Jñānābha Vol. 23, 1993

(Dedicated to Professor S.A. Naimpally on his 62th Birthday)

THE QUATTRO SPREADSHEET IN SOLVING POLYNOMIAL EQUATIONS

By

Alvin Rolland and Veena Chanda

Department of Mathematics, University of Wisconsin-Eau Claire
Eau Claire, Wisconsin, 54702-4004, U.S.A.

(Received : May 17, 1993; Revised [Final] : December 14, 1993)

ABSTRACT

The purpose of this paper is to demonstrate the use of the Quattro Pro Spreadsheet [2] to solve polynomial equations using Newton's method. The apprach uses matrix operations [3] to compute the derivative of the polynomial functions. THis approach is uique in the sene that it does not assume the knowledge of rules of differentiation. Two numerical examples explain the results.

1. INTRODUCTION

Computer spreadsheets such as Lotus 1-2-3 and Quattro Pro have many uses in addition to their obvious uses as spreadsheets. Their built-in mathematical functions and graphing capabilities have enhanced their usefulness to the mathematical world. Programming in a spreadsheet using macros and the built in functions is both exciting and powerful.

In this article we explain how we used Quattro Pro to write a program that uses the iterative Newton Raphson method to find the roots of any polynomial functions of degree five or less. On entering the coefficients of a polynomial, the computer shows a graph of the polynomial. One has to examine the graph to determine the first estimate of a root On feeding the estimated root, the computer (using Newton-Raphson's method) returns the value of a root correct to eight places. Graph is used again to estimate additional roots. At the end of the program the computer displays the polynomial, the roots that were found and a graph of the polynomial.

One of the authors has developed a unique method for evaluating the iterative values of Newton-Raphson's method using matrices [3]. The fact that the use of matrices that makes programming with macros very straight- forward is explained later in this paper.

The program is designed for students who are finding roots of equations for the first time. At the other extreme, roots found by Newton-Raphson's method can be compared. The next section explains the mathematics and matrix implementation of the program.

2. NEWTON-RAPHSON METHOD

Finding roots of polynomial functions is a very important problem in algebra and numerical analysis. There is a strong relationship between graphical and algebraic approaches. The iterative method is used here to find approximate roots of the polynomial equation of the form $P(x) = 0$, where

$$P(x) = a_m x^m + a_{m-1} x^{m-1} + \ldots + a_1 x + a_0 y$$

is a polynomial of degree m. The method begins with an initial approximation x_0 to the root of the equation. Then subsequent approximations $x_1, x_2, \ldots$ are generated using the formula

$$x_{n+1} = x_n - \frac{P(x_n)}{P'(x_n)} \text{ for } n > 0,$$

where $P'(x_n)$ denotes the derivative of $P(x)$ at x_n and is assumed to be non-zero.

The process terminates when two successive values of x are within a preset value of each other.

3. MATRIX IMPLEMENTATION

When using a computer, it is easy to use the matrix approach for finding the value of a function and its derivative. The matrix approach can be explained as : Let F be the set of all differentiable polynomial functions in variable x and M be the set of all 2×2 nonzero matrices [4].

Let $\$: F \to M$ be a map such that

$$\$P = \begin{bmatrix} P & P' \\ 0 & P \end{bmatrix},$$

where P' is the derivative of P. Then this map preserves the respective operations of addition and multiplication in F and M.

$$\$x = \begin{bmatrix} x & 1 \\ 0 & x \end{bmatrix}$$

and

$$\$x^2 = \begin{bmatrix} x & 1 \\ 0 & x \end{bmatrix} \cdot \begin{bmatrix} x & 1 \\ 0 & x \end{bmatrix}, \text{ etc.}$$

Newton-Raphson's method requires the values of P_n and P'_n of the polynomial function and its derivative for each approximation x_n. These values can be directly evaluated using the corresponding matrix for the polynomial function.

