Flux-Vector Splitting Algorithm for Chain-Rule Conservation-Law Form

E. Steinthorsson,* Z. Li,* and T. I-P. Shih†
Carnegie Mellon University, Pittsburgh, Pennsylvania 15213
and
H. L. Nguyen‡ and E. A. Willis§
NASA Lewis Research Center, Cleveland, Ohio 44135

A flux-vector splitting algorithm with Newton-Raphson iteration was developed for the "full compressible" Navier-Stokes equations cast in chain-rule conservation-law form. The algorithm is intended for problems with deforming spatial domains and for problems whose governing equations cannot be cast in strong conservation-law form. The usefulness of the algorithm for such problems was demonstrated by applying it to analyze the unsteady, two- and three-dimensional flows inside one combustion chamber of a Wankel engine under nonfiring conditions. Solutions were obtained to examine the algorithm in terms of conservation error, robustness, and ability to handle complex flows on time-dependent grid systems.

Introduction

In order to develop algorithms for studying flow problems in arbitrary geometries, the Euler and Navier-Stokes equations are often written in generalized coordinates, and cast in either chain-rule, weak, or strong conservation-law forms.[1] In general, strong conservation-law form is used because it permits conservative differencing and can capture shocks automatically. However, for deforming spatial domains that require time-dependent grid systems, strong conservation-law form can produce geometric-induced errors that are not readily removable.[2-5] Thus, an alternative approach for time-dependent grid systems is to use chain-rule conservation-law form. Although the chain-rule form is nonconservative, it does not produce any geometric-induced errors for time-dependent grid systems and can capture shocks correctly.[3] Another reason for using the chain-rule form is that there are forms of the Navier Stokes equations as well as other partial differential equations that cannot be cast in strong conservation-law form but can readily be cast in chain-rule conservation-law form.

Since an upwind algorithm has not been reported for the Euler or the Navier-Stokes equations cast in chain-rule form, the objective of this study is to develop a flux-vector splitting algorithm with Newton-Raphson iteration for the "full compressible" Navier-Stokes equations cast in chain-rule conservation-law form on a time-dependent grid system. The usefulness of the algorithm developed in this study was demonstrated by applying it to analyze the unsteady, two- and three-dimensional flows in one combustion chamber of a Wankel engine under nonfiring conditions. This is a stringent test problem because combustion chambers of Wankel engines deform considerably in volume and shape, and the flow is complex with large separated regions. For this test problem, solutions were generated to examine the algorithm in terms of conservation error, robustness, and ability to handle complex flows on time-dependent grid systems.

Received April 23, 1990; revision received Nov. 30, 1990; accepted for publication Dec. 27, 1990.

*Graduate Student, Department of Mechanical Engineering.
†Associate Professor, Department of Mechanical Engineering. Member AIAA.
‡Aerospace Engineer, Propulsion Systems Division. Member AIAA.
§Chief Engineer, Propulsion Systems Division. Member AIAA.

Solution Algorithm

In an inertial Cartesian coordinate system, the "full compressible" Navier-Stokes equations have the following form:

$$\frac{\partial U}{\partial t} + \frac{\partial E}{\partial x} + \frac{\partial F}{\partial y} + \frac{\partial G}{\partial z} = \frac{\partial R}{\partial x} + \frac{\partial S}{\partial y} + \frac{\partial T}{\partial z} \tag{1}$$

where U is a vector of conserved variables; E, F, and G represent fluxes of mass, momentum, and energy in the x, y, and z directions, respectively; and R, S, and T are the diffusion terms.

The chain-rule form of Eq. (1) is derived by replacing all occurrences of $\partial/\partial t$, $\partial/\partial x$, $\partial/\partial y$, and $\partial/\partial z$ by $\partial/\partial\tau + \xi_t\partial/\partial\xi + \eta_t\partial/\partial\eta + \zeta_t\partial/\partial\zeta$, $\xi_x\partial/\partial\xi + \eta_x\partial/\partial\eta + \zeta_x\partial/\partial\zeta$, $\xi_y\partial/\partial\xi + \eta_y\partial/\partial\eta + \zeta_y\partial/\partial\zeta$, and $\xi_z\partial/\partial\xi + \eta_z\partial/\partial\eta + \zeta_z\partial/\partial\zeta$, respectively, where ξ_t, η_t, ζ_t, ξ_x, η_x, ζ_x, ξ_y, η_y, ζ_y, ξ_z, η_z, and ζ_z are metric coefficients. The chain-rule form of Eq. (1) can be written as

$$\frac{\partial U}{\partial \tau} = -W(U)$$

and

$$W(U) = W_\xi(U) + W_\eta(U) + W_\zeta(U) \tag{2a}$$

where

$$W_\xi(U) = \xi_t\frac{\partial}{\partial\xi}U + \xi_x\frac{\partial}{\partial\xi}(E - R) + \xi_y\frac{\partial}{\partial\xi}(F - S) + \xi_z\frac{\partial}{\partial\xi}(G - T) \tag{2b}$$

$$W_\eta(U) = \eta_t\frac{\partial}{\partial\eta}U + \eta_x\frac{\partial}{\partial\eta}(E - R) + \eta_y\frac{\partial}{\partial\eta}(F - S) + \eta_z\frac{\partial}{\partial\eta}(G - T) \tag{2c}$$

$$W_\zeta(U) = \zeta_t\frac{\partial}{\partial\zeta}U + \zeta_x\frac{\partial}{\partial\zeta}(E - R) + \zeta_y\frac{\partial}{\partial\zeta}(F - S) + \zeta_z\frac{\partial}{\partial\zeta}(G - T) \tag{2d}$$

The algorithm developed in this study for Eq. (2) is described in the following five steps: 1) time differencing, 2) Newton-Raphson iteration, 3) approximate factorization, 4) flux-vector splitting for chain-rule form, and 5) block tridiagonal structure with second-order upwind differencing.

