Natalie Valett
CSE 13S Spring 2021
Prof. Darell Long

Asgn2 Design.pdf: A Small Numerical Library

This project is meant to replicate the functionality of a subset of the <math.h> library. It includes functions for arcSin, arcCos, arcTan, and Log, calculating them using either the inverse/Newton's method or using taylor Series (my program employs the former).
In mathlib.c all 4 of these functions are defined, along with 3 helper functions Sqrt(double x), Abs(double x), and Exp(double x).

The user interfaces with this program through the tester program, mathlib-test.c. mathlib-test.c should test each mathematical function from our library and compare it with the output from the C math library math.h. Certain values within the range of each function are fed to both math libraries, and our program outputs a chart that compares the 2 outputs and calculates the difference between them.

Chosen method of computation - NEWTON'S METHOD:
Ex (given by Dr. Long on Piazza https://piazza.com/class/kmfs2bmdr9syz?cid=50):

```
while (fabsl(new - old) > epsilon) {          -> While the error is larger than 10e-10
      old = new;
      new = 0.5 * (new + x / new);
}
return new;
```

- Continues until the difference between old and new calculations are less than epsilon (10e-10)
- Computes each new step using the general formula $x_{k+1} = x_k - (f(x_k) / f'(x_k))$, which can be applied to any differentiable function f(x)

ArcSin:

- Given double x
- Returns double result of arcsin(x)

Planning:

- We can either use Newton's Method or Taylor Series

Using Newton's Method, we can use $x_{k+1} = x_k - (f(x_k) / f'(x_k))$ using f(x) = arcsin(x) = a
To allow us to simplify the function, we can convert f(x) to sin(x) - a, and f'(x) = cos(x)
Function to use for Newton's Method is:
$x_{k+1} = x_k - (\sin(x_k) - a / \cos(x_k))$

Pusedocode:

```
double arcSin(double x):
    double new = x
    double old = 0.0
    // while the difference is significantly large, continue
    while (absolue val of (new - old) > epsilon):
        old = new
        new = old - ( (sin(old) - x) / cos(old) )
    return new
```

Testing:
Range = [−1,1)
Step = 0.1

ArcCos:

- Given double x
- Returns double result of arccos(x)

Planning:

- We can either use Newton's Method or Taylor Series
  OR simplify it by implementing arcSin function:

$$\arccos(x) = \frac{\pi}{2} - \arcsin(x).$$

Pusedocode:

```
double arcCos(double x):
    result = (pi/2) - arcSin(x)
    Return result
```

Testing:
Range = [−1,1)
Step = 0.1

<u>**ArcTan:**</u>

- Given double x
- Returns double result of arcsin(x)

Planning:

- We can either use Newton's Method or Taylor Series
  OR utilize predefined functions arcsin(x) or arccos(x)

$$\arctan(x) = \arcsin\left(\frac{x}{\sqrt{x^2+1}}\right) = \arccos\left(\frac{1}{\sqrt{x^2+1}}\right), \quad x > 0.$$

Pusedocode:

```
double arcTan(double x):
    result = arcSin( x / Sqrt( (x*x) + 1) )
    Return result
```