Therefore,

$$\$P|_{x_n} = \begin{bmatrix} P_n & P'_n \\ 0 & P_n \end{bmatrix}$$

and thus for

$$x_{n+1} = x_n - P_n / P'_n$$

the next approximation is

$$\$P|_{x_{n+1}} = \begin{bmatrix} P_{n+1} & P'_{n+1} \\ 0 & P_{n+1} \end{bmatrix}, \text{ etc.}$$

4. EXAMPLES

Example 1. Consider the polynomial equation

$$p(x) = x^3 - x - 1 = 0 \tag{i}$$

Since $P(1) = -1 < 0$ and $P(2) = 5 > 0$, the solution lies between 1 and 2.

Let $x_1 = 1$ be our initial approximation.

$$\$P|_1 = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}^3 - \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} - \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} -1 & 2 \\ 0 & -1 \end{bmatrix}$$

and

$$x_2 = 1 - \left(\frac{-1}{2}\right) = 1.5 \tag{ii}$$

and

$$\$P|_{1.5} = \begin{bmatrix} 1.5 & 1 \\ 0 & 1.5 \end{bmatrix}^3 - \begin{bmatrix} 1.5 & 1 \\ 0 & 1.5 \end{bmatrix} - \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} .875 & 5.75 \\ 0 & .875 \end{bmatrix}$$

and

$$x_3 = 1.5 = \left(\frac{.875}{5.75}\right) = 1.34784 \tag{iii}$$

Again

$$\$P|_{1.34783} = \begin{bmatrix} .10070 & 4.44994 \\ 0 & .10070 \end{bmatrix}$$

and

$$x_4 = 1.34783 - \frac{.10070}{4.44994} = 1.32520 \tag{iv}$$

For the next approximation

$$\$P|_{1.32520} = \begin{bmatrix} .00206 & 4.26847 \\ 0 & .00206 \end{bmatrix}$$

and

$$x_4 = 1.3252 - \frac{.00206}{4.26847} = 1.32472 \tag{v}$$

$$\$P|_{1.32472} = \begin{bmatrix} .00001 & 4.26465 \\ 0 & .00001 \end{bmatrix}$$

and

$$x_5 = 1.32472 - \frac{.00001}{4.26465} \tag{vi}$$
$$= 1.32472$$

The process terminates, approximations x_4 and x_5. are identical. Therfore, the solution is approximately $x \approx 1.3247$.

Example 2. Let

$$P(x) = 5x^5 - 7x^4 + 6x^3 + 2x^2 - x + 8$$

be the polynomial under consideration. The user enters the coefficients $5, -7, 6, 2, -1, 8$ as prompted by the computer. The initial spreadsheet is shown in Figure 1. (This part of the spreadsheet does not appear on the screen for the user.)

	A	B
1	8.000000000000000	0.000000000000000
2	0.000000000000000	8.000000000000000
4	-1.000000000000000	0.000000000000000
5	0.000000000000000	-1.000000000000000
7	2.000000000000000	0.000000000000000
8	0.000000000000000	2.000000000000000
10	6.000000000000000	0.000000000000000
11	0.000000000000000	6.000000000000000
13	-7.000000000000000	0.000000000000000
14	0.000000000000000	-7.000000000000000
16	5.000000000000000	0.000000000000000
17	0.000000000000000	5.000000000000000

Figure 1

where the constant 8 is stored in matrix $A1 \ldots B2 : -1$, the coefficient of x, is stored in matrix $A4 \ldots B5; 2$, the coefficient of x^2, is stored in matrix $A7 \ldots B8; 6$ the coefficient of x^3 is stored in matrix $A_{10} \ldots B_{11}; -7$ the coefficient of x^4 is stored in matrix $A_{13} \ldots B_{14}; 5$ the coefficient of x_n^5, is stored in matrix $A_{16} \ldots B_{17}$.

The user then has the opportunity to have the computer graph for this polynomial. He/she enters the left- and right-range values for the variable x.

