Paper 98-4152, Aug. 1998.

[2]Gupta, K. K., and Petersen, K. L., "Multidisciplinary Aeroelastic Analysis of a Generic Hypersonic Vehicle," NASA TM-4544, Oct. 1993.

[3]O'Neill, C. R., and Arena, A. S., "Comparison of Time Domain Training Signals for CFD Based Aerodynamic Identification," AIAA Paper 2004-0209, Jan. 2004.

[4]Geuzaine, P., Brown, G., Harris, C., and Farhat, C., "Aeroelastic Dynamic Analysis of a Full F-16 Configuration for Various Flight Conditions," *AIAA Journal*, Vol. 41, No. 3, 2003, pp. 363-371.

[5]Byun, C., and Guruswamy, G. P., "Aeroelastic Computations on Wing-Body-Control Configurations on Parallel Computers," *Journal of Aircraft*, Vol. 35, No. 2, 1998, pp. 288-294.

[6]Baker, L., and Smith, B. J., *Parallel Programming*, McGraw-Hill, New York, 1996.

[7]Cowan, T. J., "Finite Element CFD Analysis of Supermaneuvering and Spinning Structures," Dissertation, Oklahoma State Univ., Aug. 2003.

[8]Gupta, K. K., "STARS – An Integrated General-Purpose Finite Element Structural, Aeroelastic, and Aeroservoelastic Analysis Computer Program," NASA TM-4795, May 1997.

[9]Fisher, C. C., and Arena, A. S., "On The Transpiration Method For Efficient Aeroelastic Analysis Using An Euler Solver" AIAA Paper 96-3436, July 1996.

[10]Yates, E. C., Jr., "AGARD Standard Aeroelastic Configurations for Dynamic Response. Candidate Configuration I.–Wing445.6" NASA TM-100492, 1987.

[11]Cowan, T. J., Arena, A. S., and Gupta, K. K., "Development of a Discrete-Time Aerodynamic Model for CFD-Based Aeroelastic Analysis," AIAA Paper 99-0765, Aug. 1999.

[12]Goodwin, S. A., Weed, R. A., Sankar, L. N., and Raj, P., "Toward Cost Effective Aeroelastic Analysis on Advanced Parallel Computing Systems," *Journal of Aircraft*, Vol. 36, No. 4, 1999, pp. 710-715.

[13]Boeckman, A. A., "Accelerating Computational Fluid Dynamics Based Aeroelastic Analysis Using Distributed Processing," M.S. Thesis, Oklahoma State Univ., May 2003.