Copyright © 1998, by the International Council of the Aeronautical Sciences (ICAS)
and the American Institute of Aeronautics and Astronautics, Inc.

A98-31681

ICAS-98-6,4,5

COMPUTATIONAL ALGORITHMS FOR THE CONFIGURATION DESIGN

SC Gupta
Centre for Military Airworthiness and Certification
Bangalore, INDIA

Abstract

There has been increasing complexity in fighter aircraft design. The new emerging head-on air-to air missiles require the fighters to manoeuvre in six independent degrees of freedom. For the accurate analysis and design task, the computational alogrithms have advanced in the recent past. These algorithms also apply for computational electromagnetics for stealth featuring of configurations. Subject writeup brings out the growth in computational algorithms, optimal grid generation techniques and requirement of parallel computations for speedup.

Introduction

There has been increasing complexity in fighter aircraft design. Longitudinally unstable canard-delta configuration are considered to provide outstanding combinations of manoeuvrability, acceleration and short-field performance. Optimal wing profile is required throughout the flight envelope. The variable camber design through leading edge and trailing edge flaps deflections aims at such a commitment. This results in excellent sustained turn performance but causes bad transonic transients and high supersonic drag. The use of such flaps are therefore confined to subsonic flow regimes. High pitch manoeuvres require the use of vortex associated lift. The transonic flow drag reduction is possible through wing-body blending. Supersonic wave drag can be reduced through optimal wing warp. Unsteady flow conditions prevail during the execusion of manoeuvres. The entire task of analysis and design requires the accurate algorithms for prediction of aerodynamic values. Subject writeup makes an approach towards recent growth in computational algorithms for such purposes.

The loading on aircraft needs to be accurately predicted for varied Mach numbers and altitude conditions at various combinations of aircraft attitudes, control surfaces travel rates and accelerations, aircraft angular velocities and accelerations, and also the linear velocities and accelerations. This involves in total flow regime of subsonic, transonic and supersonic including mixed flow conditions with boundary layer separation and presence of free vortex flows. In addition the leading edge and trailing edge flap deflections and travel rates need to be taken into account. The cost of several wind tunnel blow downs would be exorbitant and would have be kept minimum. Moreover the estimation of optimal warp is essentially a theoretical exercise.

About four decades back, flow field analysis was done using theoretical aerodynamics, which essentially involves in an integral formulation. Theoretical aerodynamics provides powerful tools for optimization of wing camber. The methodology is based on surface integrals and is largely useful for linearised potential flow models. Recent research applies theoretical aerodynamic computations to nonlinear flows using field integrals[1-3]. About three decades back, the solution to quasi-linear equations for transonic flow analysis using finite difference techniques were formed. Theory resulted in transonic flow analysis including cases with imbedded weak shock waves. Separate difference formulas are used for elliptic and hyperbolic region which account for local domain of dependance of difference equation. Conservation of mass is possible by moving switching functions inside the difference operators. Finite difference methodology with

21st ICAS Congress
13-18 September 1998
Melbourne, Australia

Copyright © 1998 by ICAS and AIAA. All rights reserved

Copyright © 1998, by the International Council of the Aeronautical Sciences (ICAS)
and the American Institute of Aeronautics and Astronautics, Inc.

line relaxation process is well known. solution of unsteady transonic small disturbance equation with time-accurate approximate factorisation (AF) algorithms subsequently came-up. Stability characteristics of these algorithms remained of prime importance especially for the osciallatory flows.

Advances in the finite volume method for transonic potential-flow calculations and numerical solutions of compressible Navier-Stokes equations followed. Improvements in artifical viscosity to allow retention of second-order accuracy in supersonic flows in important. Advantage of finite volume methodology lies in its decoupling of solution process from grid-generation step. Three-dimensional, time-dependent, compressible Navier-Stokes equations offer a viable tool for handling any flow conditions in the Eulerian formulation. The conservative equations in integral form for mass, momentum, and energy with respect to a control volume (V) can be written down. Accurate numerical modelling of shock waves and other discontinuities is possible by giving the algorithm a bias towards the direction of propogation. Conservative differencing schemes are possible within the directional bias.

Numerical iterative optimization remains under the constant criticism[4]. Though optimal body-wing warp can be generated with these techniques for any flow conditions, the type of warps resulted is seen to be not very encouraging. Utilising the principles of calculus of variations and an objective function, converging good solutions are obtained but the technique is confined to linearised flow conditions[5]. The genesis of computational algorithms growth is summed up as below[1-9].

Years	Algorithms
1960s	- Theoretical aerodynamics for flow field analysis involving linearised potential flow equations.
1970s	- (i) Computational algorithms using numerical analysis mainly finite difference to quasi-linear form of equations.
	(ii) Theoretical aerodynamics for optimisation using calculus of variation.
1980s	- (i) Computational algorithms largely for solution to nonlinear equations.
	(ii) Theoretical aerodynamics with some numerical support e.g. field integral methods.
	(iii) Attempts to headway in numerical algorithms towards optimization.
1990s	- Computational algorithms refinements and application to allied fields e.g. aeroelasticity, controls and stealth features.

