JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

DIFFERENTIAL EQUATIONS : RECURRENCE RELATIONS FOR ODE

[1]Pavulurimachara Srinivas
Assistant Professor, Department of Mathematics
pmsrinivas@giet.ac.in
GIET Engineering College, Rajahmundry. A.P,India
[2]Vanka Kathyayani
Assistant Professor, Department of Mathematics
kathyayani@giet.ac.in
GIET Degree College Rajahmundry.A.P,India
[3]Guttula Surya Jyothi
Assistant Professor, Department of Mathematics
Suryajyothi1983@gmail.com
GIET Engineering College, Rajahmundry.A.P,India

Abstract :
Newton's identities can be used to compute the coefficients of multi-step recurrence formulae for simulating the solutions of ordinary linear differential equations of high order. This approach gives the exact root-matched recurrence formula without having to determine the complex roots of the characteristic polynomials.

Keywords : Differential Equations, exact root, complex roots and coefficients of multi-step recurrence formula

1. INTRODUCTION

Let x(t) and y(t) denote continuous functions of time related according to the Nth order differential equation

$$\sum_{k=0}^{N} a_k \frac{d^k y}{dt^k} = \sum_{k=0}^{N} b_k \frac{d^k x}{dt^k} \qquad a_N, a_0, b_N \neq 0 \qquad (1)$$

Where,
The coefficients a_k and b_k are constants.

Simple linear relationships of this form are widely used in numerical simulations of heat transfer, chemical reactions, fluid flows, mechanical dynamics, Markov processes, etc. [1] (Relationships of this form are also used in signal filtering and control-system compensation, but in those applications the frequency response is usually emphasized over the exact time-domain response.) A variety of methods (cf [2,3]) have been used to numerically generate the solutions y(t) of such equations for arbitrary forcing functions x(t). In the digital environment equation (1) is often simulated by means of a multi-step linear recurrence relation of the form

$$\mathbf{h}\, \mathbf{Y}_n = \mathbf{g}\, \mathbf{X}_n \qquad (2)$$

Where

Xn and **Y**n are the column vectors

$$\mathbf{Y}_n = \begin{bmatrix} y((n-N)\Delta t) \\ \vdots \\ y((n-N)\Delta t) \\ y(n\Delta t) \end{bmatrix} \qquad \mathbf{X}_n = \begin{bmatrix} x((n-N)\Delta t) \\ \vdots \\ x((n-N)\Delta t) \\ x(n\Delta t) \end{bmatrix}$$

for some constant Δt (called the *step-size* of the simulation), and **g** and **h** are constant row vectors

$$\mathbf{h} = [h_0 \ h_1 \ \dots \ h_N] \qquad \mathbf{g} = [g_0 \ g_1 \ \dots \ g_N]$$

Given the coefficients ak and bk of (1), our objective is to compute, as quickly and efficiently as possible, the components of **h** and **g**.

It is well known [4] that if the recurrence is required to match the exact homogeneous response for y(t) when x(t) is constantly zero, then the components of **h** must be proportional to the elementary symmetric functions [8] (with alternating signs) of the quantities $e^{\alpha_i \Delta t}$, i = 1, 2, .., N, where αi are the roots of the characteristic polynomial of the left side of (1). The usual procedure for determining this "root-matched" recurrence is to solve the characteristic polynomial for the roots αi, and then compute the elementary symmetric functions of these roots [5]. For high order relationships the most troublesome aspect of this approach is the determination of all the complex roots of the characteristic polynomials, especially when there are multiple roots [6]. The most common way of simplifying the calculations is to use a substitution method, e.g., bilinear substitution [7], but such methods do not reproduce the exact homogeneous response. The purpose of this note is to describe how, using Newton's Identities, we can proceed directly from the coefficients of (1) to the exact root-matched recurrence coefficients without solving the characteristic polynomials.

2. DETERMINING THE ROOT MATCHED RECURRENCE

For any integer k, let w(k) denote the sum of the kth powers of the roots $\alpha_1,\alpha_2,..,\alpha_N$. Clearly, w(0) = N. Using Newton's Identities [8,9] the values of w(k) for k = 1,2,... are given by

$$\begin{aligned} &a_N w(1) + 1a_{N-1} = 0 \\ &a_N w(2) + a_{N-1} w(1) + 2a_{N-2} = 0 \\ &a_N w(3) + a_{N-1} w(2) + a_{N-2} w(1) + 3a_{N-3} = 0 \end{aligned}$$

for k < N, and the recurrence

$$a_N w(k) + a_{N-1} w(k-1) + \dots + a_0 w(k-N) = 0$$

for all k ≥ N.

