1 Experiment No. 4

2 Experiment Title

Solving Non-linear Equations Using Newton-Raphson and Secant Method

3 Objective

The objectives of this lab are:

- To understand and implement the Newton-Raphson and Secant methods for solving non-linear equations.
- To display the approximations, root and error for each iteration in a tabular form.

4 Theory

4.1 Newton-Raphson Method

The Newton-Raphson method is an iterative technique to find the roots of a real-valued function. It is based on the idea of linear approximation using tangents. Starting from an initial guess x_0, the next approximation is computed using the formula:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

The iteration continues until the error falls below a pre-defined tolerance.

4.1.1 Algorithm:

1. Start with an initial guess x_0.
2. Compute $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$.
3. Compute the error $|x_{n+1} - x_n|$.
4. If error is less than tolerance, stop. Otherwise, update $x_n = x_{n+1}$ and repeat.

4.2 Secant Method

The Secant method is also an iterative root-finding algorithm but does not require the derivative of the function. It uses two initial approximations x_0 and x_1, and computes subsequent approximations using the formula:

$$x_{n+1} = x_n - f(x_n) \cdot \frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})}$$

4.2.1 Algorithm:

1. Choose two initial values x_0 and x_1.
2. Compute x_{n+1} using the secant formula.
3. Compute the error $|x_{n+1} - x_n|$.
4. If error is below tolerance, stop. Otherwise, update $x_{n-1} = x_n$, $x_n = x_{n+1}$ and repeat.

4.3 Solving Non-linear Equation Using Newton-Raphson Method

Here $f(x) = x^2 - 2x - 5$, error is assumed to be 0.001

4.3.1 MATLAB Code:

```

clc;
clear;

f = @(x) x^2 - 2*x - 5;
df = @(x) 2*x - 2;
error = 0.001;
N = 500;

x0 = input('Enter initial value \n');

for k = 1:N
if df(x0) == 0
fprintf('Not Converging, df(x0) is zero. Enter another initial value \n');
x0 = input('Enter another value: \n');
continue;
end

x1 = x0 - f(x0) / df(x0);
er = abs(x1 - x0);

fprintf('Iteration %d: x0 = %.6f, x1 = %.6f, f(x1) = %.6f, error = %.6f\n',
    k, x0, x1, f(x1), er);

if abs(f(x1)) < error || er < error
fprintf('The root of the equation is %.6f\n', x1);
break;
end

x0 = x1;
end
```

Listing 1: Solving Non-linear Equation Using Newton-Raphson Method in MATLAB.

4.3.2 Result Shown in Command Window

```
Enter initial value
1
Not Converging, df(x0) is zero. Enter another initial value
Enter another value:
2
Iteration 2: x0 = 2.000000, x1 = 4.500000,f(x1) = 6.250000, error = 2.500000
Iteration 3: x0 = 4.500000, x1 = 3.607143,f(x1) = 0.797194, error = 0.892857
Iteration 4: x0 = 3.607143, x1 = 3.454256,f(x1) = 0.023374, error = 0.152886
Iteration 5: x0 = 3.454256, x1 = 3.449494,f(x1) = 0.000023, error = 0.004762
The root of the equation is 3.449494
```

Listing 2: Command Window for Newton-Raphson Method

4.4 Solving Non-linear Equation Using Secant Method

Here $f(x) = x^3 - 2x - 5$, error is assumed to be 0.001

4.4.1 MATLAB Code:

```

clc
clear

f = @(x) x^3 - 2*x -5;
df = @(x) 3*x^2 - 2;
error = 0.001;
N = 500;
x0 = 5;
x1 = 10;

for k = 1:N
x2 = x1 - f(x1)*(x1-x0) / (f(x1)-f(x0));
er = abs(x2 - x1);

fprintf('Iteration %d: x0 = %.6f, x1 = %.6f,f(x1) = %.6f,error = %.6f\n', k,
     x0, x1, f(x1), er);

if abs(f(x2)) < error || er < error
fprintf('The root of the equation is %.6f\n', x2);
break;

end

x0 = x1;
x1= x2;
end
```

Listing 3: Solving Non-linear Equation Using Secant Method in MATLAB.

4.4.2 Result Shown in Command Window

```
Iteration 1: x0 = 5.000000, x1 = 10.000000,f(x1) = 975.000000,error=5.635838
Iteration 2: x0 = 10.000000, x1 = 4.364162,f(x1) = 69.391104,error=0.431839
Iteration 3: x0 = 4.364162, x1 = 3.932323,f(x1) = 47.941514,error = 0.965193
Iteration 4: x0 = 3.932323, x1 = 2.967130,f(x1) = 15.187929,error = 0.447563
Iteration 5: x0 = 2.967130, x1 = 2.519567,f(x1) = 5.955621,error = 0.288716
Iteration 6: x0 = 2.519567, x1 = 2.230851,f(x1) = 1.640562,error = 0.109768
Iteration 7: x0 = 2.230851, x1 = 2.121082,f(x1) = 0.300566,error = 0.024621
Iteration 8: x0 = 2.121082, x1 = 2.096461,f(x1) = 0.021337,error = 0.001881
The root of the equation is 2.094580
```

Listing 4: Command Window for Secant Method

5 Discussion

In this experiment, the Newton-Raphson and Secant methods was introduced to solve a non-linear equation, highlighting its effectiveness in root-finding using MATLAB.
The Newton-Raphson Method demonstrated fast convergence when the derivative was not zero. It required a correct initial guess to avoid division by zero or divergence. .It was observed that this method guarantees finding only one root at a time. From the results, Using an error tolerance of 0.001, the root was found in 4 effective iterations after correcting the initial guess to converge, yielding a root of 3.449494 with an error of 0.004762.
The Secant Method, on the other hand, does not need the derivative of the function, which makes it suitable for functions that are hard to differentiate. Starting with two initial values, the method converged after 8 iterations and provided a reliable root estimate with acceptable error.It was also observed that this method guarantees finding only one root at a time. From the results, Using an error tolerance of 0.001, the root was found in 8 effective iterations after correcting the initial guess to converge, yielding a root of 2.094580 with an error of 0.001881.The number of iterations depends on the initial guess x0 and x1.
Overall, both methods effectively found the roots, and the iteration results verified the accuracy and performance of each approach under the given stopping criteria. .