Analysis of Solution of Numerical Problem Using Maxima and Python Software

Varsha J. Patil*, Mayuri B. Babar, Gitanjali N. Pawar

Department of Mathematics, Dr. D. Y. Patil Arts, Commerce & Science College, Pimpri, Pune-18, Maharashtra, India

DOI: https://doi.org/10.51583/IJLTEMAS.2025.1413SP046

Received: 26 June 2025; Accepted: 30 June 2025; Published: 27 October 2025

Abstract: The goal of this research paper is to look into how well the Newton-Raphson method works for solving transcendental equations using two different computer programs, Python and Maxima. The Newton-Raphson method converges faster than the bisection method and Regular Falsi method. This is why this iterative method is often used to find the roots of functions with real values. We test the algorithm's speed, accuracy, and ease of use on two different platforms: Maxima, a computer algebra system and Python, a general-purpose programming language with strong numerical libraries. The study gives error analysis, convergence behavior and step-by-step instructions for solving a few selected nonlinear equations. The results show that both tools are good for analyzing numbers, but they have different advantages when it comes to computational control, flexibility, visualization and syntax simplicity. This comparative analysis helps scientists, engineers, teachers and students choose the right tools for solving mathematics problems in science and engineering.

Keywords: Numerical Analysis, Newton's Raphson Method, Python, Maxima

I. Introduction:

Numerical methods are very important when it comes to solving math problems that are hard or impossible to do analytically. The Newton-Raphson method is one of the most effective techniques for solving nonlinear equations due to its ease of implementation and high rate of convergence, making it a popular choice in applied mathematics and engineering. It turns the problem into a process that uses the function and its derivative over and over again. As numerical methods have gotten better, so have computational tools, which make it easier to use and analyze them. Maxima is a symbolic algebra system that provides a high-level interface for both analytical and numerical calculations. But Python has become the most popular language for scientific computing because of libraries that make numerical analysis easier.

This paper concentrates on the utilization of the Newton-Raphson method through Maxima and Python to resolve specific nonlinear equations. The objective is to evaluate the precision, convergence and computational efficiency provided by both software applications. We want to show the pros and cons of these tools for solving numerical problems by using specific examples and comparing them. This will give educators, students and researchers in computational mathematics useful information.

Problem:

Find the solution of the equation $e^x = 4x$ correct up to three decimal places. The initial approximation of root of the given equation is $x_0 = 2$.

II. Methodology:

Given equation is transcendental equation. There is no method to find the exact solution of this equation. So, we use numerical methods to find the solutions of such type of equations. Numerical methods give us approximate solution. We are using Newton – Raphson Method to find the approximate solution of the equation. Also, we will find the solution of the given equation using Python and Maxima Software.

Solution using Newton – Raphson Method

Given equation is $f(x) = e^x - 4x = 0$

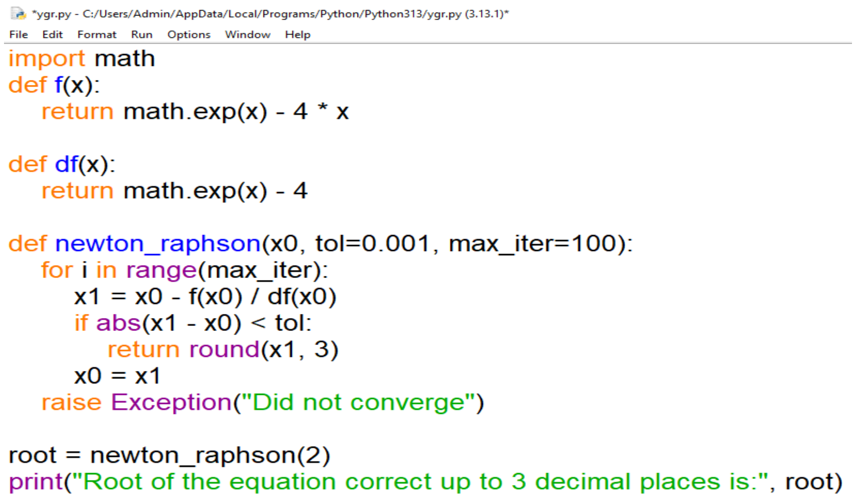
Root of this equation lies in the interval $(2,3)$ as $f(2).f(3) < 0$. We take the initial approximation $x_0 = 2$.

