INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS VOL. 21, 723–733 (1995)

GENERALIZED DIFFERENTIAL AND INTEGRAL QUADRATURE AND THEIR APPLICATION TO SOLVE BOUNDARY LAYER EQUATIONS

C. SHU AND Y. T. CHEW

Department of Mechanical and Production Engineering, National University of Singapore, 10 Kent Ridge Crescent, Singapore 0511

AND

B. E. RICHARDS

Department of Aerospace Engineering, University of Glasgow, Glasgow G12 8QQ. U.K.

SUMMARY

Based on the work of generalized differential quadrature (GDQ), a global method of generalized integral quadrature (GIQ) is developed in this paper for approximating an integral of a function over a part of the closed domain. GIQ approximates the integral of a function over the part of the whole closed domain by a linear combination of all the functional values in the whole domain with higher order of accuracy. The weighting coefficients of GIQ can be easily determined from those of GDQ. Applications of GDQ and GIQ to solve boundary layer equations demonstrated that accurate numerical results can be obtained using just a few grid points.

KEY WORDS: global method; GDQ and GIQ; polynomial approximation; weighting coefficients; boundary layer solutions

1. INTRODUCTION

In seeking an efficient method using just a few grid points to get an accurate solution of a partial differential equation, Shu and Richards[1–3] have presented a global method of generalized differential quadrature (GDQ) based on the work of Bellman *et al.*[4] Using the same approach as given by Bellman *et al.* for differential quadrature (DQ), GDQ approximates any spatial derivative at a discrete point by a linear weighted sum of all the functional values in the whole domain. GDQ overcomes the difficulty of DQ in obtaining the weighting coefficients for the derivative approximation. In GDQ the weighting coefficients of the first-order derivative are given by a simple algebraic formulation while the weighting coefficients of the second- and higher-order derivatives are given by a recurrence relationship. GDQ can be considered as a version of the spectral method since it is also based on the analysis of a high-order polynomial approximation. The spectral method is well described in References 5 and 6. However, GDQ provides a more convenient way to calculate the weighting coefficients than the conventional spectral method and can also be applied to an arbitrary distribution of grid points. Application of the GDQ scheme to solve the incompressible Navier–Stokes equations[1–3] demonstrated that accurate numerical results can be obtained using a considerably smaller number of grid points.

Based on the same concept as GDQ, generalized integral quadrature (GIQ) is also developed and presented here. If a function is smooth in the whole domain, it can be approximated by a high-order polynomial in that domain. Then the integral of the function over a part of the whole domain can be

CCC 0271–2091/95/090723–11
© 1995 by John Wiley & Sons, Ltd.
Received June 1994
Revised March 1995

approximated by integrating the approximate high-order polynomial over this part of the whole domain. As a result, this approximation involves all the functional values in the whole domain with high order of accuracy even though the integral domain contains only two points. Obviously the key procedure to this approach is how to determine the weighting coefficients of this generalized integral quadrature. We will show in this paper that the weighting coefficients of GIQ can be easily obtained from those of the first-order derivative in GDQ.

The boundary layer approximation is still an interesting area in CFD because it greatly reduces the computational effort compared with a Navier–Stokes solver. The boundary layer equations can be solved when the dependent variable is a primitive variable or the streamfunction. When the streamfunction is introduced, the continuity equation can be dropped from the solution procedure. Accordingly, the order of the differential equations is increased by one, which may create difficulties in dealing with the boundary conditions. For this case the single GDQ method can be used to solve the boundary layer equations. On the other hand, the use of a primitive variable as the dependent variable is attractive since it enables the two-dimensional methods to be extended to the three-dimensional case directly. The difficulty then is the coupling of the continuity equation with the momentum and energy equations. The reason for not using the integral form of the continuity equation is that the normal velocity obtained by integrating the continuity equation along the normal co-ordinate with the use of a classical numerical method is less accurate, because some integral domains do not contain sufficient grid points. As will be shown in this paper, the GIQ technique can provide a promising way to obtain the normal velocity accurately by an explicit formulation derived from the integration of the continuity equation. The determination of the normal velocity at any mesh point in the normal co-ordinate direction has the same order of accuracy. We will use both the GDQ and GIQ techniques in the normal direction for discretizing the derivatives and the integrals. In the streamwise direction both GDQ and low-order finite difference schemes can be used. It will be demonstrated in the paper that the GDQ-GIQ method works uniformly well for both the case where the dependent variable is the streamfunction and the case where the dependent variable is a primitive variable.

