ELSEVIER

Comput. Methods Appl. Mech. Engrg. 131 (1996) 209–232

Computer methods in applied mechanics and engineering

A weak statement perturbation CFD algorithm with high-order phase accuracy for hyperbolic problems

Subrata Roy[1], A.J. Baker*

University of Tennessee, Knoxville, TN 37996-2030, USA

Received 9 November 1993; revised 4 May 1995

Abstract

Achieving improved order of accuracy for any numerical method is a continuing quest. The discrete approximate solution error, in general, can be expressed as a truncation of a Taylor series expansion. Herein, we present a weak statement perturbation always yielding simple tridiagonal forms that can reduce, or annihilate in special cases, the Taylor series truncation error to high order. The procedure is analyzed via a von Neumann frequency analysis, and verification CFD solutions are reported in one and two dimensions. Finally, using the element specific (local) Courant number, a *continuum* (total) time integration procedure is derived that can directly produce a final time solution *independent* of mesh measure.

1. Introduction

Consider the unsteady hyperbolic conservation law form

$$\mathscr{L}(q) = \frac{\partial q}{\partial t} + \frac{\partial f}{\partial x} - s = 0\,, \quad \text{on } \Omega \times t \subset \Re^d \times \Re^+ \tag{1}$$

In (1), q is the state variable, $f = uq$, is the kinetic flux vector with convection velocity vector u and s is the source term. The boundary of the d-dimensional ($1 \leq d \leq 3$) problem domain Ω is the *union* of segments $\partial\Omega_i$, i.e. $\partial\Omega = \partial\Omega_1 \cup \partial\Omega_2 \cup \partial\Omega_3$.

Approximate (CFD) algorithms to this non-diffusive transient problem class are characterized by a phase lag dispersion error that can highly distort the approximation solution, cf. Fig. 1(a). Specifically, the '$2\,\Delta x$ wave' zero phase velocity induces a short wave-length error that cascades to smaller wave numbers. The theoretically esthetic remedies are to increase the density of degrees of freedom, e.g. number of mesh nodes, or algorithm order-of-accuracy. Conversely, the 'practical' remedy is use of an artificial dissipation method, usually involving an added elliptic perturbation with a (linear or non-linear) numerical viscosity coefficient. A variation involves flux correction operators to remove induced excessive dissipation via an antidiffusive contribution. The goal is to induce a damping mechanism for control of non-monotonicity, but solutions tend to suffer from excess diffusion, Fig. 1(b), even with a very dense mesh. Detailed comparisons of this phenomenon for the 1-D and 2-D linear convection verification problems are available (cf. [1, Ch. 4.13], [7, Ch. 17.3 and Ch. 21.3]).

The Fourier representation of the analytical solution to (1) is

* Corresponding author.

[1] Currently at Computational Mechanics Corporation, Knoxville, TN 37919-3382, USA.

SSDI 0045-7825(95)00863-2

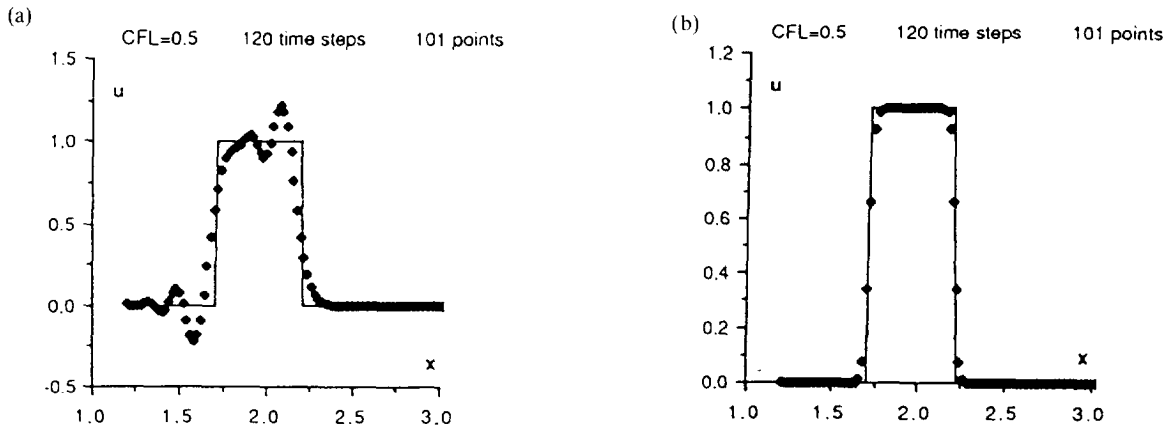

Fig. 1. Traveling square wave solution on 101 node mesh, [7, p. 566]. (a) Standard; (b) artificial dissipation with 'Superbee' limiter.

$$q(x,t)=\sum_{\zeta=-\infty}^{\infty} A_\zeta\, e^{i(\omega_\zeta x-\alpha_\zeta t)} \tag{2}$$

where $\omega_\zeta = 2\pi/\lambda_\zeta$ = wavenumber of ζth wave, λ_ζ = wavelength of ζth wave, $i=\sqrt{-1}$ and A_ζ = expansion coefficient. The linearity of (1) allows one to focus on the generic ζth mode only and consequently for (2), $\alpha_\zeta = u\omega_\zeta$. Hence, the exact solution $q(x,t)$ generic mode $q_\zeta(x,t)$ for any finite Fourier mode is

$$q_a(x,t)=e^{i\omega(x-ut)} \tag{3}$$

Assuming a semi-discrete solution $q^h(j\,\Delta x,t)$ behaves in the same manner,

$$q^h(j\,\Delta x,t)=e^{i\omega(j\,\Delta x-u^h t)}, \quad \text{where } u^h=u^h_{\Re}+iu^h_{\Im} \tag{4}$$

In (4) and hereafter, the superscript h denotes the spatial semi-discretization and the subscripts $\Re$ and $\Im$ denote real and imaginary solution components, respectively. Substituting $q_a(j\,\Delta x,t)=e^{i\omega(j\,\Delta x-ut)}$ in (4) yields

$$q^h(j\,\Delta x,t)=q_a(j\,\Delta x,t)\,e^{\omega t(u^h_{\Im}+i(u-u^h_{\Re}))} \tag{5}$$

The corresponding semi-discretization error, e^h, is

$$e^h=q_a-q^h=1-e^{\omega t(u^h_{\Im}+i(u-u^h_{\Re}))} \tag{6}$$

Thus, the phase dispersion error of the approximation solution is related to $i(u-u_{\Re})$ and the dissipation error is related to $u_{\Im}$.

For the fully discrete solution, after elapsed time $n\,\Delta t$, the fully discrete spatial Fourier component becomes,

$$\begin{aligned} Q^n_j(j\,\Delta x,t+n\,\Delta t) &\equiv e^{i\omega(j\,\Delta x-u^h(t+n\,\Delta t))}\\ &= e^{-i\omega u^h n\,\Delta t}\,e^{i\omega(j\,\Delta x-u^h t)}\\ &= G^n q^h(j\,\Delta x,t) \end{aligned} \tag{7}$$

The solution for algorithm amplification factor G determines how the fully discrete solution grows or decays in time, and

$$G^n=e^{-i\omega(u^h_{\Re}+iu^h_{\Im})n\,\Delta t}=G_{\Re}+iG_{\Im} \tag{8}$$

By definition, G^n obeys the following identity

$$G^n \, e^{i\omega u^h n \, \Delta t} = 1 \tag{9}$$

Comparing to the analytical solution (3), one may determine that in one dimension

$$G_a^n = \frac{q(j\,\delta x, t + n\,\delta t)}{q(j\,\delta x, t)} = \frac{e^{i\omega(j\,\delta x - U(t+n\,\delta t))}}{e^{i\omega(j\,\delta x - Ut)}} = e^{-i\omega U(n\,\delta t)} \tag{10}$$

where U is the magnitude of $\boldsymbol{u}$ in one dimension. For $n = 1$ and a virtual time increment δt

$$G_a = e^{-i\omega U\,\delta t} = e^{-i(\omega\,\delta x)(U\,\delta t)/(\delta x)} = e^{-imC}$$
$$= \sum_{l=0}^{\infty} a_l m^l = 1 - imC - \frac{(mC)^2}{2} + i\frac{(mC)^3}{3!} + \frac{(mC)^4}{4!} - i\frac{(mC)^5}{5!} - \frac{(mC)^6}{6!} + i\frac{(mC)^7}{7!} + \cdots \tag{11}$$

where $m = \omega\,\delta x$ is the virtual non-dimensional wavenumber with $C = U\,\delta t/\delta x$ the corresponding Courant number. In the last line of (11), G is expanded in a Laurent series on m with a_l the coefficients.

The discrete approximation solution amplification factor G is algorithm specific and may be computed via a Fourier modal analysis of the discretized CFD recursion relation. The order of accuracy of the numerical solution, hence algorithm stability, is characterized by G. Specifically, the '$2\,\Delta x$' zero phase velocity induces a short wave-length error that cascades to smaller wave numbers. Utilization of artificial viscosity, via an additional spatial derivative term in (1) or 'upwinding', can diffuse this error mode, but typically at the expense of an unacceptable amount of purely numerical dissipation, [8, 9]. An improved theory for selective dissipation is the Taylor Weak Statement (TWS), [3], where the dependent variable is expanded in a Taylor series in both time and space, and higher-order accuracy is attainable via detailed mathematical manipulations.

In this paper, we document an alternative approach of improving the algorithm accuracy by perturbing the linear basis matrix statement of a finite element discretized Galerkin weak statement (GWS). This Galerkin matrix perturbation (GMP) method is directly incorporated, and can be theoretically compared with a wide class of algorithms on a term-by-term basis. Numerical solutions are presented confirming theoretical expectations for algorithm performance for one- and two-dimensional benchmarks.

2. Weak statement algorithm

Independent of the dimension d of Ω, and for general variation of physical properties and the flux vector, the semi-discrete finite element (FE) weak statement for (1) always yields an ODE system of the form [2]

$$WS^h = [M]\{Q(t)\}' + \{R\} = \{0\} \tag{12a}$$

$$[M] = S_e[M]_e \tag{12b}$$

$$\{R\} = S_e(([U]_e + [D]_e)\{Q(t)\}_e - \{b(t)\}_e \tag{12c}$$

where $\{Q(t)\}$ is the time-dependent discrete approximation nodal coefficient set. $\{Q(t)\}'$ denotes $d\{Q\}/dt$, which may be replaced by using a θ-implicit or n-step Runge–Kutta time integration procedure. In (12a–c) and thereafter, subscript e denotes element rank matrices, and S_e symbolizes the 'assembly operator' carrying local matrix coefficients into the global arrays. On each finite element, $[M]_e$ is the 'mass' matrix associated with interpolation, the $[U]_e$ matrix carries the convection information, $[D]_e$ is the diffusion matrix resulting from genuine or artificial viscosity effects, and $\{b\}_e$ contains all known data.

The GWS linear basis one-dimensional equivalents of the interpolation matrix $[M]_e$ and the convection matrix $[U]_e$ for uniform fluid speed U are of the forms, [2],

$$[M]_e = \Delta x_e[\text{A200L}] = \frac{\Delta x_e}{6}\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}_{2\times 2}, \qquad [U]_e = U[\text{A201L}] = \frac{U}{2}\begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix}_{2\times 2} \tag{13}$$

A Galerkin Weak Statement (GWS) for (1) does not produce a diffusion matrix $[D]_e$ in (12a–c), while a TWS formulation does. The order of accuracy of a numerical algorithm may be determined from its amplification factor G by matching coefficients of powers in wavenumber in the Laurent series expansion with the analytical solutions G_a in (11). Using a Taylor series expansion one may readily see that the linear basis FE approximation to the spatial derivative about the point j is of the form

$$\frac{q_{j+1}-q_{j-1}}{2\,\Delta x}=\left[\frac{\partial q}{\partial x}+\frac{\Delta x^2}{3}\frac{\partial^3 q}{\partial x^3}+\frac{\Delta x^4}{60}\frac{\partial^5 q}{\partial x^5}\right]_j+\mathcal{O}(\Delta x^6) \tag{14}$$

which is obviously second-order accurate. Similarly, the θ-implicit temporal approximation, for $0\leqslant\theta\leqslant 1$, may also be represented in a Taylor series as

$$\{Q\}^{n+\Delta t}=\{Q\}^n+\Delta t\left(\theta\frac{\mathrm{d}\{Q\}^{n+\Delta t}}{\mathrm{d}t}+(1-\theta)\frac{\mathrm{d}\{Q\}^n}{\mathrm{d}t}\right)+\mathcal{O}(\Delta t^2,\Delta t^3) \tag{15}$$

The standard trapezoidal ($\theta=1/2$) GWS form (15) is second-order accurate for the linear basis FE construction. Higher-order accuracy may be achieved by using a higher degree FE basis, or the TWS can maintain the third-order accuracy of the linear FE basis form with added dissipation.

