INT. J. NUM. METH. HEAT FLUID FLOW, VOL. 6 NO. 2, 61–75 (1996)

SOLUTIONS OF THREE-DIMENSIONAL BOUNDARY LAYER EQUATIONS BY GLOBAL METHODS OF GENERALIZED DIFFERENTIAL-INTEGRAL QUADRATURE

C. SHU, Y. T. CHEW, B. C. KHOO AND K. S. YEO

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

ABSTRACT

The global methods of generalized differential quadrature (GDQ) and generalized integral quadrature (GIQ) are applied to solve three-dimensional, incompressible, laminar boundary layer equations. The streamwise and crosswise velocity components are taken as the dependent variables. The normal velocity is obtained by integrating the continuity equation along the normal direction where the integral is approximated by GIQ approach with high order of accuracy. All the spatial derivatives are discretized by a GDQ scheme. After spatial discretization, the resultant ordinary differential equations are solved by the 4-stage Runge-Katta scheme. Application of GDQ-GIQ approach to a test problem demonstrated that accurate numerical results can be obtained using just a few grid points.

KEY WORDS Glogal methods GDQ and GIQ Polymomial approximation Weighting coefficients Three dimensional Boundary layer solutions

INTRODUCTION

The boundary layer approximation is a useful mathematical model for many engineering problems. It can provide a good estimate of wall shear stress distribution for an attached flow. The simulation of the flow with a small separation region can be carried out by the concept of viscous-inviscid interaction, where the calculation of the viscous part is obtained by the boundary layer computation. The approach of viscous-inviscid interaction can greatly reduce the computational effort compared with a Navier-Stokes solver. For a numerical solution of boundary layer equations, low order finite difference schemes[4–10] are usually used to discretize the spatial derivatives, which can give accurate numerical results by using a large number of grid points. Boundary layer equations can be solved when they are expressed in physical coordinates or in transformed coordinates. Generally, the transformed form is favourable because it can remove the singularity occurring at the leading edge of the surface. Using transformed coordinates, most researchers prefer to use the stream function as the dependent variable. The major advantage of this is that the continuity equation can be dropped from the solution procedure as it is automatically satisfied. Accordingly, the order of the partial differential equations is increased by one, which may create difficulties in dealing with the boundary conditions. Another drawback is that the extension of the two-dimensional (2D) method to the three-dimensional (3D) case is not straightforward since the stream function does not exist for 3D cases. Some other researchers favour the use of the primitive variable (velocity) as the dependent variable enabling 2D methods to be extended to the 3D case directly. The difficulty then is the

0961–5539/96/020061–15$2.00

Received July 1994
Revised January 1995

coupling of the continuity equation with the momentum and energy equations. The continuity equation is usually solved by applying a finite difference scheme. The reason for not using the integral form of the continuity equation is that the normal velocity obtained by integrating the equation along the normal coordinate with the use of classical integral quadrature is less accurate because some integral domains do not contain sufficient grid points.

As will be shown in this paper, the global method of generalized integral quadrature (GIQ) can provide a promising way to obtain the normal velocity accurately by an explicit formulation derived from the integration of the continuity equation in the normal direction. GIQ approximates the integral over a part of the whole domain by a linear combination of all the functional values in the whole domain. Thus, the determination of the normal velocity at any mesh point involves all the functional information in the normal coordinate direction and has the same order of accuracy for all mesh points along the normal direction. The spatial derivatives of the boundary layer equations will be discretized by the global method of generalized differential quadrature (GDQ). Application of GDQ scheme to solve incompressible Navier-Stokes equations[1,2] demonstrated that accurate numerical results can be obtained using a considerably small number of grid points. Both GDQ and GIQ methods are based on the analysis of a high order polynomial approximation and the analysis of a linear vector space. Details of GDQ and GIQ will be given subsequently. Their application to solve a test three-dimensional laminar boundary layer problem will be demonstrated in detail, where transformed coordinates are used, and the velocity components are taken as the dependent variables.

GOVERNING EQUATIONS

The three-dimensional, incompressible, laminar boundary layer flows in the Cartesian coordinate system will be chosen to demonstrate the application of GDQ-GIQ method. For simplicity, the following unsteady equations are used as the governing equations. The steady state solution can be easily obtained by using an explicit scheme.

Continuity

$$\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}+\frac{\partial w}{\partial z}=0 \tag{1}$$

x-Momentum

$$\frac{\partial u}{\partial t}+u\cdot\frac{\partial u}{\partial x}+v\cdot\frac{\partial u}{\partial y}+w\cdot\frac{\partial u}{\partial z}=-\frac{1}{\rho}\cdot\frac{\partial p}{\partial x}+\nu\frac{\partial^2 u}{\partial y^2} \tag{2}$$

z-Momentum

$$\frac{\partial w}{\partial t}+u\cdot\frac{\partial w}{\partial x}+v\cdot\frac{\partial w}{\partial y}+w\cdot\frac{\partial w}{\partial z}=-\frac{1}{\rho}\cdot\frac{\partial p}{\partial z}+\nu\frac{\partial^2 w}{\partial y^2} \tag{3}$$

where x is the coordinate in the streamwise direction, z is the coordinate in the crosswise direction, y is the coordinate in the normal direction, and u, v, w are the velocity components in the x, y, z directions, respectively. The terms involving spatial derivatives of p, the pressure, represent the pressure gradient imposed on the boundary layer by the inviscid flow, which can be given by,

$$-\frac{1}{\rho}\cdot\frac{\partial p}{\partial x}=\frac{\partial Ue}{\partial t}+Ue\cdot\frac{\partial Ue}{\partial x}+We\cdot\frac{\partial Ue}{\partial z} \tag{4}$$

$$-\frac{1}{\rho}\cdot\frac{\partial p}{\partial z}=\frac{\partial We}{\partial t}+Ue\cdot\frac{\partial We}{\partial x}+We\cdot\frac{\partial We}{\partial z} \tag{5}$$

where Ue and We represent the inviscid velocity components in the x and z directions, respectively. The boundary conditions for (1)–(3) are given by,

