JOURNAL OF COMPUTATIONAL PHYSICS 10, 40–52 (1972)

Differential Quadrature: A Technique for the Rapid Solution of Nonlinear Partial Differential Equations*

RICHARD BELLMAN

Departments of Mathematics, Electrical Engineering and Medicine,

B. G. KASHEF

Department of Electrical Engineering, University of Southern California Los Angeles, California 90007

AND

J. CASTI

Systems Control, Inc. Palo Alto, California 94306

Received May 13, 1971

The numerical solution of nonlinear partial differential equations plays a prominent role in numerical weather forecasting, optimal control theory, radiative transfer, and many other areas of physics, engineering, and biology. In many cases all that is desired is a moderately accurate solution at a few points which can be calculated rapidly.

In this paper we wish to present a simple direct technique which can be applied in a large number of cases to circumvent the difficulties of programming complex algorithms for the computer, as well as excessive use of storage and computer time. We illustrate this technique with the solution of some partial differential equations arising in various simplified models of fluid flow and turbulence.

1. INTRODUCTION

The numerical solution of nonlinear partial differential equations plays a prominent role in numerical weather forecasting [1], optimal control theory [2], radiative transfer [3], and many other areas of physics, engineering, and biology. In many cases all that is desired is a moderately accurate solution at a few points which can be calculated rapidly. The standard finite difference methods currently

* Supported by the National Science Foundation under Grant No. GP 29049 and the Atomic Energy Commission under Grant No. AT(40-3), 113, Project No. 19.

in use have the characteristic that the solution must be calculated at a large number of points in order to obtain moderately accurate results at the points of interest. Consequently, both the computing time and storage required often prohibit the calculation. Furthermore, the mathematical techniques involved in finite difference schemes or in the Fourier transform methods, are often quite sophiscated and thus not easily learned or used.

In this paper we wish to present a technique which can be applied in a large number of cases to circumvent both the above difficulties. We illustrate this technique with the solution of some partial differential equations arising in various simplified models of fluid flow and turbulence.

It is not easy to provide any stability analysis because of the nonlinearity of the equations considered. In subsequent papers, however, we will examine the stability of the numerical scheme applied to various linear equations since some questions of independent interest arise. We shall also discuss a number of other types of boundary conditions and quadratures.

2. Differential Quadrature

Consider the nonlinear first-order partial differential equation

$$u_t = g(t, x, u, u_x), \quad -\infty < x < \infty, t > 0 \tag{1}$$

with initial condition

$$u(0, x) = h(x), \tag{2}$$

an equation arising in many mathematical models of physical processes. Let us make the assumption that the function u satisfying Eqs. (1) and (2) is sufficiently smooth to allow us to write the approximate relation

$$u_x(t, x_i) \cong \sum_{j=1}^{N} a_{ij} u(t, x_j), \qquad i = 1, 2,..., N. \tag{3}$$

There are many ways of determining the coefficients a_{ij}. One method for determining these coefficients will be discussed below. Substitution of Eq. (3) into Eq. (1) yields the set of N ordinary differential equations

$$u_t(t, x_i) \cong g\left(t, x_i, u(t, x_i), \sum_{j=1}^{N} a_{ij} u(t, x_j)\right) \tag{4}$$

with initial conditions

$$u(0, x_i) = h(x_i), \qquad i = 1, 2,..., N. \tag{5}$$

Hence, under the assumption that (3) is valid, we have succeeded in reducing the task of solving Eq. (1) to the task of solving a set of ordinary differential equations with prescribed initial values.

The numerical solution of such a system, Eq. (5), is a simple task for modern electronic computers using standard programs with minimal computing time and storage (in contrast to the case of partial differential equations). The additional storage needed for the solution of a partial differential equation using the differential quadrature technique is that of storing the $N \times N$ matrix a_{ij}. Considering that in practice it turns out that relatively low order differential quadrature is all that is needed, the total amount of storage and time required on the machine is thus quite low. We also note that the number of arithmetic operations to be performed for every point is the N additions and multiplications, Eq. (3), plus the amount needed for the solution of the set of ordinary differential equations (which is, of course, method dependent).

