NASA Contractor Report 4085

Supersonic Flow Computations Over Aerospace Configurations Using an Euler Marching Solver

Kuo-Yen Szema, Sukumar Chakravarthy, and Vijaya Shankar

Rockwell International Science Center
Thousand Oaks, California

Prepared for
Langley Research Center
under Contract NAS1-15820

NASA
National Aeronautics
and Space Administration
Scientific and Technical
Information Office
1987
Foreword

This final report was prepared by the Science Center of Rockwell International, Thousand Oaks, California, for the Langley Research Center, National Aeronautics and Space Administration, Hampton, Virginia. The work was performed under Contract No. NAS1-15820, “Development of Full Potential and Euler Aero Prediction Methods for Supersonic/Hypersonic Configuration Design.” Mr. Noel Talcott and Mr. Kenneth Jones were the contract monitors of this contract.

Mr. E. Bonner of the Los Angeles Division, Rockwell International, was the Program Manager; Drs. K.-Y. Szema, S.R. Chakravarthy, and V. Shankar of the Rockwell International Science Center were the Principal Investigators.
Summary

An Euler marching algorithm for computing supersonic flows was developed by Dr. Chakravarthy as part of a NASA–Langley Research Center contract (NAS1–17492). The objective of the present contract (NAS1–15820) is to apply that Euler methodology to compute supersonic flows over realistic fighter-like configurations using the geometry/grid generation package developed for a similar full potential capability known as the SIMP (Supersonic Implicit Marching Potential) code, whose development was also funded by Contract NAS1–15820.

The Euler marching capability is termed “EMTAC” (Euler Marching Technique for Accurate Computation). The EMTAC code and the SIMP code have been extensively validated against each other in the Mach number range where the isentropic assumption is valid. The EMTAC code, being based on the exact inviscid gasdynamic equations, is valid for low and high supersonic Mach number computations exhibiting strong shocks and rotational effects. However, the use of Euler methods for computing vortex dominated flows is still unresolved and needs further investigation.

Several AIAA papers have been written describing the EMTAC methodology with comparisons of Euler results with the SIMP code and experimental data. The Appendix section of this report includes several of these papers.
## Contents

1. Introduction .............................................. 1  
2. Euler Method ............................................. 2  
3. Results ..................................................... 3  
4. Conclusions ................................................. 13  
5. References .................................................. 14  

Appendix A — Code Structure ............................... 16  
Appendix B — Publications ................................. 42

1. J. of Aircraft, Vol. 24, February 1987, pp. 73-83  
2. AIAA Paper 86-0244  
3. AIAA Paper 86-1834  
4. AIAA Paper 87-0592  
5. AIAA Paper 86-0440
1. INTRODUCTION

For fully supersonic flows, an efficient strategy for obtaining numerical solutions is to employ space-marching techniques. At low supersonic Mach numbers, realistic fighter configurations give rise to subsonic pockets near the canopy, wing-body junction, wing leading edge, and wing tip regions. A full potential marching technique\(^1\)\(^-\)\(^4\) capable of handling such embedded subsonic regions was developed as part of a NASA–Langley Research Center contract (NAS1-15820). The full potential method though very efficient for treating low supersonic Mach number flows (Mach number normal to a shock front is less than 1.3) is not capable of handling strongly shocked flows with rotational and vortex effects due to the underlying isentropic assumptions.

The objective of the present contract is to extend the full potential approach to the Euler equations which model the exact nonlinear inviscid gasdynamic flow processes. Within the assumption of an inviscid flow, such an Euler marching solver can be applied to a wide class of shocked flows including the hypersonic range. The intent of the Euler contract is to maintain some of the basic features of the full potential SIMP code\(^4\) within the Euler solver in dealing with geometry input, gridding techniques, and input/output routines including post processing of results.

The algorithm for the Euler marching solver was developed by Chakravarthy\(^5\) under a NASA contract, NAS1-17492. An Euler marching capability known as the "EMTAC" code ensuring compatibility with the full potential SIMP code has been developed. Results obtained for a variety of configurations involving canard, wing, horizontal tail, flow-through inlet, and fuselage using both the EMTAC and SIMP codes are reported in Refs. 5–9. Many of these papers are included in the Appendix of this report. For shocked cases satisfying the isentropic assumption (\(M_n < 1.3\)) with negligible entropy effects, the EMTAC and the SIMP codes produced practically identical results even for complex geometry configurations. In terms of execution time, the EMTAC code is about 5 to 10 times slower than the SIMP code since the Euler formulation solves five equations involving block tridiagonal inversions.
2. EULER METHOD

The Euler marching solver is described in detail in Ref. 5 and a copy of that paper is included in Appendix B.

Some of the salient features of the method are:

- Efficient space-marching technique based on unsteady Euler equations
- Finite volume upwind-biased scheme (modified Roe's approximate Riemann solver)
- High accuracy TVD formulation (up to third order)
- Approximate factorization in cross plane; forward marching for purely supersonic regions; Gauss-Seidel relaxation in marching direction for subsonic regions
- Proper treatment of wake-like grid topology
- Numerical grid generation (marching plane by marching plane)
- Nacelle treatment
- Code can also be easily used for inviscid 3-D flows which are fully subsonic or transonic (subsonic with supersonic pockets).