$$u=0,\ w=0,\ v=0 \qquad \text{when } y=0 \tag{6a}$$

$$u=Ue(x,z),\ w=We(x,z) \qquad \text{when } y=\delta \tag{6b}$$

where δ is the boundary layer thickness. The boundary layer equations (1)–(3) can be solved in the Cartesian coordinate system. However, their solutions are singular at the leading edge of the surface, leading to the need of some specific treatment. This drawback can be removed by using the transformed coordinates. Using the following transformation

$$T=\frac{Ue\cdot t}{x}, \qquad \xi=x, \qquad \zeta=z, \qquad \eta=\sqrt{\frac{Ue}{v\cdot x}}\cdot y$$

$$F=\frac{u}{Ue} \qquad G=\frac{w}{We}$$

the boundary layer equations (1)–(3) are transformed to,

$$\xi\cdot\frac{\partial F}{\partial\xi}+\frac{\partial V}{\partial\eta}+\frac{F}{2}\cdot(1+K_1)+\frac{\xi\cdot We}{Ue}\cdot\frac{\partial G}{\partial\zeta}+G\cdot\left(K_3-\frac{K_2}{2}\right)=0 \tag{7}$$

$$\frac{\partial F}{\partial T}+\xi F\cdot\frac{\partial F}{\partial\xi}+V\cdot\frac{\partial F}{\partial\eta}+\xi G\cdot\frac{We}{Ue}\cdot\frac{\partial F}{\partial\zeta}+K_1\cdot(F^2-1)+K_2\cdot(FG-1)=\frac{\partial^2 F}{\partial\eta^2} \tag{8}$$

$$\frac{\partial G}{\partial T}+\xi F\cdot\frac{\partial G}{\partial\xi}+V\cdot\frac{\partial G}{\partial\eta}+\xi G\cdot\frac{We}{Ue}\cdot\frac{\partial G}{\partial\zeta}+K_3\cdot(G^2-1)+K_4\cdot(FG-1)=\frac{\partial^2 G}{\partial\eta^2} \tag{9}$$

where,

$$V=\sqrt{\frac{x}{v\cdot Ue}}\cdot v+xF\cdot\frac{\partial\eta}{\partial x}+xG\cdot\frac{We}{Ue}\cdot\frac{\partial\eta}{\partial z}$$

$$K_1=\frac{x}{Ue}\cdot\frac{\partial Ue}{\partial x} \qquad K_2=\frac{x}{Ue}\cdot\frac{We}{Ue}\cdot\frac{\partial Ue}{\partial z}$$

$$K_3=\frac{x}{Ue}\cdot\frac{\partial We}{\partial z} \qquad K_4=\frac{x}{We}\cdot\frac{\partial We}{\partial x}$$

Accordingly, (6) is transformed to,

$$F=0,\ G=0,\ V=0 \qquad \text{when } \eta=0 \tag{10a}$$

$$F=1,\ G=1 \qquad \text{when } \eta=\eta_\infty \tag{10b}$$

The solution of the system given by (7)–(9) requires initial conditions at $\xi=0$ and $\zeta=0$. Substituting $\xi=0$ into (7)–(9) gives,

$$\frac{\partial V}{\partial\eta}+\frac{F}{2}=0 \tag{11}$$

$$\frac{\partial F}{\partial T}+V\cdot\frac{\partial F}{\partial\eta}=\frac{\partial^2 F}{\partial\eta^2} \tag{12}$$

$$\frac{\partial G}{\partial T}+V\cdot\frac{\partial G}{\partial\eta}=\frac{\partial^2 G}{\partial\eta^2} \tag{13}$$

The boundary conditions for (11)–(13) are the same as given by (10). Equations (11)–(13) are the Blasius boundary layer equations. Their solution gives the initial condition at $\xi=0$. In some problems, the initial condition at $\zeta=0$ can be given by the symmetry condition. On the symmetrical plane, both the cross-flow velocity component in the boundary layer, w, and the inviscid cross-flow velocity component, We, are zero. This leads the z-momentum equation to be singular on the symmetrical plane. However, differentiation with respect to z will yield a non-singular equation. As a consequence, (7)–(9) can be simplified, at $\zeta=0$, to,

$$\xi\cdot\frac{\partial F}{\partial \xi}+\frac{\partial V}{\partial \eta}+\frac{F}{2}\cdot(1+K_1)+\bar{G}\cdot K_3=0 \tag{14}$$

$$\frac{\partial F}{\partial T}+\xi F\cdot\frac{\partial F}{\partial \xi}+V\cdot\frac{\partial F}{\partial \eta}+K_1\cdot(F^2-1)=\frac{\partial^2 F}{\partial \eta^2} \tag{15}$$

$$\frac{\partial \bar{G}}{\partial T}+\xi F\cdot\frac{\partial \bar{G}}{\partial \xi}+V\cdot\frac{\partial \bar{G}}{\partial \eta}+K_3\cdot(\bar{G}^2-1)+K_5\cdot(F\bar{G}-1)=\frac{\partial^2 \bar{G}}{\partial \eta^2} \tag{16}$$

where

$$\bar{G}=\frac{\partial w/\partial z}{\partial We/\partial z} \qquad K_5=\frac{x}{\partial We/\partial z}\cdot\frac{\partial}{\partial x}\left(\frac{\partial We}{\partial z}\right)$$

Equations (14)–(16) are subjected to the following boundary conditions,

$$F=0,\ \bar{G}=0,\ V=0 \quad \text{when } \eta=0 \tag{17a}$$

$$F=1,\ \bar{G}=1 \quad \text{when } \eta=\eta_\infty \tag{17b}$$

Solution of (14)–(16) provides the initial condition at $\zeta=0$.