3. Determination of Weighting Coefficients

In order to determine appropriate coefficients in the approximation

$$u_x(x_i) \cong \sum_{j=1}^{N} a_{ij} u(x_j), \qquad i = 1, 2, \ldots, N, \tag{1}$$

we can proceed by analogy with the classical quadrature case, demanding that Eq. (1) be exact for all polynomials of degree less than or equal to $N - 1$. The test function $p_k(x) = x^{k-1}$, $k = 1, \ldots, N$, for arbitrary distinct x_i leads to the set of linear algebraic equations

$$\sum_{j=1}^{N} a_{ij} x_j^{k-1} = (k-1)\, x_i^{k-1}, \qquad k = 1, 2, \ldots, N. \tag{2}$$

which has a unique solution since the coefficient matrix is a Vandermonde matrix [4].

Rather than solving a set of linear algebraic equations, we may readily determine the a_{ij} explicity once and for all if x_i are properly selected. In this discussion the x_i are chosen to be the roots of the shifted Legendre polynomial of degree N, $P_N{}^*(x)$, with orthogonality range of [0, 1]. The polynomials $P_N{}^*(x)$ are defined in terms of the usual Legendre polynomials by the relation

$$P_N{}^*(x) = P_N(1 - 2x). \tag{3}$$

The nodes x_i of $P_N{}^*(x)$, $N = 7, 9$ are given in Table I, for other values see Ref. [5].

TABLE I

Nodes of $P_N{}^*(x)$

x	$N = 7$	$N = 9$
x_1	0.02544604	0.01591988
x_2	0.12923440	0.08198445
x_3	0.29707742	0.19331428
x_4	0.50	0.33787329
x_5	0.70292257	0.50
x_6	0.80076559	0.66212671
x_7	0.97455395	0.80668572
x_8	—	0.91801555
x_9	—	0.98408012

By analogy with Lagrange's interpolation formula the test function is taken to be of the form

$$p_k(x) = P_N{}^*(x)/[(x - x_k)\, P_N^{*\prime}(x_k)]. \tag{4}$$

It follows that $p_k(x)$ is a polynomial of degree $(N - 1)$ such that $p_k(x_j) = \delta_{kj}$. Using the fact that Eq. (1) is to be exact for $u(x) = p_k(x)$, we see that

$$a_{ik} = P_N^{\prime *}(x_i)/[(x_i - x_k)\, P_N^{\prime *}(x_k)], \qquad i \neq k. \tag{5}$$

For the case $k = i$, use of L'Hospital's rule plus the fact that $P_N{}^*(x)$ satisfies the differential equation

$$x(1 - x^2)\, P_N^{*\prime\prime}(x) + (1 + 2x)\, P_N^{*\prime}(x) + N(N + 1)\, P_N{}^*(x) = 0 \tag{6}$$

gives

$$a_{kk} = (1 - 2x_k)/[2x_k(x_k - 1)]. \tag{7}$$

Using Eqs. (3), (5) and (7), the constant coefficients a_{ij} for any value of N, say $N = 3, \ldots, 15$, can be easily calculated. These values are then used as input data for a given problem once N has been chosen.

4. Numerical Results for First Order Problems

A number of computational experiments were carried out to test the efficacy of the above approach. In all numerical examples given, the reduced set of ordinary differential equations was solved by an Adams-Moulton integration scheme with step size of 0.01. The limits of integration (unless specified) were from $t = 0$ to $t = 1$.

The first numerical experiment was carried out for the equation

$$u_t(x, t) = x^2 - 1/4 u_x^2(x, t), \tag{1}$$

$$u(x, 0) = 0, \tag{2}$$

which arises in the theory of dynamic programming [2]. Replacing the derivative term on the right side of Eq. (1) by an approximating sum, we obtain the nonlinear set of ordinary differential equations

$$u_t(x_i, t) = x_i^2 - \frac{1}{4}\left[\sum_{j=1}^{N} a_{ij} u(x_j, t)\right]^2 \tag{3}$$

with initial conditions

$$u(x_i, 0) = 0, \qquad i = 1, 2, \ldots, N. \tag{4}$$

The analytic solution of Eq. (1) is known to be

$$u(x, t) = x^2 \tanh(t), \tag{5}$$

which gives a means of checking our numerical results. For a quadrature of order $N = 7$, results are displayed under the column $e(1)$ of Table II. The quantity $e(k)$ is defined to be the relative error in computation, i.e., the ratio of the absolute error to the absolute value of the actual analytical answer. Index $k = 1, 2, 3$ and 4 refers to the different experiments considered in this section.

