Lec 22: Rootfinding Problem – One Dimension

Fixed Point Iteration

Fixed Point

Definition 1 (Fixed Point)

The real number r is a **fixed point** of the function g if $g(r) = r$.

- The rootfinding problem $f(x) = 0$ can always be written as a fixed point problem $g(x) = x$ by, *e.g.*, setting[1]

$$g(x) = x - f(x).$$

- The fixed point problem is true at, and only at, a root of f.

[1] This is not the only way to transform the rootfinding problem. More on this later.

Fixed Point Iteration

A fixed point problem $g(x) = x$ naturally provides an iteration scheme:

$$\begin{cases} x_0 = \text{initial guess} \\ x_{k+1} = g(x_k), \quad k = 0, 1, 2, \ldots. \end{cases} \qquad \text{(fixed point iteration)}$$

- The sequence $\{x_k\}$ may or may not converge as $k \to \infty$.
- If g is continuous and $\{x_k\}$ converges to a number r, then r is a fixed point of g.

$$g(r) = g\left(\lim_{k\to\infty} x_k\right) = \lim_{k\to\infty} g(x_k) = \lim_{k\to\infty} x_{k+1} = r.$$

Fixed Point Iteration Algorithm

```
function x = fpi(g, x0, n)
% FPI x = fpi(g, x0, n)
% Computes approximate solution of g(x)=x
% Input:
%    g     function handle
%    x0    initial guess
%    n     number of iteration steps
    x = x0;
    for k = 1:n
        x = g(x);
    end
end
```

Examples

- To find a fixed point of $g(x) = 0.3\cos(2x)$ near 0.5 using `fpi`:

```
g = @(x) 0.3*cos(2*x);
xc = fpi(g,0.5,20)
```

```
xc = 0.260266319627758
```

Not All Fixed Point Problems Are The Same

The rootfinding problem $f(x) = x^3 + x - 1 = 0$ can be transformed to various fixed point problems:

- $g_1(x) = x - f(x) = 1 - x^3$
- $g_2(x) = \sqrt[3]{1 - x}$
- $g_3(x) = \dfrac{1 + 2x^3}{1 + 3x^2}$

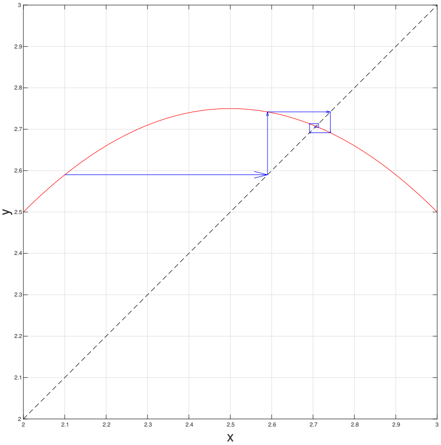
Note that all $g_j(x) = x$ are equivalent to $f(x) = 0$. However, not all these find a fixed point of g, that is, a root of f on the computer.

Exercise. Run `fpi` with g_j and $x_0 = 0.5$. Which fixed point iterations converge?

Geometry of Fixed Point Iteration

The following script[2] finds a root of $f(x) = x^2 - 4x + 3.5$ via FPI.

```
f = @(x) x.^2 - 4*x + 3.5;
g = @(x) x - f(x);
fplot(g, [2 3], 'r');
hold on
plot([2 3], [2 3], 'k--')
x = 2.1;
y = g(x);
for k = 1:5
    arrow([x y], [y y], 'b');
    x = y; y = g(x);
    arrow([x x], [x y], 'b');
end
```

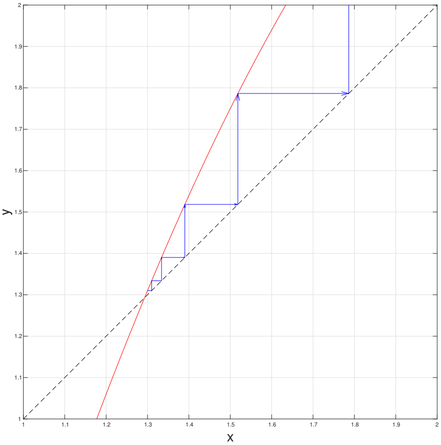


Note the line segments spiral in towards the fixed point.

