Ruchit Patel

CSE13S, Spring 2021

Prof. Darrell Long

15 April 2021

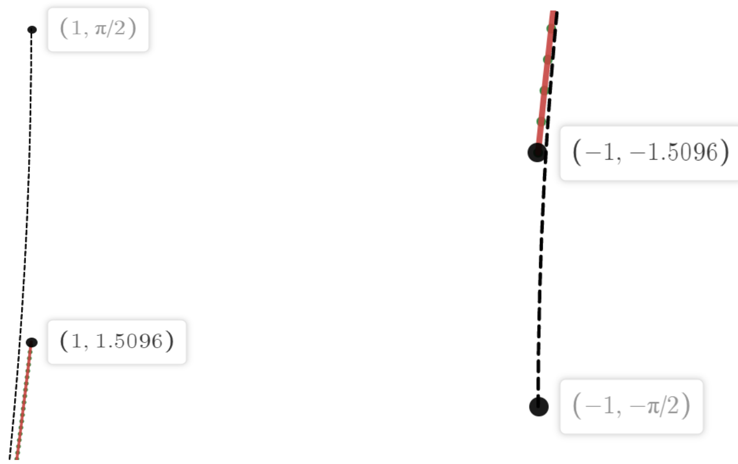
Assignment 2: A Small Numerical Library

Analysis of arcSin(x) vs <math.h> asin(x) results

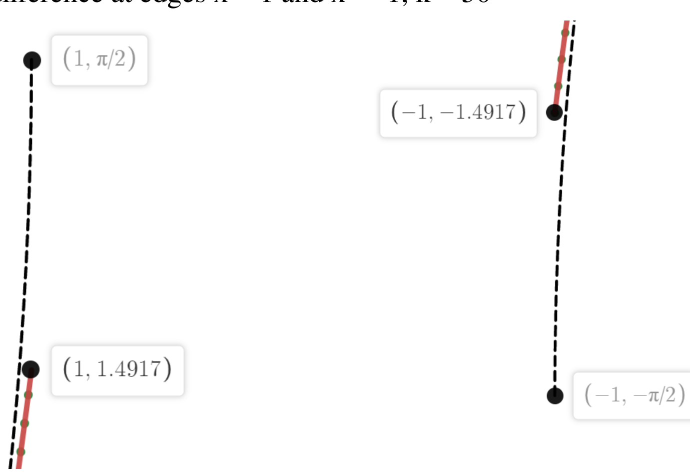
1. Taylor Series Edge Differences [I changed my implementation to use inverse method later]

   [Note: For the following graphs black dotted line is the library output and red and green represent the taylor series output]

   a. arcsin(x) difference at edges x = 1 and x = -1, k = 84



   b. arcsin(x) difference at edges x = 1 and x = -1, k = 50



➔ Possible Explanation [applies to x = -1 as well]

- The Taylor series for arcsin(x) centered at zero is given by the following formula:

$$arcsin(x) = \sum_{k=0}^{\infty} \frac{(2k)!}{2^{2k} * (k!)^2} * \frac{x^{2k+1}}{2k+1}, \ |x| <= 1.$$

- From the above series it can be seen that $\lim_{x \to 1^-} \frac{x^{2k+1}}{2k+1} = \frac{1}{2k+1}$ and thus the the coefficient $\frac{(2k)!}{2^{2k} * (k!)^2} * \frac{1}{2k+1}$ is the only relevant term during calculation of values near 1. Thus it does not converge as fast as it would with x term.
- On top of the above reason, the series becomes inaccurate as the value of x increases since the series is centered around zero.

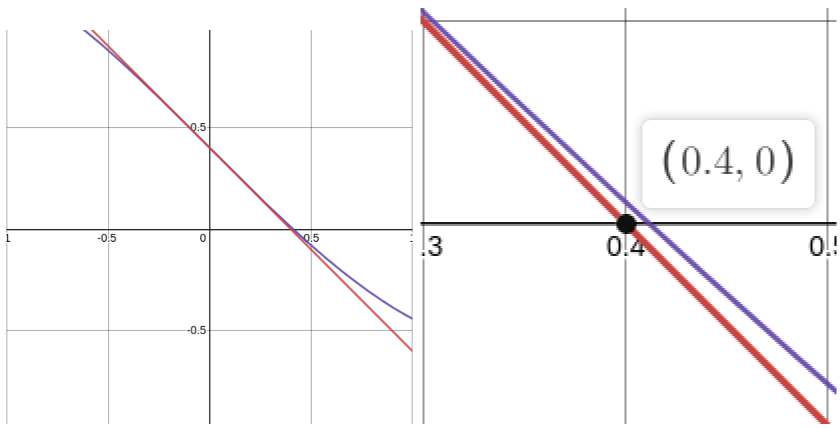
2. Inverse Method Differences
    a. Table:

