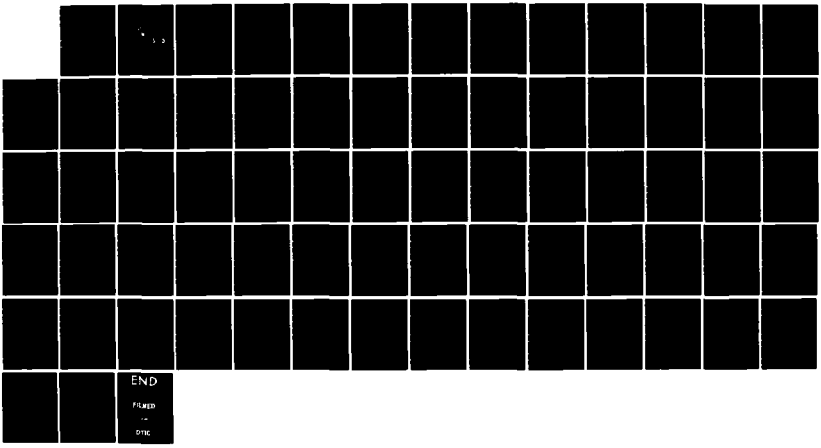
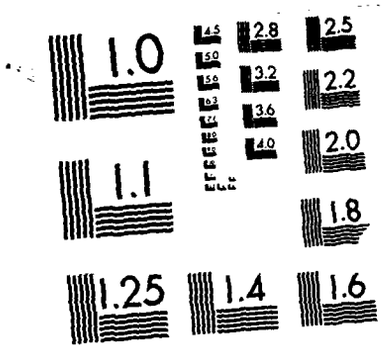
AD-A155 934 A NEW METHOD FOR BOUNDARY VALUE PROBLEMS(U) NAVAL POSTGRADUATE SCHOOL MONTEREY CA F HEREDIA N MAR 85 1/1

UNCLASSIFIED F/G 12/1 NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

AD-A155 934

NAVAL POSTGRADUATE SCHOOL

Monterey, California

DTIC
ELECTE
JUL 05 1985
S D
G

THESIS

A NEW METHOD FOR BOUNDARY VALUE PROBLEMS

by

Fernando Heredia N.

March 1985

Thesis Advisor: David Salinas

Approved for public release; distribution is unlimited

DTIC FILE COPY

85 6 11 060

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO. AD-A155 934	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) A NEW METHOD FOR BOUNDARY VALUE PROBLEMS		5. TYPE OF REPORT & PERIOD COVERED Master's Thesis March 1985
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Fernando Heredia N.		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93943		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, California 93943		12. REPORT DATE March 1985
		13. NUMBER OF PAGES 70
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution is unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Boundary Value Problems,
New Method for Value Boundary Problems.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Large matrix storage constitutes a limitation on the applicability of most numerical techniques including the Finite Element Method, when very accurate results are required. This is particularly true when dealing with Boundary Value Problems. In order to surpass this difficulty a new method to solve these problems has been devised which does not require

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-LF-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

#20 - ABSTRACT - (CONTINUED)

matrix storage while still providing the possibility of accuracy improvement.

Although restricted to one-dimensional, linear differential equations of the form $Y^{(n)}(x) = f(x)$ this new approximating technique gives acceptable results. The method will perform equally well for problems with exact or non-exact integrable forcing functions, continuous or discontinuous, or functions existing only as a set of values at discrete points.

Accession For
NTIS GRA&I ☒
DTIC TAB ☐
Unannounced ☐
Justification
By
Distribution/
Availability Codes
Dist | Avail and/or Special
A/1

Approved for public release; distribution is unlimited.

A New Method for Boundary Value Problems

by

Fernando Heredia N.
Lieutenant, Ecuadorian Navy

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL

March 1985

Author: _____
Fernando Heredia N.

Approved by: _____
D. Salinas, Thesis Advisor

_____
P. Marto, Chairman,
Department of Mechanical Engineering

_____
John Dyer,
Dean of Science and Engineering

ABSTRACT

Large matrix storage constitutes a limitation on the applicability of most numerical techniques including the Finite Element Method, when very accurate results are required. This is particularly true when dealing with Boundary Value Problems. In order to surpass this difficulty a new method to solve these problems has been devised which does not require matrix storage while still providing the possibility of accuracy improvement.

Although restricted to one-dimensional, linear differential equations of the form $Y^{(n)}(x) = f(x)$ this new approximating technique gives acceptable results. The method will perform equally well for problems with exact or non-exact integrable forcing functions, continuous or discontinuous, or functions existing only as a set of values at discrete points.

TABLE OF CONTENTS

I. INTRODUCTION ---------- 6

II. BACKGROUND ---------- 9

A. GENERAL ---------- 9

B. THE FIRST INTEGRATION ---------- 11

III. HIGHER ORDER DIFFERENTIAL EQUATIONS ---------- 17

A. GENERAL ---------- 17

B. DEVELOPMENT OF THE METHOD ---------- 22

IV. ERROR ANALYSIS AND CONVERGENCE ---------- 31

V. APPLICATIONS AND RESULTS ---------- 43

A. GENERAL ---------- 43

B. EXAMPLES ---------- 46

1. Example 1: Clamped-Free Beam ---------- 46

2. Solution ---------- 46

3. Example 2: Clamped-Roller Beam ---------- 49

4. Solution ---------- 49

5. Example 3: Pin-Roller Beam ---------- 55

6. Solution ---------- 55

7. Example 4: Clamped-Clamped Beam ---------- 59

8. Solution ---------- 59

C. THE COMPUTER PROGRAM ---------- 64

VI. CONCLUSIONS AND RECOMMENDATIONS ---------- 68

INITIAL DISTRIBUTION LIST ---------- 70

I. INTRODUCTION

For many engineering problems it is not always possible to find an exact solution. An exact solution is an analytical mathematical expression that gives the value of the unknown at any point within a previously specified range. For problems involving complex material properties and boundary conditions, different numerical techniques have been developed to approximate the exact solution to a more or less acceptable degree of accuracy. One of these techniques is the Finite Element Method, where a given problem is to be discretized, that is to say, approximate solutions are to be found at discrete points in the body. The way one accomplishes this, is by subdividing the whole body into finite elements. The solution is then formulated for each small unit and combined to obtain the solution of the whole system. Obviously the greater the number of elements, the better the accuracy.

However, despite the fact that high-speed digital computers have enabled engineers to successfully apply these numerical techniques, we still face the problem of large matrix storage. Complex problems requiring very accurate approximate solutions lead to large matrices and consequently larger computer memory will be required. This fact constitutes a limitation on the applicability of these numerical techniques.

The method we are dealing with in this work intends to solve this difficulty by reducing a B.V.P. to an I.V.P. Unlike the "shooting" method, where basically the same idea is used together with an iterative scheme to achieve a solution, the method developed here achieves a solution without iteration.

Although restricted to a particular type of linear differential equations and only to one-dimensional problems, as it will be shown, this method can be applied to many problems where no exact integrable forcing functions occur, or, even more, the function exists only as a set of values at specific points.

Basically what the method does is approximate the exact solution by a set of linear functions, each of them applying in a very small interval. However, these functions are not independent from each other. We use the previous one to find the next, until we eventually reach the other end of the range in consideration. Depending on the order of the differential equation, we will find as many sets of linear functions as required by its order, each set applying to a specific integration, and as before, every set depends on the previous one. The way in which we will do this will allow us to correlate all possible boundary conditions in an explicit manner which transforms boundary value problems into initial value problems and by doing that, get to the result.

The method is indeed an approximation and as such, it is subject to error. We can say, however, that by decreasing the

step size we get better accuracy, provided we carry enough significant digits to take care of round-off errors. With respect to the later a good programming technique is required. Finally we shall apply the method to several practical applications, specifically to beam problems where a fourth order differential equation occurs, and a variety of boundary conditions can be given.

II. BACKGROUND

A. GENERAL

In this section we are going to introduce the basic ideas of the method we will be dealing with. Let

$$Y'(x) = f(x) \tag{2.1}$$

where it is understood that the function f(x) may or may not be exactly integrable, but we can integrate it numerically. From the above relationship we have:

$$Y(x) = \int f(x)dx + Y_o \tag{2.2}$$

Here the constant of integration Y_o is assumed to be zero for the moment. In the most general case, it is clear that there is no way to know what the function Y(x) is, since we may be dealing with non-integrable functions. However, we can approximate the function Y(x) by a linear function y(x), provided the interval in which this approximation applies is sufficiently small. So we take

$$Y(x) = y(x) = mx + c \tag{2.3}$$

Now, by following the general procedure of integration, and from (2.2), we have:

$$\left| mx + c \right|_a^b = \int_a^b f(x)\,dx \qquad (2.4)$$

where (a,b) denote the limits of the interval under consideration. After simplification it is found that:

$$m = \frac{\int_a^b f(x)\,dx}{b - a} \qquad (2.5)$$

that is to say, the slope of the approximating line given by (2.3) can be explicitly determined. Actually, as we will see later, it can be said that the "general" approximate solution to (2.1) has been found within the interval from a to b. In order to determine the "particular" solution, in other words to find the intercept c of (2.3), an auxiliary condition must be imposed. Let

$$Y(a) = Y_o$$

then:

$$Y(a) = Y_o = ma + c$$

or

$$y(x) = m(x-a) + Y_o \qquad a \leq x \leq b \qquad (2.6)$$

where m is given by (2.4). Now we have fully determined the approximate solution inside the given interval. Note that (2.6) will give results at points a and b, as accurate as the numerical integration performed on (2.5).

Fig. 2.1 shows the whole process so far. The upper curve represents f(x), which is given. The lower curve is Y(x), the integral of f(x); however within the interval a to b, Y(x) is approximated by a line. Point Y_o is the given auxiliary condition and point Y_b can be determined exactly from (2.6), by letting x = b.

The next step now becomes evident, since point Y_b can be determined. We are now in the same position as before, so all we need to do is repeat the process over again. However, this time with a new auxiliary condition; the last one we have just found, and over the next interval. We keep going this way until we reach the other extreme of the range where the differential equation applies. See Fig. 2.2.

In summary, we can say that every step we take we are solving a "new" differential equation by approximating lines which gives us true values at the points of intersection. The fact that we have a "new" differential equation at every single step allows us to deal with discontinuous functions and/or functions existing only as a set of values.

B. THE FIRST INTEGRATION

At this point let us introduce a new parameter:

$$h = b-a \tag{2.7}$$

them. By similarity to an exact problem we can look at this term as being the general solution of the differential equation. Note that this term actually links the left and the right boundary conditions in a very explicit way. This is the heart of what the present method is all about. With this idea in mind we are now going to find several relationships for successive integrations.

Let us start by noticing that (2.10) is a general relation that applies not only to the first integration but also to successive ones. The only term that is particular for each integration is the summation of the slopes. So in order to perform the next integration we shall determine this term specifically and substitute it into (2.10). We are now dealing with a fourth order differential equation since this type occurs in beam problems. The application to other orders will be self-evident. Let

$$Y^{IV}(x) = f_4(x) \tag{3.6}$$

After a first integration we have:

$$Y'''_n = h \sum_{i=1}^{n}{}_3\, m_i + Y'''_o \tag{3.7}$$

where the subscript 3 in the summation indicates that these slopes belong to $Y''' = f_3(x)$ and are given by (2.11). We need to find now

There is another subscript that we must keep track of. Recall that the whole solution function is going to be represented by a set of approximating lines, see Fig. 2.2. Then if we start at the left end, we shall use the subscript n as a second indice to identify each line as we move rightwards. Be aware that we will take the subscript i immediately to the right of each line to name it locally.

Some words must be said about the auxiliary conditions. Recall that in a specific problem the number of these conditions are equal to the order of the differential equation to be solved. If all these conditions are given at the same point, then we are dealing with an Initial Value Problem. However, if the auxiliary conditions are given at both ends of the range of interest, then we have a Boundary Value Problem. When we are able to have an exact solution, we usually are led to a set of simultaneous equations involving the unknown B.C.'s; those can be solved algebraically and lead to the whole solution. However for equations with no exact solution, like the ones we are interested in, there is no way to determine the conditions at the other end of the interval, and that is precisely the goal of our approach. We need to correlate in an explicit way the known conditions and the unknown ones by means of equations of the form of (2.10), so we can be able to find them.

Note that in this equation the term $h \sum_{i=1}^{n} m_i$ is independent of B.C.'s and it can be readily found even if we do not know

Such equations arise in the study of deflections of beams, and we shall apply the method to practical beam problems. Let us introduce the notation to be followed. We will write the approximate solution line of (3.5) as:

$$y(x) = mx + c$$

in a clean way without subscripts. The approximation line to the first derivative will then be:

$$y'(x) = m_1 x + c_1$$

and for the second derivative:

$$y''(x) = m_2 x + c_2$$

and so on.