Time Differencing

All time derivatives were approximated by the following three-time level generalized time-differencing formula[6]:

$$\left(\frac{\partial U}{\partial \tau}\right)^{n+1} = \frac{1}{1+\theta_1}\frac{\Delta U^{n+1}}{\Delta \tau} - \frac{\theta_2}{1+\theta_2}\frac{\Delta U^n}{\Delta \tau}, \qquad \Delta U^{n+1} = U^{n+1} - U^n \tag{3}$$

In the above equation, superscripts n and $(n+1)$ denote time levels; U^n and U^{n-1} are known quantities and U^{n+1} is the unknown sought; $\Delta\tau$ is the time step size; and θ_1 and θ_2 are constants (e.g., $\theta_1 = \theta_2 = 0$ gives Euler implicit, and $\theta_1 = -1/3$ and $\theta_2 = 1$ give the three-point backward formula). Substitution of Eq. (2) into Eq. (3) and rearranging yield

$$\Phi(U^{n+1}) = \Delta U^{n+1} + \Delta\tau\,(1+\theta_1)\,W(U^{n+1}) - \frac{\theta_2\,(1+\theta_1)}{1+\theta_2}\Delta U^n = 0 \tag{4}$$

Newton-Raphson Iteration

The term $\Phi(U^{n+1})$ given by Eq. (4) is a nonlinear function of the unknown U^{n+1}. Here, Newton-Raphson iteration is used to linearize Eq. (4) as shown below:

$$\Phi(U^{n+1}) \approx \Phi(U^m) + \left\{\frac{\partial \Phi(U^m)}{\partial U}\right\}\Delta U^{m+1} = 0 \tag{5}$$

In the above equation, the superscript m denotes the number of iterations, with $m = 0$ being the first iteration, $m = 1$ being the second iteration, and so on. If only one iteration is used per time step, then Eq. (5) reduces to the time-linearization formula used in the implicit-factored method of Beam/Warming[6] and Briley/McDonald.[7]

At this point, it is important to note that all metric coefficients in Eq. (5) are evaluated at time level $(n+1)$; i.e., the metric coefficients are not linearized. This can be done if all information concerning the grid system is known at both time levels n and $(n+1)$, which is the case for many problems.

Substitution of Eq. (4) into Eq. (5) gives

$$\left\{I + \Delta\tau\,(1+\theta_1)\frac{\partial\, W(U^m)}{\partial U}\right\}\Delta U^{m+1} = \text{RHS} \tag{6a}$$

where I is a unit matrix; $W(U^m)$ is given by Eq. (2); and

$$\text{RHS} = -(U^m - U^n) + \frac{\theta_2(1+\theta_1)}{1+\theta_2}\Delta U^n - \Delta\tau\,(1+\theta_1)\,W(U^m) \tag{6b}$$

Note that during the first iteration, $m = 0$ and $U^m = U^n$.

Approximate Factorization

Equation (6) can be approximately factored in several different ways; e.g., LU and ADI. Here, ADI is used since this approximate factorization procedure easily allows viscous terms to be treated implicitly. By using ADI, Eq. (6) becomes

$$L_\xi \quad L_\eta \quad L_\zeta\,\Delta U^{m+1} = \text{RHS} \tag{7a}$$

In the above equation, L_ξ, L_η, and L_ζ are differential operators in the ξ, η, and ζ directions, respectively, and each are given by

$$L_j = \left\{I + \Delta\tau\,(1+\theta_1)\frac{\partial W_j(U^m)}{\partial U}\right\} \tag{7b}$$

where $j = \xi$, η, or ζ and $W_j(U^m)$ is given by Eqs. (2b–2d). The error created by the approximate factorization is second-order in time. In principle, this error can be completely eliminated by the Newton-Raphson iteration since the converged solution at each time step depends only on RHS and is independent of the left-hand side of Eq. (7a).

Once the differential operator has been approximately factored as shown in Eq. (7a), it can be split as shown below:

$$L_\xi\,\Delta U^* = \text{RHS} \tag{8a}$$

$$L_\eta\,\Delta U^{**} = \Delta U^* \tag{8b}$$

$$L_\zeta\,\Delta U^{m+1} = \Delta U^{**} \tag{8c}$$

Flux-Vector Splitting in Chain-Rule Form

In the implicit-factored method of Beam/Warming[6] and Briley/McDonald,[7] all spatial derivatives are replaced by second-order accurate central-difference formulas. Here, spatial derivatives corresponding to diffusion terms are still approximated by second-order accurate central-difference formulas, but spatial derivatives corresponding to convection terms are approximated by upwind differencing via flux-vector splitting.

To illustrate this flux-vector splitting scheme for governing equations cast in chain-rule form, consider Eq. (8a). By using Eqs. (2b) and (7b), Eq. (8a) can be written in the following expanded form:

$$\left\{I + \Delta\tau\,(1+\theta_1)\left(\xi_t^{n+1}\frac{\partial}{\partial\xi}I + \xi_x^{n+1}\frac{\partial}{\partial\xi}(A_E - A_R)^m + \xi_y^{n+1}\frac{\partial}{\partial\xi}(A_F - A_S)^m + \xi_z^{n+1}\frac{\partial}{\partial\xi}(A_G - A_T)^m\right\}\right.$$
$$\Delta U^* = \text{RHS} \tag{9}$$

where the Jacobians A_E, A_F, A_G, A_R, A_S, and A_T are given by $\partial E/\partial U$, $\partial F/\partial U$, $\partial G/\partial U$, $\partial R/\partial U$, $\partial S/\partial U$, and $\partial T/\partial U$, respectively.

