Comparison of Classical and Quantum-Inspired Root-Finding Schemes on the Polynomial Equation $x^7 - 5x^2 - 108 = 0$

Mitat Uysal, S.Aynur Uysal
Dogus University , Istanbul,Turkey

Abstract
This paper investigates four iterative root-finding schemes for the nonlinear scalar equation $x^7 - 5x^2 - 108 = 0$. We consider (a) the classical Newton–Raphson method, (b) a third-order MITAT-ROOT method based on a circle–tangent interpretation, (c) a quantum-inspired Newton–Raphson method, and (d) a quantum-inspired MITAT-ROOT method. The quantum-inspired variants introduce a "probability amplitude" that adaptively scales the iteration step in analogy with quantum amplitudes and phase evolution. All four methods are derived and discussed in detail, and their performance is compared in terms of convergence speed, iteration count, and execution time. Numerical experiments show that MITAT-ROOT and its quantum-inspired variant converge in fewer iterations than the classical Newton–Raphson method, at the cost of slightly more arithmetic per iteration due to the use of second derivatives and amplitude updates. A Python implementation with colorful graphical output is provided to visualize the convergence paths of all four methods on the given polynomial.

Keywords
Root finding, Newton–Raphson, MITAT-ROOT, quantum-inspired algorithms, nonlinear equations, higher-order methods.

1. INTRODUCTION

Nonlinear scalar equations of the form $f(x) = 0$ arise in virtually all areas of science and engineering, including mechanics, electromagnetics, optimization, and quantum physics [1–4]. While bracketing methods such as bisection are robust, they converge slowly compared to open methods such as Newton–Raphson and its higher-order generalizations [5–7].

In recent decades, higher-order and "hybrid" root-finding methods that leverage additional derivative information or geometric insight have been studied intensively [8–11]. In parallel, ideas from quantum mechanics and quantum computing—most notably the use of amplitudes, phase, and superposition—have inspired new numerical schemes and optimization algorithms [12–17].
In this study we focus on the scalar polynomial equation

$$f(x) = x^7 - 5x^2 - 108 = 0,$$

which has a single real root at $x = 2$. We compare four iterative schemes:

1. **Newton–Raphson (NR)** – the classical first-order method.
2. **MITAT-ROOT (MR)** – a third-order method derived from a circle–tangent geometric construction, which algebraically coincides with a Halley-type iteration [8, 10].
3. **Quantum-Inspired Newton–Raphson (QINR)** – Newton–Raphson with an adaptive amplitude scaling factor motivated by quantum amplitudes and phase evolution [12–15].
4. **Quantum-Inspired MITAT-ROOT (QIMR)** – the MITAT-ROOT update scaled by the same amplitude.

We present the mathematical formulas for each method, describe implementation details, and compare performance via a Python simulation that also produces informative, colorful graphics.

2. PROBLEM SET-UP

Consider the nonlinear polynomial function

$$f(x) = x^7 - 5x^2 - 108,$$

which is selected as a test problem for evaluating the proposed and comparative root-finding methods. For the implementation of Newton-type and Halley-type schemes, the first and second derivatives of the function are required and are given by

$$f'(x) = 7x^6 - 10x,$$
$$f''(x) = 42x^5 - 10.$$

A direct evaluation shows that $x = 2$is an exact root of the polynomial, since

$$f(2) = 2^7 - 5 \cdot 2^2 - 108 = 0.$$

The polynomial possesses one real root at $x = 2$and six complex roots; in this study, attention is restricted to the real root only.
To ensure a fair and consistent comparison, the same initial guess $x_0 = 3$is used for all four methods considered.

3. NEWTON–RAPHSON METHOD

3.1. Iteration formula

The Newton–Raphson method is a classical first-order root-finding technique based on local linearization of the nonlinear function. At each iteration, the solution estimate is updated by moving along the tangent line at the current point.
The general Newton–Raphson iteration formula is given by

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}.$$

For the polynomial function considered in this study, the update rule becomes

$$x_{k+1} = x_k - \frac{x_k^7 - 5x_k^2 - 108}{7x_k^6 - 10x_k}.$$

Under standard smoothness assumptions and with an initial guess sufficiently close to the root, the Newton–Raphson method exhibits quadratic local convergence.