The boundary conditions will be represented as:

$$Y(x_o) = Y_o \qquad Y(x_n) = Y_n$$

$$Y'(x_o) = Y'_o \qquad Y'(x_n) = Y'_n$$

$$Y''(x_o) = Y''_o \qquad Y''(x_n) = Y''_n$$

etc., where the subscript o refers to the left hand side, and the subscript n to the right, of the range under consideration.

$$x = \frac{1}{2}(a + b)$$

in the line $y'(x)$ and we get:

$$y'\left(\frac{a+b}{2}\right) = m_1\left(\frac{a+b}{2}\right) + c_1 = m$$

As seen, it reproduces m, the slope of the lower straight line.

Now we can have a deeper insight of the whole process, and why it works. See that what we are really going to do is to use the slope at the midpoint (the average of the slopes at the extremes) to approximate the true slope of the straight line $y(x)$, and later determine the corresponding intercepts. In fact, had we known exactly the function $Y'(x) = f_1(x)$, we would have been able to determine the true slope of $y(x)$ by direct integration as before. As we can see now, we are introducing a source of error, and from now on we will not have as accurate results as in the first integration.

B. DEVELOPMENT OF THE METHOD

Before we go into a general development of this method a slightly different notation must be introduced, and we will be using it throughout the rest of this research. The type of differential equations we will be looking at are of the form:

$$Y^{iv}(x) = f(x) \qquad (3.5)$$

have been given at the left hand side. However, it really need not be like this, as will be shown later.

At this point some considerations are in order. With reference to Fig. 3.1 where the upper curve is the given function $Y''(x) = f_2(x)$, the middle curve is $Y'(x) = f_1(x)$, and the lower one is the solution $Y(x) = f(x)$.

As explained previously we have approximated the solution by straight lines. Consider now the slope of the lower straight line m which is given by (3.2). This equation can be written as:

$$m = \frac{1}{2}[(m_1 a + c_1) + (m_1 b + c_1)]$$

$$m = \frac{1}{2}[y_1(a) + y_1(b)]$$

$$m = \frac{1}{2}[Y_a + Y_b] \tag{3.4}$$

Since the ordinate at any point x in the $Y'(x)$ curve is indeed the slope of the $Y(x)$ curve at the same point, then by virtue of (3.4), we can say that the slope m of the lower straight line is the average of the slopes corresponding to its end points, namely point Y_a and Y_b in Fig. 3.1. Recall that these two slopes are given exactly by the ordinates of the middle curve $Y'(x)$. Furthermore let us evaluate $y'(x) = m_1 x + c_1$, the equation of the upper straight line, at the midpoint of the interval, that is we let:

but by (2.4)

$$m = \frac{\int_a^b (m_1 x + c_1)\,dx}{h}$$

or:

$$m = \frac{m_1}{2}(a + b) + c_1 \qquad (3.2)$$

So we know now the slope of the next line; the line belonging to the solution function. In order to find the intercept c, another auxiliary condition must be supplied. Let:

$$Y(a) = Y_a$$

then, replacing this condition in the line $y(x) = mx + c$, we get:

$$Y(a) = Y_a = ma + c$$

It can be shown that:

$$y(x) = m(x-a) + Y_a \qquad (3.3)$$

where m is given by (3.2). Eq. (3.3) gives the approximate solution of the original second order differential equation, inside the interval a to b. Note that both auxiliary conditions

$$Y(x) = f(x)$$

be the exact result, which is the lower curve in the same figure. Now let

$$y'(x) = m_1 x + c_1$$

and

$$y(x) = mx + c$$

be the approximating lines to $f_1(x)$ and $f(x)$, respectively, that applies inside the interval from a to b only. In this figure, we can easily see that the area enclosed by the upper straight line is almost equal to the area enclosed by the function $f_1(x)$. Obviously as the interval becomes smaller and smaller, both areas tend to be equal; in the limit they are indeed equal. It is evident now that we can approximate the exact integration of $f_1(x)$, by integrating the line $y'(x) = m_1 x + c_1$, provided the interval is small enough so the error is negligible. Now, as before, we need to determine m and c. Recall that m_1 and c_1 have already been found. So by virtue of (2.5):

$$m = \frac{\int_a^b f_1(x)\,dx}{h}$$

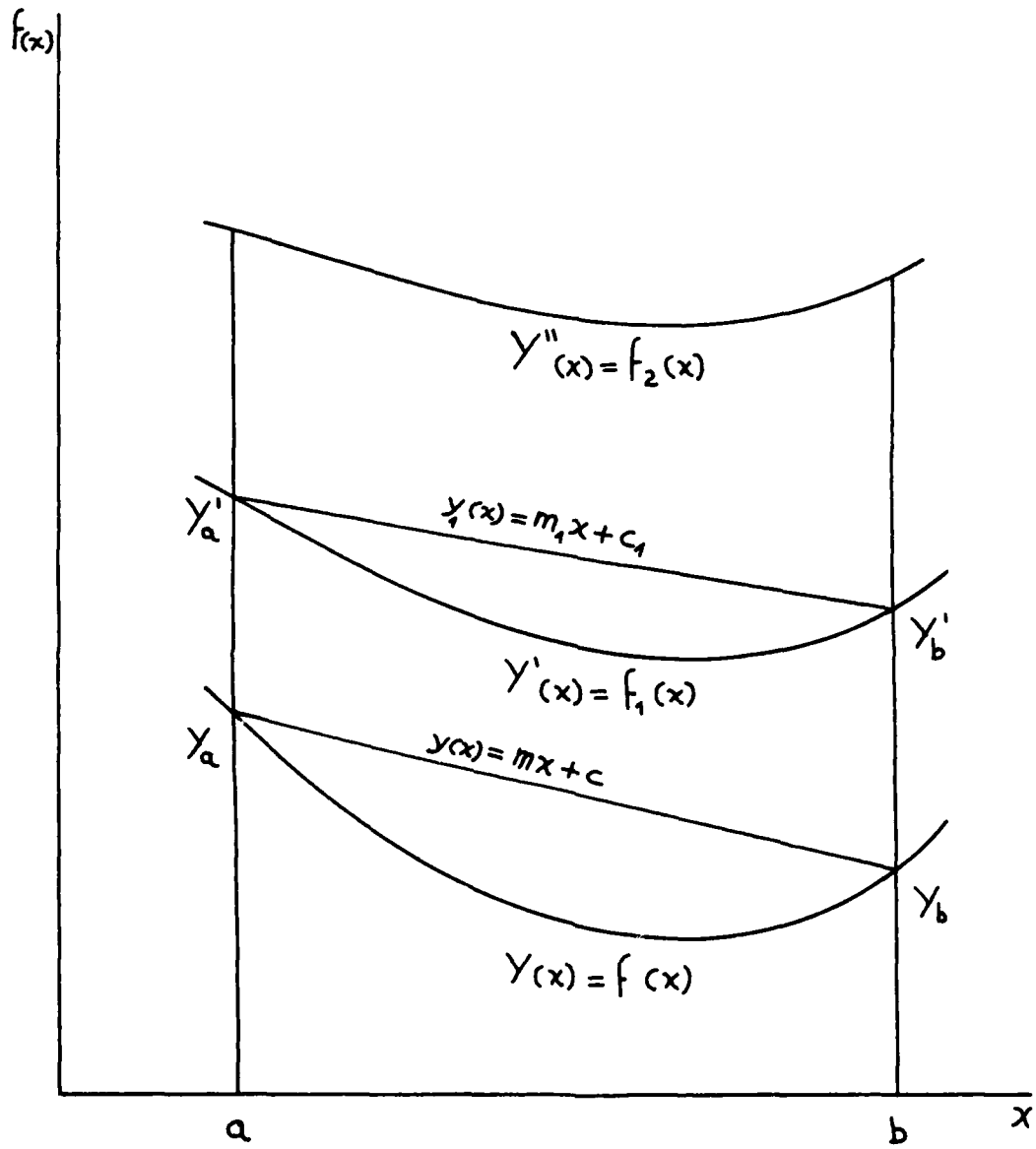


FIGURE 3.1

III. HIGHER ORDER DIFFERENTIAL EQUATIONS

A. GENERAL

The previous chapter was dealing basically with the main ideas of the method, and we have solved a first order differential equation. We are now going to extend the method to higher order equations. Essentially what we will do is, integrate several times the right hand side of the given equation; as required for its degree, using the same approach as before. However, we should keep in mind that it is only for the first integration that we will use the given function f(x). After this integration, this function is no longer available because what we have is a set of straight lines, each applying to a specific interval. We need to find now a method that will allow us to perform a second integration of the function f(x), from the given set of lines.

In order to do this, let us consider the curves of Fig. 3.1. Suppose we have been given the following second order differential equation:

$$Y''(x) = f_2(x) \tag{3.1}$$

represented in Fig. 3.1 by the upper curve. Let us assume we have determined $Y'(x) = f_1(x)$ after a first integration by the method just introduced, Let

for this first integration, so its accuracy is determined only by the exactness of the integration routine used. Second, the values of Y(x) can be directly determine at any arbitrary point n, that is to say we need not find its previous values as is the case for many numerical methods. Later this fact will become more useful. Finally, we see that, by transposing terms we can find either the left or right B.C., and this apparently unimportant fact is indeed a key step, since as we will see later, we will be able to correlate B.C.'s of higher order differential equations and solve for them without having to generate all previous values of the unknown function. In other words, we could transform a Boundary Value Problem into an Initial Value Problem and vice versa, depending on what is known, and what we are looking for.

$$Y_n = (m_1 + m_2 + \ldots + m_n)h + Y_o$$

or

$$Y_n = h \sum_{i=1}^{n} m_i + Y_o \tag{2.10}$$

Similarly,

$$m_1 = \frac{\int_0^h f(x)dx}{h}$$

$$m_2 = \frac{\int_h^{2h} f(x)dx}{h}$$

$$\vdots$$

Adding terms gives:

$$\sum_{i=1}^{n} m_i = \frac{\int_0^{nh} f(x)dx}{h} \tag{2.11}$$

We can replace in (2.10) to obtain:

$$Y_n = \int_0^{x_n} f(x)dx + Y_o \tag{2.12}$$

since x = nh. Equations (2.10) and (2.12) constitute two very important results. Eq. (2.12) is indeed not an unexpected one, and several considerations can be drawn from it. First of all the value of Y_n does not depend on the step size h, at least

where h is assumed to be small and constant throughout the analysis. We are now in a position to determine a general relationship which will allow us to find the value of Y(x) at discrete points, namely at the intersection of the straight lines. From (2.6) we have:

$$y(b) = m(b-a) + Y_o = mh + Y_o .$$

In general,

$$Y_{i-1} = m_i h + Y_{i-1} \qquad 0 \leq i \leq n \tag{2.8}$$

where

$$m_i = \frac{\int_{x_{i-1}}^{x_i} f(x)\,dx}{h} \tag{2.9}$$

but:

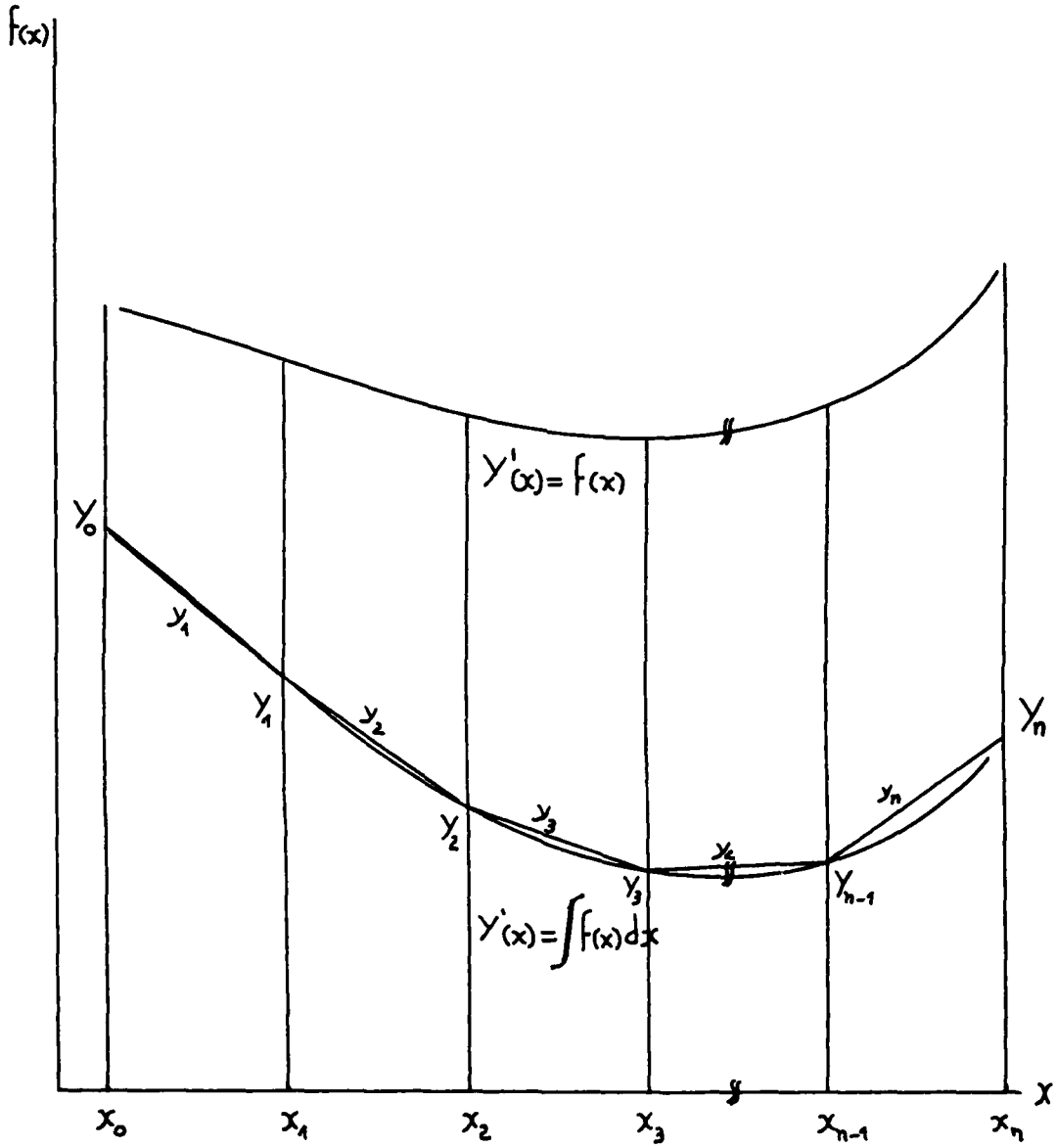
$$Y_{i-1} = m_{i-1}h + Y_{i-2}$$

$$Y_{i-2} = m_{i-2}h + Y_{i-3}$$

$$\vdots$$

$$Y_1 = m_1 h + Y_o$$

Replacing these last relationships in (2.8) and after factoring h, we have:



$y_i(x) = m_i x + c_i \qquad 0 \le i \le n$

FIGURE 2.2

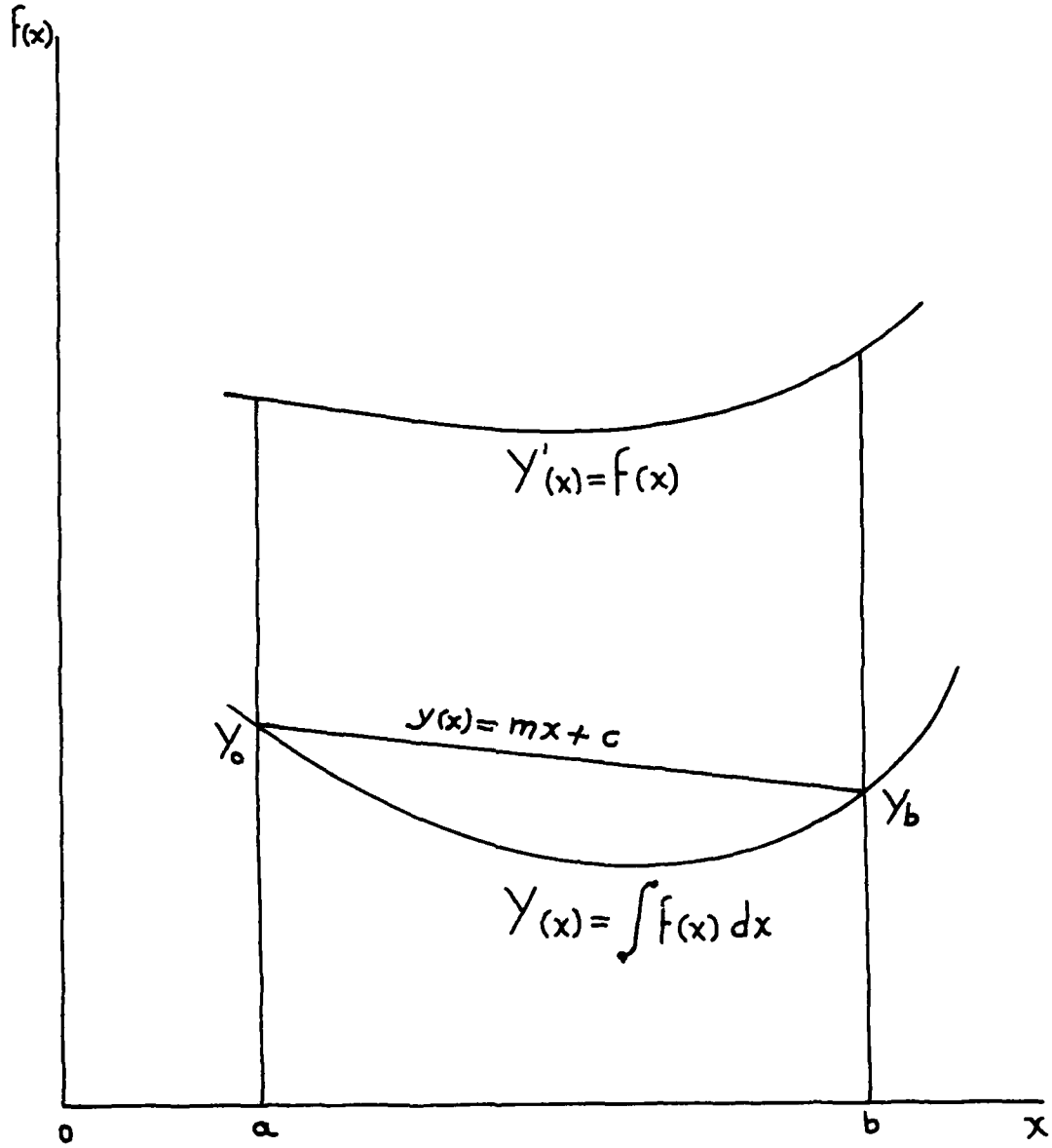


FIGURE 2.1

$$Y''_n = h \sum_{i=1}^{n}{}_2 \; m_i + Y''_o \tag{3.8}$$

Following our convention, $m_{2,i}$ is given by:

$$m_{2,i} = \frac{m_{3,i}}{2}(x_{i-1} + x_i) + b_{3,i}$$

but

$$b_{3,i} = Y'''_{i-1} - m_{3,i}x_{i-1}$$

then

$$m_{2,i} = \frac{m_{3,i}}{2}(x_{i-1} + x_i) + Y'''_{i-1} - m_{3,i}x_{i-1}$$

or

$$m_{2,i} = \frac{m_{3,i}}{2} h + Y'''_{i-1} \tag{3.9}$$

Now, since

$$Y'''_{n-1} = h \sum_{i=1}^{n-1}{}_3 \; m_i + Y'''_o$$

then

$$m_{2,n} = \frac{m_{3,n}}{2} h + h \sum_{i=1}^{n-1}{}_3\, m_i + Y_o''' \tag{3.10}$$

This last relationship in the way it has been derived becomes a general one, and expresses the slope $m_{2,n}$ at any point i as a function of the previous slopes. Similar relations can be derived for successive integrations just by shifting the first indice. Now we need to find the summation of these slopes. It becomes:

$$\sum_{i=1}^{n}{}_2\, m_i = \frac{h}{2} \sum_{i=1}^{n}{}_3\, m_i + h \sum_{i=1}^{n-1} \sum_{j=1}^{i}{}_3\, m_j + nY_o''' \tag{3.11}$$

Now, replacing (3.11) in (3.8):

$$Y_n'' = h[\frac{h}{2} \sum_{i=1}^{n}{}_3\, m_i + h \sum_{i=1}^{n-1} \sum_{j=1}^{i}{}_3\, m_j + nY_o''] + Y_o''$$

or

$$Y_n'' = h^2[\frac{1}{2} \sum_{i=1}^{n}{}_3\, m_i + \sum_{i=1}^{n-1} \sum_{j=1}^{i}{}_3\, m_j] + nhY_o''' + Y_o'' \tag{3.12}$$

Let:

$$S_2 = \frac{1}{2} \sum_{i=1}^{n}{}_3\, m_i + \sum_{i=1}^{n-1} \sum_{j=1}^{i}{}_3\, m_j$$

Note that, as stated above, this last term S_2 is also independent of boundary conditions and can be readily determined. Equation (3.12) gives the particular solution to a second order differential equation as well as, in this specific case, it represents the second derivative of Y(x).

The whole process, so far, can be summarized as follows: Equation (2.10) gives the solution at any point n of any integration, first second, etc. In order to use this equation we need to find the summation term, and this is given by a relation of the form of (3.11), which is also a general relationship and expresses the summation of slopes in terms of the previous one. Recall that the only slope summation we know is the one involving the known function f(x), and is given by (2.11). That is why all further summations must be expressed as a function of this one. This last summation is then replaced in (3.8), and simplified if possible. The next step is to identify the terms that are independent of boundary conditions and isolate them as in (3.12) and to evalute them separately.

We can continue in the same way until we get a solution formula for Y(x). It can be shown that the complete solution for a fourth order differential equation of the form $Y^{iv}(x) = f(x)$ is given by the following relationships:

$$Y_n''' = S_3 + Y_o''' \tag{3.13}$$

$$Y_n'' = hS_2 + nhY_o''' + Y_o'' \tag{3.14}$$

$$Y_n' = h^2 S_1 + N_1 h^2 Y_o''' + nhY_o'' + Y_o' \tag{3.15}$$

$$Y_n = h^3 S_o + N_o h^3 Y_o''' + N_1 h^2 Y_o'' + nhY_o' + Y_o \tag{3.16}$$

where:

$$S_3 = \sum_{i=1}^{n} m_i \tag{3.17}$$

$$S_2 = \frac{1}{2} \sum_{i=1}^{n} m_i + \sum_{i=1}^{n-1} \sum_{j=1}^{i} m_j \tag{3.18}$$

$$S_1 = \frac{1}{4} \sum_{i=1}^{n} m_i + \sum_{i=1}^{n-1} \sum_{j=1}^{j} m_j + \sum_{i=1}^{n-2} \sum_{j=1}^{i} \sum_{k=1}^{j} m_k \tag{3.19}$$

$$S_o = \frac{1}{8} \sum_{i=1}^{n} m_i + \frac{3}{4} \sum_{i=1}^{n-1} \sum_{j=1}^{i} m_j + \frac{3}{2} \sum_{i=1}^{n-2} \sum_{j=1}^{i} \sum_{k=1}^{j} m_k + \sum_{i=1}^{n-3} \sum_{j=1}^{i} \sum_{k=1}^{j} \sum_{\ell=1}^{k} m_\ell \tag{3.20}$$

$$N_1 = \frac{n}{2} + \sum_{i=1}^{n-1} i \tag{3.21}$$

$$N_o = \frac{n}{4} + \sum_{i=1}^{n-1} i + \sum_{i=1}^{n-2} \sum_{j}^{i} j \tag{3.22}$$

and

IV. ERROR ANALYSIS AND CONVERGENCE

Refer to Fig. 4.1 where we have plotted the functions $Y' = f_1(x)$ and $Y = f(x)$ together with their respective approximating straight lines inside the interval $x = a$ to $x = b$. We assume that the upper straight line belongs to a first integration; in other words we have not introduced any error so far. A second integration, however, will be performed using this line $y_1(x)$ since we do not know the function $f_1(x)$. It is worth considering now what happens when we integrate the line instead of the function itself. Let m be the slope of the lower straight line. This is given by:

$$m = \frac{\int_a^b f_1(x)\,dx}{b-a}$$

Let m* be the approximating slope given by:

$$m^* = \frac{\int_a^b (m_1 x + c_1)\,dx}{b-a} \qquad (4.1)$$

Now for the particular case shown in the figure, we have:

$$\int_a^b (m_1 x + c_1)\,dx > \int_a^b f_1(x)\,dx \qquad (4.2)$$

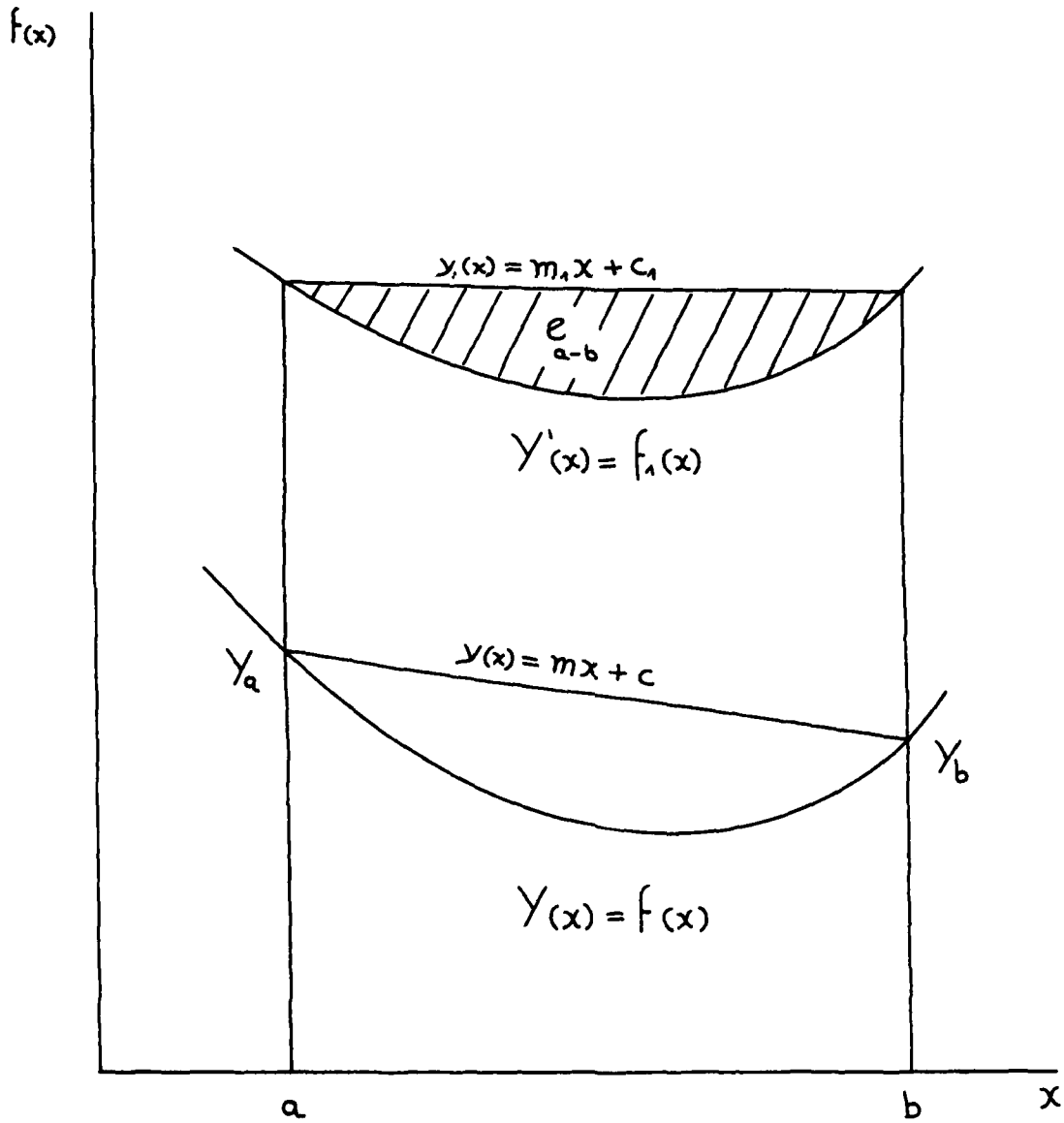


FIGURE 4.1

since the area enclosed by the curve is smaller than the area enclosed by the line (we assume that there are no inflection points and/or discontinuities inside the interval). It is now clear that:

$$m^* > m$$

and the error introduced is:

$$e_{a-b} = \int_a^b (m_1 x + c_1)\,dx - \int_a^b f_1(x)\,dx \qquad (4.3)$$

which is equal to the shaded area in Fig. 4.1.

As indicated in Chapter III, m^* comes out to be the average of the slopes at the end points of the lower straight line, and it is this slope we work with. So, the actual approximating line $y^*(x) = m^*x + c^*$, shown in Fig. 4.2 is steeper than $y(x)$ since $m^* > m$. The distance ED becomes the error introduced and we can not evaluate it because we do not know $f(x)$. Another integration using $y^*(x)$ will make things even worse. Note that the error to be committed this time will be larger and is given by the shaded area in Fig. 4.2.

So far it appears that the method is divergent in nature due to the fact that the error will grow bigger and bigger unless we find some means to correct it. At this point, the only possible way to do this is by reducing the "shaded"

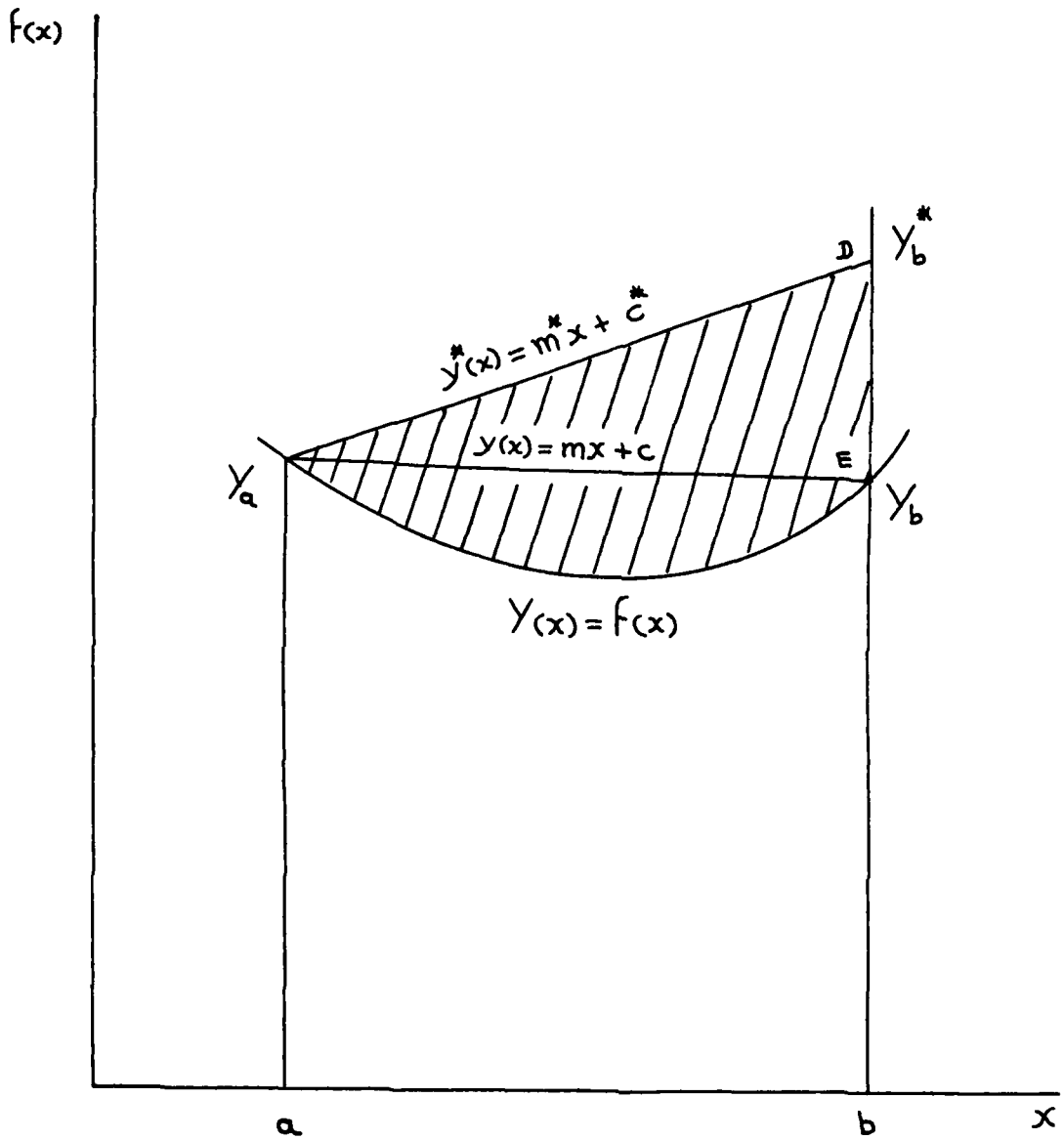


FIGURE 4.2

areas, which means that we have to reduce the step size. Fortunately, we can say, at least in theory, that in the limit there will be no error; in practice however, a very small step size will increase the round-off errors. The examples provided show that the approximation is indeed acceptable.

There is a special case, however, where an error correction can be made. This applies only to boundary value problems. Recall that a solution of any differential equation by this method is given by the general relationship:

$$Y_n = h \sum_{i=1}^{n} m_i + Y_o$$

In B.V.P., we know the conditions at both ends of the interval. So this last equation can be written as:

$$\sum_{i=1}^{n} m_i = \frac{Y_o - Y_n}{h} \qquad (4.4)$$

where the right hand side is known. Now since we are actually dealing with approximations to the true slopes, what we really have as a summation is the next term, call it M:

$$M = \sum_{i=1}^{n} m_i^*$$

Our approximate solution is then:

$$Y_n^* = h \sum_{i=1}^{n} m^* + Y_o \tag{4.5}$$

and the error at point n will be given by:

$$e_n = Y_n^* - Y_n \tag{4.6}$$

Now let us see how the correction could be carried out individually at each interval. To do this we must find the exact slope m. It is given by:

$$m = m^* - e$$

where e is the shaded area of Fig. 4.1 and is given by (4.3). The slope summation is therefore:

$$\sum_{i=1}^{j} m_i = \sum_{i=1}^{j} m_j^* - je$$

Multiplying by h and adding Y_o to both sides, we obtain:

$$h \sum_{i=1}^{j} m_i + Y_o = h \sum_{i=1}^{j} m_i^* + Y_o - jhe$$

or

$$Y_j = Y_j - jhe \tag{4.7}$$

When j = n, we have:

$$Y_n = Y_n^* - nhe$$

or

$$Y_n^* - Y_n = nhe$$

but from (4.6) we get:

$$e = \frac{e_n}{nh} \tag{4.8}$$

Replacing this last relation in (4.7):

$$Y_j = Y_j^* - \frac{j}{n} e_n$$

and the general relationship given by (2.10) transforms into:

$$Y_j = h \sum_{i=1}^{j} m_i^* + Y_o - \frac{j}{n} e_n \tag{4.9}$$

which is the corrected solution at any point j. In summary, what we need to do to correct the solution is: a) determine Y_n^* by performing the integration without any correction by

using equation (4.5); b) compare Y_n^* with the given B.C. Y_n and obtain e; c) Perform again the integration this time using the relationship given by (4.9).

From (4.9) we can find a whole set of corrected equations similar to those given by (3.13) to (3.20). However, by doing this we will enormously complicate those relationships and the computational effort, together with round-off errors, may not give any advantage at all, especially since we expect the local error e to be very small.

There is, however, another stronger reason not to do that. The fact is that we have assumed that the error e is a constant and this is not always the case. If we restrict ourselves to integrate linear and/or constant functions, then the corrected method could be justified since for these functions the error given by (4.3) is a constant, for the shaded area of Fig. 4.1 will always be the same. In more general cases, the error e will be unpredictable and no correction can be made.

This example serves only to illustrate how the error correction could be carried out. But it would not be practical and a reduction in step size appears to be the best correction as we will see in the examples.

