Application of Continuous Dynamic Grid Adaption Techniques to Meteorological Modeling. Part I: Basic Formulation and Accuracy

GARY S. DIETACHMAYER
Bureau of Meteorology Research Centre, Melbourne, Australia

KELVIN K. DROEGEMEIER
School of Meteorology, University of Oklahoma, Norman, Oklahoma

(Manuscript received 2 July 1990, in final form 6 August 1991)

ABSTRACT

The continuous dynamic grid adaption (CDGA) technique developed in astrophysics and aeronautics is applied to our knowledge, for the first time to meteorological modeling. The aim of CDGA is to improve the accuracy of numerical solutions of partial differential equations (typically those governing fluid flow) by the use of nonuniform grids that have higher local resolution in regions where the numerical error is presumed to be large. Conceptually, CDGA has some relationship to the well-known technique of grid stretching, but its power lies in its ability to determine an appropriate spatial distribution of grid points automatically and to update this distribution in response to changes in the evolving numerical solution. Application of the technique is facilitated by transforming the governing equations from physical space in which the grid is nonuniform, nonorthogonal and for which the individual grid points are in continuous motion to computational space, which by definition has both a regular and stationary distribution of grid points. The distribution of grid points is found by the solution of “grid-generator” equations, which in turn can be derived as a weighted combination of several variational problems, each of which attempts to enforce a particular desirable property of the grid. These properties include the smoothness and orthogonality of the gridpoint distribution and its response to the user-defined “weight function,” which is a quantitative measure of where the local resolution is to be increased.

The method is applied to several problems of meteorological relevance. The first, Burgers’ equation in one dimension, is used primarily to illustrate the method in a simple context, but also illuminates several features of CDGA, one of which is its ability to improve the accuracy of a numerical solution purely by inducing motion of the grid points. A kinematic frontogenesis problem is used to extend the method to two dimensions, and with the aid of a readily available exact solution, shows the very considerable gains in accuracy that may be achieved over fixed-grid methods. A surprising observation is that the formal order of accuracy of the adaptive results is, for certain parameters, actually greater than for the fixed-grid results. The ability of the technique to allocate multiple zones of high resolution is demonstrated by experiments in which several (four) “cones” are advected by a field of solid-body rotation. The final application is to the evolution of a slab-symmetric thermal front in a neutral environment. Again, considerable improvements in accuracy over fixed-grid calculations are achieved, and it is shown that the problem of spurious numerical oscillations associated with rapid variation in an advected field, a problem that has received a great deal of attention in recent times, is greatly alleviated by the CDGA formulation.

1. Introduction

The accurate representation of multiscale events in numerical models has long been a principal issue in computational fluid dynamics. For example, one typically desires to capture not only the development and evolution of small-scale features but also their interaction with and influence upon the larger-scale flow. This is a particularly important requirement in atmospheric models because numerous events such as fronts, clouds, and tornadoes are not only relatively localized with respect to their environment, but are also forced on scales larger than their own. Because practical limitations in computer size and speed prohibit the use of uniformly high spatial resolution appropriate for the smallest scales of interest, numerous techniques have been developed to deal with multiscale flows. The fundamental link between many of these methods is the notion that an appropriate choice of the numerical grid or grids plays a key role in determining the quality of the computed solution.

Meteorological problems requiring grid systems more powerful than a single, uniform, Eulerian mesh are typically solved using transformed coordinates, grid-nesting techniques, or grid-refinement techniques.

© 1992 American Meteorological Society
(we will define our discussion here to Eulerian methods). The first coordinate transformations in meteorological models were applied only in the vertical, and utilized a normalized pressure function (e.g., Phillips 1957). This methodology was designed to accommodate irregular lower boundaries and to provide increased vertical resolution near the ground in large-scale hydrostatic models. Such methods have since been extended to terrain-following systems in nonhydrostatic frameworks (e.g., Miller 1974; Gal-Chen and Somerville 1975), and most recently have been applied simultaneously in all coordinate directions (Sharman et al. 1988). Anthes (1970) introduced a nonlinear grid transformation function that maps nonuniform physical coordinates into orthogonal, uniform computational coordinates. This mapping, commonly known as grid stretching, provides increased spatial resolution in selected regions of the computational domain, and has been widely applied in convective and mesoscale modeling studies (e.g., Wilkinson and Chen 1982). A principal limitation of grid stretching and other static coordinate transformations is that one must know a priori, and for the duration of the calculation, which regions of the domain will require high spatial resolution.

Grid-nesting techniques involve the sequential placement of multiple finer-scale meshes in desired regions of the domain so as to provide increased spatial resolution locally (see the review by Koch and McQueen 1987). Although the decision to spawn one or more submeshes is typically subjective and manually directed (see discussion below for a counterexample), many formulations allow the submeshes to move with particular features in the flow, such as a hurricane (e.g., Jones 1977). The interaction among multiple nested meshes, particularly the tendency for propagating dispersive waves to discontinuously change their speeds upon passing from one mesh to the next and to reflect off the boundaries of each nest, has been a topic of great concern during the past two decades. Despite these limitations, nested-grid techniques have found widespread use in both research (e.g., Clark and Farley 1984; Zhang et al. 1986) and operational (e.g., Phillips 1979) models.

Grid-refinement techniques are a relatively new and powerful class of methods that include both grid transformation and nesting strategies, and can be subdivided into two basic categories. The first includes methods in which grid points or collocation points are added to the computational domain as the calculation proceeds. This can be performed in one of two ways. In the first, grid points are added locally (e.g., Dannenhoffer and Baron 1986), or finite elements are subdivided locally (e.g., Löhner et al. 1985), to provide increased spatial resolution based on predetermined physical criteria. The former method is rather complex due to the nature of the algorithms involved, while the latter is receiving increased attention and appears to be plausible for a number of fluid flow applications. In addition to these two methods, grid points may be added in a more structured manner through the placement of multiple and perhaps overlapping finer-scale grids in the domain. Though similar in concept to grid nesting, this adaptive refinement technique, which was pioneered by Berger and Oliger (1984), automatically determines the location, orientation, and resolution of the subgrids, and can be incorporated as a general solver in hydrodynamical models without modifying the governing equations or their numerical representation. Adaptive refinement has been successfully applied to both large- and small-scale atmospheric models in two dimensions (Skamarock et al. 1989; Skamarock and Klemp 1989), and extension to three dimensions is currently being investigated (Skamarock 1989, personal communication).

The second category of refinement techniques, which forms the basis of this paper, involves methods that redistribute, in some predetermined manner, a fixed number of grid points (e.g., Thompson et al. 1985, hereafter TWM) or collocation points (e.g., Baines and Wathen 1988; Johnson et al. 1988) so as to provide locally increased resolution and thus an improved solution in certain regions of the domain. The criteria that determine how the grid points are redistributed with time are a critical element of this method, and are typically based on physical characteristics of the evolving flow field (e.g., the magnitude of gradients in certain fields, local kinetic energy, reaction rates, or a combination of such quantities). In this manner the computational mesh structure is globally and dynamically coupled to the physics of the problem so that both are solved in a continuous manner (Thompson 1984; hereafter TWM). In the case of finite-difference discretizations, we refer to this technique as continuous dynamic grid adaptation (CDGA). A major conceptual difference between CDGA and techniques such as those of Skamarock is that grid evolution in the latter occurs in a series of discrete jumps that usually involve interpolation of data from one grid to another, whereas the grid evolution in CDGA occurs in a smooth manner, and this evolution is incorporated, through the coordinate transformation described below, into the solution of the governing equations at every time step. CDGA is similar in concept to the grid-stretching techniques described earlier except that, in dynamic adaptation, the grid-transformation terms in the governing equations change continuously with time and in direct response to the evolving flow. Consequently, the grid develops finer and finer resolution adaptively in regions where the truncation error, and thus the solution error, are large (typically where the gradient and Laplacian are large). At first sight the CDGA method appears to be at a disadvantage compared to the adaptive refinement technique, as the former can create high-reso-
solution zones only by reducing the resolution in other parts of the domain, whereas the latter creates zones of high resolution simply by adding another high-resolution grid. Of course, in practice, one has fixed computer resources in CPU time and memory available, and thus both approaches attempt to achieve an optimum allocation of a fixed number of grid points. Thus, in the application of any adaptive method, care must be taken to ensure that the resolution in coarse zones is not diminished to the extent that the ability to resolve even relatively large-scale flow features is lost. This can be achieved by using a reasonable number of grid points to begin with and by limiting the degree of adaption to a moderate extent; fortunately, most adaptive algorithms are sufficiently flexible that this last requirement is easily met (see section 4).

In this paper we apply CDGA to meteorological modeling and demonstrate its capabilities for accurately representing multiscale events in both linear and nonlinear problems. In particular, we show that through its direct coupling to the evolving solution, CDGA provides smoothly varying spatial resolution across the domain with locally high resolution in regions exhibiting sharp gradients and large second derivatives. To our knowledge, the first reported instance of the application of CDGA in meteorology is Leslie et al. (1989) in which an adaptive tropical cyclone model developed by the first author of the current paper is described; however, no details of the technique were given. Although CDGA is a very recent addition to the tools used in numerical meteorology, it is interesting to note that its use was suggested over 20 years ago by Charney (1966), who proposed a technique whereby one permits “the computational lattice itself to distort as the flow evolves in such a way that the lattice spacing adjusts to the gradient of the quantity one wishes to resolve.” It is important to recognize that CDGA is quite distinct from Lagrangian or semi-Lagrangian methods because the nonlinear advection terms are retained in CDGA and play a key role in the coordinate transformation by allowing for the motion of the grid points. Further, this motion is determined not by the fluid velocity, but as a response to the evolving spatial distribution of errors in the model.

Our CDGA formulation closely follows that proposed by Brackbill and Saltzman (1982) and Saltzman and Brackbill (1982) in that we desire the mapping between the curvilinear grid in physical space and the uniform grid in computational space to be determined by a linear combination of three parameters that describe (a) the smoothness of the grid, (b) the orthogonality of the grid, and (c) the compactness or density of the grid points. One way to join these requirements is through a variational approach that seeks to extremize the combined effects of these physical properties. The resulting variational problem can then be solved by constructing the associated Euler equations.

The steps in applying the CDGA technique to a set of hydrodynamical equations are as follows. First, the governing partial differential equations are transformed from physical space \((x, y, z, t)\) to computational space \((\xi, \eta, \psi, \tau)\), with the result that all spatial derivatives now contain mapping factors (the Jacobian of the transformation) and the advective terms have an extra component accounting for the motion of the grid points. Second, a weight function \(W\), based on physical properties of the flow (here, the gradient and Laplacian of the field variables), is constructed to determine the compactness of the grid. The compactness term essentially has the form \(W \Delta = \text{constant}\), where \(\Delta\) is the grid spacing. Thus, regions of large (small) \(W\) are associated with small (large) grid spacings. Third, the Euler equations of the variational problem are derived, yielding a set of coupled, nonlinear elliptic partial differential equations for the gridpoint locations in physical space. Finally, the two systems of equations (the transformed governing dynamical equations and the elliptic grid generator equations) are discretized and solved using appropriate techniques.

In the following sections we describe each of these steps in detail and present several physical examples to demonstrate the capabilities of CDGA. In the example, the ability of CDGA to track rapidly propagating discontinuities is investigated through a one-dimensional shock-wave problem using Burgers’ equation. We then apply CDGA to a linear, two-dimensional kinematic frontogenesis problem in which localized, sharp gradients develop with time. Direct comparisons are made between the exact and the numerical solutions to quantify the benefits gained in using dynamic adaption. Further passive advection experiments are also conducted in which four “cones” are advected by a field of solid-body rotation. These experiments demonstrate that CDGA encounters no great difficulty in dealing with problems that require multiple zones of high resolution. Finally, we apply CDGA to a fully nonlinear simulation of a buoyant convective thermal in a two-dimensional Boussinesq fluid, and conclude with some brief comments on the computational efficiency of the method (this issue is dealt with fully in Dietachmayer 1992, hereafter referred to as Part II), together with proposed directions for future research, some of which are presented in Part II.

2. The CDGA technique in one dimension

In this section we introduce the CDGA technique in one dimension and apply it to the solution of Burgers’ equation. Although we examine only one test problem, much of the discussion below is couched in general terms to facilitate later expansion to more complex problems. Our aim here is to identify and solve any problems in the CDGA technique that may
arise within this simple environment, before moving on to more complex multidimensional problems.

a. Technique

There are two basic components to the CDGA technique: the numerical solution of the governing equation(s) for the prognostic variable(s), and the generation of a grid that will enhance the accuracy of the numerical integration. We begin our discussion with the treatment of the prognostic variable, assuming that the grid is known, and then go on to consider the construction of such suitable grids.