2. GENERALIZED DIFFERENTIAL QUADRATURE (GDQ)

The GDQ approach was developed based on the differential quadrature (DQ) technique.[4] DQ approximates the first-order spatial derivative of a function with respect to a spatial co-ordinate at a given grid point as a weighted linear sum of all the functional values at all grid points in the whole domain of that spatial co-ordinate. This can be demonstrated by the following one-dimensional example. The first-order derivative of a smooth function $f(x, t)$ with respect to x at x_i can be approximated by DQ as

$$f_x(x_i, t) = \sum_{j=1}^{N} w_{ij}^{(1)} f(x_j, t), \quad i = 1, 2, 3, \ldots, N, \tag{1}$$

where N is the number of grid points in the whole computational domain. Obviously the key procedure to this approach is to determine the weighting coefficients $w_{i,j}^{(1)}$. Bellman *et al.*[4] suggested two ways to carry this out. The first is to let equation (1) be exact for test functions $g_k(x) = x^k$, $k = 0, 1, \ldots, N-1$, which leads to a set of linear algebraic equations

$$\sum_{j=1}^{N} w_{i,j}^{(1)} x_j^k = k x_i^{k-1}, \quad i = 1, 2, 3, \ldots, N, \quad k = 0, 1, 2, \ldots, N-1. \tag{2}$$

This equation system has a unique solution because its matrix is of Vandermonde form. Unfortunately, when N is large, the matrix is ill-conditioned and its inversion is difficult. The second method is similar to the first one with the exception that different test functions

$$g_k(x) = \frac{L_N(x)}{(x - x_k)L_N^{(1)}(x_k)}, \quad k = 1, 2, \ldots, N, \tag{3}$$

are chosen, where $L_N(x)$ is the Nth-order Legendre polynomial and $L_N^{(1)}(x)$ is the first-order derivative of $L_N(x)$. By choosing x_k to be the roots of the shifted Legendre polynomial, Bellman *et al.* obtained a simple algebraic formulation for calculating $w_{ij}^{(1)}$, but with the condition that the co-ordinates of grid points should be chosen as the roots of an Nth-order Legendre polynomial. From the literature, applications of DQ in engineering so far[4,7–9] have usually used Bellman's first method to obtain the weighting coefficients, because the grid points can be chosen arbitrarily. However, because of the drawback described above, the number of grid points used is less than or equal to 13. To overcome the drawbacks of DQ, GDQ was developed wherein the weighting coefficients are calculated by a simple algebraic formulation or by a recurrence relationship without any restriction on the choice of grid points.

GDQ is based on the analysis of a high-order polynomial approximation and of a linear vector space. According to the Weierstrass polynomial approximation theorem, a smooth function in a domain can be accurately approximated by a high-order polynomial. Following this approach, it is supposed that for a smooth problem the solution of a partial differential equation in a domain can be approximated by an (N-1)th-order polynomial. It is easy to show that a polynomial of degree less than or equal to N-1 constitutes an N-dimensional linear vector space V_N. From the analysis of a linear vector space there exists a set of base vectors in V_N. Here, if $r_k(x)$, $k = 1, 2, \ldots, N$, are the base polynomials, any polynomial in V_N can be uniquely expressed as a linear combination of $r_k(x)$, $k = 1, 2, \ldots, N$. Also, if all the base polynomials satisfy a linear constrained relationship such as equation (1), then so does any polynomial in the space. In other words, if all the base polynomials satisfy equation (1), then so does the solution of a partial differential equation which is approximated by an (N-1)th-order polynomial. In a linear vector space there may exist several sets of base polynomials. Each set of base polynomials can be expressed uniquely by another set of base polynomials. Thus, if one set of base polynomials satisfies a constrained relationship, then so does another set of base polynomials. It was found that if the base polynomial $r_k(x)$ is chosen to be x^{k-1}, then the same equation system (2) as given by Bellman's first method can be obtained, while if the base polynomial $r_k(x)$ is taken in the same form as equation (3), then the same formulation as given by Bellman's second method can be achieved. For generality, GDQ chooses the base polynomial $r_k(x)$ to be the Lagrange interpolation polynomial

$$r_k(x) = \frac{M(x)}{(x - x_k)M^{(1)}(x_k)}, \tag{4}$$

where

$$M(x) = (x - x_1)(x - x_2)\cdots(x - x_N), \qquad M^{(1)}(x_k) = \prod_{j=1, j\neq k}^{N} (x_k - x_j);$$

