Generating elliptic grids in three dimensions by a method of false transients

E. Ly[1] D. Norrison[2]

(Received 27 July 2007; revised 2 November 2007)

Abstract

A finite difference method based scheme incorporating a method of false transients and an approximate factorisation technique is presented for solution of a system of Poisson's equations used for grid generation. A time step cycling process with repeated endpoints enhances the convergence rate. The scheme required much less computational effort than that required by other numerical schemes. High quality grid systems over an aircraft tailplane are presented. Although, the superiority of the scheme is illustrated for the grid generation problem, it can be employed for other problems requiring the solution of a set of similar elliptic partial differential equations.

Contents

See http://anziamj.austms.org.au/ojs/index.php/ANZIAMJ/article/view/313 for this article, © Austral. Mathematical Soc. 2007. Published November 7, 2007. ISSN 1446-8735

1 Introduction

In order to solve the governing partial differential equations (PDEs) of fluid dynamics numerically, approximations to the partial differentials are introduced. Many of the numerical methods for solving such PDEs required all partial derivatives to be converted into finite difference equations (FDEs), which are solved at discrete points within the domain of interest. Hence, a set of grid points within and on the boundaries of the domain is required to be specified to form a grid system, in a process known as grid generation [3, 7, 8, 9].

We describe one of the most common methods for generating smooth boundary fitted grid systems over a body, namely the elliptic grid generation method, together with a grid stacking technique in three dimensions. In the grid stacking technique, a series of two dimensional O-type grids (say, in the xz-plane) is generated at prescribed locations along the span of the body (in the y-direction), and connected to form a final OH-type three dimensional grid. The system of Poisson's equations, subject to Dirichlet boundary conditions on all boundaries [3, 7, 8], is solved in the computational domain for the physical coordinates of the two dimensional grid points. The source terms of the Poisson's equations provide a capability for clustering of grid lines and points in a specified region of the domain. A finite difference method based scheme, incorporating a method of false transients [4, 5, 6], and an approximate factorisation technique [1, 2, 4, 5, 6, 7], is developed and implemented into a computer code named GRID3DAF. Section 3 generates grids around an aircraft tailplane, and compares the performance of the scheme with the following iterative schemes: Jacobi (GRID3DJAC), point Gauss–Seidel (GRID3DPGS), line Gauss–Seidel (GRID3DLGS), point successive over-relaxation (GRID3DPSOR) and line successive over-relaxation (GRID3DLSOR). In the last three decades, these iterative schemes were commonly employed in the grid generation process; therefore it is more appropriate to compare them (instead of more specialised methods such as the multigrid and conjugate gradient methods) to the GRID3DAF scheme.

2 Grid generation process

As an illustration, we use an aircraft tailplane as a body, with an understanding that the grid generation process can be easily modified to accommodate other bodies such as a turbine blade or a slender body with varying cross-section. The process incorporates a grid stacking technique where a series of two dimensional O-type grids [3, 7, 8, 9] in the xz-plane is generated at prescribed locations along the span of the tailplane (referred to as span stations). These grids are then connected to form a final three dimensional, boundary fitted, OH-type grid over the tailplane.

2.1 Elliptic grid systems

In the xz-plane, the mapping process from the physical coordinates $\boldsymbol{r} = (x, z)$ to the computational coordinates $\boldsymbol{\vartheta} = (\xi, \zeta)$ is described by the relation $\boldsymbol{\vartheta} = \boldsymbol{\vartheta}(\boldsymbol{r})$, which is assumed to have continuous derivatives of all orders. In order to generate an applicable grid, the mapping must be one-to-one to ensure the grid lines of the same family do not cross each other [3], and provides a smooth grid distribution with minimum skewness.

