# UNIVERSITY OF HOUSTON



PROJECT THEMIS
Information Processing Systems

ONR Contract: N00014-68-A-0151



## Identification of Systems

by
Bart Childs

RE 2-68
August, 1968



37

# IDENTIFICATION OF SYSTEMS

by

BART CHILDS
Associate Professor

Department of Mechanical Engineering
Cullen College of Engineering
University of Houston

RE 2 - 68

August, 1968

# Identification of Systems

by

Bart Childs

## ABSTRACT

This paper is a tutorial discussion of identification problems. The basic principles of one method of identification are discussed. The method of quasilinearization has been developed by Richard Bellman and Robert Kalaba and others. This method is discussed in some detail with emphasis on programming strategies. These strategies are intended to improve the utility of a general quasilinearization program. Specific strategies are concerned with expansion of convergence space, superposition of particular solutions, and convergence criteria.

## Acknowledgements

This work is based upon the works of Bellman and Kalaba. The references are omitted to add to the oneness of the thoughts presented here. I would like to thank Dr. Kalaba for many enlightning discussions on mathematics and identification. Further developments and details will be reported in future publications and an excellent reference is Dynamic Programming and Modern Control Theory by Bellman and Kalaba. This work was supported by NASA Grant NGR 44-005-060 and Project THEMIS, ONR Contract N00014-68-A-0151.

TABLE OF CONTENTS

## Introduction

System identification is that branch of the physical sciences in which we answer either of the two closely related questions:

(1) Given enough data to completely define an existing system, what is the system (generally, what are the parameters of the differential equation(s) that will yield the observed solution)?

Or

(2) Given a desired response of an embryonic system, what are the proper system parameters that will satisfy the desired response in a best fit sense and simultaneously minimize a cost function that may take on a very general form?

We will restrict ourselves to problems governed by differential equations. Those problems governed by algebraic equations are satisfactorily covered in many elementary texts.

# I

## The Simplest Identification Problem

From the introduction, it is obvious that the simplest nontrivial identification problem will be governed by a linear first order homogeneous differential equation.

We will then assume

$$\dot{y} = ay \tag{1}$$

subject to

$$\begin{aligned} y(t_1) &= y_1 \qquad t>0 \\ y(t_2) &= y_2 \qquad t_2>t_1\geq 0 \end{aligned} \tag{2}$$

The solution of the differential equation (2) is

$$y = y_o e^{at} \tag{3}$$

We now have two unknowns, namely $y_o$ and a. To determine these we write the following set of nonlinear algebraic equations

$$\begin{aligned} y_1 &= y_o e^{at_1} \\ y_2 &= y_o e^{at_2} \end{aligned} \tag{4}$$

Since this problem is exceedingly simple, we can solve for the unknowns

$$a = -\frac{\ln(y_1/y_2)}{(t_2-t_1)}$$

$$y_o = y_1 \, e^{\left\{ \frac{\ln(y_1/y_2)}{-1+t_2/t_1} \right\}} \tag{5}$$

The conclusion can now be reached that, even in the simplest form, identification problems are basically nonlinear. The nonlinearity is there in spite of the apparent linearity of the original differential equations.

We now take a different view, that is, the original differential equation is nonlinear. If we rationalize that "a" is a function of "t" and is an unknown function of "t" which is governed by a simultaneous differential euqation, then we get

$$\begin{aligned} \dot{y} &= ay \\ \dot{a} &= o \end{aligned} \tag{6}$$

Actually, we have now cast our system into its true perspective. We have two data points and two simultaneous first order nonlinear differential equations. That is what we will usually expect.

The differential equation for "a" obviously states that it doesn't change with respect to "t".

## II

## Another Simple Identification Problem

Let's take a different look at the simple harmonic motion problem. The governing differential equation is

$$\ddot{x} + \xi x = 0 \qquad t \geq 0 \tag{7}$$

The necessary boundary conditions will be taken as

$$\begin{aligned} x(t_1) &= x_1 \\ x(t_2) &= x_2 \qquad t_3 > t_2 > t_1 \geq 0 \\ x(t_3) &= x_3 \end{aligned} \tag{8}$$

The ordering for $t_i$ will be assumed hereafter and is arbitrary. Denoting $\sqrt{\xi} = \omega$ we are quickly led to

$$\begin{aligned} x_1 &= x_o \cos(\omega t_1) + \frac{\dot{x}_o}{\omega} \sin(\omega t_1) \\ x_2 &= x_o \cos(\omega t_2) + \frac{\dot{x}_o}{\omega} \sin(\omega t_2) \\ x_3 &= x_o \cos(\omega t_3) + \frac{\dot{x}}{\omega} \sin(\omega t_3) \end{aligned} \tag{9}$$

These three nonlinear algebraic equations can be solved by the use of the Newton Raphson algorithm or some other suitable algorithm. The three unknowns are obviously $x_o$, $\dot{x}_o$, and $\xi$. (Or in the form shown ($\sqrt{\xi} = \omega$)).

Another set of boundary conditions is of course possible. It is possible for boundary conditions to be on the derivative(s) of the functions. Taking the boundary conditions to be

$$\begin{aligned} \dot{x}(t_1) &= \dot{x}_1 \\ \dot{x}(t_2) &= \dot{x}_2 \\ x(t_3) &= x_3 \end{aligned} \tag{10}$$

Then the necessary algebraic equations are

$$\begin{aligned} \dot{x}_1 &= -x_o\omega \sin(\omega t_1) + \dot{x}_o \cos(\omega t_1) \\ \dot{x}_2 &= -x_o\omega \sin(\omega t_2) + \dot{x}_o \cos(\omega t_2) \\ x_3 &= x_o \cos(\omega t_3) + \frac{\dot{x}_o}{\omega} \sin(\omega t_3) \end{aligned} \tag{11}$$

The degree of difficulty of both these problems is essentially the same. The solution of either set of nonlinear algebraic equations requires about the same number of calculations, strategies, and programming skill.