As stated above, the method appears to be divergent in nature, so our approximate solution will look like Fig. 4.3. In this figure, note that for any integration after the first one, the set of approximating lines diverge from the exact

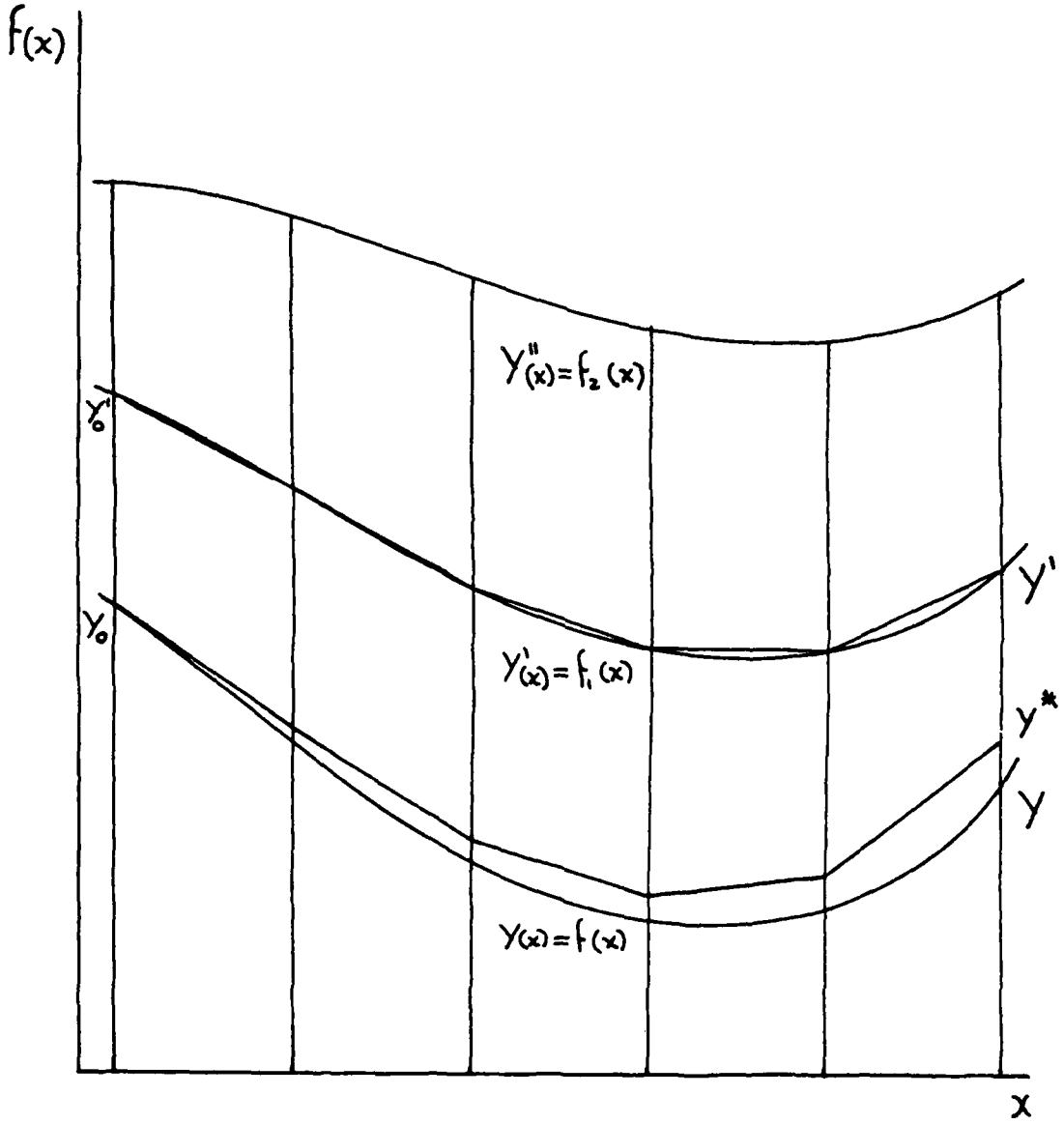


FIGURE 4.3

solution, the further we move to the right, the greater the error introduced. In dealing with I.V.P.'s, the points Y'_o and Y_o are known and these are our starting conditions. The solution will appear then as shown in this figure. Since we do not know the exact end points at the right hand side, namely points Y' and Y, there remains an uncertainty in the accuracy of the solution.

For B.V.P.'s, however, the other extreme points are known. Referring to Fig. 4.3, suppose we are given as boundary conditions, points Y'_o and Y_o in the lower curve of this figure. The method we are dealing with requires that we know the initial points to be able to start the integrating algorithm. If we know the exact starting points, namely Y'_o and Y_o, then by using these initial conditions, our solution will appear as shown in Fig. 4.3. That is to say we will not end up at point Y which is exact, but at point Y*. To be able to reach the exact point Y, which we assume is known, we need to give the algorithm a "wrong" starting point Y'^*_o. It is this approximate starting point which we find using the relationships given by equations (3.13) to (3.22) that allow us to match the exact solution in the whole range as shown in Fig. 4.4 in the lower curve.

Note that if we were using corrected relationships we would be able to supply the algorithm with "exact" starting values, and still match the exact end points. Unfortunately, it is now the upper set of solution lines that absorb the

EXAMPLE 2

CLAMPED-ROLLER BEAM

BOUNDARY CONDITIONS: Y(0)=0.0 Y(1)=0.0

Y (0)=0.0 M(1)=0.0

X:	SHEAR	MOMENT	SLOPE	DEFLECTION	EXACT DEF.
0.00	11.212190	-1.212193	0.000000	0.000000	0.000000
0.05	11.212190	-0.651584	-0.046594	-0.001280	-0.001327
0.10	11.212190	-0.090975	-0.065158	-0.004190	-0.004374
0.15	11.212190	0.469634	-0.055692	-0.007327	-0.007734
0.20	11.212190	1.030243	-0.018195	-0.009290	-0.009999
0.25	11.212190	1.590853	0.047333	-0.008677	-0.009766
0.30	11.212190	2.151463	0.140890	-0.004087	-0.005624
0.35	11.212190	2.712071	0.262479	0.005882	0.003828
0.40	11.212190	3.272680	0.412098	0.022630	0.020000
0.45	11.212190	3.833291	0.589747	0.047561	0.044297
0.50	-988.787500	-3.106097	0.764177	0.081965	0.078125
0.55	11.212190	-5.045486	0.504137	0.113698	0.110390
0.60	11.212190	-4.484878	0.265877	0.132833	0.130000
0.65	11.212190	-3.924269	0.055649	0.140755	0.138359
0.70	11.212190	-3.363659	-0.126550	0.138867	0.136875
0.75	11.212190	-2.803050	-0.280717	0.128570	0.126953
0.80	11.212190	-2.242440	-0.406855	0.111265	0.110000
0.85	11.212190	-1.681830	-0.504961	0.088354	0.087422
0.90	11.212190	-1.121222	-0.575037	0.061239	0.060625
0.95	11.212190	-0.560613	-0.617084	0.031319	0.031016
1.00	11.212190	-0.000003	-0.631098	0.000000	0.000000

All the terms involved in these expressions are defined by the relations (3.13) to (3.22), and they can be computed in advance since they are independent of boundary conditions. These initial values are only approximations as we know, but serve to match the given B.C.'s at the other extreme. The results are:

Shear:	Y_o'''	=	11.21219	Exact	= 11.25
Moment:	Y_o''	=	-1.21219	Exact	= -1.25

Running the program with these initial values, we get the results shown on the next page. The deflection appears accurate to about 2 percent for the more meaningful values of deflection. However, if we reduce the distance between the opposite forces, we improve the accuracy. But if we increase it, the discrepancies grow enormously even for a very small increment, and it does not reflect a concentrated moment behavior.

The slope shows the expected behavior. However, we should not expect high accuracy since we have started the algorithm with approximate starting points of shear and moment. At x = 1.0, we have Y_n = -0.631. The exact is -0.625. Preceding values of slope are expected to be better since here we start with an exact known initial point. Similarly for the moment we start with an approximation but get more accurate

Shear:	Y_o'''	=	Unknown	Y_n'''	=	Unknown
Moment:	Y_o''	=	Unknown	Y_n''	=	0
Slope:	Y_o'	=	0	Y_n'	=	Unknown
Deflection:	Y_o	=	0	Y_n	=	0

To start the algorithm we need to determine the initial shearing force Y_o''' and the initial bending moment Y_o''. From equations (3.13) to (3.16), after replacing the known values we get:

$$0 = hS_2 + nhY_o''' + Y_o''$$

$$Y_n' = h^2S_1 + N_1h^2Y_o''' + nhY_o''$$

$$0 = h^3S_o + N_oh^3Y_o''' + N_1h^2Y_o''$$

This is a set of three simultaneous algebraic equations with three unknowns and we solve for the initial conditions. The results are:

$$Y_o'' = \frac{h(N_oS_2 - nS_o)}{nN_1 - N_o} \tag{5.4}$$

$$Y_o''' = \frac{S_o - N_1S_2}{nN_1 - N_o} \tag{5.5}$$

In the current example we use M = 10 and it is decomposed as follows:

$$10 = 1000 \times 0.01$$

a pair of opposite forces of 1000 units acting 0.01 units of length apart from each other. In our problem we use 200 intervals of 0.005 each, one concentrated force is located at x = 0.495, and the other opposite force of equal magnitude is placed at x = 0.505. So the loading function to be integrated is defined by 5 partial functions as follows:

$$f_1(x) = 0 \qquad 0 \leq x < 0.495$$

$$\int f_2(x)\,dx = -1000 \qquad x = 0.495$$

$$f_3(x) = 0 \qquad 0.495 < x < 0.505$$

$$\int f_4(x)\,dx = 1000 \qquad x = 0.505$$

$$f_5(x) = 0 \qquad 0.505 < x \leq 1.0$$

c. Necessary Initial Conditions

We need to consider now the B.C.'s. For the present configuration we have:

$$m_i = 0 \quad \text{for} \quad x \neq x_i$$

$$m_i = \frac{P}{h} \quad \text{for} \quad x = x_i$$

$$\int f(x)\,dx = P \quad \text{for} \quad x = x_i$$

where P is the concentrated force acting at point x_i.

b. Concentrated Moments

A concentrated moment can be treated in a similar way if we use the second order differential equation. However, in our case we need to decompose the given moment into a pair of concentrated forces of equal magnitude acting in opposite directions at equidistant points from the location of the given moment. From statics we know that:

$$\text{Moment} = \text{Force} \times \text{Distance}$$

Here we can have several combinations of force times distance provided we keep this product a constant equal to the given concentrated moment. However as we will see later, accuracy is achieved only when we use very small distances and consequently very large concentrated forces. In this way of moment decomposition, the usual behavior of a concentrated moment is achieved. For larger distances the couple is not an accurate representation of a concentrated moment and the results are not so accurate.

that previous values of all other columns are even more accurate.

3. Example 2: Clamped-Roller Beam

Refer to Fig. 5.1b. This is a statically indeterminate beam loaded by a moment of 10 units of weight x unit of length, acting at the middle of the beam. As before we are going to determine the shear, moment, slope and deflection of the beam.

4. Solution

This example will emphasize three things: How to deal with a) concentrated forces, b) concentrated moments and c) the necessary initial conditions required for this particular case.

a. Concentrated Forces

We can conceive of a concentrated force as a distributed force of high intensity distributed over a very small length of beam. Recall from the previous chapters that the slope of any approximating line at any point i is given by (2.9) where f(x) is the distributed load for the case of beams. In this equation the integral in the numerator represents the area under the f(x) curve. This area is actually the total load in the specified interval. By decreasing the interval while still holding the same inside area constant (same total load), the distributed force tends to a concentrated force of equal magnitude as the total load. In the limit it becomes a concentrated force acting on a mathematical point on the beam. For our purpose, in the integrating algorithm we simply let:

EXAMPLE 1 .