Newton – Raphson formula is $x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$.

Using this formula the solution of given equation corrects up to three decimal places is 2.153.

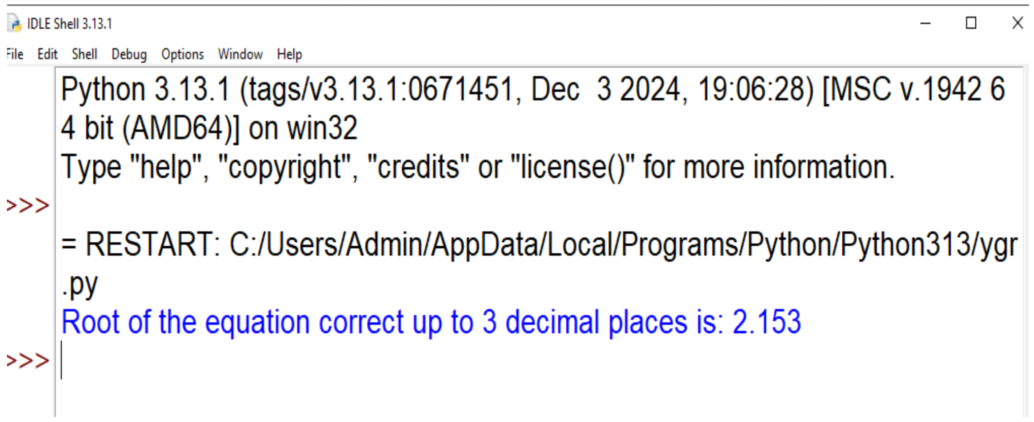
Solution using Python

INPUT


ygr.py - C:/Users/Admin/AppData/Local/Programs/Python/Python313/ygr.py (3.13.1)
File Edit Format Run Options Window Help

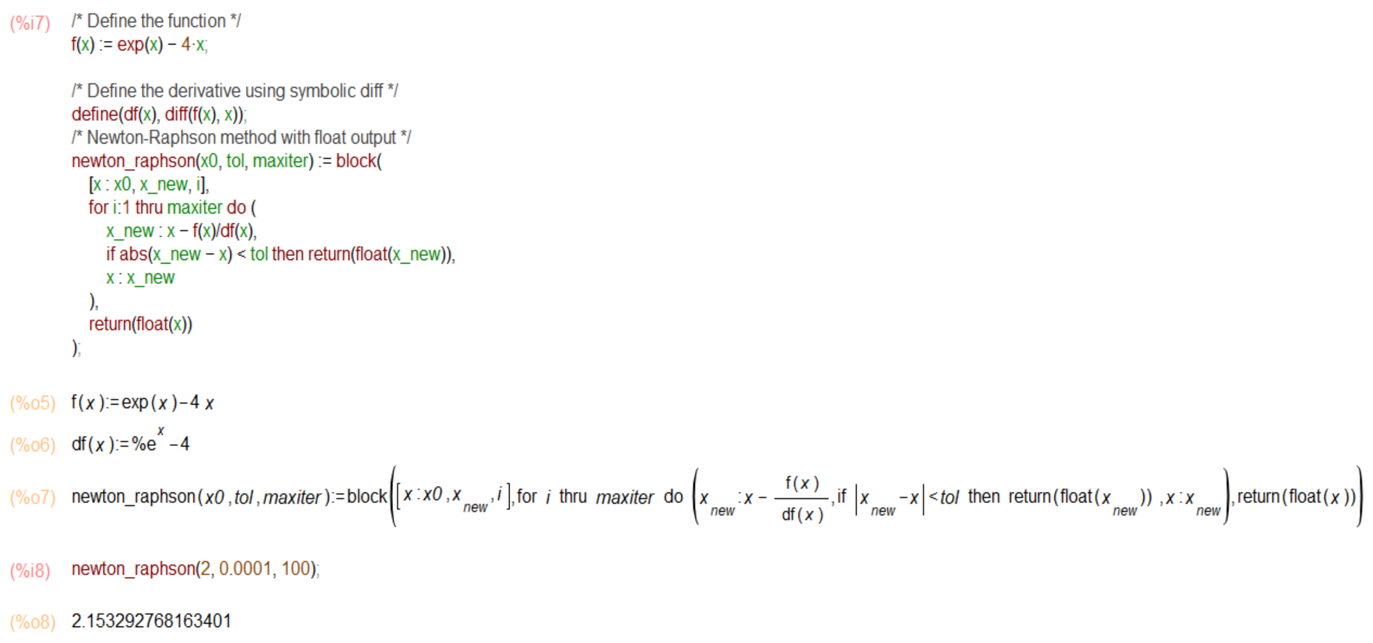
```
import math
def f(x):
    return math.exp(x) - 4 * x

def df(x):
    return math.exp(x) - 4

def newton_raphson(x0, tol=0.001, max_iter=100):
    for i in range(max_iter):
        x1 = x0 - f(x0) / df(x0)
        if abs(x1 - x0) < tol:
            return round(x1, 3)
        x0 = x1
    raise Exception("Did not converge")

root = newton_raphson(2)
print("Root of the equation correct up to 3 decimal places is:", root)
```

