The work described in this thesis was carried out by me under the supervision of Dr. Sunethra Weerakoon and Dr. G. K. Watugala, and a report on this has not been submitted to any University for another degree.

....

H. K. G. De Z Amarasekera

We certify that the above statement made by the candidate is true and that this thesis is suitable for submission to the University for the purpose of evaluation.

....
Dr. Sunethra Weerakoon

....
Dr. G. K. Watugala

DETERMINATION OF ZEROS USING FINITE DIFFERENCE NEWTON'S METHOD IN THE ABSENCE OF THE CLOSED FORM OF THE FUNCTION

By

H. K. G. DE Z AMARASEKERA

Thesis submitted to the University of Sri Jayewardenepura for the award of the degree of Master of Philosophy in Mathematics on 20th July 1999.

CONTENTS

ACKNOWLEDGMENT

I am deeply indebted to my supervisors Dr. Sunethra Weerakoon of the Department of Mathematics, University of Sri Jayewardenepura and Dr. G. K. Watugala of the Department of Mechanical Engineering, University of Moratuwa for providing me with invaluable guidance, helpful suggestions, advise and encouragement at every stage of the research process.

I wish to pay my gratitude to the staff of the Department of Mathematics and staff of the Computer centre of University of Sri Jayewardenepura. I also wish to express my sincere appreciation to the authorities of the University of Sri Jayewardenepura, for appointing me as a research assistant to perform research work for a two year period.

I am grateful to my friends for their invaluable assistance extended to me during the research process for the successful completion of this endeavour. Finally I dearly acknowledge the assistance of my family members in every stage of my educational career up to this very day of my life.

DETERMINATION OF ZEROS USING FINITE DIFFERENCE NEWTON'S METHOD IN THE ABSENCE OF THE CLOSED FORM OF THE FUNCTION

H. K. G. DE Z AMARASEKERA

ABSTRACT

Finite Difference Newton's (FDN) method is the one point iteration scheme introduced in [14] to approximate single roots of nonlinear equations. Proposed scheme replaces the derivative of the function in Newton's method by appropriately chosen forward or backward difference formulae. In this study the same method is applied to functions of two variables and later it is extended to functions of several variables. It is proved that the method is second order convergent. Computational evidence provided here, not only supports the theory but goes beyond that suggesting it is not necessary to have the initial guess within a sufficiently close neighbourhood for the convergence of the proposed method. As problems, such as looping which effect Newton's method, can be overcome with the proposed method by choosing suitable stepsizes, finite difference Newton's method provides convergent results even for functions which do not converge with Newton's iterations.

The main objective here is to determine the roots of a function whose closed form is not known, while only a discrete set of function values is available.

Of all the available interpolation techniques, natural cubic splines seem to produce best approximations for such functions in the case of functions of one variable. We adopt both FDN method introduced in [14] and the Newton's method as one point iteration

schemes, to approximate the root of the constructed spline function. As a means of testing, we apply the suggested procedure to several sets of data generated by various types of (known) functions.

Furthermore, we try to approximate the roots of two simultaneous nonlinear equations $\{ f_1(x, y) = 0 \; ; f_2(x, y) = 0\}$ using only the values of the functions in a rectangular domain $\{ D = (x_i , y_j) / a \leq x_i \leq b \; ; c \leq y_j \leq d , i = 0,1, . . ., n \; ; \; j = 0,1, . . ., m\}$. Even though we are unaware of the closed forms of the above functions, to apply FDN method we need the values of those functions at arbitrary points. Thus we use bicubic Lagrange surface patch interpolation method to approximate the functions within each of the square grid containing 16 node points.

We apply these last two procedures to several sets of data generated by various types of (known) functions getting very encouraging results.

It is best if one can check for himself / herself for the suitable combinations of the most accurate interpolation technique and the most efficient iterative technique, for the root finding problem he / she encounters. To serve this purpose, we develop a computer software package which can handle the major procedures discussed in this thesis.

Algorithms constructed were implemented by using the Turbo Pascal (Ver. 7.0) and later were converted to Microsoft Visual Basic (Ver 6.0).

Keywords : Nonlinear equations, Newton's method, Difference approximations, Finite difference Newton's method, Order of convergence, Cubic splines, Lagrange interpolation, Bilinear, Biquadratic, Bicubic.

Chapter 1

INTRODUCTION

Researchers frequently encounter systems of nonlinear equations as a result of modeling many physical processes. Newton's iterative method is the most widely used numerical scheme to find roots of systems of nonlinear equations. This method has three major drawbacks.

(1) Necessity to differentiate the functions (which sometimes could be cumbersome) and feed them into the computer.

(2) Inability to implement the algorithms when there are no closed forms for the functions of our concern. That is when there is only a discrete set of function values.