The following system of Poisson's equations [3] is considered:

$$\boldsymbol{\vartheta}_{xx} + \boldsymbol{\vartheta}_{zz} = \mathcal{S}(\boldsymbol{\vartheta})\,, \tag{1}$$

where $\mathcal{S} = (p, q)$ contains the source terms, and $\boldsymbol{\vartheta}_x$ and $\boldsymbol{\vartheta}_{xx}$ denote $\partial\boldsymbol{\vartheta}/\partial x$ and $\partial^2\boldsymbol{\vartheta}/\partial x^2$, respectively. Grid point clustering is enforced by proper selection of the functions $p(\boldsymbol{\vartheta})$ and $q(\boldsymbol{\vartheta})$, and the selection is based on grid point or line attraction in the vicinity of defined grid lines or points, or even a combination of both. Since it is more convenient to solve for $\boldsymbol{r}$, where $\boldsymbol{\vartheta}$ is known in the computational domain, the dependent and independent variables of equation (1) are interchanged to provide

$$\left(\alpha\boldsymbol{r}_{\xi\xi} + P\boldsymbol{r}_{\xi}\right) - 2\beta\boldsymbol{r}_{\xi\zeta} + \left(\gamma\boldsymbol{r}_{\zeta\zeta} + Q\boldsymbol{r}_{\zeta}\right) = \mathbf{0}\,, \tag{2}$$

where $\mathbf{0}$ is a zero vector. The constants in equation (2) are

$$\alpha = \boldsymbol{r}_{\zeta}\cdot\boldsymbol{r}_{\zeta}\,, \quad \beta = \boldsymbol{r}_{\xi}\cdot\boldsymbol{r}_{\zeta}\,, \quad \gamma = \boldsymbol{r}_{\xi}\cdot\boldsymbol{r}_{\xi}\,, \tag{3}$$

where $\cdot$ represents the dot product of two vectors. The source terms are

$$P = \frac{p}{J^2}\,, \quad Q = \frac{q}{J^2}\,, \quad J = \frac{1}{x_{\xi}z_{\zeta} - x_{\zeta}z_{\xi}}\,, \tag{4}$$

where J is the Jacobian of transformation. The solution to equation (2) is periodic in ξ-direction due to the existence of a re-entrant boundary that forms the left and right boundaries of the computational domain. Since the system is quasi-linear, linearisation must be used in the numerical solution. For simplicity, a lagging of the coefficients (3) is employed, with the coefficients evaluated at the previous iteration [3, 9].

An artificial time dependent term, $\boldsymbol{r}_{\tau}$ (where τ is the artificial time scale), is appended to (2) to incorporate the temporal numerical dissipation:

$$\boldsymbol{r}_{\tau} = \left(\alpha\boldsymbol{r}_{\xi\xi} + P\boldsymbol{r}_{\xi}\right) - 2\beta\boldsymbol{r}_{\xi\zeta} + \left(\gamma\boldsymbol{r}_{\zeta\zeta} + Q\boldsymbol{r}_{\zeta}\right). \tag{5}$$

Table 1: Well-known time difference rules for equation (6).

Time difference rule	a	b
Trapezoidal rule	0	$\frac{1}{2}$
Euler implicit	0	1
Three-point backward	$\frac{1}{2}$	1
Euler explicit	0	0
Leap frog	$-\frac{1}{2}$	0

Since the boundary conditions are time independent for a static grid system, and provided that the numerical solution converges [4, 5, 6, 7], we anticipate that $\boldsymbol{r}_\tau \to 0$ as $\tau \to \infty$.

Let τ be discretised as $\tau \equiv \tau_n = n\Delta\tau$, where $\Delta\tau$ is a discrete increment of τ and n the iteration or time level, and $\boldsymbol{r}(\tau_n) = \boldsymbol{r}(n\Delta\tau) = \boldsymbol{r}^n$. Here the spatial dependence has been temporarily suppressed. The time derivative is approximated by a general time difference rule of a form suggested by Warming and Beam [10], which includes the rules presented in Table 1,

$$\left(\frac{\partial \boldsymbol{r}}{\partial \tau}\right)^n = \frac{(1+a)\overrightarrow{\Delta}_\tau - a\overleftarrow{\Delta}_\tau}{\Delta\tau\,(1+b\overrightarrow{\Delta}_\tau)}\boldsymbol{r}^n + \left(b-a-\tfrac{1}{2}\right)\mathcal{O}(\Delta\tau) + \mathcal{O}(\Delta\tau)^2. \qquad (6)$$

