Homework 2

Arnab Bachhar

September 16, 2023

0.1 Secant Method

1. Write a program to solve tan(x) = a/x for x in (Π/2, 3Π/2), where a = 5, by using the secant method. If necessary, you may combine the secant method with the bisection method. See if your final answer changes with three different initial guesses.

→In secant method, we can calculate the root for a real-valued function. But, there are some drawbacks in this method :

1. We have to make sure that the initial x guesses satisfy this condition:

$$f(x_0) * f(x_1) < 0.0 \tag{1}$$

2. Also particularly, for this above mentioned problem, we can't take $x = \pi/2$ or $x = 3\pi/2$ as, tan(x) value is infinity in those x values; so, the secant method will not be able give correct approximated result for those two x values.

Secant method is quite simple, we have to take the function and compute the value for initial guess (x_0) and then we will update the x_i and x_j in each iteration using this equation:

$$x_j = x_j - \frac{f(x_j) \cdot (x_j - x_i)}{f(x_j) - f(x_i)} \tag{2}$$

This iterative procedure continues until

$$|x_j - x_i| < \epsilon \tag{3}$$

Algorithm 1 Secant Method

```
procedure SecantMethod(equation, $a$, $x_{\text{values}}$, tol, max_iter)
    Create random $x_i$ and $x_j$ values from $x_{values}$
    print($x_i$)
    print($x_j$)
    if equation($x_i, a$) $\cdot$ equation($x_j, a$) $< 0.0$ then
        for $i \leftarrow 1$ to max_iter do
            $f_i \leftarrow$ equation($x_i, a$)
            $f_j \leftarrow$ equation($x_j, a$)
            if abs($x_j - x_i$) $>$ tol then
                $x_{\text{next}} \leftarrow x_j - f_j \cdot (x_j - x_i)/(f_j - f_i)$
                $x_i, x_j \leftarrow x_j, x_{\text{next}}$
                printf(" Iteration No: Abs(xj-xi): xj:)
            else if $i =$ max_iter then
                println("it didn't converge")
            else
                return $x_j$
                break
            end if
        end for
    else
        println("the initial points are not suitable for this method")
    end if
end procedure
```

where, ϵ is the tolerance value.

Time spent on the code and report total: 2 hours.

A simplified algorithm is presented here.

The approximated root value is 4.033567790339982. I have implemented the secant method as it takes random initial two x values and if these values don't satisfy the equation 1 condition, it shows that these points are not suitable for secant method. The output of the code will look like this :

```
3.3896131246626715

4.3816950089541855

Iteration No: 1 Abs(xj-xi): 0.587162974197 Xj: 3.794532034757

Iteration No: 2 Abs(xj-xi): 0.139636830192 Xj: 3.934168864950

Iteration No: 3 Abs(xj-xi): 0.120828638742 Xj: 4.054997503692

Iteration No: 4 Abs(xj-xi): 0.023646647056 Xj: 4.031350856635

Iteration No: 5 Abs(xj-xi): 0.002165210439 Xj: 4.033516067074

Iteration No: 6 Abs(xj-xi): 0.000051846842 Xj: 4.033567913916
Iteration No: 7 Abs(xj-xi): 0.000000123583 Xj: 4.033567790333
Iteration No: 8 Abs(xj-xi): 0.000000000007 Xj: 4.033567790340
Approximated root: 4.033567790339982
```

Here, the first two digits are randomly chosen x values and the iteration no , difference between x values in consequent iterations and the updated x values are printed.

I have checked for three random x_0 values , it gives same approximated root 033567790339982. So, secant method without those two drawbacks is quite good at predicting . Time for running code also gets printed if you run the code which I have not included here as it depends on the initial x values.

0.2 Gauss Elimination

Write a program to solve the following linear algebraic equations by using the Gauss Elimination method. Since we did not have time to discuss pivoting, the pivoting step is not needed for this problem. Compare your numerical result with the answer, $x = \frac{21}{34}, y = \frac{43}{68}, z = \frac{-13}{34}, u = \frac{1}{4}$.

10x - 4y + 3z + 2u =3 .... (1)

x - y - 2z + u =1 ....(2)

3x + 2y - z + 2u =4............................(3)

2x + 3y + z - 3u =2...........................(4)

$\rightarrow$ Although pivoting is not needed for this problem, but I have done that for generalisation of the code as I knew that because of a previous course.

Gauss Elimination is a very fundamental technique or method to solve linear equations. It has only two main parts:

1. **Forward Elimination:** We concatenate the the augmented coefficient matrix and the vector. Then, we pivot the rows i.e. row with the largest absolute value elements will be the first row. Then, we form the upper-triangular matrix in which all the elements below the diagonal will be zero.

2. **Backward Substitution:** We first solve for the last unknown variable associated with the last row directly by dividing the right-hand side (RHS) value by the coefficient of the last variable in that row. This will give the value of the last unknown. Now, we have to move up to the next row and give the previously solved value of unknown variable in this row and that will give value of another unknown variable. Similarly, we can move up and solve for other unknown variables.

I have verified the results of my code with the analytical values given in the question.

Time spent on the code and report total: 3 hours.

Time for running code also gets printed:

0.410778 seconds (634.64 k allocations: 42.278 MiB, 14.39% gc time, 99.99% compilation time)

Here, a simple algorithm of Gauss Elimination is written below:

Algorithm 2 Gaussian Elimination

```
procedure GaussElimination($A, b$)
    $n \leftarrow$ length of vector $b$
    Create augmented matrix $Ab$ by stacking $A$ and $b$
    for $k \leftarrow 1$ to $n$ do
        Find the pivot row and swap rows if necessary
        $maxval, pivotrow \leftarrow \text{findmax}(\text{abs}(Ab[k:n,k]))$
        $pivotrow \leftarrow pivotrow + k - 1$
        Swap rows: $Ab[k,:], Ab[pivotrow,:] \leftarrow Ab[pivotrow,:], Ab[k,:]$
        for $i \leftarrow k+1$ to $n$ do
            Calculate factor: $factor \leftarrow \frac{Ab[i,k]}{Ab[k,k]}$
            Update row $i$: $Ab[i,k:n+1] \leftarrow Ab[i,k:n+1] - factor \cdot Ab[k,k:n+1]$
        end for
    end for
    Initialize solution vector $x$ with zeros of length $n$
    for $i \leftarrow n$ down to 1 do
        Calculate $x[i]$: $x[i] \leftarrow \frac{Ab[i,\text{end}] - \sum_{j=i+1}^{n} Ab[i,j] \cdot x[j]}{Ab[i,i]}$
    end for
    Return solution vector $x$
end procedure
```