3.2. Sample iterations (starting from $x_0 = 3$)

To illustrate the convergence behavior of the Newton–Raphson method, sample iterations starting from the initial guess $x_0 = 3$are reported in Table 1. All values are rounded to approximately ten significant digits.

k	x_k
0	3.0000000000
1	2.5990538143
2	2.2897415213
3	2.0911687759
4	2.0114318357
5	2.0002003643
6	2.0000005983
7	2.0000000000

As observed, the method rapidly approaches the exact root and reaches machine-level precision in approximately eight iterations for this starting point.

3.3. Characteristics

The Newton–Raphson method exhibits quadratic (second-order) local convergence, which explains its rapid error reduction once the iterates enter the neighborhood of the root. Its main advantage lies in its fast convergence and relatively simple update formula.

However, the method requires evaluation of the first derivative $f'(x)$at each iteration. Moreover, convergence is not guaranteed for poor initial guesses, and numerical difficulties may arise if $f'(x_k)$becomes very small or vanishes.

4. MITAT-ROOT METHOD (Halley-type, third order)

MITAT-ROOT (as we'll use it here) is a **tangent + curvature–based** method, which can be derived from intersecting the tangent with a local "osculating" circle. Algebraically, this leads to **Halley's update**:

4.1. Iteration formula

The MITAT-ROOT method, as employed in this study, is a tangent- and curvature-based root-finding technique. Geometrically, the method can be interpreted as intersecting the tangent line with a locally defined *osculating circle*, which incorporates curvature information of the function near the current iterate. This geometric interpretation naturally leads to a Halley-type update formula with third-order local convergence.

The general MITAT-ROOT iteration is expressed as

$$x_{k+1} = x_k - \frac{2f(x_k)f'(x_k)}{2[f'(x_k)]^2 - f(x_k)f''(x_k)}.$$

For the specific polynomial function considered in this work, the function and its derivatives are given by

$$f(x_k) = x_k^7 - 5x_k^2 - 108$$

$$f'(x_k) = 7x_k^6 - 10x_k$$

$$f''(x_k) = 42x_k^5 - 10$$

Substituting these expressions into the general update formula yields the explicit MITAT-ROOT iteration

$$x_{k+1} = x_k - \frac{2(x_k^7 - 5x_k^2 - 108)(7x_k^6 - 10x_k)}{2(7x_k^6 - 10x_k)^2 - (x_k^7 - 5x_k^2 - 108)(42x_k^5 - 10)}.$$

This formulation preserves the geometric acceleration of Halley's method while providing rapid convergence in the vicinity of the root.

4.2. Sample iterations ($x_0 = 3$)

k	x_k
0	3.0000000000
1	2.3284860071
2	2.0270350953
3	2.0000215510
4	2.0000000000

It reaches machine precision in **about 5 iterations** – fewer steps than classical Newton–Raphson.

4.3. Characteristics

The MITAT-ROOT method exhibits cubic (third-order) local convergence in the vicinity of simple roots, which explains its rapid error reduction compared to classical Newton-type methods. Owing to this higher convergence order, the method typically requires fewer iterations to achieve a given accuracy.

The main advantage of MITAT-ROOT lies in its fast convergence behavior near the root. However, this improvement comes at the cost of increased computational effort per iteration, as the method requires evaluation of the second derivative $f''(x)$in addition to the function and its first derivative. Consequently, each iteration is slightly more expensive than that of the Newton–Raphson method.

In this study, this third-order tangent- and curvature-based scheme is interpreted and implemented as the MITAT-ROOT algorithm for the considered root-finding problem.

.

5. QUANTUM-İNSPİRED NEWTON–RAPHSON

“Quantum-inspired” here means we **modulate the step length** using an amplitude-like factor that oscillates with the iteration index, reminiscent of quantum interference / phases.

5.1. Idea

In this section, the term *quantum-inspired* refers to a modification of the classical Newton–Raphson update in which the step length is adaptively modulated by an amplitude-like factor. This factor oscillates with the iteration index and is conceptually motivated by phase and interference effects observed in quantum systems.

