

Lecture 3: Root-Finding Methods

Nonlinear Equations and Numerical Solutions

Francisco Richter Mendoza

Università della Svizzera Italiana

Faculty of Informatics, Lugano, Switzerland

September 10, 2025

Overview

Root-finding represents one of the fundamental problems in numerical analysis: given a function $f(x)$, find values x^* such that $f(x^*) = 0$. This seemingly simple problem underlies countless applications in science and engineering, from solving nonlinear equations to finding equilibrium points in dynamical systems.

The mathematical challenge lies in the fact that most nonlinear equations cannot be solved analytically. Numerical methods provide systematic approaches to approximate roots with controlled accuracy. We explore both bracketing methods that guarantee convergence and open methods that offer rapid convergence when properly initialized.

Our analysis emphasizes the theoretical foundations that govern convergence behavior, stability properties, and practical implementation considerations. The interplay between mathematical theory and computational practice reveals why certain methods excel in specific contexts while failing in others.

1 Mathematical Foundations

1.1 The Root-Finding Problem

Definition. Root-Finding Problem

Given a continuous function $f : [a, b] \rightarrow \mathbb{R}$, find $x^* \in [a, b]$ such that $f(x^*) = 0$.

The value x^* is called a **root** or **zero** of f .

Theorem 1.1 (Intermediate Value Theorem). *If f is continuous on $[a, b]$ and $f(a) \cdot f(b) < 0$, then there exists at least one $c \in (a, b)$ such that $f(c) = 0$.*

This theorem provides the theoretical foundation for bracketing methods.

Definition. Types of Roots

Let f be differentiable near x^* where $f(x^*) = 0$.

Simple Root: $f'(x^*) \neq 0$

Multiple Root: $f'(x^*) = 0$, with multiplicity m if $f^{(k)}(x^*) = 0$ for $k = 0, 1, \dots, m-1$ and $f^{(m)}(x^*) \neq 0$.