$x_1, x_2, \ldots, x_N$ are the co-ordinates of grid points and can be chosen arbitrarily. By substituting equation (4) into equation (1), one can obtain a simple algebraic formulation for calculating $w_{ij}^{(1)}$ with $j \neq i$. The weighting coefficients $w_{ii}^{(1)}$ can be obtained easily by choosing the base polynomial to be x^k with $k = 0$. Since $r_k(x)$ is the base polynomial in a polynomial vector space, with the calculated weighting coefficients, the discretization of derivatives by GDQ guarantees that the solution of a partial

differential equation is approximated by a high-order polynomial. For the discretization of the second- and higher-order derivatives we can set up a similar approximation to equation (1). Using the same analysis as for the first-order derivative discretization, we can obtain a recurrence relationship to calculate the weighting coefficients for the higher-order derivatives. It has also been shown that for a multidimensional case each direction can be treated in the same fashion as used in the one-dimensional case; for details see References 1–3. Here, for brevity, only the results of the one-dimensional case are given. The nth-order partial derivative of a function $f(x, t)$ with respect to x is approximated by GDQ as

$$f_x^{(n)}(x_i, t) = \sum_{k=1}^{N} w_{ik}^{(n)} f(x_k, t), \quad i = 1, 2, \ldots, N, \quad n = 1, 2, \ldots, N-1. \tag{5}$$

The weighting coefficients are determined as follows: for the first-order derivative

$$w_{ij}^{(1)} = \frac{M^{(1)}(x_i)}{(x_i - x_j) M^{(1)}(x_j)}, \quad i, j = 1, 2, \ldots, N, \quad j \neq i; \tag{6}$$

for the second- and higher-order derivatives

$$w_{ij}^{(n)} = n\left(w_{ij}^{(1)} w_{ii}^{(n-1)} - \frac{w_{ij}^{(n-1)}}{x_i - x_j}\right), \quad i, j = 1, 2, \ldots, N, \quad j \neq i, \quad n = 2, 3, \ldots, N-1; \tag{7}$$

for all the derivatives

$$w_{ii}^{(n)} = -\sum_{j=1, j\neq i}^{N} w_{ij}^{(n)}, \quad i = 1, 2, \ldots, , N, \quad n = 1, 2, \ldots, N-1. \tag{8}$$

It is obvious from equation (7) that the weighting coefficients of the second- and higher-order derivatives can be calculated from those of the first-order derivative completely.

3. GENERALIZED INTEGRAL QUADRATURE (GIQ)

GIQ is also based on the analysis of a high-order polynomial approximation and of a linear vector space. It is supposed that a function $f(x)$ is smooth in a closed domain $[a, b]$ which can be decomposed into N-1 intervals with grid points $x_1 = a, x_2, \ldots, x_N = b$. Using the Weierstrass polynomial approximation theorem, $f(x)$ can be approximated by an $(N$-1)th-order polynomial. In particular, when the functional values at N grid points are known, $f(x)$ can be approximated by the Lagrange interpolation polynomial which is related to the functional values at all grid points. As a result, the integral of this approximate polynomial over $[x_i, x_j]$ may involve functional values outside the integral domain. As a general case it is assumed that the integral of $f(x)$ over a part of the whole domain can be approximated by a linear combination of all the functional values in the whole domain in the form

$$\int_{x_i}^{x_j} f(x)\mathrm{d}x = \sum_{k=1}^{N} c_k^{ij} f(x_k), \tag{9}$$

where x_i and x_j are numbers that can be altered. When $x_i = a$ and $x_j = b$, equation (9) reduces to a conventional numerical integral, i.e. the integral domain is the whole domain containing all the functional values. In a similar fashion to the analysis in GDQ the $(N$-1)th-order polynomial which is an approximation to $f(x)$ constitutes an N-dimensional linear vector space. Thus, if all the base polynomials satisfy equation (9), then so does any polynomial in the space. If the Lagrange

interpolation polynomials $r_k(x)$, $k=1, 2, \ldots, N$, are chosen as the base polynomials, c_k^{ij} can be determined by

$$c_k^{ij} = \int_{x_i}^{x_j} r_k(x)\,\mathrm{d}x. \tag{10}$$

The expression for c_k^{ij} is very complicated. Therefore it is difficult to calculate c_k^{ij} accurately using equation (10). We will turn to another way to determine c_k^{ij}. Setting

$$f(x) = \frac{\mathrm{d}u(x)}{\mathrm{d}x}, \tag{11}$$

we see clearly that if $f(x)$ is an (N-1)th-order polynomial, $u(x)$ should be an Nth-order polynomial. It is supposed that $f(x)$ is approximated by an (N-1)th-order polynomial in form