For uniform mesh, [10] has reduced the Taylor series truncation error of the FD approximation to the first-order spatial derivative terms by including the $j\pm 3$, $j\pm 2$ terms in (14) as

$$\begin{aligned}&\chi\frac{\mathrm{d}Q_{j-2}}{\mathrm{d}x}+\phi\frac{\mathrm{d}Q_{j-1}}{\mathrm{d}x}+\frac{\mathrm{d}Q_j}{\mathrm{d}x}+\phi\frac{\mathrm{d}Q_{j+1}}{\mathrm{d}x}+\chi\frac{\mathrm{d}Q_{j+2}}{\mathrm{d}x}\\&\quad=a\frac{q_{j+1}-q_{j-1}}{2\,\Delta x}+b\frac{q_{j+2}-q_{j-2}}{4\,\Delta x}+c\frac{q_{j+3}-q_{j-3}}{6\,\Delta x}+\cdots\end{aligned} \tag{16}$$

where the following relation must hold

$$a+b+c=1+2\phi+2\chi \tag{17}$$

3. The Galerkin matrix perturbation (GMP) algorithm

A Fourier modal analysis can determine how the error-induced perturbations propagate through a fully discrete approximation. The algorithm specific amplification factor G for an $n=1$ step method is

$$G=\frac{Q_j^{t+\Delta t}}{Q_j^t}=\mathrm{e}^{-\mathrm{i}\omega U^h\Delta t}=\mathrm{e}^{-\mathrm{i}mC^h} \tag{18}$$

where the discrete Courant number C^h depends on the specific solution algorithm.

Assuming that an arbitrary order of the Taylor series truncation error for the temporal and spatial discretization (14)–(15) may be expressed in the form of FE perturbation matrices [APERT] and [APERX], the temporal and convection one-dimensional finite element matrices of linear basis rank for (12b)–(12c) become

$$\begin{aligned}\frac{1}{\Delta x_e}[\mathrm{M}]_e&=[\mathrm{A200L}]-[\mathrm{APERT}]\\&=\frac{1}{6}\begin{bmatrix}2&1\\1&2\end{bmatrix}_{2\times 2}-\frac{1}{6}\begin{bmatrix}\Lambda_1&\Psi_1\\\Gamma_1&\Upsilon_1\end{bmatrix}_{2\times 2}=\frac{1}{6}\begin{bmatrix}2-\Lambda_1&1-\Psi_1\\1-\Gamma_1&2-\Upsilon_1\end{bmatrix}_{2\times 2}\end{aligned} \tag{19}$$

and

$$\begin{aligned}\frac{1}{U}[U]_e&=[\mathrm{A201L}]-[\mathrm{APERX}]\\&=\frac{1}{2}\begin{bmatrix}-1&1\\-1&1\end{bmatrix}_{2\times 2}-\frac{1}{2}\begin{bmatrix}\Lambda_2&\Psi_2\\\Gamma_2&\Upsilon_2\end{bmatrix}_{2\times 2}=\frac{1}{2}\begin{bmatrix}-(1+\Lambda_2)&1-\Psi_2\\-(1+\Gamma_2)&1-\Upsilon_2\end{bmatrix}_{2\times 2}\end{aligned} \tag{20}$$

In (19)–(20), [A200L] and [A201L] remain the GWS linear basis constructions, while

$$\Lambda_1 = t_1 \sum_{r=0}^{\alpha} a_r z^{-r}, \qquad \Psi_1 = t_2 \sum_{r=0}^{\beta} a_r z^{r},$$
$$\Gamma_1 = t_3 \sum_{r=0}^{\gamma} a_r z^{-r}, \qquad \Upsilon_1 = t_4 \sum_{r=0}^{\delta} a_r z^{r} \tag{21a}$$

$$\Lambda_2 = t_5 \sum_{r=0}^{\alpha} a_r z^{-r}, \qquad \Psi_2 = t_6 \sum_{r=0}^{\beta} a_r z^{r},$$
$$\Gamma_2 = t_7 \sum_{r=0}^{\gamma} a_r z^{-r}, \qquad \Upsilon_2 = t_8 \sum_{r=0}^{\delta} a_r z^{r} \tag{21b}$$

where $z = e^{im}$ is the complex modulus. The superscripts $(\alpha, \beta, \gamma, \delta) \geqslant 0$ are integers, to be determined for desired order of accuracy (Appendix A). To maintain time accuracy of the algorithm, one necessary condition for (21a–b) is

$$\Lambda_i + \Psi_i + \Gamma_i + \Upsilon_i = 0; \quad \text{for } i = 1, 2 \tag{22}$$

For (19)–(20) the fully discrete weak statement algorithm (12a–c) assembly at mesh node j with variable $Q_j^n = Q(j\,\Delta x, t + n\,\Delta t)$ yields the recursion relation

$$\left(\frac{\Delta x(1-\Gamma_1)}{6\theta\,\Delta t} - \frac{(1+\Gamma_2)U}{2}\right)_e Q_{j-1} + \left(\frac{\Delta x(4-\Lambda_1-\Upsilon_1)}{6\theta\,\Delta t} + \frac{(\Lambda_2+\Upsilon_2)U}{2}\right)_e Q_j$$
$$+ \left(\frac{\Delta x(1-\Psi_1)}{6\theta\,\Delta t} + \frac{(1-\Psi_2)U}{2}\right)_e Q_{j+1}$$
$$= \text{data from } (n-1) \text{ timestep} \tag{23}$$

The amplification factor (18) for (23) is

$$G = \frac{Q_j^{t+\Delta t}}{Q_j^t} = \frac{G_N}{G_D} \tag{24}$$

where G_N and G_D can be resolved into spatial $G(x)$ and temporal $G(t)$ components as

$$G_N = G(t) - (1-\theta)CG(x) \quad \text{and} \quad G_D = G(t) + \theta CG(x) \tag{25}$$

where, using relations (7) and (9), via (12a–c) and (23), yields

$$G(t) = \frac{(1-\Gamma_1)\,e^{-im} + (4-\Lambda_1-\Upsilon_1) + (1-\Psi_1)\,e^{im}}{6}$$
$$G(x) = \frac{(1-\Psi_2)\,e^{im} - (\Lambda_2+\Upsilon_2) - (1+\Gamma_2)\,e^{-im}}{2} \tag{26}$$

Hence, (25) becomes

$$G_N = \left(\frac{1-\Gamma_1}{6} + (1-\theta)C\frac{1+\Gamma_2}{2}\right)e^{-im} + \frac{4-\Lambda_1-\Upsilon_1}{6} + (1-\theta)\frac{C}{2}(\Lambda_2+\Upsilon_2)$$
$$+ \left(\frac{1-\Psi_1}{6} - (1-\theta)C\frac{1-\Psi_2}{2}\right)e^{im} \tag{27}$$
$$G_D = \left(\frac{1-\Gamma_1}{6} - \theta C\frac{1+\Gamma_2}{2}\right)e^{-im} + \frac{4-\Lambda_1-\Upsilon_1}{6} - \theta\frac{C}{2}(\Lambda_2+\Upsilon_2) + \left(\frac{1-\Psi_1}{6} + \theta C\frac{1-\Psi_2}{2}\right)e^{im}$$

The GMP algorithm amplification factor (24):(27) may now be expanded in a Laurent series on m (Appendix B). The procedure is very tedious; however, using 'Macsyma' (1985), and for $\alpha = \beta = \gamma = \delta = 2$ and $a_r = 1$ in (21a–b), one may obtain up to and including the second-order term of the series in the form

$$G = \sum_{l=0}^{\infty} A_l m^l = 1 + (3\mathrm{i}Ct_8 + 6\mathrm{i}Ct_6 + 3\mathrm{i}Ct_5 - 2\mathrm{i}C)m/2$$
$$- ((9C^2t_8^2 + (36C^2t_6 + 18C^2t_5 - 12C^2)t_8 + 36C^2t_6^2$$
$$+ (36C^2t_5 - 24C^2)t_6 + 9C^2t_5^2 - 12C^2t_5 + 4C^2)\theta$$
$$+ (3Ct_4 + 6Ct_2 + 3Ct_1 + 3C)t_8 + (6Ct_4 + 12Ct_2 + 6Ct_1 + 12C)t_6$$
$$+ (3Ct_4 + 6Ct_2 + 3Ct_1 - 9C)t_5 - 2Ct_4 - 4Ct_2 - 2Ct_1)m^2/4 + \mathcal{O}(m^3)\ldots \tag{28}$$

The coefficients of the series (28) may be functionally expressed as

$$A_l = f(C, \theta, t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, \alpha, \beta, \gamma, \delta) \tag{29}$$

where l corresponds to the power of m, in the Laurent expansion. Comparing the sets a_l of (11) to A_l of (28), for a given degree of m a general correlation between C, θ and the elements of [APERT] and [APERX] may be derived (Appendix B) that can yield progressively higher-order accurate algorithms without adding a diffusion term to the algorithm.

A simplified example is pertinent to illustrate the process. Selecting

$$\theta = \frac{1}{2}, \qquad \alpha = \beta = \gamma = \delta = 0, \quad \text{and} \quad t_5 = t_8 = 0 \tag{30}$$

than for the choice

$$t_1 = t_4, \qquad t_2 = t_3, \quad \text{and} \quad t_6 = -t_7 \tag{31}$$

one may produce a fourth-order accurate GMP algorithm by satisfying the relationships

$$t_1 + t_2 = 0$$
$$t_6 = -t_7 = \frac{t_2 + t_1}{3} = 0 = t_5 = t_8$$
$$t_1 = t_4 = \frac{3C^2 + (4 - C^2)t_2}{C^2 + 2} = -t_2 = -t_3 \tag{32}$$

i.e.

$$t_1 = t_4 = \frac{C^2}{2} = -t_2 = -t_3$$

Hence, for $C = 0.8$, (32) yields

$$t_5 = t_6 = t_7 = t_8 = 0 = \Lambda_2 = \Psi_2 = \Gamma_2 = \Upsilon_2$$
$$t_1 = t_4 = -t_2 = -t_3 = \frac{C^2}{2} = \frac{8}{25} = \Lambda_1 = \Upsilon_1 = -\Psi_1 = -\Gamma_1 \tag{33}$$

and the corresponding $[M]_e$ and $[U]_e$ matrices become

$$[M]_e = \Delta x_e([\mathrm{A200L}]\text{–}[\mathrm{APERT}])$$
$$= \frac{\Delta x_e}{6}\begin{bmatrix} 2 - \frac{C^2}{2} & 1 + \frac{C^2}{2} \\ 1 + \frac{C^2}{2} & 2 - \frac{C^2}{2} \end{bmatrix} = \frac{\Delta x_e}{150}\begin{bmatrix} 42 & 33 \\ 33 & 42 \end{bmatrix}$$
$$[U]_e = U([\mathrm{A201L}]\text{–}[\mathrm{APERX}]) = U([\mathrm{A201L}]) = \frac{U}{2}\begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix} \tag{34}$$

Using (32), the solution amplification factor (24) is

$$G = 1 - \mathrm{i}(Cm) - \frac{(Cm)^2}{2} + \mathrm{i}\frac{(Cm)^3}{3!} + \frac{(Cm)^4}{4!} - \mathrm{i}\frac{\frac{1}{6}(5C^5 + 5C^3 - 4C)m^5}{5!} - \frac{(5C^4 - 4C^2)m^6}{6!} - \mathrm{i}\frac{\frac{1}{12}(35C^7 - 245C^5 + 238C^3 - 40C)m^7}{7!} - \mathcal{O}(m^8) \tag{35}$$

One can readily see that (35) matches exactly with (11) up to the fourth-order term ($\mathcal{O}(m^4)$). A similar result may be obtained by using TWS algorithm [3], which can be made fourth-order accurate.

As a special case (ref. Section 4), selecting $C = 1$ yields $t_1 = t_4 = \frac{1}{2} = -t_2 = -t_3$ and $t_6 = t_8 = (t_1 + t_2)/3 = 0 = -t_5 = -t_7$, hence

$$[M]_e = \Delta x_e([\mathrm{A200L}]\text{–}[\mathrm{APERT}]) = \frac{\Delta x_e}{4}\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix},$$
$$[U]_e = U([\mathrm{A201L}]\text{–}[\mathrm{APERX}]) = \frac{U}{2}\begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix} \tag{36}$$

for which the amplification factor of the approximation solution for (10), in terms of wave number m, is

$$G = 1 - \mathrm{i}m - \frac{m^2}{2} + \mathrm{i}\frac{m^3}{3!} + \frac{m^4}{4!} - \mathrm{i}\frac{m^5}{5!} - \frac{m^6}{6!} + \mathrm{i}\frac{m^7}{7!} + \frac{m^8}{8!} - \mathrm{i}\frac{m^9}{9!} - \frac{m^{10}}{10!} + \mathrm{i}\frac{m^{11}}{11!} + \frac{m^{12}}{12!} - \mathrm{i}\frac{m^{13}}{13!} - \frac{m^{14}}{14!} + \mathrm{i}\frac{m^{15}}{15!} + \frac{m^{16}}{16!} - \mathrm{i}\frac{m^{17}}{17!} - \frac{m^{18}}{18!} + \cdots = \mathrm{e}^{-\mathrm{i}m} = G_a \tag{37}$$

which is exact (matched) with the analytical solution amplification factor g_a to *eighteenth* order.