The computer uses these ranges to compute 100 ordered pairs of points and plots these points on a rectangular coordinate system. The graph of $p(x)$ for x in $[-1.5, 1.5]$ is shown in Figure 2.

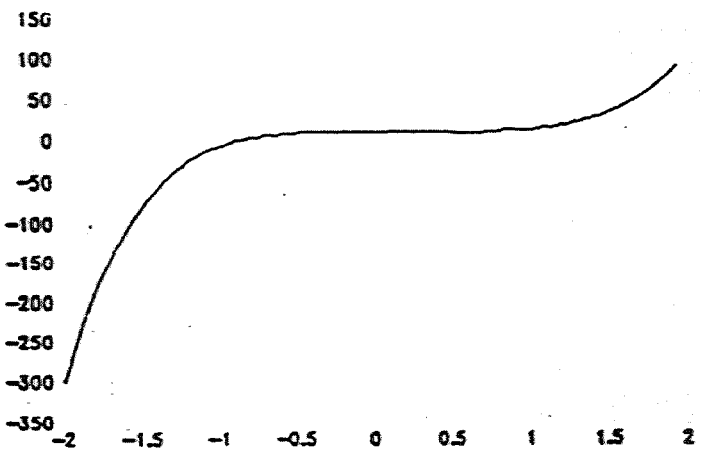


Figure 2

If the graph of the function does not provide the user with a good initial estimate of the root, he/she has the opportunity to enter new range values for varivale x. Then, from the new graph, users can choose an appropriate initial esitmate of root. The user can repeat this process to obtain a reasonable first estimate of root.

The user enters his/her first estimate at the computer prompt. If a user entered -1 as his/her first estimate it is sotred in form of matrix in $C4 \ldots D5$

The information needed for the computation is arranged in matrices in the spreadsheet shown in Figure 3. Matrix $C1 \ldots D2$ is an identity matrix.

1	8.000000000000000	0.000000000000000	1.000000000000000	0.000000000000000
2	0.000000000000000	8.000000000000000	0.000000000000000	1.000000000000000
3				
4	-1.000000000000000	0.000000000000000	-1.000000000000000	0.000000000000000
5	0.000000000000000	-1.000000000000000	0.000000000000000	-1.000000000000000
6				
7	2.000000000000000	0.000000000000000		
8	0.000000000000000	2.000000000000000		
9				
10	6.000000000000000	0.000000000000000		
11	0.000000000000000	6.000000000000000		
12				
13	-7.000000000000000	0.000000000000000		
14	0.000000000000000	-7.000000000000000		
15				
16	5.000000000000000	0.000000000000000		
17	0.000000000000000	5.000000000000000		