CLAMPED-FREE BEAM

BOUNDARY CONDITIONS: Y(0)= 0.0000 Y (0)= 0.0000

M(0)= 0.3750 V(0)= -0.5000

X:	SHEAR	MOMENT	SLOPE	DEFLECTION	EXACT DEF.
0.00	-0.500000	0.375000	0.000000	0.000000	-0.000000
0.05	-0.500000	0.350000	0.018125	0.000458	0.000458
0.10	-0.500000	0.325000	0.035000	0.001792	0.001792
0.15	-0.500000	0.300000	0.050625	0.003937	0.003938
0.20	-0.500000	0.275000	0.065000	0.006833	0.006833
0.25	-0.500000	0.250000	0.078125	0.010416	0.010417
0.30	-0.500000	0.225000	0.090000	0.014625	0.014625
0.35	-0.500000	0.200000	0.100625	0.019395	0.019396
0.40	-0.500000	0.175000	0.110000	0.024666	0.024667
0.45	-0.500000	0.150000	0.118125	0.030375	0.030375
0.50	-0.500000	0.125000	0.125000	0.036458	0.036459
0.55	-0.450000	0.101250	0.130646	0.042854	0.042855
0.60	-0.400000	0.080000	0.135167	0.049504	0.049505
0.65	-0.350000	0.061250	0.138688	0.056354	0.056355
0.70	-0.300000	0.045000	0.141334	0.063358	0.063358
0.75	-0.250000	0.031250	0.143230	0.070475	0.070475
0.80	-0.200000	0.020000	0.144501	0.077670	0.077671
0.85	-0.150000	0.011250	0.145272	0.084916	0.084916
0.90	-0.100000	0.005000	0.145667	0.092191	0.092191
0.95	-0.050000	0.001249	0.145813	0.099479	0.099479
1.00	-0.000000	-0.000001	0.145834	0.106770	0.106770

from the given configuration. Here the function f(x) to be integrated is discontinuous and given by a set of two different functions defined by:

$$f_1(x) = 0 \qquad 0 \leq x < 0.5$$

$$f_2(x) = 1.0 \qquad 0.5 \leq x \leq 1.0$$

We need to determine the initial values to start the algorithm. From statics, it can easily be shown that the initial conditions are as follows:

Shear:	Y_o'''	=	-0.5	Y_n'''	= 0.0
Moment:	Y_o''	=	0.375	Y_n''	= 0.0
Slope:	Y_o'	=	0.0	Y_n'	= Unknown
Deflection:	Y_o	=	0.0	Y_n	= Unknown

By supplying these initial values to the program, we obtain the computer output shown on the next page. The last column shows the exact deflections. In these results, note that all the initial values are exact, and that the last value found, namely the deflection of the beam at x = 1.0, is expected to have the biggest cumulative error. However, as we can see, we get 3 to 4 digits accuracy. Here we can conclude

The computer program used to solve these problems is shown at the end of this chapter, and it is a straightforward coding of equations (3.13) to (3.22). This programming technique, in particular, tries to minimize round-off errors. For now, it suffices to say that all problems are solved using SINGLE PRECISION, one hundred intervals (problem 2 uses 200), and we get accuracy to the fourth and even to the fifth decimal places in some examples. Finally, only the fourth integration, the deflection, is compared with the exact solution. Recall that the error is cumulative, and it is here that we expect the bigger error. We may conclude then that preceding integrations are more exact. However, we should keep in mind that an approximated starting point is needed in some cases, and the results will be shifted by some amount from the exact. We will see this as we proceed into the next section.

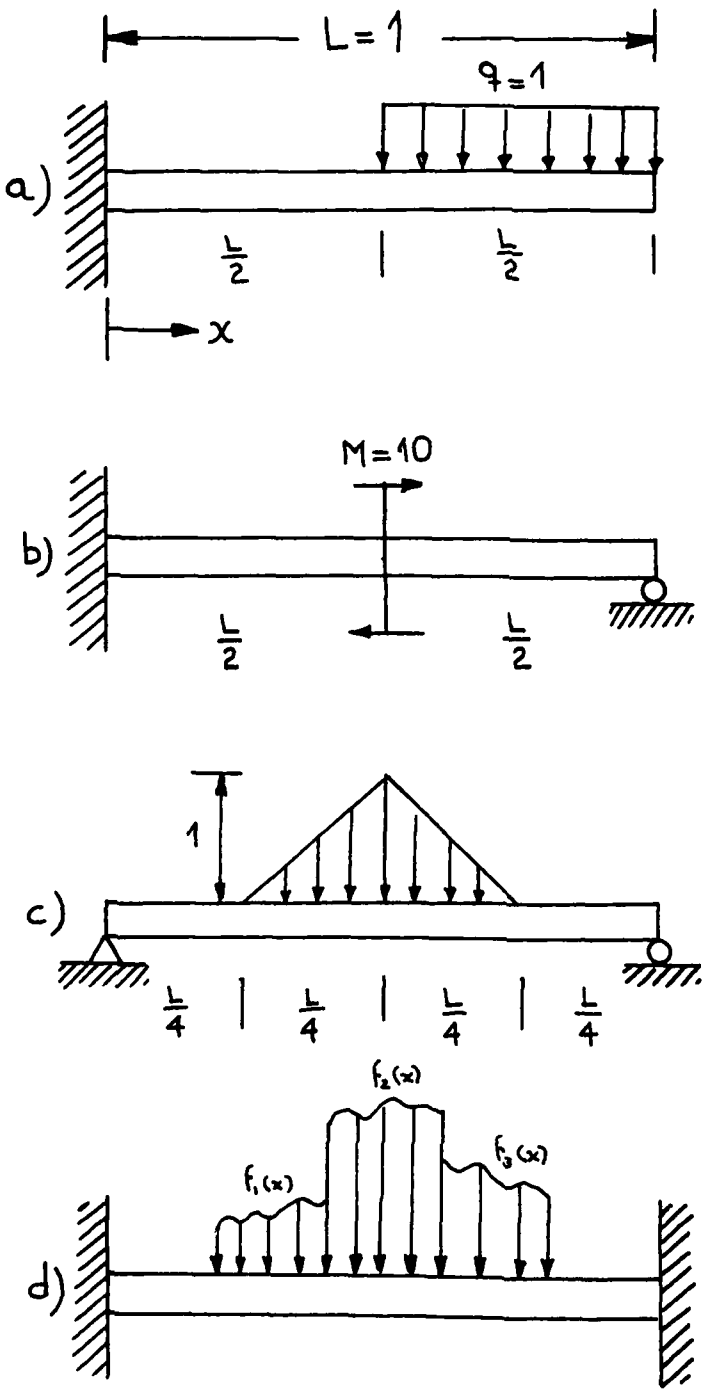
B. EXAMPLES

1. Example 1: Clamped-Free Beam

Refer to Fig. 5.1a. This is a statically determinate cantilever beam loaded over half its length by a uniformly distributed load of 1 unit of weight per unit of length as shown. We shall determine the shearing force, bending moment, slope and deflection at discrete points of the beam.

2. Solution

This is a B.V.P. but for purpose of illustrating how to use the method when dealing with I.V.P.'s, we will treat it as such. All the required initial conditions can be drawn



B.C.

$Y_o''' = -.5$
$Y_o'' = .375$
$Y_o' = .0$
$Y_o = .0$

$Y_o' = .0$
$Y_o = .0$
$Y_n'' = .0$
$Y_n = .0$

$Y_o'' = .0$
$Y_n'' = .0$
$Y_o = .0$
$Y_n = .0$

$Y_o' = .0$
$Y_n' = .0$
$Y_o = .0$
$Y_n = .0$

FIGURE 5.1

a given problem could be solved by integrating the moment second order differential equation. By doing this, we would be able to achieve more accuracy since only two integrations need to be performed and the accumulated error will be relatively small.

However, when the loading of the beam is a complicated distribution and the expression for the bending moment is difficult to obtain, then the fourth order differential equation will be the one to use. One advantage in using this expression is that the integrating algorithm will provide results for the shear force, bending moment, slope and deflection simultaneously. When using the second order equation we only get slope and deflection.

Since eq. (5.1) is a more general relationship, we are going to use this expression in the examples. In fact, most problems can be expressed in that way. Recall that a concentrated force can be treated as a distributed force acting in a very small interval, and a concentrated moment reduces to a couple of concentrated loads acting in opposite directions a small distance apart. Example 2 illustrates this procedure.

The examples provided in this chapter account for all types of B.C.'s. Fig. 5.1 shows the 4 cases to be treated, and we shall study them one by one in the next subsections. In all cases the flexural rigidity EI and the length L is set equal to 1, for simplicity. Discontinuous types of loading are emphasized, and of course, many combinations of loadings can be dealt with by simply using the principle of superposition.

V. APPLICATIONS AND RESULTS

A. GENERAL

In this chapter we are going to study a particular application of the integrating algorithm introduced in the previous sections, namely, the deflection of beams, where a fourth order linear differential equation occurs, and several combinations of B.C.'s can be considered.

From Mechanics of Solids, the relationships governing the deflection of beams are given by:

$$EIY^{'v}(x) = q(x) \quad \text{(load intensity)} \tag{5.1}$$

$$EIY'''(x) = V(x) \quad \text{(shear force)} \tag{5.2}$$

$$EIY''(x) = M(x) \quad \text{(moment)} \tag{5.3}$$

$$Y'(x) = \text{slope}$$

$$y(x) = \text{deflection}$$

The method we are dealing with is basically a method of successive integration. Because of that we [illegible] start to integrate from any of the above relationships provided we know explicitly the right hand side of the equation. In some cases,

inherent error of the method since we have given them approximate initial values. The solution now is going to look like Fig. 4.4. The lower solution becomes "exact" but the upper actually shifts apart from the exact. We can explain this by saying that the known B.C.'s are fixed by the algorithm while the free (unknown) ones will take the error, and this is exactly what happens in the examples in the next section. In summary, all we do is to shift the error from one place to another. We can not get rid of it unless we use corrected relationships or an infinite number of intervals to minimize the local error.

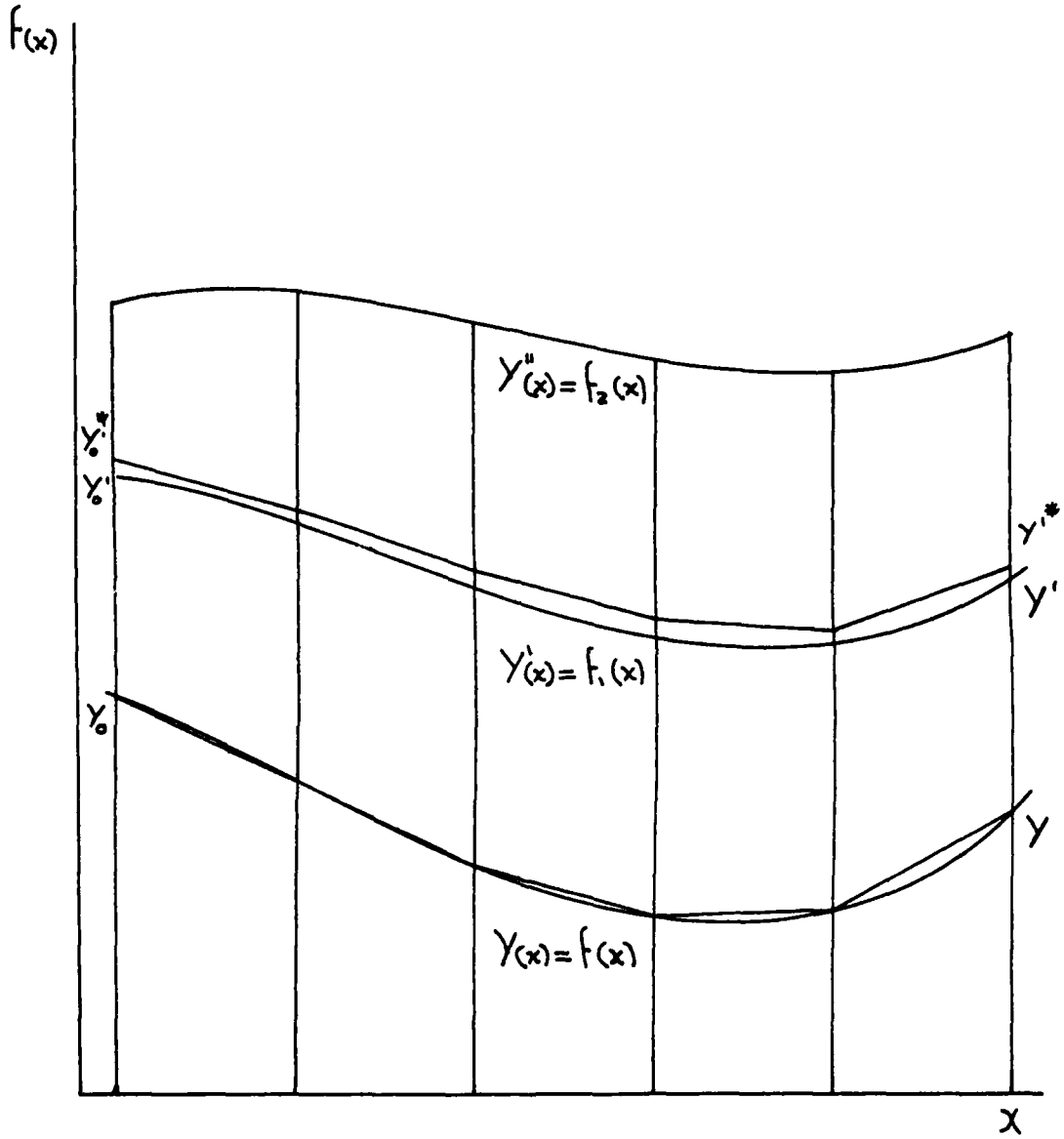


FIGURE 4.4

as we proceed, since we end up with the expected result of 0.0 at x = 1.0.