NUMERICAL TECHNIQUES

Generalized differential quadrature (GDQ)

The global method of GDQ was developed by Shu and Richards[1,2] based on the work of Bellmann *et al.*[3]. It approximates any spatial derivative at a discrete point by a linear weighted sum of all the functional values in the whole domain. In GDQ, the determination of weighting coefficients for the derivative discretization is based on the analysis of a high order polynomial approximation and the analysis of a linear vector space. The weighting coefficients of the first order derivative is given by a simple algebraic formulation while the weighting coefficients of the second and higher order derivatives are given by a recurrence relationship. For details, see References 1 and 2. Some basic results for the three-dimensional case are described as follows. For a smooth function $f(x, y, z)$, GDQ discretizes its nth order derivative with respect to x, the mth order derivative with respect to y, and the lth order derivative with respect to z, at the grid point (x_i, y_j, z_k) as,

$$f_x^{(n)}(x_i, y_j, z_k)=\sum_{q=1}^{N} w_{iq}^{(n)}\cdot f(x_q, y_j, z_k) \qquad n=1, 2, \ldots, N-1 \tag{18a}$$

$$f_y^{(m)}(x_i, y_j, z_k)=\sum_{q=1}^{M} \bar{w}_{jq}^{(m)}\cdot f(x_i, y_q, z_k) \qquad m=1, 2, \ldots, M-1 \tag{18b}$$

$$f_z^{(l)}(x_i, y_j, z_k)=\sum_{q=1}^{L} \bar{w}_{kq}^{(l)}\cdot f(x_i, y_j, z_q) \qquad l=1, 2, \ldots, L-1 \tag{18c}$$

for $i=1, 2, \ldots, N$; $j=1, 2, \ldots, M$, $k=1, 2, \ldots, L$,

where N, M, L are the number of grid points in the x, y and z direction, respectively, $w_{iq}^{(n)}$, $\bar{w}_{jq}^{(m)}$, $\tilde{w}_{kq}^{(l)}$ are the weighting coefficients to be determined as follows,

weighting coefficients for the first order derivative

$$w_{ij}^{(1)}=\frac{A^{(1)}(x_i)}{(x_i-x_j)\cdot A^{(1)}(x_j)} \qquad i,\ j=1,\ 2,\ \ldots,\ N,\ \text{but}\ j\neq i \tag{19a}$$

$$\bar{w}_{ij}^{(1)}=\frac{B^{(1)}(y_i)}{(y_i-y_j)\cdot B^{(1)}(y_j)} \qquad i,\ j=1,\ 2,\ \ldots,\ M,\ \text{but}\ j\neq i \tag{19b}$$

$$\tilde{w}_{ij}^{(1)}=\frac{C^{(1)}(z_i)}{(z_i-z_j)\cdot C^{(1)}(z_j)} \qquad i,\ j=1,\ 2,\ \ldots,\ L,\ \text{but}\ j\neq i \tag{19c}$$

where

$$A^{(1)}(x_i)=\prod_{j=1,j\neq i}^{N}(x_i-x_j)$$

$$B^{(1)}(y_i)=\prod_{j=1,j\neq i}^{M}(y_i-y_j)$$

$$C^{(1)}(z_i)=\prod_{j=1,j\neq i}^{L}(z_i-z_j)$$

weighting coefficients for the second and higher order derivatives

$$w_{ij}^{(n)}=n\cdot\left(w_{ii}^{(n-1)}\cdot w_{ij}^{(1)}-\frac{w_{ij}^{(n-1)}}{x_i-x_j}\right) \tag{20a}$$

for $i,\ j=1,\ 2,\ \ldots,\ N$, but $j\neq i$, $n=2,\ 3,\ \ldots,\ N-1$,

$$\bar{w}_{ij}^{(m)}=m\cdot\left(\bar{w}_{ii}^{(m-1)}\cdot\bar{w}_{ij}^{(1)}-\frac{\bar{w}_{ij}^{(m-1)}}{y_i-y_j}\right) \tag{20b}$$

for $i,\ j=1,\ 2,\ \ldots,\ M$, but $j\neq i$, $m=2,\ 3,\ \ldots,\ M-1$,

$$\tilde{w}_{ij}^{(l)}=l\cdot\left(\tilde{w}_{ii}^{(l-1)}\cdot\tilde{w}_{ij}^{(1)}-\frac{\tilde{w}_{ij}^{(l-1)}}{z_i-z_j}\right) \tag{20c}$$

for $i,\ j=1,\ 2,\ \ldots,\ L$, but $j\neq i$, $l=2,\ 3,\ \ldots,\ L-1$.

When $j=i$, the weighting coefficients are given by,

$$w_{ii}^{(n)}=-\sum_{j=1,j\neq i}^{N}w_{ij}^{(n)}, \qquad i=1,\ 2,\ \ldots,\ N,\ n=1,\ 2,\ \ldots,\ N-1 \tag{21a}$$

$$\bar{w}_{ii}^{(m)}=-\sum_{j=1,j\neq i}^{M}\bar{w}_{ij}^{(m)}, \qquad i=1,\ 2,\ \ldots,\ M,\ m=1,\ 2,\ \ldots,\ M-1 \tag{21b}$$

$$\tilde{w}_{ii}^{(l)}=-\sum_{j=1,j\neq i}^{L}\tilde{w}_{ij}^{(l)}, \qquad i=1,\ 2,\ \ldots,\ L,\ l=1,\ 2,\ \ldots,\ L-1 \tag{21c}$$

It is obvious from above equations that the weighting coefficients of the second and higher order derivatives can be completely determined from those of the first order derivatives. As shown in Reference 1, when some specific grid point distributions are used, (19) can be simplified. Actually, when the coordinates of grid points are chosen as the roots of the Nth order Chebyshev polynomial, GDQ provides exactly the same results as given from the Chebyshev collocation

method[11]. The GDQ method can be considered as a version of the classical spectral method since both methods are based on the high order polynomial approximation. That is, the solution of a partial differential equation is approximated by a $(N-1)$th order polynomial (N is the number of grid points). The roots of the approximated polynomial are the coordinates of grid points. In the spectral method, the solution of a partial differential equation is usually approximated by an orthogonal polynomial such as the Chebyshev polynomial. Thus the coordinates of grid points (roots of the orthogonal polynomial) cannot be chosen arbitrarily. In the GDQ method, the coordinates of grid points are given in advance, then the high order polynomial is constructed by using the Lagrange interpolated polynomial which approximates the solution of a partial differential equation. When the coordinates of grid points are chosen to be the same, e.g. the roots of Chebyshev polynomial, the GDQ method and the classical spectral method provide exactly the same results. Thus we believe that the GDQ method is more flexible in application than the classical spectral method. For details of the classical spectral method, see References 12 and 13. The numerical experiment demonstrated that the grid with stretching near the boundary can give better accuracy of numerical results and faster convergence rate.

