System theoretic implications of numerical methods applied to the solution of ordinary differential equations

by T. G. WINDEKNECHT and H. D'ANGELO
Memphis State University
Memphis, Tennessee

ABSTRACT

This paper presents a system-theoretic analysis of numerical methods used in approximating solutions of ordinary differential equations. By representing ordinary differential equations with system block diagrams (i.e., interconnections of static elements and integrators), a numerical method can be viewed as a process in which the integrators of a continuous system are replaced by discrete approximations to the integrators (i.e., by discrete subsystems made up of interconnections of static elements and delays). The main result of the paper establishes that if the system block diagram corresponding to the original differential equation has no static loops and if the discrete subsystems used to replace the integrators have no static loops and no static through paths, then the resulting discrete system can be characterized by explicit difference equations. A system-theoretic study is conducted of several of the more commonly used numerical methods (the Euler, trapezoidal, Runge-Kutta, and predictor-corrector methods) and the limitations of using these numerical methods in the real-time analysis of input-output systems are examined.

INTRODUCTION

Important insight into numerical methods used in the solution of ordinary differential equations can be obtained by studying these numerical methods from a system theoretic standpoint. The system block diagram (an interconnection of static and dynamic elemental components) has proved to be particularly helpful in this regard. With this point of view, a useful categorization of numerical methods is obtained.

THE PROBLEM

The problem addressed in this paper is the following: Determine a numerical solution to the system of ordinary differential equations given in the standard canonical form

$$\frac{dx(t)}{dt} = f(x(t), u(t)), x(t_0) = x_0, t \in [t_0, t_f] \quad (1)$$

where $x$ is an $n$-vector, $u$ is an $m$-vector, and $f$ is a function such that $f : \mathbb{R}^{n+m} \rightarrow \mathbb{R}^n$.

The ultimate objective in applying a numerical method is to compute a sequence

$$\hat{x}(t_0), \hat{x}(t_0 + \delta), \hat{x}(t_0 + 2\delta), \ldots, \hat{x}(t), \ldots, \hat{x}(t_f) \quad (2)$$

which is, in some sense, a good approximation to the sequence

$$x(t_0), x(t_0 + \delta), x(t_0 + 2\delta), \ldots, x(t), \ldots, x(t_f) \quad (3)$$

where the continuous variable $x$, defined on the interval $[t_0, t_f]$ is the solution of equation (1).

It is generally computationally advantageous to be able to compute sequence (2) as a solution to a set of difference equations in the standard state-variable form

$$v(t + \delta) = f_s(v(t), u(t)) \quad (4a)$$

$$x(t) = E \, v(t) \quad (4b)$$

where $x$ is an $n$-vector, $v$ is an $N$-vector ($N \geq n$), $u$ is an $m$-vector, and $f_s$ is a function such that $f_s : \mathbb{R}^{n+m} \rightarrow \mathbb{R}^N$, $g$ is a function such that $g : \mathbb{R}^n \rightarrow \mathbb{R}^N$, $E$ is an $n \times N$ matrix of rank $n$ whose elements are either 0 or 1 with one and only one 1 in each row (thus, the components of $x$ are a subset of the components of $v$), and $T_d$ is the discrete time set $\{t_0, t_0 + \delta, t_0 + 2\delta, \ldots, t_f\}$.

The form of difference equations (4), in which $f_s$ does not depend on $v(t + \delta)$, is said to be explicit. Explicit difference equations are computationally attractive since, for most functions $f_s$, they allow the approximation sequence (2) to be generated in a straightforward iterative fashion. More precisely:

Theorem 1: Explicit difference equation (4) has a unique solution provided only that $f_s$ is defined at each
stage of iteration. The solution can be generated on a computer provided that \( f_a \) is a computer function and that \( |v(t+\delta)| \) is no larger than the largest number in the computer.