Now let E(k) denote the sum of the kth powers of the values of $e^{\alpha_i \Delta t}$, i = 1, 2, .., N. Expanding each of the exponential functions as a Taylor's series and combining terms by powers of Δt gives the following expression for E(k):

$$E(k) = \sum_{j=0}^{\infty} \frac{(k\Delta t)^j}{j!} w(j) \qquad (3)$$

Letting σ_k denote (–1)k times the kth elementary symmetric function of the quantities $e^{\alpha_i \Delta t}$,
we can now compute σ_0 through σ_N using the identities

$$E(1) + 1\sigma_1 = 0$$
$$E(2) + \sigma_1 E(1) + 2\sigma_2 = 0$$
$$E(3) + \sigma_1 E(2) + \sigma_2 E(1) + 3\sigma_3 = 0$$
$$\text{etc.}$$

The precision of these computations, including the convergence of (3), can be checked by verifying the equality

$$\sigma_N = (-1)^N e^{-a_{N-1}\Delta t / a_N}$$

The general recurrence for the homogeneous equation in y(t) can now be expressed as:

$$\mathbf{S}_A \mathbf{Y}_n = 0$$

where **S**A is the row vector

$$\mathbf{S}_A = [\sigma_N \quad \sigma_{N-1} \quad \ldots \quad \sigma_0]$$

Figure 1 presents an implementation of the preceding algorithm in BASIC. The inputs to this program are the order N, the step-size Δt, and the coefficients a_0, a_1 .., a_N. The program computes the values of σ_0, σ_1,..,σ_N. The parameter "sumto" determines the number of terms to be evaluated in the summation (3).

By entering the coefficients b_k in place of a_k into the preceding algorithm we can determine the recurrence for the homogeneous equation in x(t)

$$\mathbf{S}_B \mathbf{X}_n = 0$$

We can then combine the two sides of the recurrence by applying the appropriate scale factors, which are easily deduced from steady-state conditions. The result is

$$\frac{a_0}{\sum \mathbf{S}_A} S_A Y_n = \frac{b_0}{\sum \mathbf{S}_B} S_B X_n \qquad (4)$$

Where
Σ**S**A and Σ**S**B denote the sums of the components of **S**A and **S**B respectively.

3. EQUILATERAL RECURRENCES

If $b_N = 0$ then the characteristic polynomial of the right side of (1) has fewer than N roots. In this case the recurrence (4) would involve fewer "past values" of x(t) than of y(t). Note that the coefficients of x_n in (4) were uniquely determined by the requirement that x(t) conform to the true homogeneous behavior when y(t) is constantly zero. However, in practice x(t) is an arbitrary forcing function so we are not really concerned with reproducing the homogeneous "response" of x(t). Instead, we are trying to reconstruct the function x(t) from the discrete samples. Thus it is often considered more appropriate to make use of as many past values of x(t) as is computationally convenient. This typically leads to recurrences that use the same number of past values of x(t) as of y(t), i.e., equilateral recurrences. The usual approach in signal processing is to augment the roots of the right-hand characteristic polynomial of (1) with fictitious roots of the form ±πi, and then proceed as before . The effect is to add roots equaling –1 to the characteristic polynomial of the recurrence, which essentially mimics bi-linear substitution.

The rationale for the "method of fictitious roots" just described is based on the frequency response characteristics. In the time domain it is often more appropriate to convert (1) into an equilateral equation by a simple change of variables. If we define the variable z(t) such that
y = λ(x+z) for some constant λ, then (1) can be written

$$\sum_{k=0}^{N} a_k \frac{d^k z}{dt^k} = \sum_{k=0}^{N} \left(\frac{b_k}{\lambda} - a_k \right) \frac{d^k x}{dt^k} \qquad (5)$$

For any choice of λ not equal to 0, (b_N/a_N), or (b_0/a_0), this equation is equilateral, so we can apply the method of (4) to give the recurrence

$$\mathbf{h}\,\mathbf{Z}_n = \mathbf{g}_\lambda\,\mathbf{X}_n$$

where $\mathbf{g}_\lambda$ is based on the coefficients of the right hand side of (5). Reverting back to the y(t) variable gives

$$\mathbf{h}\,\mathbf{Y}_n = \lambda(\mathbf{g}_\lambda + \mathbf{h})\,\mathbf{X}_n$$

For large values of λ the vector $\lambda(\mathbf{g}_\lambda + \mathbf{h})$ is quite insensitive to changes in λ, and it converges in the limit as $|\lambda| \to \infty$. In practice, any λ greater than about 10 times the largest b_i will give very nearly the limiting vector. This "change-of-variables method" enables us to determine equilateral recurrences for non-equilateral differential equations without resorting to fictitious roots.