Any method for solving the prognostic equation must allow for the fact that the grid exhibits both spatial and temporal variation. The usual approach is to define a mapping between the physical coordinates \((x, t)\) and the so-called computational coordinates \((\epsilon, \tau)\). The spatial computational coordinate measures distance, not in meters, but in terms of the number of grid intervals between two points. Thus, \(\epsilon\) at the \(i\)th grid point is simply \(i\), and the distance between two adjacent grid points is \(1\); that is, \(\Delta \epsilon = 1\). Using this definition, it is clear that the grid spacing in computational coordinates is everywhere constant (equal to 1) and that the grid points are fixed in time (since, for example, grid point \(i + m\) is always \(m\) grid intervals away from grid point \(i\)). By transforming the governing equations from physical to computational space we are left with equations that can be solved by standard (fixed-grid) finite-difference techniques. We choose this approach over the alternative of solving the governing equations directly in physical space as it more readily allows for extension to multidimensions, and results in conceptually simpler finite-difference schemes for the prognostic variables. [We acknowledge that it is feasible to formulate schemes in moving meshes directly, as is done in the Lagrangian method of Fritts and Boris (1979) and others. Ultimately, this may even be the preferred approach because of the errors that result from using the transformed set of equations (see below), but at this early stage we feel that the simplicity and generality of the mapping approach is too important an advantage to ignore.]

The one-dimensional form of Burgers' equation is

\[
\frac{\partial u}{\partial t} = -u \frac{\partial u}{\partial x} + \nu \frac{\partial^2 u}{\partial x^2}. \tag{2.1}
\]

Introducing the general transformation of coordinates from \((x, t)\) to \((\epsilon, \tau)\) (where \(t = \tau\)), we replace (2.1) by

\[
\frac{\partial u}{\partial \tau} = -u_E \frac{\partial u}{\partial \epsilon} + \nu \left( \frac{\partial x}{\partial \epsilon} \right)^{-2} \left[ \frac{\partial^2 u}{\partial \epsilon^2} - \left( \frac{\partial x}{\partial \epsilon} \right)^{-1} \frac{\partial^2 x}{\partial \epsilon^2} \frac{\partial u}{\partial \epsilon} \right], \tag{2.2}
\]

where

\[
u_E = \left( u - \frac{\partial x}{\partial \tau} \right) \left( \frac{\partial x}{\partial \epsilon} \right)^{-1}
\]

is the effective velocity. Note the appearance of an extra advective term in (2.2), (2.3) associated with the motion of the grid. By incorporating the grid motion directly into the governing equations, we avoid interpolating data from the old gridpoint positions to the new.

We turn now to the finite-difference solution of (2.2). We approximate all spatial derivatives by central differences, and recall that \(\Delta \epsilon = 1\). Although central differences are second-order accurate, this does not imply that the resulting scheme for (2.2) is second order. For example,

\[
\frac{\partial f}{\partial x} = \frac{\partial f}{\partial \epsilon} \left( \frac{\partial x}{\partial \epsilon} \right)^{-1} \tag{2.4}
\]

(where \(f\) is an arbitrary function of \(x\) only) is approximated by

\[
\frac{\partial f}{\partial x} \approx \frac{(f_{i+1} - f_{i-1})}{(x_{i+1} - x_{i-1})} \tag{2.5}
\]

(here we use the standard notation that the subscript refers to the gridpoint number, while a superscript refers to a time level), which is clearly not second-order accurate unless the grid is uniform.

The question of the order of accuracy of finite-difference expressions in arbitrary curvilinear coordinate systems has been dealt with in some detail by TWM and Thompson and Mastin (1985). They point out that some ambiguity exists in defining order on such grids as one may change the local grid spacing in two ways: by changing the distribution of grid points while keeping the total number of points fixed, or by increasing the number of grid points while keeping the distribution of grid points fixed. For the latter case they show that (2.5) is second-order accurate in the sense that the error goes to zero as the reciprocal of the number of grid points squared. They also derive the truncation error \(T\) for (2.5),

\[
T = \frac{1}{6} \left[ 3 \frac{\partial^2 x}{\partial \epsilon^2} \frac{\partial^2 f}{\partial \epsilon^2} + \left( \frac{\partial x}{\partial \epsilon} \right)^2 \frac{\partial^2 f}{\partial \epsilon^3} \right]. \tag{2.6}
\]

Since \(\partial x/\partial \epsilon \approx \Delta x/\Delta \epsilon = \Delta x\), we note that the second term in (2.6) is the same error that arises in difference approximations on uniform grids, while the first term is related to the rate of change of the grid spacing. To keep this term sufficiently small we require that the gridpoint distribution in space be a "smooth" one. Failing to do so has the potential to cause serious problems, as this error term acts as an artificial diffusion term that may have a negative coefficient of viscosity, and could thus lead to numerical instability. However,
it is also worthwhile to note that large percentage changes in the grid spacing can be tolerated in regions where \( f \) is smooth (\( \partial^2 f / \partial x^2 \approx 0 \)) or where the grid spacing is small because the error is related to the magnitude of the change in grid spacing, not the fractional change from one cell to the next.

Having defined the spatial discretization, we turn our attention to the time integration algorithm. We assume that \( u \) is given at the \( n \)th time level, that \( x \) is known at both the \( n \) and \( n + 1 \) time levels, and that the gridpoint velocity, \( x_r \), is defined at the midtime level by

\[
x_r^{n+\frac{1}{2}} = \frac{(x^{n+1} - x^n)}{\Delta t}.
\]

The algorithm is based on the two-step Heun scheme in which first-order-accurate values of \( u^{n+1} \) are found using a forward time step, and these values are averaged with the values at the previous time level to yield an approximation to \( u^{n+\frac{1}{2}} \), which in turn is used in the final centered step from \( u^n \) to \( u^{n+1} \). We do not believe that any of the presented results are particularly sensitive to this choice of time-stepping algorithm, at least to the extent that we would expect very similar results for any second-order-accurate method. When applied to the evolutionary equation \( \partial \phi / \partial t = M(\phi) \), where \( \phi \) is the prognostic variable and \( M \) is a spatial operator with corresponding discretized form \( N \), the scheme is

\[
\begin{align*}
\phi^n & = \phi^n + \Delta t N(\phi^n) \\
\phi^{n+\frac{1}{2}} & = \frac{(\phi^n + \phi^*)}{2} \\
\phi^{n+1} & = \phi^n + \Delta t N(\phi^{n+\frac{1}{2}}).
\end{align*}
\]

The application of (2.8) to (2.2) is made slightly more difficult by the fact that, in (2.2), both \( x \) and \( u \) are functions of time; thus, the gridpoint velocity is readily evaluated only at midtime levels. (To evaluate the gridpoint velocity to second-order accuracy at the \( n \)th time level would require knowledge of the gridpoint positions at the level \( n - 1 \), and thus more memory space, particularly in multidimensional calculations.) We may rewrite (2.2) as

\[
\frac{\partial u}{\partial \tau} = \left[ x_r \left( \frac{\partial x}{\partial \epsilon} \right)^{-1} \right] \frac{\partial u}{\partial \epsilon} + M(x, u),
\]

where \( M \) is now the remainder of the right-hand side of (2.2). We then apply scheme (2.8) to (2.9), with the exception that the coefficient of \( \partial u/\partial \epsilon \) in the first term on the right-hand side of (2.9) is held fixed at its midlevel value; that is,

\[
\begin{align*}
u^n & = u^n + \Delta t \left[ x_r^{n+\frac{1}{2}} \left( \frac{\partial x^{n+\frac{1}{2}}}{\partial \epsilon} \right)^{-1} \right] \frac{\partial u^n}{\partial \epsilon} \\
& \quad + \Delta t M(x^n, u^n), \\
u^{n+\frac{1}{2}} & = \frac{\left( u^n + u^n \right)}{2},
\end{align*}
\]

By design this scheme does not require gridpoint velocities at full-time levels, and is also second-order accurate. The treatment of the gridpoint advection term is based on the work of Smolarkiewicz (1985), who used a similar treatment to improve the accuracy of the Crowley advection scheme, applied to a time-dependent advecting flow, from first to second order.

Because the grid spacing is continually changing, it is desirable to employ a variable time step \( \Delta t \). Equation (2.2) suggests the following stability criterion for the advective component,

\[
|u_E \Delta t / \Delta \epsilon| \leq 1.
\]

Accordingly, we take \( \Delta t \) to be the largest value that satisfies

\[
|u_E \Delta t| \leq s,
\]

where \( s \) is a constant "safety factor" \((0 < s < 1)\), which for most of the experiments described below is set equal to 0.6. In conjunction with (2.12), we also restrict the magnitude of \( \Delta \epsilon \) to satisfy the diffusive stability criteria (see, for example, Roach 1982, p. 43) using an effective diffusion coefficient,

\[
\nu_E = \nu \left( \frac{\partial x}{\partial \epsilon} \right)^2.
\]

We now turn to the construction of grids that will enhance the numerical solution of (2.2). We assume the existence of a weight function \( W(x, t) \), which is scaled to lie between 0 and 1 and tends to its upper limit in regions in which high resolution is required, and to zero in areas in which coarse resolution is sufficient. To construct the grid generator equation we use the variational approach of Brackbill and Saltzman (1982) and TWM.

Consider the solution of the variational problem,

\[
\text{minimize } \int \left( \frac{\partial x}{\partial \epsilon} \right)^2 \, d\epsilon,
\]

where the integral is taken over the entire computational domain. Because \( \Delta \epsilon \) is constant, we may approximate the integral as a simple summation over all the grid intervals; also, since \( \partial x / \partial \epsilon \approx \Delta x \), we see that (2.14) amounts to minimizing the sum of the squares of the grid spacing, which has the trivial solution \( \Delta x = \text{constant} \), that is, the smoothest possible grid. To include a degree of adaptivity in the grid construction we could replace \( \partial x / \partial \epsilon \) in (2.14) by \( W(x, t) \partial x / \partial \epsilon \); however, this could lead to extreme variations in the grid spacing as well as allow infinitely large grid inter-
vals in regions where \( W = 0 \). Accordingly, we take a weighted combination of the two formulations; that is,

\[
\text{minimize } \int (1 + \lambda W^2) \left( \frac{\partial x}{\partial \epsilon} \right)^2 d\epsilon, \quad (2.15)
\]

where \( \lambda \) is a user-specified constant that controls the degree of adaption; small values of \( \lambda \) will result in very smooth grids with little response to the weight function, while larger values will increase the impact of the weight function, but at the possible risk of introducing rapid changes in the grid spacing (and thus increased truncation error). To find the gridpoint distribution we do not solve (2.15) directly (although this can be done; see Carcaillot et al. 1986). Instead, we solve the Euler equation for (2.15), which is found to be

\[
(1 + \lambda W^2) \frac{\partial^2 x}{\partial \epsilon^2} + \lambda W \frac{\partial W}{\partial \epsilon} \frac{\partial x}{\partial \epsilon} = 0. \quad (2.16)
\]

Equation (2.16) is a nonlinear elliptic equation for \( x \) as a function of \( \epsilon \) that is properly posed if appropriate boundary conditions on \( x \) are supplied. Here we assume that \( x \) at both ends of the computational domain is known for all time (in fact, the end points are fixed).