(3) Necessity to evaluate partial derivatives for the Jacobian matrix at each iterative step.

To overcome these difficulties, Finite Difference Newton's (FDN) method was introduced by Weerakoon[14] for nonlinear equations of one variable. Our objective here is to extend it to two-dimensions by replacing the 2x2 Jacobian matrix of Newton's method by appropriate finite difference approximations (either forward or backward approximations), without slowing down the process of convergence. It is shown that the suggested method converges quadratically and this theory is supported by computational results.

We apply the method to several functions such as polynomials, trigonometric and exponential functions and for all those cases, suggested method converges as fast as Newton's method and sometimes even faster.

We are fortunate to be able to extend the suggested FDN method even to functions of several variables, by providing the proof of second order convergence for functions of several variables.

Our next objective here is, to approximate the roots of an unknown function using only a set of discrete values $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$ available. For all available methods of finding roots of a nonlinear equation, we should know the function values at arbitrary points of the domain. Hence we have to look for a function approximation method which makes use of the available data optimally. We use natural cubic spline interpolation to approximate the function within the interval and apply both Newton's method and the FDN method using the approximated function. We check this procedure by applying it to several sets of data generated by various types of (known) functions and the results seem to be very encouraging.

To compare spline interpolants with actual functions, we apply both Newton's method and the FDN method to actual functions and the interpolants with the same initial guesses. Results obtained for the spline interpolant and for the actual function were very close. Further, when the initial guess is somewhat away from the root, FDN method seems to converge much faster, suggesting that in the circumstances FDN is the best method to use.

Furthermore, as an extension of the above method we try to approximate the roots of two simultaneous nonlinear equations $\{ f_1(x,y) = 0 \ ; \ f_2(x,y) = 0\}$ using only the values of the functions in a rectangular domain $\{ D = (x_i, y_j) / a \le x_i \le b \ ; \ c \le y_j \le d$, $i = 0,1, \ldots, n \ ; \ j = 0,1, \ldots, m \}$. Even though we are unaware of the closed forms of the above functions, to apply FDN method we need the values of those functions at arbitrary points. Thus we use bicubic Lagrange surface patch interpolation method to approximate the functions within each of the square grid (each grid contains 16 node points) and apply the FDN method to the approximated functions. We apply this procedure to several sets of data generated by various types of (known) functions and we also apply the FDN method using the closed forms of those functions with the same initial guesses and compare the results. Results for the surface interpolants and for the actual functions seem to be very close, suggesting the validity of the bicubic interpolant approach. This encourages applying this procedure to discrete sets of data without knowing the closed forms of the functions.

Chapter 2

Finite Difference Newton's method for functions of two variables

2.1 Introduction

Finite Difference Newton's (FDN) method was introduced by Weerakoon[14] for nonlinear equations of one variable. The objective here is to extend it to two-dimensions by replacing the 2x2 Jacobian matrix of Newton's method by appropriate finite difference approximations (either Forward or Backward approximations), without slowing down the process of convergence. It is shown that the suggested method converges quadratically and this theory is supported by computational results.

We apply the method to several functions such as polynomials, trigonometric and exponential functions and for all those cases, suggested method converges as fast as Newton's method and sometimes even faster.

2.2 Preliminaries

Definition(2.2.1) A continuous function $f : \mathbb{R}^n \to \mathbb{R}$ is said to be continuously differentiable (C^1) at $x \in \mathbb{R}^n$, if $(\partial f / \partial x_i)(x)$ exists and is continuous, $i = 1, 2, \ldots, n$; the gradient of f at x is then defined as

$$\nabla f(x) = [\ \partial f(x)/\partial x_1, \ldots, \partial f(x)/\partial x_n\ ]^T$$

A function f is said to be C^1 in an open region $D \subset \mathbb{R}^n$, denoted $f \in C^1(D)$, if it is C^1 at every point in D.

Definition(2.2.2) A continuous function $F: \mathbb{R}^n \rightarrow \mathbb{R}^m$ is C^1 at $x \in \mathbb{R}^n$ if each component function f_i , i = 1, . . ., m is C^1 at x. The derivative of F at x is sometimes called the Jacobian (matrix) of F at x, and its transpose is sometimes called the gradient F at x. The common notations are :

$$F'(x) \in \mathbb{R}^{mxn} , F'(x)_{ij} = \partial f_i(x)/\partial x_j , \quad F'(x) = J(x) = \nabla F(x)^T$$

F is said to be C^1 in an open set $D \subset \mathbb{R}^n$, denoted $F \in C^1(D)$, if F is C^1 at every point in D.

