Application of method of false transients to generate smooth grids around a body in motion

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

(Received 12 August 2008; revised 27 November 2008)

Abstract

A time marching finite difference scheme incorporating an efficient method of false transients, an approximate factorisation technique and a time steps cycling process, is presented for solution of a system of Poisson's equations. The solution to the equations provides a smooth three dimensional boundary fitted grid around a body in motion. The scheme required much less computational effort than that required by other iterative schemes. In closure, examples of a static grid around an aircraft tailplane and a dynamic grid around a flapping wing are presented.

http://anziamj.austms.org.au/ojs/index.php/ANZIAMJ/article/view/1425 gives this article, © Austral. Mathematical Soc. 2008. Published December 5, 2008. ISSN 1446-8735. (Print two pages per sheet of paper.)

Contents

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 require 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 [4, 8, 11, 12].

We employed the elliptic grid generation method, together with a grid stacking technique as described by the authors [10], for generating smooth boundary fitted static grids around a three dimensional body. However, in unsteady aerodynamic and aeroelasticity computations, the body moves with respect to time in a particular mode (or even a combination of different modes) of motion such as pitching, heaving or flapping. This in turn cause the grid points, especially those points near the body, to move in a similar nature, resulting in a need to regenerate a new grid around the body. Consequently, as the body moves, a system of Poisson's equations, subject to a new set of Dirichlet boundary conditions on all boundaries at each time level, is solved to generate the new grid. Here we present the time marching version of the scheme developed by Ly and Norrison [10] for generating smooth and accurate dynamic grids around a moving body. In contrast to the past work of other researchers, for a body that moves in a harmonic motion, normally the new grids are generated only at a number of time levels of the motion cycle, and an interpolation technique is employed to generate (approximate) grids for other intermediate time levels. This helps to reduce the computational time spent by the solver for the grid generation process. But, the grids are less accurate in a sense that the coordinate of many grid points close to the body are approximated rather than recalculated by solving the grid equations. With the efficient numerical scheme presented here, it is now possible to generate an accurate grid at each time level without much increases in computational effort required from the solver.

The present scheme incorporates a method of false transients [5, 6, 7], an approximate factorisation technique [2, 5, 11] and a variable time steps cycling process [5]. These techniques are implemented into a FORTRAN90 computer code named MVGRID3D. Examples of generated grids around an aircraft tailplane and a flapping rectangular wing are presented in Section 3.

2 Grid generation process

2.1 System of Poisson's equations

In the xz-plane, the mapping process from the physical coordinates $\mathbf{r} = (x, z)$ to the computational coordinates $\boldsymbol{\vartheta} = (\xi, \zeta)$ is related by $\boldsymbol{\vartheta} = \boldsymbol{\vartheta}(\mathbf{r})$, assuming that the derivatives of all orders are continuous. The mapping is one-to-one to ensure the grid lines of the same family do not cross each other [4], and

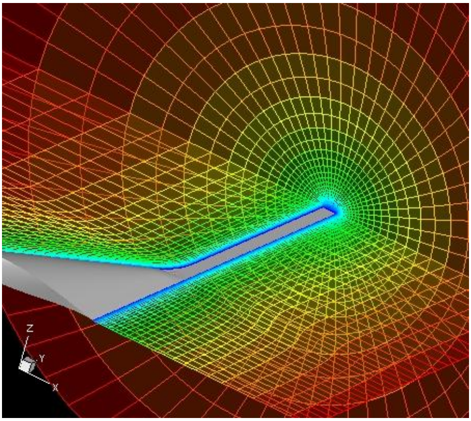

Figure 1: Grid generated around an aircraft tailplane.

that the grid distribution is smooth with minimum skewness. The following system of Poisson's equations is considered

$$\frac{\partial^2 \boldsymbol{\vartheta}}{\partial x^2} + \frac{\partial^2 \boldsymbol{\vartheta}}{\partial z^2} = \mathcal{S}\,, \tag{1}$$

where $\mathcal{S} = (p, q)$ contains the source terms. The source terms are used to enforce grid points clustering in a specific region of the domain. The dependent and independent variables of Equation (1) are interchanged to provide (mathematical derivations are described by Ly and Norrison [8])