In the above expression, a and b are constants that define the time difference rule, and $\overrightarrow{\Delta}_\tau$ and $\overleftarrow{\Delta}_\tau$ are the forward and backward time difference operators. Inserting approximation (6) into equation (5) for $\boldsymbol{r}_\tau$ at time level τ_n, and approximately factorising the left side of the resulting expression by neglecting all mixed derivatives, and omitting third and higher order terms in $\Delta\tau$, yields

$$\left[1 - \Delta\tilde{\tau}\left(\alpha^n\frac{\partial^2}{\partial\xi^2} + P^n\frac{\partial}{\partial\xi}\right)\right]\left[1 - \Delta\tilde{\tau}\left(\gamma^n\frac{\partial^2}{\partial\zeta^2} + Q^n\frac{\partial}{\partial\zeta}\right)\right]\overrightarrow{\Delta}_\tau\boldsymbol{r}^n$$
$$= \tilde{a}\overleftarrow{\Delta}_\tau\boldsymbol{r}^n + \Delta\tilde{\tau}\frac{\omega}{b}\mathcal{R}^n\,. \qquad (7)$$

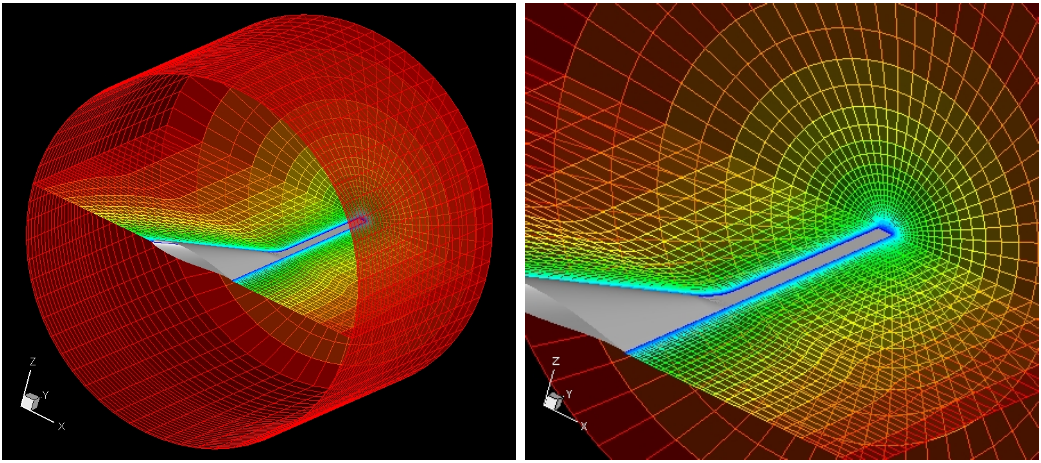

Figure 1: OH-grid generated over a tailplane.

In equation (7), ω is a relaxation factor, $\Delta\tilde{\tau} = \frac{b\,\Delta\tau}{1+a}$, $\tilde{a} = \frac{a}{1+a}$ with $a \neq -1$, and $\mathcal{R}^n$ denotes the residual (which measures how well the FDEs are satisfied by the approximate solution). The factored equation (7) is then solved in an alternating direction manner:

$$\left[1 - \Delta\tilde{\tau}\left(\alpha^n \frac{\partial^2}{\partial\xi^2} + P^n \frac{\partial}{\partial\xi}\right)\right]\Delta\boldsymbol{r}^* = \tilde{a}\overleftarrow{\Delta}_\tau \boldsymbol{r}^n + \Delta\tilde{\tau}\frac{\omega}{b}\mathcal{R}^n, \tag{8}$$

$$\left[1 - \Delta\tilde{\tau}\left(\gamma^n \frac{\partial^2}{\partial\zeta^2} + Q^n \frac{\partial}{\partial\zeta}\right)\right]\overrightarrow{\Delta}_\tau \boldsymbol{r}^n = \Delta\boldsymbol{r}^*, \tag{9}$$

$$\boldsymbol{r}^{n+1} = \boldsymbol{r}^n + \overrightarrow{\Delta}_\tau \boldsymbol{r}^n. \tag{10}$$

Ly and Norrison [7] give a detailed description on the solution and time step cycling processes, and of the finite difference discretisations used in equations (3), (8), (9) and (10).