We consider three finite-difference representations of (2.16). The first is obtained by applying central differences to yield

\[
(1 + \lambda W_i^2)(x_{i+1} - 2x_i + x_{i-1}) + \lambda W_i(W_{i+1} - W_{i-1})(x_{i+1} - x_{i-1})/4 = 0. \quad (2.17a)
\]

For the second scheme we evaluate the second term of (2.16) at half points \( i + \frac{1}{2} \) and \( i - \frac{1}{2} \) and the average to find the value at \( i \). This approach yields

\[
(1 + \lambda W_i^2)(x_{i+1} - 2x_i + x_{i-1}) + \lambda(W_{i+1} + W_i)(W_{i+1} - W_i)(x_{i+1} - x_i)/4

+ \lambda(W_i + W_{i-1})(W_i - W_{i-1})(x_i - x_{i-1})/4 = 0. \quad (2.17b)
\]

The third scheme is derived by noting that (2.16) is equivalent to

\[
\frac{\partial}{\partial \epsilon} \left[ (1 + \lambda W^2) \left( \frac{\partial x}{\partial \epsilon} \right)^2 \right] = 0, \quad (2.18)
\]

which can be approximated by

\[
[1 + \lambda(W_{i+1} + W_i)^2/4](x_{i+1} - x_i)^2 - [1 + \lambda(W_i + W_{i-1})^2/4](x_i - x_{i-1})^2 = 0. \quad (2.17c)
\]

We note that in both (2.17b) and (2.17c), first derivatives are approximated by differences over a single grid interval, while in (2.17a) such derivatives are differenced over two grid intervals and are thus less accurate and more likely to be susceptible to spatial oscillations (as is commonly the case with central differencing). A very simple example can illustrate the impact of this reduced accuracy. Let \( x_i \) be fixed in space, and assume that \( W_i \) is unity for all values of \( i \) greater than \( I \), and zero otherwise. For brevity of notation let \( \Delta x^+ = x_{i+1} - x_i \) and \( \Delta x^- = x_i - x_{i-1} \). Given the above distribution of the weight function, we would intuitively expect that \( \Delta x^- > \Delta x^+ \). However, direct substitution of the weight function values into (2.17a) yields \( \Delta x^- = \Delta x^+ \). In comparison, (2.17b) gives \( \Delta x^- = (1 + \lambda/4)\Delta x^+ \), as expected, with a similar result holding for scheme (2.17c).

Equation (2.18) is the equidistribution form of the grid-generator equation. It shows that \( (1 + \lambda W^2)^{-1/2} \partial x/\partial \epsilon \) is equidistributed (held constant) over each grid interval. This does not imply that we can achieve arbitrarily small local grid spacing just by making \( \lambda \) sufficiently large. For a computational domain with \( N \) grid points and \( x \) bounded between 0 and \( X \), it is straightforward to show from the equidistribution statement that

\[
\frac{\partial x}{\partial \epsilon} = \frac{X}{(N-1)} \left[ (\lambda^{-1} + W^2)^{-1/2} \right] \left[ \frac{X}{(N-1)} \right]^{-1} \times (\lambda^{-1} + W^2)^{-1/2}, \quad (2.19)
\]

[where \( \phi_{\text{avg}} \) is defined in the usual way, \( \phi_{\text{avg}} = (1/X) \int_0^X \phi(x)dx \), and thus for sufficiently large values of \( \lambda \), the minimum value of \( \partial x/\partial \epsilon \) (which occurs at \( W = 1 \)) is given by

\[
\left( \frac{\partial x}{\partial \epsilon} \right)_{\text{min}} = \frac{X}{(N-1)} \left[ (\lambda^{-1} + W^2)^{-1/2} \right]^{-1}. \quad (2.20)
\]

From (2.20) it is clear that further reduction in the grid spacing is possible only by making the weight function more sharply discriminating (thus reducing its average value).

To solve any of (2.17) we adopt the iterative method of Brackbill and Saltzman (1982). Let a bracketed superscript denote the iteration level of a given variable. We may write (2.17) symbolically as \( R(x_{i-1}, x_i, x_{i+1}, \ W_{i-1}, W_i, W_{i+1}) = 0 \). At a given iteration level \( k \), \( R \) will in general not be zero, but we want \( R^{(k+1)} \) to tend to zero. To achieve this we find \( x^{(k+1)} \) from the approximation
\[ 0 = R^{(k+1)} = R^{(k)} + \frac{\partial R^{(k)}}{\partial x_i} [X_i^{(k+1)} - X_i^{(k)}]. \] (2.21)

[Because \( W \) is not known as an explicit function of \( x \), we ignore its contribution to \( \partial R^{(k)}/\partial x_i \) in (2.21).] Having found \( x_i^{(k+1)} \) at each grid point, we next update the weight function at each point, \( W^{(k+1)} = W[X_i^{(k+1)}] \), which can be done efficiently via

\[ W^{(k+1)} = W^{(k)} + [X^{(k+1)} - X^{(k)}] \frac{\partial W^{(k)}}{\partial \xi} \left[ \frac{\partial X^{(k)}}{\partial \xi} \right]^{-1}. \] (2.22)

This cycle is repeated until the maximum change in any of the gridpoint positions is sufficiently small. The iterative technique is weakly underrelaxed to improve its convergence properties.

To complete the solution technique, the final element is the construction of an effective weight function. The obvious candidate is to use the lowest-order terms from the truncation error of the numerical algorithm used to solve the prognostic equation. This is generally not done for several reasons. First, for a full set of multidimensional equations it may be very difficult to derive analytical expressions for the truncation error [although Skamarock (1989) has reported some very promising results using a reasonably complete form of the truncation error]. Second, as shown in section 4, the lowest-order terms do not necessarily provide a reliable guide to the local error. Third, such error estimates usually involve third- and higher-order derivatives of the prognostic variable, making their accurate calculation difficult. Fourth, for the CDGA technique, the truncation error is dependent not only on the distribution of the prognostic variable, but also on the gridpoint distribution. Thus, the situation could arise where a high local value of the truncation error associated with too rapid change in the grid spacing is exacerbated by a further decrease in the local grid spacing as the grid generator increases the local resolution still further, at the expense of neighboring regions. Finally, we note that the weight function is usually smoothed, eliminating some of the fine detail that sophisticated error evaluators capture, and that most of the improvement that results from using an adaptive technique results from getting increased resolution close to where it is required; it is questionable whether it is cost effective to spend the extra effort to get the resolution exactly where it is required. Indeed, Babuška and Rheinboldt (1979) show that, in the context of finite-element calculations, the optimal error (that results from the use of an optimal grid) is stable with respect to perturbations about the optimal grid. In light of this, we choose a weight function based on experience of when finite-difference schemes are likely to encounter difficulties. Regions of rapid change in the prognostic variable require increased resolution to capture the change, and spurious numerical oscillations are often found near "spikes" or "corners" (regions of rapid change in the gradient) of the prognostic variable [see, for example, van Leer (1973), Dietachmayer (1987)]. Accordingly, we define the weight function as

\[ W(x, t) = W_1 \left| \frac{\partial u}{\partial x} \right| + W_2 \left| \frac{\partial^2 u}{\partial x^2} \right|, \] (2.23)

where \( W_1 \) and \( W_2 \) are user-defined constants, which in all the calculations below are both set equal to one. This choice is not the result of a careful tuning of parameters (which would reduce the general utility of the CDGA technique). In fact, this was the first guess for values of \( W_1, W_2 \), and as will be seen in the results below, it proved perfectly adequate. The same can be said of the general form of the weight function (2.23).

The weight function evaluated from (2.23) is not fed into the grid-generator equation until it has been smoothed by several passes of a 1–2–1 filter. There are several reasons for doing this. The sequencing of the solution algorithm for the prognostic variable (see the following) is such that we must predict the grid at time level \( n + 1 \) from knowledge of the prognostic variable at the \( n \)th time level only. Thus, it is prudent to smear the effect of the weight function across several grid points to allow for any motion of significant features (for example shocks and fronts) over the course of the time step. [This problem can be avoided by solving for both the grid and the prognostic variable simultaneously at the new time level. Such a procedure tends to be very computationally expensive and is probably not justifiable for meteorological applications. For examples of this approach see Miller and Miller (1981) or Winkler et al. (1985).] This smearing is also advisable to cover for the fact that the weight function is not a precise error estimate. Further, the weight function requires second derivatives of the prognostic variable, which means that it may become noisy. Since the grid generator requires that \( W \) be differentiated once again, it is clearly advantageous to ensure that the weight function is smooth. A smooth weight function enhances the accuracy of the solution of the grid generator equation (since it is solved numerically) and, most importantly, leads to a smooth gridpoint distribution, thus reducing some of the truncation error in the transformed derivatives.

The final processing of the weight function is to scale it so that it is bounded by 0 and 1. This scaling is an important part of the CDGA technique, as it allows the user a simple method of controlling the degree of grid adaption through the parameter \( \lambda \). Without the scaling the user would have to predict a priori the magnitude of the weight function in order to choose an appropriate \( \lambda \) to achieve a given level of grid adaption; this would clearly be very difficult for most problems.
of interest. The scaling also provides a safeguard when solving problems that have discontinuous (or near discontinuous) solutions; without it, the grid spacing could be forced to zero as the weight function tends to infinity (i.e., as the local resolution is increased).

We now have all the necessary ingredients for the CDGA technique; it remains to assemble them into a complete method for the integration of the prognostic variable $u$. Assume that we know both $x$ and $u$ at the $n$th time level. The new time step $\Delta t$ can be found as described above, although the gridpoint velocity used in (2.12) is from the previous time level. The weight function is evaluated from $u''$, and then the grid-generator equation (2.17) is solved to determine the new gridpoint locations, $x^{n+1}$. The gridpoint speed and location at the midtime level are then calculated, whence $u$ can be integrated from $n$ to $n+1$ using (2.10), and the entire cycle can then be repeated to find $x$ and $u$ at $n+2$, and so on.

b. Results

We consider the solution of Burgers' equation in the domain $0 \leq x \leq 1$, with $v = 0.01$ m$^2$ s$^{-1}$. The initial condition for $u$ is the ramp profile shown in Fig. 1. Boundary conditions are provided by holding $u$ fixed at its initial value at $x = 0$ and $x = 1$. The exact solution develops a shock, which then propagates at 0.5 m s$^{-1}$ until it reaches the right-hand boundary, whereupon the solution becomes stationary. The finite-difference scheme (2.10) has intentionally been formulated in nonconservative form so that the correct shock speed is not guaranteed (Roach 1982). The maximum and minimum values of the exact solution for $u$ are the extreme values from the initial field of $u$, that is, 0 and 1. This problem provides a quite stringent test of the technique, as the initial location of the shock must be found (the initial gradient in $u$ is weak by comparison), and then the shock must be tracked as it propagates across the domain.

In the following discussion, NS is the number of passes of smoothing applied to the weight function (there is no explicit smoothing of any kind applied to
$u$, and $N_P$ is the number of grid points used. Most of the runs are terminated at $t = 1.8$ s, by which time the shock has reached the right-hand boundary and become stationary.

For $N_P = 11$, $N_S = 2$, and $\lambda = 100$, it was not possible to obtain a solution for $u$ using scheme (2.17a); as the calculations progressed the time step was reduced continually until eventually the calculations were being performed with $\Delta t$ essentially 0 (zero). However, a solution could be obtained using (2.17b) or (2.17c). Figure 2 shows $u$ as a function of $x$ at $t = 1.8$ s, together with a time history of the gridpoint positions, obtained by using (2.17b). Large, high-frequency oscillations in the gridpoint positions are clearly evident. Such grid oscillations have also been observed by DeLillo and Jordan (1987) and Bell and Shubin (1983). Bell and Shubin find that such oscillations occur even at considerably higher resolutions (they used 41 grid points). To overcome this difficulty they introduce a penalty term into the variational formulation, related to the speed at which the grid adapts. The problem with this approach is that it introduces another parameter into the grid-generator equation that must be tuned for best results. If the parameter is too small the oscillations remain, while if it is too large the grid cannot adapt sufficiently quickly to changes in $u$, and the accuracy of the results is degraded. In light of this result, and the fact that we could not obtain a solution at all with scheme (2.17a), it seems worthwhile to consider the problem of grid oscillations in some detail.

The results using scheme (2.17c) (Fig. 3) are a considerable improvement over those using (2.17b). The oscillations are still present, but their magnitude has been much reduced. Thus, although all schemes (2.17)

---

**Fig. 3.** (a) As per Fig. 2a, but using grid generator (2.17c). (b) As per Fig. 2b, but using grid generator (2.17c). (c) As per (b), but a time history of the time step $\Delta t$. 

---

Unauthenticated | Downloaded 02/25/24 01:59 AM UTC
lead to equidistribution for sufficiently high resolution, the construction of schemes that satisfy the equidistribution principle at the discrete level would seem to confer considerable advantages in terms of the robustness of the grid generator.

It is possible to further reduce the grid oscillations once one has noted that they occur in conjunction with large temporal oscillations in the magnitude of the time step (Fig. 3c). Once excited, these temporal oscillations in $\Delta t$ feed back into the grid calculations through $u$, which is dependent on $x_\tau$, which in turn is dependent on $\Delta t$. To eliminate this feedback we repeat the previous experiment, but this time with $\Delta t$ replaced by the average of its predicted value from (2.12), and its value from the previous time step. As shown by Fig. 4, this procedure significantly improves both the grid-point trajectories and the evolution of the time step.