$$\frac{\partial \mathbf{r}}{\partial \zeta} \cdot \frac{\partial \mathbf{r}}{\partial \zeta} \frac{\partial^2 \mathbf{r}}{\partial \xi^2} - 2\frac{\partial \mathbf{r}}{\partial \xi} \cdot \frac{\partial \mathbf{r}}{\partial \zeta} \frac{\partial^2 \mathbf{r}}{\partial \xi \,\partial \zeta} + \frac{\partial \mathbf{r}}{\partial \xi} \cdot \frac{\partial \mathbf{r}}{\partial \xi} \frac{\partial^2 \mathbf{r}}{\partial \zeta^2} = -\frac{p}{J^2}\frac{\partial \mathbf{r}}{\partial \xi} - \frac{q}{J^2}\frac{\partial \mathbf{r}}{\partial \zeta}\,, \tag{2}$$

where $\cdot$ denotes the dot product of two vectors, and J $(J \neq 0)$ is the Jacobian of transformation,

$$\frac{1}{J} = \frac{\partial x}{\partial \xi}\frac{\partial z}{\partial \zeta} - \frac{\partial x}{\partial \zeta}\frac{\partial z}{\partial \xi}\,. \tag{3}$$

2.2 Numerical solution procedure

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

$$\frac{\partial \mathbf{r}}{\partial \tau} = \left(\frac{\partial \mathbf{r}}{\partial \zeta} \cdot \frac{\partial \mathbf{r}}{\partial \zeta} \frac{\partial^2 \mathbf{r}}{\partial \xi^2} + \frac{p}{J^2} \frac{\partial \mathbf{r}}{\partial \xi}\right) - 2 \frac{\partial \mathbf{r}}{\partial \xi} \cdot \frac{\partial \mathbf{r}}{\partial \zeta} \frac{\partial^2 \mathbf{r}}{\partial \xi\, \partial \zeta} + \left(\frac{\partial \mathbf{r}}{\partial \xi} \cdot \frac{\partial \mathbf{r}}{\partial \xi} \frac{\partial^2 \mathbf{r}}{\partial \zeta^2} + \frac{q}{J^2} \frac{\partial \mathbf{r}}{\partial \zeta}\right). \quad (4)$$

Since the boundary conditions are locally time independent at each time level, and provided that the numerical solution converges [5, 6], we anticipate that $\mathbf{r}_\tau \to 0$ as $\tau \to \infty$. The $\mathbf{r}_\tau$ term is approximated by a general time difference rule in a padé form [5, 13]. Note that right side of Equation (4) represents the residual $\mathcal{R}$ which measures how well the FDEs are satisfied by the approximate solution.

Equation (4) is approximately factorised by neglecting all mixed derivatives, and third and higher order terms in $\Delta\tau$, and solved in the following alternating direction manner

$$\left[1 - \Delta\tilde{\tau}\left(\frac{\partial \mathbf{r}}{\partial \zeta} \cdot \frac{\partial \mathbf{r}}{\partial \zeta} \frac{\partial^2}{\partial \xi^2} + \frac{p}{J^2} \frac{\partial}{\partial \xi}\right)^n\right] \Delta \mathbf{r}^* = \tilde{a} \overleftarrow{\Delta}_\tau \mathbf{r}^n + \Delta\tilde{\tau} \frac{\omega}{b} \mathcal{R}^n, \quad (5)$$

$$\left[1 - \Delta\tilde{\tau}\left(\frac{\partial \mathbf{r}}{\partial \xi} \cdot \frac{\partial \mathbf{r}}{\partial \xi} \frac{\partial^2}{\partial \zeta^2} + \frac{q}{J^2} \frac{\partial}{\partial \zeta}\right)^n\right] \overrightarrow{\Delta}_\tau \mathbf{r}^n = \Delta \mathbf{r}^*, \quad (6)$$

$$\mathbf{r}^{n+1} = \mathbf{r}^n + \overrightarrow{\Delta}_\tau \mathbf{r}^n. \quad (7)$$

In Equations (5), (6) and (7), $\overleftarrow{\Delta}_\tau$ and $\overrightarrow{\Delta}_\tau$ represent the backward and forward time difference operators,

$$\Delta\tilde{\tau} = \left(\frac{b}{1+a}\right)\Delta\tau, \qquad \tilde{a} = \frac{a}{1+a} \quad (\text{with } a \neq -1), \quad (8)$$

and ω is a relaxation factor (scheme is over relaxed when $\omega > 1$ and under relaxed when $\omega < 1$). The time difference rule used in the approximation is defined through the constants a and b. For example, when $(a, b) =$

$(0, 1)$, Euler implicit rule is used to discretise the time derivative. Numerical experiments indicate that the scheme becomes unstable if the Euler explicit and leap frog time difference rules are employed.