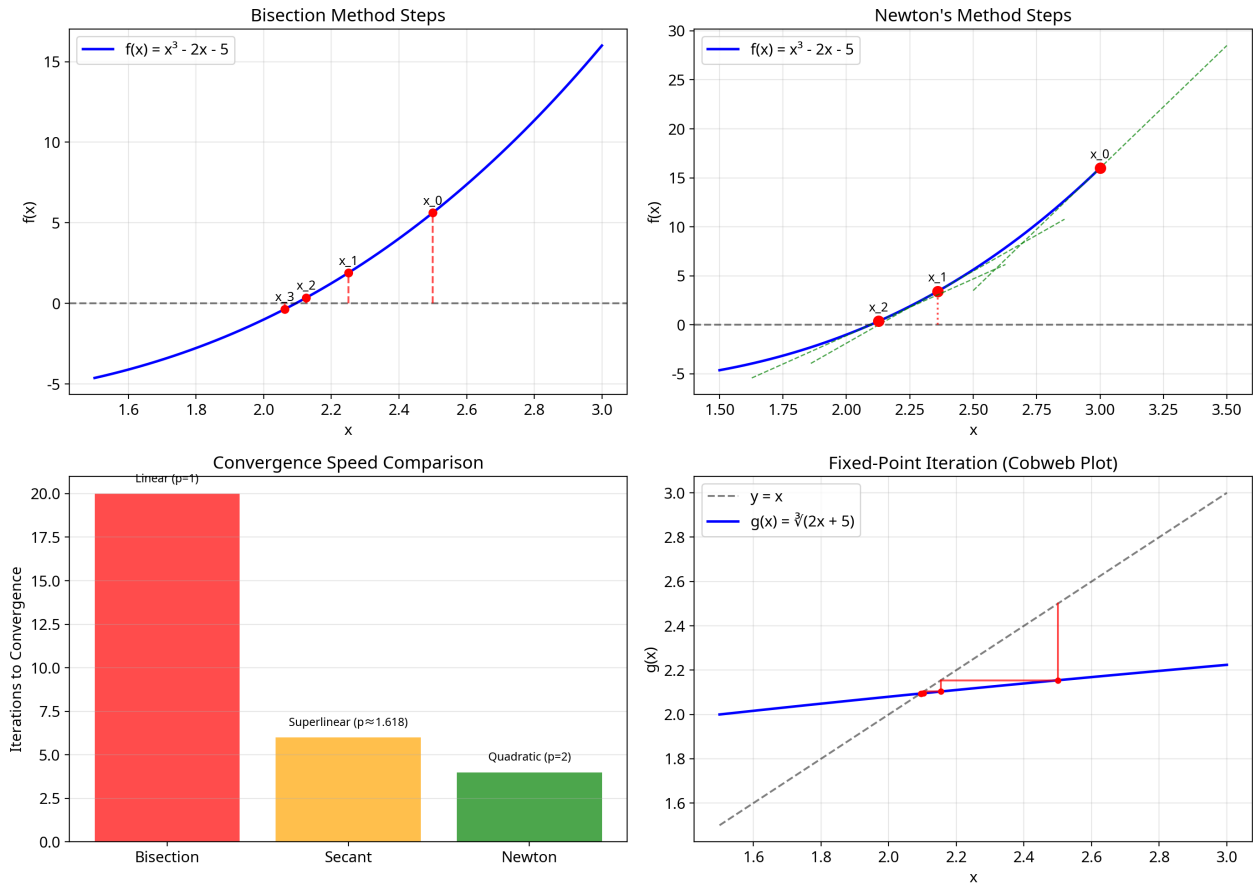


Figure 1: Comprehensive comparison of root-finding approaches. Top left: Bisection method systematically narrows the interval containing the root through repeated halving. Top right: Newton's method uses tangent lines for rapid convergence, showing the geometric interpretation of the iterative process. Bottom left: Convergence rate comparison demonstrates the superior performance of higher-order methods in terms of iterations required. Bottom right: Fixed-point iteration visualized through the cobweb plot, illustrating the geometric convergence behavior.

1.2 Convergence Analysis Framework

Definition. Order of Convergence

A sequence $\{x_k\}$ converges to x^* with order $p \geq 1$ if:

$$\lim_{k \rightarrow \infty} \frac{|x_{k+1} - x^*|}{|x_k - x^*|^p} = C$$

where $0 < C < \infty$.

- $p = 1$: Linear convergence (if $C < 1$)
- $p = 2$: Quadratic convergence
- $1 < p < 2$: Superlinear convergence

Theorem 1.2 (Convergence Rate Implications). For achieving error tolerance ϵ :

- *Linear convergence:* $O(\log(1/\epsilon))$ iterations
- *Quadratic convergence:* $O(\log \log(1/\epsilon))$ iterations

Quadratic convergence provides dramatic improvement once the iterates are sufficiently close to the root.

Intuition: Understanding Convergence Orders

The difference between linear and quadratic convergence is profound:

- **Linear:** Each iteration gains approximately one decimal digit of accuracy
- **Quadratic:** Each iteration approximately doubles the number of correct digits
- **Superlinear:** Performance between linear and quadratic, often very effective in practice

This explains why Newton's method is so powerful when it converges - the number of correct digits grows exponentially.

2 Bracketing Methods

2.1 Bisection Method

Definition. Bisection Method

Given f continuous on $[a, b]$ with $f(a) \cdot f(b) < 0$:

Algorithm:

1. Set $c = (a + b)/2$
2. If $f(a) \cdot f(c) < 0$, set $b = c$; otherwise set $a = c$
3. Repeat until $|b - a| < \text{tolerance}$

Theorem 2.1 (Bisection Convergence). *The bisection method converges linearly with:*

$$|x_n - x^*| \leq \frac{b_0 - a_0}{2^n}$$

where $[a_0, b_0]$ is the initial interval.

The method requires exactly $n = \lceil \log_2((b_0 - a_0)/\epsilon) \rceil$ iterations to achieve accuracy ϵ .