$$f(x) = a_0 + a_1 x + \cdots + a_{N-1}x^{N-1}, \tag{12}$$

where $a_0, a_1, \ldots, a_{N-1}$ are constants. Integrating equation (11) from a constant c to the variable x, we obtain

$$u(x) = \int_c^x f(t)\,\mathrm{d}t + u(c) = F(x, c) + u(c), \tag{13}$$

where

$$F(x, c) = x\left(a_0 + \frac{a_1}{2}x + \cdots + \frac{a_{N-1}}{N}x^{N-1}\right) - c\left(a_0 + \frac{a_1}{2}c + \cdots + \frac{a_{n-1}}{N}c^{N-1}\right).$$

It can be shown that $F(x, c)$ is a member of an N-dimensional linear polynomial vector space. One set of its base polynomials can be chosen as

$$p_k(x) = (x - c)r_k(x - c), \quad k = 1, 2, \ldots, N, \tag{14}$$

where $r_k(x)$ is the Lagrange interpolation polynomial. Similarly to GDQ, we can set

$$F_x(x_i, c) = \sum_{j=1}^{N} \underline{a}_{ij} F(x_j, c), \quad i = 1, 2, \ldots, N, \tag{15}$$

where $\underline{a}_{ij}$ are the weighting coefficients and can be determined in the same fashion as used in GDQ. Substituting equation (14) into equation (15) gives the weighting coefficients as

$$\underline{a}_{ij} = \frac{x_i - c}{x_j - c} w_{ij}^{(1)}, \quad j \neq i, \tag{16a}$$

$$\underline{a}_{ii} = w_{ii}^{(1)} + \frac{1}{x_i - c}, \tag{16b}$$

where $w_{ij}^{(1)}$ is the weighting coefficient of the first-order derivative in GDQ. From equations (16), c cannot be chosen to be the co-ordinates of grid points x_i. This can be seen from equation (13): when $c=x$, there is no integral term involved in equation (13). Thus no method is needed to approximate the integral for $c=x$.

On the other hand, equation (15) can be written in the vector form

$$\mathbf{F}_x = \underline{A}\mathbf{F}, \tag{17}$$

where

$$\mathbf{F} = [F(x_1, c), F(x_2, c), \cdots, F(x_N, c)]^{\mathrm{T}}, \qquad \mathbf{F}_x = [F_x(x_1, c), F_x(x_2, c), \cdots, F_x(x_N, c)]^{\mathrm{T}}.$$

Substituting equation (11) and (13) into equation (17) and setting

$$\mathbf{f}^{\mathrm{I}} = \left[\int_c^{x_1} f(x)\,\mathrm{d}x, \int_c^{x_2} f(x)\,\mathrm{d}x, \cdots, \int_c^{x_N} f(x)\,\mathrm{d}x \right]^{\mathrm{T}},$$

we then obtain

$$\mathbf{f} = \underline{A}\mathbf{f}^{\mathrm{I}}. \tag{18}$$

Setting $W^{\mathrm{I}} = A^{-1}$, equation (18) can be further reduced to

$$\int_{x_i}^{x_j} f(x)\,\mathrm{d}x = \sum_{k=1}^{N} c_k^{ij} f(x_k), \quad \text{with } c_k^{ij} = w_{jk}^{I} - w_{ik}^{I}. \tag{19}$$

4. SOLUTIONS OF BOUNDARY LAYER EQUATIONS

For the present study it was found that the constant c used in equation (16) has little effect on the accuracy of the numerical results when $|c| \leq 0{\cdot}1$. In the following studies the value of c is taken as 0·01 and the GDQ–GIQ approach is applied to solve the boundary layer equations. Three test examples will be illustrated.

4.1. Blasius boundary layer

The first test example is the Blasius boundary layer which is governed by

$$\frac{\mathrm{d}^3 f}{\mathrm{d}\eta^3} + f\frac{\mathrm{d}^2 f}{\mathrm{d}\eta^2} = 0, \tag{20}$$

where f is the streamfunction. Equation (20) is subject to the following boundary conditions:

$$f = 0 \quad \text{and} \quad f_\eta = 0 \quad \text{when } \eta = 0, \tag{21a}$$

$$f_\eta = 2 \quad \text{when } \eta \to \infty. \tag{21b}$$

Equation (20) is a partial differential equation which can be solved by the GDQ scheme effectively. In order to demonstrate the application of the GDQ–GIQ approach and for simplicity, we set $u = f_\eta$ and introduce an unsteady term on the right side of equation (20). We then obtain the following two equations which can be solved by GDQ and GIQ:

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial \eta^2} + f\frac{\partial u}{\partial \eta}, \tag{22a}$$

$$f = \int_0^{\eta} u\,\mathrm{d}\eta + f(0). \tag{22b}$$

For numerical simulation the infinite interval in the η-direction is truncated to the finite interval [0, 3]. Using GDQ and GIQ in the domain [0, 3], equations (22a) and (22b) can be discretized respectively as

$$\frac{\mathrm{d}u_j}{\mathrm{d}t} = \sum_{k=1}^{M} w_{jk}^{(2)} u_k + f_j \sum_{k=1}^{M} w_{jk}^{(1)} u_k, \tag{23a}$$

$$f_j = \sum_{k=1}^{M} \left(w_{jk}^{I} - w_{1k}^{I} \right) u_k + f_1 \tag{23b}$$

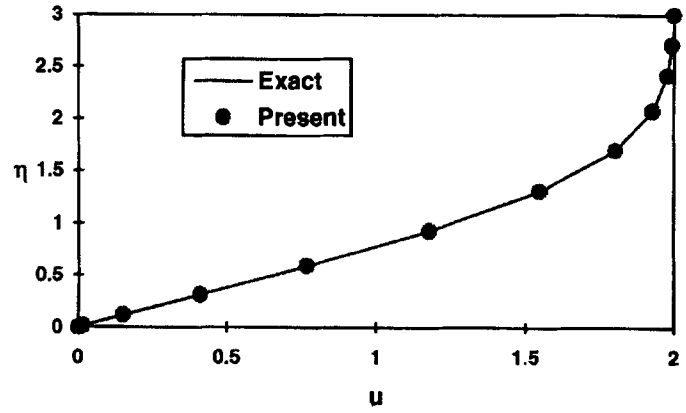


Figure 1. Velocity profile of Blasius boundary layer

for $j = 1, 2, \ldots, M$, where M is the number of grid points, $w_{ij}^{(m)}$ are the weighting coefficients of the mth-order derivative of the function with respect to η and w_{ij}^{I} are the weighting coeffients of the integral along the η-direction. The boundary conditions (21) become

$$u_1 = 0, \qquad u_M = 2, \qquad f_1 = 0, \tag{24}$$

which are easily implemented in the solution procedure. After spatial discretization in equation (23a) the resultant set of ordinary differential equations is solved by the four-stage Runge–Kutta scheme. The streamfunction in equation (23b) is approximated by the GIQ scheme. For the test problem the numerical results are very promising. Using 12 grid points and a time step size of $1{\cdot}3 \times 10^{-2}$, the numerical results, which require 385 time steps, are obtained very accurately. The obtained wall shear stress is 1·3286 while the exact value is 1·3284. Figure 1 shows the computed and the exact velocity profile of the Blasius boundary layer. Good agreement between computed and exact solutions has been achieved.

4.2. Two-dimensional Howarth boundary layer

Now we consider the two-dimensional Howarth boundary layer flow. The governing equation is[10]

$$\frac{\partial^3 f}{\partial \eta^3} + f\frac{\partial^2 f}{\partial \eta^2} + \beta(\xi)\left[1 - \left(\frac{\partial f}{\partial \eta}\right)^2\right] = 2\xi\left(\frac{\partial f}{\partial \eta}\frac{\partial^2 f}{\partial \xi \partial \eta} - \frac{\partial^2 f}{\partial \eta^2}\frac{\partial f}{\partial \xi}\right), \tag{25}$$

where $f(\xi, \eta)$ is the dimensionless streamfunction and is subject to the following boundary conditions:

$$f(\xi, 0) = \frac{\partial f}{\partial \eta}(\xi, 0) = 0, \qquad \frac{\partial f}{\partial \eta}(\xi, \eta) \to 1, \quad \text{when } \eta \to \infty. \tag{26}$$