In each iteration, a new approximation to the solution is determined by systemically solving Equation (5) for the dummy temporal differences $\Delta\mathbf{r}^*$, Equation (6) for the unknown vector $\overrightarrow{\Delta}_\tau \mathbf{r}^n$, and applying Equation (7) to update the solution vector $\mathbf{r}^{n+1}$. This solution method is potentially fast, since the method is fully vectorised, and variable time stepping is incorporated. Note that for an accurate factorisation, and to ensure that each linear equation system is strongly diagonally dominant, the time steps must be small relative to the spatial grid spacings. Ly and Gear [6] observed that large errors occur at the extreme ends of the frequency range, and suggested that this unfavourable behaviour be eliminated by cycling the time steps in a geometric fashion, but with repeated endpoints ($\Delta\tilde{\tau}_1 = \Delta\tilde{\tau}_2$ and $\Delta\tilde{\tau}_{M-1} = \Delta\tilde{\tau}_M$, where M is the number of time steps per cycle). Detail descriptions on the solution procedure, time steps cycling process and finite difference discretisations have been described by Ly and Norrison [8, 9].

In the spanwise direction, the mapping is governed by $\eta = \eta(y)$, where η and y are the computational and physical coordinates, respectively. The present code allows the user to allocate span stations along the body and in the region between the body and farfield spanwise boundary in the following manners:

1. uniform distribution,
2. cluster towards the tip section of the body in an exponential manner, and
3. at discrete locations designated by the user.

At each span station, the location of body leading and trailing edges and sectional body profile, which are required for the generation of the grids,

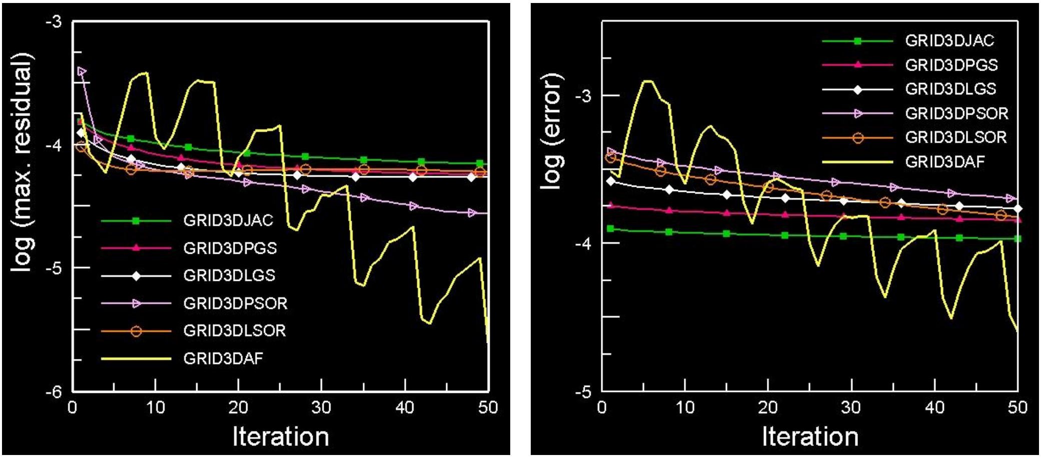

Figure 2: Comparison of convergence histories.

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 to ensure that ξ remains twice differentiable with respect to both x and y, and $\xi_y = 0$ at the spanwise boundary. This ensures that the grid generated in this region is smooth, as reported by Gear [3] and Ly and Norrison [10]. Figure 1 shows an example of a grid generated around an aircraft tailplane, where leading and trailing edge functions are extended to ensure that the grid near the tip section is smooth.