x	arcSin	Library	Difference
-	------	-------	----------
-1.0000	-1.57079632	-1.57079633	0.0000000074845405428902722633210942149162
-0.9000	-1.11976951	-1.11976951	0.0000000000000000000000000000000000000000
-0.8000	-0.92729522	-0.92729522	-0.0000000000000001110223024625156540423632
-0.7000	-0.77539750	-0.77539750	0.0000000000000000000000000000000000000000
-0.6000	-0.64350111	-0.64350111	0.0000000000000000000000000000000000000000
-0.5000	-0.52359878	-0.52359878	0.0000000000000000000000000000000000000000
-0.4000	-0.41151685	-0.41151685	0.0000000000000000000000000000000000000000
-0.3000	-0.30469265	-0.30469265	0.0000000000000000000000000000000000000000
-0.2000	-0.20135792	-0.20135792	-0.0000000000000000277555756156289135105908
-0.1000	-0.10016742	-0.10016742	0.0000000000000000000000000000000000000000
-0.0000	-0.00000000	-0.00000000	0.0000000000000000000000000000000000000000
0.1000	0.10016742	0.10016742	0.0000000000000000000000000000000000000000
0.2000	0.20135792	0.20135792	0.0000000000000000277555756156289135105908
0.3000	0.30469265	0.30469265	0.0000000000000000555111512312578270211816
0.4000	0.41151685	0.41151685	0.0000000000000000000000000000000000000000
0.5000	0.52359878	0.52359878	0.0000000000000000000000000000000000000000
0.6000	0.64350111	0.64350111	0.0000000000000000000000000000000000000000
0.7000	0.77539750	0.77539750	0.0000000000000000000000000000000000000000
0.8000	0.92729522	0.92729522	-0.0000000000000001110223024625156540423632
0.9000	1.11976951	1.11976951	0.0000000000000000000000000000000000000000
1.0000	1.57079631	1.57079631	0.0000000015553551779845520286471582949162

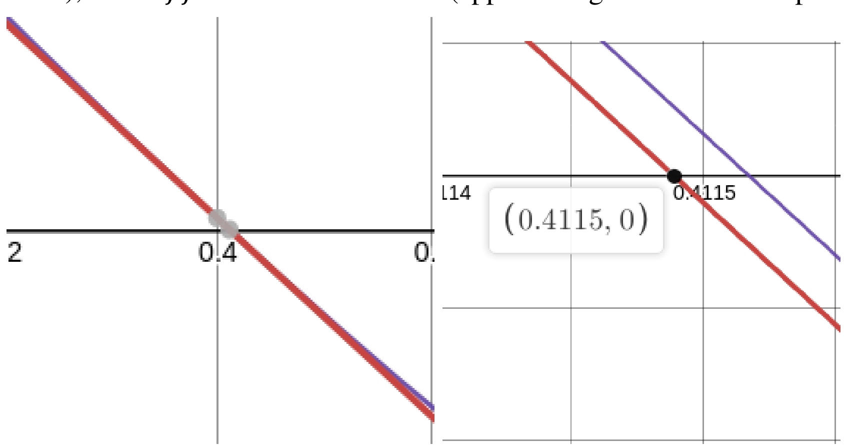
➔ Possible Explanation [applies to x = -1 as well]

- For the arcSin(x), the $\lim_{x \to 1^-} x_k + \frac{(z - sin(x_k))}{cos(x_k)} = \infty$, meaning that it has vertical asymptotes at the edge x = 1 [same with x = 1].
- Newton's method depends on taking the initial guess and finding the intersection of the tangent line and the x axis and repeating the process.
- Thus, if the slope is too steep at initial value then the next value would be extremely close to the first one and thus the difference between x_{k+1} and x_k would approach εvery quickly.

- Since it is implemented using the inverse method ($x_{k+1} = x_k + \frac{(z-sin(x_k))}{cos(x_k)}$), the initial guess gives extremely close value to root and all the values added to x_{k+1} approach epsilon εvery quickly and thus it has a minor difference at the edges since not many calculations take place.
- Example: To find value of arcsin(0.4), we need to find roots of 0.4-sin(x) [red: tangent line, purple: graph of 0.4-sin(x) for the graphs below]
    1. Let the initial value $x_k = 0$. Now we need to find where the tangent line of equation y = mx +b where m = slope of 0.4-sin(x_k), and b is 0.4-sin(x_k). From the graph below we can see that it intersects at 0.4. Now we need to repeat the process with $x_{k+1} = 0.4$. After this step ($x_k = 0$), $x_{k+1} = 0.4$. $diff = 0.4$



    2. After this step ($x_k = 0.4$), $x_{k+1} = 0.411488552615$ (extremely close to original value), and $diff = 0.0114885526151$ (approaching faster towards epsilon)



    3. After this step (0.411488552615), $x_{k+1} = 0.411516845893$, and ε = 0.0000282932778101
    4. After this step (0.411516845893), $x_{k+1} = 0.411516846067$, and ε = $1.7448801774 \times 10^{-10}$
- Hence, it only took 4 steps to reach epsilon. Thus, if the initial value has a steep slope then the calculation would not be as accurate as x is only calculated some times.