It is important to acknowledge that these experiments are very difficult ones for the CDGA technique and that the differences resulting from the use of (2.17a)–(2.17c) can certainly be reduced in less stringent tests. For example, scheme (2.17a) does work if $\text{NS}$ is increased to 3; however, the grid oscillations remain larger than those for scheme (2.17c) with $\text{NS} = 2$ (results not shown). This is an important result as it shows that the better performance of (2.17c) is not due just to the averaging of $W$ to find weight-function values between grid points. Further, the improvement in the

Fig. 4. (a) As per Fig. 3a, but with the time evolution of $\Delta t$ smoothed by averaging. (b) As per Fig. 3b, but with the time evolution of $\Delta t$ smoothed by averaging. (c) As per Fig. 3c, but with the time evolution of $\Delta t$ smoothed by averaging.
results as NS is increased supports the preceding comments on the importance of such smoothing. If the resolution is increased, the problem of grid oscillations is, as expected, much reduced. Figure 5 shows the results for the case NP = 21, NS = 2, using (2.17c). The grid oscillations have virtually disappeared. The results (not shown) using (2.17a) with NS increased to 3 are virtually the same as in Fig. 5, but significantly, no results could be obtained with NS = 2. We conclude that the use of a gridpoint velocity damping term along the lines of Bell and Shubin may be avoided by a judicious choice of the finite-difference representation of the grid-generator equation, and/or increased grid resolution, together with spatial smoothing of the weight function and temporal smoothing of the time step.

We turn our attention now to the improvement in accuracy that results from the use of the CDGA technique. Figure 6 shows the results obtained on a fixed grid (λ = 0) using 21 grid points. The occurrence of large oscillations in u behind the shock is a well-documented feature of standard finite-difference techniques (see, for example, Dietachmayer 1987) that has no counterpart in the adaptive results. The adaptive calculations with 11 grid points are clearly superior to the fixed grid results with twice as many points. Clearly, the CDGA technique results in a much more efficient usage of available resolution than fixed grid techniques, but the increase in local resolution is not the only advantage achieved. Figure 7 shows the results using scheme (2.17c) with NP = 11 and NS = 4 at t = 1.3 s, that is, while the shock is still propagating toward the right-hand boundary. Figure 8 is the result obtained using a fixed grid with NP = 34, which corresponds to using the minimum grid spacing achieved in the adaptive run. While the postshock oscillation is small in the fixed-grid calculation, it is totally absent in the adaptive calculation despite the fact that the fixed grid has higher resolution everywhere. This improved accuracy of the adaptive results can be explained by a careful examination of the gridpoint trajectories.

The following discussion refers to the gridpoint trajectory plot of Fig. 5. Initially the highest concentration
of points occurs at $x = 0.5$, the base of the ramp where $\partial^2 u / \partial x^2$ is infinite (although its finite-difference approximation is finite). For $t$ slightly greater than zero, $u$ is flat (equal to 1) in the neighborhood of the left-hand boundary (due to the imposed boundary condition on $u$), while for slightly larger values of $x$ it decreases rapidly. Thus, after $t = 0$ a second region of very high values of $|\partial^2 u / \partial x^2|$ is created just inside the left-hand boundary, and so there is a very rapid motion of grid points from right to left to accommodate the change in the structure of the weight function. As the shock forms, the faster-moving fluid behind catches up with the fluid in front of it, and we see that the two separate regions of high resolution merge into one at $t \approx 0.4$ s. From then on until $t \approx 1.4$ s, when the shock begins to interact with the wall, the motion of the grid points in the shock region is approximately constant. After $t \approx 1.4$ s the grid points become stationary and concentrated at the right-hand boundary. In the middle period, $0.4 < t < 1.4$, the speed of the grid points is approximately 0.44 m s$^{-1}$, which is very close to the theoretical shock speed. Thus, the transformed equation for $u$ reduces to $\partial u / \partial \tau = 0$, which the finite-difference scheme is able to integrate with little error, explaining the accurate results achieved by the adaptive method. The fact that the gridpoint speed is slightly slower than the shock speed indicates that the shock is moving slowly through the grid points and thus avoids compacting grid points in front of it or leaving too few grid points behind it.

We end this section with the following observations. The first, and most obvious, is the very considerable gain in accuracy that can be achieved by using the CDGA technique. We note that this gain in accuracy is due in part to the actual motion of the grid points, a property that is not shared by more traditional adaptive methods such as those based on adaptive grid nesting. We have also shown that while any of the finite-difference representations (2.17) can be made to work, (2.17c), which is derived as a discrete representation of the equidistribution principle, is more robust than the other two options. These results suggest that the problem of grid oscillation may be dealt with by careful consideration of the finite-difference representation of the grid-generator equation, together with sufficient smoothing of the weight function and smoothing of the time evolution of $\Delta \tau$. Finally, we note that boundary conditions on the prognostic variables should be carefully implemented to avoid the sudden introduction of discontinuities into the computational domain, which otherwise leads to very abrupt changes in the gridpoint distribution.

3. Adaptive grid generation in two dimensions

In this section we derive a two-dimensional grid generator as a straightforward extension of the one-dimensional case from the previous section. We note that many grid generators have been presented in the literature, and do not suggest that the one used here is in any sense the best, rather, that it is a logical extension of previous work and provides the user with a good degree of control over the gridpoint distribution.

We now have two physical spatial coordinates that we denote by $x_1$, $x_2$ or (where no potential for confusion exists) $x$, $y$, together with corresponding computational coordinates $\epsilon^1$, $\epsilon^2$ (or $\epsilon$, $\eta$). We use some
elementary results from differential geometry, some of which can be found in Sharman et al. (1988), or for a detailed presentation, see TWM. The (covariant) basis vectors for the computational coordinate system are

\[ \mathbf{a}_i = \frac{\partial \mathbf{r}}{\partial \epsilon^i} \quad i = 1, 2, \quad (3.1) \]

where \( \mathbf{r} \) is the position vector of a given point. The elements of the covariant metric tensor are then defined by

\[ g_{ij} = \mathbf{a}_i \cdot \mathbf{a}_j \quad i = 1, 2 \quad j = 1, 2, \quad (3.2) \]

and we let \( G \) denote the determinant of the metric tensor. It can then be shown that the area increment associated with infinitesimal displacements in \( \epsilon \) and \( \eta \) is given by

\[ dA = G^{1/2} d\epsilon d\eta, \quad (3.3) \]

and thus, the area of a grid cell is approximately \( G^{1/2} \) (recall that in computational space each cell has length and width equal to unity). We may now construct the two-dimensional generalization of (2.15) by simply replacing the square of the grid interval \( length \) by the square of the grid cell \( area \); that is, the gridpoint distribution is found by solving the variational problem,

\[ \text{minimize} \int (1 + \lambda W^2) G d\epsilon d\eta \quad (3.4) \]

(where the reason for the change in notation from \( \lambda \) to \( \lambda_W \) will become obvious later). Although (3.4) will generate a grid structure with high resolution where the weight function is large, for multidimensional applications there is an additional aspect of the grid response we have yet to consider.

We have already noted that the truncation error of finite-difference approximations to the governing equations in computational coordinates may be adversely affected by too rapid a change in grid spacing. In two or more dimensions the truncation error may also be increased if there is too great a departure from orthogonality, that is, if the grid cells become too skewed (TWM). To avoid this potential source of error, we follow Brackbill and Saltzman (1982) and introduce an additional term into the variational formulation that

\[ \sum_{k=1}^{2} \sum_{j=1}^{2} \left( \frac{\partial F}{\partial g_{kj}} + \frac{\partial F}{\partial g_{jk}} \right) \left( \frac{\partial^2 x_i}{\partial \epsilon^i \partial \epsilon^k} + \frac{\partial x_i}{\partial \epsilon^i} \right) \left( \frac{\partial W}{\partial x_m} \frac{\partial x_m}{\partial \epsilon^k} \right) \frac{\partial}{\partial W} \left( \frac{\partial F}{\partial g_{kj}} + \frac{\partial F}{\partial g_{jk}} \right) \right] \]

\[ + \frac{\partial x_i}{\partial \epsilon^i} \sum_{m=1}^{2} \left( \frac{\partial}{\partial g_{mn}} \left( \frac{\partial F}{\partial g_{kj}} + \frac{\partial F}{\partial g_{jk}} \right) \left( \frac{\partial^2 x_p}{\partial \epsilon^m \partial \epsilon^k} + \frac{\partial x_p}{\partial \epsilon^m} \frac{\partial^2 x_p}{\partial \epsilon^k} \right) \right) \right) \] - \frac{\partial F}{\partial W} \frac{\partial W}{\partial x_i} = 0, \quad i = 1, 2. \quad (3.7) \]

The details of the discretization and solution of (3.7) are given in the Appendix, but there are a couple of points worth noting here. The scaling of \( \lambda_W \) and \( \lambda_O \) required to ensure that all three terms in the functional

\[ F \] have the same dimensions (TWM) is slightly different from that presented in TWM due to the use here of a different grid smoothness measure. For (3.6) we simply

![Fig. 9. Construction of an orthogonality measure in terms of components of the metric tensor.](image)
scale \( \lambda_w \) by dividing it by the square of the average of \( W \). No scaling of \( \lambda_\theta \) is required. We have also found that convergence of the grid-generator equations is, occasionally, quickened if the calculations are performed in a series of passes. If we seek a grid for particular parameter values \( \lambda_w \) and \( \lambda_\theta \), we might first find the grid for parameter values \( \lambda_w/n_{\text{pass}} \), \( \lambda_\theta/n_{\text{pass}} \) (where \( n_{\text{pass}} \) is the number of passes used) and use this grid as the initial guess for the solution for parameters \( 2\lambda_w/n_{\text{pass}} \), \( 2\lambda_\theta/n_{\text{pass}} \), and so on, up until the final parameter values, in a continuation style technique in which the weight function is completely recalculated before each pass. We have needed to use this approach only to derive the initial grid for the thermal experiments of section 5; no difficulty was encountered in finding grids as the thermal evolved because the grid from the previous time level was available as a good first guess for the current one. Without a reasonable first guess, the weight function can drift away from its original distribution if there have been many iterative steps due to the repeated application of (2.22) (or its multidimensional equivalent). This problem could be avoided by recalculating the weight function directly from the prognostic variables at each iterative step, but this would be considerably more expensive and hence, in light of the above, probably not worthwhile.

4. The CDGA technique in two dimensions:

The kinematic frontogenesis problem

Having derived a multidimensional grid generator in the previous section, the remainder of this paper looks at the application of the CDGA technique to two two-dimensional problems of meteorological relevance.

We consider first the problem of kinematic frontogenesis, as described in a series of papers by Doswell (1984, 1985) and Davies-Jones (1985). The first paper defines the problem; we are interested in the time evolution of a passive scalar \( Q \) (representing, for example, potential temperature) as it is advected by a steady, nondivergent flow field with structure similar to that of a smoothed Rankine vortex (Fig. 10). The governing equation for \( Q \) is simply

\[
\frac{\partial Q}{\partial t} + u \frac{\partial Q}{\partial x} + v \frac{\partial Q}{\partial y} = 0. \tag{4.1}
\]

Doswell solves (4.1) approximately, and finds that strong frontogenesis occurs that transforms an initially diffuse distribution of \( Q \) into one consisting of zones of high gradient in \( Q \). He also illustrates the strong influence that vorticity exerts on frontogenesis, despite the fact that it does not appear in the conventional frontogenesis equation [his Eq. (5)], and further observes strong similarity between his results and satellite imagery of cloud patterns in an intense vortex over the ocean. Davies-Jones constructs exact solutions to (4.1), with the only restriction being that the advecting flow is a function of radius only. His results confirm those of Doswell, although they do reveal quantitative errors in that work. In the final paper of the series, Doswell uses these exact solutions to examine the evolution of \( Q \) in more detail, and at later times.

The kinematic frontogenesis problem provides an ideal first test for the two-dimensional CDGA technique. One obvious advantage is simplicity; one need consider only a single, linear, hyperbolic equation for \( Q \). The ability of the technique to refine the local resolution appropriately, in response to an evolving distribution of \( Q \) that includes highly localized regions of rapid variation in \( Q \), will also be severely tested. Finally, and perhaps most important, the existence of an exact solution will allow us to quantify the benefits of the CDGA technique in comparison to uniform grid methods.