Example 2.1 (Bisection Method Analysis). *To find the root of $f(x) = x^3 - 2x - 5$ on $[2, 3]$ with accuracy 10^{-6} :*

Number of iterations needed: $n = \lceil \log_2(1/10^{-6}) \rceil = 20$

Each iteration reduces the interval by exactly half, providing guaranteed convergence but relatively slow progress.

```

1 def bisection_method(f, a, b, tol=1e-10, max_iter=100):
2     if f(a) * f(b) >= 0:
3         raise ValueError("f(a) and f(b) must have opposite signs")
4 
```

```

5   for i in range(max_iter):
6       c = (a + b) / 2
7       if abs(f(c)) < tol or (b - a) / 2 < tol:
8           return c
9
10      if f(a) * f(c) < 0:
11          b = c
12      else:
13          a = c
14
15  return c

```

Listing 1: Bisection Method Implementation

2.2 Regula Falsi (False Position)

Definition. Regula Falsi Method

Instead of bisecting the interval, use linear interpolation:

$$c = a - f(a) \frac{b - a}{f(b) - f(a)} = \frac{af(b) - bf(a)}{f(b) - f(a)}$$

This often provides faster convergence than bisection by using function values to guide the choice of the next point.

Theorem 2.2 (Regula Falsi Convergence). *The regula falsi method converges linearly, but typically faster than bisection. However, it can exhibit slow convergence when one endpoint remains fixed for many iterations.*

Modified versions (Illinois algorithm, Pegasus method) address this limitation by modifying function values to prevent stagnation.

Intuition: Bracketing Methods Trade-offs

Bracketing methods offer guaranteed convergence at the cost of slower convergence rates:

- **Robustness:** Always converge if initial bracket is valid
- **Simplicity:** Easy to implement and understand
- **Reliability:** Cannot diverge or fail to converge
- **Speed:** Linear convergence is slower than open methods
- **Generalization:** Difficult to extend to systems of equations

These methods are ideal when robustness is more important than speed.

3 Fixed-Point Iteration

3.1 Mathematical Framework

Definition. Fixed-Point Problem

Given $g : \mathbb{R} \rightarrow \mathbb{R}$, find x^* such that $g(x^*) = x^*$.

The root-finding problem $f(x) = 0$ can be reformulated as a fixed-point problem by setting $g(x) = x + f(x)$ or $g(x) = x - \alpha f(x)$ for some $\alpha \neq 0$.

Definition. Fixed-Point Iteration

Starting with initial guess x_0 , generate the sequence:

$$x_{k+1} = g(x_k), \quad k = 0, 1, 2, \dots$$

Theorem 3.1 (Fixed-Point Theorem). *Let g be continuously differentiable on $[a, b]$ with $g([a, b]) \subseteq [a, b]$. If $|g'(x)| \leq L < 1$ for all $x \in [a, b]$, then:*

1. g has a unique fixed point $x^* \in [a, b]$
2. For any $x_0 \in [a, b]$, the iteration $x_{k+1} = g(x_k)$ converges to x^*
3. The convergence is linear with rate L :

$$|x_{k+1} - x^*| \leq L|x_k - x^*|$$

Theorem 3.2 (Order of Convergence for Fixed-Point Iteration). *If g is sufficiently smooth near the fixed point x^* :*

- If $g'(x^*) \neq 0$: Linear convergence
- If $g'(x^*) = 0$ and $g''(x^*) \neq 0$: Quadratic convergence
- If $g'(x^*) = g''(x^*) = \dots = g^{(p-1)}(x^*) = 0$ and $g^{(p)}(x^*) \neq 0$: Order p convergence

Example 3.1 (Fixed-Point Reformulations). *For $f(x) = x^2 - 2 = 0$ (finding $\sqrt{2}$):*

Method 1: $g_1(x) = x^2 - x + 2$, $g'_1(x) = 2x - 1$, so $g'_1(\sqrt{2}) = 2\sqrt{2} - 1 > 1$ (diverges)

Method 2: $g_2(x) = 2/x$, $g'_2(x) = -2/x^2$, so $g'_2(\sqrt{2}) = -1$ (slow convergence)

Method 3: $g_3(x) = \frac{1}{2}(x + 2/x)$ (Newton's method), $g'_3(x) = \frac{1}{2}(1 - 2/x^2)$, so $g'_3(\sqrt{2}) = 0$ (quadratic convergence)

This example illustrates the critical importance of choosing the right reformulation.

