FAST COMPUTER ANALYSIS OF NONLINEAR SYSTEMS

by

R. H. Allen

A. F. Fath

Information Sciences Report No. 38
Information Science: Laboratory
BOEING SCIENTIFIC RESEARCH LABORATORIES
September 1969
ABSTRACT

Computer aided design of nonlinear differential systems require many response calculations from the system model $\dot{x} = F(x,u)$. Many nonlinear systems, especially control systems, can be separated into a stiff linear part and a nonstiff nonlinear part. Fast numerical integration techniques taking advantage of the partitioned form are presented.
I. INTRODUCTION

Computer aided design of nonlinear differential systems requires many response calculations from the system model $\dot{x} = F(x,u)$ where $x$ is the state vector and $u$ is the control vector. Accurate models include both short and long term effects, and thus have widely separated or stiff eigenvalues. For large stiff sets of equations the analysis takes excessive execution time as most integration techniques (including the Runge Kutta rules) require that the integration time step be limited by the magnitude of the largest and possibly most uninteresting eigenvalues.

Many nonlinear differential systems, especially control systems, can be separated into a linear and a nonlinear part of the form $\dot{x} = Ax + f(x,u)$ where all the stiff eigenvalues are in the linear part represented by the matrix $A$. This partitioning is possible for most cases because the nonlinear equations usually describe the motion of the system to be controlled (e.g. airplane, missile, hydrofoil, etc.) which has time responses much slower than those of the controlling sensors, compensators and actuators. Further, most sensors, compensators and fast actuators are designed to operate in a linear or almost linear region of their response curves. Nonlinearities, such as physical limits on actuator travel, can most often be incorporated into the plant equation as input saturation. Thus the total system including feedback which is represented by
F(x,u) can be partitioned so that the large eigenvalues occur in the linear part.

In the numerical response calculation for nonlinear differential systems, there are three types of errors to consider; truncation error, roundoff error, and error due to numerical instability of the integrating rule. Roundoff error can be controlled by use of sufficient word length and programming techniques such as inner-product accumulation. Truncation error is the one step error due to the discrete approximation to the integral. This error is a function of the integration rule, the time step size, and the state of the system. For a particular integration rule and step size, bounds for the truncation error can be determined, however, actual truncation error can be significantly less depending on the state of the system. For example, if the state of the system is a stable singular point of the nonlinear equations, truncation error could be negligible. The third type of error, that arising from numerical instability of the integration rule, is often dismissed because truncation error control schemes will normally force the step size small enough so that the integration rule is stable. For stiff systems this may cause an unreasonably large number of integration steps to be required by some rules.

Numerical stability of integration rules [1,2] normally depends on the product of the maximum modulus eigenvalue (of the Jacobian) and the time step size T. For example, for a Runge Kutta fourth order rule $|\lambda_{\text{max}}| T$ must be less than 2.8. Thus even though the truncation error might be negligible for larger step sizes, $T$ must remain small to guarantee the computed solution does not grow without bound.

The purpose of this paper is to present several integration rules
which are stable over much larger regions than standard rules and thus allow the step size to be increased when the truncation error allows. For these rules, the region of numerical stability depends on the maximum modulus eigenvalue of the nonlinear part only, which due to partitioning may be orders of magnitude less than that of the composite system.

II. RESPONSE CALCULATION

Transformation

Assume a system of the form \( \dot{x} = Ax + f(x, u) \) where stiff eigenvalues are only in the linear part \( A \). Transforming the state variables, \( x = e^{At}y \), gives a new set of nonstiff differential equations, \( \dot{y} = G(y, u) \) obtained as follows,

\[
\begin{align*}
\dot{x} &= Ax + f(x, u) \\
Ae^{At}y + e^{At}\dot{y} &= Ae^{At}y + f(e^{At}y, u) \\
\dot{y} &= e^{-At}f(e^{At}y, u).
\end{align*}
\]

The dynamic eigenvalues of a nonlinear system are the eigenvalues of the instantaneous Jacobian. The eigenvalues of the original system are the eigenvalues of \( A + f(x, u) \). The eigenvalues of the transformed system are the eigenvalues of

\[
G_y = e^{-At}f_y = e^{-At}f_x e^{At} y = e^{-At}f_x(x, u) e^{At}.
\]
of the original system, $f$.