Transforming (4.1) to computational coordinates yields

\[
\frac{\partial Q}{\partial \tau} = -u_E \frac{\partial Q}{\partial \xi} - v_E \frac{\partial Q}{\partial \eta}, \tag{4.2}
\]

where

\[
u_E = [(u - x_\tau)y_\eta - (v - y_\tau)x_\eta]G^{-1/2} \tag{4.3a}
\]

and

\[
v_E = [(v - y_\tau)x_\xi - (u - x_\tau)y_\xi]G^{-1/2}. \tag{4.3b}
\]

Because \( u \) and \( v \) are known for all time, there is no difficulty in finding \( u_E \) and \( v_E \) at midtime levels, and hence, no need to split the advective terms up (and treat them differently) in the time-stepping algorithm, as was done in section 2. Thus, the discretization of (4.2) and (4.3) is straightforward, using central differences for the spatial derivatives. For this preliminary study we have chosen to use the advective form in pref-
ference to the flux formulation, as the latter requires a more careful implementation, when used in an adaptive framework, if spurious sinks and sources are to be avoided (TWM, p. 158). It remains to choose an appropriate weight function to drive the grid generator; here we construct one using a geometrical approach, but emphasize that the details of the formulation are unlikely to be important and that any reasonable weight function should work well. The effect of using different weight functions is investigated in Part II.

If \( \mathbf{n} \) is the unit vector in the direction of greatest change in \( Q \), we may find first and second derivatives of \( Q \) with respect to \( n \) from

\[
\frac{\partial Q}{\partial n} = \cos \theta \frac{\partial Q}{\partial x} + \sin \theta \frac{\partial Q}{\partial y},
\]

\[
\frac{\partial^2 Q}{\partial n^2} = \cos^2 \theta \frac{\partial^2 Q}{\partial x^2} + 2 \sin \theta \cos \theta \frac{\partial^2 Q}{\partial x \partial y} + \sin^2 \theta \frac{\partial^2 Q}{\partial y^2},
\]

where

\[
\cos \theta = \frac{\partial Q}{\partial x} \left| \frac{\partial Q}{\partial n} \right|^{-1} \quad \sin \theta = \frac{\partial Q}{\partial y} \left| \frac{\partial Q}{\partial n} \right|^{-1}
\]

and expressions for second derivatives of \( Q \) in the computational coordinate system are given in section 5. We can now generalize a one-dimensional weight function described in TWM to obtain

\[
W = \left[ 1 + W_1 \left( \frac{\partial Q}{\partial n} \right)^2 \right]^{1/2} \left( 1 + W_2 |K| \right) - 1,
\]

where \( K \) is the curvature defined by

\[
K = \frac{\partial^2 Q}{\partial n^2} \left[ 1 + \left( \frac{\partial Q}{\partial n} \right)^2 \right]^{-3/2}
\]

and \( W_1, W_2 \) are user-defined constants. The rationale for (4.7) is as follows. Consider the one-dimensional equidistribution grid generator defined by the equation

\[
\left[ 1 + \left( \frac{\partial Q}{\partial \xi} \right)^2 \right] \left( \frac{\partial \xi}{\partial \xi} \right)^2 = \text{constant}.
\]

Since \( \xi \approx \Delta x \) we may write

\[
\left[ 1 + \left( \frac{\partial Q}{\partial \xi} \right)^2 \right] \Delta x^2 \approx \Delta x^2 + \Delta Q^2 \approx \text{constant},
\]

and thus, the arc length between successive points \((\xi_i, Q_i), (\xi_{i+1}, Q_{i+1})\) is approximately constant. Hence, the first part of the weight function (4.7) will ensure that there is high resolution wherever there is a large gradient in \( Q \). This would be insufficient in and of itself as it would lead to low resolution at extrema in \( Q \); thus, we include the factor related to the curvature. The last \((-1)\) term sets the weight function to zero if \( Q \) is locally constant.

As can be seen from Fig. 11, the initial \( Q \) field has no \( x \) dependence and only weak gradients in \( y \). The velocity field is virtually zero at the boundaries of the domain, and hence, \( Q(x, y) \) is held fixed there [note that \( Q(\xi, \eta) \) will vary as the grid points move along the boundary]. All of the presented results apply at \( t = 4.0 \) s, the computational domain is a square \((-4 \leq x \leq +4, -4 \leq y \leq +4)\), and contour plots of \( Q \) are presented using a contour interval of 0.1. Three error measures are discussed; \( E_{\text{max}} \) is the maximum difference between point values of the exact and the numerical values of \( Q \) anywhere in the domain, and \( E_{\text{rms}} \) and \( E_{\text{mod}} \) are defined as

\[
E_{\text{rms}} = \left( \frac{\int \int [Q(x, y) - Q_E(x, y)]^2 \, dx \, dy}{\int \int dx \, dy} \right)^{1/2},
\]

\[
E_{\text{mod}} = \frac{\int \int |Q(x, y) - Q_E(x, y)| \, dx \, dy}{\int \int dx \, dy},
\]

where \( Q_E \) is the exact solution.

The exact solution (Fig. 12) shows the spiral structure of the \( Q \) field, with each “arm” consisting of a relatively flat top, surrounded by regions of very strong gradient. The numerical solution on a fixed grid with \( 31 \times 31 \) points is given in Fig. 13, and contains, as expected, spurious over- and undershoots along the spiral arms due to the inability of the grid to resolve the intense \( Q \) gradients. As the resolution is increased, these errors are reduced until the stage is reached that,
Fig. 12. Exact solution for $Q$ at $t = 4.0$ s.

Fig. 13. Numerical solution for $Q$ at $t = 4.0$ s on a fixed $31 \times 31$ grid. $E_{\text{rms}} = 0.0714$.

Fig. 14. As per Fig. 13, but with the resolution increased to $71 \times 71$. $E_{\text{rms}} = 0.0156$.

qualitatively at least, the numerical solution (Fig. 14) is identical to the exact one.

We now turn to the adaptive results. First, to save unnecessary repetition in the discussion to follow, we define default parameters for the adaptive runs: $\lambda_W = 20, \lambda_O = 0, W_1 = 1, W_2 = 1$, and $NS = 4$, with a grid resolution of $31 \times 31$ points. Default values are used unless otherwise stated. As was the case for the one-dimensional experiments, these parameters were first choices and have not been tuned, although the relatively high value of $NS$ was influenced by the results of section 2. The sensitivity of the results to specific choices of these parameters is not great, and will be discussed later in this section.

The adaptive solution with $31 \times 31$ (hereafter denoted by $31^2$) grid points is clearly superior to its fixed-grid counterpart (compare Figs. 13 and 15), and is in fact both qualitatively and quantitatively closer to the fixed-grid result with $51^2$ points (the rms errors are $0.0305$ for the fixed $51^2$ grid and $0.0281$ for the $31^2$ adapted). Thus, the same degree of accuracy has been achieved with a factor of $2.8$ fewer grid points. As expected, there is a clustering of grid points in the center of the domain, but the grid response to the individual arms is almost impossible to discern. This is quite an intelligent response on the part of the grid generator, as it has only a small number of grid points available to distribute, and thus could force high resolution of the individual arms only at the risk of introducing abrupt changes in the grid spacing together with extremely skewed cells. When the resolution is increased to $61^2$ points, this restriction is lifted and the grid response clearly shows the detailed structure of the arms (Fig. 16).

To provide an overall indication of the accuracy of the adaptive versus the fixed-grid results, we have run both versions of the code at resolutions varying from $11^2$ up to $91^2$, and plotted the logarithm of the rms error against the logarithm of the resolution in Fig. 17. The two straight lines on the plot show the theoretical error distributions that would occur if (a) the error was
Fig. 15. (a) Numerical solution for $Q$ at $t = 4.0$ s using CDGA with default parameters. $E_{rms} = 0.0281$. (b) Gridpoint distribution at $t = 4.0$ s using CDGA with default parameters.

Fig. 16. (a) As per Fig. 15a, but with the resolution increased to $61 \times 61$. $E_{rms} = 0.0111$. (b) As per Fig. 15b, but with the resolution increased to $61 \times 61$.

proportional to the resolution (i.e., a first-order accurate scheme) or (b) the error was proportional to the square of the resolution (second-order accuracy). In the following discussion we take care to distinguish between two alternative definitions of the order of accuracy of a scheme. We use order of accuracy (hereafter OA) in its traditional sense as referring to the lowest power of the grid spacing that is found in the truncation error of the scheme, where the truncation error has been derived by Taylor series expansions of the difference approximations. By the effective order of accuracy (hereafter EOA), we mean the rate of change of the logarithm of the rms error with respect to variations in the logarithm of the resolution; that is, the slope of the lines in Fig. 17. It is true that the EOA is a slightly less rigorous definition of scheme accuracy than the OA as it depends on the evaluation of rms errors from a known exact solution and hence is problem dependent. On the other hand, it is also a more relevant measure as it provides an indication of how changes
that the EOA of the adaptive method is actually better than the fixed, rather than worse as expected. With the advantage of hindsight, this result can be readily explained by noting that the regions of very rapid variation in $Q$, which are major components of the truncation error in the fixed-grid results, are highly localized and, in the adaptive results, occur only in conjunction with local regions of high resolution. Thus, their contribution to the total error is minimal. For higher resolutions, the EOA steadily falls, and the adaptive results converge to the fixed. This is due to the fact that in the current formulation the degree of grid adaptation is determined only by $\lambda_W$ and is not related to the actual error. Thus, even when, as shown by the fixed-grid results, there is sufficient resolution to achieve essentially the exact solution, the grid generator does not know this and concentrates grid points in the spiral arms anyway. Because the resolution was already high enough to resolve these features, the improvement in accuracy due to the increased resolution is marginal, and the errors associated with the nonuniformity of the grid play a more and more predominant role as the resolution increases. This is one occasion where it would be beneficial for the weight function to be directly related to the error field or for $\lambda_W$ to be automatically reduced as the global error reduces. While these ideas are worthy of future research, we note that it is rare, in practical models, for the resolution to be sufficiently high that the exact solution is almost attained.

We now turn our attention to the sensitivity of CDGA to the grid generation parameters. Table 1 gives the three error measures as functions of NS for adaptive calculations using the default parameters. As discussed in section 2, the method has difficulty if there is insufficient smoothing of the weight field. The $E_{mod}$ errors reveal that there is an optimum value for NS, $NS_{opt} \approx 4$. We surmise that for NS too small the grid response is too irregular, due to rapid variations in the weight function, while for NS too large the weight function’s ability to discriminate between regions that do and do not require high resolution is severely impaired, and hence, the generator opts for a single, broad region of (reduced) high resolution. This effect is evident in Figs. 18 and 19. These results also suggest that controls on grid orthogonality and smoothness can be applied, al-

<table>
<thead>
<tr>
<th>NS</th>
<th>$E_{mod}$</th>
<th>$E_{rms}$</th>
<th>$E_{max}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.0094</td>
<td>No solution possible</td>
<td>0.407</td>
</tr>
<tr>
<td>2</td>
<td>0.0091</td>
<td>0.0281</td>
<td>0.330</td>
</tr>
<tr>
<td>3</td>
<td>0.0085</td>
<td>0.0281</td>
<td>0.296</td>
</tr>
<tr>
<td>4</td>
<td>0.0089</td>
<td>0.0286</td>
<td>0.268</td>
</tr>
<tr>
<td>5</td>
<td>0.0097</td>
<td>0.0296</td>
<td>0.267</td>
</tr>
<tr>
<td>6</td>
<td>0.0099</td>
<td>0.0310</td>
<td>0.271</td>
</tr>
</tbody>
</table>

in the resolution will actually affect the accuracy of the results, for all choices of resolution, rather than just in the asymptotic sense as the grid spacing tends to zero (as is the case with the OA).

Examination of the fixed-grid results reveals that the two measures are by no means equivalent. For very high resolutions it is clear that the OA and the EOA yield, as expected, the same result. But for very coarse resolutions, the EOA falls to less than unity and does not attain the expected value of 2 until the resolution is at least $51^2$. In other words, the scheme does not become second-order accurate, in the sense that a doubling of the resolution results in a quartering of the error, until the resolution is increased to such an extent that the exact solution (in the qualitative sense just described) has been almost achieved. The reason for the low values of the EOA at coarse resolutions is that the higher-order derivatives of $Q$ are extremely large, and hence, even though they are multiplied by higher powers of the grid spacing in the Taylor series for the truncation error, they still make a significant contribution to the error.