Definition(2.2.3)

If f is a function of two variables, its partial derivative with respect to x and y are the functions f_x and f_y defined by

$$f_x(x , y) = \lim_{h \to 0} \frac{f(x+h , y) - f(x , y)}{h}$$

$$f_y(x , y) = \lim_{h \to 0} \frac{f(x, y+h) - f(x , y)}{h}$$

To give a geometric interpretation of partial derivatives, we should understand that the equation $z = f(x , y)$ represents a surface S (the graph of f). If $f(a , b) = c$ then the point P(a, b, c) lies on S. The vertical plane y = b intersects S in a curve C_1 (in other word, C_1 is the trace of S in the plane y = b). Likewise the vertical plane x = a intersects S in a curve C_2. Both of the curves C_1 and C_2 pass through the point P (see figure 2.2.1).

Notice that the curve C_1 is the graph of the function $g(x) = f(x , b)$, so the slope of its tangent T_1 at P is $g'(a) = f_x(a , b)$. The curve C_2 is the graph of the function $G(y) = f(a , y)$, so the slope of its tangent T_2 at P is $G'(b) = f_y(a , b)$.

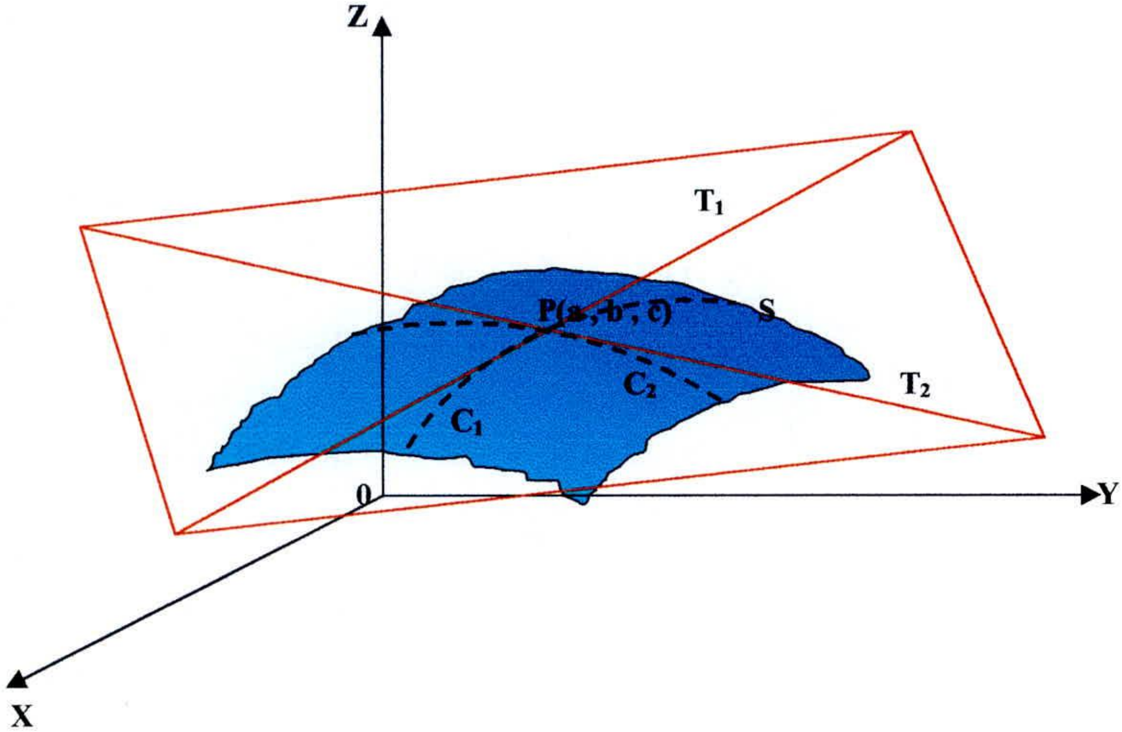

Figure 2.2.1

Thus the partial derivatives $f_x(a\ ,\ b)$ and $f_y(a\ ,\ b)$ can be interpreted geometrically as the slope of the tangent lines at P(a , b, c) to the traces C_1 and C_2 of S in the planes y = b and x = a.

Definition(2.2.4)

Suppose a surface has equation $z = f(x\ ,\ y)$, where f has continuous first partial derivatives, and let P(a , b , c) be a point on S. As in the previous definition, let C_1 and C_2 be the curves obtained by intersecting the vertical planes y = b and x = a with surface S. Then the point P lies on both C_1 and C_2. Let T_1 and T_2 be the tangent lines to the curves C_1 and C_2 at the point P. Then the tangent plane to the surface S at the point P is defined to be the plane that contains both of the tangent lines T_1 and T_2 (see figure 2.2.1).

It can be shown that if C is any other curve that lies on the surface S and passes through P, then its tangent line at P also lies in the tangent plane. Therefore, we can