When the functional values at all grid points are obtained, it is easy to calculate the functional values in the whole computational domain with high order of accuracy in terms of the polynomial approximation, i.e.,

$$f(x, y_j, z_k)=\sum_{i=1}^{N} f(x_i, x_j, z_k)\cdot r_i(x) \tag{22a}$$

$$f(x_i, y, z_k)=\sum_{j=1}^{M} f(x_i, y_j, z_k)\cdot s_j(y) \tag{22b}$$

$$f(x_i, y_j, z)=\sum_{k=1}^{M} f(x_i, y_j, z_k)\cdot t_k(z) \tag{22c}$$

$$f(x, y, z)=\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{k=1}^{L} f(x_i, y_j, z_k)\cdot r_i(x)\cdot s_j(y)\cdot t_k(z) \tag{22d}$$

where $r_i(x)$, $s_j(y)$, $t_k(z)$ are the Lagrange interpolation polynomials in the x, y and z directions, respectively. Equation (22) can be used to calculate the flow parameters at some specific points without losing accuracy.

Generalized integral quadrature (GIQ)

It is well-known that an integral of a function $f(x)$ over a closed domain $[a, b]$ can be approximated by

$$\int_a^b f(x)\cdot \mathrm{d}x=\sum_{i=1}^{N} c_i\cdot f(x_i) \tag{23}$$

where c_i is the weighting coefficient, x_i is the coordinate of grid points in the domain $[a, b]$. There are a number of conventional rules such as low order Simpson's rule and Gauss-Legendre method to determine c_i. Usually, the accuracy of numerical integration by these conventional rules depends on the number of grid points used in the integral domain. Thus if the integral domain contains only a few grid points, the resultant numerical integration may be less accurate. On the other hand, we may encounter some cases in practice where the function is smooth in a whole domain containing sufficient grid points when, however, only the approximation of numerical integration over a part of the whole domain is required. For this case, when the conventional rules are applied, the results of numerical integration may be less accurate since the integral domain does not contain sufficient grid points. As will be shown subsequently, the approach of generalized integral quadrature (GIQ) provides a promising way to approximate the integral of the function over a part of the whole domain with high order accuracy.

The GIQ is also developed based on the analysis of a high order polynomial approximation and the analysis of a linear vector space. 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 approximated by integrating the approximated 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. As a general case, it is supposed that the integral of a smooth function $f(x)$ over a part of the whole domain $[a, b]$ is approximated by a linear combination of all the functional values in the whole domain with the form

$$\int_{x_i}^{x_j} f(x)\cdot dx = \sum_{k=1}^{N} c_k^{ij}\cdot f(x_k) \tag{24}$$

Here N is the number of grid points in the overall domain $[a, b]$. When $x_i = a$, $x_j = b$, (24) reduces to a conventional numerical integral, that is, the integral domain is the whole domain containing all the functional values. Obviously, the key procedure to this approach is to determine the weighting coefficients. We will show that the weighting coefficients of GIQ can be easily obtained from those of the first order derivative in GDQ.

In a similar fashion to the analysis in GDQ, the smooth function $f(x)$ is approximated by a $(N-1)$th order polynomial, which constitues an N-dimensional linear vector 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,

$$c_k^{ij} = \int_{x_i}^{x_j} r_k(x)dx \tag{25}$$

The expression of c_k^{ij} is very complicated. We will turn to another way to determine c_k^{ij}. Setting,

$$f(x) = \frac{du(x)}{dx} \tag{26}$$

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

$$f(x) = a_0 + a_1\cdot x + \cdots + a_{N-1}\cdot x^{N-1} \tag{27}$$

where $a_0, a_1, \ldots, a_{N-1}$ are constants. Integrating (26) from a constant c to the variable x and using (27), we obtain,

$$u(x) = \int_c^x f(t)\cdot dt + u(c) = F(x, c) + u(c) \tag{28}$$

where,

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

It is easy to show that $F(x, c)$ constitutes an N dimensional linear vector space. One set of its base polynomials can be chosen as,

$$p_k(x) = (x-c)\cdot r_k(x), \qquad k = 1, 2, \ldots, N \tag{29}$$

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

$$F_x(x_i, c) = \sum_{j=1}^{N} a_{ij}\cdot F(x_j, c) \tag{30}$$

for $i = 1, 2, \ldots, N$,

where $F_x(x_i, c)$ is the first order derivative of function $F(x, c)$ with respect to x at x_i, and a_{ij} is the weighting coefficient, which can be determined using the same fashion as used in GDQ. Substituting (29) into (30) gives the weighting coefficients a_{ij} as,

$$a_{ij}=\frac{x_i-c}{x_j-c}\cdot w_{ij}^{(1)} \qquad j\neq i \tag{31a}$$

$$a_{ii}=w_{ii}^{(1)}+\frac{1}{x_i-c} \qquad j=i \tag{31b}$$

where $w_{ij}^{(1)}$ is the weighting coefficient of the first order derivative in GDQ. From (31), it is clear that c cannot be chosen to be the coordinates of grid points, x_i. This condition can also be seen from (28). It is seen from (28) that, when $c=x_i$, there is no integral term involved. Thus no numerical method is needed to approximate the integral for $c=x_i$. To derive formulations of numerical integration by GIQ, we write (30) as a vector form,

$$\mathbf{F}_x=\mathbf{A}\mathbf{F} \tag{32}$$

where,

$$\mathbf{F}=[F(x_1, c), F(x_2, c), \ldots, F(x_N, c)]^T$$

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

From (28), (26), F and $\mathbf{F}_x$ can also be written as,

$$\mathbf{F}=\mathbf{f}^I=\left[\int_c^{x_1} f(x)\,dx, \int_c^{x_2} f(x)\,dx, \ldots, \int_c^{x_N} f(x)\,dx\right]^T$$

$$\mathbf{F}_x=\mathbf{f}=[f(x_1), f(x_2), \ldots, f(x_N)]^T$$

So equation (32) can be rewritten as,

$$\mathbf{f}=\mathbf{A}\cdot\mathbf{f}^I \tag{33}$$

Setting $W^I=\mathbf{A}^{-1}$, equation (33) gives,

$$\int_c^{x_i} f(x)\cdot dx=\sum_{k=1}^{N} w_{ik}^I\cdot f(x_k) \tag{34a}$$

$$\int_{x_i}^{x_j} f(x)\cdot dx=\int_{x_i}^{c} f(x)\cdot dx+\int_c^{x_j} f(x)\cdot dx=\sum_{k=1}^{N} w_{jk}^I\cdot f(x_k)-\sum_{k=1}^{N} w_{ik}^I\cdot f(x_k) \tag{34b}$$