For another example, selecting $C = 2$ yields $t_1 = t_4 = 2 = -t_2 = -t_3$ and $t_6 = t_8 = (t_1 + t_2)/3 = 0 = -t_5 = -t_7$, hence

$$[M]_e = \Delta x_e([\mathrm{A200L}]\text{–}[\mathrm{APERT}]) = \frac{\Delta x_e}{2}\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix},$$
$$[U]_e = U([\mathrm{A201L}]\text{–}[\mathrm{APERX}]) = \frac{U}{2}\begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix} \tag{38}$$

Here again, the solution amplification factor, in terms of wave number m, is

$$G = 1 - 2\mathrm{i}m - 2m^2 + \mathrm{i}\frac{4m^3}{3} + \frac{2m^4}{3} - \mathrm{i}\frac{4m^5}{15} - \frac{4m^6}{45} + \mathrm{i}\frac{8m^7}{315} + \frac{2m^8}{315} - \mathrm{i}\frac{4m^9}{2835} - \frac{4m^{10}}{14175} + \mathrm{i}\frac{8m^{11}}{155925} + \frac{4m^{12}}{467775} - \mathrm{i}\frac{8m^{13}}{6081075} - \frac{8m^{14}}{42567525} + \cdots = \mathrm{e}^{-2\mathrm{i}m} = G_a \tag{39}$$

which is exact (matched) to *fourteenth* order with the analytical solution amplification factor G_a. However, in this case, one may not obtain the exact solution numerically using (38) (Appendices A and C). In general, a von Neumann frequency analysis is necessary but *not sufficient* to guarantee the approximate solution accuracy. Specifically, matching the coefficients of equal powers of m in (11) and (28) only guarantees that the ratio $C^h/C = 1$, but does not guarantee the phase velocity relationship $U^h = U$.

A specific order of solution accuracy, e.g. *sixth order*, can be achieved by selecting the summation limits $\alpha, \beta, \gamma, \delta$ in (21a)–(21b) for [APERT] and [APERX], as described in Appendix A. The corresponding coefficients t_1–t_8 in (21a)–(21b) can then be determined. For example, see Appendix B, selecting $\beta = \gamma = \alpha = \delta = 2$ guarantees a *sixth-order* accurate solution for *any* a_r, $0 \leq r \leq \alpha, \beta, \gamma, \delta$. Specifically, for $\theta = 1/2$ and $a_r = 1$, the coefficients in (21a)–(21b) become

$$
\begin{aligned}
t_2 &= -\frac{4C^4 + 30C^2 + 6}{45C^2 - 405}, \\
t_1 &= \frac{(15C^4 + 75C^2 - 90)t_2 + 3C^6 + 22C^4 + 47C^2 + 8}{30C^4 - 30C^2 - 360}, \\
t_4 &= -\frac{(6C^2 - 6)t_2 + (3C^2 - 36)t_1 + C^2}{3C^2 + 12}, \\
t_8 &= -\frac{21t_4 + 72t_2 - 39t_1 + 5C^2}{54}, \\
t_5 &= -\frac{3t_8 + 2t_4 + 4t_2 + 2t_1}{15} \quad \text{and} \quad t_6 = -\frac{t_8 + t_5}{2}
\end{aligned} \tag{40}
$$

Although the GMP algorithm needs only two element support for each node, i.e. the equation for Q_j involves Q_{j-1} and Q_{j+1} only, the procedure (19)–(20) results can be compared directly with compact finite difference schemes, cf. [10]. However, the theory and implementation of GMP is much simpler. Also note that the GMP assembly matrices in (34)–(36) retain the efficiency of a linear basis (or centered FD) algorithm form.

4. Direct time integration

The necessary condition for the approximation solution of (1) to agree exactly with the continuum solution (2) is certainly

$$G \equiv G_a \tag{41}$$

Hence, the formulation (theoretical) goal is for Eqs. (10) and (18) to match to extremum order. The general rational form of (18)–(19) is

$$G = \frac{a_1 e^{-im} + a_3 e^{im} + a_2}{b_1 e^{-im} + b_3 e^{im} + b_2} = \frac{a_1 + a_3 e^{2im} + a_2 e^{im}}{b_1 + b_3 e^{2im} + b_2 e^{im}} \tag{42}$$

where a_i and b_i, $1 \leqslant i \leqslant 3$, are specific CFD algorithm coefficients. From (6) and (13), one can write ([11]),

$$
\begin{aligned}
&(a_1 + a_2 e^{im} + a_3 e^{2im}) e^{imC} - (b_1 + b_2 e^{im} + b_3 e^{2im}) \\
&\quad = a_1 \sum_{n=0}^{\infty} \left(\frac{i^n}{n!} C^n\right) m^n + a^2 \sum_{n=0}^{\infty} \left(\frac{i^n}{n!} (C+1)^n\right) m^n + a_3 \sum_{n=0}^{\infty} \left(\frac{i^n}{n!} (C+2)^n\right) m^n \\
&\qquad - b_1 - b_2 \sum_{n=0}^{\infty} \left(\frac{i^n}{n!}\right) m^n - b_3 \sum_{n=0}^{\infty} \left(\frac{i^n}{n!} 2^n\right) m^n \\
&\quad = (a_1 + a_2 + a_3 - b_1 - b_2 - b_3) + \sum_{n=0}^{\infty} \frac{i^n}{n!} [a_1 C^n + a_2 (C+1)^n + a_3 (C+2)^n - b_2 - 2^n b_3] \\
&\quad = 0
\end{aligned} \tag{43}
$$

i.e.

$$
\begin{aligned}
&\det \begin{vmatrix}
-1 & -2 & C & (C+1) & (C+2) \\
-1 & -2^2 & C^2 & (C+1)^2 & (C+2)^2 \\
-1 & -2^3 & C^3 & (C+1)^3 & (C+2)^3 \\
-1 & -2^4 & C^4 & (C+1)^4 & (C+2)^4 \\
-1 & -2^5 & C^5 & (C+1)^5 & (C+2)^5
\end{vmatrix} \\
&\quad = 4C^3 (C+1)^2 (C+2)(C-1)^2 (C-2) = 0
\end{aligned} \tag{44}
$$

The condition (41) yields the following roots for C

$$C = 0, \pm 1, \pm 2 \tag{45}$$

For a time integration ranging from t to $t + n_e \Delta t_e$, where the subscript 'e' denotes a specific element, the weak statement algorithm (12a–c) yields the following θ-implicit relation

$$S_e n_e(([M]_e + \theta\, \Delta t_e[U]_e)\{\Delta Q\}_e - \Delta t_e(\{b\}_e - [U]_e\{Q^{n_e}\}_e)) = \{0\} \tag{46}$$

where

$$\{\Delta Q\}_e = \{Q^{n_e+1} - Q^{n_e}\}_e\,, \qquad n_e = \frac{T}{\Delta t_e} \quad \text{and} \quad \Delta t_e = \frac{C_e\, \Delta x_e}{|U_e|} \tag{47}$$

The condition for (46) to be true is

$$C_e = \frac{T|U_e|}{\Delta x_e} \tag{48}$$

Consequently, the weak statement modified to total time T is

$$S_e\left(\left(\frac{T|U_e|}{C_e\, \Delta x_e}\Delta x_e[M] + \theta(n_e\, \Delta t_e)U_e[U]\right)\{\Delta Q\}_e - (n_e\, \Delta t_e)(\{b\}_e - U_e[U]\{Q^{n_e}\}_e)\right) = \{0\} \tag{49}$$

Hence, for element specific C_e, taking the continuum total time T outside the element matrix assembly S_e, one may derive

$$S_e\left(\left(\frac{|U_e|}{C_e}[M] + \theta U_e[U]\right)\{\Delta Q\}_e - (\{b\}_e - U_e[U]\{Q^{n_e}\}_e)\right) = \{0\} \tag{50}$$

The multiplier coefficient of the mass matrix $[M]$ is always positive (and equal to $|U_e|/C_e$) while for the convection matrix $[U]$ the multiplier is strictly U_e.

Then, for $C_e = 1$, (50) in one dimension becomes

$$S_e((|U_e|[\mathrm{A200}] + \theta U_e[\mathrm{A201}])\{\Delta Q\}_e - (\{b\}_e - U_e[\mathrm{A201}]\{Q^{n_e}\}_e)) = \{0\} \tag{51}$$

Using (51), one can enforce the element specific Courant number $C_e = 1$, and hence, the approximation solution $Q(n_e\, \Delta t_e, j\, \Delta x_e)$ from (12a–c) becomes exactly equal to the analytical solution q in (2) at any time T. If $C \neq 1$, then one may derive for one dimension

$$S_e\left(\left(\frac{|U_e|}{C_e}[\mathrm{A200}] + \theta U_e[\mathrm{A201}]\right)\{\Delta Q\}_e - (\{b\}_e - U_e[\mathrm{A201}]\{Q^{n_e}\}_e)\right) = \{0\} \tag{52}$$

5. Multidimensional forms

5.1. Tensor product factorization algorithm

The fluid convection contribution to (1) can be resolved into components parallel to the global coordinate system x_i. The multi-dimensional form of (12a–c) is [2]

$$S_e(([M]_e + \theta\, \Delta t_e[U]_e)\{\Delta Q\}_e + \Delta t_e([U]_e\{Q^{n_e}\}_e)) = \{0\} \tag{53}$$

where boldface U indicates a vector field. On assembly over all Ω_e of Ω^h, (53) yields a system matrix $[S]$ which can be approximated as

$$[S] \equiv S_e([M]_e + \theta\, \Delta t_e[U]_e) \approx [SX] \otimes [SY] \tag{54}$$

where the symbol $\otimes$ denotes the outer (*tensor*) matrix product. The product $[SX] \otimes [SY]$ in (54) is inexact with truncation error of order $(\theta\, \Delta t^2)$, which is the same order as the forward $(\theta = 0)$ or backward $(\theta = 1)$ Euler integration procedures in (10).

The definitions for forming (54) are

$$[SX] \equiv S_e([SX]_e) = S_e([MX]_e + \theta\, \Delta t_e [UX]_e) \tag{55a}$$

$$[SY] \equiv S_e([SY]_e) = S_e([MY]_e + \theta\, \Delta t_e [UY]_e) \tag{55b}$$

From (55a), via the modified weak statement (50)–(51), one may derive that for $C_e = 1$

$$n_e S_e([SX]_e) = TS_e(|U_e|[\mathrm{A200}] + \theta U_e[\mathrm{A201}]) \tag{56a}$$

$$n_e S_e([SY]_e) = TS_e(|V_e|[\mathrm{A200}] + \theta V_e[\mathrm{A201}]) \tag{56b}$$

where U_e and V_e are element dependent velocity components in the x and y directions, respectively. The corresponding approximation linear algebra solution procedure for (53) becomes the two-step sequence

$$\begin{aligned} &[SX]\{P\} = -\Delta t\{R\} \\ &[SY]\{\Delta Q\} = \{P\} \end{aligned} \tag{57}$$

where $\{P\}$ is an intermediate array of data.

5.2. The time-splitting (TS) algorithm

In addition to (54), the residual in (57) may also be approximately resolved into directional components

$$\{R\} \approx \{RX\} \otimes \{RY\} = S_e(\{RX\}_e + \{RY\}_e) \tag{58}$$

where

$$\{RX\}_e = U_e[\mathrm{A201}] \quad \text{and} \quad \{RY\}_e = V_e[\mathrm{A201}] \tag{59}$$

The insertion of (58) into (57) is termed time-splitting, and in two dimensions the TS formulation is operationally

$$\begin{aligned} &S_e([SX]_e)\{P\} = -\frac{1}{2} S_e(\{RX(Q)^{n_e}\}_e) \\ &\{Q\}^{n_e+1/2} \equiv \{Q\}^{n_e} + \{P\} \end{aligned} \tag{60a}$$

$$\begin{aligned} &S_e([SY]_e)\{\Delta Q\} = -\frac{1}{2} S_e(\{RY(Q)^{n_e+1/2}\}_e) \\ &\{Q\}^{n_e+1} \equiv \{Q\}^{n_e+1/2} + \{\Delta Q\} \end{aligned} \tag{60b}$$

where $\{Q\}^{n_e+1/2}$ is interpreted as an intermediate solution approximation.

6. Results and discussion

The analytical solution to (1) in one and two dimensions, for $s = 0$, is exact preservation of the initial condition translated on Ω over $n\,\Delta t$. Computational results herein verify theory and summarize performance for the traveling wave propagation verification problem, for a range of Courant numbers ($C \leq 2$) and with four different initial condition sets including a Gaussian wave, a full sine function distribution, and two square waves of different initial wavelength. The two-dimensional unsteady verification problem is solid body rotation of an axisymmetric initial condition, i.e. the 'cosine hill' convection problem. The non-dissipative trapezoidal rule ($\theta = 0.5$) time integration algorithm is used for solving all verification problems.