Mesh grading and grid-point redistribution is required using optimization techniques for the accurate analysis and design task. Some representative form of objective function can be considered and associated scaling parametring and nonlinear programming could be done. However, multiple objective environment is subjective to the objectivity of parameters (weightages) which could be varied suiting the solutions convenience[6]. The problem of grid generation lies in discretizing the physical domain of interest to an appropriate mesh suitable for accurate interpolation or approximation. Computational efficiency, accuracy and stability are enhanced if the mesh is graded in regions where the solutions are fast changing. Smoothness and orthogonality functions are imparative in the objective function.

Speedup of computational algorithms is required since the process building is extremely slow [7-8]. Typical computer architectures are

Copyright © 1998, by the International Council of the Aeronautical Sciences (ICAS) and the American Institute of Aeronautics and Astronautics, Inc.

required to Parallel up the operations to reduce the run time[8-9]. Moreover the development of aircraft requires integration of multi-disciplinary technologies such as : 1) fluid dynamics for flow management, (2) aeroelastic effects of structures with favourable tailoring, (3) thrust through complex propulsive system including thrust nozzle vectoring, and (4) controls for stability including special requirements of stabilisation of pitch unstable airframes. Computational fluid dynamics (CFD) based codes are both vectorized and parallelized for efficient execusion on a vector machine. Computers with gigaflop execusion speed are expected to perform well to handle complete aircraft configuration for solutions to Navier-Stokes equations. Development of multi-displinary computations are on the horizon which involve, (1) CFD and aeroelastic coupling including eigenstructure assignment, (2) CFD, aeroelastic and control coupling, and (3) development of a computational electromagnetic (CEM) technology based on CFD methods. Considering low observability and aerodynamic disciplines, the geometric shape optimization with CFD/CEM constraints make the problem size larger. Computers with teraflop capability with artificial intelligence would be required for such tasks. Numerical process based on knowledge, judgement, reasoning and perception could provide artificial intelligence (AI) base.

Theoretical Algorithms

Much of the flow field analysis in theoretical aerodynamics is possible through Laplace Eqn. (1) below. Equation truly applies to linearised incompressible flow conditions. Solution to Laplace Eqn. can be written as Eqn.(2) below.

$$L\,\phi = 0 \quad \ldots \qquad (1)$$

$$\phi = \iint_S \Gamma(s)\, dx\, dy \quad \ldots \qquad (2)$$

where L is Laplace operator, ϕ is the velocity potential, Γ is the circulation in a s-domain and considered piece-wise continuous on the dx × dy area. Γ(dx, dy) of higher order is also possible.

The compressibility effects can be progressed through $x \;/\; \sqrt{\left|1 - M^2\right|}$ term, where M is the Mach number. Thus, the elliptic and hyperbolic form of equations can be separately written for subsonic and supersonic flow regimes as below for the x, y & z Cartesian co-ordinate system.

$$(1-M^2)\,\phi_{xx} + \phi_{yy} + \phi_{zz} = 0,\; M < 1 \qquad (3)$$

$$(1-M^2)\,\phi_{xx} - \phi_{yy} - \phi_{zz} = 0,\; M > 1 \qquad (4)$$

Loss in accuracy in the supersonic linearised flow is associated with the use of linearised boundary conditions, Eqns.(5) & (6) below. The exact boundary condition is that of normal mass flux to boundary surface being nil. However the boundary condition used here is the normal velocity to surface being zero. The increasing value of Mach number in the Eqn.(6) makes u << O (U) lesser valid.

$$\hat{n}\,\rho q = 0,\; \bar{q} = (U+u)\,\bar{i} + v\,\bar{j} + w\,\bar{k} \qquad (5)$$
$$u \ll O(U)$$

$$\hat{n}\,q = 0,\; \rho\bar{q} \approx (U + \beta^2 u)\,\bar{i} + v\,\bar{j} + w\,\bar{k} \qquad (6)$$
$$M^2 u \ll O(U)$$

where $\beta = \sqrt{\left|1 - M^2\right|}$, and $\hat{n}$ is the normal unit vector to surface, ρ is density, q is the resultant velocity. U is the freestream velocity; u,v & w are perturbation velocities.

The outward unit normal at any point of boundary surface can be denoted by Eqn.(7). Much of the accuracy comes from establishing the normal unit vector using direction cosines[3].

Copyright © 1998, by the International Council of the Aeronautical Sciences (ICAS)
and the American Institute of Aeronautics and Astronautics, Inc.

$$\bar{n} = \hat{n}_x \bar{i} + \hat{n}_y \bar{j} + \hat{n}_z \bar{k} \quad (7)$$

The supersonic use of these methods lies mainly in the optimization effort in design using calculus of variation. Table-1 shows drag reduction of a delta wing of aspect ratio (AR) of 2.25. A well proven code of Ref. 5 is utilised. The wing is slightly clipped at the tips to prevent infinite solutions towards tips. Two conditions in Table-1 for optimization are for constraints of $(\bar{L}\ and\ \bar{L}, \overline{M_x})$, where $\bar{L}$ is the lift constraint and $\overline{M_x}$ is the pitching moment constraint. Figures 1 shows this wing optimally warped for M=1.25 for angle of attack (α) of 2° at off-design operating conditions of various Mach numbers and angles of attack combinations, only the lift constraint is used.