In Eq. (9), the Jacobians A_R, A_S, and A_T correspond to diffusion terms; derivatives involving these Jacobians are all approximated by second-order accurate central-difference formulas with all cross derivative terms treated explicitly by second-order accurate linear extrapolation in iteration space. The remaining derivatives are to be upwind differenced. The upwind differencing of these derivatives in Eq. (9) must be done with respect to the direction of the characteristic base curves in the $\tau - \xi$ plane. Since these derivatives are all multiplied by a metric coefficient, the direction of the characteristics corresponding to each derivative depends both on the sign of the metric coefficient and on the sign of the eigenvalues of I, A_E, A_F, or A_G. Accordingly, these derivatives must be decomposed as shown below before they can be treated by upwind differencing:

$$\left(\xi_t^{n+1}\frac{\partial}{\partial\xi}I\right)\Delta U^* = \xi_t^{n+1}\frac{\partial}{\partial\xi}\{(I^+ + I^-)\,\Delta U^*\} \tag{10a}$$

$$\left(\xi_x^{n+1}\frac{\partial}{\partial\xi}A_E\right)\Delta U^* = \xi_x^{n+1}\frac{\partial}{\partial\xi}\{(A_E^+ + A_E^-)\,\Delta U^*\} \tag{10b}$$

$$\left(\xi_y^{n+1}\frac{\partial}{\partial\xi}A_F\right)\Delta U^* = \xi_y^{n+1}\frac{\partial}{\partial\xi}\{(A_F^+ + A_F^-)\,\Delta U^*\} \tag{10c}$$

$$\left(\xi_z^{n+1}\frac{\partial}{\partial\xi}A_G\right)\Delta U^* = \xi_z^{n+1}\frac{\partial}{\partial\xi}\{(A_G^+ + A_G^-)\,\Delta U^* \tag{10d}$$

In the above equation, matrices I^+ and A_i^+ ($i = E$, F, and G) are defined such that $\text{sign}(\xi_t)I^+$, $\text{sign}(\xi_x)A_E^+$, $\text{sign}(\xi_y)A_F^+$, and $\text{sign}(\xi_z)A_G^+$ all have zero or positive eigenvalues. Similarly, matrices I^- and A_i^- are defined so that $\text{sign}(\xi_t)I^-$, $\text{sign}(\xi_x)A_E^-$, $\text{sign}(\xi_y)A_F^-$, and $\text{sign}(\xi_z)A_G^-$ all have zero or negative eigenvalues. Based on this decomposition, derivatives involving I^+ and A_i^+ should be backward differenced, and derivatives involving I^- and A_i^- should be forward differenced.

There are many different ways to split the Jacobians that would satisfy Eq. (10).[8-10] Here, the splitting proposed by Jameson and Turkel[8] was modified for governing equations cast in chain-rule form as shown below:

$$I^+ = \tfrac{1}{2}(I + \nu_I I), \qquad I_i^- = \tfrac{1}{2}(I_i - \nu_I I) \tag{11a}$$

$$A_i^+ = \tfrac{1}{2}(A_i + \nu_i I), \qquad A_i^- = \tfrac{1}{2}(A_i - \nu_i I) \tag{11b}$$

where I is a unit matrix; $i = E$, F, or G; and

$$\nu_I = \frac{\xi_t^{n+1}}{|\xi_t^{n+1}|}, \qquad \nu_E = \frac{\xi_x^{n+1}}{|\xi_x^{n+1}|}\rho(A_E)$$

$$\nu_F = \frac{\xi_y^{n+1}}{|\xi_y^{n+1}|}\rho(A_F), \qquad \nu_G = \frac{\xi_z^{n+1}}{|\xi_z^{n+1}|}\rho(A_G) \tag{11c}$$

$$\rho(A_E) = |u| + c, \qquad \rho(A_F) = |v| + c$$

$$\rho(A_G) = |w| + c \tag{11d}$$

In Eq. (11d), $\rho(A_i)$ is the spectral radius of A_i; c is the local speed of sound; and u, v, and w are the local fluid velocity in the x, y, and z directions, respectively. Here, it is noted that $\xi_j^{n+1}/|\xi_j^{n+1}| = +1$ or -1, where $j = t$, x, y, and z, so that derivatives with respect to these terms are zero except where they change sign. By using this information, substitution of Eq. (11) into Eq. (10) followed by differentiation yields

$$\left(\xi_t^{n+1}\frac{\partial}{\partial\xi}I^+\right)\Delta U^* = \frac{1}{2}\left\{\xi_t^{n+1}\frac{\partial}{\partial\xi}\Delta U^* + |\xi_t^{n+1}|\frac{\partial}{\partial\xi}\Delta U^*\right\} \tag{12a}$$

$$\left(\xi_j^{n+1}\frac{\partial}{\partial\xi}A_i^+\right)\Delta U^* = \frac{1}{2}\left\{\xi_j^{n+1}\frac{\partial}{\partial\xi}A_i\,\Delta U^* + |\xi_j^{n+1}|\frac{\partial}{\partial\xi}\rho(A_i)\,\Delta U^*\right\} \tag{12b}$$