6.1. One-dimensional smooth and non-smooth wave convection, $C \leq 1$

An isolated energy packet is a standard verification problem that has been studied for one-dimensional pure convection wave propagation by [1, 6] using FE and a finite difference (FD) Crank–Nicolson algorithms. The one-dimensional form of Eq. (1) is

$$\mathscr{L}(q) = \frac{\partial q}{\partial t} + U\frac{\partial q}{\partial x} = 0\,, \quad \text{on } \Omega \times t \subset \Re^1 \times \Re^+ \tag{61}$$

where U is a constant and the analytical solution $q(x,t)$ is preservation of the initial condition $q(x, t = t_0)$ as it is convected parallel to x. In Fig. 2, the initial and the analytical final solutions of a Gaussian distribution convection GWS and GMP solution data are compared for Courant number $C = 1$ after 20 timesteps. The initial wave is interpolated across 10 nodes, hence the waveform has been translated 2 wavelengths. The GWS solution is non-dissipative and dispersive, as theoretically predicted, with trailing oscillatory maxima about 20% and with peak level reduced (via distortion) by almost 10%, Fig. 2(a). The comparative GMP algorithm (35) solution is dispersion-free and nodally exact to within round-off error (order 10^{-16}), Fig. 2(b).

For propagation of a full sine wave, 20 timesteps at $C = 1$ translates q_0 two wavelengths and the leading (negative) extremum of the GWS solution decays by nearly 40% and trailing dispersion error dominates the entire solution, Fig. 3(a). The companion GMP algorithm solution is totally devoid of $2\Delta x$ error oscillations and the sine wave pattern is nodally exact (to round-off), Fig. 3(b). Fig. 4 documents the companion $C = 0.5$ GWS and GMP solutions after 40 timesteps (two wavelengths displacement). The lower Courant number slightly improves the GWS solution, but it is still unacceptably polluted by dispersion error, Fig. 4(a). Conversely, for $\beta = \gamma = 1$ and $t_1 = t_4 = 1/8$, $t_2 = t_3 = -1/8$, the GMP algorithm solution remains nodally exact, Fig. 4(b), albeit to a lower order in round-off (order 10^{-6}).

A range of CFD algorithms applied to evolution of a square wave (non-smooth) propagation verification is detailed by Hirsch [7, Ch. 21.3], recall Fig. 1. For an initially 20-node wide square wave, Fig. 5 compares the GWS and GMP solutions for $C = 1$ for propagation over 3 wavelengths. The linear basis GWS solution dispersion error totally distorts the initial wave geometry. In distinction, the GMP algorithm solution, using (36), is a nodewise exact solution to within round-off error ($\sim 10^{-16}$).

Solutions obtained for $C = 0.5$ after 120 timesteps are compared in Fig. 6. In qualitative agreement

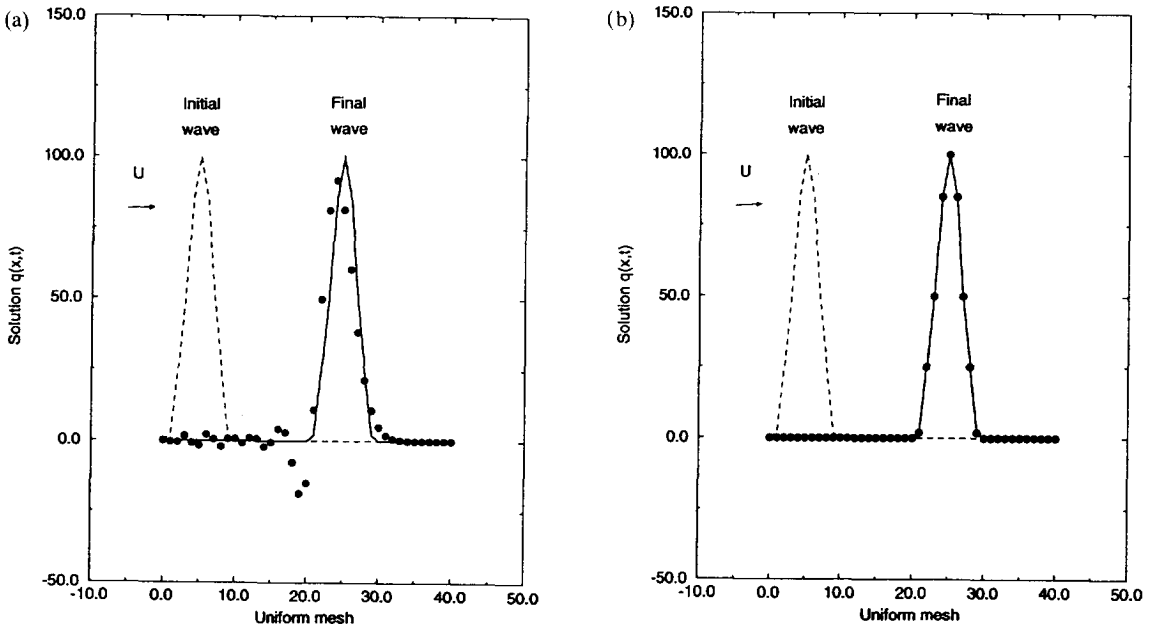

Fig. 2. Gaussian wave propagation, two wavelengths at $C = 1$. (a) GWS nodal solution; (b) GMP nodal solution.

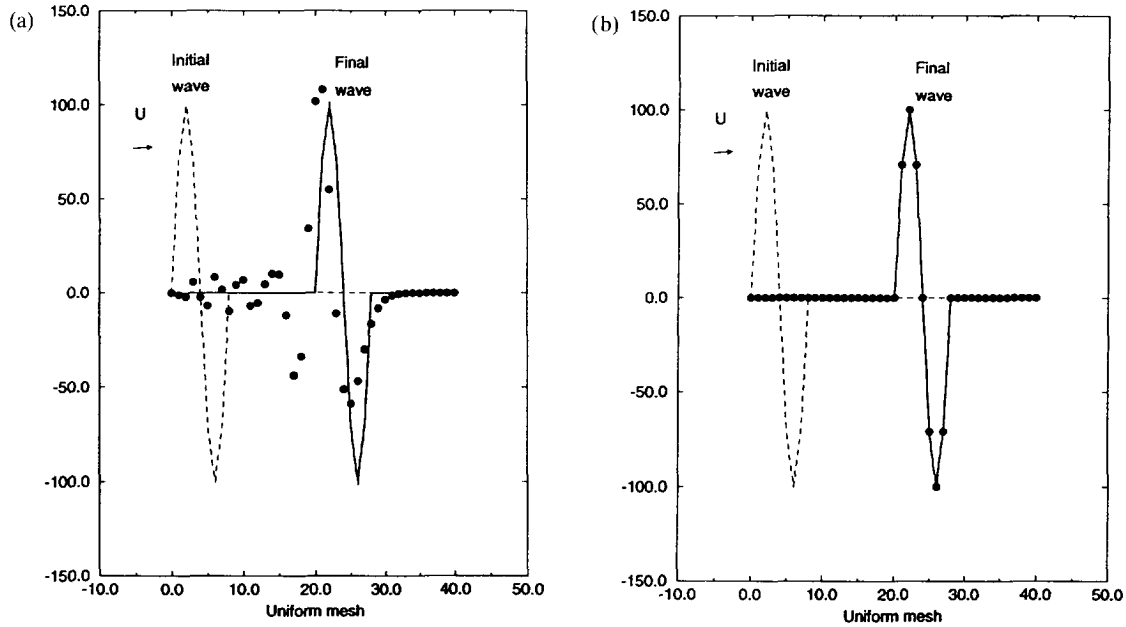

Fig. 3. Full sine wave propagation, two wavelengths at $C = 1$. (a) GWS solution; (b) GMP solution.

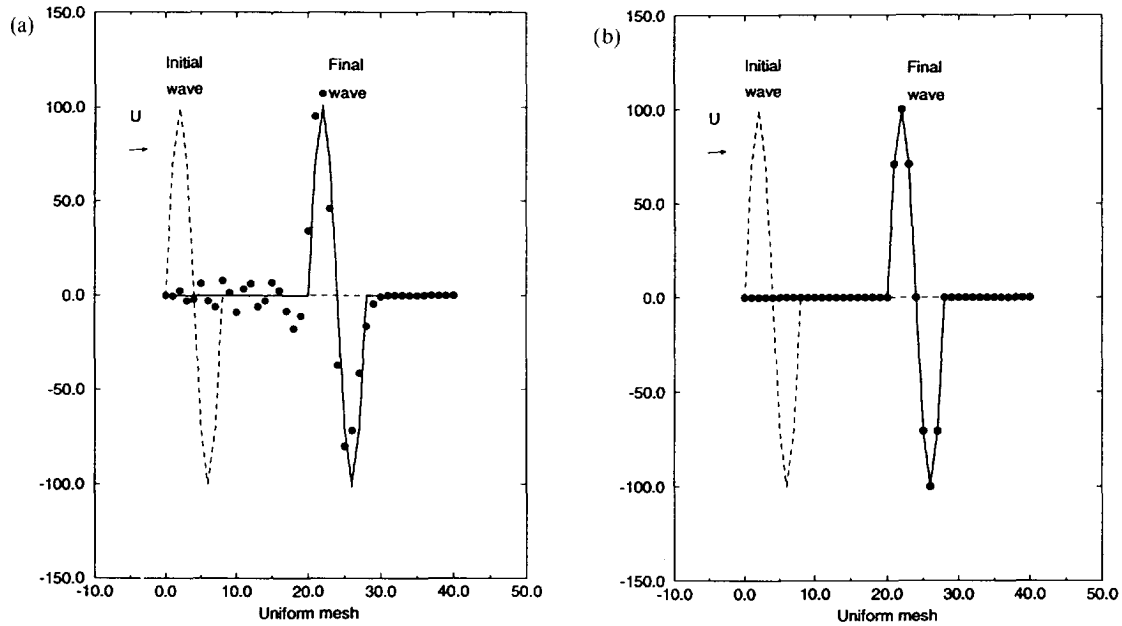

Fig. 4. Full sine wave propagation, two wavelengths at $C = 0.5$. (a) GWS solution; (b) GMP solution.

with Fig. 1(a), the linear basis GWS solution, Fig. 6(a), remains very dispersive with an error extremum of 22%. For the data selection in (32), the GMP algorithm is only fourth-order accurate, hence it produces the dispersive solution shown in Fig. 6(b). However, using $\beta = \gamma = 2$, which yields tenth-order accuracy, the GMP solution is again nodally exact to a lower order ($\sim 10^{-8}$) in round-off, Fig. 6(c). Unlike Fig. 1(b), this excellent non-unit Courant number GMP solution does not suffer from any artificial diffusion.

The most challenging definition for this problem is the minimum ($2\Delta x$ wide) initial wavelength square wave. Fig. 7 summarizes comparative solutions for this square wave propagation over 10 initial wavelengths at $C = 1$ and $C = 0.5$. Clearly, the linear basis GWS solution is totally distorted by dispersion error, with trailing wake and peak loss error of nearly 50%, Fig. 7(a). As predicted by the

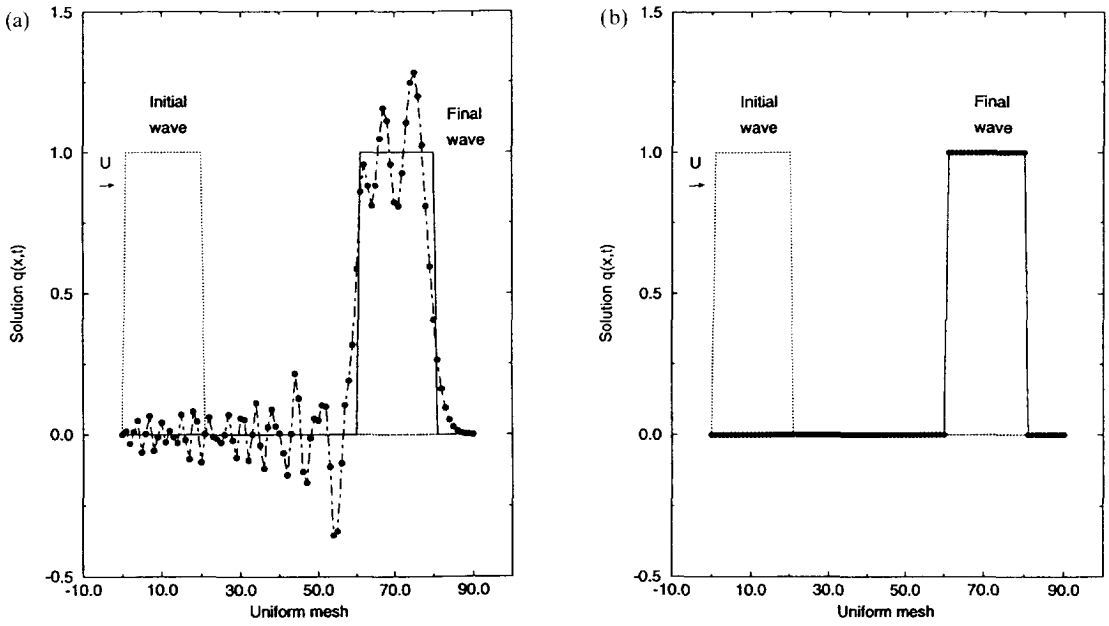

Fig. 5. 20-nodewide square wave propagation, three wavelengths at $C = 1$. (a) GWS solution; (b) GMP solution.

