What Lies Between The Roots

Pratham Prakash Gupta(IHM2017003), Aditya Gupta(IHM2017005),
Priyal Gupta(IHM2017004) and Rohit Ranjan(IHM2017001)

25 March 2019

Abstract: **This paper is used to find the value of $\sqrt{2}$ and $\sqrt{5}$ using variable tangent method, where we first make an approximation of the value and then we make use of the formula x(n+1)=x(n)-f(x)/f'(x)**

Keywords: Variable Tangent Method.

1 Introduction and Literature Survey

1.1 Question

Find the the value of $\sqrt{2}$ and $\sqrt{5}$ using variable tangent method. Also plot the following graphs:- Accuracy vs time and Accuracy vs number of iterations.

1.2 Introduction

Variable tangent method is a method for finding successively better approximations to the roots (or zeroes) of a real-valued function. It is one example of a root-finding algorithm.

1.3 Idea

The idea of the method is as follows: one starts with an initial guess which is reasonably close to the true root, then the function is approximated by its tangent line (which can be computed using the tools of calculus), and one computes the x-intercept of this tangent line (which is easily done with elementary algebra). This x-intercept will typically be a better approximation to the function's root than the original guess, and the method can be iterated.

2 Algorithm Design

2.1 Variable and Parameters

EPSILON: 0.0001

2.2 Pseudo Code

Algorithm 1
func(float x,int y)

```
return x*x-y
```

Algorithm 2
derivFunc(double x)

return 2*x

Algorithm 3
varTangent(float x,int y)

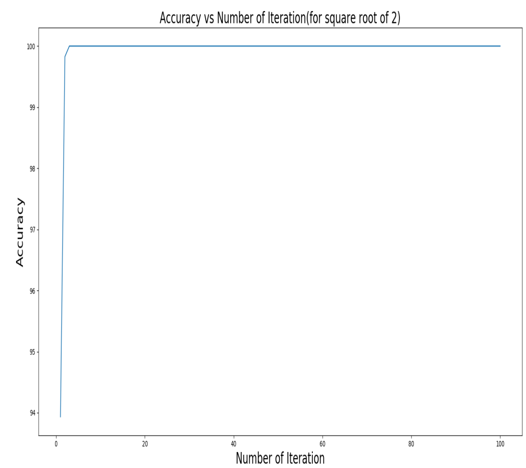
h← $func(x, y)/derivFunc(x)$
while abs(h)≥ $EPSILON$ **do**
h ← $func(x, y)/derivFunc(x)$
x ← $x - h$
print(x)

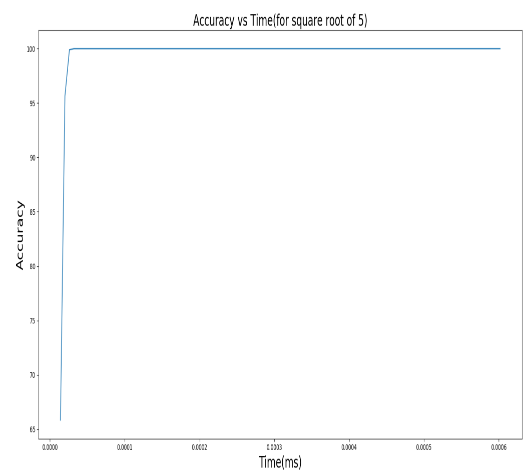
Algorithm 4
main()

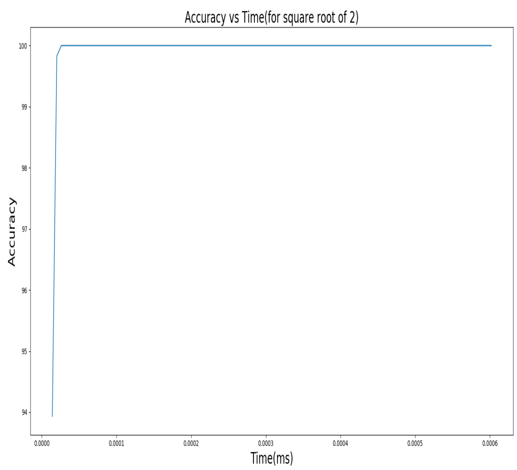
x0← 1.0
varTangent(x0,2)
varTangent(x0,5)
return 0

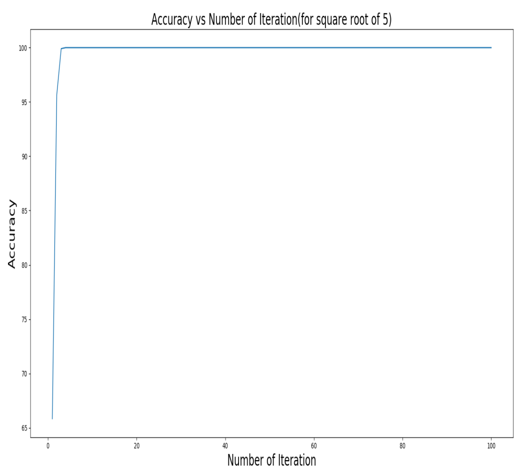
3 Time Complexity

The time complexity is the computational complexity that describes the amount of time it takes to run an algorithm. Time complexity is commonly estimated by counting the number of elementary operations performed by the algorithm, supposing that each elementary operation takes a fixed amount of time to perform. Thus, the amount of time taken and the number of elementary operations performed by the algorithm are taken to differ by at most a constant factor. Here, the worst case and best case complexity both comes out to be O(log(n).F(n)), where F(n) is the the cost of calculating f(x)/f'(x), with n-digit precision.