Application of GDQ-GIQ approach to the boundary layer equations

For the present numerical computation, we will apply the GDQ scheme to discretize all the spatial derivatives in the ξ, η, and ζ directions. The integral in the η direction derived from the integration of the continuity equation is approximated by GIQ scheme. The coordinates of grid points are chosen as,

$$\xi_i=-\frac{1}{2}\cdot\left[\cos\left(\frac{i-1}{N-1}\cdot\pi\right)-1\right]\cdot XL \qquad i=1, 2, \ldots, N \tag{35a}$$

$$\zeta_k=-\frac{1}{2}\cdot\left[\cos\left(\frac{k-1}{L-1}\cdot\pi\right]\cdot ZL \qquad k=1, 2, \ldots, L \tag{35b}$$

$$\eta_j=[\alpha\cdot\bar{\eta}_j^2+(1-\alpha)\cdot\bar{\eta}_j]\cdot\eta_\infty \qquad j=1, 2, \ldots, M \tag{35c}$$

where XL, ZL, η_∞ are the length scales of the computational domain in the ξ, ζ and η directions respectively, and,

$$\bar{\eta}_j = -\frac{\cos\left[\frac{2j-1}{2M}\cdot\pi\right]-\cos\left[\frac{1}{2M}\cdot\pi\right]}{2\cdot\cos\left[\frac{1}{2M}\cdot\pi\right]}$$

It is noted that with the basic grid point distribution $\bar{\eta}_j$, η_j is stretched towards the solid wall when $\alpha > 0$.

In the following, the discretization of (11)–(13) is chosen to demonstrate the numerical discretization of boundary layer equations by GDQ-GIQ approach. It is obvious from (11)–(13) that the solution F is exactly the same as the solution G. Thus, we only consider the solution of (11)–(12) which are the Blasius boundary layer equations. Application of GDQ to (12) gives,

$$\frac{\mathrm{d}F_{i,jk}}{\mathrm{d}t} = \sum_{q=1}^{M} \bar{w}_{jq}^{(2)} \cdot F_{i,q,k} - V_{i,j,k} \cdot \sum_{q=1}^{M} \bar{w}_{jq}^{(1)} \cdot F_{i,q,k} \tag{36}$$

The solution of V is obtained by integrating (11) along the η direction, which gives,

$$V_{i,j,k} = -\frac{1}{2} \cdot \int_0^{\eta} F(\xi_i, \eta, \zeta_k) \cdot \mathrm{d}\eta + V_{i,1,k} \tag{37}$$

In practice, it is not usual to obtain V from (37) using a classical integral quadrature. The reason is that some integral domains do not contain sufficient grid points, which greatly reduces the accuracy of numerical solutions. However, as we introduced in the above section, GIQ is a global method, which approximates the integral of a function over a part of the whole domain by a linear combination of all the functional values in the whole domain with high order of accuracy. Thus application of the GIQ scheme to (37) can provide the normal velocity V at any mesh point in the η direction with the same order of accuracy. This gives,

$$V_{i,j,k} = -\frac{1}{2} \cdot \sum_{q=1}^{M} (w_{1q}^{I} - w_{1q}^{I}) \cdot F_{i,q,k} + V_{i,1,k} \tag{38}$$

The boundary conditions for (36), (38) are given by (10), which are easily implemented in the solution procedure. Similarly, the spatial derivatives in the boundary layer equations (15)–(16) on the symmetrical plane, and the general boundary layer equations (8)–(9) can be discretized by the GDQ scheme. After spatial discretization, the resultant ordinary differential equations for F and G are then solved by the 4-stage Runge-Kutta scheme (4th order accurate). The normal velocity V is determined by integrating the continuity (7) or (14) along the η direction. The resultant integral is then approximated by the GIQ scheme.

RESULTS AND DISCUSSION

The three-dimensional laminar flow past a flat plate with attached circular cylinder is chosen as a test example to validate the application of the GDQ-GIQ approach. The geometry of the problem is displayed in *Figure 1*. This flow problem has been extensively studied by many researchers[4,5]. The inviscid velocity components are given by,

$$Ue = U_\infty \left[1 + a^2 \cdot \frac{\Delta_2}{\Delta_1^2}\right] \tag{39a}$$

$$We = -2U_\infty \cdot a^2 \cdot \frac{\Delta_3}{\Delta_1^2} \tag{39b}$$

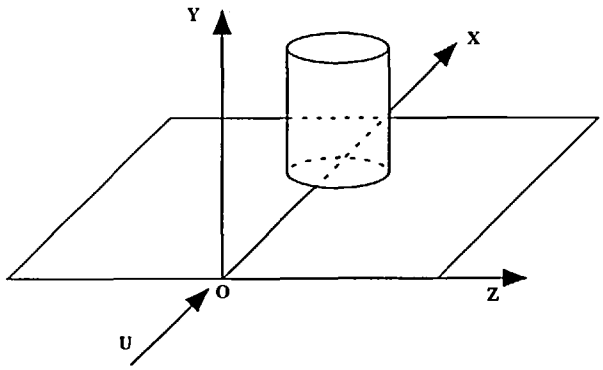


Figure 1 Geometry of the flow past a flat plate with attached cylinder

where

$$\Delta_1 = (x - x_0)^2 + z^2$$
$$\Delta_2 = -(x - x_0)^2 + z^2$$
$$\Delta_3 = (x - x_0) \cdot z$$