However, under certain severe modeling constraints, it may be necessary to settle for less attractive numerical methods in which the approximation sequence (2) is computed as a solution to a set of difference equations of the form

\[
v(t+\delta) = f_a(v(t), v(t+\delta), u(t))\tag{5a}
\]

\[
x(t) = E(v(t))\tag{5b}
\]

where \( f_a \) is a function such that \( f_a: \mathbb{R}^{n+m} \rightarrow \mathbb{R}^n \). Difference equations of this form, in which \( f_a \) depends on \( v(t+\delta) \), are said to be \textit{implicit}. The computational difficulties associated with implicit difference equations stem from the fact that, depending on the nature of the function \( f_a \), it may not be possible to solve for \( v(t+\delta) \) and thus obtain the desirable recursive form of equations (4). In such cases, where the solutions are known to exist, and they may not exist, computing \( v(t+\delta) \) from a knowledge of \( v(t) \) and \( u(t) \) is not straightforward. Often, an iterative method of solution, such as a Newton method, with generally unknown convergence properties, is required to approximate \( v(t+\delta) \). Certainly, one should avoid this latter form if possible.

**SYSTEM BLOCK DIAGRAMS**

In system theory it is sometimes convenient to represent sets of equations characterizing dynamic systems by interconnections of elemental components. Systems characterized by differential equations, and thus defined on the continuous time set \([t_0, t_f] \), \( t_f > t_0 \), are called \textit{continuous systems}; systems characterized by difference equations, and thus defined on the discrete time set \( T_d \), are called \textit{discrete systems}. A sufficient set of elemental components for synthesizing the necessary interconnections consists of the general static element and two fundamental dynamic elements, the delay and the integrator. Each elemental component defines a causal relationship between an \textit{output variable} and one or more \textit{input variables}. The set of elemental components is defined:

a. A general static element, represented graphically in Figure 1a, is a functional relationship between \( q \) input variables and an output variable. It has the form

\[
v(t) = f(v_1(t), v_2(t), \ldots, v_q(t))\tag{6}
\]

where \( f \) is a function such that \( f: \mathbb{R}^q \rightarrow \mathbb{R} \).

b. A \( \delta \)-delay element, represented in Figure 1b, is a single-input, single-output element satisfying the quasi-functional relationship between two dynamic variables \( v \) and \( y \)

\[
y(t+\delta) = v(t), \ t \in T_d\tag{7}
\]

A fixed-structure discrete dynamic system is any \textit{consistent} interconnection of delays and static elements. An interconnection is consistent if no two outputs of elemental components are connected together (i.e., no two outputs should represent the same variable). Similarly, a fixed-structure continuous dy-
namic system is any consistent interconnection of integrators and static elements. Such interconnections of elemental systems are called system block diagrams.

It is important to establish the conditions under which system block diagrams are characterized by canonical state variable equations, i.e., by equations of the form

\[ x(t+\delta) = f(x(t), u(t)) \quad t \in \mathbb{T}_d \]  

(9)

for discrete systems or by equations of the form

\[ \frac{dx(t)}{dt} = f(x(t), u(t)) \quad t \in [t_m, t_f] \]  

(10)

for continuous systems. Toward this end, the concept of a proper interconnection is introduced. An interconnection is proper provided every closed path (in the direction of the arrows) on the system block diagram contains at least one delay element (or, in the case of a continuous system, one integrator). A closed path that does not contain a delay (or integrator) is called a static loop. Thus, a system with no static loops is a proper interconnection.

Two important results relating to system block and canonical state equations are found in References 2 and 3.

**Theorem 2** Every proper interconnection of static elements and delays admits a state-variable characterization of the form of equations (9); every proper interconnection of static elements and integrators admits a state-variable characterization of the form of equations (10).

**Theorem 3** Every set of difference equations in state-variable canonical form (equations (9)) can be represented by a proper interconnection of static elements and delays; every set of differential equations in state-variable canonical form (equations (10)) can be represented by a proper interconnection of static elements and integrators.

**COMPUTATIONAL IMPLICATIONS OF STATIC LOOPS**

Consider the system block diagrams in Figure 2 corresponding to the following two first-order difference equations, the first explicit and the second implicit:

\[ v(t+\delta) = f_1(v(t), u(t)) \]  

(11a)

\[ v(t+\delta) = f_2(v(t), v(t+\delta), u(t)) \]  

(11b)

Note that the explicit first-order difference equation can always result in a proper system block diagram (i.e., one without static loops), whereas the implicit first-order difference equation always results in a system interconnection with a static loop.