The EMTAC code is a single zone code just like the SIMP code. At present, the EMTAC code doesn't include the yaw capability for computing combined yaw and angle of attack cases (the SIMP code does). A multizone version of the EMTAC known as the EMTAC-MZ\textsuperscript{5} is currently under development which will accommodate any number of computational zones with proper flux balancing treatment at zonal boundaries. Treatment of combined yaw and angle of attack cases can be handled with ease using the EMTAC-MZ multizonal capability. The EMTAC code is currently operational on the VPS-32 at NASA-Langley Research Center.
3. RESULTS

The geometry input format for the EMTAC code is the same as that for the SIMP code\(^4\). See Appendix A for details.

Results obtained using the EMTAC code for a number of configurations are reported in Refs. 5–9 and some are included in Appendix B. The following configurations have been successfully computed using both the EMTAC and the SIMP codes:

1) Forebody geometry with a subsonic canopy region (Fig. 1)
2) Fighter configuration with vertical tail and flow-through nacelle (Fig. 2)
3) Shuttle Orbiter (Fig. 3)
4) Waverider (Fig. 4)
5) Shuttle–like configuration (Fig. 5)
6) Canard–wing fighter with nacelle (Fig. 6)
7) Wing–horizontal tail fighter with nacelle (Fig. 7)
8) Wing–body–strake configuration (Fig. 8).

The results for Cases 1–3 are reported in Ref. 5. Cases 4 and 5 are presented in AIAA Paper 86–0244. Cases 6–8 are included in AIAA Paper 87–0592.

In addition to these results, the Euler code was also tested for computing flows with vortex features. Numerical issues in computing supersonic vortex flows over conical delta wings are discussed in Ref. 10 (AIAA Paper 86–0440). Appendix B includes this paper also. References 11 and 12 also report discussions relevant to the use of an Euler solver for computing vortex flows. Figure 9 shows results for a conical flat plate delta wing at \(M_\infty = 2\), \(\alpha = 10^\circ\), \(\Lambda = 70^\circ\). Though Euler codes seem to produce the vortex features emanating from a sharp leading edge, computation of vortex flows around rounded leading edges still needs further study to understand the influence of numerical viscosity in predicting the correct location of the separation point.

AIAA Paper 86–1834\(^8\), included in Appendix B, includes SIMP code results for combined yaw and angle of attack cases.
Fig. 1. Forebody canopy geometry.
Fig. 2. Geometry and surface grid for a fighter with vertical tail and nacelle.
Fig. 3. Shuttle Orbiter configuration.
Fig. 4. Waverider geometry and grid.
Fig. 5. Shuttle-like configuration.
Fig. 6. Canard-wing fighter with nacelle.
Fig. 7. Wing-horizontal tail fighter with nacelle.
Fig. 8. Wing-body-strake configuration.
Fig. 9. Conical flat plate delta wing results. $M_\infty = 2$, $\alpha = 10^\circ$, $\Lambda = 70^\circ$. 
4. CONCLUSIONS

The full potential SIMP code and the EMTAC Euler code have matured into powerful nonlinear tools for computing supersonic flows over complex aerospace configurations with canard, wing, tail, fuselage, and flow-through nacelle. The geometry setup and grid generation are common to both the codes. Several configurations have been computed using both the SIMP and the EMTAC codes over a wide range of Mach number and angle of attack. For cases with weaker shocks (satisfying the isentropic assumption) the codes agreed very well with each other. The real use of the EMTAC code is in computing high Mach number flows with strong shock, rotational and vortex effects.

The codes are operational on the CRAY–XMP and the VPS–32 supercomputers. The SIMP code runs 5 to 10 times faster than the EMTAC code.
5. REFERENCES


APPENDIX A — CODE STRUCTURE

CODE ORGANIZATION

The EMTAC analysis code is applicable to arbitrary wing–body–nacelle–tail arrangements from moderate supersonic Mach numbers ($M_\infty \sim 1.2$) to values of the hypersonic range ($M_\infty \sim 40$). The lower code limit is governed by the extent of the embedded subsonic flow while the upper limit results from a breakdown in the perfect gas assumption for the flow.

The program is written in FORTRAN V language. It can be executed on supercomputers such as the CRAY–XMP and CYBER 205, as well as on superminicomputers such as the VAX and ELXSI. The program consists of a main routine (UDRIVE) and several subroutines. A brief description of the code along with input instructions needed to execute the code are given in this Appendix.

Program UDRIVE

Program UDRIVE coordinates the entire operation. A flowchart and subroutines describing the various operations performed by the UDRIVE program are given in Fig. A1. The UDRIVE program sets up the initial (known) data plane and the body–fitted grid system and performs the marching procedure to advance the solution. The various read and write tapes used in the calculation are listed below.