Fourier modal analysis (37), the corresponding eighteenth-order accurate GMP solution remains nodally exact (to order 10^{-16}) and dispersion-free, Fig. 7(b). The companion $C = 0.5$ solutions attest to GMP algorithm relative robustness, Fig. 8(a) and (b).

6.2. *Two-dimensional convection via a solid body rotation*

Originally developed for atmospheric dispersion prediction CFD assessments, this is an extremely informative non-diffusive 2-d verification test case for evaluating any algorithm. A finite difference (FD), a finite volume (FV), and several finite element (FE) algorithm tests are reported by [4], and a FE adaptive *h-p* mesh refinement strategy result is published by [5]. The solution to this unsteady, linear pure convection verification problem satisfies

$$\begin{aligned} &\frac{\partial q}{\partial t} + \boldsymbol{u} \cdot \nabla q = 0 \quad \text{in } \Omega \\ &q = 0 \qquad\qquad\quad \text{on } \partial\Omega \end{aligned} \tag{62}$$

where the imposed solid body rotational velocity field is $\boldsymbol{u} = r\dot{\theta}\hat{e}_\theta$. The 'rotating cone' initial condition is an elevated cosine-hill, see Fig. 9(a), with specification

$$q_0 = \begin{Bmatrix} \frac{1}{2}(1 + \cos 4\rho\pi) & \text{if } \rho \leqslant 0.125 \\ 0 & \text{otherwise} \end{Bmatrix} \tag{63}$$

where $1 \leqslant j \leqslant 2$ and

$$\rho^2 = (x_j - x_j^c)^2 \quad 0 \leqslant x_j \leqslant 1 \tag{64}$$

In (64), x_j^c denotes the mesh centroidal coordinate of the axi-symmetrical initial distribution. The analytical solution is again exact preservation of the initial condition, as the distribution is circularly convected around the z-plane.

The full-rotation solutions obtained on a uniform 33×33 node mesh for $k = 1$ and $k = 2$ FE GWS algorithms, using the (non-diffusive) trapezoidal time integration scheme and for $C = 1.0$ at the distribution centroid are shown in Fig. 9(b) and (c). Both GWS solutions suffer large dispersive error

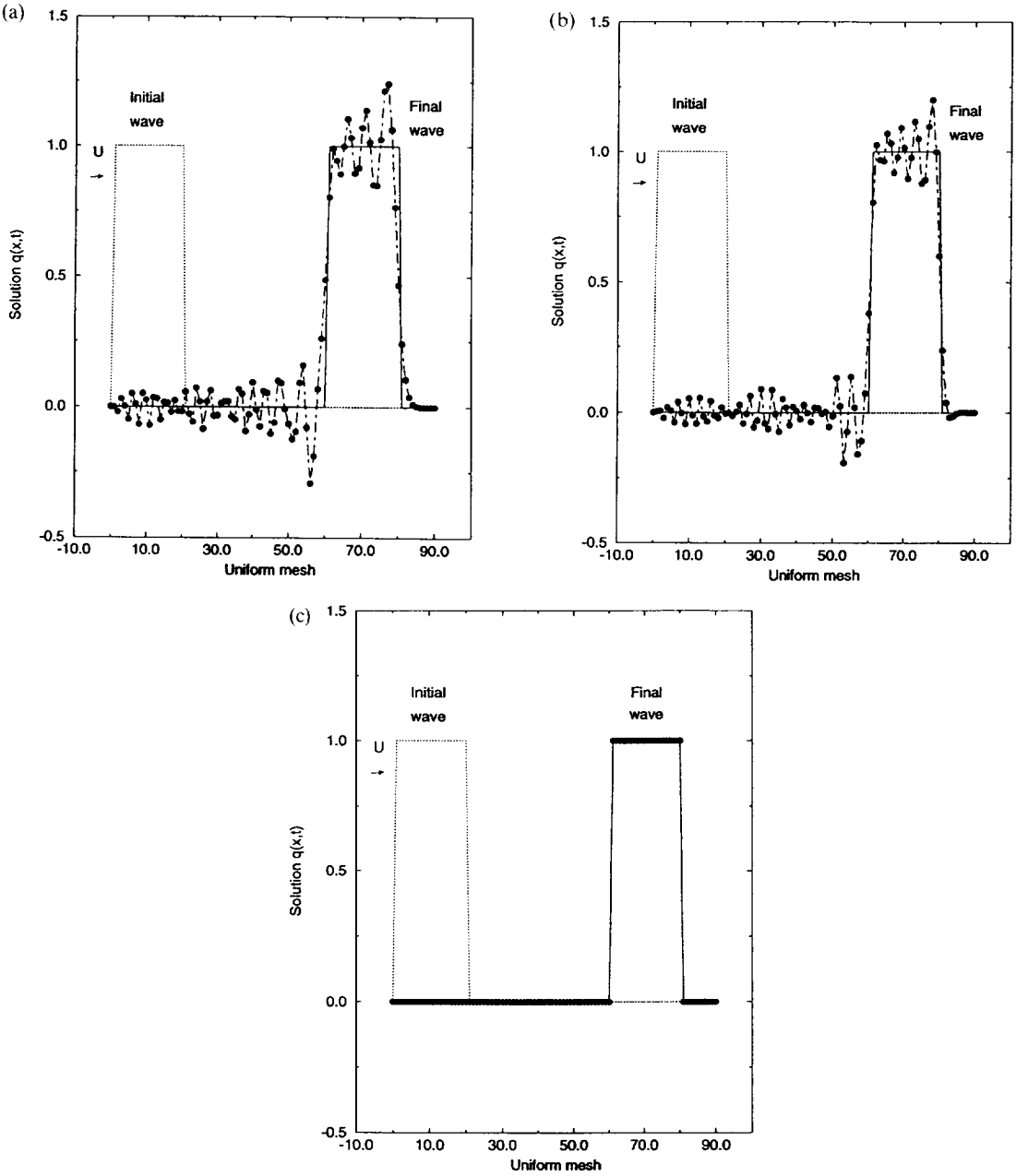

Fig. 6. 20-nodewide square wave propagation, three wavelengths at $C = 0.5$. (a) GWS solution; (b) GMP solution ($\beta = \gamma = 0$); (c) GMP solution ($\beta = \gamma = 2$).

accumulation on this mesh at this large Courant number. The $k = 1$ FE solution has trailing wake error extremum exceeding 25%, Fig. 9(b), and the solution peak is reduced by nearly 30% and severely lags behind the correct location. Although an improvement in the peak level (~5%) accrues to the quadratic basis solution, Fig. 9(c), the dispersion error extremum is nominally worsened by 3%. Therefore, increasing basis degree k of an FE implementation is of marginal impact for this problem.

For comparison, the Crank–Nicolson finite difference (CNFD) solution, Fig. 9(d), after one full revolution is poorer than either FE result, with solution peak level loss to 45% and a trailing wake error extremum of 28%. The 'best appearing' solution in the literature is the h-p adaptive FE GWS mesh solution obtained after 400 timesteps for one full revolution [5], reproduced as Fig. 9(e). The surface perspective plot appears smooth, but nearly 2% of the peak level is lost even for this low Courant

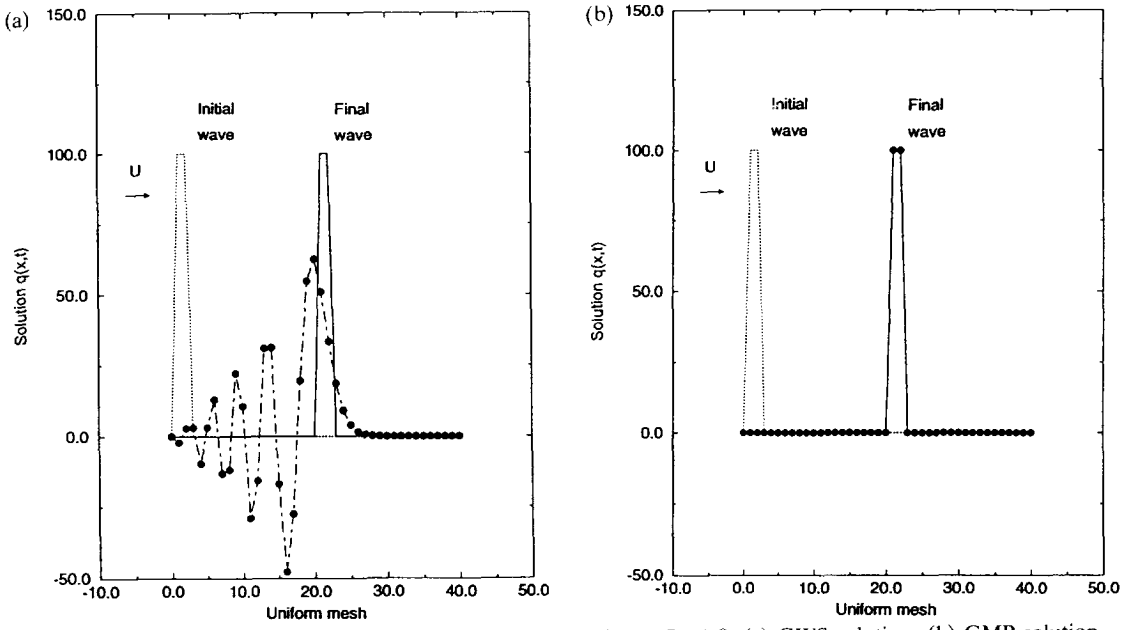

Fig. 7. 2-nodewide square wave propagation, ten wavelengths at $C = 1.0$. (a) GWS solution; (b) GMP solution.

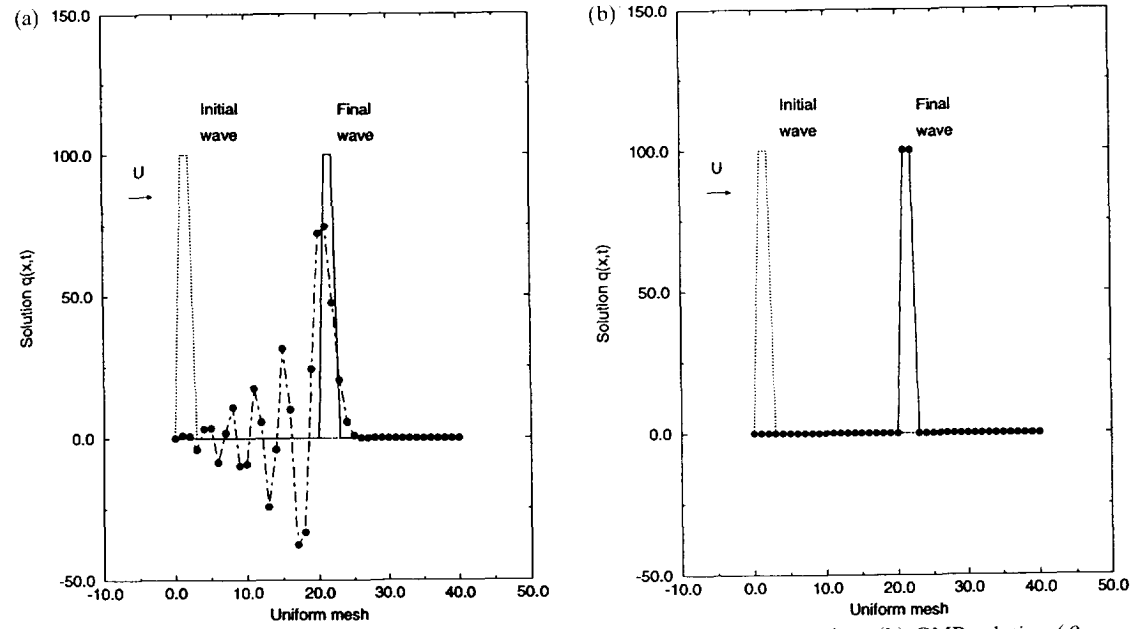

Fig. 8. 2-nodewide square wave propagation, ten wavelengths at $C = 0.5$. (a) GWS solution; (b) GMP solution ($\beta = \gamma = 2$).

number ($C \sim 1/6$). No results are reported for unit Courant number, which severely aggravates short wavelength error oscillation creation throughout the entire solution, cf. Fig. 9(b) and (d).