For higher order systems, the results are almost the same: Explicit difference equations (4a) can always result in a proper system block diagram, whereas implicit difference equations (5a) generally result in system interconnections with a static loop. The exceptional case where an implicit difference equation does not result in any static loops is the somewhat trivial case where a simple reindexing of variables renders the set of difference equations in the following "lower triangular form":

\[
\begin{align*}
v_1(t+\delta) &= f_1(v(t), u(t)) \\
v_2(t+\delta) &= f_2(v(t), v_1(t+\delta), u(t)) \\
& \quad \vdots \\
v_N(t+\delta) &= f_N(v(t), v_1(t+\delta), v_2(t+\delta), \ldots, v_{N-1}(t+\delta), u(t))
\end{align*}
\]

(12)

where \( v \) is an \( N \)-vector consisting of the scalar components \( v_1, v_2, \ldots, v_N \), and \( f_i: \mathbb{R}^N \rightarrow \mathbb{R} \) is a function such that \( f_i: \mathbb{R}^N \rightarrow \mathbb{R} \). Note that this is a rather exceptional implicit form in that

a. The sequence \( v(t), v(t+\delta), \ldots, v(t_f) \) can be computed in a simple iterative fashion.

b. This form admits a system block diagram representation with no static loops.

That the above statements a and b are equivalent statements is easily established by showing that equations (12) can always be put into explicit form by substituting the first equation into the second equation to eliminate \( v_1(t+\delta) \), then substituting the first and new second equation into the third equation to eliminate \( v_2(t+\delta) \) and \( v_3(t+\delta) \), etc. Therefore, the computationally desirable form can be associated with system block diagrams having no static loops and, with the exception of the "lower triangular forms", the computationally difficult implicit form can be associated with block diagrams having at least one static loop.

**DISCRETE APPROXIMATIONS TO CONTINUOUS SYSTEMS**

It can be shown that every ordinary differential equation in canonical state-variable form can be represented by a proper system block diagram (i.e., an inter-
connection of static elements and integrators containing no static loops). For example, the first-order differential equation

\[
\frac{dx(t)}{dt} = f(x(t), u(t))
\]  

(13)
is represented by the system block diagram shown in Figure 3. Similarly, the set of two simultaneous first-order differential equations

\[
\begin{align*}
\frac{dx_1(t)}{dt} &= f_1(x_1(t), x_2(t), u(t)) \\
\frac{dx_2(t)}{dt} &= f_2(x_1(t), x_2(t))
\end{align*}
\]  

(14)
is represented by the system block diagram shown in Figure 4.

One may, taking a system-theoretic viewpoint, interpret most numerical methods for approximating solutions to ordinary differential equations to be equivalent to substituting discrete subsystems for the integrators in the system, thus converting the continuous system to a discrete system. In some numerical methods integrators are replaced by single-input, single-output discrete subsystems on an individual basis whereas in other numerical methods all the integrators are considered to make up a single subsystem and this subsystem is replaced as a single entity. In the latter case, the n integrators in a system are considered to be an n-input, n-output subsystem consisting of n parallel non-interacting integrators; this subsystem is then replaced by an n-input, n-output discrete subsystem. We speak of these subsystems which replace the integrators in a system as discrete approximations to integration or simply as discrete integrators. If the resulting discrete system has no static loops, such a substitution for the integrators in a proper system block diagram of a continuous system have

(i) no static loops
(ii) no static through paths (i.e., paths from input to output without a delay element)

then the system block diagram of the resulting discrete system is also proper and can therefore be characterized by the set of explicit difference equations (4).

As an example, consider using the Euler zero-order approximation to integration in obtaining a numerical solution to the following first-order differential equation:

\[
\frac{dx(t)}{dt} = \cos(x(t)u(t)), \quad x(0) = 1
\]  

(15)
The system block diagram corresponding to equation (15) is shown in Figure 5. An integrator with input \( v(t) \) and output \( x(t) \), shown in Figure 6a, is characterized by the equation

\[
x(t+\delta) = x(t) + \int_t^{t+\delta} v(\tau) d\tau
\]  

(16)
The Euler method approximates the definite integral by a rectangle:

\[
\int_t^{t+\delta} v(\tau) d\tau \approx \delta v(t)
\]  

(17)
Thus, using the Euler approximation in equation (16) gives the discrete approximation to integration:

\[
x(t+\delta) = x(t) + \delta v(t)
\]  

(18)
The system block diagram for this discrete approximation to integration is shown in Figure (6b). Note that
the Euler discrete approximation to integration has neither static loops nor static through paths.