In the case of the shear force, we see it is constant throughout the beam length as expected but it is shifted by an amount of 0.0378 from the exact. It is interesting to note that at the point of the loading we have a very strong shearing force. For a pure moment acting at this place, we shall neglect this result. However, if we do have an actual couple than a big shearing at this point should be expected. Note that the algebraic sum of shear forces gives:

$$11.21219 - (-988.7875) = 1000.0$$

which is indeed the couple acting at this point.

5. Example 3: *Pin-Roller Beam*

Refer to Fig. 5.1c. This is a statically determinate simple supported beam subjected to a triangular type of load distributed over the central portion of the beam as indicated in the figure. We need to determine the shear, moment, slope and deflection of the beam.

6. Solution

First of all, some comments about this configuration are in order. This example was chosen in order to emphasize the ease with which we can switch from one kind of load distribution to another. As explained at the beginning of this work, it is the fact that the algorithm solves a "new" differential equation at every single step that allows us to deal

with different loads. Therefore it is possible to have a different kind of loading per subdivision. Another reason for selecting this problem is that here we deal with linear forcing functions and we are going to perform four successive integrations of a linear function. This will lead to a polynomial of fifth degree and a bigger cumulative error is to be expected.

Proceding with the solution, the distributed load f(x) to be integrated is given by four different functions:

$$f_1(x) = 0 \qquad 0 \leq x < 0.25$$

$$f_2(x) = 4x-1 \qquad 0.25 \leq x < 0.50$$

$$f_3(x) = -4x+3 \qquad 0.50 \leq x < 0.75$$

$$f_4(x) = 0 \qquad 0.75 \leq x \leq 1.0$$

As indicated above, the computer program is such that it can switch from one kind of loading to the other at the specified point. Now, in order to start the algorithm we need to determine the initial points from the B.C.'s of a simple supported beam. These are:

Shear:	$Y_o''' =$ Unknown	$Y_n''' =$ Unknown
Moment:	$Y_o'' = 0$	$Y_n'' = 0$
Slope:	$Y_o' =$ Unknown	$Y_n' =$ Unknown
Deflection:	$Y_o = 0$	$Y_n = 0$

Now, from the known relationships after we substitute in the given values, we get:

$$0 = hS_2 + nhY_o'''$$

$$0 = h^3S_o + N_oh^3Y_o''' + nhY_o'$$

This is a pair of simultaneous algebraic equations in two unknowns. We solve for Y_o''' and Y_o'. The results are:

$$Y_o' = (\frac{h}{n})^2(N_oS_2 - nS_o) \tag{5.6}$$

$$Y_o''' = -\frac{S_2}{n} \tag{5.7}$$

Similarly, the computer supplies the values of the variables involved in these relations and these are as follows:

Shear:	Y_o'''	= -0.12499	Exact	=	-0.125
Moment:	Y_o'	= 0.014972	Exact	=	0.014974

which shows a remarkable accuracy. Running the program with these initial values we obtain the results shown on the next page. A comparison between the approximate deflection and the exact reveals an error of about 2 percent at the point where the largest deflection takes place. In this example, we expected a larger error, but the algorithm appears to perform

EXAMPLE 3

PIN-ROLLER BEAM

BOUNDARY CONDITIONS: Y(0)=0.0 Y(1)=0.0

M(0)=0.0 M(1)=0.0

X:	SHEAR	MOMENT	SLOPE	DEFLECTION	EXACT DEF.
0.00	-0.124999	0.000000	0.014972	0.000000	0.000000
0.05	-0.124999	-0.006250	0.014816	0.000746	0.000746
0.10	-0.124999	-0.012500	0.014347	0.001476	0.001477
0.15	-0.124999	-0.018750	0.013566	0.002175	0.002176
0.20	-0.124999	-0.025000	0.012472	0.002827	0.002828
0.25	-0.124999	-0.031250	0.011066	0.003417	0.003418
0.30	-0.119999	-0.037415	0.009348	0.003929	0.003920
0.35	-0.104999	-0.043080	0.007333	0.004347	0.004322
0.40	-0.079999	-0.047745	0.005057	0.004658	0.004609
0.45	-0.044999	-0.050910	0.002534	0.004849	0.004766
0.50	0.000000	-0.052075	0.000000	0.004914	0.004785
0.55	0.045000	-0.050910	-0.002584	0.004849	0.004766
0.60	0.080000	-0.047745	-0.005057	0.004658	0.004609
0.65	0.105000	-0.043080	-0.007333	0.004347	0.004322
0.70	0.120000	-0.037415	-0.009348	0.003929	0.003920
0.75	0.125000	-0.031250	-0.011066	0.003417	0.003418
0.80	0.125000	-0.025000	-0.012472	0.002827	0.002828
0.85	0.125000	-0.018750	-0.013565	0.002175	0.002176
0.90	0.125000	-0.012500	-0.014347	0.001476	0.001477
0.95	0.125000	-0.006250	-0.014816	0.000746	0.000746
1.00	0.125000	0.000000	-0.014972	-0.000000	0.000000

fairly well even in this case. Nevertheless, the error should be expected at the "free" (unknown) B.C.'s since the other extremes are fixed. Here we make use of the fact that the loading is symmetrical and the results should be symmetrical too. This is actually the case in the computer printout, so we can conclude that the results are indeed accurate.

7. Example 4: Clamped-Clamped Beam

Refer to Fig. 5.1d. This is a statically indeterminate beam clamped at both ends and loaded with a totally arbitrary discontinuous load known only as a table of values at discrete points as shown on the next page. We shall determine the shear, moment, slope and deflection of the beam.

8. Solution

Before we proceed with this example, let us remind ourselves that we are not restricted to handle only exact-integrable forcing functions. We can deal with arbitrary functions as well, and this example intends to illustrate the procedure. In this problem, the computer program reads in data from a table of values instead of obtaining them by evaluating a given function. With this data the program performs trapezoidal integration in the usual way. The forcing function is defined by three other functions as:

$$f_1(x) = 0 \qquad 0 \leq x < 0.3$$

$$f_2(x) = \text{(from data deck)} \qquad 0.3 \leq x \leq 0.7$$

$$f_3(x) = 0 \qquad 0.7 < x \leq 1.0$$

DATA DECK FOR EXAMPLE 4

DISTANCE	LOAD
0.30	0.000000
0.31	0.000000
0.32	0.000000
0.33	0.000000
0.34	0.000000
0.35	0.000000
0.36	0.000000
0.37	0.000000
0.38	0.000000
0.39	0.000000
0.10	0.000000
0.11	0.000000
0.12	0.000000
0.13	0.000000
0.14	0.000000
0.15	0.000000
0.16	0.000000
0.17	0.000000
0.18	0.000000
0.19	0.000000
0.20	0.000000
0.21	0.000000
0.22	0.000000
0.23	0.000000
0.24	0.000000
0.25	0.000000
0.26	0.000000
0.27	0.000000
0.28	0.000000
0.29	0.000000
0.30	1.000000
0.31	1.600000
0.32	1.300000
0.33	2.000000
0.34	2.200000
0.35	2.800000
0.36	3.200000
0.37	3.000000
0.38	3.400000
0.39	3.600000
0.40	5.000000
0.41	5.300000
0.42	6.200000
0.43	6.600000
0.44	6.700000
0.45	6.400000
0.46	6.200000
0.47	7.000000
0.48	6.900000
0.49	6.300000
0.50	6.100000
0.51	6.000000
0.52	5.300000
0.53	6.200000
0.54	6.600000
0.55	6.700000
0.56	6.400000
0.57	6.200000
0.58	7.000000
0.59	6.900000
0.60	6.300000
0.61	4.100000
0.62	4.000000
0.63	4.600000
0.64	3.800000
0.65	4.000000
0.66	3.200000
0.67	3.800000
0.68	3.200000
0.69	3.400000
0.70	3.000000
0.71	0.000000
0.72	0.000000
0.73	0.000000
0.74	0.000000
0.75	0.000000
0.76	0.000000
0.77	0.000000
0.78	0.000000
0.79	0.000000
0.80	0.000000
0.81	0.000000
0.82	0.000000
0.83	0.000000
0.84	0.000000
0.85	0.000000
0.86	0.000000
0.87	0.000000
0.88	0.000000
0.89	0.000000
0.90	0.000000
0.91	0.000000
0.92	0.000000
0.93	0.000000
0.94	0.000000
0.95	0.000000
0.96	0.000000
0.97	0.000000
0.98	0.000000
0.99	0.000000
1.00	0.000000

Unfortunately, no exact solution has been obtained for this case and we are not able to compare the computer results with the exact. Nevertheless some conclusions can be drawn. The B.C.'s for a clamped-clamped beam are as follows:

Shear:	Y_o'''	=	Unknown	Y_n'''	=	Unknown
Moment:	Y_o''	=	Unknown	Y_n''	=	Unknown
Slope:	Y_o'	=	0	Y_n	=	0
Deflection:	Y_o	=	0	Y_n	=	0

To determine the initial conditions, we use the known relationships and substituting in the known values, we obtain:

$$0 = h^2 S_1 + N_1 h^2 Y_o''' + nhY_o''$$

$$0 = h^3 S_o + N_o h^3 Y_o''' + N_1 h^2 Y_o''$$

This is a pair of simultaneous algebraic equations with two unknowns. Solving for the initial conditions we get:

$$Y_o'' = \frac{n(N_o S_2 - nS_o)}{nN_1 - N_o} \tag{5.8}$$

$$Y_o''' = \frac{S_o - N_1 S_2}{nN_1 - N_o} \tag{5.9}$$

For this example the values supplied by the computer are:

Shear: $Y_o''' = -0.948411$

Moment: $Y_o'' = 0.230507$

The computer printout is shown on the next page. Here we have another column to show the arbitrary loading. Note that we are using 100 intervals, but actually every fifth is printed.

Perhaps the only way to analyze these results would be to check if the B.C.'s have been met or not. The results show that they have. Another clue to assure some accuracy would be to look at the maximum deflection and the minimum slope. The loading was intentionally concentrated on the middle of the beam so we can expect the highest deflection and minimum slope in this neighborhood since we have similar B.C.'s at both ends. We can say that this too checks. Consequently we see that the results are likely to be trusted.

EXAMPLE 4

CLAMPED-CLAMPED BEAM

BOUNDARY CONDITIONS: Y(0)=0.0 Y(1)=0.0

Y'(0)=0.0 Y'(1)=0.0

X:	LOAD	SHEAR	MOMENT	SLOPE	DEFLECTION
0.00	0.000000	-0.948411	0.230507	0.000000	0.000000
0.05	0.000000	-0.948411	0.183086	0.010340	0.000268
0.10	0.000000	-0.948411	0.135666	0.018309	0.000994
0.15	0.000000	-0.948411	0.088245	0.023906	0.002059
0.20	0.000000	-0.948411	0.040825	0.027133	0.003344
0.25	0.000000	-0.948411	-0.006596	0.027939	0.004732
0.30	1.000000	-0.948411	-0.053991	0.026474	0.006103
0.35	2.000000	-0.848411	-0.099067	0.022628	0.007339
0.40	3.000000	-0.672411	-0.137487	0.016680	0.008330
0.45	4.000000	-0.357411	-0.163428	0.009094	0.008979
0.50	5.000000	-0.030911	-0.173116	0.000614	0.009224
0.55	6.000000	0.279086	-0.167032	-0.007952	0.009039
0.60	5.000000	0.609085	-0.144953	-0.015819	0.008440
0.65	4.000000	0.825585	-0.108841	-0.022207	0.007483
0.70	3.000000	0.996580	-0.063138	-0.026542	0.006255
0.75	0.000000	1.011580	-0.012634	-0.028440	0.004870
0.80	0.000000	1.011580	0.037894	-0.027810	0.003454
0.85	0.000000	1.011580	0.088472	-0.024651	0.002132
0.90	0.000000	1.011580	0.139050	-0.019963	0.001032
0.95	0.000000	1.011580	0.189629	-0.010746	0.000279
1.00	0.000000	1.011580	0.240207	0.000000	-0.000000