TABLE II

Differential Quadrature of Order $N = 7$

t	x	$e(1)$	$e(2)$	$e(3)$	$e(4)$
0.1	x_1	6.*E*-08	2.*E*-10	2.*E*-05	8.*E*-08
0.1	x_4	8.*E*-08	3.*E*-10	6.*E*-06	1.*E*-10
0.1	x_7	1.*E*-07	1.*E*-07	3.*E*-05	2.*E*-07
0.5	x_1	4.*E*-06	7.*E*-08	6.*E*-04	5.*E*-07
0.5	x_4	9.*E*-08	2.*E*-08	8.*E*-05	1.*E*-07
0.5	x_7	5.*E*-07	8.*E*-07	3.*E*-04	1.*E*-06
1.0	x_1	3.*E*-05	1.*E*-07	1.*E*-04	1.*E*-04
1.0	x_4	2.*E*-07	4.*E*-08	2.*E*-04	3.*E*-05
1.0	x_7	7.*E*-07	2.*E*-06	1.*E*-03	1.*E*-03

The second set of numerical experiments concerned a hyperbolic nonlinear problem of the form

$$u_t(x, t) = uu_x(x, t), \qquad \text{in} \quad 0 < x \leqslant 1, \qquad 0 \leqslant t \leqslant T, \tag{6}$$

$$u(x, 0) = g(x), \qquad \text{for} \quad 0 < x \leqslant 1. \tag{7}$$

This is a well-known test equation which possesses the implicit solution

$$u(x, t) = g(x + ut). \tag{8}$$

The shock phenomenon inherent in the solution of Eq. (6) can be pushed far into the future by a suitable selection of g, and thus will not be considered here. As a first example, we let $g(x) = 0.1x$. In this case the exact solution of Eq. (6) is

$$u(x, t) = x/(t - 10). \tag{9}$$

Replacing the x-derivative by a differential quadrature of order $N = 7$ and integrating the resulting set of equations with $T = 1$ we obtain the results of Table II, Column $e(2)$.

In both examples 1 and 2, excellent results were to be expected since the solution of the equation chosen was a polynomial in the discretized variable. The experiments were thus tests of the numerical stability of the algorithm.

We now examine equations where the solution is more complex. Consider Eq. (6) with the initial function

$$g(x) = (0.1) \sin \pi x. \tag{10}$$

The analytic solution is

$$u(x, t) = (0.1) \sin \pi(x + ut), \tag{11}$$

with a well-behaved solution for $0 \leqslant t \leqslant 1$. We compute the solution of Eq. (11) by the Newton–Ralphson method, using as our initial approximation the computed value obtained from the differential quadrature version of Eq. (6). The relative error for $N = 7$ is given by $e(3)$ of Table II.

The last experiment in this section involves solving Eq. (6) with the initial condition

$$g(x) = 0.2x^2. \tag{12}$$

In this case the explicit analytic solution is

$$u(x, t) = \frac{(1 - (0.4)\, tx) - \sqrt{1 - (0.8)\, tx}}{(0.4)\, t^2}. \tag{13}$$

Using the same order quadrature and integration scheme as before, we obtain $e(4)$ of Table II.

5. Systems of Nonlinear Partial Differential Equations

Consider next the nonlinear system of equations

$$u_t = uu_x + vu_y , \qquad u(x, y, 0) = f(x, y), \tag{1}$$

$$v_t = uv_x + vv_y , \qquad v(x, y, 0) = g(x, y). \tag{2}$$

We wish to use differential quadrature to obtain numerical values for the functions u and v. To check the accuracy of our results, we note that Eqs. (1) and (2) possess the implicit solution

$$u(x, y, t) = f(x + tu, y + tv), \tag{3}$$

$$v(x, y, t) = g(x + tu, y + tv), \tag{4}$$

a straightforward extension of the one-dimensional case above. Adopting the notation

$$u_{ij}(t) = u(x_i , y_j , t), \tag{5}$$

$$v_{ij}(t) = v(x_i , y_j , t), \tag{6}$$

and using the foregoing approximations for the partial derivatives, we may write Eqs. (4) and (5) as the system of ordinary differential equations