In distinction to each of these solutions, using the time-splitting technique (53)–(60b), the linear basis modified weak statement GMP solution, for $C_e = 1$ after one revolution, yields the nodally exact solution with absolutely no dispersion error or phase lag (to round-off, order 10^{-4}, Fig. 9(f)). The GMP method utilizes 32 timesteps to complete one full revolution. Further, the peak travelled the correct distance, and the solution is certainly the 'best' among those available for comparison. The mesh independence of the GMP weak statement formulation (50) for $C_e = 1$ is also verified on a 17×17 node

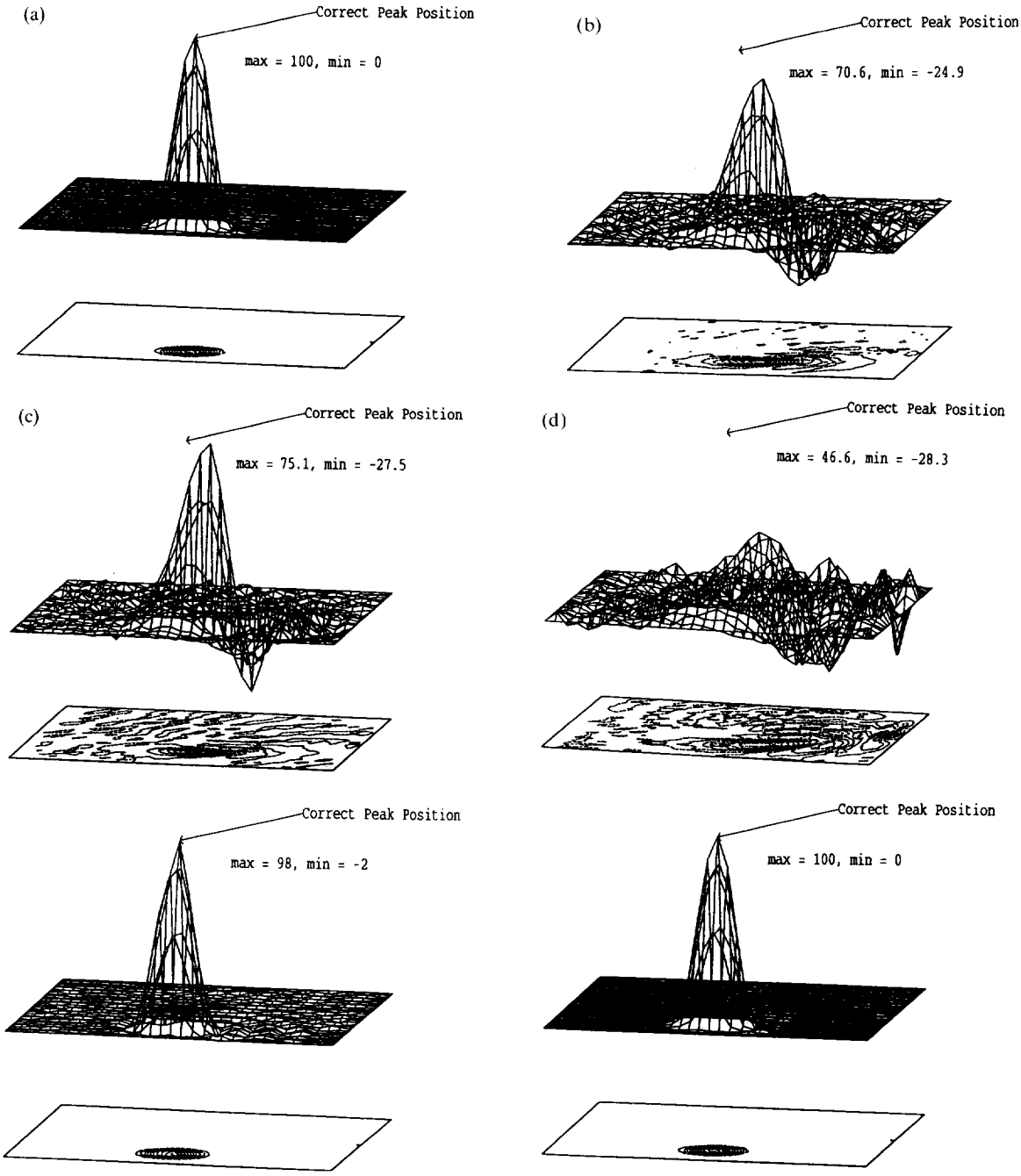

Fig. 9. Rotating cone solutions, one revolution on a 33×33 mesh. (a) Initial condition and exact final solution; (b) $k = 1$ FE GWS, $C = 1$; (c) $k = 2$ FE GWS, $C = 1$; (d) Crank–Nicolson FD, $C = 1$; (e) h-p FE, $C \sim 1/6$, $\Delta t = 2\pi/400$; (f) GMP solution, $C_e = 1$, $\Delta t = 2\pi/32$.

uniform mesh using $\Delta t = 2\pi/16$. The solution after one full revolution, Fig. 10, confirms exact preservation of the initial condition without dispersion error or phase lag to round-off (order 10^{-4}). Via (48), the GMP mesh measure becomes directly proportional to the solution time-step for a particular C_e in (50). Hence, for a large timestep, the GMP algorithm will produce a nodally accurate (to round-off) solution only on a very coarse mesh.

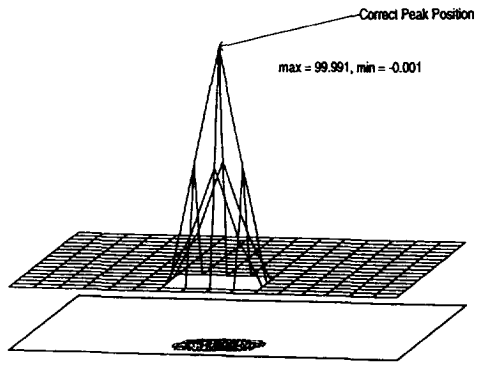

Fig. 10. Rotating cone GMP solution on a 17^2 mesh, after one revolution at $C_e = 1$, $\Delta t = (2\pi)/16$.

7. Conclusions

A weak statement perturbation technique based on Fourier modal analysis of the approximation solution amplification factor has been developed for generating progressively higher-order accurate CFD algorithm constructions for non-diffusive (pure convection) applications. The GMP theory is complete and easy to incorporate in one dimension, or in two dimensions using time-splitting. It exhibits the efficiency of a strictly linear basis FE GWS (or centered FD) algorithm. The approximation solution process for the modified weak statement, using a local element Courant number definition is mesh and time-step independent.

Acknowledgments

This work has been partially supported by grant MSS-9015912 from the National Science Foundation. The authors gratefully acknowledge the support provided by the UT CFD Laboratory and its corporate sponsors. The authors also acknowledge the helpful suggestions from Dr. S.B. Mulay and Mr. D.J. Chaffin.

Appendix A

Consider the standard FE linear basis one-dimensional matrices [A200] and [A201] in (13), i.e.

$$[M]_e = \Delta x_e[\mathrm{A200L}] = \frac{\Delta x_e}{6}\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}_{2\times 2}$$
$$[U]_e = U[\mathrm{A201L}] = \frac{U}{2}\begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix}_{2\times 2} \tag{A.1}$$

The corresponding amplification factor (18) for algorithm (12a–c), (24) becomes

$$G = \frac{\left(\frac{1}{6} + (1-\theta)C\frac{1}{2}\right)e^{-im} + \frac{4}{6} + \left(\frac{1}{6} - (1-\theta)C\frac{1}{2}\right)e^{im}}{\left(\frac{1}{6} - \theta C\frac{1}{2}\right)e^{-im} + \frac{4}{6} + \left(\frac{1}{6} + \theta C\frac{1}{2}\right)e^{im}} \tag{A.2}$$

In general, for linear basis FE and FD, via (A.1), Eq. (A.2) can be written as

$$f(z) = \frac{a_1 + a_2 z + a_3 z^2}{b_1 + b_2 z + b_3 z^2} \tag{A.3}$$

where, a_i and b_i, $1 \leq i$, are approximation coefficients and $z = e^{im}$.

DEFINITION: If $p(x) = \Sigma_{i \geq k} c_i x^i$ is a Laurent series (i.e. k could be a negative integer) which is not identically zero then, by the *order* of $p(x)$ we mean the *least* index i such that $c_i \neq 0$. We denote the order of $p(x)$ by $\mathcal{O}(p(x))$. For example, if $p(x) = -3x^3 + x^4 - x^5 + \cdots$ then $\mathcal{O}(p(x)) = 3$, if $p(x) = 5x^{-2} + 1 + x^7 + \cdots$ then $\mathcal{O}(p(x)) = -2$, if $p(x) = 1 - x + x^5 + \cdots$ then $\mathcal{O}(p(x)) = 0$, etc.

If $\mathcal{O}(p(x)) = r$ then one can write $p(x) = x^r q(x)$, where $q(x)$ is a power-series (i.e. non-negative components only) of order zero. For example $p(x) = x^{-2}(5 + x^2 + x^9 + \cdots)$ in the second example above.

If $q(x)$ is a power-series of order zero, then its reciprocal is also a power-series of order zero. Therefore, if $p(x)$ is of order r then its reciprocal is $x^{-r} q^*(x)$, where $q^*(x)$ denotes the reciprocal of $q(x)$. Also, if $\sigma(x)$ is another Laurent series of order s, then the order of $p(x)\sigma(x)$ is exactly $r + s$, and the order of $p(x)/\sigma(x)$ is exactly $r - s$.

Note that if the Taylor expansions of $f(e^{im})$ in (A.3) and e^{imC} match up to (and including) the $(r-1)$th power of x then, the order of $e^{imC} - f(e^{im})$ will be $\geq r$ ([11]).

THEOREM. Assume that $C \neq 0, \pm 1, \pm 2$. Then, for (A.3) there does not exist any value of a_j, b_j, such that $\mathcal{O}(e^{imC} - f(e^{im})) \geq 6$. In other words, for any choice of function $f(z)$ we must have $\mathcal{O}(e^{imC} - f(e^{im})) \leq 5$.

PROOF. We have

$$e^{imC} - f(e^{im}) = \frac{b_1 e^{imC} + b_2 e^{im(C+1)} + b_3 e^{im(C+2)} - a_1 - a_2 e^{im} - a_3 e^{i2m}}{b_1 + b_2 e^{im} + b_3 e^{i2m}} = \frac{N(m)}{D(m)} \tag{A.4}$$

where $N(m)$ and $D(m)$ denotes the numerator and the denominator. Clearly, both $N(m)$ and $D(m)$ are power series, in particular, $\mathcal{O}(D(m)) \geq 0$. Since $\mathcal{O}(e^{imC} - f(e^{im})) = \mathcal{O}(N(m)) - \mathcal{O}(D(m))$, it follows that $\mathcal{O}(e^{imC} - f(e^{im})) \leq \mathcal{O}(N(m))$.

Also, we can see that

$$N(x) = (a_1 + a_2 + a_3 - b_1 - b_2 - b_3) + \sum_{i \geq 1} \frac{i^n}{n!} u_n m^n \tag{A.5}$$

where $u_n = a_1 C^n + a_2 (C+1)^n + a_3 (C+2)^n - b_2 - 2^n b_3$.

Now suppose that $\mathcal{O}(N(x)) \geq 6$. Then, we must have $u_1 = \cdots = u_5 = 0$, as well as $(a_1 + a_2 + a_3 - b_1 - b_2 - b_3) = 0$. Now, $u_1 = \cdots = u_5 = 0$ yields five homogeneous linear equations in a_1, a_2, a_3, b_2, b_3. The determinant of this system can be easily computed. Up to a sign, via (44), it turns out to be $4(C-2)(C-1)^2 C^3 (C+1)^2 (C+2)$. Since, by assumption $C \neq 0, \pm 1, \pm 2$, this determinant is non-zero, forcing $a_1 = a_2 = b_2 = b_3 = 0$. But then $b_1 = 0$ too. This is obviously absurd. In conclusion, we must always have $\mathcal{O}(N(x)) \leq 5$.

Consequently, for (A.3), $\mathcal{O}(e^{imC} - f(e^{im}))$ can never exceed 5. □

For the GMP method (19)–(24), Eq. (A.3) for algorithm (12a–c), is of the form

$$\mathcal{F}(z) = \frac{a_1 + a_2 z + \cdots + a_P z^{P-1}}{b_1 + b_2 z + \cdots + b_P z^{P-1}} \tag{A.6}$$

where $P - 3 = 2 \times$ max degree $(\alpha, \beta, \gamma, \delta)$. Eq. (A.6) directly relates P to the width of the computational stencil. For example, $\gamma = 1$, $\beta = 1$, and $\alpha = \delta = 0$ in (21a–b), (A.3) takes the form

$$\mathcal{F}(z) = \frac{a_1 + a_2 z + a_3 z^3 + a_4 z^3 + a_5 z^4}{b_1 + b_2 z + b_3 z^2 + b_4 z^3 + b_5 z^4} \tag{A.7}$$

which is equivalent to the $P = 5$ pentadiagonal (quadratic FE basis, or five point centered FD) form. Again, for $\gamma = 2 = \beta$, (A.6) is equivalent to the $P = 9$, i.e. 9-wide diagonal (quartic FE basis, or nine point centered FD), and so on. It can also be shown by induction that

$$\mathcal{O}(e^{imC} - \mathcal{F}(e^{im})) \leq (2P - 1) \tag{A.8}$$

Appendix B

For $\alpha = \beta = \gamma = \delta = 2$, and Λ_1:la1,Λ_2:la2, Ψ_1:ps1, Ψ_2:ps2, Γ_1:ga1, Γ_2:ga2, Y_1:up1, Y_2:up2, a_1:a, a_2:b, in (21a–b) and (27), this appendix, generated using Macsyma (1985) symbolic logic program, verifies the analytical process leading to the determination of the algorithm order-of-accuracy.