OUTPUT



```
Python 3.13.1 (tags/v3.13.1:0671451, Dec  3 2024, 19:06:28) [MSC v.1942 6
4 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/Admin/AppData/Local/Programs/Python/Python313/ygr
.py
Root of the equation correct up to 3 decimal places is: 2.153
>>>
```

Solution using Maxima



```
(%i7) /* Define the function */
      f(x) := exp(x) - 4·x;

      /* Define the derivative using symbolic diff */
      define(df(x), diff(f(x), x));
      /* Newton-Raphson method with float output */
      newton_raphson(x0, tol, maxiter) := block(
         [x : x0, x_new, i],
         for i:1 thru maxiter do (
            x_new : x - f(x)/df(x),
            if abs(x_new - x) < tol then return(float(x_new)),
            x : x_new
         ),
         return(float(x))
      );
```

(%o5) $f(x):=\exp(x)-4\,x$

(%o6) $df(x):=\%e^{x}-4$

(%o7) $\mathrm{newton\_raphson}(x0,tol,maxiter):=\mathrm{block}\left(\left[x:x0,x_{new},i\right],\mathrm{for}\ i\ \mathrm{thru}\ maxiter\ \mathrm{do}\ \left(x_{new}:x-\frac{f(x)}{df(x)},\mathrm{if}\ \left|x_{new}-x\right|<tol\ \mathrm{then}\ \mathrm{return}(\mathrm{float}(x_{new})),x:x_{new}\right),\mathrm{return}(\mathrm{float}(x))\right)$

```
(%i8) newton_raphson(2, 0.0001, 100);

(%o8) 2.153292768163401
```

III. Result:

Method: Newton – Raphson method

Given equation $e^x - 4x = 0$

Solution: 2.153 correct up to three decimal places.

We got the same solution in Maxima software and Python.

IV. Conclusion:

In this research paper, we looked at and compared how well Maxima and Python software work for solving a variety of numerical problems. The comparison showed that Maxima is best for educational and theoretical purposes where exact answers are needed. Python, on the other hand, is better for real-world numerical modeling and big calculations. Overall, using both Maxima and Python in the computational workflow can make solving problems easier by finding a balance between symbolic computation and high-performance numerical analysis.

References:

1. Numerical Methods, S. R. K. Iyengar, R. K. Jain
2. Maxima Software, Dr. Kalyanrao Takale, Golden Series
3. Introduction to Machine Learning with Python