C. THE COMPUTER PROGRAM

```
FILE: SLOPE     WATFIV    A1

$JOB                XREF
C*****************************************************
C***************  THE COMPUTER PROGRAM  ******************
C*****************************************************
C
C THIS PROGRAM CALCULATES THE SHEARING FORCE, BENDING MOMENT, SLOPE AND
C DEFLECTION OF BEAMS BY PERFORMING FOUR SUCCESIVE INTEGRATIONS ON
C THE LOADING FUNCTION. THE FOLLOWING CONFIGURATIONS ARE CONSIDERED:
C         1.-CANTILEVER (CLAMPED-FREE) BEAM
C         2.-CLAMPED-ROLLER BEAM
C         3.-SIMPLE SUPORTED (PIN-ROLLER) BEAM
C         4.-CLAMPED-CLAMPED BEAM
C THE LOADING CAN BE ANY NUMBER OF DISTRUBUTED AND/OR CONCENTRATED
C FORCES. CONCENTRATED MOMENTS CAN BE DECOMPOSED INTO A COUPLE AT THE
C MINIMUN POSSIBLE DISTANCE ACTING AT THE POINT OF APPLICATION OF THE
C MOMENT.
C
C THE PROGRAM CONSISTS OF THREE ROUTINES:
C 1. SUBROUTINE SLOPE
C 2. SUBROUTINE INCOM
C 3. SUBROUTINE LOAD
C WHICH ARE DEFINED IN THE FOLLOWING PARAGRAPHS
C
C******** MAIN PROGRAM ***************************
C
C READS IN NUMBER OF STEPS DEFINED BY THE INTEGER VARIABLE N
C CALLS SUBROUTINE INCOM FOR INITIAL CONDITIONS
C CALLS SUBROUTINE SLOPE FOR FINAL RESULTS
C
C******** SUBROUTINE INCOM ************************
C
C READS IN THE INTEGER VARIABLE K WHICH SPECIFIES THE KIND OF PROBLEM
C WITH THE FOLLOWING CODE:
C    K=1 CANTILEVER BEAM (INITIAL CONDITIONS MUST BE SPECIFIED)
C    K=2 CLAMPED ROLLER BEAM
C    K=3 CLAMPED-CLAMPED BEAM
C    K=4 SIMPLE SUPPORTED BEAM
C
C    THE PARAMENTERS ARE DEFINED AS FOLLOWS:
C              CALL INCON(N,Y0,Y10,Y20,Y30)
C    N  = NUMBER OF STEPS
C Y0    = INITAL DEFLECTION
C Y10   = INITIAL SLOPE
C Y20   = INITIAL BENDING MOMENT
C Y30   = INITIAL SHERING FORCE
C
C THIS SUBROUTINE SUPPLIES THE INITIAL CONDITIONS BY CALLING SLOPE.
C FOR A CANTILEVER BEAM THIS STATEMENT IS NOT EXECUTED
C
C******** SUBROUTINE SLOPE ************************
C PERFORM TWO TASKS: 1. CALCULATES AND SUPPLY  INCOM WITH THE NECESARY
C                       PARAMETERS  TO DETERMINE INITIAL CONDITIONS
C                    2. EXECUTES THE SOLUTION OF THE PROBLEM
C
C THIS SUBROUTINE IS A THE CODING OF THE RELATIONSHIPS GIVEN BY
C EQUATIONS 3.13 TO 3.32. THE SUMMATION TERMS ARE CONTAINED IN
C THE COMMON STATEMENT
C
C THE CALLING PARAMETERS ARE DEFINED AS FOLLOWS:
C              CALL SLOPE(N,Y0,Y10,Y20,Y30,J)
C    N = SAME AS IN INCOM
C Y0   = "
C Y10  = "
C Y20  = "
C Y30  = "
C    J = TAKE TWO VALUES: J=0: TO FIND INITIAL CONDITIONS ONLY
C                         J=1: TO EXECUTE THE SOLUTION
C
C******** SUBROUTINE LOAD ************************
C
C SUPPLIES SLOPE WITH THE NECESARY INFORMATION OBTAINED FROM THE
C LOADING FUNCTION. IT CAN BE AN ACTUAL FUNCTION OR A DATA DECK.
```

```
FILE: SLOPE     WATFIV    A1

C MUST BE WRITTEN SPECIFICALLY FOR EACH NEW PROBLEM. WHEN DEALING
C WITH CONCENTRATED LOADS, THE VARIABLE RINT IN ROUTINE SLOPE IS
C IS SET EQUAL TO THE CONCENTRATED LOAD DIRECTLY, AT THE POINT OF
C ACTION.
C THE CALLING PARAMETERS ARE DEFINED AS FOLLOWS
C            CALL LOAD(B,H2)
C     B= CURRENT VALUE OF THE VARIABLE X (LENGHT) GIVEN BY ROUTINE SLOPE
C    H2= CONTAINS THE VALUE OF THE LOADING FUNCTION AT X=B
C
C*****************************************
C     MAIN PROGRAM
C*****************************************
      REAL Y0,Y1C,Y2C,Y3C
      INTEGER J,N
      COMMON S0,S1,S2,SN0,SN1,RN,H
      READ(5,10)N
   10 FORMAT(I3)
      H=1./FLOAT(N-1)
      RN=FLOAT(N-1)
      CALL INCON(N,Y0,Y1C,Y2C,Y3C)
      J=0
      CALL SLOPE(N,Y0,Y1C,Y2C,Y3C,J)
      STOP
      END
C
C*****************************************
      SUBROUTINE LOAD(B,H2)
C*****************************************
      REAL B,H2
      F(X)=....
      H2=F(B)
      RETURN
      END
C
C*****************************************
      SUBROUTINE INCON(N,Y0,Y1C,Y2C,Y3C)
C*****************************************
      REAL Y0,Y1C,Y2C,Y3C
      INTEGER J,K,N
      COMMON S0,S1,S2,SN0,SN1,RN,H
      READ(5,10)K,Y0,Y1C,Y2C,Y3C
   10 FORMAT(I1,4F10.4)
      IF((K.EQ.0).OR.(K.GT.4)) GO TO 60
      IF(K.EQ.1) GO TO 50
      J=1
      CALL SLOPE(N,Y0,Y1C,Y2C,Y3C,J)
      IF(K-3)20,30,40
C*****************************************
C     CLAMPED-ROLLER
C*****************************************
   20 WRITE(6,25)
   25 FORMAT('1','EXAMPLE 2'//1X,'CLAMPED-ROLLER BEAM'//1X,'BOUNDARY CON
     *DITIONS:       Y(0)=0.0     Y(1)=0.0'//27X,'Y (0)=0.0    M(1)=0.0'//)
      Y2C=H*(S2*SN0-RN*S0)/(RN*SN1-SN0)
      Y3C=  (S0-SN1*S2)/(RN*SN1-SN0)
      RETURN
C*****************************************
C     CLAMPED-CLAMPED
C*****************************************
   30 WRITE(6,35)
   35 FORMAT('1','EXAMPLE 4'//1X,'CLAMPED-CLAMPED BEAM'//1X,'BOUNDARY CO
     *NDITIONS:     Y(0)=0.0     Y(1)=0.0'//25X,'Y (0)=0.0    Y (1)=0.0'//)
      Y2C=H*(S0*SN1-S1*SN0)/(RN*SN0-SN1*SN1)
      Y3C=  (S1*SN1-RN*S0)/(RN*SN0-SN1*SN1)
      RETURN
C*****************************************
C     PIN-ROLLER
C*****************************************
   40 WRITE(6,45)
   45 FORMAT('1','EXAMPLE 3'//1X,'PIN-ROLLER BEAM'//1X,'BOUNDARY CONDITI
     *ONS:       Y(0)=0.0     Y(1)=0.0'//27X,'M(0)=0.0     M(1)=0.0'//)
      Y1C=((H/RN)**2)*(SN0*S2-RN*S0)
```

```
      Y30= -S2/RN
      RETURN
C*****************************************
C     CLAMPED-FREE
C*****************************************
   50 WRITE(6,55)Y0,Y10,Y20,Y30
   55 FORMAT('1','EXAMPLE 1'//1X,'CLAMPED-FREE BEAM'  //1X,'BOUNDARY CON
     *DITIONS:       Y(0)=',F8.4,4X,'Y (0)=',F8.4,4X,//27X, 'M(0)=',F8.4,
     *4X,'V(0)=',F8.4//)
      GO TO 90
C*****************************************
   60 WRITE(6,65)
   65 FORMAT(1X,'N FROM 1 TO 4 ONLY'/)
   90 RETURN
      END
C
C*****************************************
      SUBROUTINE SLOPE(N,Y0,Y10,Y20,Y30,J)
C*****************************************
      REAL SU00,SU01,SU02,SU03,SU11,SU12,SU13,SU22,SU23,SU33,RINT,
     *RINT1,RINT2,RINT3,H1,H2,RI,B,SNM1,SNM2
      INTEGER I,J,K,L,N,M,NS1K,NSOL,NS1L,IM1
      COMMON S0,S1,S2,SN0,SN1,RN,H
      SU00=0.
      SU01=0.
      SU02=0.
      SU03=0.
      SU11=0.
      SU12=0.
      SU13=0.
      SU22=0.
      SU23=0.
      SU33=0.
      RINT=0.
      RINT1=0.
      RINT2=0.
      RINT3=0.
      NS1K=0
      NSOL=0
      NS1L=0
      H1=0.0
      H2=0.0
      IF(J.GT.0) GO TO 100
      WRITE(6,20)
  100 DO 400 I=1,N
      IM1=I-1
      RI=FLOAT(IM1)
      B=RI*H
      CALL LOAD(B,H2)
      IF(I.EQ.1)GO TO 200
      RINT=0.5*H*(H1+H2)
  200 RINT4=RINT3
      RINT3=RINT2
      RINT2=RINT1
      RINT1=RINT
C
      SU33=SU33+RINT1
      S3  =SU33
C
      SU23=SU23+RINT2
      SU22=SU22+SU23
      S2=.5*SU33+SU22
C
      K=I-2
      IF(I.LE.2) K=0
      SU13=SU13+RINT3
      SU12=SU12+SU13
      SU11=SU11+SU12
      NS1K=NS1K+K
      SNM1=FLOAT(NS1K)
      SN1 =0.5*RI+SNM1
      S1=.25*SU33+SU22+SU11
```

```
FILE: SLOPE     WATFIV    A1

C
      L=I-3
      IF(I.LE.3) L=0
      SU03=SU03+RINT4
      SU02=SU02+SU03
      SU01=SU01+SU02
      SU00=SU00+SU01
      NS1L=NS1L+L
      NS0L=NS0L+NS1L
      SNM2=FLOAT(NS0L)
      SN0 =0.25*RI+SNM1+SNM2
      S0=.125*SU33+.75*SU22+1.5*SU11+SU00
      IF(J.GT.0) GO TO 300
C
      Y3=S3+Y30
      Y2=H*(S2+RI*Y30)+Y20
      Y1=(((S1+SN1*Y30)*H+RI*Y20)*H+Y10
      Y=(((S0+SN0*Y30)*H+SN1*Y20)*H+RI*Y10)*H+Y0
      WRITE(6,10)B,Y3,Y2,Y1,Y
  300 H1=H2
  400 CONTINUE
      IF(J.GT.0) GO TO 999
      WRITE(6,99)
   10 FORMAT(1X,F4.2,2X,4F12.6/)
   20 FORMAT(2X,'X:', 9X,'SHEAR',6X,'MOMENT',6X,'SLOPE',5X,'DEFLECTION')
   99 FORMAT('1')
  999 RETURN
      END
C
$ENTRY
```

VI. CONCLUSIONS AND RECOMMENDATIONS

Based upon the research carried out in this thesis and the results obtained, the following conclusions can be drawn:

1) First of all, the method, in the way it has been developed, shows the fact that the solutions are totally independent of each other. Any value of the unknown can be found without generating all previous ones. A closer look at equations (3.13) to (3.16) reveals that we can apply any of these relationships at any point i ($0 \leq i \leq n$) directly, provided we know the initial conditions and the summation terms which can be generated in advance. In all of these equations there is only one summation term which depends on the given function f(x), the other summations are series of integers totally independent of the given problem. This fact represents a good saving in terms of computer time if it is conveniently exploited.

2) As it has been shown in the examples, the power of the method is perhaps its ability to deal with arbitrary functions and this is important since many engineering problems lead to these kinds of functions.

3) So far, there has not been any error correction in a strict sense. The only corrective measure has been step size reduction. However, equation (4.9) shows that correction can be performed provided we know how the local error behaves.

When integrating constant and/or linear functions, (4.9) applies, but for other cases it does not. In any case, equation (4.3) indicates that the local error depends only on the second integral which in turn is one degree higher than the given function itself. This suggests the idea that if we know how the original function behaves, linearly, quadratically, etc., we could, to some extent, predict the error behavior and carry out a correction. Further research is recommended in this particular case.

Nevertheless, as shown in the examples, some accuracy has been achieved even working in single precision. For more complicated problems involving complex functions and requiring very accurate solutions, double precision is still a good possibility.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943	2
3.	Department Chairman, Code 69 Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93943	2
4.	Professor David Salinas, Code 69Zc Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93943	5
5.	LT Fernando Heredia N. Primera Zona Naval Guayaquil, Ecuador	5

END

FILMED

8-85

DTIC