4 Newton's Method

4.1 Derivation and Analysis

Definition. Newton's Method

Starting with initial guess x_0 , generate the sequence:

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}, \quad k = 0, 1, 2, \dots$$

This is equivalent to fixed-point iteration with $g(x) = x - f(x)/f'(x)$.

Theorem 4.1 (Newton's Method Convergence). *Let f be twice continuously differentiable in a neighborhood of a simple root x^* . If x_0 is sufficiently close to x^* , then Newton's method converges quadratically:*

$$\lim_{k \rightarrow \infty} \frac{|x_{k+1} - x^*|}{|x_k - x^*|^2} = \frac{|f''(x^*)|}{2|f'(x^*)|}$$

Proof sketch. Using Taylor expansion around x^* :

$$f(x_k) = f'(x^*)(x_k - x^*) + \frac{f''(\xi_k)}{2}(x_k - x^*)^2$$

Since $f(x^*) = 0$:

$$x_{k+1} - x^* = x_k - x^* - \frac{f(x_k)}{f'(x_k)} = \frac{f''(\xi_k)}{2f'(x_k)}(x_k - x^*)^2$$

For x_k near x^* , this gives quadratic convergence. $\square$

Theorem 4.2 (Newton's Method for Multiple Roots). *If x^* is a root of multiplicity $m > 1$, then Newton's method converges linearly with convergence factor $(m - 1)/m$.*

Modified Newton's Method: $x_{k+1} = x_k - m \frac{f(x_k)}{f'(x_k)}$ restores quadratic convergence if m is known.