$$u'_{ij}(t) = u_{ij}(t) \sum_{k=1}^{N} a_{ik}u_{kj}(t) + v_{ij}(t) \sum_{k=1}^{N} a_{jk}u_{ik}(t), \tag{7}$$

$$v'_{ij}(t) = u_{ij}(t) \sum_{k=1}^{N} a_{ik}v_{kj}(t) + v_{ij}(t) \sum_{k=1}^{N} a_{jk}v_{ik}(t), \tag{8}$$

$$i, j = 1, 2,..., N.$$

The initial conditions are

$$u_{ij}(0) = f(x_i , y_j), \tag{9}$$

$$v_{ij}(0) = g(x_i , y_j), \qquad i, j = 1, 2,..., N. \tag{10}$$

Using the above reduction, numerical experiments were carried out for a number of different initial functions f and g. Since the solution of Eqs (1) and (2) can also possess a shock phenomenon for finite t, care was taken in selecting f and g to insure that the shock took place for a value of t far away from our region of interest.

The first experiment involved the case

$$f(x, y) = g(x, y) = x + y. \tag{11}$$

The explicit solution is given by

$$u(x, y) = v(x, y) = (x + y)/(1 - 2t). \tag{12}$$

Using Eqs. (7) and (8) with $N = 7$ and an integration step size of 0.01, the results corresponding to $e_u(1) = e_v(1)$ of Table III were obtained.

TABLE III

Differential Quadrature of Order $N = 7$

t	x	y	$e_u(1) = e_v(1)$	$e_u(2)$	$e_v(2)$
0.1	x_1	y_1	9.*E*-08	5.*E*-07	5.*E*-09
0.1	x_7	y_1	9.*E*-08	5.*E*-07	5.*E*-09
0.1	x_4	y_4	9.*E*-08	9.*E*-08	5.*E*-09
0.1	x_1	y_7	9.*E*-08	4.*E*-07	5.*E*-09
0.1	x_7	y_7	9.*E*-08	5.*E*-07	5.*E*-09
0.2	x_1	y_1	6.*E*-07	1.*E*-04	6.*E*-10
0.2	x_7	y_1	6.*E*-07	1.*E*-03	6.*E*-10
0.2	x_4	y_4	6.*E*-07	3.*E*-05	6.*E*-10
0.2	x_1	y_7	6.*E*-07	1.*E*-04	6.*E*-10
0.2	x_7	y_7	6.*E*-07	1.*E*-03	6.*E*-10
0.3	x_1	y_1	3.*E*-06	—	—
0.3	x_7	y_1	3.*E*-06	—	—
0.3	x_4	y_4	3.*E*-06	—	—
0.3	x_1	y_7	3.*E*-06	—	—
0.3	x	y_7	3.*E*-06	—	—

Next, the initial functions were changed to $f(x, y) = x^2, g(x, y) = y$. The solution for this case is

$$u(x, y, t) = [(1 - 2tx) - \sqrt{1 - 4tx}]/2t^2 \tag{13}$$

$$v(x, y, t) = y/(1 - t). \tag{14}$$

The shock occurs at $t = 1/4x$. Integrating Eqs. (7) and (8) from $t = 0$ to $t = 0.20$ with $N = 7$, we obtain $e_u(2)$ and $e_v(2)$ of Table III which are relative errors in functions u and v, respectively.

The final case was for the initial functions

$$f(x, y) = 0.1 \sin \pi x \sin \pi y/2, \tag{15}$$

$$g(x, y) = 0.1 \sin \pi x/2 \sin \pi y. \tag{16}$$

Since no explicit solution exists in this case, Eqs. (3) and (4) together with the Newton–Raphson method were used to produce the solution. Table IV summarizes the results for the differential quadrature of order $N = 7$. The values corresponding to $x_k = y_k$, $k = 1,\dots, N$ for $N = 7$ are listed in Table I.