U_∞ is a reference velocity, a is the cylinder radius, and x_0 is the distance of the cylinder axis from the leading edge. For the present numerical simulation, these parameters are chosen as,

$$U_\infty = 30.50 \text{ m/sec}, \; a = 0.061 \text{ m}, \; x_0 = 0.457 \text{ m}, \; \eta_\infty = 8.0$$

and the mesh size used is 11 grid points in the ξ and ζ directions, and 12 grid points in the η direction. The computational domain in the z direction is chosen as between $z = 0$ and $z = 0.20$ m. Since the cylinder causes an adverse pressure gradient in front of itself, it is expected that the streamwise velocity will reverse in direction along a line in front of the cylinder. Since the adverse pressure gradient is a maximum along the plane of symmetry, the flow reversal in the boundary layer will first occur in the plane of symmetry. Thus the computational domain in the x direction should be truncated before the separation point in the plane of symmetry because of the Goldstein singularity.

The choice of constant c and initial solution

From numerical experiments, it was found that numerical results are independent of the choice of the constant c used in (31) in the GIQ method when c is chosen as a small value, for example, $|c| \leqslant 0.1$. When c is chosen as a large value, for example, $|c| > 10$, the choice of c has some effect on the accuracy of numerical results. For the Blasius boundary layer, when c is chosen as 20 the computed wall shear stress is 0.33345 while the exact value is 0.33210. When c is chosen to satisfy $|c| \leqslant 0.1$, all the computed wall shear stresses are 0.33215. Thus in the present study, we choose the constant c as 0.01.

Since the three-dimensional boundary layer equation is parabolic in the x and z directions, the initial solutions at $x = 0$ and $z = 0$ are required before the boundary layer equation is solved in the whole computational field. The initial solution at $x = 0$ can be obtained by solving (11)–(13) or (36), (38). Their solution represents the Blasius boundary layer solution. Compared with the exact results, our GDQ-GIQ results are obtained very accurately by using only 12 grid points in the η direction. For example, the computed wall shear stress $(\partial F/\partial \eta)_w$ by GDQ-GIQ is 0.33215 while the exact value is 0.33210. The computed velocity profile of the Blasius boundary layer at $x = 0$ is almost identical to the exact solution.

After the solution at $x=0$ is obtained, the initial solution at $z=0$ (symmetrical plane) can be obtained by solving (14)–(16). For the solution of (14)–(16), the initial solution at $x=0$ (Blasius boundary layer solution) is needed. For this case, a variety of numerical experiments have been conducted for the use of length scale in the x direction, XL. It was found that the GDQ-GIQ approach is very sensitive to the Goldstein singularity. When the computational domain in the x direction is taken as $0 \leqslant x \leqslant 0.25970$, the steady state resolution can be obtained very quickly (51.65 seconds CPU time on the IBM 3081) and accurately. But when the computational domain is taken as $0 \leqslant x \leqslant 0.25975$, the computation will diverge after a few time steps. Thus it appears that the separation point in the plane of symmetry is between 0.25970 and 0.25975. This result is a little different from that given in Reference 4. The numerical results in Reference 4 showed that the separation point first starts at $x=0.25$ in the plane of symmetry.

Streamwise and crosswise velocity profiles

Figures 2 and *3* show the non-dimensional streamwise and crosswise velocity profiles along the plane of symmetry. Due to the adverse pressure gradient, the steamwise velocity near the

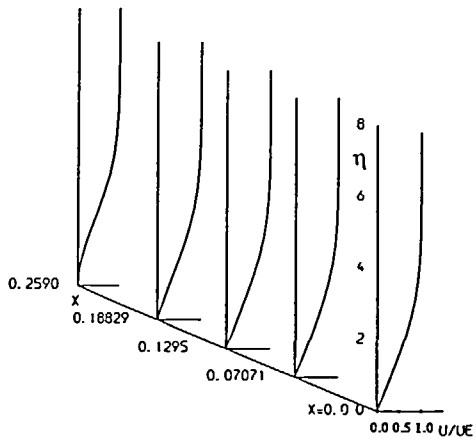


Figure 2 Streamwise velocity profile on the symmetrical plane ($z=0$)

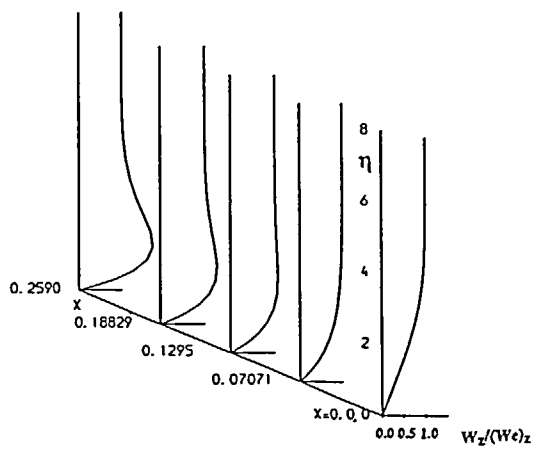


Figure 3 Crosswise velocity profile on the symmetrical plane ($z=0$)

wall decreases along the x direction, and its first order derivative vanishes at the separation point. In the meantime, the crosswise velocity increases rapidly and has a value larger than that of the inviscid crosswise velocity. In the present study, the computational domain cannot include any separation region because of numerical instability. This also differs from the results in Reference 4, where the numerical computation can go through a flow reversal. It is our opinion that, like the parabolized Navier-Stokes equation, the three-dimensional boundary layer equation is unable to simulate the flow with a flow reversal in the streamwise direction because of the elliptic nature in separated region and separation point itself.

The wall shear stress distribution

After the initial solutions at $x=0$ and $z=0$ are obtained, the solution in the full flow field can be obtained by solving (7)–(9). In the present work, the GDQ scheme is applied in all the spatial directions, and the resultant ordinary differential equations for F, G are solved by the 4-stage Runge-Kutta scheme.