3 Flapping wing

For the computational examples presented here, the mvgrid3d code employed an Euler implicit time difference rule to discretise the time derivative term, and a sequence of eight time steps per cycle, cycled from $\Delta\tau = 5\times10^{-3}$ to 5, to generate an oh-type grid around the body. In general, the maximum time step is typically be up to hundreds times larger than the mini-

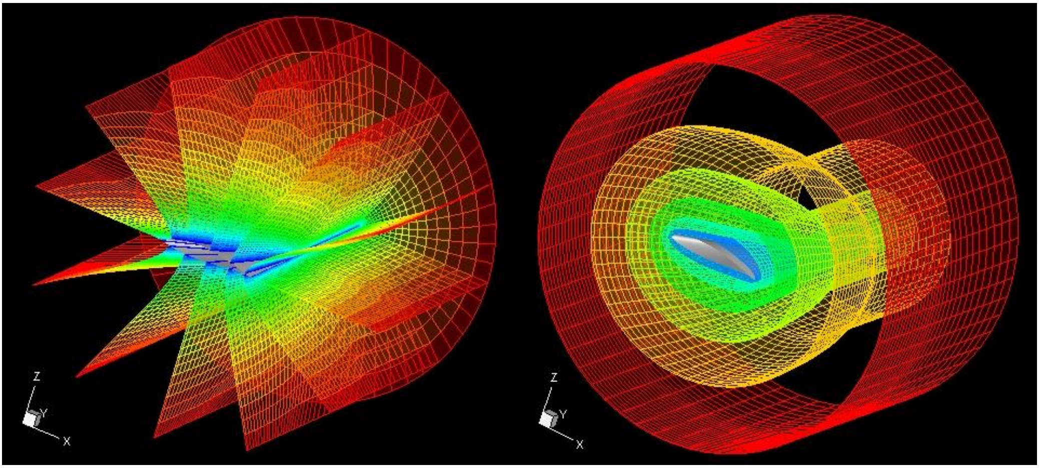

Figure 3: Different views of the grid around an aircraft tailplane.

mum. When the time step is at its largest value the linear equation systems are slightly (or not even) diagonally dominant, and information rapidly distributes throughout the grid system [3]. This rapid distribution is the mechanism that enhances the convergence rate, which is shown by the oscillating behaviour of the residual and error curves in Figure 2. Immediately after reaching its maximum value the time step is reset to its minimum value. This ensures that any instabilities that have occurred while the time step was large are rapidly decay.

Figure 2 compares the convergence histories between the present scheme (GRID3DAF, static grid version) and five other iterative schemes: Jacobi iterative scheme (GRID3DJAC), point/line Gauss–Seidel scheme (GRID3DPGS/GRID3DLGS) and point/line successive over-relaxation scheme (GRID3DPSOR/GRID3DLSOR) for a fixed number of iterations for the tailplane. These iterative schemes were commonly employed in the grid generation process in the last three decades; therefore it is more appropriate to compare them, instead of more specialised methods such as the multigrid and conjugate gradient methods, to the present scheme. In Figure 2, the maximum residual

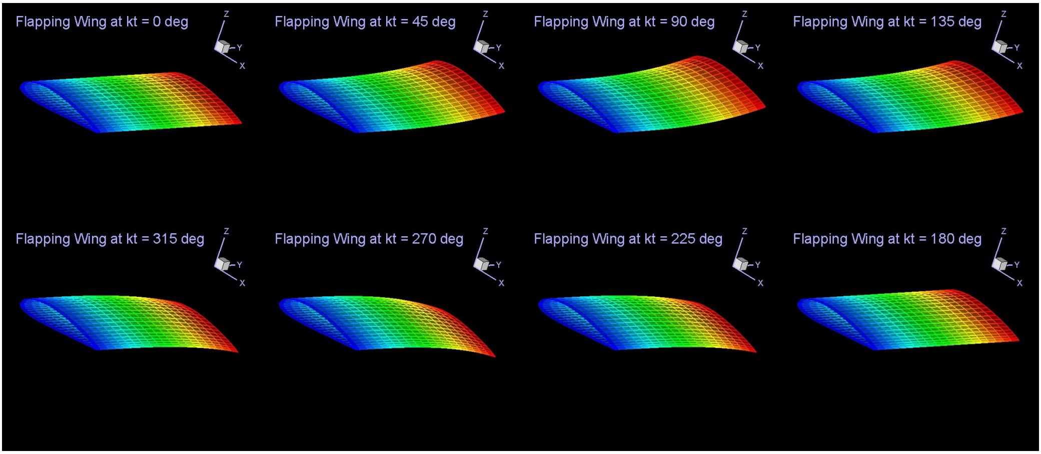
Flapping Wing at kt = 0 deg
Flapping Wing at kt = 45 deg
Flapping Wing at kt = 90 deg
Flapping Wing at kt = 135 deg
Flapping Wing at kt = 315 deg
Flapping Wing at kt = 270 deg
Flapping Wing at kt = 225 deg
Flapping Wing at kt = 180 deg
Z
Y
X