2.2 Grid stacking technique

The developed computer code, GRID3DAF, allows the user to allocate span stations on the tailplane and in the region between the tailplane and far-field spanwise boundary in the following manners: (1) uniform distribution, (2) cluster towards the tailplane tip in an exponential manner, and (3) at discrete locations designated by the user.

In the spanwise direction, the mapping is governed by $\eta = \eta(y)$, where η and y are the computational and physical coordinates, respectively. At each span station, the location of leading and trailing edges, and the sectional profile of the body, which are required for the generation of O-type grids, are computed from the functions describing the body shape. However, at the tip section and further spanwise, it is necessary to extend the leading and trailing edge functions, $x_{\rm le}(y)$ and $x_{\rm te}(y)$, to ensure that ξ remains twice differentiable with respect to both x and y, and $\xi_y = 0$ at the spanwise boundary [2]. This guarantees that the grid generated in this region is smooth, see Figure 1 (for visual clarity only relevant portions of the grid are shown). To ensure ξ_{yy} is continuous at the tip section, it is necessary to have $x_{\rm le}$, $x_{\rm te}$, $dx_{\rm le}/dy$, $dx_{\rm te}/dy$, $d^2x_{\rm le}/dy^2$ and $d^2x_{\rm te}/dy^2$ continuous. For $\xi_y = 0$ to hold at the spanwise boundary, we set $dx_{\rm le}/dy$, $dx_{\rm te}/dy$, $d^2x_{\rm le}/dy^2$ and $d^2x_{\rm te}/dy^2$ to zero at some point y_c between the tip section and spanwise boundary. This point is required as an input parameter in the code, with the value varying from 0 (at the tip section) to 1 (spanwise boundary). Usually, it is set to 0.5. The leading and trailing edge functions are then continued to the boundary with zero slope with respect to y, as shown in Figure 1. Gear [2] suggested that the extended leading edge function should take the following form:

$$x = \begin{cases} x_{\rm le}(\hat{y}) & \text{for} \quad \hat{y} \in [-1, 0]\,, \\ x_{\rm le}(0) + \hat{y}\left[1 - \left(\hat{y}/\hat{y}_c\right)^2 + \frac{1}{2}\left(\hat{y}/\hat{y}_c\right)^3\right] \frac{dx_{\rm le}}{d\hat{y}}\Big|_{\hat{y}=0} & \text{for} \quad \hat{y} \in [0, \hat{y}_c]\,, \\ x_{\rm le}(0) + \frac{1}{2}\hat{y}\frac{dx_{\rm le}}{d\hat{y}}\Big|_{\hat{y}=0} & \text{for} \quad \hat{y} \in [\hat{y}_c, \hat{y}_m]\,, \end{cases} \tag{11}$$

where $\hat{y} = y/y_b - 1$, $\hat{y}_c = y_c/y_b - 1$ and $\hat{y}_m = y_m/y_b - 1$. In the above expressions, y_m is the maximum spanwise value of y, the outermost grid line

Table 2: Comparison of numerical schemes.

Codes	ω	$\lvert\text{Error}\rvert$	$\log_{10}\lvert\text{Error}\rvert$	$\lvert\mathcal{R}\rvert$	$\log_{10}\lvert\mathcal{R}\rvert$
		(10^{-4})		(10^{-6})	
GRID3DJAC		1.067	-3.972	69.415	-4.159
GRID3DPGS		1.430	-3.845	56.822	-4.246
GRID3DLGS		1.720	-3.765	54.278	-4.265
GRID3DPSOR	1.665	1.984	-3.703	27.499	-4.561
GRID3DLSOR	1.580	1.496	-3.825	59.770	-4.224
GRID3DAF	1.975	0.252	-4.599	2.314	-5.635

on the tailplane has y value given by y_b, which is less than or equal to the semispan length of the tailplane. A similar trailing edge function can be derived by replacing all x_{le} in (11) with x_{te} . If the tailplane is tapered, y_c is adjusted to ensure that the extended edge functions do not intersect.