Figure 2 shows affect of canard interactions with a wing at supersonic conditions. Code of Ref.11 is utilised. Canard is placed in line with wing to provide maximum interaction. The presence of wing makes the canard trailing edge pressure recoveries at supersonic Mach numbers.

Computational Algorithms

Computational fluid dynamics utilises the numerical techniques. The hierachy of fuid dynamics lies in Eqn. (8) below.

$$Q_t + E_x + F_y + G_z = \text{Source} \quad (8)$$

where Q is the solution vector and E_x, F_y and G_z are the fluxes in the x, y and z directions. Accuracy of these methods lie in the body-fitting co-ordinates and the stability of the scheme. Finite difference schemes for the solution of these equations to transonic small disturbance Eqn. (9) below was developed about three decades back. The mass conservative form of solution process in a finite difference technique is given by Eqn. (10) with the use of artifical viscosity parameter (μ).

$$\left[K\phi_x - (\gamma+1)\phi_x^2/2\right]_x + \phi_{yy} + \phi_{zz} = 0 \quad (9)$$

$$p_{i,j,k} + q_{i,j,k} + r_{i,j,k} + \left(\mu_i\, p_{i,j,k} - \mu_{i-1}\, p_{i-1,j,k}\right) = 0 \quad (10)$$

where p, q and r are the difference operators in three directions and μ is the switching function (artificial viscosity). The upwind bias in supersonic flow provides the discontinuity to capture waves. Rotated differences introduced by well known (Albone & Jameson) ensure stable calculations of a locally supersonic domain (Fig.3). Artificial viscosity ensures mass conservation when difference schemes are switched in solving potential flow equations. The use of transonic small perturbation does not allow conservation in momentum, which can be done by using the Euler's Eqn.(11) below.

$$\left.\begin{array}{ll}\text{Continuity} & \rho_t + (\rho V)_x = 0 \\ \text{Momentum} & (\rho V)_t + (\rho V^2 + \rho a^2)_x = 0\end{array}\right| \quad (11)$$

Using the vectors, the Eqn. (11) can be written as Eqn. (12) in the conservative form because the physically conservative law gets naturally applied.

$$u_t + F_x = 0 \quad (12)$$

Finite difference are regarded simply as a make shift for infinite signals, making the difference small that errors due to finite size would be diminishing. But reducing the finite difference size has implications. Figure (4) shows the pressure difference coefficient on a 2-d flat plate. A5 x5 mesh in a finite difference scheme results in the ΔC_p which is comparable with the stable values of boundary integral methodology. Larger mesh sizes result in drop in ΔC_p values [9]. Larger paneling in boundary integral does not result in changes in ΔC_p values.

Finite volume methods provide to calculate flow past arbitrary geometrical configurations with larger accuracy. Advantage of finite volume method lies in its decoupling of

Copyright © 1998, by the International Council of the Aeronautical Sciences (ICAS) and the American Institute of Aeronautics and Astronautics, Inc.

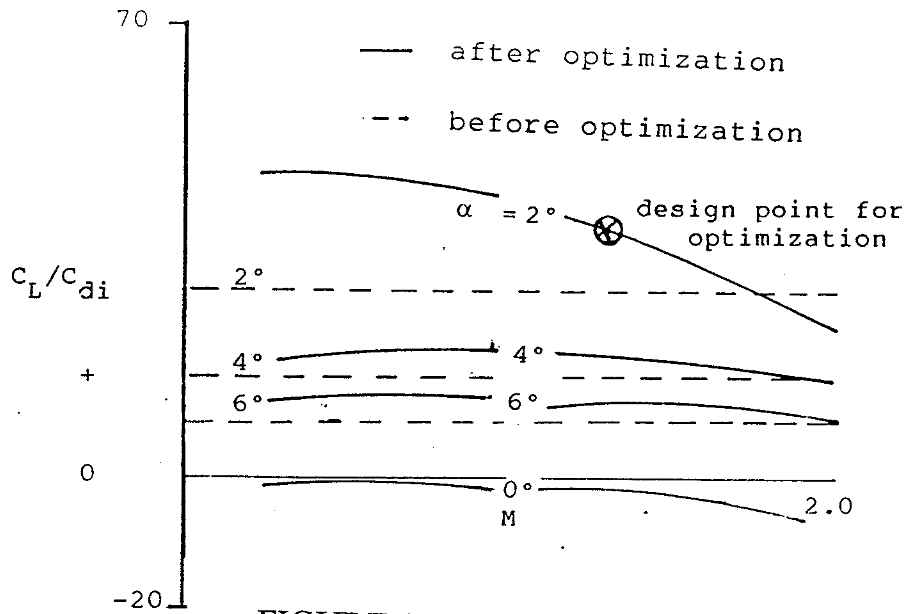

FIGURE 1 - Off-design operating Lift/Drag values for a Wing Optimised at a Supersonic Speed