and

$$\left(\xi_t^{n+1}\frac{\partial}{\partial\xi}I^-\right)\Delta U^* = \frac{1}{2}\left\{\xi_t^{n+1}\frac{\partial}{\partial\xi}\Delta U^* - |\xi_t^{n+1}|\frac{\partial}{\partial\xi}\Delta U^*\right\} \tag{13a}$$

$$\left(\xi_j^{n+1}\frac{\partial}{\partial\xi}A_i^-\right)\Delta U^* = \frac{1}{2}\left\{\xi_j^{n+1}\frac{\partial}{\partial\xi}A_i\,\Delta U^* - |\xi_j^{n+1}|\frac{\partial}{\partial\xi}\rho(A_i)\,\Delta U^*\right\} \tag{13b}$$

where the j in ξ_j^{n+1} is x, y, and z when the i in A_i, A_i^+, and A_i^- is E, F, or G respectively.

All derivatives in Eq. (12) are replaced by backward-difference formulas that can be either first- or second-order-accurate; i.e.,

$$\frac{\partial}{\partial\xi}f_{i,j,k} \approx \delta_\xi^{b1}f_{i,j,k} = (f_{i,j,k} - f_{i-1,j,k})/\Delta\xi \tag{14a}$$

$$\frac{\partial}{\partial\xi}f_{i,j,k} \approx \delta_\xi^{b2}f_{i,j,k} = (3f_{i,j,k} - 4f_{i-1,j,k} + f_{i-2,j,k})/2\Delta\xi \tag{14b}$$

where δ_ξ^{b1} is first-order; δ_ξ^{b2} is second-order; and i,j,k denote location of the grid point where the derivative is evaluated. All derivatives in Eq. (13) are replaced by forward-difference formulas that also can be either first- or second-order-accurate; i.e.,

$$\frac{\partial}{\partial\xi}f_{i,j,k} \approx \delta_\xi^{f1}f_{i,j,k} = (f_{i+1,j,k} - f_{i,j,k})/\Delta\xi \tag{15a}$$

$$\frac{\partial}{\partial\xi}f_{i,j,k} \approx \delta_\xi^{f2}f_{i,j,k} = (-3f_{i,j,k} + 4f_{i+1,j,k} - f_{i+2,j,k})/2\Delta\xi \tag{15b}$$

where δ_ξ^{f1} is first-order and δ_ξ^{f2} is second-order.

This completes the description of how spatial derivatives on the left hand side (LHS) of Eq. (8a) are replaced by finite-difference formulas. The procedure for replacing spatial derivatives in Eqs. (8b) and (8c) is identical to the one just described for the LHS of Eq. (8a). The procedure for replacing spatial derivatives in the RHS of Eq. (8a) is also identical to that just described if one notes that $U = I\,U$, $E = A_E\,U$, $F = A_F\,U$, and $G = A_G\,U$.

At the conclusion of this step, a set of finite-difference equations (FDEs) are derived which approximate the governing equations given by Eq. (1). These FDEs are applied at every interior grid point. At boundary grid points, boundary conditions are applied. Here, all boundary conditions were implemented implicitly in the manner proposed by Shih et al.[11]

Block Tridiagonal Structure with Second-Order Upwind

When convection terms in the LHS of Eq. (8) are replaced by first-order upwind formulas, namely Eqs. (14a) and (15a) or their equivalent with index j or k shifted instead of i, then the resulting FDE's along with the implicit boundary conditions form systems of linear equations with block tridiagonal coefficient matrices. Here, these systems of equations were solved by the Thomas algorithm.

When convection terms in the LHS of Eq. (8) are replaced by second-order upwind formulas, namely Eqs. (14b) and (15b) or their equivalent, then the coefficient matrices of the resulting systems of linear equations will have a bandwidth that is wider than block tridiagonal. Since a wider bandwidth increases computational cost, the second-order upwind formulas used in the LHS of Eq. (8) were treated partly implicitly and partly explicitly so that the resulting coefficient matrices will be block tridiagonal. This was accomplished by using a procedure described by Vanka[12] but had to be modified before it can be used with Newton-Raphson iteration. That modified procedure is given below in the framework of Eqs. (14b) and (15b):

$$\delta_\xi^{b2}f_{i,j,k}^{m+1} = 2(f_{i,j,k} - f_{i-1,j,k})^{m+1}/\Delta\xi + [(\omega f)_{i-2,j,k} - (\omega f)_{i,j,k}]^m/2\Delta\xi \tag{16a}$$

$$\delta_\xi^{f2}f_{i,j,k}^{m+1} = 2(f_{i+1,j,k} - f_{i,j,k})^m/\Delta\xi + [(\omega f)_{i,j,k} - (\omega f)_{i+2,j,k}]^m/2\Delta\xi \tag{16b}$$

In the above equation, $\omega = 1$ during the first Newton-Raphson iteration of each time step; this is because when $m = 0$, $f^m = f^n$ so that $\Delta U^m = \Delta U^n$. At second and higher Newton-Raphson iterations, $\omega = (2f^m - f^{m-1})/f^m$; this is

because when $m \geq 1$, $\Delta U^{m+1} \ll \Delta U^m$ due to second-order convergence rate of Newton-Raphson iteration. Treating part of the difference operator explicitly as shown in Eq. (16) could affect numerical stability; this was examined numerically and described in the Results section.

Finally, it is noted that special treatment is needed at all grid points next to boundaries when second-order upwind formulas are used. Here, when upwind differencing uses grid points away from the boundary, second-order upwind formulas are still used. However, when upwind differencing uses grid points on the boundary, then first-order upwind formulas are used.