TABLE IV

Differential Quadrature of Order 7

t	x	y	$e_u(3)$	$e_v(3)$
0.1	x_1	y_1	5.E-07	5.E-07
0.1	x_7	y_1	4.E-06	2.E-05
0.1	x_4	y_4	3.E-06	3.E-06
0.1	x_1	y_7	2.E-05	4.E-06
0.1	x_7	y_7	3.E-05	3.E-05
0.5	x_1	y_1	7.E-06	7.E-06
0.5	x_7	y_1	1.E-04	6.E-04
0.5	x_4	y_4	8.E-05	8.E-05
0.5	x_1	y_7	6.E-04	1.E-04
0.5	x_7	y_7	3.E-04	3.E-04
1.0	x_1	y_1	2.E-05	2.E-05
1.0	x_7	y_1	7.E-04	9.E-05
1.0	x_4	y_4	1.E-04	1.E-04
1.0	x_1	y_7	9.E-05	7.E-04
1.0	x_7	y_7	1.E-03	1.E-03

6. Higher Order Problems

We have seen that a good approximation to the first derivative of a function may often be obtained in the form

$$u_x(x_i) \cong \sum_{j=1}^{N} a_{ij}u(x_j), \qquad i = 1, 2,\dots, N. \tag{1}$$

Let us now indicate how this idea may be extended to higher order derivatives Viewing Eq. (1) as a linear transformation of u,

$$u_x = Au, \tag{2}$$

we see that the second-order derivatives can be approximated by

$$u_{xx} = (u_x)_x = A(Au) = A^2u, \tag{3}$$

Thus higher-order derivative approximations are obtained by iterating the linear transformation A.

When the method is applied to two and higher dimensional systems, the choice of N becomes critical. The previous results and the following numerical experiment shows that low order approximations can be expected to yield both qualitative and quantitative information.

Let us now apply the foregoing ideas to the treatment of Burger's equation [6]

$$u_t + uu_x = \epsilon u_{xx}, \qquad \epsilon > 0 \tag{4}$$

with pure initial value condition

$$u(0, x) = f(x) \tag{5}$$

Burger's equation enjoys a Riccati-like property in the sense that its solutions are expressible in terms of the solution of the linear heat equation

$$w_t = \epsilon w_{xx}, \tag{6}$$

$$w(0, x) = g(x), \tag{7}$$

by the transformation

$$u = -2\epsilon w_x/w, \tag{8}$$

$$f(x) = 2\epsilon g_x(x)/g(x). \tag{9}$$

This property will allow us to compare our numerical results with analytic solutions of Eqs. (6)—(9).

Adopting the notation

$$u_i(t) = u(t, x_i) \tag{10}$$

yields the set of ordinary differential equations

$$u_i'(t) + u_i(t) \sum_{j=1}^{N} a_{ij}u_j(t) = \epsilon \sum_{k=1}^{N} \sum_{j=1}^{N} a_{ik}a_{kj}u_j(t), \qquad i = 1, 2,\dots, N. \tag{11}$$

The initial conditions at $t = 0$ are determined by the initial function

$$u_i(0) = f(x_i), \qquad i = 1, 2,\dots, N. \tag{12}$$

In the following numerical experiment, the initial function $f(x)$ was chosen to be

$$f(x) = -2\epsilon[b\pi \cos \pi x + c\pi/2 \cos \pi x/2]/[b \sin \pi x + c \sin \pi x/2 + d], \tag{13}$$

where b, c, d are constants. The analytic solution of the heat equation in this case is

$$w(t, x) = be^{-\epsilon\pi^2 t} \sin \pi x + ce^{-\epsilon\pi^2 t/4} \sin \pi x/2 + d. \tag{14}$$

Table V summarizes the results of two cases $N = 7$ and 9 order differential quadrature. The values of the constants are taken to be $\epsilon = 0.01$, $b = 0.2$, $c = 0.1$ and $d = 0.3$. The relative errors e_u and e_w correspond to the Burgers' and the heat equation, respectively. Values of x_k are those listed in Table I.