Replacing the integrator in the system block diagram of Figure 5 by the Euler discrete approximation to it, Figure 6b, results in the discrete system shown in Figure 7. Importantly, the resulting discrete system has no static loops and is thus characterized by an explicit difference equation:

\[ x(t+\delta) = x(t) + \delta \cos(x(t), u(t)) \] (18)

Some other well-known discrete approximations to integration and their system theoretic interpretations are given in the following section.

**EXAMPLES OF SOME FREQUENTLY USED NUMERICAL METHODS**

In this section some well-known numerical methods will be studied from a system theoretic viewpoint. Much of the literature on the numerical analysis of ordinary differential equations deals with the problem of finding solutions to the class of differential equations defined by

\[ \frac{dx(t)}{dt} = f(x(t), t) \] (18a)

Note that the class of differential equations defined by equation (18a) is somewhat narrower than the class defined by equation (1). Specifically, equation (1) is equivalent to equation (18a) only for the special case that

\[ u(t) = t \]

\[ \int_t^{t+\delta} v(\tau) d\tau = \frac{\delta}{2} (v(t) + v(t+\delta)) \] (19)

Thus, the relation between the input \( v \) and the output \( x \) of the trapezoidal approximation to integration is

\[ x(t+\delta) = x(t) + \frac{\delta}{2} (v(t) + v(t+\delta)) \] (20)

A system block diagram of the system characterized by equation (20) is shown in Figure 8. The difficulty with this system is that although it provides \( x(t) \) as an output, it requires \( v(t+\delta) \) as an input; \( v(t) \) should be the input if this discrete system is to be used as a substitution for integration. This difficulty can be resolved by introducing a time shift in the original continuous system which is to be discretized for computational purposes. For example, the continuous system shown in Figure 5 can be relabeled with the necessary time shift as shown in Figure 9. With such a time shift we can consider using a discrete approximation to an integrator with input \( v(t+\delta) \) and output \( x(t+\delta) \). Such a subsystem is easily achieved with the trapezoidal approximation to integration of Figure 8 simply by considering the output to be the input of the rightmost delay element (i.e., \( x(t+\delta) \) rather than the output of that delay (i.e., rather than \( x(t) \)). Figure 10 shows this rearranging.

\[ u(t+\delta) \rightarrow f(t, \cdot) \rightarrow \frac{dx(t+\delta)}{dt} \rightarrow \sum \rightarrow x(t+\delta) \]

**Figure 10**—Time-shifted system block diagram for system of Figure 3
It can be seen that with \( v(t+\delta) \) as the input and \( x(t+\delta) \) as the output, the trapezoidal approximation to integration has a static through path. Substituting the trapezoidal integrator into the time-shifted original continuous system results in a discrete system with one static loop (Figure 11) which in turn results in implicit difference equations.

Note that two delay elements are used in the approximation to one integrator. Thus, two initial values are required for the resulting discrete system and the single initial value provided for the first-order continuous system is not sufficient to provide a unique solution for the approximating discrete system. For example, if the discrete process is started at \( t = -\delta \), one must have the two initial values \( f(x(0), u(0)) \) and \( x(0) \). In this case, computing the initial value \( f(x(0), u(0)) \) from the original initial value \( x(0) \) and the initial value of the input \( u(0) \) is relatively straightforward. However, in a systems sense, the time shift from \( t \) to \( t+\delta \) is a severe modification. As a result of the time shift, the discrete system operates in future time (i.e., at time \( t \), the system requires input \( u(t+\delta) \)).

If such an approximating system were required to operate in real time, then the time shift would represent an impossible realization. Of course one may decide to use the system, which as an interconnection of elements is realizable, in real time nevertheless. This implies shifting time back again from \( t+\delta \) to \( t \) and starting the process at \( t = 0 \). The difficulty with this is that the initial values required now are \( x(\delta) \) and \( f(x(\delta), u(\delta)) \) which, of course, are not generally known at time \( t \).

(ii) Runge-Kutta methods: The Runge-Kutta methods are based on computing \( x(t+\delta) \) as a perturbation of \( x(t) \) by approximating the terms in a truncated Taylor series with comparable terms which do not involve derivatives. The Runge-Kutta methods are typically given as a means to approximating solutions to the class of differential equations defined by equation (18) (i.e., the special case of equation (1) where \( u(t) = t \)). Although this is not the class of primary interest in system theory, let us present the systems implications of the method in its normal context. We will consider modifying it later.