TAPE1 — Disk data input file containing starting solution to be read in for restart

TAPE2 — Disk data output file containing final solution to be stored in current run for later use

TAPE5 — Disk data input file containing input data needed (including the geometry data)

TAPE7 — Disk data output file to output solution in the form needed by plotting program and postprocessing
FLOW CHART (UDRIVE)

CALL UINPUT

DO 2000 IGLOB = ITERGS, ITERGE

CALL UNIT

DO 1000 NMAR = NMARS, NMA, NDMAR

CALL ISTEP

CALL UOUTPUT

1000 CONTINUE

2000 CONTINUE

STOP

END

NMARS = Starting counter for marching
NMARE = Ending counter for marching
NDMAR = 1 Forward marching
-1 Backward marching

Fig. A1. Flow chart for EMTAC code.
Fig. A1. Flow chart for EMTAC code (concluded).
Subroutine ISTEP

Subroutine ISTEP performs the marching procedure and updates the solution one step at a time.

Subroutine IAPFAC

The factored implicit scheme for the governing Euler equations can be written as

\[
\begin{align*}
I + \frac{1}{V} \hat{A}^{-1} \left\{ B_{k-1/2}^+ \Delta_{k-1/2} + B_{k+1/2}^- \Delta_{k+1/2} \right\} \\
I + \frac{1}{V} \hat{A}^{-1} \left\{ C_{i-1/2}^+ \Delta_{i-1/2} + C_{i+1/2}^- \Delta_{i+1/2} \right\} \Delta^* q \\
= \frac{1}{V} \hat{A}^{-1} \text{[Right Hand Side]}
\end{align*}
\]

The subroutine IAPFAC calls Subroutines ILHSL (left hand side \( L \)-direction), ILHSK (left hand side \( K \)-direction) and IRHS (right hand side) to calculate the solution by using the approximate factorization method.

Subroutine IROE

The numerical flux at cell surface \( m + 1/2 \) is given as

\[
h_{m+1/2} = \frac{1}{2} \left[ f \left( Q_{m+1}, N_{m+1/2} \right) + f \left( Q_m, N_{m+1/2} \right) \right] \\
- \frac{1}{2} \left[ \sum_i \left( \lambda_{m+1/2}^{i+} - \lambda_{m+1/2}^{i-} \right) \alpha_{2i}^r \right] \\
= f \left( Q_{m+1}, N_{m+1/2} \right) + \sum_i \lambda_{m+1/2}^{i-} \alpha_{2i}^r \\
= f \left( Q_{m+1}, N_{m+1/2} \right) - \sum_i \lambda_{m+1/2}^{i+} \alpha_{2i}^r
\]

where \( \alpha_i = \ell_i dQ \).

The right eigenvector \((r)\), left eigenvector \((\ell)\), and parameter \( \alpha \) are calculated in this subroutine.
Subroutines UBCLB, UBCLE, UBCKB, UBCKE

UBCLB: Apply boundary conditions at $L = 1$.
UBCLE: Apply boundary conditions at $L = LGRD$ (end of $L$).
UBCKB: Apply boundary conditions at $K = 1$.
UBCKE: Apply boundary conditions at $K = KGRD$ (end of $K$).

Subroutine MGEOM (N9, NRP)

$N9 = 0$, geometry data at $X_1$ and $X_2$ are read in
$> 0$, geometry data at $X_1$ is updated and $X_2$ is read in
$NRP = 0$, constant $x$ marching plane geometry calculation
$= 1$, spherical marching plane geometry calculation

Subroutine MGEOM sets up the body grid points from a prescribed geometry shape. From the input geometry points, a key point system is established using cubic splines. These key points are then joined from one prescribed geometry station to the next to provide the geometry at any intermediate marching plane.$^{12}$

Subroutine MGRID

Once the body points are obtained at a marching plane from MGEOM, subroutine MGRID sets up the entire crossflow plane grid using an elliptic grid solver that satisfies certain grid constraints.

Subroutine NFORCE (PX, PY, PM, AREA, KFG)

At the end of each marching plane calculation, this subroutine computes the axial force, $PX$, vertical force, $PY$, and the side force, $PZ$, by integrating the pressure force acting on an elemental area, $dA$.

$KFG = 0$, conical or blunt body nose force calculation
$= 1$, rest of the body force calculation.

The program also prints the force coefficients, $C_L$ and $C_D$, information based on a prescribed reference area, and moment coefficients, $C_M$, about a given reference point $(X_0, Y_0)$.
Header Data