TABLE V

Differential Quadrature of Orders 7 and 9

	$N = 7$			$N = 9$		
t	x	e_u	e_w	x	e_u	e_w
0.1	x_1	4.E-04	2.E-05	x_1	1.E-05	6.E-07
0.1	x_3	2.E-05	2.E-07	x_3	2.E-07	2.E-09
0.1	x_4	3.E-06	2.E-07	x_5	8.E-09	1.E-09
0.1	x_5	3.E-05	2.E-07	x_7	2.E-07	2.E-08
0.1	x_7	5.E-04	2.E-05	x_9	1.E-05	4.E-07
0.5	x_1	1.E-03	2.E-04	x_1	8.E-05	9.E-06
0.5	x_3	9.E-06	4.E-07	x_3	1.E-06	7.E-08
0.5	x_4	1.E-06	3.E-07	x_5	3.E-09	2.E-09
0.5	x_5	1.E-05	3.E-07	x_7	2.E-06	6.E-08
0.5	x_7	2.E-03	1.E-04	x_9	1.E-04	6.E-06
1.0	x_1	5.E-03	6.E-04	x_1	3.E-04	4.E-05
1.0	x_3	8.E-05	3.E-06	x_3	1.E-05	1.E-06
1.0	x_4	5.E-06	2.E-07	x_5	2.E-09	1.E-09
1.0	x_5	1.E-04	3.E-06	x_7	2.E-05	9.E-07
1.0	x_7	5.E-03	5.E-04	x_9	3.E-04	3.E-05

7. Error Representation

We are interested in finding the error $R'(x)$ in the N-th order differential quadrature

$$u'(x_i) = \sum_{j=1}^{N} a_{ij}u(x_j) + R'(x_i). \tag{1}$$

The well-known interpolation error $R(x)$ vanishes at N points x_j, $j = 1, N$ and has N continuous derivatives (provided that $u^{(N)}(x)$ is continuous). But it is not generally true that $R^{(N)}(x)$ is the N-th derivative of the above function. In order to obtain a practical error estimate we observe that by virtue of Rolle's theorem, $R'(x)$ has at least one distinct root $\bar{x}_j$ ($j = 1, N - 1$) in the open interval (x_j, x_{j+1}). We may also apply Rolle's theorem to the function

$$r(X) = R'(X) - b \prod_{j=1}^{N-1} (X - \bar{x}_j), \tag{2}$$

where $b(x)$ is picked in such a way that $r(x) = 0$ for any fixed x distinct from the $\bar{x}_j$. So that $r^{(N-1)}(X)$ is continuous and has a root $\bar{\bar{x}}$ in the interval containing x and the $\bar{x}_j$. From $r^{(N-1)}(\bar{\bar{x}}) = 0$ follows that

$$b = u^{(N)}(\bar{\bar{x}})/(N - 1)!\,. \tag{3}$$

Now substituting this value of b in $r(x) = 0$ we obtain the final result

$$R'(x) = \frac{u^{(N)}(\bar{\bar{x}})}{(N-1)!} \prod_{j=1}^{N-1} (x - \bar{x}_j) \tag{4}$$

which is valid for *all* x.

The same sort of argument shows that the error $R''(x)$ in the approximation $u_{xx} = A^2u$ of Section 6 can be written as in Eq. (5) with $x_j < \bar{x}_j < x_{j+2}$.

$$R''(x) = \frac{u^{(N)}(\bar{\bar{x}})}{(N-2)!} \prod_{j=1}^{N-2} (x - \bar{x}_j). \tag{5}$$

Assuming that all the x_j and x lie in an interval h and that in this interval $|u^{(N)}(x)| \leqslant K$, the bound on the error of Eqs. (4) and (5) is

$$|R'(x)| \leqslant K \frac{h^{N-1}}{(N-1)!}, \tag{6}$$

$$|R''(x)| \leqslant K \frac{h^{N-2}}{(N-2)!}. \tag{7}$$

References

1. A. Kasahara, Simulation of the earth's atmosphere, *in* "Computational Approaches in Applied Mechanics" (E. Sevin ed.), American Society Mechanical Engineers, New York, 1969.
2. R. Bellman, "Introduction to the Mathematical Theory of Control Processes, Vol. 1, Academic Press, New York, 1967.

3. R. Bellman, H. Kagiwada and R. Kalaba, Invariant imbedding and radiative transfer in spherical shells, *J. Comp. Physics*, **1** (1966), 245.
4. R. Bellman, H. Kagiwada, R. Kalaba and M. Prestrud, "Invariant Imbedding and Time Dependent Transport Processes," American Elsevier, New York, 1964.
5. R. Bellman, R. Kalaba and J. Lockett, "Numerical Inversion of the Laplace Transform, American Elsevier, New York, 1966.
6. E. Hopf, *Commun. Pure Appl. Math.*, **3** (1950), 201.