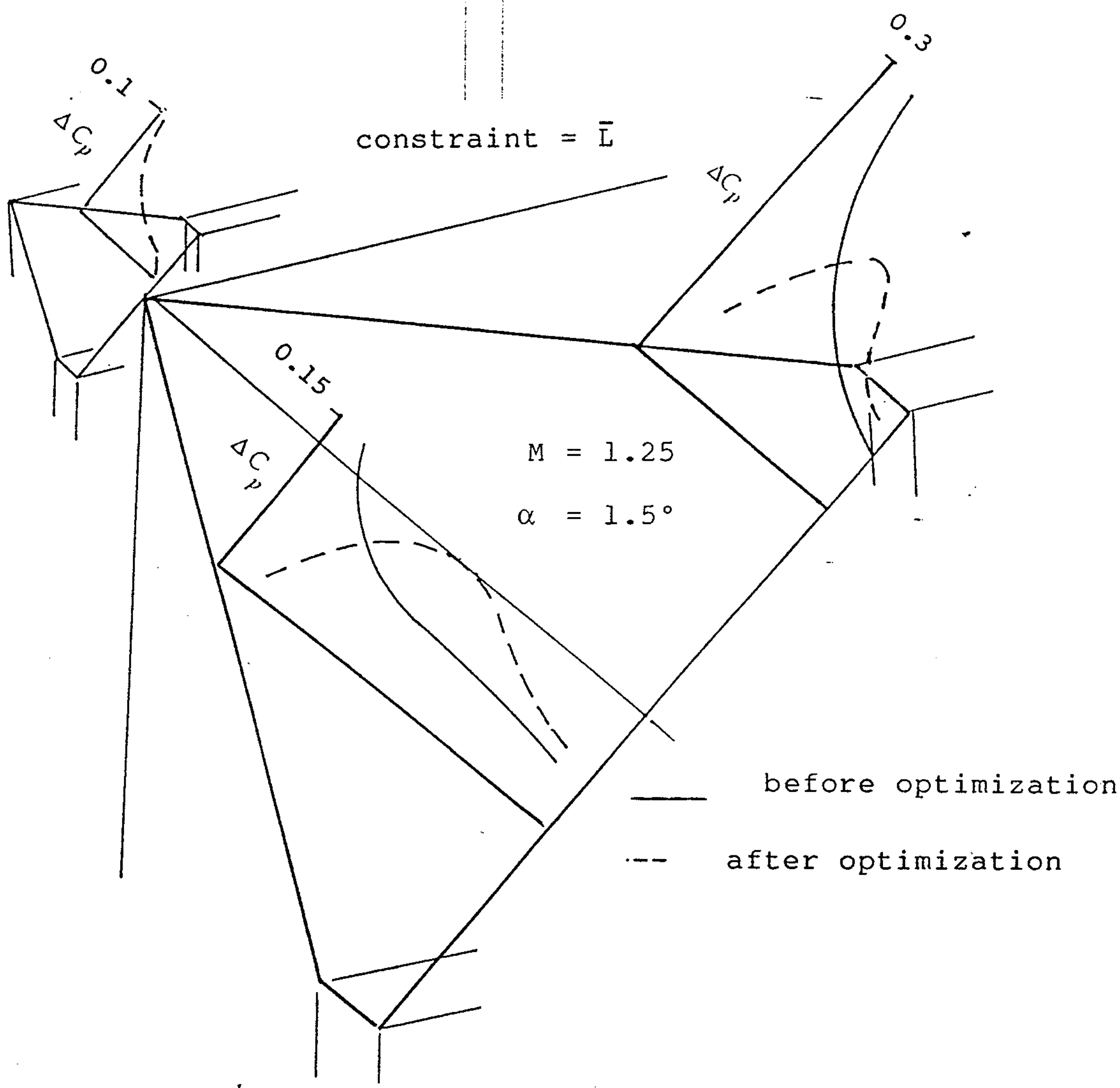

FIGURE 2 - Wing Optimization in the Presence of Canard at a Supersonic Speed

Copyright © 1998, by the International Council of the Aeronautical Sciences (ICAS)
and the American Institute of Aeronautics and Astronautics, Inc.