Note: (a) 'gh' in (c21) is the same as G in (24),
(b) 'term1' in (c29) is A_1 in (28),
(c) 'term2' in (c30) is A_2 in (28),
(d) 'term3' in (c36) is A_3 in (28), 'term4' in (c42) is A_4 in (28),
(e) 'term5' in (c50) is A_5 in (28) and 'term6' in (c56) is A_6 in (28).

```
(c2) z:%e^(m*%i)$
(c3) x1:z^-1$
(c4) x2:z$
(c5) theta:1/2$
(c6) la1:t1*(1+a*z^-1+b*z^-2)$
(c7) ps1:t2*(1+a*z^1+b*z^2)$
(c8) ga1:t3*(1+a*z^-1+b*z^-2)$
(c9) up1:t4*(1+a*z^1+b*z^2)$
(c10) la2:t5*(1+a*z^-1+b*z^-2)$
(c11) ps2:t6*(1+a*z^1+b*z^2)$
(c12) ga2:t7*(1+a*z^-1+b*z^-2)$
(c13) up2:t8*(1+a*z^1+b*z^2)$
(c14) ga1:-(up1+la1+ps1)$
(c15) ga2:-(up2+la2+ps2)$
(c16) ght:((1-ga1)*x1+(4-la1-up1)+(1-ps1)*x2)/6%
(c17) ghx:((-ps2)*x2-(la2+up2)-(1+ga2)*x1)/2$
(c18) ghn:ght-(1-theta)*c*ghx$
(c19) ghd:ght+theta*c*ghx$
(c20) gh:ratsimp(ghn/ghd)$
(c21) th:taylor(gh,m,0,6)$
(c23) ta:taylor(z^-c,m,0,9)$
(c24) term1:coeff(th,m,1);
(d24) /r/ ((%i b+%i a+%i) c t8+(2 %i b+2 %i a+2 %i) c t6 + (%i b+%i a+%i) c
      t5-2 %i c)/2
(c25) ta1:coeff(ta,m,1);
(d25) /r/                                  -%i c
(c26) solve([term1-ta1],[t6]);
                                          t8+t5
(d26)                              [t6=-------]
                                           2
(c27) t6:rhs(%[1])$
(c28) th:ev(th,eval)$
(c29) term1:coeff(thm,1);
```

```
(d29) /r/                                   -%i c
(c30) term2:coeff(th,m,2);
(d30) /r/ ((3 b+3 a+3) c t8+(27 b+15 a+3) c t5+(2b+2 a+2) c t4
      + (4 b+4 a+4) c t2+(2 b+2 a+2) c t1-6c^2)/12
(c31) ta2:coeff(ta,m,2);
(d31) /r/                                   -c^2/2
(c32) solve([term2-ta2],[t5]);
(d32) [t5=-((3 b+3 a+3) t8+(2 b+2 a+2) t4
          +(4 b+4 a+4) t2
          +(2 b+2 a+2) t1)/(27 b+15 a+3)]
(c33) t5:rhs(%[1])$
(c34) th:ev(th,eval)$
(c35) term2:coeff(th,m,2);
(d35) /r/                                   -c^2/2
(c36) term3:coeff(th,m,3);
(d36) /r/ ((60 %i b^2+(66 %i a+12 %i) b+18 %i a^2+6 %i a) c t8
      +(31 %i b^2+(30 %i a-2 %i) b+7 %i a^2-2%i a-%i) c t4
      +(80 %ib^2+(88 %i a+16 %i) b+24 %i a^2+8 %i a) c t2
      +(-41 %i b^2+(-46 %i a-10 %i) b-13 %i a^2-6 %i a-%i) c t1
      +(27 %i b+15 %i a+3 %i) c^3)/(108 b+60 a+12)
(c37) ta3:coeff(ta,m,3);                    %i c^3/6
(d37) /r/
(c38) solve([term3-ta3],[68]);
(d38) [t8=-((31 b^2+(30 a-2) b+7 a^2-2 a-1) t4
      +(80 b^2+(88 a+16) b+24 a^2+8 a) t2
      +(-41 b+(-46 a-10) b-13 a^26 a-1) t1+
      (9 b+5 a+1) c^2)
      /(60 b^2+(66 a+12) b+18 a^2+6 a)]
(c39) t8:rhs(%[1])$
(c40) th:ev(th,eval)$
(c41) term3:coeff(th,m,3);                  %i c^3/6
(d41) /r/
(c42) term4:coeff(th, m, 4);
(d42) /r/ (((10 b^3+(21 a+12) b^2+(14 a^2+14 a+2) b
                                  +3 a^3+4 a^2+a) c^3
      +(80 b^3+126 a b^2+(70 a^2+16 a+16) b+12 a^3+2 a^2+2 a) c) t4
      +((20 b^3+(42 a+24) b^2+(28 a^2+28 a+4) b
                              +6 a^3+8 a^2+2 a) c^3
      +(-20 b^3+(-42 a-24) b^2+(-28 a^2-28 a-4) b
                               -6 a^3-8 a^2-2 a) c)
      t2+((10 b^3+(21 a+12) b^2+(14 a^2+14 a+2) b+3 a^3+4 a^2+a) c^3
      +(-400 b^3+(-498 a+96) b^2+(-218 a^2+64 a+16) b-36 a^3+2 a^2
      +2 a) c) t1+(30 b^2+(33 a+6) b+9 a^2+3 a) c^4
      +(30 b^2+(15 a-18) b+3 a^2-3 a) c^3)
      /(720 b^2+(792 a+144) b+216 a^2+72 a)(c43) ta4:coeff(ta,m,4);
(d43) /r/                                   c^4/24
(c44) term4:coeff(th, m, 4);
(d45) /r/ (((10 b^3+(21 a+12) b^2+14 a^2+14 a+2) b+3 a^3+4 a^2+a) c^3
      +(80 b^3+126 a b^2+(70 a^2+16 a+16) b+12 a^3+2 a^2+2 a) c) t4
      +((20 b^3+(42 a+24) b^2+(28 a^2+28 a4) b+6 a^3+8 a^2+2 a) c^3
      +(-20 b^3+(-42 a-24) b^2+(-28 a^2-28 a-4) b-6 a^3-8 a^2-2 a) c)
      t2+((10 b^3+(21 a+12) b^2+(14 a^2+14 a+2) b+3 a^3+4 a^2+a) c^3
      +(-400 b^3+(-498 a+96) b^2+(-218 a^2+64 a+16) b-36 a^3+2 a^2
```

```
      + 2 a) c) t1+(30 b^2+(33 a+6) b+9 a^2+3 a) c^4
      + (30 b^2+(15 a-18) b+3 a^2-3 a) c^3)
      /(720 b^2+(792 a+144) b+216 a^2+72 a)
(c46) solve([term4-ta4], [t4]);
(d46) [t4=-((20 b^3+(42 a+24) b^2+(28 a^2+28 a+4) b+6 a^3+8 a^2
      + 2 a) c^2-20 b^3+(-42 a-24) b^2+(-28 a^2-28 a-4) b-6 a^3-8 a^2
      -2 a) t2+((10 b^3+(21 a+12) b^2+(14 a^2+14 a+2) b+3 a^3+4 a^2+a)
      c^2-400 b^3+(96-498 a) b^2+(-218 a^2+64 a+16) b-36 a^3+2 a^2+2 a)
      t1+(30 b^2+(15 a-18) b+3 a^2-3 a) c^2)
      /((10 b^3+(21 a+12) b^2+(14 a^2+14 a+2) b+3 a^3+4 a^2+a) c^2+80 b^3
      + 126 a b^2+(70 a^2+16 a+16) b+12 a^3+2 a^2+2 a)]
(c47) t4:rhs(%[1])$
(c48) th:ev(th,eval)$
(c49) term4:coeff(th,m,4);
(d49) /r/                                   c^4
                                            ---
                                            24
(c50) term5:coeff(th, m, 5);
(d50) /r/ (((50 %i b^4+(155 %i a+110 %i) b^3
      + (175 %i a^2+235 %i a+70 %i) b^2+(85 %i a^3+160 %i a^2+85 %i a+10 %i)
      b+15 %i a^4+35 %i a^3+25 %i a^2+5 %i a) c^5
      + (950 %i b^4+(1955 %i a-430 %i) b^3+(1675 %i a^2-5 %i a+610 %i) b^2
      + (625 %i a^3+100 %i a^2+445 %i a+70 %i) b+75 %i a^4-25 %i a^3+25 %i a^2
      + 5 %i a) c^3+(-1000 %i b^4+(-2110 %i a+320 %i) b^3
      + (-1850 %i a^2-230 %i a-680 %i) b^2
      + (-710 %i a^3-260 %i a^2-530 %i a-80 %i) b-90 %i a^4-10 %i a^3
      -50 %i a^2-10 %i a) c) t2+((-200 %i b^4+(-520 %i a-240 %i) b^3
      + (-490 %i a^2-400 %i a-40 %i) b^2+(-200 %i a^3-220 %i a^2-40 %i a) b
      - 30 %i a^4-40 %i a^3-10 %i a^2) c^5+(2200 %i b^4+(3080 %i a-1680 %i)b^3
      + (1550 %i a^2-2080 %i a-40 %i) b^2+(340 %i a^3-820 %i a^2-40 %i a) b
      + 30 %i a^4-100 %i a^3-10 %i a^2) c^3+(16000 %i b^4
      + (20840 %i a-12480 %i) b^3+(11540 %i a^2-9760 %i a+2240 %i) b^2
      + (3100 %i a^3-2920 %i a^2+440 %i a) b+360 %i a^4-220 %i a^3+20 %i a^2) c)
      t1+(-50 %i b^3+(105 %i a+420 %i) b^2+(65 %i a^2+260 %i a-10 %i) b
      + 25 %i a^2-5 %i a) c^5+(520 %i b^3+(1272 %i a+1164 %i) b^2
      + (443 %i a^2+428 %i a-76 %i) b+21 %i a^3+43 %i a^2-8 %i a) c^3
      + (160 %i b^3+252 %i a b^2+9140 %i a^2+32 %i a+32 %i) b+24 %i a^3
      + 4 %i a^2+4 %i a) c)/((3600 b^3+(7560 a+4320) b^2
      + (5040 a^2+5040 a+720) b+1080 a^3+1440 a^2+360 a) c^2+28800 b^3
      + 45360 a b^2+(25200 a^2+5760 a+5760) b+4320 a^3+720 a^2+720 a)
(c51) ta5:coeff(ta,m,5);
(d51) /r/                                   %i c^5
                                          - ------
                                            120
(c52) solve([term5-ta5], [t1]);
(d52) [t1+(((50 b^4+(155 a+110) b^3+(175 a^2+235 a+70) b^2
      + (85 a^3+160 a^2+85 a+10) b+15 a^4+35 a^3+25 a^2+5 a) c^4
      + (950 b^4+(1995 a-430) b^3+(1675 a^2-5 a+610) b^2
      + (625 a^3+100 a^2+445 a+70) b+75 a^4-25 a^3+25 a^2+5 a) c^2-1000 b^4
      + (320-2110 a) b^3+(-1850 a^2-230 a-680) b^2
      + (-710 a^3-260 a^2-530 a-80) b-90 a^4-10 a^3-50 a^2-10 a) t2
      + (30 b^3+(63 a+36) b^2+(42 a^2+42 a+6) b+9 a^3+12 a^2+3 a) c^6
      + (190 b^3+(483 a+420) b^2+(275 a^2+308 a+38) b+36 a^3+31 a^2+a) c^4
      + (520 b^3+(1272 a+1164) b^2+(443 a^2+428 a-76) b+21 a^3+43 a^2-8 a)
      c^2+160 b^3+252 a b^2+(140 a^2+32 a+32) b+24 a^3+4 a^2+4 a)
      /((200 b^4+(520 a+240) b^3+(490 a^2+400 a+40) b^2
      + (200 a^3+220 a^2+40 a) b+30 a^4+40 a^3+10 a^2) c^4
      + (-2200 b^4+(1680-3080 a) b^3+(-1550 a^2+2080 a+40) b^2
```

```
      +(-340 a^3+820 a^2+40 a) b-30 a^4+100 a^3+10 a^2) c^2-16000 b^4
      +(12480-20840 a) b^3+(-11540 a^2+9760 a-2240) b^2
      +(-3100 a^3+2920 a^2-440 a) b-360 a^4+220 a^3-20 a^2)]
(c53) t1:rhs(%[1])$
(c54) th:ev(th,eval)$
(c55) term5:coeff(th,m,5);
                                   %i c^5
(d55) /r/                        - ------
                                    120
(c56) term6:coeff(th,m,6);
(d56) /r/ (((500 b^5+(1600 a+700) b^4+(2005 a^2+1620 a+200) b^3
      +(1235 a^3+1395 a^2+360 a+20) b^2+(375 a^4+530 a^3+195 a^2+20 a) b
      +45 a^5+75 a^4+35 a^3+5 a^2) c^7+(-21000 b^5+(-51000 a-7800) b^4
      +(-48030 a^2-6480 a+10920) b^3+(-23250 a^3-4770 a^2+7560 a-2280) b^2
      +(-5850 a^4-2100 a^3+2430 a^2-480 a) b-630 a^5-450 a^4+150 a^3-30 a^2)
      c^5+(220500 b^5+(349200 a-252900) b^4+(268245 a^2-230220 a+102780) b^3
      +(110115 a^3-99045 a^2+57240 a+180) b^2
      +(23175 a^4-22230 a^3+7155 a^2+180 a) b+2205 a^5-1125 a^4+315 a^3
      +45 a^2) c^3+(-200000 b^5+(-299800 a+260000) b^4
      +(-222220 a^2+235080 a-113920) b^3+(-88100 a^3+102420 a^2-65160 a
      +2080) b^2+(-17700 a^4+23800 a^3-9780 a^2+280 a) b-1620 a^5+1500 a^4
      -500 a^3-20 a^2) c) t2+(-100 b^4+(-260 a-120) b^3
      +(-245 a^2-200 a-20) b^2+(-100 a^3-110 a^2-20 a) b-15 a^4-20 a^3
      -5 a^2) c^10+(120 b^4+(276 a+96) b^3+(228 a^2+120 a-24) b^2
      +(84 a^3+60 a^2-12 a) b+12 a^4+12 a^3) c^9
      +(11000 b^4+(1540 a-840) b^3+(775 a^2-1040 a-20) b^2
      +(170 a^3-410 a^2-20 a) b+15 a^4-50 a^3-5 a^2) c^8
      +(680 b^4+(1210 a+1560) b^3+(145 a^2+280 a-1040) b^2
      +(-175 a^3+40 a^2-170 a) b-60 a^4-35 a^3-5 a^2) c^7
      +(8000 b^4+(10420 a-6240) b^3+(5770 a^2-4880 a+1120) b^2
      +(1550 a^3-1460 a^2+220 a) b+180 a^4-110 a^3+10 a^2) c^6
      +(-37320 b^4+(-23472 a+26808) b^3+(-5856 a^2+9120 a+768) b^2
      +(-738 a^3-690 a^2+264 a) b+66 a^4-84 a^3+30 a^2) c^5
      +(-68280 b^4+(90 a+77040) b^3+(3255 a^2+10320 a-1560) b^2
      +(225 a^3+1110 a^2-330 a) b-90 a^4+135 a^3-45 a^2) c^3
      +(-3200 b^4+(296 a+2496) b^3+(2228 a^2+1760 a+1856) b^2
      +(604 a^3-520 a^2+248 a) b+72 a^4-28 a^3+20 a^2) c)
      /((72000 b^4+(187200 a+86400) b^3+(176400 a^2+144000 a+14400) b^2
      +(72000 a^3+79200 a^2+14400 a) b+10800 a^4+14400 a^3+3600 a^2) c^4
      +(-792000 b^4+(-1108800 a+604800) b^3
      +(-558000 a^2+748800 a+14400) b^2+(-122400 a^3+295200 a^2+14400 a) b
      -10800 a^4+36000 a^3+3600 a^2) c^2-5760000 b^4+(-7502400 a+4492800) b^3
      +(-4154400 a^2+3513600 a-806400) b^2
      +(-1116000 a^3+1051200 a^2-158400 a) b-129600 a^4+79200 a^3-7200 a^2)
(c57) ta6:coeff(ta,m,6);
                                    c^6
(d57) /r/                         - ---
                                    720
(c60) solve([term6-ta6], [t2]);
(d60) [t2=-((120 b^4+(276 a+96) b^3+(228 a^2+120 a-24) b^2
      +(84 a^3+60 a^2-12 a) b+12 a^4+12 a^3) c^8
      +(680 b^4+(1210 a+1560) b^3+(145 a^2+280 a-1040) b^2
      +(-175 a^3+40 a^2-170 a) b-60 a^4-35 a^3-5 a^2) c^6
      +(-37320 b^4+(26808-23472 a) b^3+(-5856 a^2+9120 a+768) b^2
      +(-738 a^3-690 a^2+264 a) b+66 a^4-84 a^3+30 a^2) c^4
      +(-68280 b^4+(90 a+77040) b^3+(3255 a^2+10320 a-1560) b^2
      +(225 a^3+1110 a^2-330 a) b-90 a^4+135 a^3-45 a^2) c^2-3200 b^4
      +(296 a+2496) b+(2228 a^2+1760 a+1856) b^2
```

```
      +(604 a^3-520 a^2+248 a) b+72 a^4-28 a^3+20 a^2)
      /((500 b^5+(1600 a+700) b^4+(2005 a^2+1620 a+220) b^3
      +(1235 a^3+1395 a^2+360 a+20) b^2+(375 a^4+530 a^3+195 a^2+20 a) b
      +45 a^5+75 a^4+35 a^3+5 a^2) c^6+(-21000 b^5+(-51000 a-7800) b^4
      +(-48030 a^2-6480 a+10920) b^3+(-23250 a^3-4770 a^2+7560 a-2280) b^2
      +(-5850 a^4-2100 a^3+2430 a^2-480 a) b-630 a^5-450 a^4+150 a^3-30 a^2)
      c^4+(220500 b^5+(349200 a-252900) b^4+(268245 a^2-230220 a+102780) b^3
      +(110115 a^3-99045 a^2+57240 a+180) b^2
      +(23175 a^4-22230 a^3+7155 a^2+180 a) b+2205 a^5-1125 a^4+315 a^3
      +45 a^2) c^2-20000 b^5+(260000-299800 a) b^4
      +(-222220 a^2+235080 a-113920) b^3+(-88100 a^3+102420 a^2-65160 a
      +2080) b^2+(-17700 a^4+23800 a^3-9780 a^2+280 a) b-1620 a^5+1500 a^4
      -500 a^3-20 a^2)]
(c61) t2:rhs(%[1])$
(c62) th:ev(th,eval);
(d62) /r/
      c^6 m^6+6 %i c^5 m^5-30 c^4 m^4-120 %i c^3 m^3+360 c^2 m^2+720 %i c m-720
      ----------------------------------------------------------------------
                                        720
(c63) term6:coef(th,m,6);              c^6
(d63) /r/                             ---
(c65) quit();                         720
```