3 Results and discussion

The AFGRID3D scheme has been used with a second order accurate, three point, backward time difference rule and a sequence of eight time steps, cycled from $\Delta\tau = 10^2$ to $\Delta\tau = 10^5$, to generate an OH-type grid around a tailplane. In general, the maximum time step will typically be up to hundreds times larger than the minimum. When the time step is at its largest the linear systems will not be, or are only slightly diagonally dominant, and information rapidly distributes throughout the grid system. This rapid distribution is the mechanism that enhances the convergence rate. Immediately after reaching its maximum value the time step is reset to its minimum value. This ensures that any instabilities, that may have occurred while the time step was large, rapidly decay [2]. Numerical experiments indicate that the scheme becomes unstable if the Euler explicit and leap frog time difference rules are used.

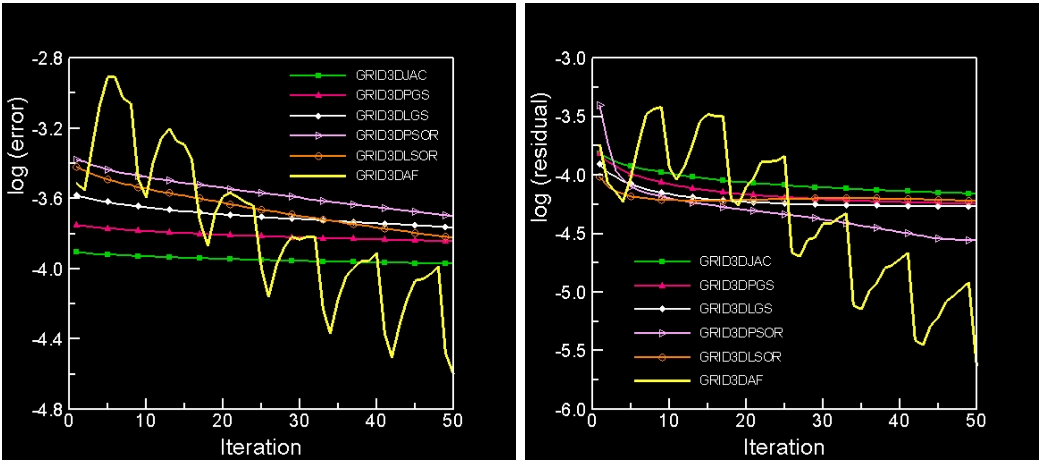

Figure 2: Comparison of convergence histories.

The tailplane has a NACA 63_2-615 section with maximum thickness of 0.15ℓ located at 0.341ℓ from the leading edge, and has a semispan length of 0.72ℓ, where ℓ is the chord length measured at the root section. The grid, which is in a form of a six sided, rectangular shape in the computational domain and in a circular cylinder shape in the physical domain, has a radius and length of 2ℓ. In the chordwise and radial direction, 61 and 31 grid points are respectively allocated. While in the spanwise direction, 15 points are uniformly distributed on the tailplane and 15 points in the region beyond the tailplane tip (clustered towards the tip). The source terms, p and q, were computed according to equation (14) of Ly and Norrison [7], so that grid lines are clustered toward the tailplane.

Optimal relaxation values were obtained by numerical experiments and were used in the GRID3DPSOR, GRID3DLSOR and GRID3DAF scheme, see Table 2. Initially, the optimal ω used in GRID3DLSOR scheme was very close to one [7], and it was this reason that it took much longer than the GRID3DPSOR scheme to produce a converged solution. It was later found that the direction in which the GRID3DLSOR scheme swept through the computational domain