In a manner similar to the earlier one, we can also formulate this problem as a set of nonlinear differential equations

$$\begin{aligned} \ddot{x} + \xi x &= 0 \\ \dot{\xi} &= 0 \end{aligned} \tag{12}$$

We will find it convenient later to always write the set of differential equations as a set of first order equations. Thus, with the definition stated in the first of the following equations we have

$$\begin{aligned} \dot{x} &= z \\ \dot{z} &= -x\xi \\ \dot{\xi} &= 0 \end{aligned} \tag{13}$$

If we attempt a slight variation of the preceeding problem and add a damping term, the problem will become seemingly formidable in an analytical identification sense. The governing equation is

$$\ddot{x} + \mu\dot{x} + \xi x = 0 \tag{14}$$

As you might guess, if the damping and frequency terms are unknown, we automatically write

$$\begin{aligned} \dot{x} &= z \\ \dot{z} &= -\mu z - \xi x \\ \dot{\xi} &= 0 \\ \dot{\mu} &= 0 \end{aligned} \tag{15}$$

The next four topics of discussion will be background to aid in solving the nonlinear multipoint boundary value problems we have been formulating.

## III

## The Newton Raphson Method

We are now going to discuss a generalization of Newton's method of tangents and later we will generalize this a bit more.

First, let's review the method of tangents and point out the properties of the generalized methods that are desirable and difficult to prove.

Consider the transcendental function cos(x) and assume that we wish to find its first zero, and that from some obscure "a priori" knowledge we find that the solution is near one radian. We expand our equation

$$f(x) = \cos(x) = 0 \tag{16}$$

in a Taylor series about the approximate root $x_a$

$$f(x) = f(x_a) + f'(x_a)\frac{(x-x_a)}{1!} + f''(x_a)\frac{(x-x_a)^2}{2!} + \ldots \tag{17}$$

Neglecting the terms of order $(x-x_a)^2$ and higher we get the approximation

$$f(x_a) + f'(x_a)(x-x_a) \tilde{} 0 \tag{18}$$

Then

$$\begin{aligned} x-x_a &= -f(x_a)/f'(x_a) \\ x &= x_a + \Delta x_a \\ \Delta x_a &= -f(x_a)/f'(x_a) \end{aligned} \tag{19}$$

The indicated algorithm is generally written as

$$x_{n+1} = x_n - f(x_n)/f'(x_n) \tag{20}$$

The last equation states what is called the Newton Raphson algorithm for a one dimensional problem.

The algorithm for the zero of the cosine function indicated does reduce to

$$x_{n+1} = x_n - (-\cot(x_n)) = x_n + \cot(x_n) \qquad (21)$$

For the initial condition mentioned, Table I shows the convergence on the true answer which we already knew to be $\pi/2$.

TABLE I

Newton Raphson Data of Eq. (21)

| n | Value of $x_n$ |
|---|---|
| 0 | 1.0000 |
| 1 | 1.6421 |
| 2 | 1.5687 |
| 3 | 1.5698 |
| 4 | 1.5706 |
| ... | ... |
| $\infty$ | 1.5708 |

The table was constructed by interpolating the values given in the CRC Handbook.

Notice that in the early iterations that there was an oscillation about the true answer. However, as the final solution was approached, the convergence was monotone. R. Kalaba has proven that often the method gives quadratic and monotone convergence in the limiting iterations. An approximate interpretation of quadratic convergence is that at each iteration

the number of significant figures is doubled. Of course, that implies that an extremely high number of significant figures must be used in the calculations. Because of the "finiteness" of the tables used, the quadratic convergence property is not illustrated by the above table. It will also be shown by the following refinement that the monotone behavior is accidental.

The results of solving the same problem, but having it done on a digital computer using 16+ significant figures in each calculation are shown in Table II. Notice that the quadratic convergence is illustrated by this data. An obvious question is

TABLE II

Improved Newton Raphson Data

| n | Value of $x_n$ Eq. (21) |
|---|---|
| 0 | 1.0000000000000000 |
| 1 | 1.6420926159343 31 |
| 2 | 1.5706752771612 51 |
| 3 | 1.5707963267954 88 |
| 4 | 1.5707963267948 97 |
| 5 | 1.5707963267948 97 |

what is the accuracy of the sine and cosine subroutines? The final answer is accurate to sixteen significant decimal digits. Also notice that there is an oscillation about the answer again.

An interesting variation is to apply the algorithm to a problem involving complex variables. In approaching the solution of the biharmonic equation where two dimensional Cartesian co-ordinates with at least one of the coordinates having finite limits via Laplace or Fourier transforms, it is often necessary to

know the zeroes of the transcendental.

$$\sin(Z) + Z = 0 \tag{22}$$

The Newton Raphson algorithm for this problem becomes

$$Z_{n+1} = Z_n - (\sin(Z_n) + Z_n)/(1 + \cos(Z_n)) \tag{23}$$

Obviously the problem now is arithmetical except for the question of initial guesses. Let's assume that a little bird flew past and said, "Awk!, the first non zero root is near $4 + i2$ and each successive root will be approximately $2\pi$ further out the real axis." The name of that little bird is experience.