A typical analysis of a complete configuration requires several regions of marching calculations for a complete analysis. Each region calculation has a different set of header instructions for describing grid parameters, wake information if pertinent, restart directions, and number of mesh points for each patch of the region. A sample input is given in Fig. A2, and a brief description of each variable is given in this section.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NMARCH</td>
<td>I5</td>
<td>Number of axial marching steps. If NMARCH = 0, and XSTART = ( \bar{\zeta} ) and DISKIN = F the code generates geometry and grid data at ( x = \bar{\zeta} ) for plotting. For NMARCH ( \neq 0 ), the code will march for NMARCH steps unless XEND is encountered first. NMARCH must include NCON iterations if applicable. (NMARCH = 0 option for grid plot is provided to allow the user to review the quality of grid at various axial stations before the flow solver is turned on.)</td>
</tr>
<tr>
<td>KMAX</td>
<td>I5</td>
<td>Mesh points in the normal direction (( \eta )). Present maximum is 30. This can be increased by increasing the dimension.</td>
</tr>
<tr>
<td>LMAX</td>
<td>I5</td>
<td>Mesh points in circumferential direction (( \xi )) (maximum value: 80). If this number is incorrectly specified, the code will reset LMAX properly using the ( LMAX = 1 + (IPT1-1) + (IPT2-1) + (IPT3-1) + \cdots + (IPTn-1) + 1 ) (definition of IPT follows in the next section). “n” is the number of patches.</td>
</tr>
</tbody>
</table>
Fig. A2. Sample input data for the EMTAC code.
<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NRM</td>
<td>I5</td>
<td>Number of grid regions (separated by dashed lines in Fig. A3). Maximum of 6 allowed.</td>
</tr>
<tr>
<td>NDISK</td>
<td>I5</td>
<td>Write restart data for every NDISK marching step.</td>
</tr>
<tr>
<td>NPRNT</td>
<td>I5</td>
<td>Print out solution for every NPRNT step.</td>
</tr>
</tbody>
</table>
| MRCHAC    | I5   | Accuracy parameter in marching direction.  
1: first order accuracy  
2: second or higher order accuracy  
(Also see SCHEME)  
Recommended value: 1 |
| CROSAC    | I5   | Accuracy parameter in L and K direction.  
1: first order accuracy  
2: second or higher order accuracy  
(Also see SCHEME)  
Recommended value: 2 |
| GLOBIT    | I5   | Number of internal iterations to perform before proceeding to next marching step.  
Recommended value: 2 |
Fig. A3. Cross section patches and nomenclature.
NCON 15 Number of iterations for conical starting solution (usually set to 30). To establish this starting solution, the geometry is initially assumed to be conical. The geometry at XSTART is projected forward conically to a point at (0,0,0). (The nose of the configuration is assumed to be at (0,0,0). If the geometry is not input this way, shift the geometry using PTNOSE and YSHIFT.) The solution is then obtained for this conical geometry based on NCON iterations. The conical solution is used as a starting solution for the nonconical case, beginning at XSTART.

The user should be aware that NCON is included in the NMARCH total. Also, XSTART output values have no physical significance during conical calculation.

NITER 15 Number of iterations to generate the marching grid using an elliptic grid solver. Usually set to 30. If grid routine fails, set this to 0 to analyze the geometry and the initial grid generated before grid relaxation (this is for debugging purposes). Set NITER back to 30 for flow field analysis. NMARCH should be set to zero for analyzing the grid quality.

BCONAC 15 On the B.C. surface, the solution is extrapolated.
0: set surface value equal to the first cell centroid values.
1: the value is obtained by using 2 points extrapolation
2: the value is obtained by using 3 points extrapolation. Recommended value: 1 or 2

LWKSU  I5  L value of starting point of a patch containing wake (Fig. A3).

LWKEL  I5  L value of ending point of a patch containing wake (Fig. A3).

ITERGS  I5  Starting number of global iterations for subsonic region calculations. Set to 1 for supersonic marching case.

ITERGE  I5  Ending number of global iterations for subsonic region calculations. Set to 1 for supersonic marching case. The number of global iterations = ITERGE=ITERGS.

CFLIN  F10.5  Not used.

DZTAIN  F10.5  Initial step size. For nonconical geometry calculations, DZTAIN is chosen to be either DZMIN or DZMAX. If DZTAIN is set to less than DZMAX, then during marching calculation, \( \Delta \zeta \) will be slowly increased to DZMAX.

DZMAX  F10.5  Maximum step size.

DZMIN  F10.5  Minimum step size. (DZMAX and DZMIN depend on the complexity of the geometry. Suggested value: DZMAX = total length/400 and DZMIN = DZMAX/2.) If DZMIN is set equal to DZMAX, then constant step size is used.
### FSMACH F10.5
Freestream Mach number.

### ALFA F10.5
Angle of attack (degrees).

### THTO F10.5
Angle of outer boundary (degrees).
This angle must be larger than the bow shock wave in order for the code to capture the bow shock. Often the best way to choose this value is to calculate the bow shock wave half angle and add 10°.

![Diagram of THTO](image)

### GAM F10.5
Ratio of specific heat.

### SCHEME F10.5
Parameter to pick particular TVD scheme.
- $\frac{1}{3}$: third order accurate scheme
- $-1$: fully second order upwind scheme
- $0$: Fromm’s second order scheme
- $\frac{1}{2}$: low truncation error second order scheme
Recommended value: $-1$

### CMPRES F10.5
Compression factor.
Choose in the range
$1 < \text{CMPRES} < \frac{3-\text{SCHEME}}{1-\text{SCHEME}}$
Normally pick $(3-\text{SCHEME})/(1-\text{SCHEME})$

### GLOBER F10.5
Not used.

### DETA

### DXI F10.5
Set to 0.1 (do not change).

### DZTA F10.5
Set to 0.1 (do not change).
XSTART  F10.5  Starting X location. If DISKIN = TRUE, this value is overwritten by stored restart value.

XEND    F10.5  Final X location for this run.