Analysis of arcCos(x) vs <math.h> acos(x) results

1. Taylor Series Edge Differences [I changed my implementation to use inverse method later]
    - Since Taylor Series of arccos(x) is given by $arccos(x) = \frac{\pi}{2} - arcsin(x) = \frac{\pi}{2} - \sum_{k=0}^{\infty} \frac{(2k)!}{2^{2k} * (k!)^2} * \frac{x^{2k+1}}{2k+1}$, $|x| <= 1$, the analysis of arccos(x) is similar to that of arcsin(x).

2. With Inverse Method
    - The inverse method explanation for arcSin(x) holds true for arcCos(x) as well. The only difference is the equation used for arcCos(x) namely $x_{k+1} = x_k + \frac{(cos(x_k) - z)}{sin(x_k)}$.

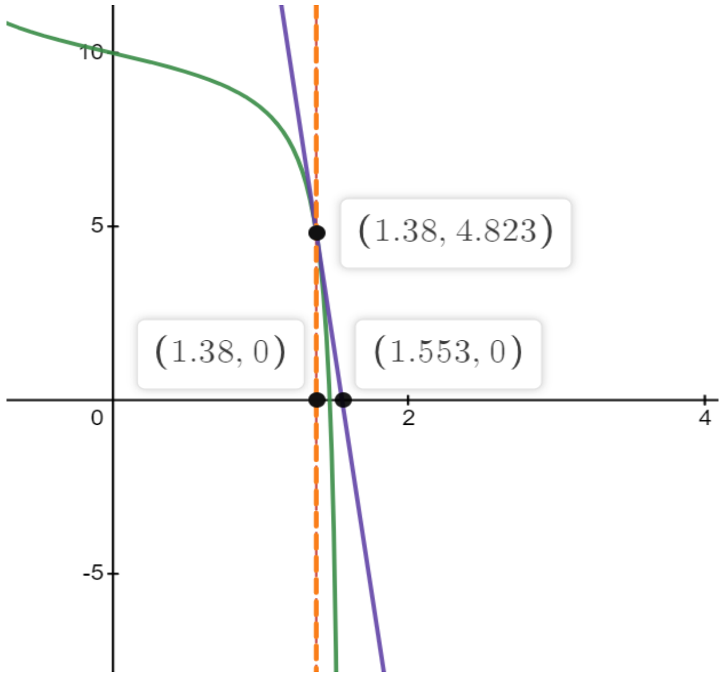
Analysis of arcTan(x) vs <math.h> atan(x) results

1. With Inverse Method [I used the Square Root method. This is just an analysis]
    - The arcTan(x) is calculated using the formula $x_{k+1} = x_k + \frac{z - tan(x_k)}{tan'(x_k)}$, where z is the input value of arcTan(x).
    - The arcTan(x) had interesting results based on the initial guess.
    - I chose the initial guess in three ways: value near x, midpoint from start to end point (in this case 5.5), and hardcoded.
    - The first two cases fail to compute values of x as x increases (see example later). The hardcoded value only works for a certain range. In my case I test from 1.2 to 1.41. The initial values in the range 1.38-1.41 work for calculating the result for arcTan(x) if and only if $1 \leq x \leq 10$.

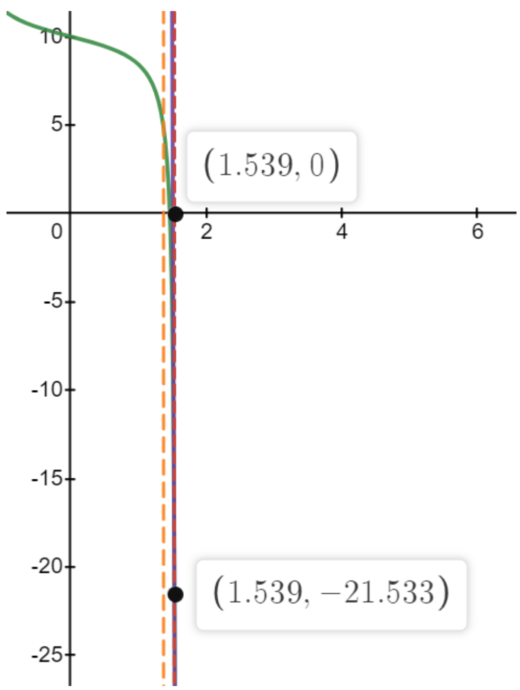
➔ Example 1
- Initial guess 1.38, x in arcTan(x) is 10. Actual value of arcTan(10) = 1.4711276743.
- In the graphs below the red dotted line represents the the line $x = x_k$, the green line is the graph of 10 - tan(x) [the equation's root to be calculated], the purple line is the tangent line to 10 - tan(x), and the orange dotted line x = 1.38 represents the initial guess. Also, domain shown in the graph is $-\frac{\pi}{2} \leq x \leq \frac{\pi}{2}$.