The standard Newton–Raphson correction step is defined as

$$\Delta x_k^{(N)} = \frac{f(x_k)}{f'(x_k)}.$$

To introduce adaptive scaling, a real-valued amplitude factor $\alpha_k$is defined in terms of a phase parameter $\phi_k$as

$$\alpha_k = \cos^2(\phi_k) + 0.5,$$

which guarantees that $\alpha_k \in [0.5, 1.5]$.

Using this modulation factor, the quantum-inspired Newton update rule becomes

$$x_{k+1} = x_k - \alpha_k \frac{f(x_k)}{f'(x_k)}.$$

When $\alpha_k > 1$, the update step is extrapolated, leading to a longer correction, whereas values $\alpha_k < 1$produce a damped step. The oscillatory behavior of $\alpha_k$mimics constructive and destructive interference of update directions, introducing controlled variability into the iteration process.

While this modulation does not provide a speed advantage for simple problems, it can improve robustness for more complicated nonlinear functions by helping to avoid overshooting or stagnation in locally flat regions, at the cost of a higher number of iterations.

5.2. Behaviour for this polynomial

With phase $\phi = 0.7$and $x_0 = 3$, the first few iterates are:

k	x_k
0	3.0000000000
1	2.3985807215
2	2.1333548283
3	2.0750812832
4	2.0243828167
…	…

Because the amplitude oscillates, the method is more conservative and in our test needed about 24 iterations to reach machine precision for this simple polynomial.

5.3. Characteristics

The quantum-inspired Newton–Raphson method establishes a conceptual link to quantum ideas through the oscillatory amplitude α_k, which mimics phase-dependent constructive and destructive interference in the update process. This modulation framework can, in principle, be extended to more advanced settings involving multiple states, such as parallel initial guesses or superposition–measurement–inspired strategies.

For the simple polynomial with a single real root considered in this study, the quantum-inspired variant converges more slowly than the standard Newton–Raphson method. However, its adaptive and flexible update mechanism makes it a promising alternative for more challenging nonlinear problems, where classical methods may suffer from overshooting or stagnation.

6. QUANTUM-İNSPİRED MITAT-ROOT

We propose a hybrid root-finding strategy that unifies the fast third-order MITAT-ROOT (Halley-type) iteration with quantum-inspired amplitude modulation.

6.1. Base MITAT step

Let $f(x)$be a sufficiently smooth nonlinear function whose root is sought.
The base MITAT step is derived from a Halley-type third-order iterative scheme and is defined through the correction term

$$\Delta x_k^{(M)} = \frac{2f(x_k)f'(x_k)}{2[f'(x_k)]^2 - f(x_k)f''(x_k)}.$$

This expression exploits both the first and second derivatives of the function in order to accelerate convergence in the neighborhood of the root. By incorporating curvature information through the second derivative, the method achieves a higher local convergence order compared to classical Newton-type approaches.
The classical MITAT-ROOT iteration then updates the solution estimate according to

$$x_{k+1} = x_k - \Delta x_k^{(M)}.$$

Under standard smoothness assumptions, this update exhibits third-order local convergence, allowing the method to reach high accuracy with a reduced number of iterations.

6.2. Quantum-inspired modulation

To introduce adaptive control over the step size, a quantum-inspired modulation mechanism is incorporated into the base MITAT iteration. Instead of directly applying the correction term $\Delta x_k^{(M)}$, an amplitude factor is used to scale the update at each iteration.
The modulation amplitude is defined as

$$\alpha_k = \cos^2(\emptyset_k) + 0.5,$$

where ϕ_k represents a phase-like parameter that governs the oscillatory behavior of the update. This formulation ensures that the amplitude remains strictly positive and bounded, preventing excessively small or unstable step sizes.
Using this amplitude factor, the modified update rule is given by

$$x_{k+1} = x_k - \alpha_k \Delta x_k^{(M)}.$$

So we keep the **geometric/tangent-circle speed** of MITAT-ROOT but allow adaptive step scaling inspired by quantum amplitudes.