4 Space Complexity

The Space Complexity of an algorithm is the maximum amount of space used at any one step. The above code does not use any array, so, the space complexity of this algorithm comes out to be O(1).

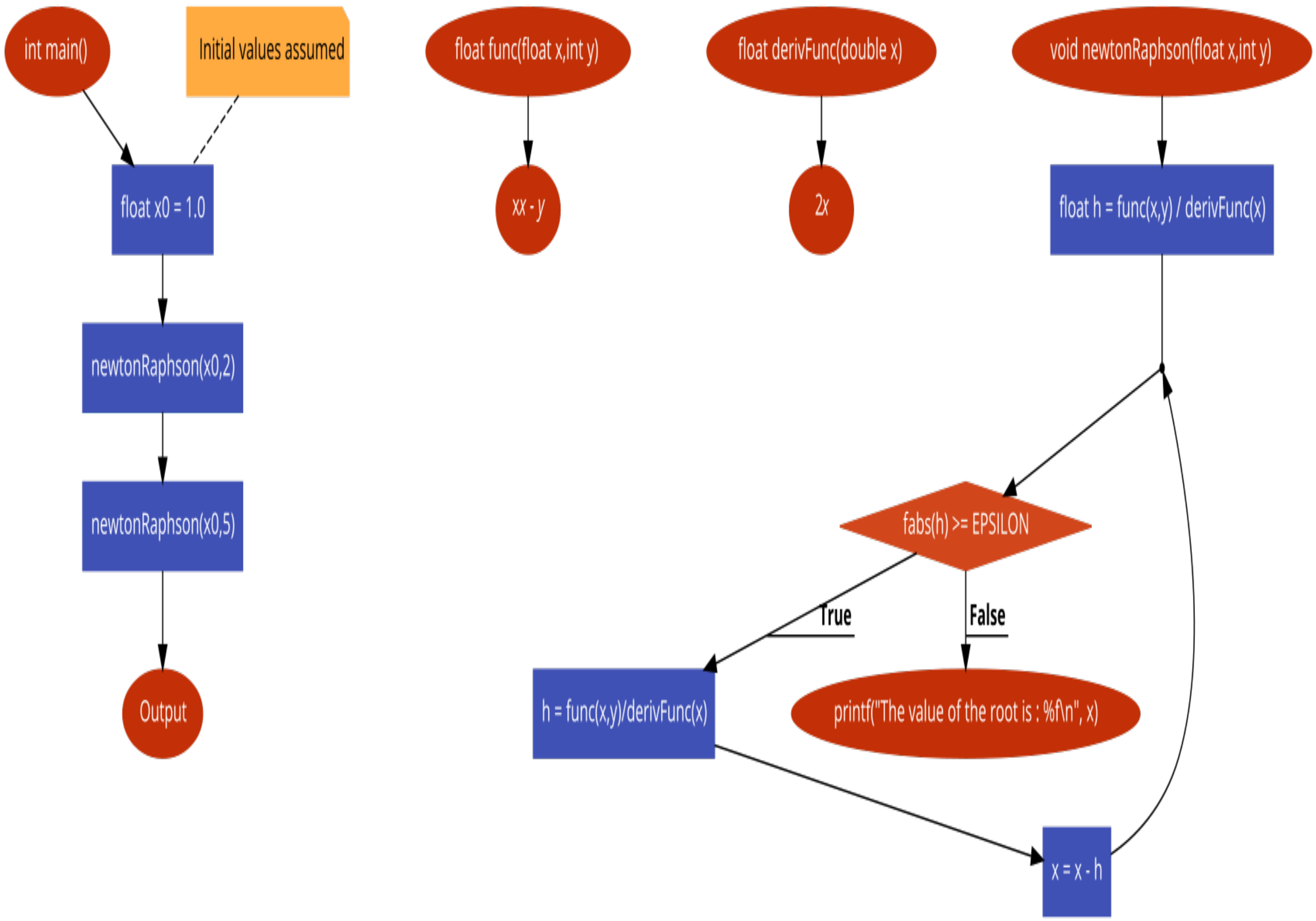
5 Experimental Setup

Language Used :- C++
Plotting tool :- matplotlib
Report Making Tool :- TextPortable (Latex)

6 Conclusion

We had to find the the value of $\sqrt{2}$ and $\sqrt{5}$ using variable tangent method and also plot the following graphs:- Accuracy vs time and Accuracy vs number of iterations. In the above algorithm we started with an initial guess which is reasonably close to the true root, then the function is approximated by its tangent line (which can be computed using the tools of calculus), and the x-intercept of this tangent line is computed (which is easily done with elementary algebra). This x-intercept is a better approximation to the the function's root than the original guess, and the method is then iterated.In this way we can find the square root of 2 and 5 easily, and the above method is known as variable tangent method. This method has a worst case and best case complexity both comes out to O(log(n).F(n)), where F(n) is the the cost of calculating f(x)/f'(x), with n-digit precision.

7 Flow Chart



8 References

Gnu plot :- https://matplotlib.org
Flow Chart :- https://code2flow.com
Concept :-

- https://www.geeksforgeeks.org/program-for-newton-raphson-method/
- https://en.wikipedia.org/wiki/Newton%27s_method