4. THE EXPECTED VALUE METHOD

For comparison we briefly describe a method that explicitly treats x(t) as an independent variable and y(t) as a dependen*t* variable. From this point of view it can be argued that the expected *value* of x(t) based on the N+1 most recent samples is given by the Nth degree polynomial that passes through those samples. Although the method is a straight-forward application of polynomial curve fitting, no explicit formulation of the general Nth-order recurrence formula appears in the literature, so a brief description is given below.

As in our previous recurrence formulas, we require that the homogeneous response of y(t) be matched exactly, so we use the algorithm shown in Figure 1 to determine **S**A. Combining this with the particular solution based on the Nth order polynomial curve fit of x(t) leads directly to the total recurrence

$$\mathbf{S}_A\mathbf{Y}_n = \mathbf{S}_A\mathbf{T}\mathbf{A}^{-1}\mathbf{B}\mathbf{T}^{-1}\mathbf{X}_n$$

where
A, **B**, and **T** are square matrices such that the elements in the jth columns of the kth rows (where j,k range from 0 to N) are given by

$$A_{kj} = \begin{cases} \frac{j!}{k!} a_{j-k} & \text{if } j \geq k \\ 0 & \text{if } j < k \end{cases} \qquad B_{kj} = \begin{cases} \frac{j!}{k!} b_{j-k} & \text{if } j \geq k \\ 0 & \text{if } j < k \end{cases} \qquad T_{kj} = \begin{cases} (k\Delta t)^j & \text{if } k+j > 0 \\ 1 & \text{if } k+j = 0 \end{cases}$$

This method gives, arguably, the best possible reconstruction of arbitrary forcing functions x(t), although the matrix inversions make it somewhat laborious when applied to high order equations.

5. EXAMPLES

To illustrate the basic algorithm for determining the root-matched recurrence relation for a homogeneous equation, consider the differential equation

$$\frac{d^5y}{dt^5} + 7\frac{d^4y}{dt^4} + 21\frac{d^3y}{dt^3} + 33\frac{d^2y}{dt^2} + 28\frac{dy}{dt} + 10y = 0 \qquad (6)$$

To simplify the recurrence expressions we will denote $y(n\Delta t)$ by y_n. Taking a time interval of $\Delta t = 0.1$, the algorithm described in Section 2 gives the following recurrence formula for y_n

$$y_n = -h_4y_{n-1} - h_3y_{n-2} - h_2y_{n-3} - h_1y_{n-4} - h_0y_{n-5} \qquad (7)$$

where

$$\begin{aligned} h_0 &= -0.4965853038 \\ h_1 &= 2.8479524294 \\ h_2 &= -6.5428490609 \\ h_3 &= 7.5263050469 \\ h_4 &= -4.3347524368 \end{aligned}$$

(Notice that we've scaled the coefficients to make h_5 equal to 1.) The accuracy of this recurrence can be illustrated by noting that (6) has the characteristic roots –1, –2+i, –2–i, –1+i, and –1–i, so one particular analytical solution has the simple form

$$y_n = -8e^{-n\Delta t} + \left(-6e^{-2n\Delta t} + 14e^{-n\Delta t}\right)\cos\left(n\Delta t\right) \qquad (8)$$

Taking y_0 through y_4 from (8) as initial values, it is easily verified that the recurrence (7) numerically reproduces the exact analytical values of y_5, y_6, y_7 ... etc., very precisely. After 35 iterations of the recurrence formula (carried out to 10 significant decimal digits) the value of y_{40} differs from the analytical value by only 0.0000013, this discrepancy being due entirely to accumulated round-off error. In contrast, using the recurrence relation given by bilinear substitution, the error at y_{40} is 0.0013094. For larger step-sizes the error using bilinear substitution becomes progressively worse, whereas the root-matched recurrence method is equally applicable to any step-size.