After Step:

1. [x_k was 1.38] x_{k+1} = 1.55343692924, ε = 4.4999999.



2. [x_k was 1.55343692924] x_{k+1} = 1.1.53909420311, ε = −0.0143427261339.



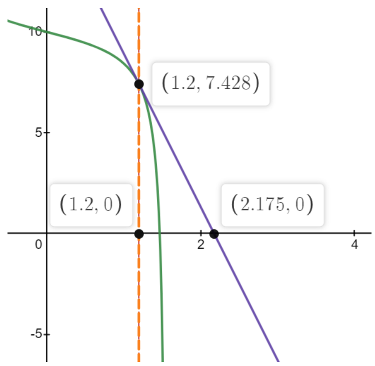
- If we continue this after step 9, we get [x_k was 1.47112813658] x_{k+1} = 1.4711276743, ε = $-6.2653160979\times10^{-12}$.

➔ Example 2 (initial guess less than 1.38)

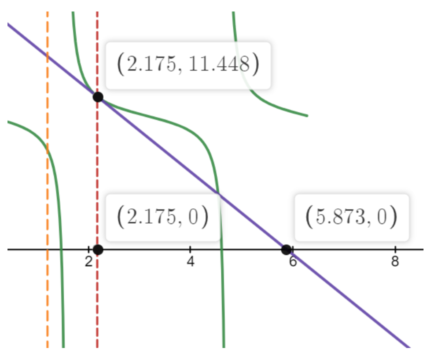
- Initial guess 1.2, x in arcTan(x) is 10. Actual value of arcTan(10) = 1.4711276743.
- In the graphs below the red dotted line represents the the line $x = x_k$, the green line is the graph of 10 - tan(x)[the equation's root to be calculated], the purple line is the tangent line to 10 - tan(x), and the orange dotted line x = 1.2 represents the initial guess. Also, domain shown in the graph is $-\frac{\pi}{2} \leq x$.

After Step:

1. [x_k was 1.2] x_{k+1} = 2.17529983202, ε = 0.975299832018.

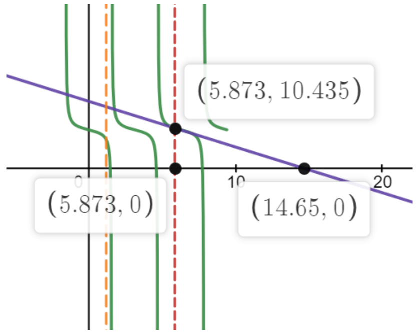


2. [x_k was 2.17529983202] x_{k+1} = 5.87319091023, ε = 3.69789107821.



Note that instead of intersecting within domain $-\frac{\pi}{2} \leq x \leq \frac{\pi}{2}$, the tangent line intersects at 2.175 which is outside of arctan(x)'s range.

3. [x_k was 5.87319091023] x_{k+1} = 14.6499070052, ε = 8.77671609501.



- If we continue this process, we would never get a definite value arctan(10) with our initial guess being less than 1.38 (in this case 1.2).

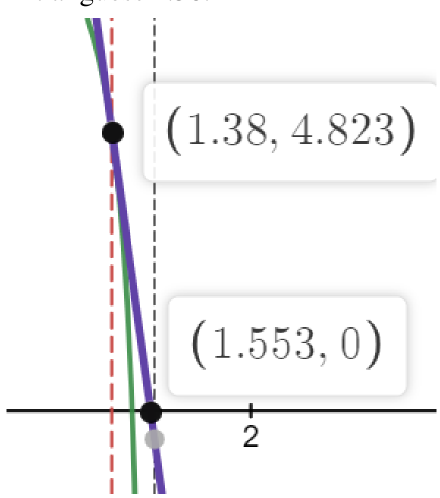
➔ Possible Explanation

- Why would we never reach a definite value with the wrong initial guess?
  - tan(x) is not a continuous graph since it has vertical asymptotes at $x = \frac{n\pi}{2}$where $n \in odd\ integers$.Hence, if the initial guess leads to the point intersecting with x-axis outside its asymptotes, Newton's method would always diverge due to jumping from one range to another.