The transformed system of equations can be integrated by fast explicit integration techniques without requiring small time steps to prevent numerical instabilities due to stiff eigenvalues. Integrating $\dot{y}$ over one time step, $t_{n+1} - t_n = T$, gives

$$y_{n+1} = y_n + \int_{t_n}^{t_{n+1}} G(y,u) dt$$

(5)

Premultiplying by $e^{AT_{n+1}}$ gives the convolution integral sequential difference approximation to $x_{n+1}$.

$$x_{n+1} = e^{AT} x_n + \int_{t_n}^{t_{n+1}} e^{A(t_{n+1} - t)} f(x,u) dt$$

(6)

Numerical stability is maintained by using either an exact analytic expression for $e^{AT}$ or by using the numerically stable Padé approximations. Table 1 presents some symmetric Padé approximations to $e^{AT}$.

Numerical techniques for the analytical determination of $e^{AT}$ using similarity transformations are similar to those presented in [3].

**Table 1.** Numerically stable symmetric Padé approximations to $e^{AT}$.

<table>
<thead>
<tr>
<th>Order</th>
<th>Approximation</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>$(I - \frac{1}{2}TA)^{-1}(I + \frac{1}{2}TA)$</td>
</tr>
<tr>
<td>4</td>
<td>$(I - \frac{1}{2}TA + \frac{1}{12}T^2A^2)^{-1}(I + \frac{1}{2}TA + \frac{1}{12}T^2A^2)$</td>
</tr>
<tr>
<td>6</td>
<td>$(I - \frac{1}{2}TA + \frac{1}{10}T^2A^2 - \frac{1}{120}T^3A^3)^{-1}(I + \frac{1}{2}TA + \frac{1}{10}T^2A^2 + \frac{1}{120}T^3A^3)$</td>
</tr>
<tr>
<td>8</td>
<td>$(I - \frac{1}{2}TA + \frac{3}{28}T^2A^2 - \frac{1}{84}T^3A^3 + \frac{1}{1680}T^4A^4)^{-1}(I + \frac{1}{2}TA + \frac{3}{28}T^2A^2 + \frac{1}{84}T^3A^3 + \frac{1}{1680}T^4A^4)$</td>
</tr>
</tbody>
</table>
The above transformed system of equations can be stably inte-
grated by any explicit integration rule such as the Adams-Basforth
multistep rules. The Runge Kutta rules are used here because they
are self starting and allow easier error detection and correction.
Transformed Runge Kutta rules for order's 1, 2 and 4 are as follows:

1 \[ x_{n+1} = e^{AT}[x_n + T f(x_n)] \] (7)

2 \[ k = T e^{AT} f(x_n) \]
\[ x_{n+1} = e^{AT} x_n + \frac{1}{2} T \left[e^{AT} f(x_n) + f(e^{AT} x_n + k)\right] \] (8)

3 \[ k_1 = T f(x_n) \]
\[ k_2 = T f(e^{\frac{1}{2}AT} x_n + \frac{1}{2} e^{\frac{1}{2}AT} k_1) \]
\[ k_3 = T f(e^{\frac{1}{2}AT} x_n + \frac{1}{2} k_2) \]
\[ k_4 = T f(e^{\frac{1}{2}AT} x_n + e^{\frac{1}{2}AT} k_3) \]
\[ x_{n+1} = e^{AT}(x_n + \frac{1}{6} k_1 + e^{\frac{1}{2}AT} \frac{1}{3} (k_2 + k_3) + \frac{1}{6} k_4) \] (9)

Truncation Error
An example stiff system of equations in a later section of this
paper demonstrates how the transformed rules maintain numerical stability
for large step sizes even though the truncation errors become large.
For a linear system of equations, \( x = Cx \), the truncation error for a
p-th order Runge Kutta rule is

\[ e_n = x(t_n) - x_n = \frac{n^{p+1} \epsilon^{p+1}}{(p+1)!} x_n + \mathcal{O}(t^{p+2}) \] (10)

For a transformed Runge-Kutta rule applied to \( \dot{x} = Ax + f - f(x) - Ax \).
then the truncation error for a $p$-th order rule using a $p$-th order
Pade approximation is dominated by

$$e_n = KT^{p+1}A^{p+1}x_n$$  \hspace{1cm} (11)

since $||A|| >> ||B||$. If the approximation to $e^{AT}$ is of order
greater than $p$ then

$$e_n = KT^{p+1}A^pBx_n$$  \hspace{1cm} (12)

For $T$ much greater than $||A||^{-1}$ the error gets large so that
significant increases in $T$ cannot be made.

For example if $A$ has eigenvalues like $-1000$ and $f_x$ has
eigenvalues like $-1$ then a standard Runge Kutta rule will require a
time step of about .001 to maintain numerical stability. The trans-
formed method will be numerically stable for any step size up to 1,
but the truncation error can be like $T^{p+1}10^{3p/(p+1)}$. To control
this truncation error $T$ might still have to be kept small.