DTINOW F10.5  Inverse of the time step.
Set to 0.01 for supersonic flow.
For the subsonic flow region, set to ~ 10.0 and gradually decreasing to 0.01. The user provides the necessary update changes in Subroutine UDRIVE or through input variable ITERGS and ITERGE for this variation. Usually the variation from 10 to 0.01 can be imposed in ten time-relaxation sweeps.

DTISUB  F10.5  Not used.

DTISUP  F10.5  Not used.

XXX1    F10.5  Not used.

XWAKE   F10.5  Wake starting location in the axial direction (see Fig. A3).

ZWAKE   F10.5  Not used.

CHL     F10.5  Geometry scale factor. If set to total length, X will be scaled from 0 to 1. If set to 1, actual dimensions of the geometry are used. Use of dimensional (CHL = 1) or nondimensional (CHL = ℓ) option is left to user’s choice.
PTNOSE  F10.5  Axial geometry shift. Equal to negative of apex of the forebody (i.e., shifts configuration nose to $\zeta = 0$).

YSHIFT  F10.5  Vertical geometry shift (i.e., shifts configuration nose to $\eta = 0$).

XMO  F10.5  Moment reference X location (unit ~ length).

YMO  F10.5  Moment reference Y location (unit ~ length).

AAA  F10.5  Reference area to compute aerodynamic force coefficients (unit ~ length$^2$).

ALL  F10.5  Reference length to compute aerodynamic moment coefficients (unit ~ length).

XO, YO, AAA, and ALL are to be chosen (dimensional or nondimensional) based on CHL.

OMEGA  F10.5  Overrelaxation parameter for grid generation. Suggested value:

1.0 (for vectorized code)
1.75 (for scalar code).

OPRNT  L5  T: boundary output only
F: full output

NUGRID  L5  T: Numerical grid generation (normally used).
F: User must adapt code for his particular need.
IREAD     L5     T: Read body geometry input which must be supplied in the format described in the next section titled “Geometry Data”.

F: Analytic geometry (which must be supplied by the user and inserted in subroutine GRID).

RPLANE    L5     Not used.

DISKIN    L5     T: Restart the calculation.

F: Start the calculation from freestream.

TAPEW     L5     T: Write restart data on Tape 2.

F: No data storage for restart.

TAPE8W    L5     T: Write entire flow field data for subsonic iterations on Tape 8.

F: No flow field data saved.

FORCE     L5     T: Compute aerodynamic forces and moments.

F: No force computation.

THTU(5)    5I5    Grid region terminal points (k)

(see Fig. A3). These values are the K values of the points where the dashed lines intersect the body.

INU(5)    5F10.4 Polar angle (degrees) at respective terminal point.

ISC       I5     Number of patches (geometry) that define the cross-sectional shape of the configuration for this region of the configuration (see Fig. A4). (Maximum number of patches = 15.)
NPT(15) 15I5 Number of output points on each patch
(maximum number of points per patch is 30).
Fig. A4. Sample problem.
Geometry Data

The cross-sectional geometry of a typical aircraft changes considerably in the axial direction due to emergence of various components such as canopy, wing, nacelle, and tail, etc. The marching computation, as it sweeps along the marching direction $\xi$, has to account for this geometry variation to set up the proper body-fitted coordinate system to aid in the application of body boundary conditions. To treat complex geometry cross sections, patches are introduced to define the geometry as indicated in Fig. A4. Using patches, a configuration is defined by several regions of cross sections. The number of patches defining a section is constant for a given region (Fig. A4).

A complete computation over a configuration such as the one in Fig. A4 is usually done in segments rather than in one shot. The calculation starts from the nose and proceeds along $\xi$. Even within a region (defined by the same number of patches), the calculation might be done in segments using the restart option in the code. Restart is used any time the calculation is halted and then continued with another run that picks up where the previous run left off. Pure restart is performed only when there is no alteration to the number of points along $\eta$ and along $\xi$, and no change in the number of grid points per patch between the previous run and the current restart run. If there is any alteration to the grid structure, the restart run will automatically perform a respace operation to interpolate the solution from the previous solution grid to the current grid. Respace is used whenever the following situations are encountered:

1) Number of patches defining the cross section is changed. This situation occurs when the cross-sectional geometry becomes more complex. This is illustrated in Fig. A4.

2) Number of KGRD ($KGRD = KMAX - 1$) and/or LGRD ($LGRD = LMAX - 1$) points is changed (even if the number of patches defining the cross section is kept the same as before). This situation often occurs for cases where a patch length is increasing with $\xi$. For example, a swept wing is very small when it first appears in the cross section of the geometry and only requires a few grid points for accurate computation of the flow field. However, as the analysis is continued in the $\xi$ direction, the wing patches grow and will require more points for accurate flow field analysis.

3) Number of grid points per patch is changed (even if KGRD is kept the same as before).

Any time a respace is required, the code must be stopped. The code will automatically do a respace if KGRD or LGRD is different from the previous values of KGRD and LGRD.

One may be able to compute the entire configuration using the same number of patches and same KGRD and LGRD values throughout to avoid the respace requirement. This will
mean even in the forebody region of a configuration, where the cross-sectional geometry is usually simple, more grid points and more patches are to be used than necessary to adequately resolve the flow field. Use of the same number of patches and grid points for throughout the length of the configuration is generally not recommended. This can substantially increase the total execution time.

