Activity 4.1: Derivatives

Things to compare: which one works better, which one looks nicer, which one is closer to the derivative function, in the conclusions have the error for each method (by hand or comparing the value obtained and the real value, and having the difference of those 2 values). The bare minimum is to plot and compare the solutions at least in a qualitative way (the absolute error is quantitative)

**Problem:** Analyze the behavior of the numerical derivative function of any real function by plotting the numerical data using the different derivative formulas.

**Code:**

module functions

    implicit none

    contains

        real function f(*x*)

            implicit none

            real, intent (in) :: x

            f = x\*x\*x + 3\*x\*x + 3\*x + 1

            return

        end function

end module

*!All the first order derivative subroutines*

subroutine twoPointForward(*x0*, *h*, *derivative2PF*)

    use functions

    implicit none

    real, intent(in) :: x0, h

    real, intent(out) :: derivative2PF

    derivative2PF = ((f(x0+h) - f(x0))/h)

end subroutine twoPointForward

subroutine twoPointBackward(*x0*, *h*, *derivative2PB*)

    use functions

    implicit none

    real, intent(in) :: x0, h

    real, intent(out) :: derivative2PB

    derivative2PB = ((f(x0) - f(x0-h))/h)

end subroutine twoPointBackward

subroutine threePointEndpoint(*x0*, *h*, *derivative3PE*)

    use functions

    implicit none

    real, intent(in) :: x0, h

    real, intent(out) :: derivative3PE

    derivative3PE = (-3\*f(x0) + 4\*f(x0 + h) - f(x0 + 2\*h))/(2\*h)

end subroutine threePointEndpoint

subroutine threePointMidpoint(*x0*, *h*, *derivative3PM*)

    use functions

    implicit none

    real, intent(in) :: x0, h

    real, intent(out) :: derivative3PM

    derivative3PM = (f(x0 + h) - f(x0 - h))/(2\*h)

end subroutine threePointMidpoint

subroutine fivePointEndpoint(*x0*, *h*, *derivative5PE*)

    use functions

    implicit none

    real, intent(in) :: x0, h

    real, intent(out) :: derivative5PE

    derivative5PE = (-25\*f(x0) + 48\*f(x0+h) - 36\*f(x0 + 2\*h) + 16\*f(x0 + 3\*h) - 3\*f(x0 + 4\*h))/(12\*h)

end subroutine fivePointEndpoint

subroutine fivePointMidpoint(*x0*, *h*, *derivative5PM*)

    use functions

    implicit none

    real, intent(in) :: x0, h

    real, intent(out) :: derivative5PM

    derivative5PM = (f(x0 - 2\*h) - 8\*f(x0 - h) + 8\*f(x0 + h) - f(x0 + 2\*h))/(12\*h)

end subroutine fivePointMidpoint

*!All the higher order derivative subroutines.*

subroutine threePointMidpointSecond(*x0*, *h*, *derivative3PM2*)

    use functions

    implicit none

    real, intent(in) :: x0, h

    real, intent(out) :: derivative3PM2

    derivative3PM2 = (f(x0 - h) - 2\*f(x0) + f(x0 + h))/(h\*h)

end subroutine threePointMidpointSecond

subroutine fivePointMidpointSecond(*x0*, *h*, *derivative5PM2*)

    use functions

    implicit none

    real, intent(in) :: x0, h

    real, intent(out) :: derivative5PM2

    derivative5PM2 = (-f(x0-(2\*h)) + 16\*f(x0 - h) - 30\*f(x0) +16\*f(x0 + h) - f(x0))/(12\*h\*h)

end subroutine fivePointMidpointSecond

subroutine fivePointMidpointThird(*x0*, *h*, *derivative5PM3*)

    use functions

    implicit none

    real, intent(in) :: x0, h

    real, intent(out) :: derivative5PM3

    derivative5PM3 = (-f(x0 - 2\*h) + 2\*f(x0-h) - 2\*f(x0 + h) + f(x0 + 2\*h))/(2\*h\*h\*h)

end subroutine fivePointMidpointThird

subroutine fivePointMidpointFourth(*x0*, *h*, *derivative5PM4*)

    use functions

    implicit none

    real, intent(in) :: x0, h

    real, intent(out) :: derivative5PM4

    derivative5PM4 = (f(x0 - 2\*h) - 4\*f(x0 - h) + 6\*f(x0) - 4\*f(x0 + h) + f(x0 + 2\*h))/(h\*h\*h\*h)

end subroutine fivePointMidpointFourth

*!The program*

program derivatives

    use functions

    implicit none

*!José Antonio Solís Martínez . 162442 . Activity 4.1*

    real :: a, b, h=0.01, delta, x0

    real :: derivative2PF, derivative2PB, derivative3PE, derivative3PM, derivative5PE, derivative5PM

    integer :: selection, i, n

    real, dimension(200) :: derivativeArray2PF, derivativeArray2PB,derivativeArray3PE,derivativeArray3PM, &

