Fixed-Point Iteration

Tamas Kis | kis@stanford.edu | https://github.com/tamaskis

Contents

1 FIXED-POINT ITERATION

Consider a univariate function $f(x)$. A **fixed point** of $f(x)$, which we denote as c, satisfies

$$f(c) = c$$

Essentially, at the fixed point, the value of the dependent variable is the same as the value of the independent variable. Another way to view fixed points are as the intersections of the curve $y = f(x)$ with the curve $y = x$.

To solve for the fixed point c, we use a technique called **fixed-point iteration.** There are two basic algorithms for implementing fixed-point iteration. The first implementation, shown in Algorithm 1 below, does *not* store the result of each iteration. On the other hand, the second implementation, shown in Algorithm 2, *does* store the result of each iteration. `fixed_point_iteration` implements both of these algorithms [1].

Since Algorithm 2 first needs to preallocate a potentially huge array to store all of the intermediate solutions, Algorithm 1 is significantly faster. Even if $i_{\max}$ (determines size of the preallocated array) is set to be a small number (for example, 10), Algorithm 1 is still faster. The reason we still consider and implement Algorithm 2 is so that convergence studies may be performed.

Algorithm 1:
Fixed-point iteration (fast implementation).

Given:

- $f(x)$ - function
- x_0 - initial guess for fixed point
- TOL - tolerance
- $i_{\max}$ - maximum number of iterations

Procedure:

1. Initialize the error so that the loop will be entered.

$$\varepsilon = (2)(\text{TOL})$$

2. Manually set the fixed point estimate at the first iteration based on the initial guess.

$$x_{\text{old}} = x_0$$

3. Initialize x_{new} so its scope will not be limited to within the while loop.

$$x_{\text{new}} = 0$$

4. Initialize the loop index.

$$i = 1$$

5. Find the fixed point using fixed-point iteration.

$$\textbf{while } (\varepsilon > \text{TOL}) \textbf{ and } (i < i_{\max})$$

(a) Update fixed point estimate.

$$x_{\text{new}} = f(x_{\text{old}})$$

(b) Calculate error.

$$\varepsilon = |x_{\text{new}} - x_{\text{int}}|$$

(c) Store the current fixed point estimate for the next iteration.

$$x_{\text{old}} = x_{\text{new}}$$

(d) Increment loop index.

$$i = i + 1$$

end

Return:

- $c = x_{\text{new}}$ - converged fixed point

Algorithm 2:

Fixed-point iteration (storing intermediate fixed point estimates).

Given:

- $f(x)$ - function
- x_0 - initial guess for fixed point
- TOL - tolerance
- $i_{\max}$ - maximum number of iterations

Procedure:

1. Initialize the error so that the loop will be entered.

$$\varepsilon = (2)(\text{TOL})$$

2. Preallocate $\mathbf{x} \in \mathbb{R}^{i_{\max}}$ to store the estimates of the fixed point at each iteration.
3. Manually set the fixed point estimate at the first iteration based on the initial guess (note that x_1 is the first element of $\mathbf{x}$, while x_0 is the input initial guess).

$$x_1 = x_0$$

4. Initialize the loop index.

$$i = 1$$

5. Find the fixed point using fixed-point iteration.

while $(\varepsilon > \text{TOL})$ **and** $(i < i_{\max})$

(a) Update fixed point estimate.

$$x_{i+1} = f(x_i)$$

(b) Calculate error.

$$\varepsilon = |x_{i+1} - x_i|$$

(c) Increment loop index.

$$i = i + 1$$

end

Return:

- $\mathbf{c} = \mathbf{x}$ - vector where the first element is the initial guess for the fixed point, the subsequent elements are the intermediate fixed point estimates, and the final element is the converged fixed point

1.1 Iterative Approaches in Engineering

Fixed-point iteration is especially useful for solving many problems in engineering. Consider the case where we have two unknown quantities, and two highly nonlinear equations that relate them to one another. Mathematically, we have two variables, x and y, and the following two functions:

$$y = f(x) \tag{1}$$

$$x = g(y) \tag{2}$$

In one function, you input x and get y, while in the other function, you input y and get x. Since $f(x)$ and $g(y)$ are nonlinear (as previously mentioned), we cannot obtain closed-form solutions for x and y.

Let's say we're primarily interested in the variable x, where the variable y mainly serves to place a constraint on x. We want to find the value $x = c$ such that both equations are satisfied simultaneously (i.e. $y = f(c)$ and $c = g(y)$). Then we can define a new function $h(x)$ as the function composition $h = g \circ f$ by substituting Eq. (1) into Eq. (2).

$$h(x) = g(f(x))$$

The solution to our problem, $x = c$, is just the fixed point of $h(x)$. See Examples #4a and #4b on the "Examples" tab of `https://www.mathworks.com/matlabcentral/fileexchange/86992-fixed-point-iteration-fixed_point_iteration` for an implementation of this approach to a pipe flow problem[1].

1.2 Relationship to Newton's Method

Newton's method for finding the root of a univariate function is fundamentally related to fixed-point iteration. The equation defining the iterative procedure in Newton's method is

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$

1 This example is adapted from my personal solutions to Problem 8.96 in [2, p. 476]. However, this fluid mechanics text, in general, does not take a computational approach to such problems. Rather, it performs a "trial and error" procedure (including hand calculations and reading values off of a chart), which essentially follows the same process as fixed-point iteration – an example of this can be found in Example 8.7 [2, p. 444].

Let's define a function $g(x)$.

$$g(x) = x - \frac{f(x)}{f'(x)}$$

Then Newton's method becomes

$$x_{i+1} = g(x_i)$$

Thus, Newton's method is just a fixed-point iteration of the function $g(x) = x - f(x)/f'(x)$ [1]. See Example #3 on the "Examples" tab of `https://www.mathworks.com/matlabcentral/fileexchange/86992-fixed-point-iteration-fixed_point_iteration`.

REFERENCES

[1] James Hateley. *Nonlinear Equations*. MATH 3620 Course Reader (Vanderbilt University). 2019.

[2] Bruce R. Munson et al. *Fundamentals of Fluid Mechanics*. 7th. Hoboken, NJ: John Wiley & Sons, 2013.