The analysis is remarkably different for the adaptive results. We have noted in previous sections how the order of accuracy of the adaptive method may be adversely affected by rapid changes in the grid spacing and/or extreme skewness of the grid cells. However, for resolutions up to $51^2$ (approximately) it is clear
predicted, the orthogonality control is particularly effective in application to the larger, highly skewed cells located in the outer part of the domain in Fig. 15. One undesirable effect of near orthogonality is that it forces changes in resolution in regions where the weight function is zero, simply due to topological requirements; coordinate lines that are crowded together in the center of the domain are not permitted to diverge in the outer parts, and hence a cross pattern of relatively high resolution is formed.

Fig. 18. (a) As per Fig. 15a, but with NS = 2. $E_{mn} = 0.0281$.
(b) As per Fig. 15b, but with NS = 2.

As loosely, by appropriate smoothing of the weight function, and hence, it may be possible to construct simpler grid generators than the ones presented here; this approach will be explored further in Part II.

When $\lambda_0$ is varied (Table 2) no optimum value is found, and in fact the error continually increases as the grid becomes more orthogonal. This can only be the result of reduced resolution in the center of the domain, and illustrates the trade-off between grid orthogonality on the one hand and response to the weight function on the other (compare Fig. 20 to Fig. 15). As

Fig. 19. (a) As per Fig. 15a, but with NS = 7. $E_{mn} = 0.0310$.
(b) As per Fig. 15b, but with NS = 7.
Table 2. CDGA error measures as a function of $\lambda_0$.

<table>
<thead>
<tr>
<th>$\lambda_0$</th>
<th>$E_{\text{mod}}$</th>
<th>$E_{\text{rms}}$</th>
<th>$E_{\text{max}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.0085</td>
<td>0.0281</td>
<td>0.296</td>
</tr>
<tr>
<td>1.0</td>
<td>0.0097</td>
<td>0.0315</td>
<td>0.296</td>
</tr>
<tr>
<td>2.0</td>
<td>0.0098</td>
<td>0.0321</td>
<td>0.289</td>
</tr>
<tr>
<td>5.0</td>
<td>0.0103</td>
<td>0.0331</td>
<td>0.273</td>
</tr>
<tr>
<td>10.0</td>
<td>0.0101</td>
<td>0.0353</td>
<td>0.288</td>
</tr>
<tr>
<td>40.0</td>
<td>0.0114</td>
<td>0.0408</td>
<td>0.335</td>
</tr>
<tr>
<td>100.0</td>
<td>0.0116</td>
<td>0.0409</td>
<td>0.325</td>
</tr>
</tbody>
</table>

There is also no optimum value for $\lambda_w$, but now the rms error continually decreases with increasing $\lambda_w$ until the point that no solution is attainable (Table 3), although once $\lambda_w$ is 20 or more the degree of sensitivity is quite small. The inability to obtain a converged solution for large $\lambda_w$ was also reported by Brackbill and Saltzman (1982), and is believed to be associated with a loss of the elliptic property of the grid-generator equations (B. Soni 1989, personal communication). The reducing degree of sensitivity to $\lambda_w$ is in accord with Bradley (1988), who recommends $\lambda_w \sim O(10)$. This result reinforces the prior observation that, for this particular (coarse) resolution, considerations of smoothness or orthogonality are secondary to the requirement of increased resolution. The accurate solution achieved for $\lambda_w = 200$ (results not shown) demonstrated the enormous range in grid spacing, which varied by a factor of 5 across the domain, that can be supported by the CDGA technique.

The remaining sensitivity we have yet to consider is the dependence of the results on the form of the weight function. Table 4 presents the error measures as functions of $W_2$, the parameter controlling the emphasis placed on the curvature of $Q$. The only clear impression that emerges is that the error is increased if $W_2$ is significantly larger than $W_1$. The interpretation is made more difficult by the large value of $\eta$ used in this study; accordingly, we conducted two further experiments with $\eta$ reduced to 3. For $W_1 = 1$, $W_2 = 0$ and $W_1 = 0$, $W_2 = 1$ we found the rms errors to be 0.0320 and 0.0326, respectively. Examination of the results (figures not shown) revealed that when the weight function is based solely on the curvature, it exhibits finestructure, due either to numerical error in its calculation or to the variation in $Q$ itself, and this structure is reflected in a fairly ragged grid and an associated increased error.

While the aforementioned parameter study provides useful insight into the grid generation process and its interaction with the numerical solution for $Q$, perhaps the most important conclusion we can draw is that, relative to the fixed-grid calculations, there is very little sensitivity of the accuracy of the method to the particular choice of grid parameters. For example, while we have shown that both the grid structure and the errors vary significantly as $\lambda_w$ is increased from 20 to 200 ($E_{\text{rms}}$ varies from 0.0281 to 0.0228), this variation is very small compared to the difference between any of these adaptive results and those of the fixed grid ($E_{\text{rms}} = 0.0714$). Thus, in applying CDGA to a particular problem, there is no need to search for particular grid parameters to make the method work; some choices will yield better results than others, but any reasonable selection will produce solutions clearly superior to those attainable on a fixed grid (assuming, of course, that the problem was suitable for solution by an adaptive technique to begin with).

**Fig. 20.** (a) As per Fig. 15a, but with $\lambda_0 = 100$. $E_{\text{rms}} = 0.0409$.
(b) As per Fig. 15b, but with $\lambda_0 = 100$. 
One deficiency of Doswell’s kinematic frontogenesis problem as a test case for the CDGA method is that the optimum gridpoint distribution is rather simple; the resolution should be high in the center of domain and reduced everywhere else. Thus, it is of interest to consider the performance of the method on more complex problems, in particular those for which multiple zones of high resolution are required. We have conducted several experiments using a modified form of the kinematic frontogenesis model in which the modified rankine vortex used to advect $Q$ is replaced by a velocity field of solid-body rotation, and the initial $Q$ distribution consists of four cones (minimum value of 0 (zero), maximum value of 1) placed on a zero-$Q$ background. The cones should be rotated about the center of domain without change in shape or magnitude. The results obtained on a fixed grid of $41^2$ points (Fig. 21a) are fairly typical (see, for example, Smolarkiewicz 1982) for this type of experiment, revealing considerable distortion of the cones, together with a large reduction in $Q_{\max}$ and the existence of large dispersive ripples trailing the cones. In contrast, the adaptive results using the same number of grid points (Figs. 21b and 21c) show a greatly improved maintenance of $Q_{\max}$, a smaller degree of distortion, and a marked reduction in the magnitude of the dispersive ripples. Thus, the CDGA method does not appear to have any difficulty in handling problems requiring several zones of high resolution. This conclusion is further supported by results presented in Part II in which the motion of multiple (up to three) barotropic vortices is successfully simulated using the CDGA technique.

5. The CDGA technique in two dimensions:

The evolution of a buoyant thermal in a neutral environment

The previous section has shown the impressive gains in accuracy that can be achieved using the CDGA method. It is true, however, that the problem studied was an extremely simple one, and one is thus led naturally to ask whether the method will enjoy such success when applied to situations of more direct relevance to the atmosphere. To provide at least a partial answer to this question, we have applied CDGA to model the evolution of a dry slab-symmetric thermal in a neutrally stable (isentropic), Boussinesq atmosphere.

We use the streamfunction–vorticity formulation [see, for example, Roach (1982)], in which case the governing equations are

$$\frac{\partial \xi}{\partial t} = -u \frac{\partial \xi}{\partial x} - w \frac{\partial \xi}{\partial z} - \frac{\partial \theta'}{\partial \eta} + \nu \nabla^2 \xi, \tag{5.1}$$

$$\frac{\partial \theta'}{\partial t} = -u \frac{\partial \theta'}{\partial x} - w \frac{\partial \theta'}{\partial z} + \nu \nabla^2 \theta', \tag{5.2}$$

where

$$\xi = \frac{\partial u}{\partial z} - \frac{\partial w}{\partial x} = \nabla^2 \psi, \tag{5.3}$$

$\theta'$ is the potential temperature perturbation from the constant (in both space and time) reference state $\theta_0$, and all other terms have their usual meaning. The transformation to computational space is straightforward for the prognostic equations,

$$\frac{\partial \xi}{\partial \tau} = -u_E \frac{\partial \xi}{\partial \xi} - w_E \frac{\partial \xi}{\partial \eta},$$

$$-g \left(z_E \frac{\partial \theta'}{\partial \xi} - z_E \frac{\partial \theta'}{\partial \eta}\right) = \frac{1}{\theta_0 G^{1/2}} + \nu \nabla^2 \xi, \tag{5.4}$$

$$\frac{\partial \theta'}{\partial \tau} = -u_E \frac{\partial \theta'}{\partial \xi} - w_E \frac{\partial \theta'}{\partial \eta} + \nu \nabla^2 \theta', \tag{5.5}$$

where

$$u_E = [(u - x_c) z_a - (w - z_c) x_a] G^{-1/2}, \tag{5.6a}$$

$$w_E = [(w - z_c) x_a - (u - x_c) z_a] G^{-1/2}. \tag{5.6b}$$

The use of a nonorthogonal grid does result in quite complicated expressions for the second derivative (TWM, p. 126),

$$\xi_{xx} = (z_E^2 \xi_{xx} - 2 z_E \xi_{x} \xi_{\eta} + \xi_{\eta}^2) G^{-1}$$

$$+ a_1 (x_E^2 \xi_{x} - x_E \xi_{\eta} + a_2 (z_E^2 \xi_{xx} - z_a \xi_{x}), \tag{5.7a}$$

$$\xi_{zz} = (x_E^2 \xi_{x} - 2 x_E \xi_{x} + x_E^2 \xi_{xx}) G^{-1}$$

$$+ b_1 (x_E \xi_{x} - x_E \xi_{\eta} + b_2 (z_E \xi_{xx} - z_a \xi_{x}), \tag{5.7b}$$

<table>
<thead>
<tr>
<th>$W_z$</th>
<th>$E_{mod}$</th>
<th>$E_{rms}$</th>
<th>$E_{max}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.0092</td>
<td>0.0304</td>
<td>0.395</td>
</tr>
<tr>
<td>0.1</td>
<td>0.0079</td>
<td>0.0269</td>
<td>0.327</td>
</tr>
<tr>
<td>0.5</td>
<td>0.0092</td>
<td>0.0286</td>
<td>0.295</td>
</tr>
<tr>
<td>1.0</td>
<td>0.0085</td>
<td>0.0281</td>
<td>0.296</td>
</tr>
<tr>
<td>2.0</td>
<td>0.0098</td>
<td>0.0304</td>
<td>0.324</td>
</tr>
<tr>
<td>10.0</td>
<td>0.0103</td>
<td>0.0319</td>
<td>0.319</td>
</tr>
</tbody>
</table>
where

\( a_1 = (z_\eta^2 z_\xi - 2z_\eta z_\xi z_{\xi\eta} + z_\eta^2 z_{\eta\eta})G^{-3/2} \)  \hspace{1cm} (5.7c)

\( a_2 = (z_\eta^2 x_\xi - 2z_\eta z_\xi x_{\eta\eta} + z_\eta^2 x_{\eta\eta})G^{-3/2} \)  \hspace{1cm} (5.7d)

\( b_1 = (x_\eta^2 z_\xi - 2x_\eta z_\xi z_{\xi\eta} + x_\eta^2 z_{\eta\eta})G^{-3/2} \)  \hspace{1cm} (5.7e)

\( b_2 = (x_\eta^2 x_\xi - 2x_\eta x_\xi x_{\eta\eta} + x_\eta^2 x_{\eta\eta})G^{-3/2} \)  \hspace{1cm} (5.7f)

This increase in complexity has been noted previously by Sharman et al. (1988, p. 1129), who go on to suggest that orthogonal coordinate systems are to be preferred. Although we feel that this is a reasonable conclusion for their application, for our purposes it is too restrictive, as it would greatly inhibit the ability of the grid to respond to the weight function. While it is true that the use of a nonorthogonal coordinate system may add to the truncation error (and at this stage we can make no definite statements about how large this increase may be when we are dealing with second, rather than first, derivatives), the results of section 4 suggest that, for reasonable grids, this increase will be small compared to the other components of the truncation error. The results presented below tend to support this view quite strongly. The issue of efficiency is another question entirely, and we agree with Sharman et al. that...
the use of numerically generated grids may provide a further advantage to the fully compressible models, the latter of which represent a completely prognostic system and thus do not involve the solution of an elliptic equation to obtain the pressure field.