                            derivativeArray5PE,derivativeArray5PM, x0Array *!Size 200 just to be sure*

    write(\*,\*) 'Use default values or input custom values? (0=Default) (1=Custom)'

    read(\*,\*) selection

    if ( selection .EQ. 1 ) then

        write(\*,\*) 'Input the values of a, b and n '

        read(\*,\*) a, b, n

    else

        a = -13

        b = 11

        n = 100

    end if

    delta = (b-a)/n

    x0 = a

    do i = 1, n

        call twoPointForward(x0, h, derivative2PF)

        call twoPointBackward(x0, h, derivative2PB)

        call threePointEndpoint(x0, h, derivative3PE)

        call threePointMidpoint(x0, h, derivative3PM)

        call fivePointEndpoint(x0, h, derivative5PE)

        call fivePointMidpoint(x0, h, derivative5PM)

*!Now store which derivative in the array???*

        derivativeArray2PF(i) = derivative2PF

        derivativeArray2PB(i) = derivative2PB

        derivativeArray3PE(i) = derivative3PE *!This is the good one*

        derivativeArray3PM(i) = derivative3PM

        derivativeArray5PE(i) = derivative5PE

        derivativeArray5PM(i) = derivative5PM

        x0Array(i) = x0

*!write(\*,\*)'For x0 =', x0*

*!write(\*,\*)derivative2P,derivative3PE,derivative3PM,derivative5PE,derivative5PM*

        x0 = x0 + delta

    end do

*!Write the x0 and derivative to the file*

    open(1,*file* = 'infoDerivatives.txt')

    write(1,\*) '#x                  TwoPoint ThreePointEndpoint ThreePointMidpoint FivePointEndpoint FivePointMidpoint '

    do i = 1, n

        write(1,\*)  x0Array(i), derivativeArray2PF(i), derivativeArray2PB(i), derivativeArray3PE(i), &

                    derivativeArray3PM(i), derivativeArray5PE(i), derivativeArray5PM(i)

*!5 format(F9.2, F9.1, F9.2, F9.3, F9.4, F9.5)*

    end do

    close(1)

    write(\*,\*) 'Output can be found in "infoDerivatives.txt"'

*!Plot with gnuplot*

    call execute\_command\_line('gnuplot -p plot.plt')

end program derivatives

Code 1. Full code of derivatives.f90

Summarizing Code 1 written in fortran, the function of which we find the derivative is , then the subroutines for the forward two point formula(2PF) backward two point formula(2PB) three point endpoint(3PE) three point midpoint(3PM) five point endpoint(5PE) and five point midpoint(5PM) are defined

This program asks to the user if they want to input custom values for a, b and n, or the program can use the default values. The default values were a = -13, b = 11 and n = 100; these were the values chosen as they show better the behavior of the derivative function when plotted.

Next, the derivatives are calculated using each of the six formulas and stored into an their corresponding array, to later be written to a file that was in the first column the x value, then in the next column the derivative calculated using the forward two point approximation, then in the next the one using the backward two point approximation, and so on and so forth.

Finally, it tells to the user that the information was written to the file “infoDerivatives.txt”, and the results are plotted using gnuplot.

# plot.plt from https://cyber.dabamos.de/programming/modernfortran/gnuplot.html

set terminal windows 0

set title "Derivative plot"

set grid

set xlabel "x"

set ylabel "y"

points="infoDerivatives.txt"

f(x)=3\*x\*\*2 + 6\*x + 3

plot points using 1:2 title "2PF" with points, points using 1:3 title "2PB" with points, points using 1:4 title "3PE" with points, points using 1:5 title "3PM" with points, points using 1:6 title "5PE" with points, points using 1:7 title "5PM" with points, f(x) title "Derivative function"

Code 2. Plot.plt file

In order to plot the results when the program finishes calculating, a plot.plt file is given to gnuplot so it has the commands needed to plot the information. In this case it gives the plot a title, sets the labels for the x and y axis and plots the real derivative of the function, in this case being , and also plots the points calculated by each of the derivative subroutines by loading them from the file.

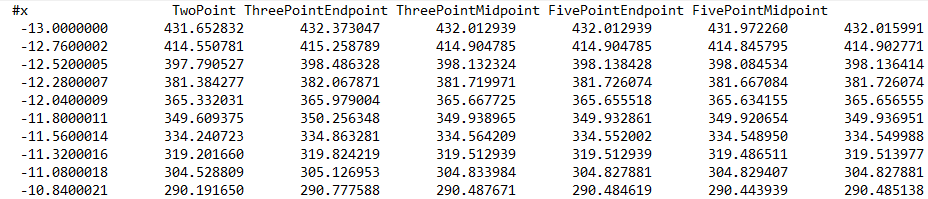


Image 1. infoDerivatives.txt content

The file from Image 1 is generated at run time, and due to the difficulties of adding it to this word document, and amount of points, the file itself is omitted but can be generated by running the compiled “derivatives.f90” code.

In the infoDerivatives.txt file, as explained before, the x point and the derivative in that point using each method is presented in a column, this format allows to easily plot the points all together.

**Analysis of results:**

**Conclusions:**