The wall shear stress can be used to plot the wall shear lines. The wall shear lines are defined as,

$$\frac{dx}{(\partial F/\partial \eta)_w}=\frac{dz}{(\partial G/\partial \eta)_w} \tag{40}$$

which further gives,

$$x=x_0+(\partial F/\partial \eta)_w\cdot \Delta t \tag{41}$$

$$z=z_0+(\partial G/\partial \eta)_w\cdot \Delta t \tag{42}$$

By choosing $\Delta t=0.01$, $x_0=0$, $z_0=i\cdot(0.20/21)$, $i=1, 2, \ldots, 20$, we can plot 20 wall shear lines as shown in *Figure 4*. Accordingly, the streamlines of the inviscid flow which are defined as,

$$\frac{dx}{Ue}=\frac{dz}{We} \tag{43}$$

can be plotted from,

$$x=x_0+Ue\cdot \Delta t \tag{44}$$

$$z=z_0+We\cdot \Delta t \tag{45}$$

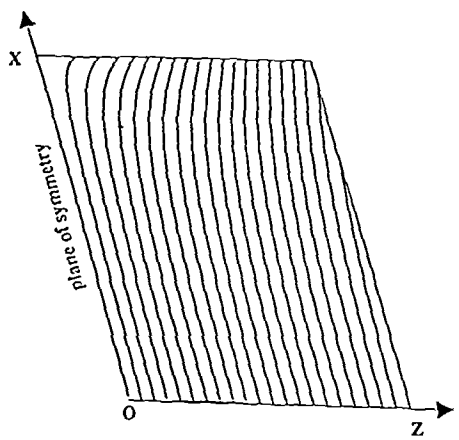


Figure 4 Wall shear lines of the boundary layer

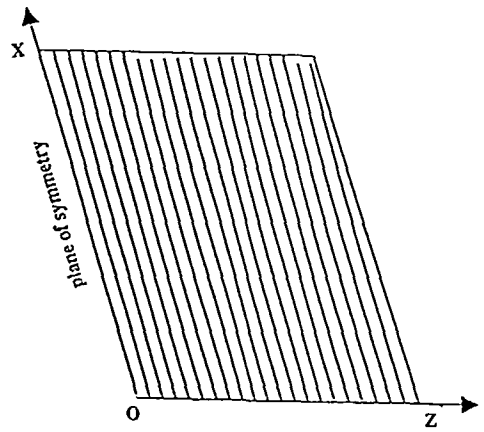


Figure 5 Streamlines of the inviscid flow

Table 1 Comparison of present results with Cebeci's results

	$z=0$			
x	Cebeci $(\partial F/\partial\eta)_w$	Present $(\partial F/\partial\eta)_w$	Cebeci $(\partial G/\partial\eta)_w$	Present $(\partial G/\partial\eta)_w$
0	0.332066	0.332150	0.332066	0.332150
0.0488	0.324951	0.325421	0.702488	0.702926
0.0976	0.312821	0.312826	1.124300	1.124816
0.1464	0.290184	0.290724	1.524250	1.623479
0.1952	0.243524	0.243864	2.250740	2.247442
0.2440	0.125972	0.123785	3.126830	3.119367
	$z=0.0488$			
x	Cebeci $(\partial F/\partial\eta)_w$	Present $(\partial F/\partial\eta)_w$	Cebeci $(\partial G/\partial\eta)_w$	Present $(\partial G/\partial\eta)_w$
0	0.322066	0.331150	0.332066	0.331150
0.0488	0.325184	0.325653	0.717599	0.696219
0.0976	0.314423	0.314859	1.124210	1.107046
0.1464	0.295233	0.295767	1.605920	1.586223
0.1952	0.259005	0.259305	2.191930	2.171279
0.2440	0.181979	0.177312	2.959350	2.941173

By choosing the same values of Δt, x_0, z_0 for the wall shear lines, we can then plot 20 streamlines of the inviscid flow which are displayed in *Figure 5*. Compared with *Figure 4*, it is obvious that the boundary layer flow has increased the movement in the crosswise direction in the front of the cylinder.

Table 1 lists the computed values of $(\partial F/\partial\eta)_w$ and $(\partial G/\partial\eta)_w$ at some specific points. These values are obtained by using (22) to interpolate from the values at grid points without losing accuracy. Also included in *Table 1* are the results of Cebeci[5] which are obtained by the Keller-box finite difference scheme with Richardson extrapolation processing. The present numerical results agree well with Cebeci's results when x and z are close to 0. However, there are some differences between present results and Cebeci's results when x and z are far away from $x=0$ and $z=0$.

The numerical methods used in the present study and in Reference 5 are quite different. In the present work, GDQ scheme is applied in all the spatial directions, and the boundary layer solutions are obtained in the full computational field at the same time. Thus it can be expected that the boundary layer solutions at different stations have the same order of accuracy. Furthermore, the use of (22) does not affect the accuracy of numerical results since GDQ and GIQ are also based on the high order polynomial approximation. In Reference 5, the low order Keller-box finite difference scheme is used, and the solution is obtained by a space-marching technique. As we know, the low order method has higher numerical damping than the global method, which may affect the accuracy of numerical solutions at following stations. So, we may conclude that the present results are more accurate than the Cebeci's results.

Applicability of GDQ-GIQ approach to a marching strategy

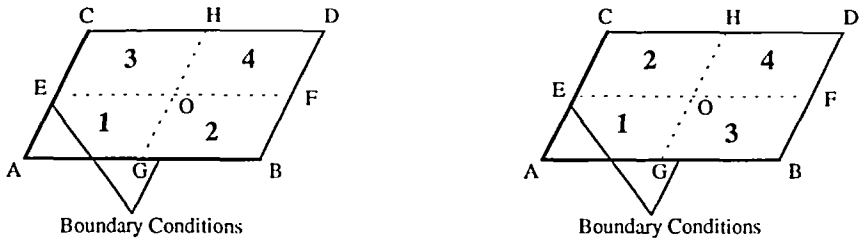
Since the boundary layer equations are nonlinear equations, their solutions should be given by iterations. As shown above, a time-marching method was used in the present study. Compared with the classical low order space-marching method which requires iterations line by line, the time marching method requires iterations for the whole computational domain. However, since GDQ-GIQ uses much less grid points in the ξ and ζ directions than low order methods, the computational effort required is greatly reduced. The numerical advantages of GDQ-GIQ approach is that accurate numerical solutions can be obtained using a considerably small number of grid points. Its disadvantage is that it requires a great deal of virtual storage because it needs to store the solution for the whole computational domain.