Having derived the transformed governing equations, the task of discretization is straightforward. We use a nonstaggered grid, together with second-order central differencing for all spatial derivatives, and time advancement of the solution is achieved using the algorithm given in section 2. The diagnostic equation for the streamfunction is solved iteratively using the same algorithm that solves the grid-generator equations. Boundary conditions are provided by assuming that the boundaries are impermeable (zero normal component of velocity) and exert zero stress (normal derivative of the tangential velocity is zero: vorticity is zero). The boundary temperature is held fixed at \( \theta_0 \), and in all the calculations \( \theta_0 = 300.0 \) K. The model runs begin from a state of no motion, with an imposed “hot bubble” in the lower half of the domain (Fig. 22). Specifically, the temperature excess is 0.5 K at the center of the bubble, decreasing linearly with radius to zero at \( r = 200 \) m, and the bubble center is 250 m above the lower boundary. The grid generator is exactly that of the previous section, as is the weight function, with the exception that \( Q \) is replaced by \( \theta' \).

Before presenting the results themselves, it is important to realize that the thermal evolution problem is a very difficult one for standard finite-difference techniques. Very strong temperature gradients are generated by differential vertical advection, and these transition zones are almost impossible to resolve as their scale is determined by the level of mixing present, which is usually small. The advection of these zones by centered-difference algorithms results in large, spurious oscillations (Smolarkiewicz 1983, 1984; Bott 1989). In simple passive advection experiments these oscillations do not grow indefinitely, and the solution remains stable. However, in the thermal studies, the oscillations produce erroneous vorticity fields, which contort the temperature field further, resulting in even greater oscillations, and a positive feedback loop is established leading finally to numerical instability (Pearson and O'Connell 1977). The oscillations may be avoided by introducing extra diffusion into the problem, but the difficulty here is that this “artificial viscosity” often needs to be larger than the desired level of physical viscosity. Similarly, use of first-order upwind differencing for the advection terms yields monotone (nonoscillatory) results, but the associated truncation error leads to an excessively large degree of artificial diffusion in the calculations. Current work (Smolarkiewicz and Clark 1986; Dietachmayer 1987; Carpenter et al. 1990) has shown that the use of sophisticated non-linear advection algorithms yields nonoscillatory results with only small levels of artificial diffusion, but at a cost of increased programming complexity.

In light of the above considerations, we choose \( \nu = 2.0 \) m\(^2\) s\(^{-1}\), and emphasize that this choice is dominated not by physical but computational requirements. The computational domain is 800 m across by 1000 m high, and we present contour plots of the potential temperature perturbation field (with contour interval of 0.02 K), together with plots showing the gridpoint distribution.

As a reference experiment, we perform a relatively high-resolution (61 × 61 grid points) fixed-grid calculation with results shown in Figs. 23a–c. They clearly show the nonlinear steepening of the leading edge of the thermal, the evolution of the characteristic mushroom shape with cold-air entrainment into the rear, and the general expansion of the thermal with time. These features are well known, and the reader is referred to the literature (e.g., Lilly 1962, 1964) for details. From a computational standpoint, we note that the temperature field is smooth at all times, due to the large value of \( \nu \) used in this study, and that the minimum perturbation temperature value never falls below 0.0 K, as one would expect given (5.2) and the imposed boundary conditions. These observations suggest that the current grid provides sufficient resolution for this problem.

We turn now to the solution obtained using a relatively coarse (31 × 31) fixed grid (Figs. 24a–c). After 5 min, the solution is still reasonably accurate (Fig.
24a), but the leading edge of the thermal is much broader, as one would expect. By 10 min (Fig. 24b) the solution has markedly deteriorated. The leading edge is still smeared, and there are now significant

trailing oscillations in the entrainment region, together with a ragged structure along the edge of the thermal and a minimum value of $\theta$ significantly less than zero. We also note the appearance of spurious off-axis ex-

FIG. 23. (a) The solution for the temperature perturbation field after 5 min of integration, using a $61 \times 61$ fixed grid. $\theta_{\text{min}} = 0.0$ K. (b) As per (a), but after 10 min of integration. $\theta_{\text{min}} = 0.0$ K. (c) As per (a), but after 15 min of integration. $\theta_{\text{min}} = 0.0$ K.
trema in the temperature field. The results are similar at 15 min (Fig. 24c). The spurious off-axis extrema have been almost eliminated by viscous damping, but the minimum value of $\theta'$ has decreased yet further.

The formation of local temperature extrema in the descending arms of the thermal is correctly modeled, but again, the inside of the thermal is extremely ragged, and the oscillations remain. By 20 min (results not

FIG. 24. (a) The solution for the temperature perturbation field after 5 min of integration, using a $31 \times 31$ fixed grid. $\theta_{\text{min}}' = 0.0$ K. (b) As per (a), but after 10 min of integration. $\theta_{\text{min}}' = -0.019$ K. (c) As per (a), but after 15 min of integration. $\theta_{\text{min}}' = -0.028$ K.
shown) the simulation has recovered slightly for two reasons. First, the high level of diffusion acts to continually damp out oscillations and spurious extremes, and second, the continued expansion of the thermal implies that more and more of its structure can be adequately resolved by the grid.

Finally, the results using the CDGA method on a $31\times31$ grid (with $\lambda_H = 20$, $\lambda_O = 0$, $W_1 = 1$, $W_2 = 0.1$, $\lambda_U = 0.002$ K. (c) The gridpoint distribution for the adaptive model with $31\times31$ grid points after 5 min of integration. (d) As per (b), but after 10 min of integration. $\theta_{\text{min}} = -0.003$ K. (e) As per (c), but after 10 min of integration. (f) As per (b), but after 15 min of integration. $\theta_{\text{min}} = -0.004$ K. (g) As per (c), but after 15 min of integration.
and NS = 4) are shown in Figs. 25a–g. The initial grid (Fig. 25a) has increased resolution both in and, more particularly, around the edge of the thermal where the second derivative component of the weight function is extremely large. At 5 min (Fig. 25b), despite the presence of very small-amplitude noise together with a
The slight drop in $\theta_{\text{min}}$ below zero, the results are very good, with no artificial broadening of the leading edge (compare Fig. 24a). The grid has been able to track the thermal as it rises, and has evolved from an initially symmetric distribution to a very asymmetric one with increased resolution along the leading edge. At 10 min the improvement in accuracy is more dramatic (Fig. 25d), as the trailing oscillations and ragged inner edge of the fixed-grid calculations are totally absent (compare Fig. 24b). There are no spurious off-axis extremes and, again, no artificial broadening of the leading edge. The grid response has continued to evolve, and there are now high-resolution regions along the arms of the thermal. This high degree of accuracy continues through 15 min (Fig. 25f). The lack of oscillations in the entrainment region is worthy of some comment, given that the grid resolution there is worse in the adaptive case than the fixed. We surmise that the increased resolution in the thermal itself is able to resolve the steep gradients sufficiently well that downstream oscillations are not generated. Thus, the current grid generator-weight function combination has shown itself to be sufficiently intelligent to increase the local resolution not where the errors are most easily manifest, but rather where they are first likely to be created. At 20 min the adaptive results are still excellent (results not shown), although a small element of raggedness is just beginning to appear in the bottom of the arms of the thermal. This slight reduction in accuracy is to be expected, as the continual expansion of the thermal means that a larger and larger percentage of the grid cells are asking for high resolution, and hence, on average they all receive somewhat less.

6. Conclusions

We have illustrated how CDGA works in practice by applying it to the one-dimensional Burgers’ equation. It was shown that the motion of the grid points resulted in improvements in accuracy over and above those expected due to increased resolution. This extra improvement in accuracy was linked to a reduction in complexity of the governing equation in the transformed system [recall that the grid points moved at just under the shock speed and thus (2.2) was reduced to $\partial u/\partial \tau \approx 0$], and as such, there is no a priori reason why it could not occur in more complex problems. For example, in a numerical weather prediction (NWP) model the grid points might track a moving front, reducing the temperature equation to $\partial \theta/\partial \tau \approx 0$. The problem of gridpoint oscillations was considered in some detail, and was essentially solved without recourse to ad hoc fixes. The importance of the method of discretization of the grid generator equation was highlighted, and work is currently underway on a multidimensional grid generator based on the discrete equation concept.

The kinematic frontogenesis calculations provided quantitative measures of the improvement in accuracy that can be achieved with the CDGA method, revealing that, at coarse resolution, the adaptive results were as accurate as those on a fixed grid with three times as many points. In this case not only was the accuracy itself much higher, but the order of accuracy (in the sense of error reduction gained per unit increase in number of grid points) was also greater than the fixed-grid model. We note that this last result did not apply at high resolutions, but argue that this is not relevant for realistic atmospheric models. An obvious example is the representation of tropical cyclones in NWP models. Most NWP models run with resolutions of the order of 100 km, and thus have approximately one grid interval within the radius of maximum winds. It is clear that, if the modeling of the tropical cyclone is regarded as one of the aims of the NWP model, then the model has very coarse resolution indeed. Similar comments can be made concerning the representation of fronts in NWP models and thunderstorms in mesoscale models. The sensitivity of the CDGA performance to variations in the grid-generator parameters was examined in some detail, with the important result that there is little such sensitivity over a wide range of parameter values.

The success of the adaptive thermal simulations proved that the extra complexity of the dynamic equations (involving nonlinearity and the solution of both parabolic and elliptic partial differential equations) presents no extra difficulty to the CDGA technique. The rapid, but smooth, evolution of the gridpoint distribution across virtually the entire computational domain was also evident, which was not the case in the kinematic frontogenesis experiments where one knew beforehand that the resolution in the center (only) should be increased. The observation that the tricky problem of spurious oscillations in advected fields could be virtually eliminated by adaptive techniques is in accord with the view expressed by Thompson (1985, p. 24) that “... when the grid is right, most numerical solution methods work well.” This is not to say that research into more and more elaborate difference representations of the governing equations is to be discouraged, but rather that the adaptive technique provides an additional avenue of attack, particularly for problems that are known to be numerically difficult. The introduction of adaptive techniques may change current perceptions of the value of various difference schemes. For example, first-order upwining may become a viable option as a monotonic advection algorithm, as the coefficient of artificial viscosity would only be significant in regions of relatively coarse resolution where the variation in the solution field is small, and hence, the effect of diffusive processes is minimal.

One significant advantage of adaptive techniques over traditional nested grid methods is their ability to allocate multiple zones of high resolution in response
to multiple small-scale flow features, even in cases where the number of such features changes in time. This ability is more clearly demonstrated in Part II in which the CDGA approach is used to achieve simultaneous high resolution of multiple (up to three) small-scale features, in this case vortices. Other calculations examining the phenomena of vortex merging show that the grid-generation algorithm has no difficulty in taking separate zones of high resolution and merging them into one.

We acknowledge that the current work is only the first step in applying CDGA to meteorological modeling. A critical issue that we have not yet addressed is the efficiency of the method in terms of accuracy achieved per unit expenditure of computer processing time. This omission is deliberate as we do not feel that the models constructed so far provide a reasonable framework for such an evaluation. There are several reasons for this belief. First, the solution of the grid-generator equations, which is the most expensive component of the technique, is not dependent on the particular governing equations being solved, and hence, the efficiency of the method is reduced for simpler problems such as the ones previously examined. In the kinematic frontogenesis problem, for example, the governing equation is simply one advection equation, while the grid generation requires the solution of two elliptic problems, which is obviously much more expensive. Further, the cost of the evaluation of the metric terms required for the transformed equations is continually reduced, in percentage terms, as the number of governing equations is increased; if we have a system with n advected quantities, the effective velocity components are still calculated only once. Second, our experiments have shown that by using relatively large values of NS we may effectively ignore the orthogonality constraint in the grid-generation process, thus opening the way for the design of simpler and, most important, faster generators than the one used here. Finally, we might achieve a very large reduction in the computational overhead of the method if we follow Skamarock and update the grid only every so often, rather than every time step as is done at the present.

However, because the CDGA method cannot be judged without at least some indication of its efficiency, we present the results of some preliminary calculations, the details of which are provided in Part II.