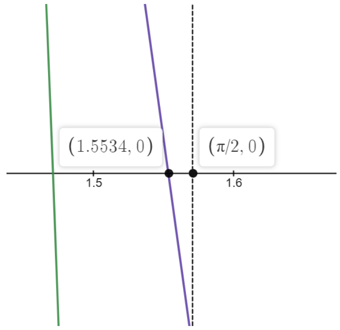
- Why initial guesses less than 1.38 specifically?
  - To tell the truth, there is nothing special about 1.38. From the above answer, it can be noted that any initial guess that makes the tangent line at that initial point intersect outside of the vertical asymptotes would diverge.

- Examples
  - Note: In both examples below I am calculating the roots of 10 - tan(x) to find value for arctan(10). The only addition to above graphs is the black dotted lines which represent the asymptotes of tan(x) [everything else is the same].

1. Initial guess 1.38.

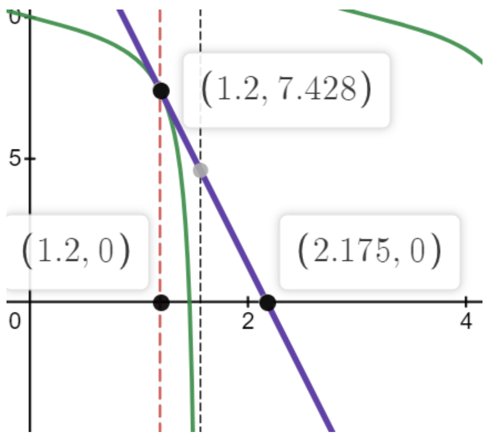


Zoomed in:



- From the zoomed in version it can easily be seen that the tangent line intersects the x-axis before the asymptote and thus Newton's method would converge.

2. Initial guess 1.2.



- From the above graph, we can see that the initial guess intersects outside of the asymptote and thus would diverge.

➔ Solution

- Thus, we know that the initial guess should be such that the tangent line intersects before the next asymptote (in this case before $\frac{\pi}{2}$ or after $\frac{-\pi}{2}$ since the range of arctan(x) is ($\frac{-\pi}{2}$, $\frac{\pi}{2}$).
- The tangent line is calculated using the formula $y_t = mx + b$, where $m = tan'(x_k)$, $x = (x - x_k)$, $b = (z - tan(x_k))$, where z is the input value of arctan(x).

Finding solution for initial guess $x_k = x_0$

When y_t intersects the x-axis, we get:

$$0 = mx + b$$

$$\therefore 0 = -\tan'(x_0)(x - x_0) + (z - \tan(x_0))$$

$$\therefore 0 = -\tan'(x_0)(x - x_0) + (z - \tan(x_0))$$

$$\therefore \tan(x_0) - z = -\tan'(x_0)(x - x_0)$$

$$\therefore \frac{\tan(x_0) - z}{\tan'(x_0)} = -x + x_0$$

$$\therefore \frac{\tan(x_0) - z}{\tan'(x_0)} - x_0 = -x$$

$$\therefore x = x_0 - \frac{\tan(x_0) - z}{\tan'(x_0)}$$

Thus we need to find $x = x_0 - \frac{\tan(x_0) - z}{\tan'(x_0)}$ such that $x < \pi/2$

$$\therefore x_0 - \frac{(\tan(x_0) - z)}{\tan'(x_0)} < \pi/2$$

$$\therefore x_0 < \pi/2 + \frac{(\tan(x_0) - z)}{\tan'(x_0)}$$

$$\therefore \frac{\pi}{2} + \frac{(\tan(x_0) - z)}{\tan'(x_0)} > 1$$

As long as the above inequality holds true, we can assure that the method would always converge.

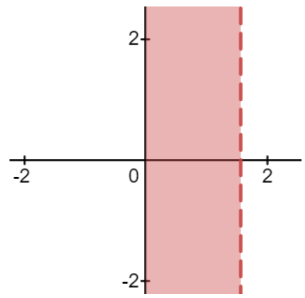
NOTE: the final inequality should be divided by x_0 on the left hand side

- NOTE: The above analysis and equality also holds true for $x < 0$ inequality. The only difference, however, would be that the initial guess would have to be $-x_0$ after x_0 satisfies the inequality. Another solution would be to find arctan(z) and return -arctan(z) if $z < 0$ since arctan(z) is symmetric.