The GDQ-GIQ approach can be applied together with low order finite difference schemes to enable a space-marching strategy. For example, at point $(\xi_{i-1/2}, \eta_j, \zeta_{k-1/2})$, when a second order finite difference scheme is used to discretize the derivatives in the ξ and ζ directions, GDQ scheme is used to discretize the derivatives in the η direction, and an implitic Euler difference scheme is used to discretize the time derivative, (8) or (9) can be reduced to a set of algebraic equations by linearizing the non-linear terms,

$$\mathbf{A} \cdot \mathbf{U} = \mathbf{b} \tag{46}$$

where $\mathbf{U} = (u_{i,2,k}, u_{i,3,k}, \ldots, u_{i,M-1,k})^T$, u can be F or G, $\mathbf{A}$ is a full matrix and $\mathbf{b}$ is a known vector. With (40) and the initial solutions, the boundary layer solution can be obtained by marching the solution station by station. The advantage of this strategy is that the virtual storage is greatly reduced. The disadvantage is that a large number of grid points in the ξ and ζ directions may be required for accurate numerical solutions.

It seems that the above disadvantage can be improved by using the multi-domain GDQ-GIQ approach. The basic idea of this approach is to decompose the whole computational domain into several blocks. Then in each block, the GDQ method is applied in all the spatial directions. Since the boundary layer equation is parabolic in the x and y direction, its solution can be obtained by marching solution block by block without iteration. As shown in *Figure 6*, there are two ways to march the solution for 3D case. Before the 3D boundary layer equation is solved, the initial solutions along the line AB and AC should be given in advance by some ways. Then in block 1, we apply GDQ in all the spatial directions. Using the same fashion as used in a single domain, the boundary layer solution in Block 1 (AGOE) including the line EO and GO can be obtained quickly and accurately. After that, there are two ways to march the solution. The first way is shown in *Figure 6*(a). With the solution in the line GO and the initial solution in the line GB, the boundary layer solution can be obtained by marching first to Block 2 (GBFO), then to Block 3 (EOHC), and finally to Block 4 (OFDH). The second way is shown in *Figure 6*(b). The boundary layer solution can be marched first to Block 2 (EOHC), then to Block 3 (GBFO), and finally to Block 4 (OFDH). It seems that the multi-domain GDQ-GIQ approach combines the marching capabilities of low order methods with the potential for accuracy of the GDQ scheme.



(a) Case One (a) Case Two

Figure 6 Marching strategy of the multi-domain GDQ-GIQ approach

CONCLUSIONS

The global methods of generalized differential quadrature (GDQ) and generalized integral quadrature (GIQ) are applied to solve three-dimensional, laminar boundary layer equations. GDQ scheme is applied to discretize the derivatives in all the spatial directions, and GIQ scheme is used to approximate the integral derived from the integration of continuity equation. After special discretization, the resultant set of ordinary differential equations are solved by using the 4-stage Runge-Kutta scheme. Application of GDQ-GIQ approach to simulate a test problem demonstrated that accurate numerical results can be achieved using just a few grid points. Although application of GDQ scheme in all the spatial directions is demonstrated in the paper, it can be combined with low order finite difference schemes to enable a space-marching strategy. Furthermore, a multi-domain GDQ-GIQ approach is expected to be developed, which combines the marching capabilities of low order methods with the potential for accuracy of GDQ scheme.

REFERENCES

1 Sju, C. and Richards, B. E. Application of generalized differential quadrature to solve two-dimensional incompressible Navier-Stokes equations, *Int. J. Num. Meth. Fluids*, **15**, 791–798 (1992)
2 Sju, C. and Richards, B. E. Parallel simulation of incompressible viscous flows by generalized differential quadrature, *Computing Systems in Eng.*, **3**, 271–281 (1992)
3 Bellman, R., Kashef, B. G. and Casti, J. Differential quadrature: a technique for the rapid solution of nonlinear partial differential equations, *J. Comput. Phys.*, **10** (1972)
4 Dwyer, H. A. Solution of a three-dimensional boundary-layer flow with separation, *AIAA J.*, **6**, 1336–1342 (1968)
5 Cebeci, T. Calculation of three-dimensional boundary layers II. Three-dimensional flows in Cartesian coordinates, *AIAA J.*, **13**, 1056–1064 (1975)
6 Keller, H. K. and Cebeci, T. Accurate numerical methods for boundary layer flows, I. Two-dimensional laminar flows, *Lecture Notes in Physics*, **8**, 92–100 (1970)
7 Keller, H. K. Accurate difference methods for two-point boundary value problems, *SIAM J. Numer. Anal.*, **11**, 305–320 (1974)
8 Werle, M. J. and Bertke, S. D. A finite difference method for boundary layers with reverse flow, *AIAA J.*, **10**, 1250–1252 (1972)
9 Cebeci, Y. The laminar boundary layer on a circular cylinder started impulsively from rest, *J. Comput. Phys.*, **31**, 153–172 (1979
10 Cebeci, T. Unsteady boundary layers with an intelligent numerical scheme, *J. Fluid Mech.*, **163** (1986)
11 Ehrenstein, U. and Peyret, R. A Chebyshev collocation method for the Navier-Stokes equations with application to double-diffusive convection, *Int. J. Num. Meth. Fluids*, **9**, 427–452 (1989)
12 Gottlieb, D. and Orszag, S. A. *Numerical Analysis of Spectral Methods: Theory and Applications*, (SIAM-DBMS, Philadelphia), 1977
13 Canuto, C., Hussaini, M. Y., Quarterioni, A. and Zang, T. A. *Spectral Methods in Fluid Dynamics*, Springer, New York, Berlin (1987