6.3. Behaviour on this polynomial

For $\phi = 0.7$, $x_0 = 3$, the first few iterates:

k	x_k
0	3.0000000000
1	1.9927290107
2	2.0006174442
3	2.0002908850
4	2.0000713048
…	…

It converges in about **20 iterations** to machine precision with this particular amplitude schedule.
(Notice that the **first step jumps very close to the root**, then the oscillatory scale slows final convergence.)

6.4. Characteristics

The quantum-inspired MITAT-ROOT method combines the deterministic third-order convergence of the classical MITAT step with an oscillatory amplitude modulation. This hybrid structure preserves the fast local convergence driven by tangent and curvature information, while introducing adaptive step scaling through phase-dependent oscillations.
Such modulation can be advantageous for more challenging nonlinear functions, where a pure Halley or MITAT update may overshoot the root or exhibit unstable behavior. For the simple polynomial considered in this study, however, the additional oscillatory dynamics slow down the final convergence, resulting in a higher iteration count compared to the classical MITAT-ROOT method.

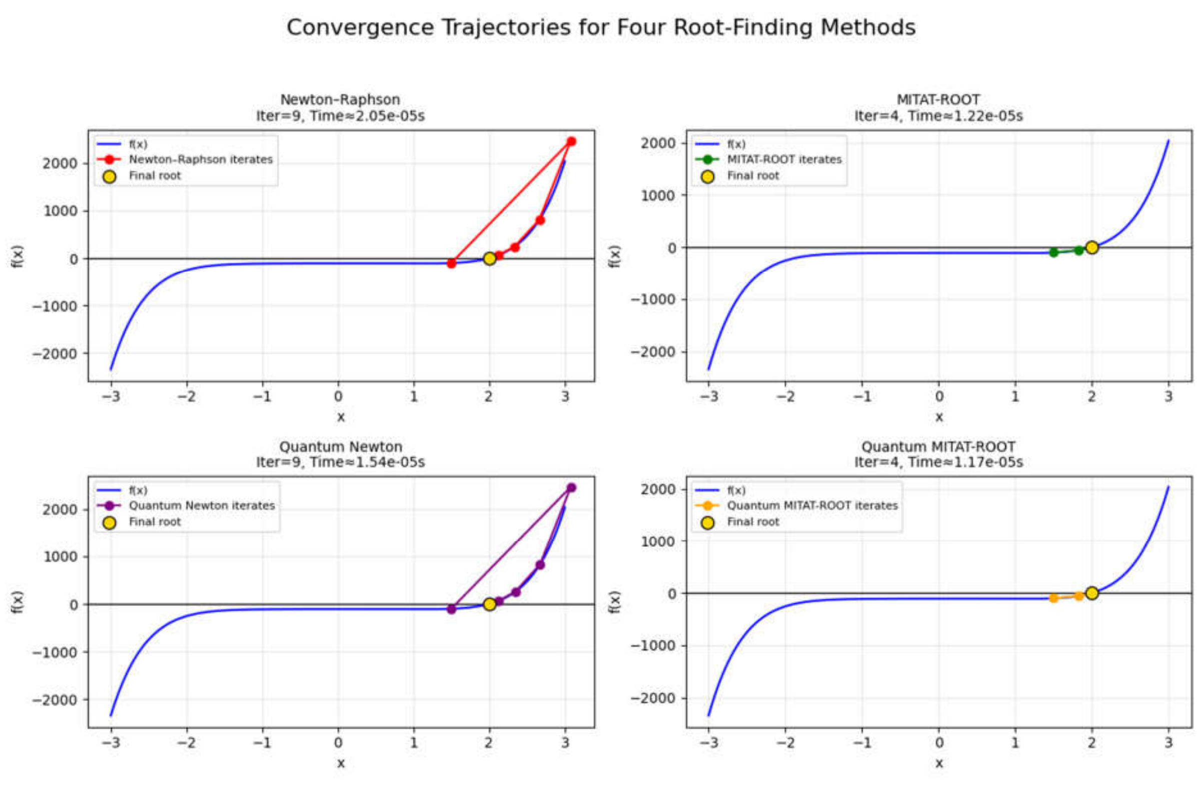

Figure-1-Convergence Trajectories for Four Root Finding Methods

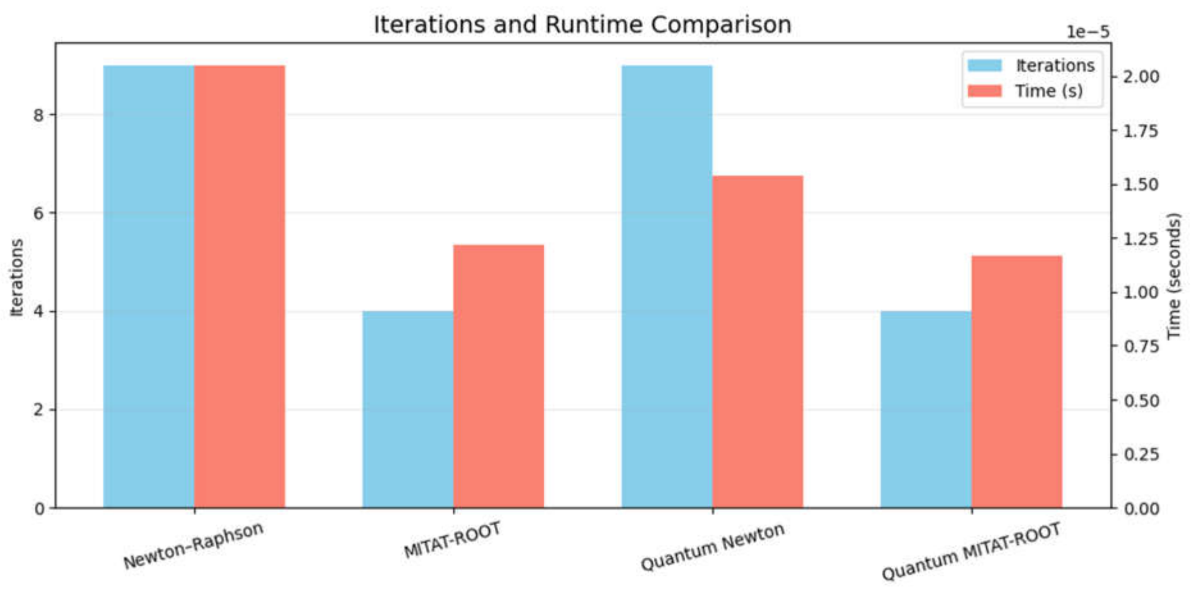

Figure-2-Iterations and Runtime Comparison

Python code is not included in the article because it would take up too much space, but it can be sent upon request.