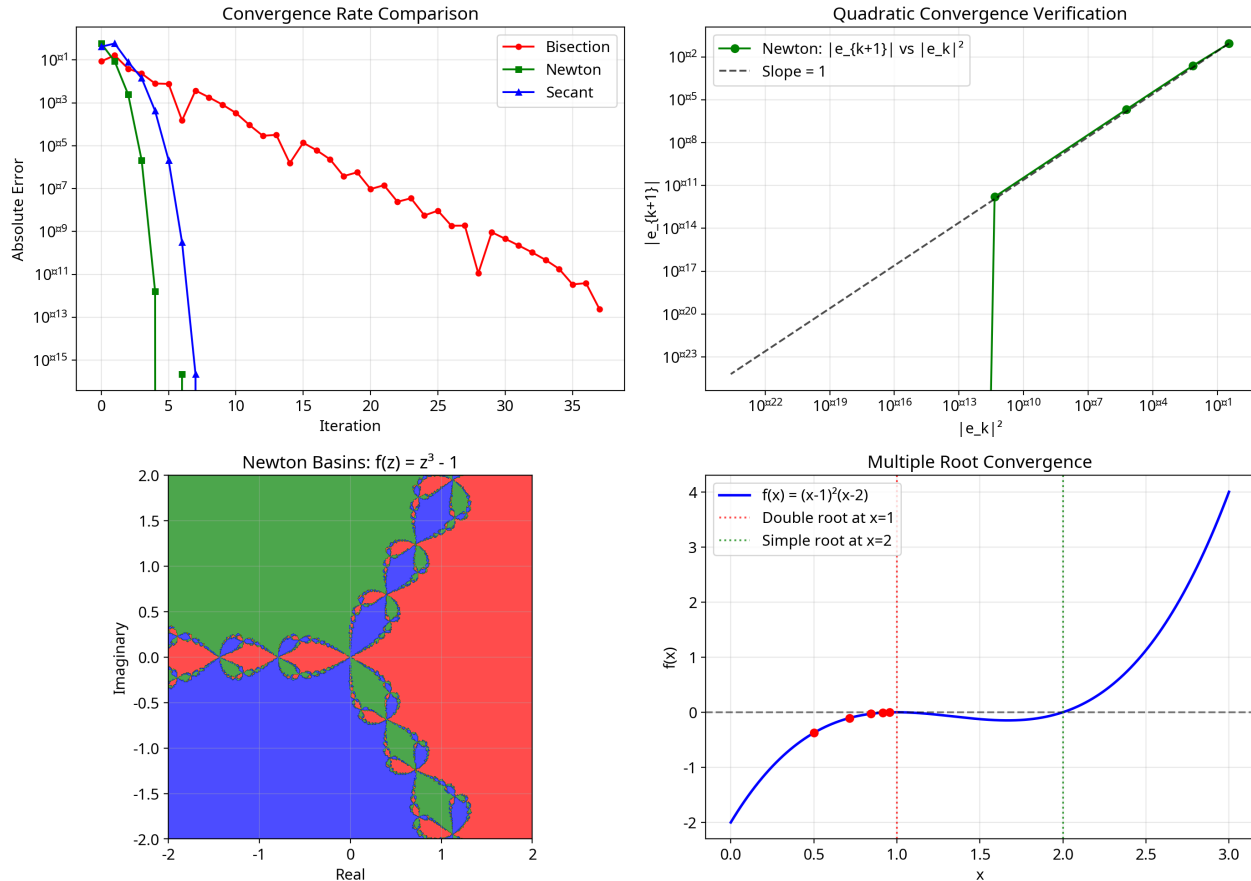


Figure 2: Detailed convergence behavior analysis. Top left: Error reduction rates comparing bisection, Newton, and secant methods, demonstrating the dramatic superiority of higher-order methods. Top right: Quadratic convergence verification for Newton's method, plotting $|e_{k+1}|$ vs $|e_k|^2$ to confirm theoretical predictions. Bottom left: Complex basin analysis showing fractal boundaries in Newton's method convergence regions, illustrating sensitive dependence on initial conditions. Bottom right: Multiple root convergence challenges, showing how root multiplicity affects convergence behavior.

4.2 Practical Considerations

Example 4.1 (Newton's Method Implementation).

```

1 def newton_method(f, df, x0, tol=1e-10, max_iter=100):
2     x = x0
3     for i in range(max_iter):
4         fx = f(x)
5         dfx = df(x)
6
7         if abs(dfx) < 1e-15:
8             raise ValueError("Derivative too small, method may fail")
9
10        x_new = x - fx / dfx
11
12        if abs(x_new - x) < tol:
13            return x_new
14
15        x = x_new
16
17    raise ValueError("Maximum iterations reached without convergence")

```

Listing 2: Newton's Method with Error Handling

Theorem 4.3 (Basin of Attraction). *For a simple root x^* , there exists a neighborhood $N(x^*)$ such that Newton's method converges to x^* for any starting point in $N(x^*)$.*

The size of this basin depends on the function's behavior and can be quite small for some functions.

Intuition: Newton's Method Trade-offs

Newton's method exemplifies the speed-robustness trade-off in numerical methods:

- **Speed:** Quadratic convergence when it works
- **Efficiency:** Requires only one function evaluation per iteration
- **Sensitivity:** Requires good initial guess
- **Derivatives:** Needs $f'(x)$, which may be problematic
- **Failure modes:** Can diverge, oscillate, or find wrong root

The method is powerful but requires careful implementation and initialization.

5 Secant and Related Methods

5.1 Secant Method

Definition. Secant Method

Approximate the derivative in Newton's method using finite differences:

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

This requires two initial guesses x_0 and x_1 .