Here (ξ, η) are the Levy–Lees co-ordinates; ξ increases in the freestream direction and η increases away from the wall. the function $\beta(\xi)$ is given by

$$\beta(\xi) = \frac{\xi}{\xi - 4}.$$

Using the same approach as for the Blasius boundary layer, we set $u = \partial f/\partial \eta$. then the third-order differential equation (25) is reduced to

$$\frac{\partial^2 u}{\partial \eta^2} + f\frac{\partial u}{\partial \eta} + \beta(\xi)(1-u^2) = 2\xi\left(u\frac{\partial u}{\partial \xi} - \frac{\partial u}{\partial \eta}\frac{\partial f}{\partial \xi}\right), \tag{27a}$$

$$f = \int_0^{\eta} u\,d\eta + f(0). \tag{27b}$$

For numerical simulation the infinite interval in the η-direction can be truncated to the finite interval [0, 6]. The spatial derivatives and the integral in the η-direction can be approximated by the GDQ–GIQ approach as

$$\left(\frac{\partial u}{\partial \eta}\right)_{i,j} = \sum_{k=1}^{M} \bar{w}_{jk}^{(1)} u_{ik}, \qquad \left(\frac{\partial^2 u}{\partial \eta^2}\right)_{i,j} = \sum_{k=1}^{M} \bar{w}_{jk}^{(2)} u_{ik}, f_{i,j} = \sum_{k=1}^{M} (w_{jk}^{I} - w_{1k}^{I}) u_{i,k} + f_{i,1},$$

where $\bar{w}_{ij}^{(m)}$ are the weighting coefficients of the mth-order derivative of the function with respect to η and w_{ij}^{I} are the weighting coefficients of the integral along the η-direction. The boundary conditions (26) become

$$u_{i,1} = 0, \qquad u_{i,M} = 1, \qquad f_{i,1} = 0. \tag{28}$$

If a second-order finite difference scheme is used at $(\xi_{i-1/2}, \eta_j)$ for the discretization of the derivative of u or f with respect to ξ, after linearization of the non-linear terms, equation (27a) can be reduced to

$$\mathbf{AU} = \mathbf{b}, \tag{29}$$

where $\mathbf{U} = (u_{i,2}, u_{i,3}, \ldots, u_{i,M-1})^{\mathrm{T}}$, $\mathbf{A}$ is a full matrix and $\mathbf{b}$ is a known vector. With equation (29) and the initial solution at the first station the boundary layer solution can be obtained by marching the solution along the ξ-direction. If the GDQ scheme is also applied in the ξ-direction, after linearization of the non-linear terms, we can obtain a similar form of the algebraic equation system to equation (29), but the vector $\mathbf{U}$ includes all the interior functional values $u_{i,j}$. Thus the marching technique cannot be used in this case. In the present study the GDQ and GIQ schemes are applied in both the ξ- and η-directions and for simplicity an unsteady term is introduced in equation (27a), resulting in

$$\frac{\partial^2 u}{\partial \eta^2} + f\frac{\partial u}{\partial \eta} + \beta(\xi)(1-u^2) = \frac{\partial u}{\partial t} + 2\xi\left(u\frac{\partial u}{\partial \xi} - \frac{\partial u}{\partial \eta}\frac{\partial f}{\partial \xi}\right). \tag{30}$$

The spatial discretization of equation (30) by GDQ leads to a set of ordinary differential equations which is then solved by the four-stage Runge–Kutta scheme. The streamfunction f is determined by the GIQ scheme.

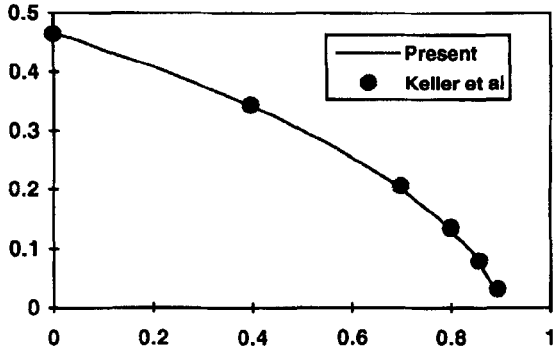


Figure 2. Wall shear stress distribution of Howarth boundary layer

The Howarth boundary layer flow as studied by Keller and Cebeci[10] has a separation point at $\xi = 0{\cdot}901$. Thus the computational domain in the ξ-direction should be $[0, b]$ with $b < 0{\cdot}901$ because of the Goldstein singularity. It was found that the GDQ–GIQ approach is very sensitive to the choice of b when b is close to the separation point. Actually, when b is taken as 0·90, the computation will diverge quickly after a few time steps, but when b is chosen as less than or equal to 0·894, the steady state resolution can be obtained fast and accurately. The convergence rate is very fast when b is far from the separation point and is slightly slower when b is very close to the separation point. Figure 2 displays the computational wall shear stress distribution of the Howarth boundary layer. The present numerical results are obtained using a mesh size of 11×12 and $b = 0{\cdot}894$; they agree well with the results given by the Keller box finite difference scheme using a large mesh of 51×121.