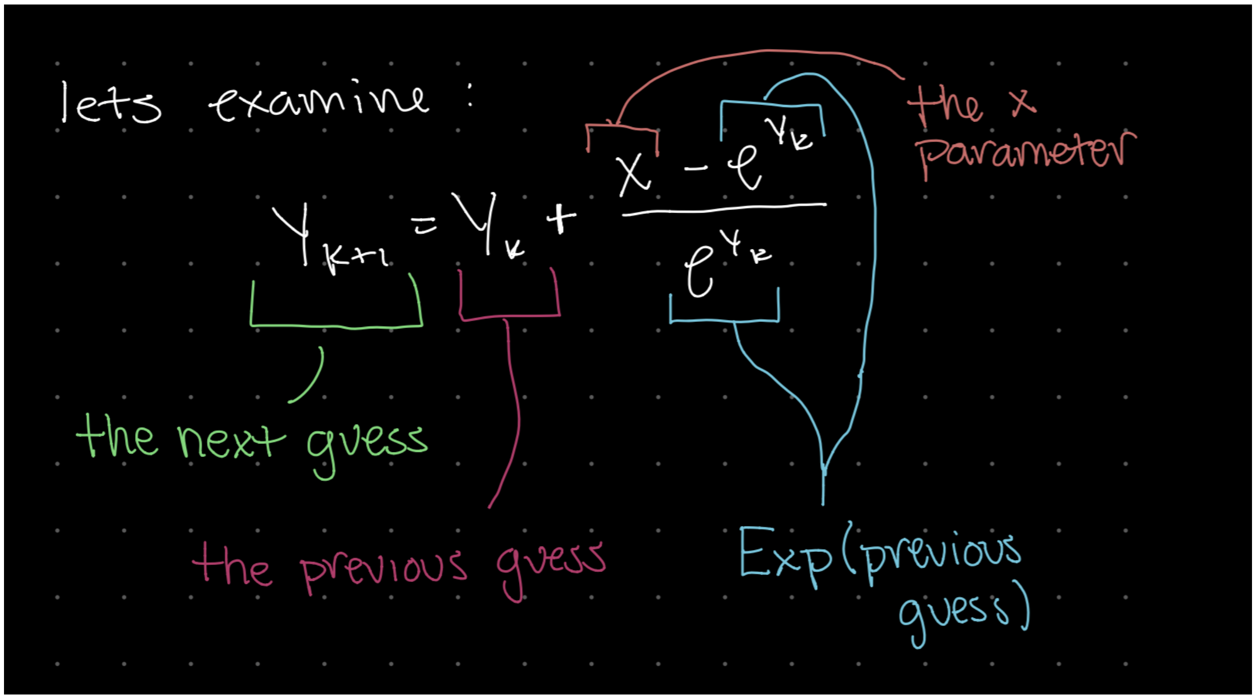
Testing:
Testing range = [1,10)
Step = 0.1

<u>**Log:**</u>

- Given double x
- Returns double result of arcsin(x)

Planning:

- Must use Newton's Method



^Credit to Eric Hernandez for sharing log pdf on discord

Pusedocode:

```
double Log(double x):
    double new = 1.0
    double old = 0.0

    // while the difference is significantly large, continue
    while (absolue val of (new - old) > epsilon):
        old = new
        new = old + ( x - (e^old)) / (e^old) )

    return new // should hold close enough approximation to actual value of log(x)
```

Testing:

Testing range = [1,10)

Step = 0.1

<u>mathlib-test.c:</u>

Purpose: interacts with user as a console for them to select which functions they want to test. Uses getopt() function to process user input given at runtime (a = all, s = arcsin, c = arccos, t = arctan, l = log). Then the program will run tester functions for all indicated math functions, starting from lower end of its range and iterating by steps of 0.1 until the iterator reaches upper limit of the function. The program will generate and format a header and table based on the values tested and resulting values each function returns. Lastly, each function is compared to the corresponding function from the <math.h> C library, and computes and outputs the difference.

Psuedocode:

```
ints arcsin, arccos, arctan, log = 0
while (user input still exists) { //uses getopt()
    // can use switch here or if statements
    If user input == 'a'
        arcsin = 1
        arccos = 1
        arctan = 1
        log = 1
    If user input == 's'
        arcsin = 1
    If user input == 'c'
        arccos = 1
    If user input == 'l'
        log = 1
}
if arcsin == 1
    Print test header for arcsin
    double i = -1.0
    while (i < 1):
        Print i
        Print result of arcSin(i) (my function)
        Print result of asin(i) (math library function)
        Print difference between arcSin(i) and asin(i)
        i += 0.1
if arccos == 1
    Print test header for arccos
    double i = -1.0
    while (i < 1):
        Print i
        Print result of arcCos(i) (my function)
        Print result of acos(i) (math library function)
        Print difference between arcCos(i) and acos(i)
        i += 0.1
if arctan == 1
    Print test header for arctan
    double i = 1.0
    while (i < 10):
        Print i
        Print result of arcTan(i) (my function)
        Print result of atan(i) (math library function)
        Print difference between arcTan(i) and atan(i)
        i += 0.1
if log == 1
    Print test header for log
    double i = 1.0
    while (i < 10):
        Print i
        Print result of Log(i) (my function)
        Print result of log(i) (math library function)
        Print difference between Log(i) and log(i)
        i += 0.1
```