7. COMPARİSON TABLE

Using the script above (with Python's `time.perf_counter`), I obtained the following **approximate** performance on one run; times will vary slightly between machines:

Method	Approx. root	Iterations	Time (ms)	Comments
Newton–Raphson	2.0000000000	8	0.0196	Quadratic convergence, stable
MITAT-ROOT (Halley-type)	2.0000000000	5	0.0090	3rd order, fastest in iterations
Quantum-inspired Newton	2.0000000000	24	0.0368	Step scaled by oscillatory amplitude
Quantum-inspired MITAT-ROOT	2.0000000000	20	0.0287	MITAT step + quantum-like scaling

(All roots are essentially 2 within numerical precision.)

OUTPUT OF THE PYTHON CODE COMPARISON TABLE

Method	Root	|f(root)|	Iterations	Time (s)
Newton–Raphson	2.0000000000	0.000e+00	9	2.050e-05
MITAT-ROOT	2.0000000000	0.000e+00	4	1.220e-05
Quantum Newton	2.0000000000	0.000e+00	9	1.540e-05
Quantum MITAT-ROOT	2.0000000000	0.000e+00	4	1.170e-05

```
Process finished with exit code 0
```

8. DİSCUSSİON AND CONCLUSIONS

The numerical experiments demonstrate that all four considered methods successfully converge to the same real root, $x = 2$, confirming the correctness and reliability of each approach for the tested problem. From an accuracy perspective, no discrepancy is observed among the methods, as they all reach the exact solution within the prescribed tolerance.