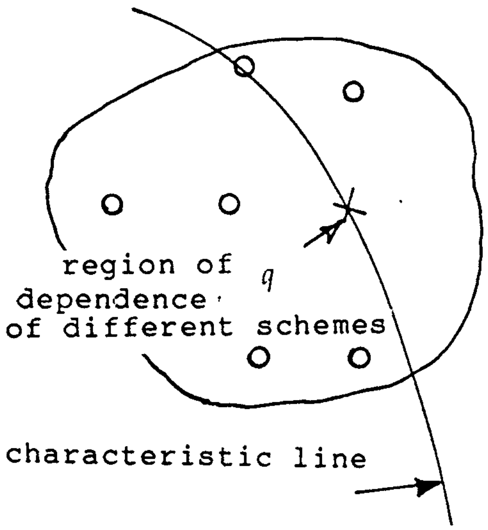

FIGURE 3 - Need for a Rotated Difference Scheme

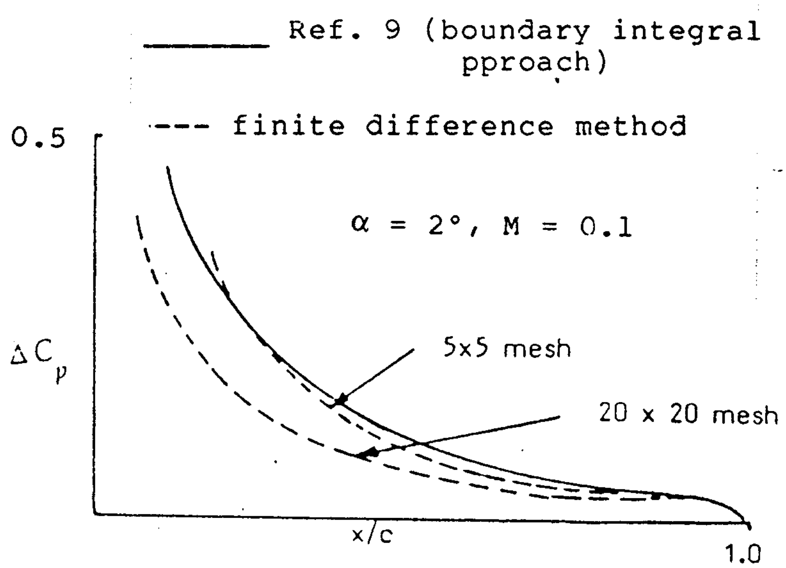

FIGURE 4 - Subsonic Pressure Difference Coefficient

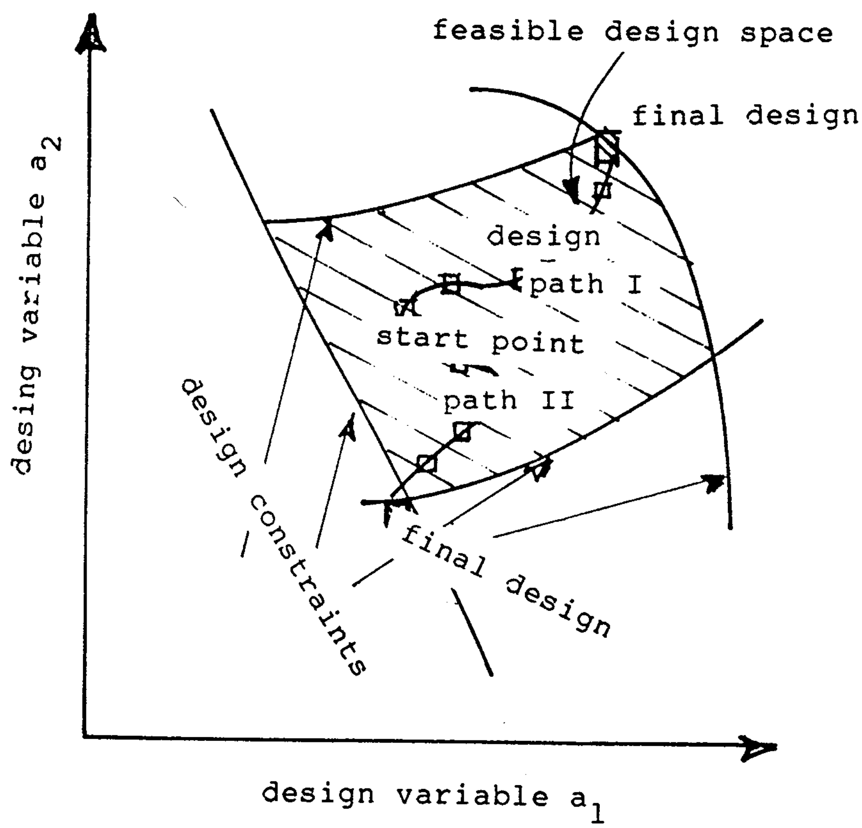

FIGURE 5 - Numerical Progress in Design

Copyright © 1998, by the International Council of the Aeronautical Sciences (ICAS)
and the American Institute of Aeronautics and Astronautics, Inc.

the solution procedure from the grid-generation step. This allows the grid to be genereated in any convenient manner, and allows application of an essentially universal algorithm to any problem for which a boundary-confirming coordinate system can be originated.

Numerical optimization aims at arriving at minimum drag configurations[4] or shapes subject to required constraints. Numerical optimization results in geometry shapes which have large off-design penalities. This is because the path to final design is bound by number of constraints and only one constraint can be met at a time during iterative process, please see Fig.(5).

Optimization of Computational Grids

Grids are either structured or unstructured. The later drew attention in the recent past. Arbitrarily shaped volume elements can be well handled by unstructured grids. Structural grids can be generated by algebraic or partial differential equation based or through conformal mapping. Algebraic grids have good computational efficiency since these are derived from functional relations. An example of unidirectional interpolation algebraic based grid generation is a below.