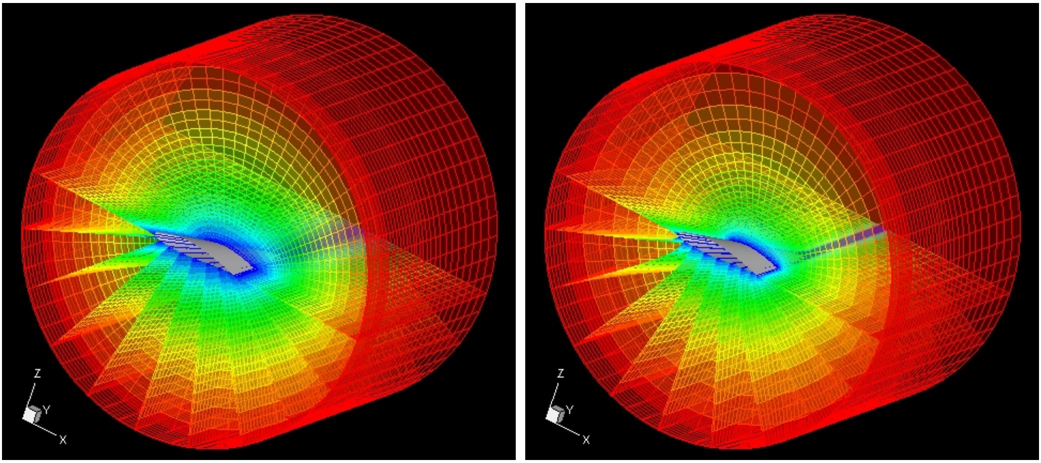

Figure 3: Initial (left) and final (right) OH-type grids over a tailplane.

was not in the direction of the greatest changes in the solution. This issue was rectified, and an optimal ω was found to be about 1.58. All schemes were executed for 50 iterations. The errors which represent the total changes in the solution per interior grid point, and the absolute maximum residual values are compared in Table 2. The convergence histories are displayed in Figure 2, and the plots show that the error and residual of the GRID3DAF scheme reduce substantially whenever the scheme completed one cycle of time step cycling process. Table 2 indicates that after 50 iterations, the error of the GRID3DAF scheme is about 4.2 to 7.9 times smaller than all other schemes, and the residual is about 11.9 to 30 times smaller. In fact, GRID3DAF converged within 50 iterations if a maximum residual smaller than 10^{-5} is the required tolerance.

We observed that the speed at which the scheme converges mainly depended on the manner in which the shape of the branch-cuts are changing from one iteration to the next. In the GRID3DAF scheme, the grid points along the branch-cut are included implicitly into the equation system, and so the branch-cut points are updated with all other interior points simultane-

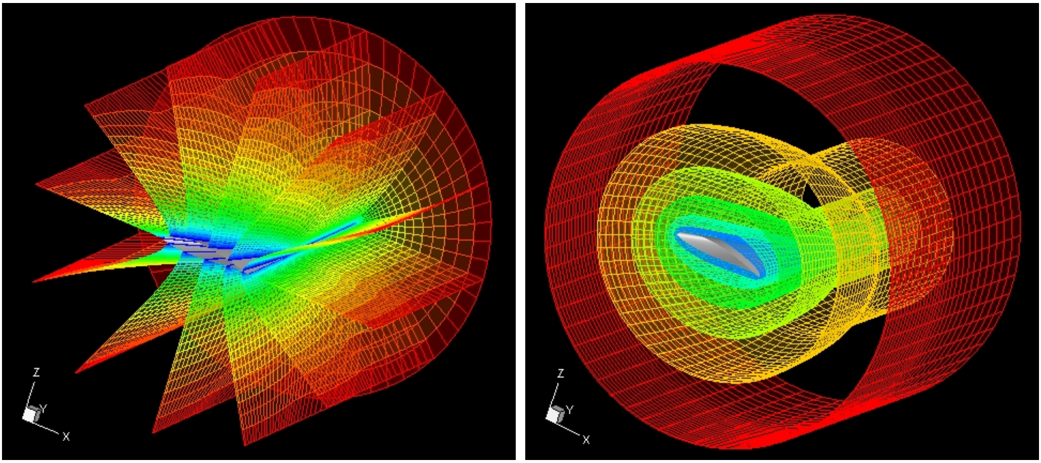

Figure 4: Different views of the final OH-type grid over a tailplane.

ously. In comparison, the branch-cut points of all other schemes are updated with one iteration lapsed. The initial and final grid grids over a tailplane, generated by the GRID3DAF scheme are shown in Figures 3 and 4, where only relevant planes of grid points are displayed to improve the visual clarity of these plots. Careful examination of these plots reveal a smooth distribution of the grid points within the domain (note the adjustment of the grid points on the branch-cut), and grid points are clustered toward and conformed to the inner boundary (tailplane). The natural (due to the elliptic nature of the governing PDEs) and enforced clustering of grid points toward the body can also be seen. For instance, a group of blue lines in the initial grid in Figure 3 contracts towards the tailplane as shown in the final grid (Figure 3) and in Figure 4 as well.