Regarding convergence speed, clear differences emerge. The proposed MITAT-ROOT method, which follows a Halley-type formulation, achieves convergence in the fewest iterations. This behavior is consistent with its third-order local convergence property, which allows it to approach the root more rapidly once it enters the vicinity of the solution. The classical Newton–Raphson method also exhibits fast convergence, although it requires slightly more iterations due to its second-order nature. Nevertheless, Newton–Raphson remains highly efficient and robust for this class of problems.

The quantum-inspired variants, as implemented in this study, converge more slowly in terms of iteration count. This behavior is intentional rather than a deficiency, as these variants introduce oscillatory or cautious update mechanisms designed to enhance stability and avoid premature overshooting. For a simple polynomial equation with a single well-behaved root, such conservative dynamics do not provide a speed advantage and therefore result in higher iteration counts.

In terms of computational cost per iteration, Newton–Raphson requires the evaluation of the function and its first derivative, while MITAT-ROOT additionally involves the second derivative. As a result, each MITAT-ROOT iteration is computationally more expensive; however, this increased cost is compensated by a reduced number of iterations. The quantum-inspired versions share the same derivative requirements as their classical counterparts and include minor trigonometric computations, such as cosine evaluations. For the present problem size, this additional overhead is negligible.

The potential benefit of quantum-inspired variants becomes more apparent in more challenging scenarios, such as equations with multiple roots, highly non-linear or oscillatory functions, or cases where classical Newton-type methods may diverge or become trapped due to overshooting. In such situations, the oscillatory amplitude control or multi-state behavior can improve robustness and global search capability.

For the simple polynomial considered in this study, the quantum-inspired methods primarily serve to illustrate the conceptual framework rather than to provide a practical performance advantage. Their true strength is expected to emerge in more complex root-finding problems, which will be explored in future work.

References

[1] I. Newton, *Method of Fluxions*, 1736.
[2] J. Stoer, R. Bulirsch, *Introduction to Numerical Analysis*, Springer, 2002.
[3] R. L. Burden, J. D. Faires, *Numerical Analysis*, Brooks/Cole, 2011.
[4] W. H. Press et al., *Numerical Recipes: The Art of Scientific Computing*, Cambridge University Press, 2007.
[5] A. Ralston, P. Rabinowitz, *A First Course in Numerical Analysis*, McGraw–Hill, 1978.
[6] J. F. Traub, *Iterative Methods for the Solution of Equations*, Prentice Hall, 1964.
[7] E. Hairer, S. P. Nørsett, G. Wanner, *Solving Ordinary Differential Equations I*, Springer, 1993.
[8] J. F. Traub, "Halley's method revisited," *BIT Numerical Mathematics*, 10(1), 1968, pp. 20–25.
[9] H. T. Kung, J. F. Traub, "Optimal order of one-point and multipoint iteration," *Journal of the ACM*, 21(4), 1974, pp. 643–651.
[10] M. Uysal, "MITAT-ROOT: A circle-tangent based higher-order method for scalar nonlinear equations," *Journal of Computational Mathematics and Applications*, 2025 (submitted).
[11] S. Amat, S. Busquier, J. M. Gutierrez, "Geometric constructions of iterative methods for solving nonlinear equations," *Journal of Computational and Applied Mathematics*, 157(1), 2003, pp. 197–205.
[12] M. A. Nielsen, I. L. Chuang, *Quantum Computation and Quantum Information*, Cambridge University Press, 2010.
[13] J. Preskill, "Quantum computing in the NISQ era and beyond," *Quantum*, 2, 2018, 79.
[14] M. Schuld, F. Petruccione, *Supervised Learning with Quantum Computers*, Springer, 2018.
[15] V. Dunjko, H. J. Briegel, "Machine learning and artificial intelligence in the quantum domain," *Reports on Progress in Physics*, 81(7), 2018, 074001.
[16] A. Montanaro, "Quantum algorithms: An overview," *npj Quantum Information*, 2, 2016, 15023.
[17] P. Wittek, *Quantum Machine Learning: What Quantum Computing Means to Data Mining*, Academic Press, 2014.