Table III is a list of the values of Z that will be generated in obtaining the first five non zero roots with the aid of the little bird's information. The improvement of an answer is stopped whenever the modulus of the last correction is less than $10^{-5}$.

It is obvious that the convergence is rapid for the above problem. The interpretation of "doubling the number of significant figures" with each iteration now raises a question. Which significant figures are doubled, if any? A rule that sometimes applies is, "the largest element's significant figures are doubled".

The example was included for tutorial purposes only. The most efficient method of solving the above problem includes asymptotic studies of Eq. (22).

Consider the problem of solving a set of nonlinear algebraic equations in "N" real unknowns. We will write this as the vector equation

$$\vec{R}(\vec{X}) = \vec{0} \tag{24}$$

where the right hand side is the null vector.

TABLE III

Complex Newton Raphson Data (Eq. (22)

| Root Index | n | Real ($Z_n$) | Imag. ($Z_n$) |
|---|---|---|---|
| 1 | 0 | 4. | 2. |
| | 1 | 4.2793583E 00 | 2.2714714 |
| | 2 | 4.2143169E 00 | 2.2488951 |
| | 3 | 4.2123884E 00 | 2.2507277 |
| | 4 | 4.2123922E 00 | 2.2507286 |
| 2 | 0 | 1.0492392E 01 | 2.2507286 |
| | 1 | 1.0938640E 01 | 3.6368255 |
| | 2 | 1.0801822E 01 | 3.2135148 |
| | 3 | 1.0721907E 01 | 3.1062368 |
| | 4 | 1.0712570E 01 | 3.1031115 |
| | 5 | 1.0712537E 01 | 3.1031487 |
| 3 | 0 | 1.6992537E 01 | 3.1031487 |
| | 1 | 1.7109956E 01 | 3.6695510 |
| | 2 | 1.7077178E 01 | 3.5574931 |
| | 3 | 1.7073389E 01 | 3.5511021 |
| | 4 | 1.7073365E 01 | 3.5510873 |
| 4 | 0 | 2.3353365E 01 | 3.5510873 |
| | 1 | 2.3411799E 01 | 3.9114766 |
| | 2 | 2.3398998E 01 | 3.8601203 |
| | 3 | 2.3398356E 01 | 3.8588097 |
| | 4 | 2.3398355E 01 | 3.8588090 |
| 5 | 0 | 2.9678355E 01 | 3.8588090 |
| | 1 | 2.9714814E 01 | 4.1235346 |
| | 2 | 2.9708304E 01 | 4.0941315 |
| | 3 | 2.9708120E 01 | 4.0937050 |
| | 4 | 2.9708120E 01 | 4.0937049 |

The Newton Raphson approach is to expand the function about some approximate root $\vec{X}_a$.

Then

$$\vec{R}(\vec{X}_a) + \frac{\partial \vec{R}}{\partial \vec{X}} (\vec{X}_a) \{\vec{X} - \vec{X}_a\} + \ldots = \vec{R}(\vec{X}) \stackrel{?}{=} \vec{0} \qquad (25)$$

Hopefully assuming the questioned equality and neglecting the indicated higher order terms yields

$$\vec{X} = \vec{X}_a - \{\frac{\partial \vec{R}(X_a)}{\partial \vec{X}}\}^{-1} \vec{R}(\vec{X}_a) \qquad (26)$$

or in terms of the generally expected subscripts the algorithm is

$$\vec{X}_{n+1} = \vec{X}_n - \{\frac{\partial \vec{R}}{\partial \vec{X}}\}_n^{-1} \vec{R}_n \qquad (27)$$

The derivative quantity is a square matrix of order "N" and we hope also of rank "N". In general it is and we seldom need consider the case where it might be singular in the neighborhood of the solution X.

Let's assume the problem is to solve the equations

$$\begin{aligned} x_1 + x_2 + x_3 &= 3 \\ x_1^2 + 3x_2 + 4x_3 &= 8 \\ x_2^2 - x_3 &= 0 \end{aligned} \qquad (28)$$

Then

$$\begin{aligned} r_1 &= x_1 + x_2 + x_3 - 3 \\ r_2 &= x_1^2 + 3x_2 + 4x_3 - 8 \\ r_3 &= x_2^2 - x_3 \end{aligned} \qquad (29)$$

The square matrix or as we will call it, the "Jacobian matrix" for this problem is

$$J_n = \frac{\partial \vec{R}}{\partial \vec{X}}\bigg|_n = \begin{bmatrix} 1 & 1 & 1 \\ 2x_1 & 3 & 4 \\ 0 & 2x_2 & -1 \end{bmatrix}_n \tag{30}$$

Assuming the null vector for the first approximation $\vec{X}_o$,

$$\vec{X}_1 = \vec{X}_o + (-J_o)^{-1} \vec{R}_o \tag{31}$$

Substituting

$$\vec{X}_1 = \vec{0} + \begin{pmatrix} -1 & -1 & -1 \\ 0 & -3 & -4 \\ 0 & 0 & +1 \end{pmatrix}^{-1} \begin{bmatrix} -3 \\ -8 \\ 0 \end{bmatrix} \tag{32}$$

The successive solution vectors, as calculated on a 7094, are shown in Table IV.