Results

To demonstrate the usefulness and feasibility of the algorithm described above, numerical solutions were obtained on a test problem to investigate 1) the conservation error created by using chain-rule conservation-law form, 2) the robustness of the flux-vector splitting algorithm developed in this study, and 3) the ability of the algorithm to handle complex flow on time-dependent grid systems.

Description of Test Problem

The test problem selected was the flow in one combustion chamber of a Wankel engine (Fig. 1) under nonfiring conditions. Such a flow problem provides a good test of the algorithm because the flowfield in Wankel engine combustion chambers are complex with jets, large separated regions, and very sharp gradients near rotor apices. Also, the combustion chamber deforms considerably in both shape and volume. The description of the Wankel engine studied has been given elsewhere[13-15] and will not be repeated here in detail. Only a brief description of the problem is given to facilitate interpretation of results.

The Wankel engine studied is as follows. It had a width d of 0.07 m, a generating radius R of 0.1064 m, an eccentricity E of 0.01542 m, a clearance of 0.004 m between the rotor apex and the rotor housing, and an intake and exhaust port width d_p of 0.05 m. All solid surfaces of the combustion chamber were impermeable to mass diffusion and maintained at a wall temperature of 300 K.

The combustion chamber considered is the one bounded by surfaces 1 and 2 in Fig. 1. Initially, that combustion chamber was located at the position shown in Fig. 1. At that time, the gas in the combustion chamber was air in a state of homogeneous turbulence with zero mean velocity, a static pressure of 1.5 bars, and a static temperature of 1000 K. At time $t = 0$, the combustion chamber suddenly started to rotate at a crankshaft speed Ω of 10,000 rpm, and the peripheral exhaust port was suddenly opened to initiate the exhaust process. The back pressure of the exhaust port was maintained at 0.85 bar. As the rotor rotated, the peripheral intake port opened to initiate the intake process. The gas that entered into the combustion chamber through the peripheral intake port (i.e., the intake charge) was a nonhomogeneous gaseous octane-air mixture. The stagnation pressure and temperature of the intake charge were 1.2 bars and 300 K, respectively. As the rotor continued to rotate, first the exhaust port closed then the intake port closed. Once the intake port closed, compression process started. Once the rotor passed the position of minimum volume (TDC), the expansion process started. The whole cycle of exhaust, intake, compression, and expansion is then repeated as the rotor continued its rotation.

The spatial domain inside the combustion chamber was represented by a time-dependent grid system shown in Fig. 2. That grid system moved and deformed with the combustion chamber, and consisted of IL grid lines from leading apex to trailing apex, JL grid lines from rotor to rotor housing, and KL grid lines from side housing to one grid line above the

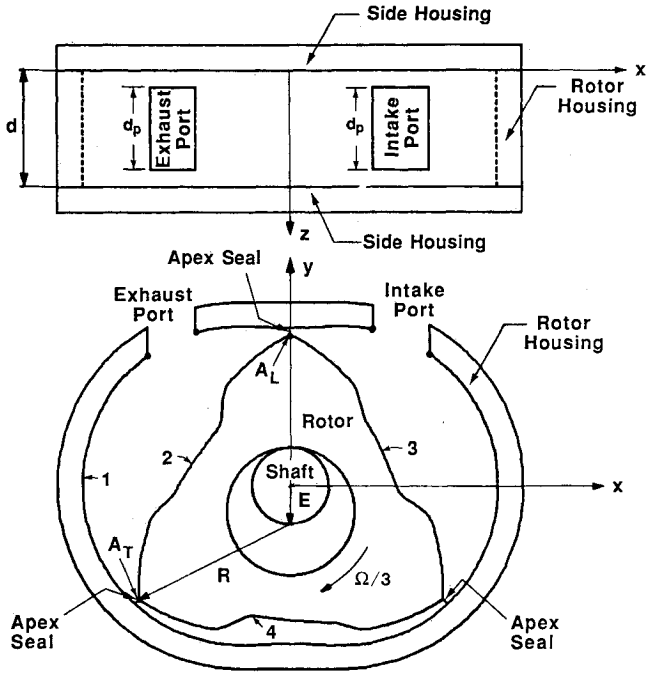

Fig. 1 A schematic diagram of the Wankel engine studied.