4 Conclusion

A system of Poisson's equations and a grid stacking technique are employed to generate a structured OH-type grid around a single body of arbitrary shape in three dimensions. The scheme incorporates a method of false transients and an approximate factorisation technique where a time step cycling process is used. It was shown that the scheme is significantly faster in reaching convergence to the user's required accuracy than all other numerical schemes compared in this article. Furthermore, if the scheme is numerically stable and solution converged, a correct final grid system can always be obtained independent of the form of its initial grid system. Hence, the method is suitable for automatic grid generation computer code. The grid generation process could be extended to the generation of dynamic grid systems for a body in motion.

References

[1] Catherall, D., Optimum Approximate-Factorization Schemes for Two-Dimensional Steady Potential Flows, *AIAA Journal*, **20**, 8, 1982, pp. 1057–1063. C172

[2] Gear, J. A., Time Marching Approximate Factorization Algorithm for the Modified Transonic Small Disturbance Equation, Research Report Number 6, Department of Mathematics, RMIT University, Melbourne, Australia, Feb. 1996, 11 pages. C172, C176, C177

[3] Hoffmann, K. A., *Computational Fluid Dynamics for Engineers*, Engineering Educational System, Texas, USA, 1989. C171, C172, C173

[4] Ly, E., Improved Approximate Factorisation Algorithm for the Steady Subsonic and Transonic Flow over an Aircraft Wing, in *Proceedings of*

the 21st Congress of the International Council of the Aeronautical Sciences (ICAS98), AIAA and ICAS, Melbourne, Australia, Sep. 1998, Paper A98-31699. C172, C174

[5] Ly, E., and Gear, J. A., Time-Linearized Transonic Computations Including Shock Wave Motion Effects, *Journal of Aircraft*, **39**, 6, Nov-Dec 2002, pp. 964–972. C172, C174

[6] Ly, E., and Nakamichi, J., Time-Linearised Transonic Computations Including Entropy, Vorticity and Shock Wave Motion Effects, *The Aeronautical Journal*, Nov. 2003, pp. 687–695. C172, C174

[7] Ly, E., and Norrison, D., Automatic Elliptic Grid Generation by an Approximate Factorisation Algorithm, *ANZIAM Journal*, **48** (CTAC2006), pp. C188–C202, July 2007. C171, C172, C174, C175, C178

[8] Mathur, J. S., and Chakrabartty, S. K., An Approximate Factorization Scheme for Elliptic Grid Generation with Control Functions, *Numerical Methods for Partial Differential Equations*, **10**, 6, 1994, pp. 703–713. C171, C172

[9] Thompson, J. F., Thames, F. C., and Mastin, C. W., *Boundary-Fitted Curvilinear Coordinate Systems for Solution of Partial Differential Equations on Fields Containing any Number of Arbitrary Two-Dimensional Bodies*, NASA Contractor Report CR-2729, Washington DC, USA, July 1977, 253 pages. C171, C172, C173

[10] Warming, R. F., and Beam, R. M., On the Construction and Application of Implicit Factored Schemes for Conservation Laws, in *SIAM-AMS Proceedings*, **11**, USA, 1978, pp. 85–129. C174

Author addresses

1. **E. Ly**, School of Mathematical and Geospatial Sciences, Science, Engineering and Technology Portfolio, RMIT University, Melbourne, Australia.
   `mailto:eddie.ly@rmit.edu.au`

2. **D. Norrison**, School of Mathematical and Geospatial Sciences, Science, Engineering and Technology Portfolio, RMIT University, Melbourne, Australia.