Transitioning from one region to the next (number of patches is changed) requires an overlapped zone, as illustrated in Fig. A5, to allow for increased or decreased number of patches in the next region. The extent of this overlapped zone must be sufficient to include at least the final three marching data planes of the prior region. In the overlapped region, the data from the previous region is interpolated onto the grids of the new region. For the example of Fig. A5, the results from the 3-patch region are interpolated onto a 4-patch region grid at the same $x$ location. This is required in order to continue marching along the body with the new patch definition.

Figure A5 illustrates how to transition from a fuselage computation to a wing–fuselage computation. First, the calculation is performed for the fuselage section denoted by REGION1 which ends just prior to the starting point of the wing. This calculation might involve, say, three patches. Then, to introduce the wing, a four patch representation is used in REGION2. In the overlapped zone, the fuselage which is defined using a three patch representation in REGION1 is represented by a four patch representation as part of REGION2. The second and third patch locations on the fuselage in REGION2 within the overlapped zone are chosen in the vicinity of where the leading edge of the wing is expected to emerge from the fuselage.
Fig. A5. Cross section patches in overlap region.
**Wake Geometry**

Behind the trailing edge of a lifting surface, a wake cut is introduced (see Fig. A3). The treatment of wake cut within the code requires the knowledge of starting and ending \( L \) index values of the upper wake cut and the lower one. Depending on the sweep of the trailing edge, the wake cut is appropriately modeled. This is illustrated in Fig. A3. The user has to define the shape of the trailing edge and also the starting \( x \) value in Subroutine MGRID where the wake begins to appear in the cross-sectional geometry (XWAKE). The wake cut is part of a patch which contains the wing also as illustrated in Fig. A3. As marching proceeds along the axial direction, the extent of the wake cut grows within that patch. The nomenclature for the starting and ending points of the wake cut are also indicated in Fig. A3. The number of points in the patch containing the wake cut is not allowed to change during the calculation. Thus, while exercising the respace option in the region containing the wake, the user has to ensure that the number of points in the wake patch (usually there are two wake patches; one corresponding to the upper cut and one for the lower cut) is not altered.

The shape of the trailing edge is provided by the user using the update option.

For the wing-body-vertical case of Fig. A4, a 3-patch initial region, a 6-patch center region, and an 8-patch final region was used. Zero length patches are not permissible. Since the analysis is marching in nature, a complete geometry data set is not required to begin and partially process a problem. Appropriate use of restart solutions allows continuation of the analysis as new or modified geometry becomes available.

The format for a typical station is shown below. The group of cards is repeated for each station of a region. The last point of each patch (except for the last patch of a station) should have the same coordinates as the first point of the next patch.

<table>
<thead>
<tr>
<th>Card No.</th>
<th>Format</th>
<th>Field</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>F15.6,I5</td>
<td>1</td>
<td>X1</td>
<td>The ( x ) value (longitudinal) of this station.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>ISC1</td>
<td>The number of patches for this section. ( 1 \leq \text{ISC1} \leq 15 ).</td>
</tr>
</tbody>
</table>

The group of cards A2 through A3 are repeated ISC1 times.

A2       | 3I5    | 1     | ITH  | Patch number \( \leq 15 \). |
The A3 card is repeated IPT times.

<table>
<thead>
<tr>
<th>2</th>
<th>IPT</th>
<th>Number of points in this patch. $2 \leq IPT \leq 30$.</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>ND</td>
<td>Mesh spacing parameters*. Typically the same for all stations of a region.</td>
</tr>
</tbody>
</table>

Cubic spline interpolation is performed on input patch data to derive the geometry. Linear interpolation is performed to define the geometry at a marching plane between input stations.

Sample geometry data for the problem of Fig. A4 is presented in Table 1 and was developed using CDS$^{13}$.

*For Segment AB: 0 equal space; 1 cluster near A; 2 cluster near B.
<table>
<thead>
<tr>
<th>Station</th>
<th>Geometry Data</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>First Patch</td>
</tr>
<tr>
<td></td>
<td>Second Patch</td>
</tr>
<tr>
<td>1</td>
<td>Third Patch</td>
</tr>
<tr>
<td>1.548782</td>
<td>0.984999</td>
</tr>
<tr>
<td>1.54643</td>
<td>0.800923</td>
</tr>
<tr>
<td>1.545183</td>
<td>0.813005</td>
</tr>
<tr>
<td>1.54843</td>
<td>0.899744</td>
</tr>
<tr>
<td>1.532095</td>
<td>0.951806</td>
</tr>
<tr>
<td>1.52201</td>
<td>0.896484</td>
</tr>
<tr>
<td>1.51055</td>
<td>0.870012</td>
</tr>
<tr>
<td>1.495798</td>
<td>0.860136</td>
</tr>
<tr>
<td>1.49303</td>
<td>0.916745</td>
</tr>
<tr>
<td>1.46389</td>
<td>0.101503</td>
</tr>
<tr>
<td>1.45534</td>
<td>0.103120</td>
</tr>
<tr>
<td>1.45395</td>
<td>0.114202</td>
</tr>
<tr>
<td>1.41823</td>
<td>0.101520</td>
</tr>
<tr>
<td>1.39432</td>
<td>0.860137</td>
</tr>
<tr>
<td>1.36708</td>
<td>0.864460</td>
</tr>
<tr>
<td>1.35768</td>
<td>0.851017</td>
</tr>
<tr>
<td>1.35841</td>
<td>0.85787</td>
</tr>
<tr>
<td>1.34759</td>
<td>0.818250</td>
</tr>
<tr>
<td>1.34157</td>
<td>0.808000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Station</th>
<th>Geometry Data</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>First Station</td>
</tr>
<tr>
<td></td>
<td>Second Station</td>
</tr>
</tbody>
</table>