4.3. Unsteady boundary layer flow past an impulsively started circular cylinder

The third test example is the two-dimensional unsteady viscous flow past a circular cylinder started impulsively from rest. This problem has been chosen as a test example by many researchers for the study of unsteady boundary layer behaviour. Unlike the steady boundary layer equations, there are arguments as to whether there exists a finite time singularity in the solution of the unsteady counterparts. For this test problem some researchers[11,12] have claimed that there is a finite time singularity in the solution procedure while others[13,14] have suggested that there is no finite time singularity. The non-dimensional form of the governing equations for this problem is[11]

$$u_t + uu_x + vu_y = u_e(u_e)_x + u_{yy}, \tag{31}$$

$$v(x, y, t) = -\int_0^y u_x \, dy + v(x, 0, t), \tag{32}$$

with initial condition

$$u(x, y, 0) = u_e(x) = \sin(x), \quad y \neq 0,$$

and boundary conditions

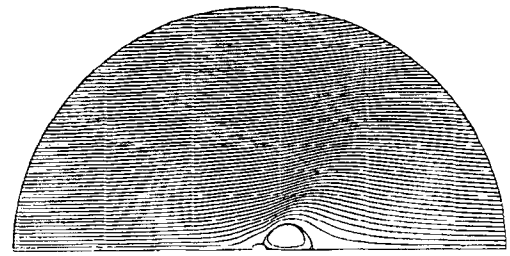
$$u(x, 0, t) = v(x, 0, t) = 0, \qquad u(x, \infty, t) = u_e(x) = \sin(x), \qquad u(0, y, t) = 0.$$

The computational domain in the y-direction can be obtained by truncating the infinite domain to $[0, 35]$. Using GDQ and GIQ in the y-direction, equations (31) and (32) can be discretized respectively as

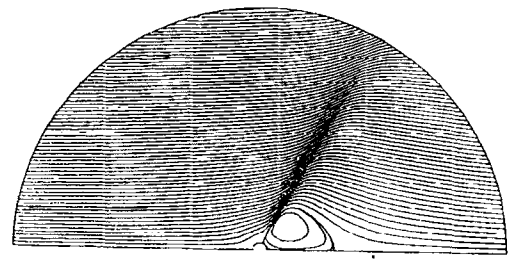
$$\frac{du_{i,j}}{dt} + u_{i,j}(u_x)_{i,j} + v_{i,j} \sum_{k=1}^{M} \bar{w}_{j,k}^{(1)} u_{i,k} = \sin(x)\cos(x) + \sum_{k=1}^{M} \bar{w}_{j,k}^{(2)} u_{i,k}, \tag{33}$$

$$v_{i,j} = -\sum_{k=1}^{M} (w_{j,k}^{I} - w_{1,k}^{I})(u_x)_{i,k} + v_{i,1} \tag{34}$$

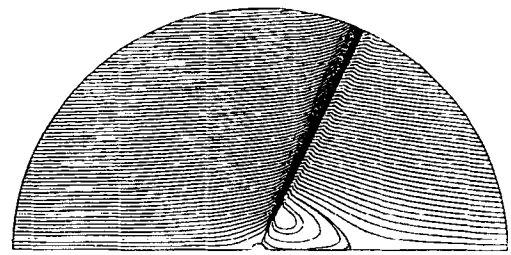
for $i = 1, 2, \ldots, N$ and $j = 1, 2, \ldots, M$, where N is the number of grid points in the x-direction, M is the number of grid points in the y-direction, $\bar{w}_{ij}^{(m)}$ are the weighting coefficients of the mth-order derivative of the function with respect to y and w_{ij}^{I} are the weighting coefficients of the integral along the y-direction. In the present study the derivative of u_x is discretized by a second-order finite difference scheme and the resultant ordinary differential equations for velocity u are solved by the four-stage Runge–Kutta scheme. Using equation (34), the velocity v can be given by the GIQ scheme. For numerical simulation the mesh size used is 81×31. It was found that reverse flow first starts at $\theta = 180^{c}$ and time $t = 0{\cdot}644$, which is in agreement with other researchers' results. As time



(a) t= 2.0



(b) t = 2.5



(c) t= 2.8

Figure 3. Computed streamlines past a circular cylinder

increases, the point of zero wall shear moves along the surface of the cylinder towards the steady state value $\theta_G = 104{\cdot}5°$ (the position of the Goldstein singularity).[11–15] However, the computation cannot reach steady state resolution, because the numerical instability breaks down the calculation at $t \approx 3{\cdot}0$. This is in agreement with the findings of some other researchers[11,12] but contradicts the