[2] Modified from FNC.

However, with a different starting point, the process does not converge.

```
clf
fplot(g, [1 2], 'r');
hold on
plot([1 2], [1 2], 'k--'),
ylim([1 2])
x = 1.3; y = g(x);
for k = 1:5
    arrow([x y], [y y], 'b');
    x = y; y = g(x);
    arrow([x x], [x y], 'b');
end
```



Custom function: `arrow = @(p1, p2, varargin) quiver(p1(1), p1(2), p2(1)-p1(1), p2(2)-p1(2), 0, varargin{:})`

Series Analysis

Let $\epsilon_k = x_k - r$ be the sequence of errors.

- The iteration formula $x_{k+1} = g(x_k)$ can be written as

$$\begin{aligned}\epsilon_{k+1} + r &= g(\epsilon_k + r)\\ &= g(r) + g'(r)\epsilon_k + \frac{1}{2}g''(r)\epsilon_k^2 + \cdots, \qquad \text{(Taylor series)}\end{aligned}$$

  implying

$$\epsilon_{k+1} = g'(r)\epsilon_k + O(\epsilon_k^2)$$

  assuming sufficient regularity of g.

- Neglecting the second-order term, we have $\epsilon_{k+1} \approx g'(r)\epsilon_k$, which is satisfied if $\epsilon_k \approx C\left[g'(r)\right]^k$ for sufficiently large k.
- Therefore, the iteration converges if $\left|g'(r)\right| < 1$ and diverges if $\left|g'(r)\right| > 1$.

Note: Rate of Convergence

Definition 2 (Linear Convergnece)

Suppose $\lim_{k\to\infty} x_k = r$ and let $\epsilon_k = x_k - r$, the error at step k of an iteration method. If

$$\lim_{k\to\infty} \frac{|\epsilon_{k+1}|}{|\epsilon_k|} = \sigma < 1,$$

the method is said to obey **linear convergence** with rate σ.

Note. In general, say

$$\lim_{k\to\infty} \frac{|\epsilon_{k+1}|}{|\epsilon_k|^p} = \sigma$$

for some $p \geqslant 1$ and $\sigma > 0$.

- If $p = 1$ and
  - $\sigma = 1$, the convergence is *sublinear*;
  - $0 < \sigma < 1$, the convergence is *linear*;
  - $\sigma = 0$, the convergence is *superlinear*.
- If $p = 2$, the convergence is *quadratic*;
- If $p = 3$, the convergence is *cubic*, …

Convergence of Fixed Point Iteration

Theorem 3 (Convergence of FPI)

Assume that g is continuously differentiable, that $g(r) = r$, and that $\sigma = |g'(r)| < 1$. Then the fixed point iterates x_k generated by

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

converge linearly with rate σ to the fixed point r for x_0 sufficiently close to r.

In the previous example with $g(x) = x - f(x) = -x^2 + 5x - 3.5$:

- For the first fixed point, near 2.71, we get $g'(r) \approx -0.42$ (convergence);
- For the second fixed point, near 1.29, we get $g'(r) \approx 2.42$ (divergence).

Note. An iterative method is called **locally convergent** to r if the method converges to r for initial guess sufficiently close to r.

Contraction Maps

Lipschitz Condition

A function g is said to satisfy a **Lipschitz condition** with constant L on the interval $S \subset \mathbb{R}$ if

$$\left|g(s) - g(t)\right| \leqslant L\,|s - t| \quad \text{for all } s, t \in S.$$

- A function satisfying the Lipschitz condition is continuous on S.
- If $L < 1$, g is called a **contraction map**.

When Does FPI Succeed?