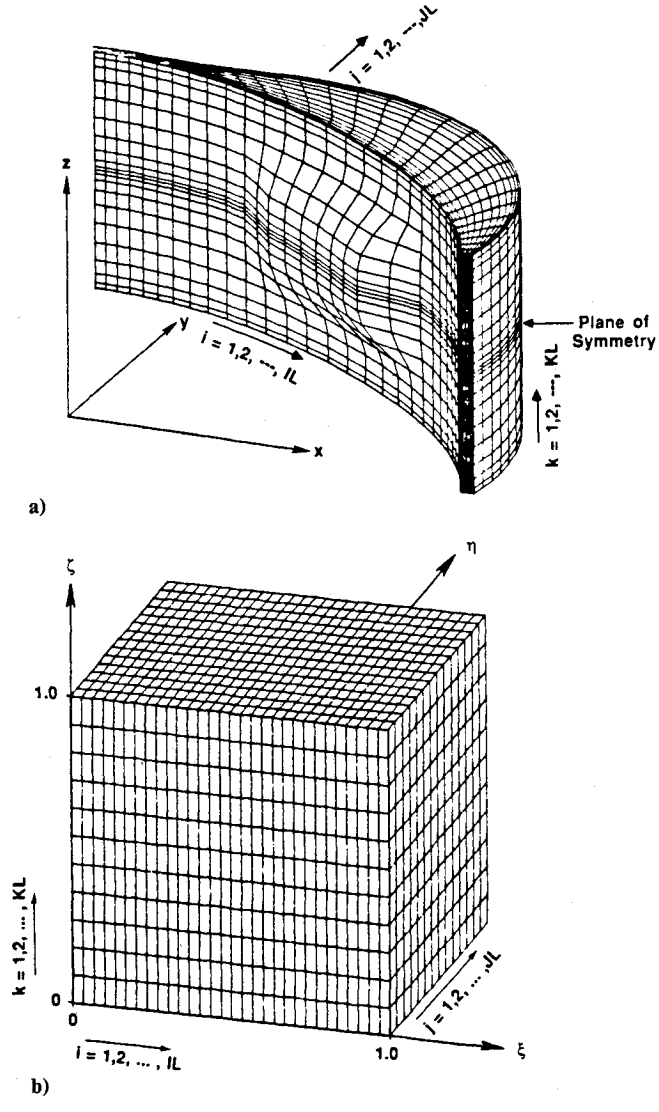

Fig. 2 a) Grid system in the x-y-z-t coordinate system. b) Grid system in the ξ-η-ζ-τ coordinate system.

plane of symmetry. The details of how that grid system was generated can be found in Refs. 15 and 16.

The governing equations employed were the ensemble-averaged conservation equations of mass, momentum (full compressible Navier-Stokes), total energy, and species, closed by a k-ε model of turbulence with wall functions. The details of those equations can be found in Ref. 15, and have the same form as Eq. (1) except for the k-ε model which also has a source term.

Code Development

Two codes based on the algorithm developed in this study were programmed to study the flow inside the Wankel engine just described and to examine the algorithm in terms of conservation error, robustness, and ability to handle complex flows on time-dependent grid systems. One code referred to as Lewis-2D analyzes unsteady, two-dimensional flows in Wankel engines—assuming no variation of the flow in the z direction (Fig. 1). The other code referred to as Lewis-3D analyzes unsteady, three-dimensional flows in Wankel engines.

Conservation Error

The conservation error of the algorithm was studied by using Lewis-2D. This study was conducted by simulating a number of cases in which the intake and exhaust ports were always closed so that the mass in the combustion chamber was a known constant. Since chain-rule conservation-law form does not enforce conservation, this test is a good measure of the accuracy of the algorithm in terms of conservation error. Effects of the following parameters on conservation error were examined: total number of grid points, time-step size, type of differencing used to approximate the convection terms including first-order upwind, second-order upwind as well as central. This study indicated that when central differencing is used, conservation error (defined as total change in mass divided by the correct mass) varied by less than 0.1% over 5000 time steps for all time-step sizes (10^{-8} to 10^{-5} s) and grid spacings ($IL = 21 \times JL = 15$, $IL = 31 \times JL = 21$) examined. When first-order upwind differencing is used, conservation errors varied by as much as 20% for the coarse grid ($IL = 21 \times JL = 15$) but decreased to 7% for a finer grid ($IL = 31 \times JL = 21$). Thus, when first-order upwind dif-