Table I. Comparison of times and positions of zero wall shear stress

References	180°	166°	146°	138°	124°	110°
Bar-Lev and Yang[15]	0·644	0·660	0·778	0·876	1·204	2·188
Cebeci[13]	0·640	0·660	0·780	0·872	1·192	2·200
Present	0·644	0·668	0·791	0·878	1·196	2·204

results obtained by Cebeci,[13,14] who claimed that there is no finite time singularity in the solution of unsteady boundary layer equations. From the calculated velocity components u and v the streamfunction can be computed accordingly. Figure 3 shows the computed instantaneous streamlines for three times, $t = 2{\cdot}0$, $2{\cdot}5$ and $2{\cdot}8$. Clearly there is a finite time singularity. The unsteady computation cannot reach the position of the Goldstein singularity. It should be noted that GDQ and GIQ are both global methods. They are very sensitive to any singularity in the flow field. Thus they are good for detecting a singularity. Table I lists the computed positions and times of zero wall shear stress. Also included in Table I are the boundary layer results given by Cebeci[13] and the Navier–Stokes results of Bar-Lev and Yang.[15] Obviously the present numerical results agree well with other researchers' results.

5. CONCLUSIONS

The global method of generalized integral quadrature (GIQ) has been presented in this paper based on the analysis of a polynomial linear vector space. If a function is continuous in the whole domain, then GIQ approximates the integral of the function over a part of the whole domain (including the case of a whole domain) by a linear sum of all the functional values in the whole domain. The weighting coefficients in GIQ can be determined from those of GDQ. Application of the GDQ–GIQ approach to solve boundary layer equations is very successful. If GDQ and GIQ are applied in all the spatial directions, accurate numerical results can be achieved using just a few grid points. It appears that the GDQ–GIQ approach can be extensively applied in the solution of boundary layer equations.

REFERENCES

1. C. Shu, 'Generalized differential–integral quadrature and application to the simulation of incompressible viscous flows including parallel computation', *Ph.D. Thesis*, University of Glasgow, 1991.
2. C. Shu and B. E. Richards, 'Application of generalized differential quadrature to solve two-dimensional incompressible Navier–Stokes equations', *Int. j. numer. methods fluids*, **15**, 791–798 (1992).
3. C. Shu and B. E. Richards, 'Parallel simulation of incompressible viscous flows by generalized differential quadrature', *Comput. Syst. Eng.*, **3**, 271–281 (1992).
4. R. Bellman, B. G. Kashef and J. Casti, 'Differential quadrature: a technique for the rapid solution of nonlinear partial differential equations', *J. Comput. Phys.*, **10**, 40–52 (1972).
5. D. Gottlieb and S. A. Orszag, *Numerical Analysis of Spectral Methods: Theory and Applications*, SIAM-CBMS, Philadelphia, PA, 1977.
6. C. Canuto, M. Y. Hussaini, A. Quarteroni and T. A. Zang, *Spectral Methods in Fluid Dynamics*, Springer, New York, 1987.
7. F. Civan and C. M. Sliepcevich, 'Application of differential quadrature to transport processes', *J. Math. Anal. Appl.*, **93**, 206–221 (1983).
8. F. Civan and C. M. Sliepcevich, 'Differential quadrature for multi-dimensional problems', *J. Math. Anal. Appl.*, **101**, 423–443 (1984).
9. S. K. Jang, C. M. Bert and A. G. Striz, 'Application of differential quadrature to static analysis of structural components', *Int. j. numer. methods eng.*, **28**, 561–577 (1989).
10. H. K. Keller and T. Cebeci, 'Accurate numerical methods for boundary layer flows, I. Two-dimensional laminar flows', in *Lecture Notes in Physics*, Vol. 8, Springer, Berlin, 1970, pp. 92–100.
11. A. Liakopoulos, 'Pseudospectral solutions of separated flows', *AIAA Paper 88-3643-CP*, 1988.
12. L. L. Van Dommelen and S. F. Shen, 'The spontaneous generation of the singularity in a separating laminar boundary layer', *J. Comput. Phys.*, **38**, 125–140 (1980).
13. T. Cebeci, 'The laminar boundary layer on a circular cylinder started impulsively from rest', *J. Comput. Phys.*, **31**, 153–172 (1979).
14. T. Cebeci, 'Unsteady boundary layers with an intelligent numerical scheme', *J. Fluid Mech.*, **163**, 129–140 (1986).
15. M. Bar-Lev and H. T. Yang, 'Initial flow field over an impulsively started circular cylinder', *J. Fluid Mech.*, **72**, 625–647 (1975).