Theorem 5.1 (Secant Method Convergence). *For a simple root, the secant method converges superlinearly with order $p = \frac{1+\sqrt{5}}{2} \approx 1.618$ (the golden ratio):*

$$\lim_{k \rightarrow \infty} \frac{|x_{k+1} - x^*|}{|x_k - x^*|^p} = C$$

This is faster than linear but slower than quadratic convergence.

Example 5.1 (Method Comparison). *To achieve 6 decimal places of accuracy:*

- Bisection: ≈ 20 iterations
- Secant: ≈ 8 iterations
- Newton: ≈ 5 iterations

The secant method provides a good compromise between robustness and speed.

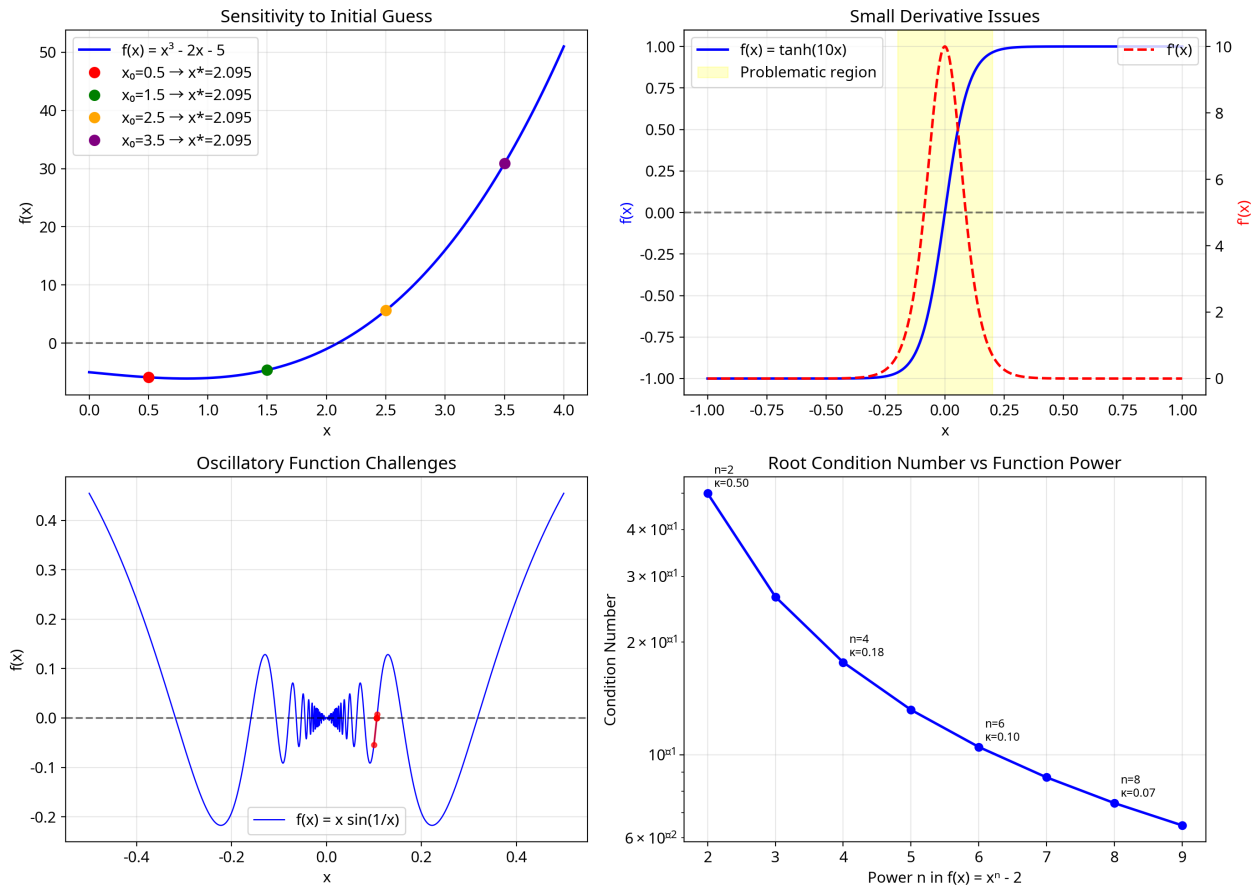


Figure 3: Real-world implementation challenges. Top left: Sensitivity to initial conditions showing how different starting points can lead to convergence to different roots or divergence. Top right: Small derivative issues demonstrating instability when $f'(x) \approx 0$, causing large Newton steps. Bottom left: Oscillatory function difficulties where traditional methods struggle with highly oscillatory behavior. Bottom right: Condition number effects showing how problem sensitivity affects numerical accuracy and method reliability.

6 Advanced Methods and Practical Implementation

6.1 Hybrid Methods

Definition. Brent's Method

Combines the robustness of bisection with the speed of faster methods:

1. Start with bisection to ensure convergence
2. Switch to inverse quadratic interpolation when conditions are favorable
3. Fall back to bisection if faster methods show signs of trouble

Theorem 6.1 (Hybrid Method Properties). *Hybrid methods like Brent's method achieve:*

- *Guaranteed convergence (from bisection component)*
- *Superlinear convergence rate (from faster methods)*
- *Robust performance across diverse function types*

6.2 Deflation Techniques

For polynomials with multiple roots, deflation allows systematic root finding:

Definition. Polynomial Deflation

If r is a known root of polynomial $p(x)$, then:

$$p(x) = (x - r)q(x)$$

where $q(x)$ is a polynomial of degree one less than $p(x)$.