FIGURE 4: Body profile of the flapping wing at different stages of the flapping motion ($kt = 360^\circ$ for one full cycle of motion).

is referred to $\max\{|\mathcal{R}_{i,k}|\}$ for all grid points, and the error is computed from $\frac{1}{N}\sqrt{|\Delta \mathbf{r}|^2}$, where N is the total number of grid points. The tailplane has a NACA 63_2-615 section [1] with a maximum thickness of 0.15ℓ at 0.341ℓ from its leading edge, and has a semispan length of 0.72ℓ, where ℓ is the chord length measured at the root section of the body. The grid consists of 61 chordwise points, 31 radial points and 30 spanwise points (15 points equally spaced along the span and 15 points between the body and farfield spanwise boundary). The source term in Equation (1) is chosen such that grid lines are clustered toward the tailplane, as shown in Figures 1, 3 and 5. The comparison shows that the present scheme is superior to other schemes compared here, and noting that the error and residual reduce substantially (oscillating behaviour) whenever the present scheme completed one cycle of the time step cycling process (Figure 2 shows an example).

When the body moves, the grid around the body is required to be regenerated at each time level, since the grid points, especially those close to the body, move substantially. At each time level, the scheme generates a new

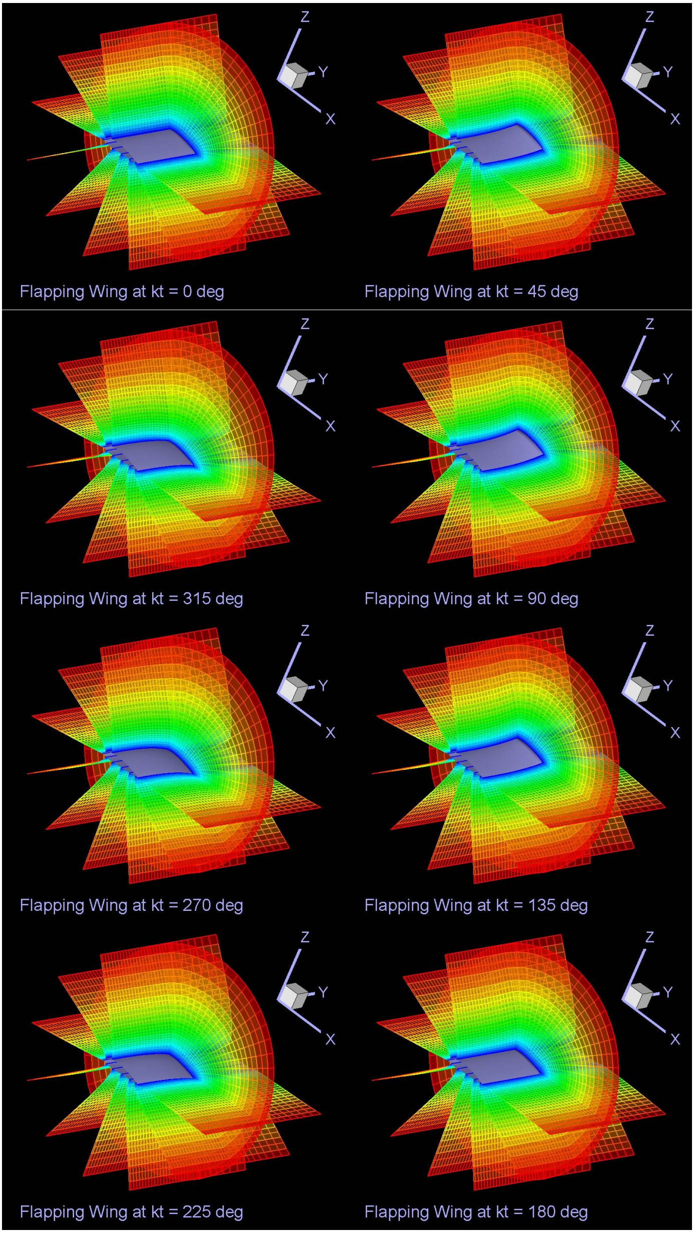
Flapping Wing at kt = 0 deg
Flapping Wing at kt = 45 deg
Flapping Wing at kt = 315 deg
Flapping Wing at kt = 90 deg
Flapping Wing at kt = 270 deg
Flapping Wing at kt = 135 deg
Flapping Wing at kt = 225 deg
Flapping Wing at kt = 180 deg
Z
Y
X