Finally, in (d62) the approximation solution G of (28), expanded in Laurent series up to and including the sixth term, for particular choices of t_i matches exactly up to the sixth term of G_a in (11). Hence, the GMP algorithm for $\beta = 2 = \gamma$ is *at least* sixth-order accurate for any a_1, a_2.

Appendix C

Via weak statement algorithm (12a–c), the θ-implicit relation for (61) is

$$([M] + \theta\,\Delta t[U])\{Q^{n+1}\} - ([M] + \theta\,\Delta t[U])\{Q^n\} + \Delta t[U]\{Q^n\} = \{0\} \tag{C.1}$$

In one-dimensional finite element form, for $C = 1$ and $\theta = 1/2$, (C.1) yields

$$\left(\frac{1}{4}\begin{bmatrix}1 & 1\\ 1 & 1\end{bmatrix} + \frac{1}{4}\begin{bmatrix}-1 & 1\\ -1 & 1\end{bmatrix}\right)\{Q^{n+1}\} = \left(\frac{1}{4}\begin{bmatrix}1 & 1\\ 1 & 1\end{bmatrix} - \frac{1}{4}\begin{bmatrix}-1 & 1\\ -1 & 1\end{bmatrix}\right)\{Q^n\} \tag{C.2}$$

Hence, the global assembly form of (C.2) is

$$\begin{bmatrix}0 & 1 & & & & \\ 0 & 1 & 1 & & & \\ & 0 & 1 & 1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & 0 & 1 & 1\\ & & & & 0 & 1\end{bmatrix}\begin{Bmatrix}Q_1\\ Q_2\\ Q_3\\ \vdots\\ Q_M\\ Q_{M+1}\end{Bmatrix}^{n+1} = \begin{bmatrix}1 & 0 & & & & \\ 1 & 1 & 0 & & & \\ & 1 & 1 & 0 & & \\ & & \ddots & \ddots & \ddots & \\ & & & 1 & 1 & 0\\ & & & & 1 & 0\end{bmatrix}\begin{Bmatrix}Q_1\\ Q_2\\ Q_3\\ \vdots\\ Q_M\\ Q_{M+1}\end{Bmatrix}^{n} \tag{C.3}$$

It is easy to see that the solution of (C.3) is

$$Q_j^{n+1} = Q_{j-1}^n, \text{ i.e. } \quad \text{if } \{Q^n\} = \{0, a, b, 0, 0, \ldots, 0, c, 0\}^{\mathrm{T}}$$
$$\text{then } \{Q^{n+1}\} = \{0, 0, a, b, 0, \ldots, 0, 0, c\}^{\mathrm{T}} \tag{C.4}$$

which is exact for $C = 1$.

Similarly, for $C = 2$, the exact one-dimensional solution is

$$Q_j^{n+1} = Q_{j-2}^n, \text{ i.e. } \quad \text{if } \{Q^n\} = \{0, a, b, 0, 0, \ldots, 0, c, 0, 0\}^{\mathrm{T}}$$
$$\text{then } \{Q^{n+1}\} = \{0, 0, 0, a, b, \ldots, 0, 0, 0, c\}^{\mathrm{T}} \tag{C.5}$$

This is not possible to extract from the tridiagonal matrix form (C.3), since the residual matrix (right-hand side) has to be at least four-wide, i.e.

$$\begin{bmatrix} 0 & 1 & & & & \\ 0 & 1 & 1 & & & \\ & 0 & 1 & 1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & 0 & 1 & 1 \\ & & & & 0 & 1 \end{bmatrix} \begin{Bmatrix} Q_1 \\ Q_2 \\ Q_3 \\ \vdots \\ Q_M \\ Q_{M+1} \end{Bmatrix}^{n+1} = \begin{bmatrix} 1 & 0 & 0 & & & \\ 1 & 1 & 0 & 0 & & \\ & 1 & 1 & 0 & 0 & \\ & & \ddots & \ddots & \ddots & \\ & & & 1 & 1 & 0 & 0 \\ & & & & 1 & 0 & 0 \end{bmatrix} \begin{Bmatrix} Q_1 \\ Q_2 \\ Q_3 \\ \vdots \\ Q_M \\ Q_{M+1} \end{Bmatrix}^{n} \tag{C.6}$$

such that (C.5) can be satisfied. This is impossible to achieve within the context of linear basis (tridiagonal) matrix equations.

References

[1] A.J. Baker, Finite Element Computational Fluid Mechanics (Hemisphere Publishers Co., New York, 1983).
[2] A.J. Baker and D.W. Pepper, Finite Elements 1-2-3 (McGraw-Hill, New York, 1991).
[3] A.J. Baker and J.W. Kim, Int. J. Numer. Methods Fluids 7 (1987) 489–520.
[4] A.J. Baker, P.T. Williams, D.J. Chaffin and S. Roy, in: Sam S.Y. Wang, ed., Advances in Hydro-Science and Engineering, Vol. 1 (1993) 32–45.
[5] L. Demkowicz, J.T. Oden, W. Rachowicz and O. Hardy, Toward a universal *h-p* adaptive finite element strategy, Part 1. Constrained approximation and data structure, Comput. Methods Appl. Mech. Engrg. 77 (1989) 79–112.
[6] Z. Haras and S. Ta'asan, Finite difference schemes for long-time integration, NASA CR-191471, ICASE Report No. 93-25, 1993.
[7] C. Hirsch, Numerical Computation of Internal and External Flows, Vol. 2, Computational Methods for Inviscid and Viscous Flows (John Wiley & Sons, Chichester, 1992).
[8] H.O. Kreiss and J. Lorenz, Initial-Boundary Value Problems and the Navier–Stokes Equations (Academic Press, New York, 1989).
[9] P.D. Lax, Weak solutions of nonlinear hyperbolic equations and their numerical computation, Commun. Pure Appl. Math. 7 (1954) 159–193.
[10] S.K. Lele, Compact finite difference schemes with spectral-like resolution, J. Comput. Phys. 103 (1992) 16–42.
[11] S.B. Mulay, Private communications, 1993.
[12] R. Vichenevetsy and J.B. Bowles, Fourier analysis of numerical approximations of hyperbolic equations, SIAM (1982) 140 pp.