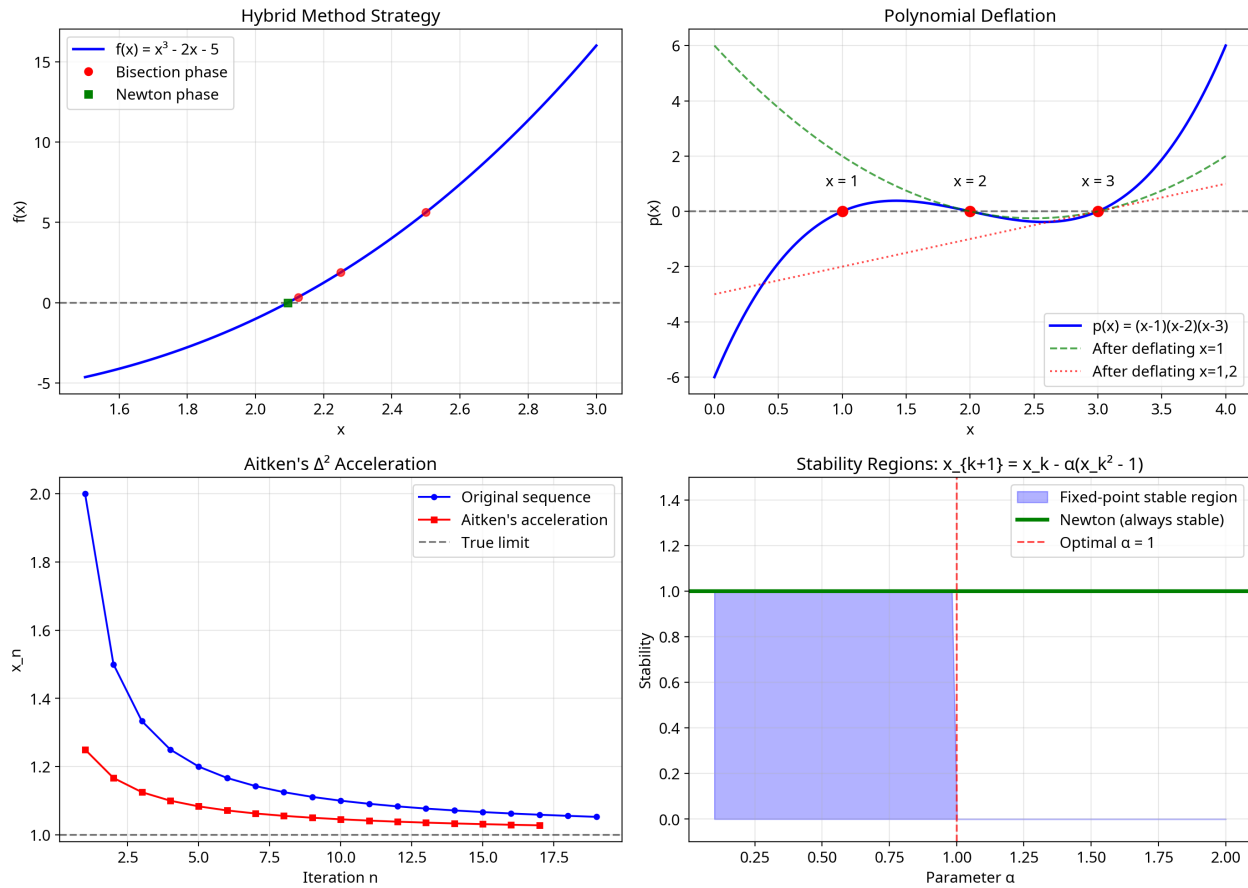


Figure 4: Sophisticated root-finding techniques. Top left: Hybrid method strategy combining bisection's reliability with Newton's speed, showing the transition between methods. Top right: Polynomial deflation for systematically finding multiple roots by reducing polynomial degree. Bottom left: Aitken acceleration technique for improving convergence of slowly converging sequences. Bottom right: Stability region analysis showing parameter ranges for which different methods remain stable and convergent.

6.3 Error Analysis and Stopping Criteria

Definition. Practical Stopping Criteria

Combine multiple conditions for robust termination:

$$|x_{k+1} - x_k| < \epsilon_{\text{abs}} + \epsilon_{\text{rel}}|x_k| \quad \text{AND} \quad |f(x_k)| < \epsilon_f$$

This ensures both that the iterates have converged and that we're close to a true root.

Theorem 6.2 (Condition Number for Root-Finding). *The condition number for root-finding is:*

$$\kappa = \frac{|x^*|}{|f'(x^*)|}$$

Large condition numbers indicate that small changes in the function can cause large changes in the root location.

7 Applications and Examples

7.1 Engineering Applications

Example 7.1 (Colebrook Equation). *In fluid mechanics, the friction factor f for turbulent pipe flow satisfies:*