Perhaps the most widely used Runge-Kutta method is of order four:

\[
x(t+\delta) = x(t) + \frac{\delta}{6} (k_1 + 2k_2 + 2k_3 + k_4)
\]

where

\[
k_1 = f(x(t), u(t))
\]

\[
k_2 = f\left(x(t) + \frac{\delta}{2} k_1, t + \frac{\delta}{2}\right)
\]

\[
k_3 = f\left(x(t) + \frac{\delta}{2} k_2, t + \frac{\delta}{2}\right)
\]

\[
k_4 = f\left(x(t) + \delta k_3, t + \delta\right)
\]

Thus, in this Runge-Kutta method, integration is approximated as follows:

\[
\int_{t}^{t+\delta} x(r) \, dr = \frac{\delta}{6} (k_1 + 2k_2 + 2k_3 + k_4)
\]

Figure 12 shows the original continuous system and the Runge-Kutta discrete approximation to it. It is noteworthy that:

(i) the Runge-Kutta integrator has only one delay element and no static loops or static through paths. Thus, only one initial value is required and, if the original continuous system contains no static loops, the resulting discrete system contains no static loops and an explicit set of difference equations results.

(ii) the Runge-Kutta integrator depends on the function \( f \), unlike the Euler and trapezoidal integrators considered earlier. Thus, the Runge-Kutta integrator is adaptive in the sense that the integration process varies as a function of the signal being integrated.

(iii) the values of \( t + \frac{\delta}{2} \) and \( t + \delta \) are obtained in the system simply by adding \( \frac{\delta}{2} \) and \( \delta \) to \( t \), respectively.

The point to the seemingly trivial observation (iii) is that static components are used to obtain future values of time. In particular, the computation of \( x(t+\delta) \) (for the subsequent evaluation of \( f\left(x(t) + \frac{\delta}{2} k_i, t + \frac{\delta}{2}\right), i = 1, 2, \) and \( f\left(x(t) + \delta k_3, t + \delta\right) \)) does not require the knowledge of any function of time for values of time greater than \( t \). In trying to use this Runge-Kutta method for finding the solutions of equation (1) \((u(t) \neq t)\), one is faced with evaluating \( f\left(x(t) + \frac{\delta}{2} k_i, u\left(t + \frac{\delta}{2}\right)\right), i = 1, 2, \) and \( f(x(t) + \delta, u) \).
dx(t) = f(x(t))  \quad (25)

This means that at the time x(t+δ) is computed, the input u must be known for times t + \frac{δ}{2} and t + δ as well as for time t. Thus, if a real-time continuous system starts at time t=0, the corresponding real-time Runge-Kutta discrete system cannot start until time t=δ, at which time the values u(0), u(\frac{δ}{2}), and u(δ) are available, in addition to the original initial condition x(0).

**IMPLICIT METHODS (THE PREDICTOR-CORRECTOR)**

The class of predictor-corrector methods for solving differential equations provides an excellent example of how an implicit method can sometimes be utilized in numerical analysis. In this section, the typical predictor-corrector is defined. The system-theoretic implications of the predictor-corrector integrator are noted. In particular, the predictor-corrector approach to the analysis of discrete systems with static loops is examined.

The predictor-corrector method is perhaps best introduced with respect to finding solutions of the no-input, time-invariant, one-variable system characterized by

\[
\frac{dx(t)}{dt} = f(x(t))
\]

Since some of the more accurate numerical integration methods, such as the trapezoidal method, are implicit methods (i.e., x(t+δ) is used in the computation of x(t+δ)), one may be forced to decide between computational expediency and computational accuracy. In spite of the computational difficulties associated with static loops encountered in using implicit methods, one may still decide to use them.

Most commonly used implicit methods can be represented by the following difference equation:

\[
x(t+S) = x(t) + c(f(x(t+S)), f(x(t)), \ldots, f(x(t-kS)))
\]  \quad (26)

where x(t+S), x(t), and the last k values of x are used to compute x(t+δ). Thus the function c provides the approximation to integration; it is called the corrector. If the trapezoidal method is used to solve equation (25), the result, in a form corresponding to that of equation (25), is

\[
x(t+δ) = x(t) + \frac{δ}{2}(f(x(t+δ)) + f(x(t)))
\]  \quad (27)

To simplify what follows, but with no loss of generality, consider a corrector which depends only on x(t+δ) and x(t) (e.g., that of the trapezoidal method):

\[
x(t+δ) = x(t) + c(f(x(t+δ)), f(x(t)))
\]  \quad (28)