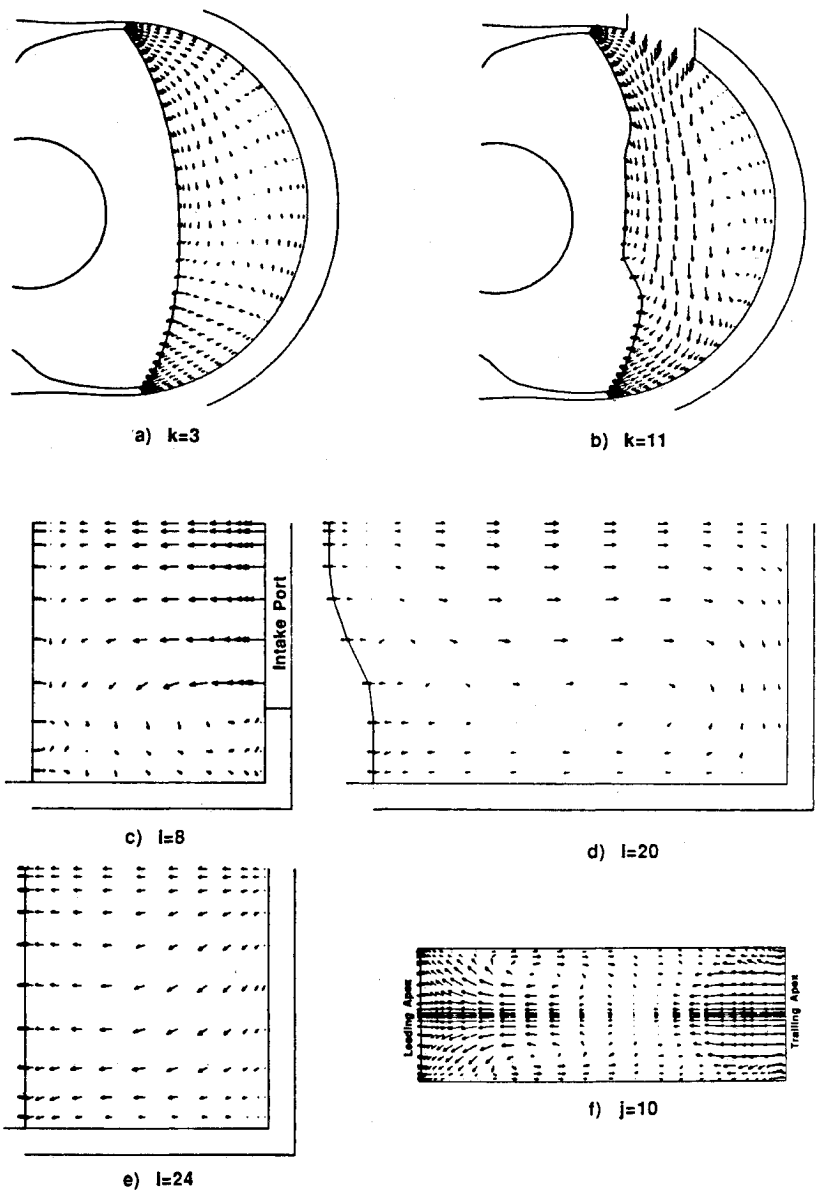

Fig. 3 Flow pattern at crankangle 4.96 radians before TDC: a) $k = 3$ plane; b) $k = 11$ plane; c) $i = 8$ plane; d) $i = 20$ plane; e) $i = 24$ plane; and f) $j = 10$ plane.

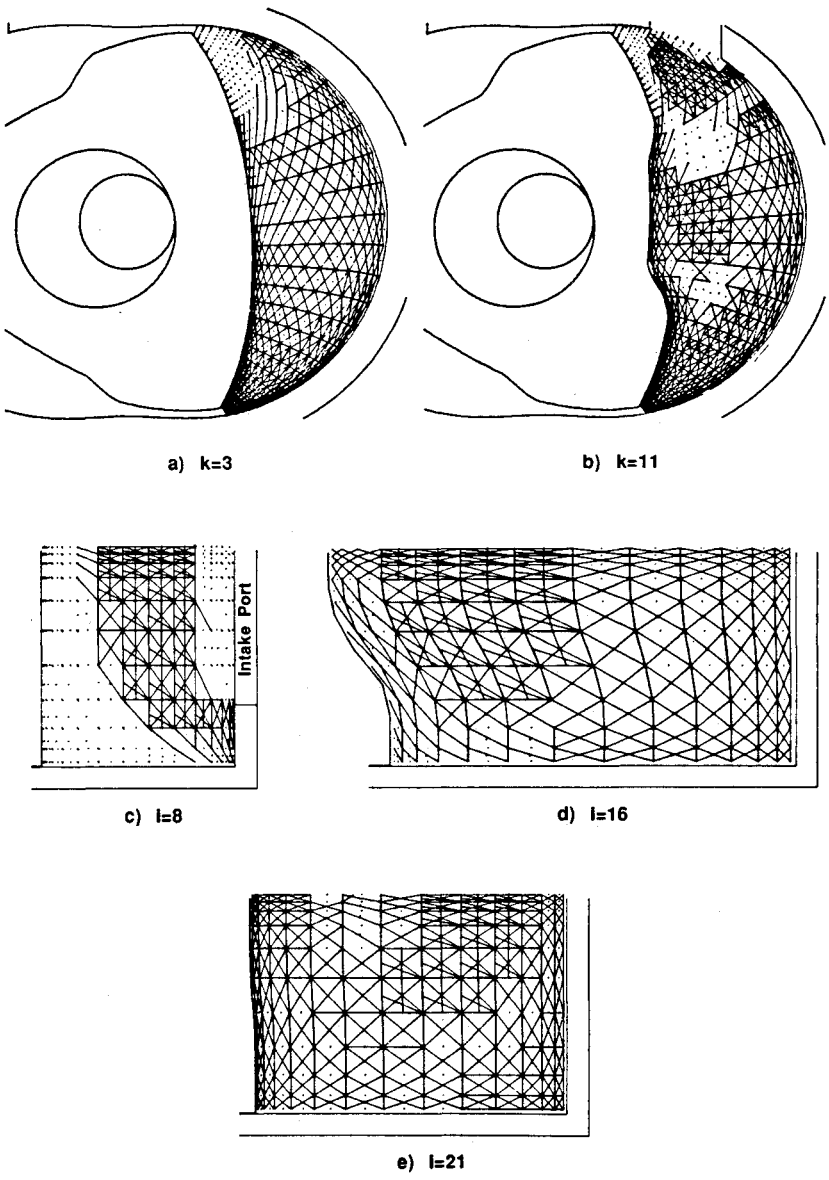

Fig. 4 Mass fraction of fuel at crankangle 4.96 rad before TDC. Darker regions indicate regions with more fuel: a) $k = 3$ plane; b) $k = 11$ plane; c) $i = 8$ plane; d) $i = 20$ plane; e) $i = 24$ plane; and f) $j = 10$ plane.

ferencing is used, the grid system used must be sufficiently fine. When second-order upwind differencing is used, the conservation errors introduced are considerably less than those of first-order upwind: $\leq 2\%$ on the coarse grid and $\leq 0.1\%$ on the fine grid over 5000 time steps.

Here, it is noted that chain-rule conservation-law form can be made identical to weak conservation-law form by appropriate evaluation of metric coefficients. The conservation error of the weak conservation-law form thus obtained was also investigated. It turns out that the results obtained for the weak conservation-law form were comparable to those obtained for the chain-rule conservation-law form just described.