TABLE IV

Elements of Solution Vector at Each Iteration

| n | $x_1$ | $x_2$ | $x_3$ |
|---|---|---|---|
| 0 | 0. | 0. | 0. |
| 1 | .33333333 | 2.6666667 | 1.5E-8 |
| 2 | .72191529 | 1.4825046 | .79558007 |
| 3 | .94095068 | 1.0736089 | .98544037 |
| 4 | .99710697 | 1.0026408 | 1.0002522 |
| 5 | .99999356 | 1.0000044 | 1.0000020 |
| 6 | .99999999 | 1.0000000 | 1.0000000 |

The elements of the residue vector, $\vec{R}$, are shown in Table V.

TABLE V

Elements of Residue Vector at Each Iteration

| n | $r_1$ | $r_2$ | $r_3$ |
|---|---|---|---|
| 0 | -3. | -8. | 0. |
| 1 | 1.49E-8 | 1.11E-1 | 7.11 |
| 2 | 1.49E-8 | 1.51E-1 | 1.40 |
| 3 | 7.45E-9 | 4.80E-2 | 1.67E-1 |
| 4 | 0. | 3.15E-3 | 5.04E-3 |
| 5 | 4.47E-8 | 8.26E-6 | 6.93E-6 |
| 6 | 0. | 0. | 0. |

The solution shown above might cause questions as to whether the convergence is quadratic and monotone. The quadratic question is needless because the convergence is obviously fast. The monotone question is meaningful at this time. Monotone convergence in the one dimensional case meant that the solution was always approached from the same side. In other words $\Delta\vec{X}$ was always of the same sign. The generalization of this is that $\Delta\vec{X}_n^T\Delta\vec{X}_n$-1 is positive when monotone and negative when oscillating. In the limiting iterations, the above was indeed monotone.

Referring to the not quite so simple identification problem, we will write (see Eq. (9))

$$\vec{R}(\vec{x}) = \begin{bmatrix} x_1 \cos(x_3 \cdot 0) + \frac{x_2}{x_3} \sin(x_3 \cdot 0) - 1.00000 \\ x_1 \cos(x_3 \cdot 0.5) + \frac{x_2}{x_3} \sin(x_3 \cdot 0.5) - 1.35701 \\ x_1 \cos(x_3 \cdot 1.) + \frac{x_2}{x_3} \sin(x_3 \cdot 1.) - 1.38177 \end{bmatrix} \quad (33)$$

The above equations are for the harmonic oscillator having unit initial conditions and a unit circular frequency ($x_1 = x_o = 1$, $x_2 = \dot{x}_o = 1$, $x_3 = \omega = \sqrt{\xi} = 1$) solution for t = 0, 0.5, and 1.0 are then used.

Table VI is a list of the vector $\vec{X}$ starting from an initial guess in which each element is 50% the final solution.

TABLE VI

Newton Raphson Data

| n | $x_1$ | $x_2$ | $x_3$ |
|---|---|---|---|
| 0 | 0.5 | 0.5 | 0.5 |
| 1 | 1.00000 | 0.98410 | 1.86895 |
| 2 | 1.00000 | 0.74143 | 0.92949 |
| 3 | 1.00000 | 1.00173 | 1.00862 |
| 4 | 1.00000 | 0.99999 | 1.00002 |
| 5 | 1.00000 | 1.00000 | 1.00000 |

With nonlinear equations such as these, there is always the possibility of finding a solution that isn't desired. Using an initial guess vector of (0, -2,2), the same program that converged on the nice answers above converged upon the answer (1, -36.7,-36.7).

Let's now consider the solution of a function to be optimized subject to a set of equality constraints. For the simple first example we want to consider only functions whose constraints are linear and the function to be optimized is of a quadratic norm type. Assuming a two dimensional problem, we write typically

$$\begin{aligned} x_1 + \alpha x_2 &= \beta \\ (\gamma x_2 + \delta x_2 - \theta)^2 &= \text{minimum} \end{aligned} \qquad (34)$$

By differentiating the function to be minimized with respect to $x_1$, we get

$$2\gamma(\gamma x_1 + \delta x_2 - \theta) = 0 \tag{35}$$

Thus

$$\begin{aligned} x_1 + \alpha x_2 &= \beta \\ \gamma x_1 + \delta x_2 &= + \theta \end{aligned} \quad \text{or} \quad \begin{bmatrix} 1 & \alpha \\ \gamma & \delta \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \beta \\ \theta \end{bmatrix} \tag{36}$$

$$x_1 = \frac{\beta\delta - \alpha\theta}{\delta - \alpha\gamma} \qquad x_2 = \frac{\theta - \beta\gamma}{\delta - \alpha\gamma}$$

which is nice as long as $\delta \neq \alpha\gamma$ which is the usual case. Before this simple problem lulls us into trouble, let's point out that the equation resulting from differentiating with respect to $x_2$ would have been

$$2\delta(\gamma x_1 + \delta x_2 - \theta) = 0 \tag{37}$$

which is equivalent to the one we had.

Considering a problem in three or more dimensions we need to be careful. Assume a constraint

$$\sum_i^N a_{1i} x_i = c_1 \tag{38}$$

If we assume a cost function to be minimized, we can take it to be of the form

$$\sum_{j=2}^{M} \left(\sum_i^N a_{ji} x_i - c_j\right)^2 = \text{cost} \tag{39}$$

where $M > N - 1$.

We supposedly can differentiate the cost function with respect to

each element of $\vec{X}$ and obtain N - 1 or N linearly independent equations, depending on M. Because there can be only N linearly independent equations, one might be led to believe that any N - 1 of the equations will give the same solution when coupled with the constraint equation given above. That isn't so because the right-hand-sides can be picked at random. Regardless, the best solution will probably be obtained by an iterative method.