Figure 3

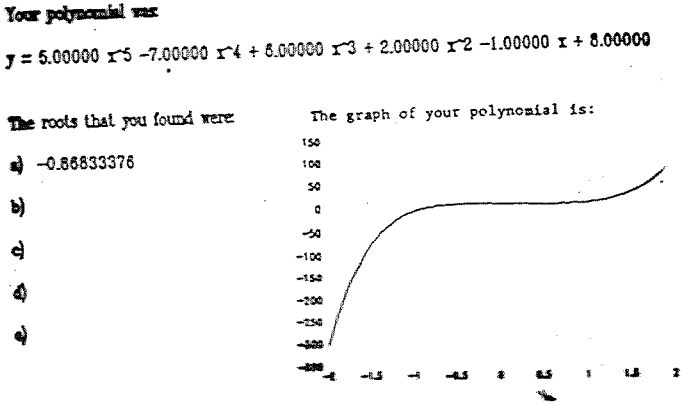
As indicated previously, the numerical coefficients, the constant terms, and the estimate of the root have been arranged in the computer as matrices. The arrangement of the matrices in the spreadsheet was determined in such a way as to make the writing of the macro instructions simple. See Figure 4 for an example of the macro instructions that were used.

```
83  {let c9,0}
84  {if a5=0}{let e5,0}{let e6,0}{let f5,0}{let f6,0}{branch a86}
85  {/ math;multiplyMatrix}a5..b6{cr}c5..d6{cr}e5..f6{cr}
86  {/ math;multiplyMatrix}c8..d9{cr}c8..d9{cr}c11..d12{cr}
87  {if a8=0}{let e8,0}{let e9,0}{let f8,0}{let f9,0}{branch a89}
88  {/ math;multiplyMatrix}a8..b9{cr}c8..d9{cr}e8..f9{cr}
89  {/ math;multiplyMatrix}c8..d9{cr}c11..d12{cr}c14..d15{cr}
90  {if a11=0}{let e11,0}{let e12,0}{let f11,0}{let f12,0}{branch a92}
91  {/ math;multiplyMatrix}a11..b12{cr}c11..d12{cr}e11..f12{cr}
92  {/ math;multiplyMatrix}c11..d12{cr}c11..d12{cr}c17..d18{cr}
93  {if a14=0}{let e14,0}{let e15,0}{let f14,0}{let f15,0}{branch a95}
94  {/ math;multiplyMatrix}a14..b15{cr}c14..d15{cr}e14..f15{cr}
95  {/ math;multiplyMatrix}c8..d9{cr}c17..d18{cr}c20..d21{cr}
96  {if a17=0}{let e17,0}{let e18,0}{let f17,0}{let f18,0}{branch a98}
97  {/ math;multiplyMatrix}a17..b18{cr}c17..d18{cr}e17..f18{cr}
98  {if a20=0}{let e20,0}{let e21,0}{let f20,0}{let f21,0}{branch a100}
99  {/ math;multiplyMatrix}a20..b21{cr}c20..d21{cr}e20..f21{cr}
100 {let e23,e5+e8+e11+e14+e17+e20}
101 {let e24,e6+e9+e12+e15+e18+e21}
102 {let f23,f5+f8+f11+f14+f17+f20}
```

Figure 4

Notice the pattern that appears in the cell labels. This is one illustration of the power of macros as a programming tool. The entire

program is written as a series of macros that are executed one after another. Using a spreadsheet and macros enables the programmer to use the functions that are build into the spreadsheet rather than recreating them. After the multiplications and additions have been completed, the diference between the last two approximations of the root is computed. If this difference is less than a preset value, the aproximation for the root is printed out. If the difference is greater than the preset value (.00001), the program loops to the beginning of the series of macros and repeats the process. The computer tells the user when each iteration has been completed. The maximum number of iterations has been set to twenty. If a root has not been found after twenty iterations, the computer suggests to the user that he/she should try a different number as an esitmate of the root. After a root has been determined, the user has the option to find another root, or to quit. When he/she quits, the computer displays the original polynomial, the roots that were found, and a graph of the polynomial.



This program can be used by teachers at several different levels.

Algebra teachers discussing roots of polynomials could use the program to show students the relationship between the graphs of the polynomials and the value of the roots of the polynomials. Since the numerical computations never appear on the screen, students do not need to understand the method being used to determine the roots. Students could use the program in a computer laboratory to develop an uderstanding of the relationships that exist. They could examine various types of polynomials and discover many interesting and

important ideas. This use of the program could range from elementary algebra through pre-calculus courses.

REFERENCES

[1] Howard Anton, *Calculus with Analytic Geometry*, Third eddition, John Wiley Sons, New York, 1988.

[2] Borland International, Quattro Pro, 1990.

[3] V. Chadha Differentiating polynomial/rational functions with matrices, Mathematics in College. 1990, 44-47.

[4] I.N. Herstein, *Topics in Algebra*, Second edition, Xerox, Lexington, Massachusetts, 1975.

[5] V.S. Ramaurthi, Spreadsheet works : finding maximum and minimum of functions without calculus, *J. Comput. Mat. Sci. Teach.* 7 no. 4, (1988), 81-83.