Project 1

ByeongKyu Park

The Algorithm

Implemented in Python, this method combines the secant and tangent approaches for finding square roots.

```
# Python code for the combined root-finding method
def combined_root_finding(N, intervals):
    # Definitions of f(x) and df(x) omitted for brevity
    x0, x1 = initial_x0, initial_x1
    for i in range(num_iter):
        # Calculation of x_secant and x_tangent
        x_new = (x_secant + x_tangent) / 2
        if i in intervals:
            approximations[i] = x_new
        x0, x1 = sorted([x_new, x0, x1], key=lambda x: abs(x - x_new))[:2]
    return approximations
```

Explanation

The primary motivation behind the development of this combined root-finding method was to improve the efficiency and convergence speed of traditional methods, particularly for functions where computing derivatives analytically is costly or impractical. Traditional root-finding methods, such as the secant and Newton's method, each have their advantages and limitations. The secant method, while not requiring the derivative of the function, can sometimes converge more slowly than Newton's method. On the other hand, Newton's method, which involves the function's derivative, converges more quickly but can be computationally expensive due to the necessity of calculating derivatives at each iteration. This algorithm introduces a hybrid approach that leverages the conceptual strengths of both the secant and tangent (Newton's) methods. The key observation is that the slopes of the secant line, defined by two points on the function, and the tangent line, at the midpoint of these two points, are mathematically identical for the specific function form $f(x) = x^2 - N$. This identity is derived from the fact that the slope of the secant line connecting x_0 and x_1 is given by $\frac{x_1^2 - x_0^2}{x_1 - x_0} = x_0 + x_1$, and the slope of the tangent at the midpoint is $2\left(\frac{x_0 + x_1}{2}\right) = x_0 + x_1$.

The iterative steps involve computing the x-intercepts (x_{sec} and $x_{\tan}$) of both the secant and tangent lines. For the secant line, the intercept x_{sec} is found using the formula:

$$0 = (x_1 + x_0)(x_{\text{sec}} - x_1) + (x_1^2 - N),$$

and for the tangent line at the midpoint, the intercept $x_{\tan}$ is calculated as:

$$0 = (x_0 + x_1)\left(x_{\tan} - \frac{x_0 + x_1}{2}\right) + \left(\left(\frac{x_0 + x_1}{2}\right)^2 - N\right).$$

Analysis

After conducting a Monte Carlo simulation with $n = 10\,000$, I observed that the modified method tends to yield a lower average error in the first few iterations compared to the traditional secant method. This suggests that the combined use of secant and tangent lines effectively accelerates the convergence process in the early stages.

Interval	Modified Secant's Average Error	Secant Method's Average Error
2	0.5084796670581864	0.7335521818290671
3	0.310602 2555102186	1.6469349397165645
4	0.1196650704267478	0.2783513649879494
5	0.0	0.0

Order of Convergence for Combined Root Finding Method

Considering the function $f(x) = x^2 - N$ with $N > 0$, we define the iterative process where x_2 is computed as the midpoint of x_0 and x_1. Let $k = x_1 - x_0$, and denote the x-intercepts of the secant line connecting x_0 and x_1, and the tangent at the midpoint, as x_{sec} and $x_{\tan}$ respectively. The expressions for x_{sec} and $x_{\tan}$ are given as:

$$x_{\text{sec}} = \frac{N - x_1^2}{x_0 + x_1} + x_1,$$
$$x_{\tan} = \frac{N - \left(\frac{x_0 + x_1}{2}\right)^2}{x_0 + x_1} + \frac{x_0 + x_1}{2}.$$

The new approximation x_2 can then be expressed as:

$$
\begin{aligned}
x_2 &= \frac{x_{\text{sec}} + x_{\tan}}{2} \\
&= \frac{2N - x_1^2 - \frac{1}{4}(x_0^2 + 2x_0x_1 + x_1^2)}{2(x_0 + x_1)} + \frac{1}{4}x_0 + \frac{3}{4}x_1 \\
&= \frac{2N + \frac{1}{4}x_0^2 + \frac{3}{2}x_0x_1 + \frac{1}{4}x_1^2}{2(x_0 + x_1)} \\
&= \frac{2p^2 + \frac{1}{4}x_0^2 + \frac{3}{2}x_0x_1 + \frac{1}{4}x_1^2}{2(x_0 + x_1)}
\end{aligned}
$$

Introducing $\hat{x} = |x - p|$, where p is the root

$$
\hat{x}_2 = \frac{2p^2 + \frac{1}{4}(\hat{x}_0 + p)^2 + \frac{1}{4}(\hat{x}_1 + p)^2 + \frac{3}{2}(\hat{x_0} + p)(\hat{x_1} + p)}{2(\hat{x}_0 + \hat{x}_1 + 2p)} - p
$$

$$
\hat{x}_2 = \left(\frac{4p^2 + \frac{1}{4}\hat{x}_1^2 + \frac{1}{4}\hat{x}_0^2 + 2\hat{x}_1p + 2\hat{x}_0p + \frac{3}{2}\hat{x}_0\hat{x}_1}{2(\hat{x}_0 + \hat{x}_1 + 2p)} \right) - p = p + \frac{1}{2}(\hat{x}_1 + \hat{x}_0) - p
$$

the error term for the new approximation simplifies to $\hat{x}_2 = \frac{1}{2}(\hat{x}_0 + \hat{x}_1)$. Defining $h_1 = \hat{x}_2 - \hat{x}_1$, we find $h_1 = -\frac{1}{2}h_0$. As $n \to \infty$, both $\hat{x}_n$ and $\hat{x}_{n-1}$ approximate $k/(1-(-\frac{1}{2})) = \frac{2}{3}k$. k—the initial interval between estimates—acts as a constant backdrop, not actively influencing the convergence rate. In the best-case scenario, k equals $x_1 - x_0$, where the root p is situated between those two initial estimates.

Consequently, the order of convergence α is computed as:

$$
\lim_{n\to\infty} \frac{\ln(\hat{x}_n)}{\ln(\hat{x}_{n-1})} = 1,
$$

and the factor λ becomes:

$$
\lim_{n\to\infty} \frac{\hat{x}_n}{\hat{x}_{n-1}} = 1.
$$

This indicates that the convergence is linear, with an order of convergence of 1. This is a slower convergence that the standard secant method, which has $\alpha \approx 1.618$.

Future Improvement

Given the linear order of convergence ($\alpha = 1$) indicating that this algorithm does not effectively narrow down the range in which p resides. Adopting a strategy similar to the false-position method could be beneficial. This approach involves

selecting the next approximation based on the intercept with a different sign, thus ensuring a more targeted and potentially faster convergence towards p

Further exploration into adaptive hybrid methods that combine the strengths of secant, tangent, and false-position approaches could lead to more robust and efficient root-finding algorithms.