For our purposes we can assume that the above least squares type approximation is adequate. The fact that each of the equations from minimizing the error or cost term involves all the data points is responsible for this. It should be expected that any iterative scheme would need this type approximation for a starting approximation anyway.

## IV

## Function Space

This section will be very incomplete. It is included for emphasis of a point that would require great labor to explain with any significant rigor. Function space is, uh, a very nebulous abstraction when an attempt is made to visualize it in the Cartesian frame of mind we are accustomed to. Some discussion of function space is needed because we are going to differentiate expressions with respect to functions of variables and not with respect to variables. Of course, we can always rationalize that a variable is simply a special case of a function. The point is though, we are to be differentiating with respect to continuous dependent functions of an independent variable. This

function space might be thought to have independent variables that are the functions of the true independent variable. Another independent variable might be an iteration index where we are continuously updating approximations to these functions. Think about it??? A discussion of function space is in Lanczos' text entitled Linear Differential Operators.

Some appreciation of function space is necessary for complete confidence in performing Newton-Raphson-Kantorovich expansions in function space. We'll call them NRK expansions.

We have written some vector differential equations earlier. Now, we're going to differentiate these vector equations with respect to some of the functions within. The equations are like

$$\dot{\vec{y}} = \vec{f}\ (\vec{y},t) \tag{40}$$

The "independent variables" in this function space, that we're about to expand this equation in, are the elements of the vectors $\vec{y}$ and $\dot{\vec{y}}$. Now, both sides of the equation should be expanded with respect to the totality of $\vec{y}$ and $\dot{\vec{y}}$. However!, the LHS is a function of $\dot{\vec{y}}$ only and the RHS is a function of $\vec{y}$ only. This is one reason why we always write the equations in this first order form. Let's diverge for a moment.

Recall that:

The derivative of a vector with respect to another vector is a matrix or maybe it's a second order tensor. If both the differentiated and differentiating vector are the same, the matrix is the unit or identity matrix. If they are not the same, the resulting second order tensor (?) is called the Jacobian Matrix. The second derivative of a vector with respect to another vector gives a result that has three subscripts.

Anyway

$$\dot{\vec{y}} = \vec{f}\,(\vec{y},t) \tag{40}$$

expanding about the nth iteration where $\dot{\vec{y}}_n$ and $\vec{y}_n$ are known

$$\dot{\vec{y}}_n + \left.\frac{\partial \dot{\vec{y}}}{\partial \vec{y}}\right]_n (\dot{\vec{y}}_{n+1} - \dot{\vec{y}}_n) + \ldots$$

$$= \vec{f}\,(\vec{y}_n,t) + \left.\frac{\partial \vec{f}}{\partial \vec{y}}\right]_n (\vec{y}_{n+1} - \vec{y}_n) + \ldots \tag{41}$$

We have neglected all the higher order terms, the first of which on the right-hand side cannot be written like

$$\left.\frac{\partial^2 \vec{f}}{\partial \vec{y}^2}\right]_n (\vec{y}_{n+1} - \vec{y}_n)^2 \tag{42}$$

but must be written as

$$\left(\left.\frac{\partial^2 \vec{f}}{\partial \vec{y}^2}\right]_n (\vec{y}_{n+1} - \vec{y}_n)\right) (\vec{y}_{n+1} - \vec{y}_n) \tag{43}$$

where the order in which the matrix multiplication or tensor contraction must be performed is shown by the parentheses.

Recognizing the unit matrix or identity operator resulting from differentiating $\dot{\vec{y}}$ with respect to itself and making a slight introduction of $J_n$ for the Jacobian matrix on the RHS, we easily write

$$\dot{\vec{y}}_{n+1} = J_n \vec{y}_{n+1} + \vec{f}_n - J_n \vec{y}_n \tag{44}$$

which with an introduction of obvious nomenclature can be written (some would say, "simplifies to")

$$\dot{\vec{y}}_{n+1} = J_n \vec{y}_{n+1} + \vec{F}_n \tag{45}$$

Henceforth, if it is implied that NRK's are relevant, we automatically transform a nasty nonlinear equation

$$\dot{\vec{y}} = \vec{f}\ (y,t) \tag{40}$$

into the simple

$$\dot{\vec{y}}_{n+1} = J_n \vec{y}_{n+1} + \vec{F}_n \tag{45}$$

which is linear with known variable coefficients and a known forcing function $\vec{F}_n$. An obvious problem dealing with the first assumption for $\vec{y}_o$ will be dealt with later.

## V

## Numerical Integration of Initial Value Problems

The wheel has suffered through more re-inventions in the field of numerical integration than any sane person would like to count. The necessary information presented here is covered in great detail, often with so-called "rigor", in no less than one-hundred-thirty-seven and one-half good books. The reason it appears here is for the sake of sameness in the names applied to the methods used. Occasionally there is somebody from a "culturally and socially deprived area" who may not know about these techniques.

I assume that we all know about the forward, backward, and central difference approximations to first derivatives. We have been writing our differential equations in a first order form

and will continue to do so. Because of this, we only need difference approximations to first derivatives. It is common knowledge that the error term for forward and backward difference approximations is of order h while the order is $h^2$ for central difference expressions. The following characteristics of these basic approximations should be recognized:

(1) Forward difference methods O(h) are self-starting and yield explicit formulae.

(2) Central difference methods have to be started by other means but do yield explicit formulae.

(3) Backward difference methods do not yield explicit formulae for nonlinear differential equations.