Figure 13a shows a block diagram corresponding to the simple corrector system of equation (28). Note that a static loop exists in the corrector integrator.
The essence of the predictor-corrector method lies in the approach used to eliminate the static loop in the corrector integrator. An approximation to \( x(t+\delta) \) is used in the corrector function \( c \) rather than \( x(t+\delta) \) itself (which causes the static loop). The approximation to \( x(t+\delta), x_0(t+\delta) \), is obtained by using an explicit method requiring only \( x(t) \) and the last \( k \) values of \( x \):

\[
x_n(t+\delta) = x(t) + p(f(x(t)), f(x(t-\delta)), \ldots, f(x(t-k\delta)))
\]  

(29)

The function \( p \) used to anticipate the value of \( x(t+\delta) \) for use in the corrector is called the predictor. For example, one might use the trapezoidal integrator as a corrector (equation (27)) and the Euler integrator \( x(t+\delta) = x(t) + \delta f(x(t)) \) as a predictor. Again, for simplicity, consider a predictor which depends only on \( x(t) \) (e.g., that of the Euler method):

\[
x_n(t+\delta) = x(t) + p(f(x(t)))
\]  

(30)

Figure 13b shows a block diagram corresponding to the simple predictor-corrector system defined by equations (28) and (30). Note the predictor-corrector integrator has no static loops or static through paths.

For the case that a predictor-corrector method is to be used for a continuous system with an arbitrary input (i.e., equation (13)), the situation is again more complicated. Specifically, in real-time situations where a time-shifted system is not tolerable, one is faced with the problem of having to predict at time \( t \) the value of input \( u(t+\delta) \), as well as the value of \( x(t+\delta) \). In theory, since nothing is known about the exogenous system generating the input \( u \), such a prediction is not possible. Practically, however, if the input \( u \) is not a totally random signal and \( \delta \) is not too large, some method of extrapolation using past values of \( u \) might be used to predict \( u(t+\delta) \).

The predictor-corrector configuration is also frequently used as the basis for an iterative scheme in which the predictor provides the first approximation to \( x(t+\delta) \) and then the corrector, starting with the predictor's value, is used to iteratively generate a sequence of subsequent approximations to \( x(t+\delta) \). The final value of this sequence of approximations to \( x(t+\delta) \) is then taken to be the best approximation.

The iterative scheme for generating the sequence of approximations to \( x(t+\delta) \), for the simple predictor-corrector defined by equations (28) and (30), is as follows:

\[
\begin{align*}
x^{(1)}(t+\delta) &= x(t) + p(f(x(t))) \\
x^{(2)}(t+\delta) &= x(t) + c(f(x^{(1)}(t+\delta)), f(x(t))) \\
x^{(3)}(t+\delta) &= x(t) + c(f(x^{(2)}(t+\delta)), f(x(t))) \\
&\vdots \\
x^{(N)}(t+\delta) &= x(t) + c(f(x^{(N-1)}(t+\delta)), f(x(t))) \\
x(t+\delta) &= x(t) + c(f(x^{(N)}(t+\delta)), f(x(t)))
\end{align*}
\]  

(31)

The lower-triangular form of equations (31) is sufficient to assure that the system block diagram corresponding to this set of equations can be constructed so as to contain no static loops. However, an alternate representation, resulting in a significantly smaller block diagram, can be obtained by defining a new time set \( T'_d \) such that

\[
T'_d = \left\{ t_0, t_0 + \frac{\delta}{N}, t_0 + 2 \frac{\delta}{N}, \ldots, t_0 + \delta, t_0 + \delta + \frac{\delta}{N}, \ldots, t_1 \right\}
\]  

(32)

Note that \( T_c \subseteq T'_d \). With respect to the new time set \( T'_d \) we write the following set of difference equations:

\[
\begin{align*}
x \left( t + \frac{\delta}{N} \right) &= x(t) + \frac{c(f(x(t)) + p(f(x(t))))}{\delta} + \frac{f(x(t))}{\delta} T_d \\
x \left( t + \frac{\delta}{N} \right) &= x(t) + \frac{c(f(x(t)) + p(f(x(t))))}{\delta} + \frac{f(x(t))}{\delta} T_d
\end{align*}
\]  

(33)

where \( x_n \left( t + \frac{\delta}{N} \right) \) is the corrector estimate of \( x(t+\delta) \).
Figure 13c shows the system block diagram corresponding to equations (33). Note the subsequence \( (x(t), x(t+\delta), x(t+2\delta), \ldots, x(t)) \), corresponding to the time set \( T_d' \), obtained from the output sequence of this system defined on time set \( T_d' \), is the same sequence one would obtain from the iterative predictor-corrector scheme defined by equations (31).