We chose a homogeneous equation with simple characteristic roots for the preceding example so that the recurrence values could be easily compared to the analytical solution. Also, the characteristic roots in the preceding example all have negative real parts, so the solutions are "stable". However, our interest is not limited to "stable" systems. For example, problems involving combustion and thermal "run-away" often lead to "unstable" systems. To illustrate the various methods discussed in this note as applied to one of these "unstable" systems, and to include "numerator dynamics" [1], consider the differential equation

$$\frac{d^5y}{dt^5} - \frac{d^3y}{dt^3} - 2\frac{d^2y}{dt^2} + y = \frac{d^3x}{dt^3} - \frac{dx}{dt} - x$$

Taking a time interval of $\Delta t = 1$, the relation between x and y can be simulated using a linear recurrence of the form (2), where the components of **h** and **g** are as listed in the table below (normalized for $h_5 = 1$).

	h_i		g_i			
i	all except bilinear substitut	bilinear substitut	expected value method	change-of variables method	fictitious roots method	bilinear substit
0	-1.000000	-1.823529	-0.083184	-0.083311	-0.250012	-0.411765
1	5.060084	8.764706	-0.530963	-0.537182	0.386879	0.529412
2	-11.732310	-18.235294	2.456330	2.489417	0.365333	0.588235
3	14.904837	21.294118	-3.428658	-3.485152	-1.180006	-1.764706
4	-7.146380	-9.117647	0.407632	0.446686	-0.658437	-1.117647
5	1.000000	1.000000	0.092613	0.083311	0.250012	0.294118

All the methods except Bilinear Substitution made use of the BASIC program in Figure 1 to determine the components of **h** and to assist in determining the components of **g**. Interestingly, the Change-of-Variables Method gives components of **g** that closely match those given by the Expected Value Method, even though the former makes no use of a 5th order curve fit of x(t). In contrast, the methods of fictitious roots and bilinear substitution give very different **g** vectors, and therefore will clearly yield significantly different values of y(t) in response to certain forcing functions x(t).

6. CONCLUSIONS

The technique described in this paper uses Newton's Identities to efficiently determine the root-matched recurrence relations for numerically simulating linear ordinary differential equations. The algorithm is simple to implement and can easily be incorporated into any numerical simulation program. The main benefit of the method is that it eliminates the need to determine the characteristic roots of the differential equation, which can be difficult for high order equations. This makes the method particularly useful in real-time applications that require frequent computation of recurrence relations corresponding to given continuous transfer functions. The recurrences given by this method (unlike bi-linear substitution) are valid for any step size Δt, provided only that the sampling period is small enough to adequately resolve the forcing function x(t). Also, the algorithm requires no special handling of characteristic equations with repeated roots, so it is applicable to any equation of the form (1). We have also shown how the "Change-of-Variables" method can be used to generate robust equilateral recurrences without resorting to fictitious roots.

References

1. E. O. Doebelin, System Dynamics, Modeling and Response, Charles E Merrill Publishing Company, 1972.

2.V. Raman, "Explicit Multistep Methods In System Simulation", IEEE International

3.Symposium on Circuits and Systems, 1989, vol 3, p 1792-1795.

4. S. Sallam, W. Ameen, "Numerical Solution of General nth-Order Differential Equations Via Splines", Applied Numerical Mathematics, Vol 6, 1989/90, pp 225-238.

5.A. L. Rabenstein, Elementary Differential Equations With Linear Algebra, Academic Press, 1970.

6. J. M. Smith, Mathematical Modeling and Digital Simulation For Engineers and Scientists, 2nd Ed, John Wiley & Sons, 1987.

7. T. Miyakodo, "Iterative methods for multiple zeros of a polynomial by clustering", Journal of Computational and Applied Mathematics, vol 28 (1989), p 315-326.

8.V. J. Bucek, Control Systems, Continuous and Discrete, Prentice Hall, 1989.

9.A. Clark, Elements of Abstract Algebra, Dover Publications, 1984.

10. I. N. Herstein, Topics In Algebra, 2nd Ed, John Wiley & Sons, 1975.

11. G. F. Franklin, J. David Powell, M. L. Workman, Digital Control of Dynamic Systems, 2nd Ed, Addison-Wesley, 1990.