For the reasons given above, the CDGA technique as implemented here is not cost effective, in terms of CPU time required to achieve a given level of accuracy, when applied to the kinematic frontogenesis problem of section 4. In fact, to achieve a prescribed level of accuracy the adaptive model runs two orders of magnitude slower than the equivalent fixed grid model. As predicted above, much of the overhead is contained in the grid-generation procedure, which accounts for over 80% of the total CPU time for a given model run. It is very important to realize that this rather gloomy picture is by no means representative of the effectiveness of the CDGA technique in other implementations and/or applications. In Part II we develop a fast discrete-equidistribution grid generator, which we apply to a barotropic tropical cyclone model. This model is used to examine the dynamics of multiple-vortex interaction, and the results are compared to an otherwise identical fixed-grid model. Because both the model formulation and the grid-generation procedure used are very different to the approach taken here, we give only a summary of the results, and defer the details to Part II. In Table 5, taken from Part II, we present results from seven experiments, four using the fixed-grid model with between 41$^2$ and 121$^2$ grid points, and three using the adaptive model with between 41$^2$ and 81$^2$ grid points. The table gives the maximum and minimum values of the height h field and the east–west velocity component u after 72 h of model integration, together with the total CPU time for the calculations, all of which were performed on a 25 MHz 386 IBM PC-AT. Also shown is the initial values of the h and u extremes. It is clear from the fixed grid calculations that as the resolution is increased, the model is better able to resolve the vortices; and thus, the minimum value of the height field is better maintained. Taking maintenance of $h_{\text{min}}$ as a reasonable guide to model accuracy, we see that the two higher-resolution (61$^2$ and 81$^2$) adaptive experiments are more accurate than any of the fixed-grid

<table>
<thead>
<tr>
<th>Model</th>
<th>$h_{\text{max}}$</th>
<th>$h_{\text{min}}$</th>
<th>$u_{\text{max}}$</th>
<th>$u_{\text{min}}$</th>
<th>Total CPU time</th>
<th>CPU time per step</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial</td>
<td>4200</td>
<td>3700</td>
<td>52.89</td>
<td>-50.64</td>
<td>1 h 13 min</td>
<td>20.0 s</td>
</tr>
<tr>
<td>F 41$^2$</td>
<td>4264</td>
<td>4112</td>
<td>25.7</td>
<td>-22.7</td>
<td>4 h 18 min</td>
<td>36.4 s</td>
</tr>
<tr>
<td>F 61$^2$</td>
<td>4362</td>
<td>4032</td>
<td>31.8</td>
<td>-30.5</td>
<td>10 h 18 min</td>
<td>57.0 s</td>
</tr>
<tr>
<td>F 81$^2$</td>
<td>4265</td>
<td>3965</td>
<td>38.0</td>
<td>-36.6</td>
<td>39 h 45 min</td>
<td>128.0 s</td>
</tr>
<tr>
<td>F 121$^2$</td>
<td>4262</td>
<td>3855</td>
<td>45.2</td>
<td>-43.5</td>
<td>5 h 27 min</td>
<td>21.6 s</td>
</tr>
<tr>
<td>A 41$^2$</td>
<td>4280</td>
<td>3865</td>
<td>39.9</td>
<td>-41.4</td>
<td>20 h 28 min</td>
<td>43.5 s</td>
</tr>
<tr>
<td>A 61$^2$</td>
<td>4263</td>
<td>3743</td>
<td>47.9</td>
<td>-46.9</td>
<td>52 h 34 min</td>
<td>77.9 s</td>
</tr>
</tbody>
</table>

The letter F denotes calculations performed with the fixed-grid version of the model, A denotes calculations performed with the adaptive version of the model, and quoted CPU times are based on calculations performed on a 25 MHz 386 IBM PC-AT.
results. The adaptive calculation with 41^2 points lies, in terms of accuracy, somewhere between the fixed-grid results using 81^2 and 121^2 points. Thus, we have shown that, in this application at least, the CDGA technique is clearly cost effective, yielding results equivalent in accuracy to those obtained using a fixed-grid model in no more than half the time, and perhaps as little one-sixth.

As with any relatively new technique, the scope for future research is very broad indeed. The design of appropriate weight functions for general flows is obviously an important concern; another is the extension of the method to three dimensions. There seem to be no conceptual difficulties in such an extension (see TWM, appendix B for the derivation of a variational three-dimensional grid generator), but questions such as the additional expense involved, both in the computation of additional terms in the governing equations and in the generation of the grid, have yet to be considered. These issues can be resolved only by numerical experimentation, in which case it seems prudent to begin with simpler two-dimensional experiments, and then build forward upon those results. Even if three-dimensional grid generation was to prove prohibitively expensive for current computers, two-dimensional grid adaption could still be gainfully applied to three-dimensional models. One example would be the development of a three-dimensional baroclinic tropical cyclone model, which used a fixed vertical distribution of grid points (perhaps with stretching near the sea surface to better resolve the boundary layer) together with a grid using dynamical adaption in the horizontal to achieve high resolution over the cyclone itself. In short, grid adaption could be applied in the horizontal alone, in much the same way as traditional nesting has been applied for several decades. Another issue that will not have arisen in the previous atmospheric and aeronautical applications of CDGA is the question of how to extend the many parameterizations (e.g., turbulence, cloud physics, radiation) used in meteorological modeling to a model in which the grid spacing, and hence the length scale for which the parameterization is designed, may vary quite significantly across the domain. However, the above difficulties notwithstanding, we find that the performance of the CDGA method, even judged on the basis of these early experiments, is sufficiently encouraging to conclude that it has tremendous potential for application to a wide variety of problems in meteorological modeling.

Acknowledgments. The first author is extremely grateful to Drs. Douglas Lilly and Tzvi Gal-Chen, who jointly arranged for his postdoctoral fellowship at CIMMS, during which time the bulk of this work was completed. The members of the COMMA (Central Oklahoma Mesoscale Modeling and Analysis) project have our sincere thanks for the enthusiasm and support they lent to this work. Both authors enjoyed valuable discussions with Kriste Lyon, who also provided some of the error calculations, with Joe Bradley of the Geosciences Computing Network, and with Dr. Joe Thompson, Mississippi State University. The paper was significantly improved by the comments of the two anonymous referees. Graeme Hubbert provided a great deal of assistance with the analysis of the results on the various computer systems used within BMRC. This research was supported by the National Oceanic and Atmospheric Administration, Environmental Research Laboratories under Cooperative Agreement NA85RAH05046 to the University of Oklahoma, and by the National Science Foundation under Grant ATM86-57013 to the second author. The simulations and data analysis were performed on Alliant FX/4 and VAX 11/785 computers at the Geosciences Computing Network, University of Oklahoma, and Facom M200 mainframe and IBM-AT computers at the Bureau of Meteorology Research Center, together with various VAX computers at Monash University.

APPENDIX

Numerical Solution of the Grid-Generator Equations (3.6) and (3.7)

Equation (3.7) is formulated for a general functional $F$. For our particular choice of $F$ we have the following relationships, required to evaluate (3.7):

\[ F = (1 + \lambda_W W^2)G + \lambda_O g_{12}^2 \]  
\[ G = g_{11} g_{22} - (g_{12})^2 \]  
\[ \frac{\partial F}{\partial g_{11}} = g_{22}(1 + \lambda_W W^2) \]  
\[ \frac{\partial F}{\partial g_{22}} = g_{11}(1 + \lambda_W W^2) \]  
\[ \frac{\partial F}{\partial g_{12}} = g_{22}[\lambda_O - (1 + \lambda_W W^2)] \]  
\[ \frac{\partial F}{\partial g_{21}} = g_{11}[\lambda_O - (1 + \lambda_W W^2)] \]  
\[ \frac{\partial}{\partial g_{mn}} \left( \frac{\partial F}{\partial g_{11}} \right) = \begin{cases} 1 + \lambda_W W^2, & \text{if } m = n = 2; \\ 0, & \text{otherwise.} \end{cases} \]  
\[ \frac{\partial}{\partial g_{mn}} \left( \frac{\partial F}{\partial g_{22}} \right) = \begin{cases} 1 + \lambda_W W^2, & \text{if } m = n = 1; \\ 0, & \text{otherwise.} \end{cases} \]  
\[ \frac{\partial}{\partial g_{mn}} \left( \frac{\partial F}{\partial g_{12}} \right) = \begin{cases} \lambda_O - (1 + \lambda_W W^2), & \text{if } m = 2 \text{ and } n = 1; \\ 0, & \text{otherwise.} \end{cases} \]
\[
\frac{\partial}{\partial g_{mn}} \left( \frac{\partial F}{\partial g_{21}} \right) = \begin{cases} 
\lambda_D - (1 + \lambda_w W^2), & \text{if } m = 1 \text{ and } n = 2; \\
0, & \text{otherwise.}
\end{cases}
\]

(A.9)

\[
\frac{\partial F}{\partial W} = 2G\lambda_w W
\]

(A.10)

\[
\frac{\partial}{\partial W} \left( \frac{\partial F}{\partial g_{11}} \right) = 2g_{11}\lambda_w W
\]

(A.11)

\[
\frac{\partial}{\partial W} \left( \frac{\partial F}{\partial g_{22}} \right) = 2g_{22}\lambda_w W
\]

(A.12)

\[
\frac{\partial}{\partial W} \left( \frac{\partial F}{\partial g_{12}} \right) = -2g_{21}\lambda_w W
\]

(A.13)

\[
\frac{\partial}{\partial W} \left( \frac{\partial F}{\partial g_{21}} \right) = -2g_{12}\lambda_w W
\]

(A.14)

and

\[
\frac{\partial W}{\partial x_i} = \left( \frac{\partial W}{\partial x_i} \frac{\partial x_m}{\partial x^m} - \frac{\partial W}{\partial x_m} \frac{\partial x_m}{\partial x^i} \right),
\]

(A.15)

where \( m = 1 \) if \( l = 2 \), and vice versa. From (3.1) and (3.2),

\[
g_{ij} = \sum_{p=1}^{2} \frac{\partial x_p \partial x_p}{\partial x^i \partial x^j}.
\]

(A.16)

For simplicity, we use here second-order central differencing; multidimensional grid generators based on discrete principles as described in section 2 will be presented in Part II. In this case, all the elements of \( q \) at a particular grid point \( i, j \) are independent of the values of \( x_{i,j} \) and \( y_{i,j} \), as are all the quantities defined in (A1.1)–(A1.15). In addition, the only derivatives of \( x \) and \( y \) in (3.7) that are dependent on the central values are \( \partial^2 x_i / \partial x^i \) and \( \partial^2 y_i / \partial y^i \).

Having discretized (3.7), we apply an analogous iterative technique to (2.21),

\[
0 = R_i^{(k+1)}
\]

\[
= R_i^{(k)} + \frac{\partial R_i^{(k)}}{\partial x_{ij}} \delta x_{ij} + \frac{\partial R_i^{(k)}}{\partial y_{ij}} \delta y_{ij} \quad l = 1, 2
\]

(A.17)

[where \( \delta x_{ij} = x_{ij}^{(k+1)} - x_{ij}^{(k)} \) and \( R_i \) is the residual of (3.7)] to find the next guess for \( x_{i,j} \) and \( y_{i,j} \). Because many of the terms in (3.7) are independent of \( x_{i,j} \) and \( y_{i,j} \), the calculation of the derivative of the residual is relatively straightforward.

\[\frac{\partial R_i}{\partial x_q} = -\delta_q \left( \frac{\partial F}{\partial g_{11}} + \frac{\partial F}{\partial g_{22}} \right) \]

\[+ 2 \sum_{k=1}^{2} \sum_{j=1}^{2} \frac{\partial x_j}{\partial x^j} \sum_{m=1}^{2} \left( \frac{\partial x_q}{\partial g_{kl}} \frac{\partial^2 F}{\partial g_{kl} \partial g_{ij}} + \frac{\partial^2 F}{\partial g_{km} \partial g_{ij}} \right) \]

\[+ \frac{\partial^2 F}{\partial g_{km} \partial g_{ij}} \]  

(A.18)

where \( l = 1, 2, q = 1, 2, \delta \) is the Kronecker delta, and for clarity we have dropped the grid index subscripts. Note that the above equations can be further simplified by making use of the symmetry \( g_{mm} = g_{nm} \).

Finally, we require boundary conditions on both \( x \) and \( y \). We consider only rectangular computational domains, although the method can (and has been) be used to generate grids for arbitrary domain shapes. High grid points are constrained to lie on their original boundary, but move along it in response to the second boundary condition, which is that the grid is required to be orthogonal at the boundaries. Fixed boundary conditions on both \( x \) and \( y \) would severely limit the adaptive properties of the grid, while the orthogonal property should simplify the application of boundary conditions on the prognostic variables.

REFERENCES


Dannenhoffer, J. F., and J. R. Baron, 1986: Robust grid adaption