Linear fit between two end points is given by Eqn. (13) below:

$$r(\xi) = r(1) + \bar{\xi}\ \left[r\left(\xi_{\max}\right) - r\left(1\right)\right] \qquad (13)$$

where $\bar{\xi} = (\xi - 1)/(\xi_{\max} - 1)$

Physical plane coordinate (x,y or z) is related to computational index (ξ). Undirectional interpolation is uncontrollable boundary phenomenon. Grid boundary point condition can be satisfied with transfinite interpolation. An example of 2-d domain is as below.

$$\begin{aligned}\bar{r}(\xi,\eta) &= \left(1-\bar{\xi}\right) r\left(1-\bar{\eta}\right) + \bar{\xi}\, r\left(\xi_{\max},\eta\right) \\ &+ \left(1-\bar{\eta}\right) r\left(\xi,1\right) + \bar{\eta}\, r\left(\xi,\eta_{\max}\right) \\ &- \left(1-\bar{\xi}\right)\left(1-\bar{\eta}\right) r\left(1,1\right) - \left(1-\bar{\xi}\right)\bar{\eta}\, r\left(1,\eta_{\max}\right) \\ &- \left(\bar{\xi}\right)\left(1-\bar{\eta}\right) r\left(\xi_{\max},1\right) - \left(\bar{\xi}\right)\left(\bar{\eta}\right) r\left(\xi_{\max},\eta_{\max}\right)\end{aligned}$$

$$\text{where } \bar{\xi} = \left(\frac{\xi-1}{\xi_{\max}-1}\right),\ \bar{\eta} = \frac{\eta-1}{\eta_{\max}-1} \qquad (14)$$

Accuracy of computational techniques depend upon mesh optimization and proper distribution of grid points. Computational efficiency, accuracy and stability are enchanced if the mesh is graded in regions where the solutions are largely or/ and fast changing. A global objective function could consist of smoothness, orthogonality and/or solution adaptivity. Smoothness, orthogonality and adaptive grading can be expressed through serveral functions[4,6].

Completely unrelated course and fine grid can be used. Thus course grid can be designed to optimize the speed of convergence and the fine mesh using solution adaptivity, orthogonality etc. as objectives can be aimed to result in accurate solutions. The course to fine grid function propagation is possible through interpolation of a course mesh function ϕ^{2h} to a fine mesh locations x^h_{ij} , y^h_{ij} . Uniform interpolation and nonuniform interpolation approximation are used depending upon the curvi-nonlinearity of grids.

Effect of grid tropologies on the computational efficiency can be expressed in the form : $d\,\bar{\theta}/d\,t = A\,\bar{\theta} - \bar{f}$, where A is a large sparse matrix which encompasses flux Jacobian, grid data and space differencing. A peculiar matrix A could look like as shown in Fig.6a for a finite difference scheme in second order. In this the mesh coordinates are represented in X,Y and Z. A time march process retaining the accuracy order of algorithm is possible through factored fully implicit technique and Fig. 6b shows as to how the matrix [A]

Copyright © 1998, by the International Council of the Aeronautical Sciences (ICAS) and the American Institute of Aeronautics and Astronautics, Inc.

would become. Once arranged in this manner, it would also be possible to progress on computations in parallel at line intervals to retain accuracy or successive line over to retain accuracy and speed of convergence. The order of accuracy (p) of algorithm must satisfy $p \leq r + s$, where r are upwind data points and s are downwind data points. For a smaller time step it has been found that $p \leq 2 \min(r, s + 1)$.

Parallel Computational Computing

Computational algorithms run slow in sequential[7,8]. Thus these algorithms can be extensively paralleled and run in parallel in time hence parallel computational computing i.e. parallel algorithms on parallel machines provides speedup. However, this could result in deterioration in convergence rate due to paralleling of operations and, excessive data tracking and data management would be necessitated. This also involves in architectured parallel hardware support, otherwise the parallel algorithms could run chaotic. Modular parallel programming is also possible to provide architecture independence by featuring modularity. Efficiency of parallel computations can be seen from the following equation.

$$SR = \frac{\textit{Computing time on a serial machine}}{\textit{Computing time on a parallel machine}} \quad (15)$$

where SR is the speedup ratio and the possible SR forms could be following for p number of processors and machine dependent quantity (K), $O < K < I$.

SR forms	examples
$S = KP$	matrix calculations
$S = KP / \log_2 p$	tridiogonal linear system
$S = K \log_2 p$	search
$S = K$	certain nonlinear recurrences

An algorithm is a sequence of vector operations of varying length. Construction of parallel algorithms are idealised on two principles. The first principle is to convert serial algorithm into a procedure which operates on vectors, since vector operations can be carried in parallel. The second principle involves in vector iterations. This entails substituting an iterative parallel algorithm for a direct (non-iterative) serial algorithm. Speedup ratio depends on the ratio of steps needed in a direct version to those required by the iterations.