The simple forward difference methods have the most advantages but they also have error problems, relatively. The recurrence formulae for a problem like

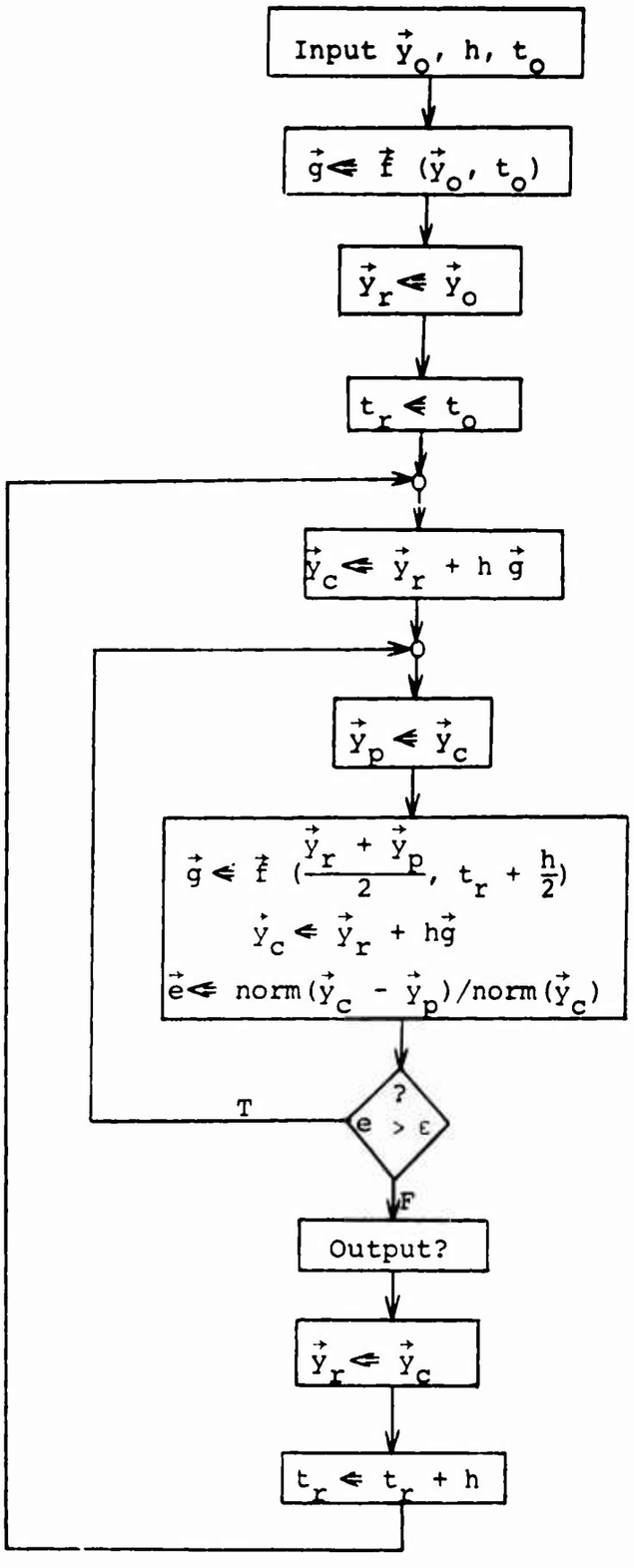
$$\dot{\vec{y}} = \vec{f}\ (\vec{y},t) \tag{40}$$

is

$$\vec{y}_{j+1} \leftarrow \vec{y}_j + h\vec{f}\ (\vec{y}_j,t_j) \tag{46}$$

From the known initial conditions of this initial value problem, $\vec{y}_o$ is known and then all other $\vec{y}$'s can be calculated. A simple iterative form that will retain all desirable characteristics of the forward difference method and gain in accuracy at the expense of speed, only, is called a predictor corrector method.

The predictor corrector method is easily illustrated by the following flow diagram


Input $\vec{y}_o$, h, $t_o$
$\vec{g} \Leftarrow \vec{f}(\vec{y}_o, t_o)$
$\vec{y}_r \Leftarrow \vec{y}_o$
$t_r \Leftarrow t_o$
$\vec{y}_c \Leftarrow \vec{y}_r + h\vec{g}$
$\vec{y}_p \Leftarrow \vec{y}_c$
$\vec{g} \Leftarrow \vec{f}(\frac{\vec{y}_r + \vec{y}_p}{2}, t_r + \frac{h}{2})$
$\vec{y}_c \Leftarrow \vec{y}_r + h\vec{g}$
$\vec{e} \Leftarrow \text{norm}(\vec{y}_c - \vec{y}_p)/\text{norm}(\vec{y}_c)$
? e > ε
T
F
Output?
$\vec{y}_r \Leftarrow \vec{y}_c$
$t_r \Leftarrow t_r + h$

The PC method above has the advantages of the forward difference method outlined previously, an error of $(\frac{h}{2})^2$ and is self-starting. It is more expensive than the central difference method of the same accuracy in computing time. However, its self-starting characteristic is usually reward enough to offset that expense.

We will have need to integrate equations like the one shown above and equations like

$$\dot{\vec{y}} = J(t)\ \vec{y} + \vec{F}(t) \tag{47}$$

This "new type" of equation is simply a linear differential equation with variable coefficients, namely the matrix J(t) a and a forcing function $\vec{F}(t)$. When this is a deterministic form, there are essentially no unanswered questions if we are competent enough such that the previous brief discussion on numerical integration was very boring. However!, we are not going to be discussing deterministic forms. The matrix J(t) and the vector $\vec{F}(t)$ will be functions of discrete data (actually, they will be formed from the discrete solutions of $\dot{\vec{y}} = \vec{f}(\vec{y},t)$).