It is significant that an iterative scheme used to solve the nonlinear algebraic equations resulting from the existence of static loops in a discrete dynamic system corresponds to another dynamic system with no static loops. In essence, then, the static loop of the corrector system is eliminated by inserting a \( \frac{\delta}{N} \) delay within it. To complete the iterative scheme, sufficient logic must be added to initiate the iterative scheme with the predictor estimate of \( x(t+\delta) \) for every \( t \) such that \( t \in T_d' \), and to assure that on this new time set, \( T_d' \), \( x(t) \) changes only for times \( t \) such that \( t \in T_d' \).

In Figure 13c this logic is conveniently achieved using the time-varying static element, the \( T_d \) delay, and the time-varying dynamic element, the \( T_d \) delay.

The \( T_d \) delay is a two-input \((v_1, v_2)\), single-output \((v)\) static element defined such that if the dot input \((v)\) is \( v_i \), then

$$v = \begin{cases} v_i, & t \in T_d' \\ v, & \text{otherwise} \end{cases}$$

In Figure 13c it is seen that a \( T_d \) delay is used to decide whether the predictor estimate of \( x(t+\delta) \) or the corrector's own previous estimate of \( x(t+\delta) \) is to be used in the corrector.

The \( T_d \) delay is a subsystem which is used to replace the \( \delta \) delays in the original corrector system (defined on the time set \( T_d' \)-Figure 13a). The \( T_d \) delay is, in effect, a \( \delta \) delay defined on the new time set \( T_d' \). The \( T_d \) delay with input \( v \) and output \( x \) is defined as follows:

$$x(t+\delta) = \begin{cases} v(t), & t \in T_d' \\ x(t), & t \notin T_d' \end{cases}$$

Figure 14 shows how a \( T_d \) delay can be realized using a \( T_d \) delay and two \( \frac{\delta}{N} \) delays.

Note there are three \( \frac{\delta}{N} \) delays in the predictor-corrector recursive system shown in Figure 13c (two are within the \( T_d \) delay and thus not shown explicitly in the figure). The initial value of the \( \frac{\delta}{N} \) delay within the \( T_d \) delay which provides the input to the \( T_d \) OR must be set to the same initial value given for the state variable \( x \) of the original continuous system (i.e., set to \( x_0 \); the initial values of the remaining two \( \frac{\delta}{N} \) delays can be set to any values.

CONCLUSIONS

This paper presents a system-theoretic analysis of numerical methods used in approximating solutions of ordinary differential equations. By representing ordinary differential equations by system block diagrams (i.e., interconnections of static elements and integrators), a numerical method can be viewed as a process in which the integrators of a continuous system are replaced by discrete approximations to integrators (i.e., by discrete subsystems made up of interconnections of static elements and delays). Of special concern are questions concerning existence, uniqueness, and computability by digital computer of the solutions of the resulting difference equations which characterize the discrete system approximating the original continuous system.

A study is made of the properties of the system block diagrams characterizing canonical state-variable differential and difference equations. It is noted that proper interconnections of delays and static elements (i.e., discrete interconnections with no static loops) can always be characterized by explicit difference equations. Thus, a proper interconnection represents a computationally attractive form and as such is a goal in devising a numerical method. The main result of the paper establishes that if the system block diagram corresponding to the original differential equation has no static loops and if the discrete subsystems used to replace the integrators have no static loops and no static through paths, then the resulting discrete system has no static loops (and can thus be characterized by explicit difference equations).

With the main result in hand, a system-theoretic study is conducted of several of the more commonly used numerical methods: the Euler, trapezoidal, Runge-Kutta, and predictor-corrector methods. For each of these methods the discrete subsystem used to replace the integrators of the continuous system is detailed. The limitations of using these numerical methods in real-time analysis are examined, particularly with respect to the problem often encountered in which inputs must be anticipated. The system-theoretic
implications of using the iterative predictor-corrector scheme are examined.

REFERENCES