$$\frac{1}{\sqrt{f}} = -2 \log_{10} \left(\frac{\epsilon/D}{3.7} + \frac{2.51}{Re\sqrt{f}} \right)$$

This implicit equation requires iterative solution, typically using Newton's method with careful initialization.

Example 7.2 (Kepler's Equation). *In orbital mechanics, the eccentric anomaly E satisfies:*

$$M = E - e \sin E$$

where M is the mean anomaly and e is the eccentricity. This transcendental equation is fundamental to satellite orbit calculations.

7.2 Computational Considerations

Theorem 7.1 (Computational Complexity). *For tolerance ϵ :*

- **Bisection:** $O(\log(1/\epsilon))$ function evaluations
- **Newton:** $O(\log \log(1/\epsilon))$ function + derivative evaluations
- **Secant:** $O(\log(1/\epsilon)^{0.618})$ function evaluations

The choice depends on the relative cost of function vs derivative evaluations.

Practice Problems

1. Find all roots of $f(x) = x^3 - 6x^2 + 11x - 6$ using different methods. Compare convergence rates.
2. For $f(x) = e^x - 2x - 1$, analyze the convergence of Newton's method starting from different initial guesses.
3. Implement the secant method and compare its performance to Newton's method for $f(x) = \cos(x) - x$.
4. Design a hybrid method that combines bisection and Newton's method. Test it on challenging functions.
5. Analyze the condition number for finding $\sqrt{a}$ using $f(x) = x^2 - a$. How does it depend on a ?