The best means of integrating these non-deterministic problems is an open question. The PC method outlined above is adequate. The values of J(t) and $\vec{F}(t)$ used will be at the half-step of the integration step under consideration.

Many refinements can be made to this lacksadasial procedure. However, these refinements may be inconsistent with the accuracy of the problems in mind. For these reasons and others, Runge-

Kutta and Adams-Moulton techniques are often not used for the problems in mind. For problems requiring an extreme accuracy, there are generally better methods than the RK and AM schemes.

## VI

## Linear Multipoint Boundary Value Problems as Initial Value Problems

We are going to discuss an old, old method for solving these problems. It is frequently used in analytic studies, but, for some reason it is relatively rare in numerical integration studies. Consider a linear differential equation of Nth order

$$\dot{\vec{y}} = J\,\vec{y} + \vec{F} \tag{48}$$

that is subject to at least N boundary conditions. It is elementary that the solution may be written in terms of a particular solution plus a linear combination of N linearly independent homogeneous solutions. The particular solution is the solution of

$$\dot{\vec{y}}^{(o)} = J\,\vec{y}^{(o)} + \vec{F} \tag{49}$$

subject to a finite initial condition vector $\vec{y}^{(o)}(o)$. The homogeneous solutions are solutions of

$$\dot{\vec{y}}^{(k)} = J\,\vec{y}^{(k)} \qquad 1 < k < N \tag{50}$$

subject to linearly independent finite initial conditions. For the initial conditions to be linearly independent, the following must hold;

$$\det \begin{pmatrix} \vec{y}^{(1)}_{(o)} & \vec{y}^{(2)}_{(o)} & \cdots & \vec{y}^{(N)}_{(o)} \end{pmatrix} \neq 0 \tag{51}$$

This is a generalization of what is commonly referred to as the Wronskian monstrosity.

Thus, we can unhesitatingly write

$$\vec{y} = \vec{y}^{(o)} + \sum_{k=1}^{N} a_k \vec{y}^{(k)} \tag{52}$$

where the combining coefficients, $(a_k)$, must be determined in such a manner so as to satisfy the boundary conditions. If there are N boundary conditions, the a's will be unique. If there are more than N boundary conditions, the a's will also be dependent upon the minimization of the norm chosen.

This method has the obvious advantage of accuracy and efficiency in storage. It often has efficiency in execution time too. With a little ingenuity, about all that has to be stored in an (N+1) by (N+2) matrix for the solution of the exactly determined problem. For a problem with P boundary conditions, a (P+1) by (N+2) array must be stored for use in calculating the a's.

The initial conditions of the particular solution should reflect the maximum knowledge of the true initial conditions. If the a's are all zero, then the initial conditions used are the true initial conditions for the problem. However, the initial conditions of the homogeneous solutions have many possibilities. An obvious valid choice for the vectors $\vec{y}_{(o)}^{(k)}$, $1 < k < N$ is the columns of an identity matrix. This is a very simple nonsingular matrix and the columns (or rows) of any nonsingular matrix satisfy the generalized monstrosity given above.

However, practicality and generality occasionally wreck the simple plans of men. I've encountered some problems in which it is necessary to use the following scheme. Use the same vector that is considered apropos for the particular solutions except multiply the kth element of the vector by some scalar. This scalar has few restrictions but something positive, nonunity, and less than two seems preferable. Generally 1.1 to 1.5 seems very good. A further exception is that if the kth element is null, use something other than zero for that element, say one.

Although the one shot procedure described theoretically works, it can be rendered useless by roundoff error. It is therefore a good procedure to make it a repetitive procedure such that we are seemingly trying to find the initial conditions of the problem. When we have obtained the true initial conditions, all the a's go to zero or some negligible values.

A further improvement is to superimpose particular solutions of Eq. (48). To do this, we write

$$\vec{y} = \sum_{k=1}^{n+1} g_k \vec{y}(k) \tag{53}$$

where the elements of $\vec{y}$ are chosen to satisfy the boundary conditions and the auxiliary condition

$$\sum_{k=1}^{n+1} g_k = 1 \tag{54}$$

This auxiliary condition obviously states that in the superposition, the forcing functions must sum to be only $\vec{F}(t)$. This strategy does add greatly to the control of roundoff error in the super-

position of solutions. The initial conditions used are those previously discussed, except, the solution $\vec{y}^{(o)}$ is now $\vec{y}^{(n+1)}$. If the true initial conditions are used for $\vec{y}^{(n+1}$

$$g_{n+1} = 1$$
$$g_j = 0 \qquad 1 < j < n \tag{55}$$

VII

## Quasilinearization

We will now attempt to tie all these things together to attempt to solve the identification problems mentioned previously. We wish to solve problems that have nonlinear solutions. The governing differential equations are often linear, but, the problem is nonlinear because as shown in the simplest identification problem, the desired answers are not linear combinations of the given data. We have discussed Newton Raphson techniques whereby we could solve nonlinear problems by successive linear steps. We have also discussed one of many methods for solving linear multipoint boundary value problems.

My guess as to what is quasilinearization is that it is the totality of the following computational schemes and portions of computational schemes:

1) Formulate the differential equations in the first order coupled form as shown earlier.
2) If there are unknown parameters, supplement the above differential equations with the null differential equations described earlier.