| 1.512623| 0.800000 |
| 0.95236 | 1.704391 |
| 0.85226 | 0.704301 |
| 0.70010 | 3.104900 |
| 0.71674 | 4.202395 |
| 0.71674 | 4.202395 |
| 0.71208 | 5.320932 |
| 0.19479 | 6.250178 |
| 0.56149 | 7.095905 |
| 0.40377 | 7.739478 |
| 0.31990 | 8.233154 |
| 0.31522 | 8.465744 |
| 0.31522 | 8.465744 |

| 1.02745 | 0.274061 |
| 1.14000 | 0.707605 |
| 0.64931 | 0.734938 |
| 0.26573 | 0.517667 |
| 0.86710 | 0.517667 |
| 0.86710 | 0.517667 |
| 0.38635 | 4.317586 |
| 0.38423 | 2.167079 |
| 0.96707 | 1.507678 |
| 0.96707 | 1.507678 |
| 0.96707 | 1.507678 |

**Table 1. Geometry Data**
<table>
<thead>
<tr>
<th>N</th>
<th>( x )</th>
<th>( z )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1.42648</td>
<td>( 193.784922 )</td>
</tr>
<tr>
<td>0</td>
<td>1.358828</td>
<td>( 101.145747 )</td>
</tr>
<tr>
<td>0</td>
<td>1.284687</td>
<td>( 06.947678 )</td>
</tr>
<tr>
<td>0</td>
<td>1.058096</td>
<td>( 04.368501 )</td>
</tr>
<tr>
<td>0</td>
<td>0.917631</td>
<td>( 00.622636 )</td>
</tr>
<tr>
<td>0</td>
<td>0.640671</td>
<td>( 66.637686 )</td>
</tr>
<tr>
<td>0</td>
<td>0.257427</td>
<td>( 61.430713 )</td>
</tr>
<tr>
<td>0</td>
<td>-8.130001</td>
<td>( 76.311234 )</td>
</tr>
<tr>
<td>0</td>
<td>-0.450283</td>
<td>( 71.150048 )</td>
</tr>
<tr>
<td>0</td>
<td>-0.455550</td>
<td>( 65.007915 )</td>
</tr>
<tr>
<td>0</td>
<td>-0.942980</td>
<td>( 61.979340 )</td>
</tr>
<tr>
<td>0</td>
<td>-1.161150</td>
<td>( 57.469606 )</td>
</tr>
<tr>
<td>0</td>
<td>-1.204498</td>
<td>( 52.036866 )</td>
</tr>
<tr>
<td>0</td>
<td>-1.473180</td>
<td>( 45.258186 )</td>
</tr>
<tr>
<td>0</td>
<td>-1.456888</td>
<td>( 36.650965 )</td>
</tr>
<tr>
<td>0</td>
<td>-1.600638</td>
<td>( 27.401102 )</td>
</tr>
<tr>
<td>0</td>
<td>-1.718206</td>
<td>( 18.323632 )</td>
</tr>
<tr>
<td>0</td>
<td>-1.914595</td>
<td>( 9.163994 )</td>
</tr>
<tr>
<td>0</td>
<td>-1.968518</td>
<td>( 0.800000 )</td>
</tr>
<tr>
<td>0</td>
<td>257.726203</td>
<td>( 6 )</td>
</tr>
<tr>
<td>0</td>
<td>56.810779</td>
<td>( 0.000000 )</td>
</tr>
<tr>
<td>0</td>
<td>56.810779</td>
<td>( 6.919568 )</td>
</tr>
<tr>
<td>0</td>
<td>40.824403</td>
<td>( 3.40812 )</td>
</tr>
<tr>
<td>0</td>
<td>40.187465</td>
<td>( 6.64347 )</td>
</tr>
<tr>
<td>0</td>
<td>40.833110</td>
<td>( 0.890950 )</td>
</tr>
<tr>
<td>0</td>
<td>46.350802</td>
<td>( 12.845306 )</td>
</tr>
<tr>
<td>0</td>
<td>46.499002</td>
<td>( 12.945306 )</td>
</tr>
<tr>
<td>0</td>
<td>44.232117</td>
<td>( 15.28275 )</td>
</tr>
<tr>
<td>0</td>
<td>41.709866</td>
<td>( 17.06559 )</td>
</tr>
<tr>
<td>0</td>
<td>90.201430</td>
<td>( 18.72600 )</td>
</tr>
<tr>
<td>0</td>
<td>91.837119</td>
<td>( 18.92102 )</td>
</tr>
<tr>
<td>0</td>
<td>91.263851</td>
<td>( 19.32878 )</td>
</tr>
<tr>
<td>0</td>
<td>34.538315</td>
<td>( 10.328766 )</td>
</tr>
<tr>
<td>0</td>
<td>34.538315</td>
<td>( 10.328766 )</td>
</tr>
<tr>
<td>0</td>
<td>32.000007</td>
<td>( 21.765604 )</td>
</tr>
<tr>
<td>0</td>
<td>20.693047</td>
<td>( 26.516281 )</td>
</tr>
<tr>
<td>0</td>
<td>20.452254</td>
<td>( 31.036555 )</td>
</tr>
<tr>
<td>0</td>
<td>23.468408</td>
<td>( 37.25181 )</td>
</tr>
<tr>
<td>0</td>
<td>21.147877</td>
<td>( 42.08042 )</td>
</tr>
<tr>
<td>0</td>
<td>20.270148</td>
<td>( 43.02132 )</td>
</tr>
<tr>
<td>0</td>
<td>19.470877</td>
<td>( 46.844967 )</td>
</tr>
<tr>
<td>0</td>
<td>19.717727</td>
<td>( 48.34198 )</td>
</tr>
<tr>
<td>0</td>
<td>17.326115</td>
<td>( 53.30514 )</td>
</tr>
<tr>
<td>0</td>
<td>17.326115</td>
<td>( 53.30514 )</td>
</tr>
<tr>
<td>0</td>
<td>16.954680</td>
<td>( 59.162178 )</td>
</tr>
<tr>
<td>0</td>
<td>14.963228</td>
<td>( 65.008917 )</td>
</tr>
<tr>
<td>0</td>
<td>13.750480</td>
<td>( 68.714011 )</td>
</tr>
<tr>
<td>0</td>
<td>13.102150</td>
<td>( 71.15070 )</td>
</tr>
<tr>
<td>0</td>
<td>12.580350</td>
<td>( 73.73709 )</td>
</tr>
<tr>
<td>0</td>
<td>11.936160</td>
<td>( 76.31263 )</td>
</tr>
<tr>
<td>0</td>
<td>11.212303</td>
<td>( 78.90149 )</td>
</tr>
<tr>
<td>0</td>
<td>10.451151</td>
<td>( 81.47466 )</td>
</tr>
<tr>
<td>Parameter</td>
<td>Value 1</td>
<td>Value 2</td>
</tr>
<tr>
<td>-----------</td>
<td>---------</td>
<td>---------</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Table 1. Geometry Data (continued)
This Appendix contains the following papers:

2. AIAA Paper 86–0244
3. AIAA Paper 86–1834
4. AIAA Paper 87–0592
5. AIAA Paper 86–0440

Permission to reprint the papers appearing in this Appendix was granted by the AIAA.
At the 24th Aerospace Sciences Meeting, where this paper was presented, we learnt from R. W. Newsome that 1) Newsome had tried fixed time steps with his MacCormack scheme formulation and still observed large cross-flow separation, and 2) Newsome had also tried spatially varying time steps with the upwind-biased formulation and yet failed to observe large separation. In this paper, we have shown that variable time steps can lead to a different solution for the delta-wing problem considered. But this conclusion may be valid for only certain ranges of recipes for varying the time step spatially and over the sequence of time steps. While this may serve as a good counter-example (to the argument that the solution with large cross-flow separation is only due to numerical diffusion on the coarse grid), more research must surely be performed to understand other possible mechanisms.

8.0 References


**Abstract**

For fully supersonic flows, an efficient strategy for obtaining numerical solutions is to employ space-marching techniques. At low supersonic Mach numbers, realistic fighter configurations give rise to subsonic pockets near the canopy, wing-body junction, wing leading edge, and wing tip regions. A full potential marching technique, known as the SIMP code and capable of handling such embedded subsonic regions, has achieved some success analyzing low supersonic Mach number flows. The SIMP code, however, is not capable of handling strongly shocked flows with rotational and vortex effects due to the underlying isentropic assumptions.

This paper presents the extension of the full potential approach to the Euler equations which model the exact nonlinear inviscid gas dynamic flow processes. Within the assumption of an inviscid flow, such an Euler marching solver can be applied to a wide class of shocked flows including the hypersonic range. The intent is to maintain some of the basic features of the full potential SIMP code within the Euler solver in dealing with geometry input, gridding techniques, and input/output routines including post processing of results. The Euler marching code known as EMTAC has been developed. Results obtained for a variety of configurations involving canard, wing, horizontal tail, flow-through inlet, and fuselage using both the EMTAC and SIMP codes are reported. For shocked cases satisfying the isentropic assumption, the EMTAC and SIMP codes produced practically identical results. In terms of execution time, the EMTAC code is 5 to 10 times slower than the SIMP code since the Euler formulation solves five equations involving block tridiagonal inversions. Both codes are operational on the CRAY-XMP and VPS-32 supercomputers.