Numerical stability of parallel algorithms involves in stability, rounding of errors, and the propagation of errors in relation to parallel algorithms. A case is shown here where better results are possible while being in parallel. Considering the sum below, Eqn.(16), the errors are governed by Eqn.(17) and (18).

$$S_N = \sum_{K=1}^{N=2} aK \quad (16)$$

serial algorithm

$$\text{error} \leq 2^{-S+1} a \sum_{i=1}^{N} i = 2^{-S} a N (N+1) \quad (17)$$

parallel algorithm

$$\text{error} \leq \sum_{K=1}^{\log_2 N} Na \; 2^{-S+1} =$$

$$2^{-S+1} \; Na \log_2 N = 0 \, (N \log_2 N) \quad (18)$$

However, the effect of rounding of numerical digits could be significant. Equation below shows linear algebraic equation where accurate solution could be demonstrated through Eqn. (20), while doing computations.

$$Ax = b \quad (19)$$

$$(A + H) \bar{x} = b \quad (20)$$

where $\bar{x}$ is the computed solution. If H can be shown to be small, than algorithm is stable.

Copyright © 1998, by the International Council of the Aeronautical Sciences (ICAS)
and the American Institute of Aeronautics and Astronautics, Inc.

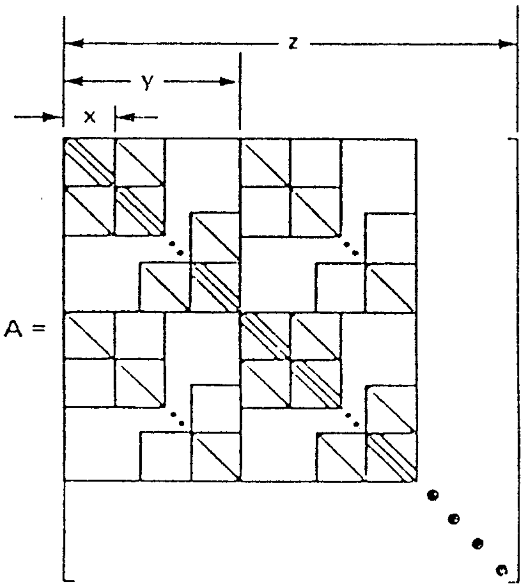

FIGURE 6b - Matrix Above Arranged For Easier Solution Process

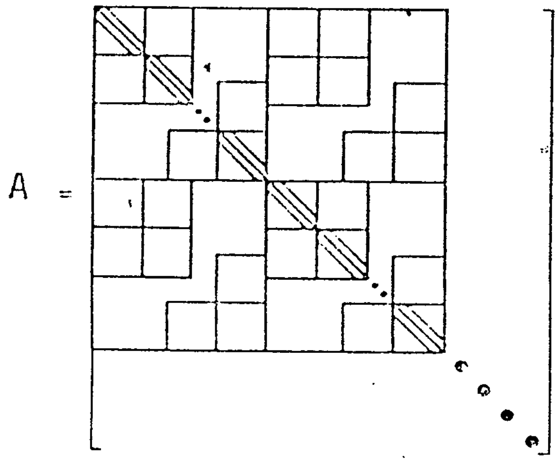

FIGURE 6a - Nature of Matrix in a Second Order Finite - Difference Technique

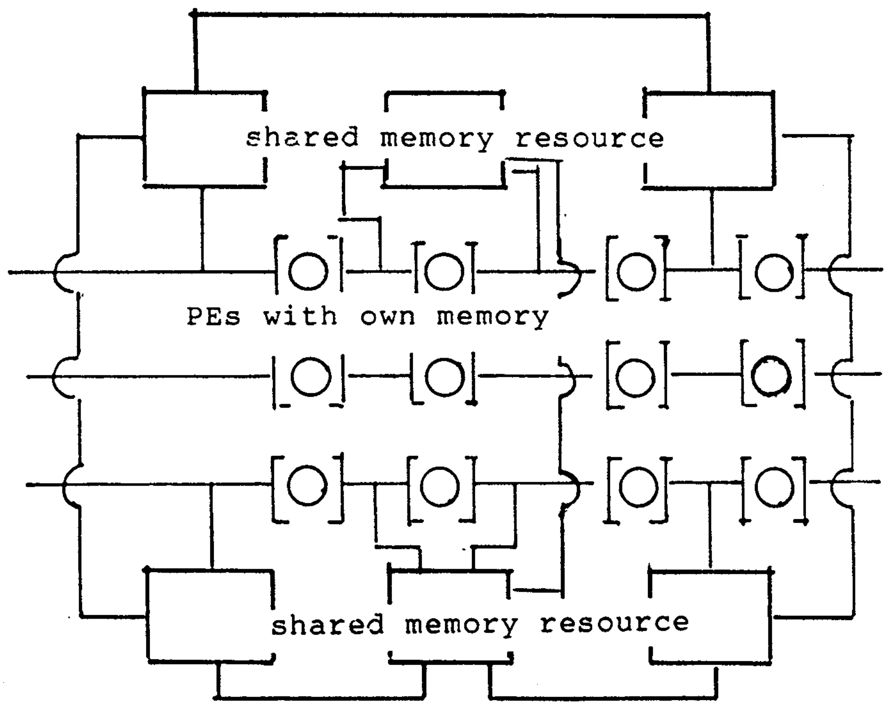

FIGURE 7 - Generic Requirement of Architecture

Copyright © 1998, by the International Council of the Aeronautical Sciences (ICAS) and the American Institute of Aeronautics and Astronautics, Inc.