3) If there are more data points than unknowns, select a norm which will be used as a "goodness of fit" measure.

4) Do a Newton-Raphson-Kantorovich expansion of the nonlinear equations to obtain a related set of linear differential equations.

5) Solve the set of linear variable coefficient differential equations subject to the boundary conditions that may be exact and/or best fit. This solution will generally be obtained as a sequence of initial value problems to have the highest reasonable degree of accuracy. The procedure in this step is repeated until the initial conditions are known for the nonlinear differential equations such that the solutions generated for these initial conditions satisfy the given boundary conditions. If the problem is not exactly determined, the question arises as to what does satisfy mean.? This question has to be answered for each problem.

## VIII

## Applications

The applications of such methods are ubiquitous. The essence of engineering design might be said to be, "select a system governed by the laws of physics and that benefits mankind with economics as a constraint". Generally in engineering design

the benefit to mankind is established before we get the problem. Thus, we have only to formulate the laws of physics, which for simple mechanical problems is simply Newton's law, and pick the parameters that make it economical. Let's mention a few examples:

1) For some unknown reason, it is beneficial to mankind to fly out in space. A space vehicle is attracted by the earth's gravitational field. There is a potential function whose gradient is the acceleration due to gravity. The acceleration due to gravity is not a constant for a given radial distance from the earth's center. Identification of the Fourier coefficients of an eigenfunction expansion of this potential should be possible via the means we have described.

2) A control system that performs a very simple function like raising a bulldozer blade is very nonlinear and might be represented by

$$\ddot{x} + a_1 \dot{x} + a_2 x + a_3 \dot{x}^2 x + a_4 \dot{x} x^2 = a_5 u(t)$$

The a's are functions of blade weight, pump and pipe sizes, oil properties, pressures, etc. It would be desirable to pick the a's such that a desired system response is obtained and economy is also achieved. This could be done if the proper variation of the a's can be controlled. However, these a's are not yet known. They can be determined by measuring the responses

of real systems. From these responses, empirical values of a's can be obtained.

3) Similar problems abound in almost all process plants, control systems, and any time variant system.

## IX

## Data Structures

The concepts of data structures should be applied to most large programs on digital computers. These concepts involve knowing the manners in which data are stored internally in digital computers.

To visualize a very simple example, consider a Newton Raphson algorithm like

$$\vec{x}_{n+1} = \vec{x}_n - J_n^{-1} \vec{x}_n$$

In the calculation procedure, the usual thing to do would be to have two vectors reserved for the $\vec{x}_n$ and $\vec{x}_{n+1}$. When $\vec{x}_{n+1}$ is calculated and convergence has not been achieved, the obvious thing to do would be to store the $\vec{x}_{n+1}$ in the space used for $\vec{x}_{no}$ on the last iteration. A frightfully more efficient scheme would be to interchange the names of $\vec{x}_{n+1}$ and $\vec{x}_n$.

Ideas of this type are easily implemented if one is familiar with data storage characteristics. Such methods could reduce execution times of many quasilinearization programs by up to 30%.

# X

## Extensions

The extensions that can be made to the groundwork we have laid here are numerous.

Typically:

1) Formulating general rules governing cost functions.
2) Inclusion of norms other than least square norms. All other norms will require iterative solutions within each iteration. Is this expense worth it?
3) Development of high accuracy integration schemes for the variable coefficient differential equations. Problems like the geopotential problem demand it.
4) Use of symbol manipulation compilers to reduce the effort in formulating the first order and variable coefficient differential equations.

## DOCUMENT CONTROL DATA - R & D

*(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)*

| | |
|---|---|
| 1. ORIGINATING ACTIVITY (Corporate author)<br>University of Houston | 2a. REPORT SECURITY CLASSIFICATION<br>Unclassified<br>2b. GROUP<br>Unclassified |

3. REPORT TITLE
Identification of Systems

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)
Technical Report

5. AUTHOR(S) (First name, middle initial, last name)
S. Bart Childs

| | | |
|---|---|---|
| 6. REPORT DATE<br>August, 1968 | 7a. TOTAL NO. OF PAGES<br>34 | 7b. NO. OF REFS<br>- |
| 8a. CONTRACT OR GRANT NO.<br>N00014-68-A-0151<br>b. PROJECT NO.<br>c.<br>d. | 9a. ORIGINATOR'S REPORT NUMBER(S)<br>RE 2-68<br>9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report) | |

10. DISTRIBUTION STATEMENT
Reproduction in whole or in part is permitted for any purpose of the United States Government. This document has been approved for public release and sale; its distribution is unlimited.

| | |
|---|---|
| 11. SUPPLEMENTARY NOTES | 12. SPONSORING MILITARY ACTIVITY<br>ONR-Washington |

13. ABSTRACT

This paper is a tutorial discussion of identification problems. The basic principles of one method of identification are discussed. The method of quasilinearization has been developed by Richard Bellman and Robert Kalaba and others. This method is discussed in some detail with emphasis on programming strategies. These strategies are intended to improve the utility of a general quasilinearization program. Specific strategies are concerned with expansion of convergence space, superposition of particular solutions, and convergence criteria.

| 14 KEY WORDS | LINK A | | LINK B | | LINK C | |
|---|---|---|---|---|---|---|
| | ROLE | WT | ROLE | WT | ROLE | WT |
| numerical integration<br>nonlinear ordinary differential equations<br>system identification<br>multi-point boundary value problems<br>Newton-Rhapson methods | | | | | | |