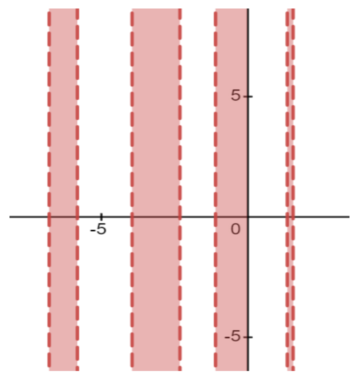
- Methods for finding x_0
    1. Random Guessing
        - A trivial solution is to guess the initial values randomly and hope that the inequality is satisfied.

    2. Analytical Approach
        - Graphing the region of inequality:

        a. $z = 1$

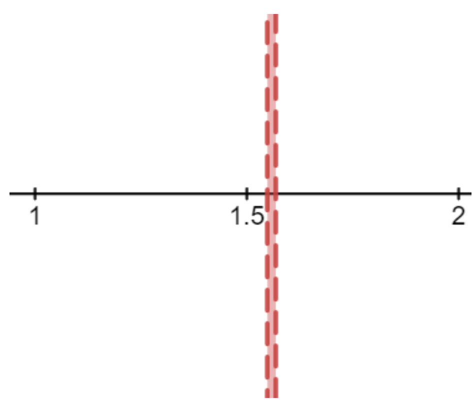


b. $z = 10$



- Note it can be seen that as z increases by a factor of 10 the valid initial guess region decreases exponentially. Thus, there is a relation between the z and the initial guess. [The exponentiality would become pertinent later]

c. $z = 1000$



- The region grows shrink more and more towards $\pi/2$. Reason: Note that as z increases the graph z - tan(x) moves up the y-axis and thus the tangents to the initial guess need to become steeper and steeper to intersect with the x-axis before $\pi/2$(as mentioned earlier).
- No matter how much z increases if we pick a value close enough to $\pi/2$, we should be able to get a valid output.
- But how close? Recall that earlier we noted that the area for initial guess is related with the value of z. As z increases exponentially, the area decreases simultaneously.
- More analysis:

- I manually tried some values to get some sort of formula. Examples:
    - Note: We subtract the guess from $\pi/2$ since the region shrinks near $\pi/2$.

    - Table:

z	x_k	Inequality
1	$\frac{\pi}{2} - 1\ or\ (\frac{\pi}{2} - 10^{0})$	Satisfied
1	$\frac{\pi}{2} - 0.1\ or\ (\frac{\pi}{2} - 10^{-1})$	Satisfied
10	$\frac{\pi}{2} - 1\ or\ (\frac{\pi}{2} - 10^{0})$	Unsatisfied
10	$\frac{\pi}{2} - 0.1\ or\ (\frac{\pi}{2} - 10^{-1})$	Satisfied
100	$\frac{\pi}{2} - 0.1\ or\ (\frac{\pi}{2} - 10^{-1})$	Unsatisfied
100	$\frac{\pi}{2} - 0.1\ or\ (\frac{\pi}{2} - 10^{-2})$	Satisfied
1000	$\frac{\pi}{2} - 0.1\ or\ (\frac{\pi}{2} - 10^{-2})$	Unsatisfied
1000	$\frac{\pi}{2} - 0.1\ or\ (\frac{\pi}{2} - 10^{-3})$	Satisfied

- I tried many more values and noted that as long as we choose $10^{-log(z)}$ (here's where the exponential growth and shrink come into play), the inequality was satisfied (of course after subtracting from $\frac{\pi}{2}$).
- Thus, the formula $x_k = \frac{\pi}{2} - 10^{-\log(z)}$ should satisfy the inequality and thus we should get a convergent outcome using Newton's method.
- For smaller values, though, log(x) would be asymptotic and thus the initial value guess would result in negative values. To tackle this problem, one could use the modulo function. Thus, the final formula for initial value x_k mod($\frac{\pi}{2} - 10^{-\log(z)}$, $\frac{\pi}{2}$) assuming mod returns value between 0 and $\frac{\pi}{2}$.