Figure 5: Grids generated around a flapping wing at different stages of the flapping motion ($kt = 360°$ for one full cycle of motion).

grid, subject to the new body boundary conditions, by utilising the grid at the previous time level as a starting state (treating it as an initial grid in a local sense). If the time step is small, the scheme requires several iterations to generate the new grid for an error tolerance of 10^{-8}. As an example, a dynamic grid system is generated around a rectangular wing of 1.25ℓ semispan length and with a NACA 4418 section [1] (maximum thickness of 0.1806ℓ at 0.2978ℓ from its leading edge). The wing consists of 51 chordwise points with 20 points equally spaced along the span, 11 points allocated to the region between the wing tip and the farfield spanwise boundary, and 31 points in the radial direction. The wing flaps harmonically as shown in Figure 4, and Figure 5 shows the generated grids around the body at different stage of the flapping motion cycle. The plots clearly show that the generated grids are smooth and clustered towards the wing, and that the grid smoothly changed to accommodate the new wing boundary.

4 Concluding remarks

A system of Poisson's equations and a grid stacking technique are employed to generate a structured OH-type grid around a three dimensional body in motion. The scheme incorporates a method of false transients and an approximate factorisation technique, where a variable time steps cycling process is used. The computational example for the aircraft tailplane shows that the scheme is significantly faster in reaching convergence to the required accuracy than all other numerical schemes considered in the comparison study. For a dynamic grid system around a moving body, only several iterations are required to regenerate a smooth and accurate boundary fitted grid at each new time level, as illustrated by the example of a flapping wing motion. The present method is suitable for automatic grid generation computer code, since a correct final grid system can always be obtained independent of the form of its initial grid.

Acknowledgements We thank RMIT University, Melbourne, Australia, for supporting this research work through a research grant, RMIT Emerging Researcher Grant 2007.

References

[1] Abbott, I. H., and von Doenhoff, A. E., *Theory of Wing Sections*, Dover Publications Inc., New York, USA, 1959. C513, C515

[2] Catherall, D., Optimum Approximate-Factorization Schemes for Two Dimensional Steady Potential Flows, *AIAA Journal*, **20**, 8, 1982, pp. 1057–1063. C507

[3] 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. C511, C512

[4] Hoffmann, K. A., *Computational Fluid Dynamics for Engineers*, Engineering Educational System, Texas, USA, 1989. C506, C507

[5] 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. C507, C509

[6] 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, http://mams.rmit.edu.au/3j9p7fh8xj2o1.pdf C507, C509, C510

[7] 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. C507

[8] Ly, E., and Norrison, D., Automatic Structured Grid Generation by an Approximate Factorisation Algorithm, Mathematics and Statistics Research Report 2006/02, School of Mathematical and Geospatial Sciences, RMIT University, Melbourne, Australia, July 2006, 28 pages, http://mams.rmit.edu.au/1354efwypnzn.pdf C506, C508, C510

[9] Ly, E., and Norrison, D., Automatic Elliptic Grid Generation by an Approximate Factorisation Algorithm, *ANZIAM Journal*, **48** (CTAC 2006), pp. C188–C202, July 2007. http://anziamj.austms.org.au/ojs/index.php/ANZIAMJ/article/view/48 C510

[10] Ly, E., and Norrison, D., Generating Elliptic Grids in Three Dimensions by a Method of False Transients, *ANZIAM Journal*, **49** (EMAC2007), pp. C170–C183, Nov. 2007. http://anziamj.austms.org.au/ojs/index.php/ANZIAMJ/article/view/313 C506, C507, C511

[11] 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. C506, C507

[12] 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. C506

[13] 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. C509

Author addresses

1. **E. Ly**, School of Mathematical and Geospatial Sciences, Science, Engineering and Technology Portfolio, RMIT University, Melbourne, Victoria 3001, Australia.
   `mailto:eddie.ly@rmit.edu.au`

2. **D. Norrison**, School of Mathematical and Geospatial Sciences, Science, Engineering and Technology Portfolio, RMIT University, Melbourne, Victoria 3001, Australia.