



PyNum
Interactive Studio

Numerical Analysis Series

Fixed-Point Iteration

(Also known as Functional Iteration)

Author
Shahad Uddin
shahaduddin.com

Date
February 2, 2026
Version 1.0

1. Theoretical Background

Definition

A number p is a **fixed point** for a given function g if $g(p) = p$.

The connection to root-finding is simple: Given a problem $f(x) = 0$, we can rewrite it in the form $x = g(x)$. If we find a fixed point p for g , then p is also a zero of $f(x)$.

The Iterative Formula: To approximate the fixed point, we choose an initial approximation p_0 and generate the sequence $\{p_n\}_{n=0}^{\infty}$ by letting:

$$p_n = g(p_{n-1}) \quad \text{for } n \geq 1 \quad (1)$$

Convergence Criteria

Not all choices of $g(x)$ will result in a converging sequence. According to the **Fixed-Point Theorem**:

- **Existence:** If $g \in C[a, b]$ and $g(x) \in [a, b]$ for all $x \in [a, b]$, then g has at least one fixed point in $[a, b]$.
- **Uniqueness & Convergence:** If, in addition, $g'(x)$ exists on (a, b) and there exists a constant k with $0 < k < 1$ such that:

$$|g'(x)| \leq k < 1 \quad \text{for all } x \in (a, b)$$

Then the fixed point is unique and the iteration converges for any p_0 in $[a, b]$.

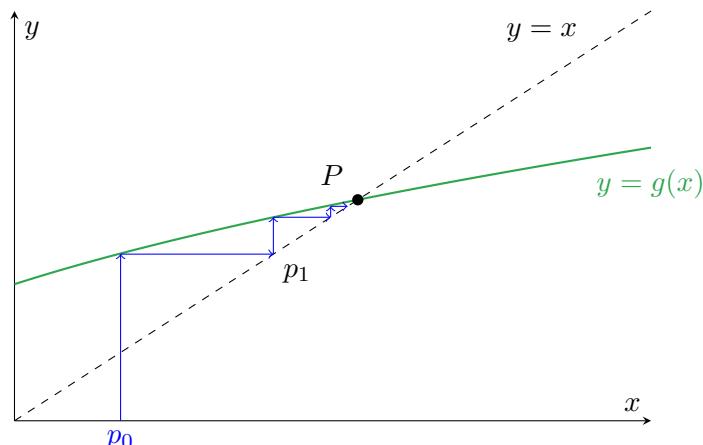


Fig 1. "Staircase" convergence where $0 < g'(x) < 1$.

2. Algorithm

Fixed-Point Iteration Algorithm

To find a solution to $p = g(p)$ given an initial approximation p_0 :

1. **Input:** Initial approximation p_0 ; tolerance TOL ; max iterations N_0 .
2. **Step 1:** Set $i = 1$.
3. **Step 2:** While $i \leq N_0$ do Steps 3–6:
 - **Step 3:** Set $p = g(p_0)$ (Compute p_i).
 - **Step 4:** If $|p - p_0| < TOL$ then **OUTPUT** p ; **STOP**.
 - **Step 5:** Set $i = i + 1$.
 - **Step 6:** Set $p_0 = p$ (Update).
4. **Step 7:** Output "Method failed after N_0 iterations". **STOP**.

3. Code Implementations

Example: Finding the Golden Ratio by solving $x^2 - x - 1 = 0$. Two rearrangements are used: $g(x) = 1 + 1/x$ (Python) and $g(x) = \sqrt{x+1}$ (Fortran/C++).

>PythonImplementation

(.py)

```

1 def fixed_point_iteration(g, x0, tol=1e-6, max_iter=100):
2     """
3         Solves x = g(x) given an initial guess x0.
4         Find roots by rearranging f(x)=0 into x = g(x).
5     """
6     x = x0
7     for i in range(max_iter):
8         x_new = g(x)
9
10        if abs(x_new - x) < tol:
11            return x_new
12
13        x = x_new
14
15        print("Warning: Max iterations reached")
16    return x
17
18 # Example: Root of x^2 - x - 1 = 0 -> x = 1 + 1/x
19 g = lambda x: 1 + 1/x
20 root = fixed_point_iteration(g, 1.5)
21 print(f"Root: {root}")

```

>FortranImplementation

(.f90)

```

1 program fixed_point
2     implicit none
3     real :: x, x_next, tol
4     integer :: i, max_iter
5
6     ! Parameters
7     x = 1.0
8     tol = 1e-6; max_iter = 100
9
10    print *, "Solving x = g(x) via Fixed Point Iteration..."
11
12    do i = 1, max_iter
13        x_next = g(x)
14
15        ! Convergence check
16        if (abs(x_next - x) < tol) exit
17
18        x = x_next
19
20        if (i == max_iter) print *, "Warning: Max iterations reached."
21    end do
22
23    print *, "Root found: ", x_next
24
25 contains
26     real function g(x)
27         real, intent(in) :: x
28         ! Example: Solving x^2 - x - 1 = 0 => x = sqrt(x + 1)
29         ! This converges to the Golden Ratio (approx 1.618034)
30         g = sqrt(x + 1.0)
31     end function g
32 end program

```

>C ++Implementation

(.cpp)

```

1 #include <iostream>
2 #include <cmath>
3 #include <iomanip>
4 #include <functional>
5
6 /**
7 * Fixed Point Iteration method for solving x = g(x).
8 * Converges if |g'(x)| < 1 in the neighborhood of the root.
9 */
10 double fixed_point_iteration(std::function<double(double)> g, double x0,
11     double tol = 1e-6, int max_iter = 100) {
12     double x = x0;
13     for (int i = 0; i < max_iter; ++i) {
14         double x_next = g(x);
15
16         // Stop if the difference is within tolerance
17         if (std::abs(x_next - x) < tol) {
18             return x_next;
19         }

```

```
20         x = x_next;
21     }
22     std::cout << "Warning: Max iterations reached." << std::endl;
23     return x;
24 }
25
26 int main() {
27     // Example: Solving x^2 - x - 1 = 0 => x = sqrt(x + 1)
28     // The fixed point of this g(x) is the Golden Ratio.
29     auto g = [] (double x) { return std::sqrt(x + 1.0); };
30
31     double root = fixed_point_iteration(g, 1.0);
32
33     std::cout << std::fixed << std::setprecision(6);
34     std::cout << "Fixed Point Root: " << root << std::endl;
35
36     return 0;
37 }
```

For more numerical analysis resources, visit shahaduddin.com/PyNum