2. With Square Root Identity
    a. Table:

x	arcTan	Library	Difference
1.0000	0.78539816	0.78539816	0.0000000000000001110223024625156540423632
1.1000	0.83298127	0.83298127	-0.0000000000000001110223024625156540423632
1.2000	0.87605805	0.87605805	-0.0000000000000001110223024625156540423632
1.3000	0.91510070	0.91510070	-0.0000000000000001110223024625156540423632
1.4000	0.95054684	0.95054684	-0.0000000000000001110223024625156540423632
1.5000	0.98279372	0.98279372	0.0000000000000000000000000000000000000000
1.6000	1.01219701	1.01219701	-0.0000000000000002220446049250313080847263
1.7000	1.03907226	1.03907226	0.0000000000000000000000000000000000000000
1.8000	1.06369782	1.06369782	0.0000000000000000000000000000000000000000
1.9000	1.08631840	1.08631840	0.0000000000000000000000000000000000000000
2.0000	1.10714872	1.10714872	0.0000000000000000000000000000000000000000
2.1000	1.12637712	1.12637712	0.0000000000000000000000000000000000000000
2.2000	1.14416883	1.14416883	0.0000000000000000000000000000000000000000
2.3000	1.16066899	1.16066899	0.0000000000000000000000000000000000000000
2.4000	1.17600521	1.17600521	0.0000000000000000000000000000000000000000
2.5000	1.19028995	1.19028995	0.0000000000000000000000000000000000000000
2.6000	1.20362249	1.20362249	0.0000000000000000000000000000000000000000
2.7000	1.21609067	1.21609067	0.0000000000000000000000000000000000000000
2.8000	1.22777239	1.22777239	0.0000000000000000000000000000000000000000
2.9000	1.23873686	1.23873686	0.0000000000000000000000000000000000000000
3.0000	1.24904577	1.24904577	0.0000000000000000000000000000000000000000
3.1000	1.25875421	1.25875421	0.0000000000000000000000000000000000000000
3.2000	1.26791146	1.26791146	0.0000000000000000000000000000000000000000
3.3000	1.27656176	1.27656176	0.0000000000000000000000000000000000000000
3.4000	1.28474489	1.28474489	0.0000000000000000000000000000000000000000
3.5000	1.29249667	1.29249667	0.0000000000000000000000000000000000000000
3.6000	1.29984948	1.29984948	0.0000000000000000000000000000000000000000
3.7000	1.30683260	1.30683260	0.0000000000000000000000000000000000000000
3.8000	1.31347261	1.31347261	0.0000000000000000000000000000000000000000
3.9000	1.31979364	1.31979364	0.0000000000000000000000000000000000000000
4.0000	1.32581766	1.32581766	0.0000000000000000000000000000000000000000
4.1000	1.33156473	1.33156473	0.0000000000000000000000000000000000000000
4.2000	1.33705315	1.33705315	0.0000000000000000000000000000000000000000
4.3000	1.34229969	1.34229969	0.0000000000000000000000000000000000000000
4.4000	1.34731973	1.34731973	0.0000000000000000000000000000000000000000
4.5000	1.35212738	1.35212738	0.0000000000000000000000000000000000000000
4.6000	1.35673564	1.35673564	0.0000000000000000000000000000000000000000
4.7000	1.36115648	1.36115648	0.0000000000000000000000000000000000000000
4.8000	1.36540094	1.36540094	0.0000000000000000000000000000000000000000
4.9000	1.36947922	1.36947922	0.0000000000000000000000000000000000000000
5.0000	1.37340077	1.37340077	0.0000000000000000000000000000000000000000
5.1000	1.37717433	1.37717433	0.0000000000000000000000000000000000000000
5.2000	1.38080804	1.38080804	0.0000000000000000000000000000000000000000
5.3000	1.38430943	1.38430943	0.0000000000000000000000000000000000000000
5.4000	1.38768551	1.38768551	0.0000000000000002220446049250313080847263
5.5000	1.39094283	1.39094283	0.0000000000000000000000000000000000000000
5.6000	1.39408747	1.39408747	0.0000000000000000000000000000000000000000
5.7000	1.39712513	1.39712513	0.0000000000000000000000000000000000000000
5.8000	1.40006112	1.40006112	0.0000000000000000000000000000000000000000
5.9000	1.40290040	1.40290040	0.0000000000000002220446049250313080847263

➔ Possible Explanation

- The arcTan(x) using the inverse method twice, namely to calculate the square root of 1+ x^2 and to calculate the arcCos(x). Thus, the analysis for arcSin(x)'s inverse method also holds true for arcTan(x).
- However, the arcCos(x) is being passed $\frac{1}{\sqrt{1+x^2}}$ as a parameter and arcCos itself is calculated using cos(x) where x is again $\frac{1}{\sqrt{1+x^2}}$. From the graph it could be seen that it has an horizontal asymptote if the input is and thus the output value would highly differ based on the initial guess as described in arcSin(x).