**Convolution Integral Partitioning**

When the stiff system of equations can be further partitioned
so that part of the system has no stiff part as is shown in figure 1
and equation 13, then low error convolution integral approximations
can be made.
Numerically stable, low error integration rules can be obtained by applying first and second order explicit one step end point approximations to the convolution integral representation of $x_2$.

$$x_{2}^{n+1} = e^{A_2 T} x_2^n + \int_{t_n}^{t_{n+1}} e^{A_2 (t_{n+1} - t)} f_2(t) dt$$  \hspace{1cm} (14)

This approximation to $x_2$ is coupled with compatible Runge-Kutta rules to approximate $x_1$.

A first order approximation to the convolution integral can be obtained by expanding $f_2(x)$ about $t_n$ and then integrating the exponential term exactly.

$$f_2(x(t)) = f_2^n + 0(T)$$  \hspace{1cm} (15)

$$\int_{t_n}^{t_{n+1}} e^{A_2 (t_{n+1} - t)} dt = A_2^{-1} (e^{A_2 T} - 1) = I_1$$  \hspace{1cm} (16)
The integral $I_1$ is well defined even if $A_2$ is not of full rank and can be calculated, as before, using stable Padé approximations or evaluating exact analytical expressions. Using equations (15) and (16), equation (14) becomes

$$x_{2}^{n+1} = e^{A_2 T} x_2^n + I_1 f_2^n + 0(T^2). \quad (17)$$

From a first order rule

$$x_1^{n+1} = x_1^n + T f_1^n \quad (18)$$

Similarly to get a second order rule

$$f_2(x(t)) = f_2^n + \int_{t_n}^{t_{n+1}} e^{A_2(t_n+1-t)} [f_2(t) - f_2(t_n)] dt = A_2^{-1} (T_1 - T) = I_2 \quad (19)$$

A first order approximation to $f_2$ is needed so evaluating $f_2$ at $x_1^{n+1}$ and $x_2^{n+1}$ from (17) and (18) gives a first order approximation to $f_2^{n+1}$.

$$f_2^n = \frac{f_2^{n+1} - f_2^n}{T} \quad (20)$$

Using equations (19), (20), and (21) equation (14) becomes

$$x_2^{n+1} = e^{A_2 T} x_2^n + I_1 f_2^n + I_2 [f_2^{n+1} - f_2^n]/T + 0(T^3) \quad (22)$$

From a second order Runge Kutta rule

$$x_1^{n+1} = x_1^n + \frac{1}{2} T (f_1^n + f_1^{n+1}) \quad (23)$$
where again $f_1^{n+1}$ is evaluated at the first order points in equations (17) and (18). Equations (22) and (23) define a
partitioned second order rule.

Under the special circumstances when $f_2$ is a function of
only $x_1$ and the Jacobian of $f_2$ is readily available (such as
when $f_2$ is linear) then slightly better accuracy can be obtained by
using the following approximation to $f_2$:

$$f_2(x_1(t)) = f_2^n + \frac{f_2^n}{2} (t-t_n) + O(T^2)$$

(24)

Now equation (22) takes the form

$$\begin{align*}
  x_2^{n+1} &= e^{A_2 T_n} x_2^n + I_{12} f_1^n + I_{22} f_2^n x_1^n \\
  \end{align*}$$

(25)

Using (18) and (25) $f_1^{n+1}$ can be approximated to be used to define $x_1^{n+1}$ in equation (23). Equations (22) and (23) define a partitioned
second order rule when the Jacobian of $f_2$ is available.

The truncation errors for these partitioned rules are on the
order of the truncation errors for a system of equations with eigen-
values on the order of the eigenvalues of the nonlinear parts. For
example if $A_2$ has eigenvalues like -1000 and $f_1$ and $f_2$ have
eigenvalues like -1 then the truncation error will be like $T^{p+1}/(p+1)!$. To get 3 place accuracy with a first order rule $T$ should be about .04 and with a second order rule $T$ should be about .1.

III. EXAMPLE

In order to illustrate the application of these solution

In order to illustrate the application of these solution techniques, consider a simple pendulum problem in which the angular
rate is measured and used as a signal to a torque motor to provide damping. The nonlinear equations describing the motion of the pendulum are

\[ M \ddot{\theta}(t) = -Mg \sin \theta(t) + T(t)/L \]  

(26)

where \( M \) is the mass, \( L \) is the length, and \( T(t) \) is the torque applied by the torque motor. Let the dynamics of the angle sensor be given by a first order lag with a time constant \( 1/\tau_s \). Likewise let the torque motor be represented by a first order lag with time constant \( 1/\tau_m \). To obtain the state equations, let \( x_1 \) be \( \theta \), \( x_2 \) be \( \dot{\theta} \), \( x_3 \) be the output of the angular rate sensor, and let \( x_4 \) be the torque. The equations of motion are then

\[
\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2 \\
\dot{x}_3 \\
\dot{x}_4
\end{bmatrix} =
\begin{bmatrix}
x_2 \\
-g/L \sin x_1 - x_4/ML^2 \\
K_s \tau_s x_2 - \tau_s x_3 \\
K_m \tau_m x_3 - \tau_m x_4
\end{bmatrix} 
\]  

(27)

For the purpose of this example, let \( g/L = 1.0 \), \( 1/ML^2 = 0.5 \), \( \tau_s = 1000.0 \), \( K_s = 1.0 \), \( \tau_m = 50.0 \), and \( K_m = 3.0 \).