Self dependencies in DO loops could be single or multiple. Intermittent nondependence could also occur inside the loops. It could be loop independent or loop carried. For example, A(I) = B * A(I - 1) + C * A(I) has loop independent nondependence whereas A(I) = B * A(I - 1) + C * (I + 1) has loop carried nondependence. Data dependency in iterations results in slowing down of iteration process and therefore data arrangements being crucial.

Pertinent arrangement of processors working on shared memeory resource are slow, whereas processor arrangement with adjacent memory block working on message passing are fast machines. Modular parallel programming is also possible for hardware architecture independence. In this the programmes are so constructed by being explicitly declared communication channels to plug together program modules called processors. Operations on channels can be restricted so as to guarantee deterministic execution. Computational algorithms can make much use of modular parallel programming. Figure 7 shows a architecture which is the most generic for flow field computations. Processing elements with their own memories could be used to progress iterations and retain data of immediate interest for next iteration. Inputs on a grid geometric variations could be ratained at the shared memory resource. Inter-processor communications is therefore minimised.

The data tracking being the most crucial if the grid geometry variation is simultaneously progressed. One thumb rule is to index all the iterations with the sequence of changes e.g. a parameter could be written as F[GRID (O), GRID STATION (I, J, K), Iteration (N)]. This would ensure nil error in data fetching, however excessive data and memory communication link would be required. All processors communicate (however, minimised) to all the processors in parallel computations. There are four well known exchange system on information in the processors. These are namely, linear exchange, pairwise exchange, recursive exchange and balance exchange. In the linear exchange, processor receive message from every other processor except itself. At a step i, $0 \leq i < p$, where i are the processors and p are the steps. In the case of pairwise exchange, at setp i, $1 \leq i \leq p - 1$, each processor exchanges a message with the processor determined by taking the exclusive OR of its processor number with i. In the case of recursive exchange, number of processors are halved in each step and each processor exchanges data with the corresponding processor in the other half. Number of steps required in this case are log(p). In each step i, $1 \leq i \leq \log(p)$ and each processor i exchanges with $j = \pm p/2^i$. Balance exchange processor is a cluster where first exchange is completely with the other processor in first cluster and then exchange with processors is in other cluster. In steps 0 to p/2-1, two processors in each cluster of size p/2 communicate across cluster while rest communicate within the cluster. In steps p/2 to p - 1, two processors in each cluster of size p/2 communicate within the cluster while other processors communcicate across cluster. While the performance of pairwise exchange recursive exchange and balance exchange is comparable, the linear exchange is very inferior.

Table - 1

Aerodynamic optimization effort for delta wing of aspect ratio of 2.25, $\alpha = 2°$

M	β A	$C_{d_i}/\beta C_L^2$		
		Before optimization	After optimization: $\overline{L}$ Constraint	After optimization: $\overline{L}, \overline{M_x}$ Constraint
1.05	.72	.83	.506	.5427
1.1	1.031	.6155	.388	.40
1.2	1.492	.436	.311	.311
1.3	1.87	.37	.277	.277
1.4	2.204	.329	.26	.26
1.5	2.51	.3	.25	.25
1.6	2.8	.285	.244	.245
1.8	3.36	.255	.24	.24

C_{d_i} is induced drag and C_L is lift coefficient.

Copyright © 1998, by the International Council of the Aeronautical Sciences (ICAS) and the American Institute of Aeronautics and Astronautics, Inc.

References

1. Iserles, A. and Powell, M.J.D., "The State of the Art in Numerical Analysis", Clearendron Press, Oxford (1987).

2. Gupta, S.C., "Aerodynamics Analysis and Design", (C) Interline, India (1995).

3. Holst, T.L and Ballhaus, W.F., "Fast Conservative Schemes for the Full Potential Equation Applied to Transonic Flows", AIAA Journal, Vol.17, No.2, February 1979.

4. Vanderplaats, G., "Numerical Optimization Techniques for Engineering Design", MC Graw-Hill Book Company, New York (1984).

5. Gupta, S.C., "Computer Code for Multi-Constraint Wing-Optimization", Journal of Aircraft, Vol.25, No.8 August 1988.

6. Pradhanai, Anand, and Garey, F. Graham, "Optimization of Computational Grids", Numerical Methods for Partial Differential Equations, 4, 95-117 (1988). (c) 1988 John Wiley & Sons, Inc.

7. Vijaya Shankar, "Research to Application-Supercomputing Trends for the 1990s, Opportunities for Interdisciplinary Computations", AIAA Paper No.91-0002.

8. Derek L. Eager; John Zahorjan and Edward D.Lozowska, "Speedup versus Efficiency in Parallel System", IEEE Transaction Computers, Vol.38, No.3 March 1989.

9. Ghosh, S.S and Gupta, S.C., "Accelerating the Finite Difference Relaxation Techniques", J.Ae. S.I., Vol. 40, No.4, November-December 1988.

10. Gupta, S.C., GENMAP: "Computer Code for Mission Adaptive Profile Generation," Journal of Aircraft, Vol.25, No.8, August 1988.

11. Gupta, S.C., "COPTIM: Computer Code for Canard Wing Optimization", Canadian Aeronautics and Space Journal, Vol.37, No.4, December 1991.