Analysis of Log(x) vs <math.h> log(x) results

1. With Inverse Method

x	Log	Library	Difference
1.0000	0.00000000	0.00000000	0.00000000000000001024763767952619419392589
1.1000	0.09531018	0.09531018	0.0000000000130423311039962186441698577255
1.2000	0.18232156	0.18232156	0.0000000000257557308813716190343257039785
1.3000	0.26236426	0.26236426	0.0000000000128221322448496266588335856795
1.4000	0.33647224	0.33647224	0.00000000001125820547898115364660043269396
1.5000	0.40546511	0.40546511	0.0000000000229053442879489921324420720339
1.6000	0.47000363	0.47000363	0.00000000000946284162139932050195056945086
1.7000	0.53062825	0.53062825	0.00000000003013509441984751902054995298386
1.8000	0.58778666	0.58778666	0.0000000000423363566426360193872824311256
1.9000	0.64185389	0.64185389	0.00000000001060858068058223580010235309601
2.0000	0.69314718	0.69314718	0.00000000002358362394261348526924848556519
2.1000	0.74193734	0.74193734	0.0000000000293266522177759725309442728758
2.2000	0.78845736	0.78845736	0.0000000000582938142201783193740993738174
2.3000	0.83290912	0.83290912	0.00000000001080741052206235508492682129145
2.4000	0.87546874	0.87546874	0.00000000001889989276193659861746709793806
2.5000	0.91629073	0.91629073	0.00000000003145508298274535263772122561932
2.6000	0.95551145	0.95551145	0.0000000000366742192170477210311219096184
2.7000	0.99325177	0.99325177	0.0000000000586032333771413505019154399633
2.8000	1.02961942	1.02961942	0.0000000000904392116751751018455252051353
2.9000	1.06471074	1.06471074	0.00000000001353608336529532607528381049633
3.0000	1.09861229	1.09861229	0.00000000001971796059763164521427825093269
3.1000	1.13140211	1.13140211	0.00000000002803861587352685091900639235973
3.2000	1.16315081	1.16315081	0.0000000000322264437357944188988767564297
3.3000	1.19392247	1.19392247	0.0000000000451403359136293147457763552666
3.4000	1.22377543	1.22377543	0.0000000000620421491959177728858776390553
3.5000	1.25276297	1.25276297	0.0000000000838191738239402184262871742249
3.6000	1.28093385	1.28093385	0.00000000001114803804824759936309419572353
3.7000	1.30833282	1.30833282	0.00000000001461630816379511088598519563675
3.8000	1.33500107	1.33500107	0.00000000001891382606089564433204941451550
3.9000	1.36097655	1.36097655	0.00000000002418132361015068454435095191002
4.0000	1.38629436	1.38629436	0.0000000000280806489172391593456268310547
4.1000	1.41098697	1.41098697	0.0000000000357622820246206174488179385662
4.2000	1.43508453	1.43508453	0.0000000000450799397810897062299773097038
4.3000	1.45861502	1.45861502	0.0000000000562863089470511113177053630352
4.4000	1.48160454	1.48160454	0.0000000000696587232340561968157999217510
4.5000	1.50407740	1.50407740	0.0000000000854962767249389798962511122227
4.6000	1.52605630	1.52605630	0.00000000001041255970335441816132515668869
4.7000	1.54756251	1.54756251	0.00000000001258977366802582764648832380772
4.8000	1.56861592	1.56861592	0.00000000001511906155826636677375063300133
4.9000	1.58923521	1.58923521	0.00000000001804083549217239124118350446224
5.0000	1.60943791	1.60943791	0.00000000002139808330525738711003214120865
5.1000	1.62924054	1.62924054	0.0000000000255309107188850248348899185658
5.2000	1.64865863	1.64865863	0.0000000000303039815463534978334791958332
5.3000	1.66770682	1.66770682	0.0000000000357753826563111942959949374199
5.4000	1.68639895	1.68639895	0.0000000000420183887683833745541051030159
5.5000	1.70474809	1.70474809	0.0000000000491109375388987245969474315643
5.6000	1.72276660	1.72276660	0.0000000000571374059177287563215941190720
5.7000	1.74046617	1.74046617	0.0000000000661846133453991569695062935352
5.8000	1.75785792	1.75785792	0.0000000000763451524221636645961552858353
5.9000	1.77495235	1.77495235	0.0000000000877178329972139181336387991905
6.0000	1.79175947	1.79175947	0.00000000001004034633211858817958272993565
6.1000	1.80828877	1.80828877	0.00000000001145101791166780458297580480576
6.2000	1.82454929	1.82454929	0.00000000001301505569983874011086300015450
6.3000	1.84054963	1.84054963	0.00000000001474413924285045141004957258701
6.4000	1.85629799	1.85629799	0.00000000001665052540289480020874179899693
6.5000	1.87180218	1.87180218	0.00000000001874687072245251329150050878525

- ➔ Possible Explanation:

- The arcTan(x) using the inverse method using the formula $x_{k+1} = x_k + \frac{(y - e^{x_k})}{e^{x_k}}$. Thus, the analysis for arcSin(x)'s inverse method also holds true for arcTan(x). However, the major differences are due to implementation of e^x.
- e^x is calculated using the Taylor series $\frac{x^k}{k!}$ which is centered around zero. Thus, as x increases, the series differs from the original values pretty fast while the Taylor series itself converges as k increases.
- Hence, the Log(x) implementation has significant differences from the library implementation.