Now factor the system into the form

\[
\dot{x} = \begin{bmatrix}
0 & 0 \\
-1000 & 0 \\
150 & -50
\end{bmatrix} x + \begin{bmatrix}
x_2 \\
\sin x_1 \\
-0.5x_4
\end{bmatrix} 
\]  

(28)
Note that the Jacobian of $F$ for small $x_1$ is

\[
\frac{\partial F}{\partial x} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & -0.5 \\ 0 & 1000 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}
\] (29)

which has eigenvalues of $0.0, 0.0, \pm 1.0$. The eigenvalues of the composite system for small $x_1$ are $-1000.08, -48.371, -0.77484 \pm 0.65818$.

For the example system of equations considered here some indications of the acceptability of the suggested integration rules can be made. Figure 2 is a plot of discretization error curves for several rules. The log of the error versus the log of the integration time step has a slope which is equal to the order of the integration rule. Table 2 compares the number of significant figures of accuracy obtained for the integrating rules at several time steps. Also computer execution times are given.

A fourth order Runge Kutta rule had high accuracy for small step sizes, but it went unstable for acceptable step sizes. An order two Runge Kutta rule went unstable for smaller step sizes. The transformed Runge Kutta rules remained stable for large step sizes but the accuracy was poor. The partitioned methods gave acceptable accuracy for reasonable step sizes. Using the Jacobian technique gave about 1 place better accuracy than the time derivative approximation technique. Using a Padé approximation to $e^{AT}$ instead of an exact expression did not degrade the accuracy much, but in some problems this may cause loss of accuracy because large negative Padé exponents approach $-1$, instead of $0$. The partitioned 2nd order rule allows integration to acceptable
accuracy about 200 times faster than a standard 4th order Runge Kutta rule for this example.

Table 2. Number of significant figures and timings for example.

<table>
<thead>
<tr>
<th>Integration rule/time step</th>
<th>$10^{-3}$</th>
<th>$10^{-2}$</th>
<th>$10^{-1}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Order 4 Runge Kutta</td>
<td>12</td>
<td>unstable</td>
<td>unstable</td>
</tr>
<tr>
<td>Order 2 Runge Kutta</td>
<td>unstable</td>
<td>unstable</td>
<td>unstable</td>
</tr>
<tr>
<td>Order 2 transformed</td>
<td>3</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Order 2 partitioned</td>
<td>6</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>Order 2 partitioned (with Pade approximation)</td>
<td>6</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>Order 2 partitioned (with Jacobian)</td>
<td>7</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>Execution time in seconds for all order 2 rules (360/44)</td>
<td>30</td>
<td>3</td>
<td>.3</td>
</tr>
</tbody>
</table>
Figure 2. INTEGRATION ERROR CURVES

Legend
- (Transformed)
- (Partitioned)
- (Partitioned (with Jacobian))
- Runge Kutta

Integration Rule Error for Example

Time Step Size
IV. CONCLUSIONS

In many simulations where one or .1 percent accuracy is sufficient, the larger step sizes allowed by the low order stable integration rules reduces run times appreciably. Control systems for nonlinear plants which use position and rate feedback with compensation usually allow partitioning as indicated in Eq. 13 with \( f_2 \) being a function of \( x_1 \) and \( u \) only. Experience has shown the second order rule to be ideally suited for this problem. Where acceleration feedback is used, \( f_2 \) now becomes a function of \( x_2 \) in addition to \( x_1 \) and \( u \) which in some cases reduces the effectiveness of the integration rule.

The rules presented here show advantages over other rules such as standard Runge Kutta only when the truncation errors of the primary variables are small enough to warrant larger step sizes. Where the problem is truncation error limited, the higher order rules offer advantages in error control. It should be noted that in many simulations, however, errors due to numerical instabilities are the limiting factor. For these systems, stable integration rules can significantly reduce computation times.

One obvious example where these rules are particularly effective is in determining steady state operating conditions of nonlinear systems. Here, large errors in the transient solution can be tolerated making stable integration rules with large step sizes quite desirable.
REFERENCES