Robustness

The robustness of the flux-vector splitting algorithm with first- and second-order upwind differencing was examined by using both Lewis-2D and Lewis-3D. This study was conducted by using different time-step sizes and grid systems with all ports closed at all times. This study indicated that first-order upwind is much more robust than the second-order upwind. First-order upwind was stable for all of the cases studied, but second-order upwind became unstable when the Courant number exceeded approximately 5. This lack of robustness for the second-order upwind scheme could be the result of the implicit-explicit treatment of the second-order upwind operator on the LHS of Eq. (8). However, once the Newton-Raphson iteration is invoked, the robustness of the second-order upwind scheme was comparable to that of the first-order upwind scheme.

Here, it is noted that the robustness of the first-order upwind is achieved at the expense of accuracy. Reference 14 describes a study that examined the errors in the flux-vector splitting algorithm presented in this study by using Lewis-2D. That study showed second-order upwind to generate much more accurate solutions than first-order upwind for Wankel engine flowfields.

Complex Flows

To demonstrate the usefulness of the algorithm for handling complex flows on time-dependent grid systems, calculations were performed to determine the density, mass fraction of fuel and air, velocity, temperature, pressure, turbulent kinetic energy, and its dissipation rate inside a Wankel engine combustion chamber during exhaust, intake, compression, and expansion by using both Lewis-2D and Lewis-3D. All results generated were given in Refs. 14 and 15. As was discussed

in Refs. 14 and 15, the results generated are consistent with results of previous numerical and experimental studies. Some typical results generated by Lewis-3D for the velocity field and fuel-air mixing are given in Figs. 3 and 4. These solutions were obtained by using $(IL = 31) \times (JL = 15) \times (KL = 12)$ grid points distributed as shown in Fig. 2. These representative results demonstrate that the algorithm presented in this study can be used to study flows in deforming spatial domains with time-dependent grid systems.

Acknowledgment

This research was supported by NASA Grant NAG 3-997. The authors are grateful to NASA Lewis Research Center for this support.

References

[1]Vinokur, M., "Conservation Equations of Gasdynamics in Curvilinear Coordinate Systems," *Journal of Computational Physics*, Vol. 14, No. 2, 1974, pp. 105–125.

[2]Thomas, P. D., and Lombard, C. K., "Geometric Conservation Law and Its Application to Flow Computations on Moving Grids," *AIAA Journal*, Vol. 17, No. 10, 1979, pp. 1030–1037.

[3]Hindman, R. G., "Generalized Coordinate Forms of Governing Fluid Equations and Associated Geometric Induced Errors," *AIAA Journal*, Vol. 20, No. 10, 1982, pp. 1359–1367.

[4]Steger, J. L., "On Application of Body Conforming Curvilinear Grids for Finite-Difference Solution of External Flows," *Numerical Grid Generation*, edited by J. L. Thompson, Elsevier Science Publishing Co., New York, NY, 1982, pp. 295–316.

[5]Vinokur, M., "An Analysis of Finite-Difference and Finite-Volume Formulations of Conservation Laws," *Journal of Computational Physics*, Vol. 81, No. 1, 1989, pp. 1–52.

[6]Beam, R. M., and Warming, R. F., "An Implicit Factored Scheme for the Compressible Navier-Stokes Equations," *AIAA Journal*, Vol. 16, No. 4, 1978, pp. 393–402.

[7]Briley, W. R., and McDonald, H., "Solution of the Multidimensional Compressible Navier-Stokes Equations by a Generalized Implicit Method," *Journal of Computational Physics*, Vol. 24, No. 4, 1977, pp. 372–397.

[8]Jameson, A., and Turkel, E., "Implicit Schemes and LU Decompositions," *Mathematics of Computations*, Vol. 37, 1981, pp. 385–397.

[9]Steger, J. L., and Warming, R. F., "Flux-Vector Splitting of the Inviscid Gasdynamics Equations with Application to Finite-Difference Methods," *Journal of Computational Physics*, Vol. 40, No. 2, 1981, pp. 263–293.

[10]Van Leer, B., "Flux-Vector Splitting for the Euler Equations," *Lecture Notes in Physics*, Vol. 170, 1982, pp. 507–512.

[11]Shih, T. I-P., Smith, G. E., Springer, G. S., and Rimon, Y., "Boundary Conditions for the Solution of the Compressible Navier-Stokes Equations by an Implicit Factored Method," *Journal of Computation Physics*, Vol. 52, No. 1, 1983, pp. 54–79.

[12]Vanka, S. P., "Second-Order Upwind Differencing in a Recirculating Flow," *AIAA Journal*, Vol. 25, No. 11, 1987, pp. 1435–1441.

[13]Steinthorsson, E., Shih, T. I-P., Schock, H. J., and Stegeman, J., "Calculations of the Unsteady, Three-Dimensional Flowfield Inside a Motored Wankel Engine," SAE Paper 880625, 1988; also, *SAE Transactions Journal of Engines*, Sec. 6, Vol. 97, 1988, pp. 6.1160–6.1188.

[14]Li, Z., Steinthorsson, E., Shih, T. I-P., and Nguyen, H. L., "Modelling and Simulation of Wankel Engine Flowfields," SAE Paper 900029, 1990; also, *SAE 1990 Transactions Journal of Engines* (to be published).

[15]Steinthorsson, E., Li, Z., Shih, T. I-P., Nguyen, H. L., and Willis, E. A., "Development of a Wankel Engine Code Based on a Flux-Vector Splitting Algorithm for Deforming Spatial Domains," NASA TM (to be published).

[16]Yang, S.-L., and Shih, T. I-P., "An Algebraic Grid-Generation Technique for Time-Varying Two-Dimensional Spatial Domains," *International Journal for Numerical Methods in Fluids*, Vol. 6, No. 5, 1986, pp. 291–304.