Contraction Mapping Theorem

Suppose that g satisfies Lipschitz condition on S with $L < 1$, *i.e.*, g is a contraction map on S. Then S contains exactly one fixed point r of g. If $x_1, x_2, \ldots$ are generated by the fixed point iteration $x_{k+1} = g(x_k)$, and $x_1, x_2, \ldots$ all lie in S, then

$$|x_k - r| \leqslant L^{k-1} |x_1 - r|, \quad k > 1.$$

Newton's Method

Newton's Method

To find the root of f:

Newton's Method (Algorithm)

- Begin at the point $(x_0, f(x_0))$ on the curve and draw the tangent line at the point using the slope $f'(x_0)$:

$$y = f(x_0) + f'(x_0)(x - x_0).$$

- Find the x-intercept of the line and call it x_1:

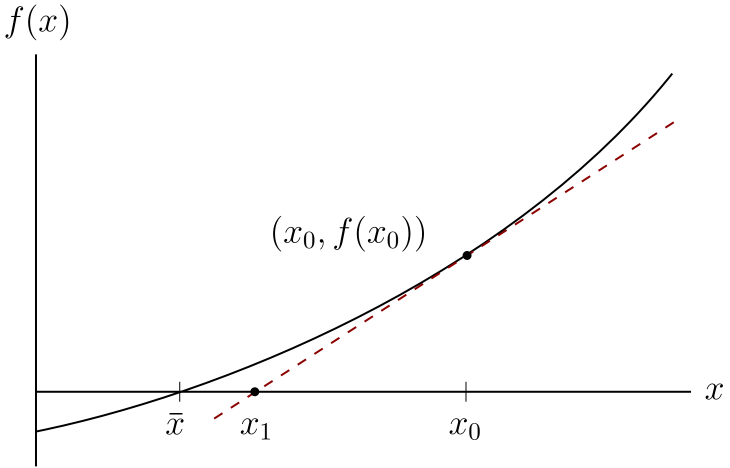
$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}.$$

- Continue this procedure to find $x_2, x_3, \ldots$ until the sequence converges to the root.

General iterative formula:

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)} \quad \text{for } k = 0, 1, 2, \ldots \tag{$\star$}$$

Newton’s Method: Illustration



Series Analysis

Let $\epsilon_k = x_k - r$, $k = 1, 2, \ldots$, where r is the limit of the sequence and $f(r) = 0$.

Substituting $x_k = r + \epsilon_k$ into the iterative formula ($\star$):

$$\epsilon_{k+1} = \epsilon_k - \frac{f(r+\epsilon_k)}{f'(r+\epsilon_k)}.$$

Taylor-expand f about $x = r$ and simplify (assuming $f'(r) \neq 0$):

$$\begin{aligned}
\epsilon_{k+1} &= \epsilon_k - \frac{f(r) + \epsilon_k f'(r) + \frac{1}{2}\epsilon_k^2 f''(r) + O(\epsilon_k^3)}{f'(r) + \epsilon_k f''(r) + O(\epsilon_k^2)} \\
&= \epsilon_k - \epsilon_k \left[1 + \frac{1}{2}\frac{f''(r)}{f'(r)}\epsilon_k + O(\epsilon_k^2)\right]\left[1 + \frac{f''(r)}{f'(r)}\epsilon_k + O(\epsilon_k^2)\right]^{-1} \\
&= \frac{1}{2}\frac{f''(r)}{f'(r)}\epsilon_k^2 + O(\epsilon_k^3).
\end{aligned}$$

- Previous calculation shows that $\epsilon_{k+1} \approx C\epsilon_k^2$, eventually. Written differently,

$$\left|\epsilon_{k+1}\right| / \left|\epsilon_k\right|^2 \to \text{(some positive number)}, \text{ as } k \to \infty.$$

that is, each Newton iteration roughly squares the previous error. This is **quadratic convergence**[3].

- Alternately, note that

$$\log\left|\epsilon_{k+1}\right| \approx 2\log\left|\epsilon_k\right| + \text{(constant)},$$

ignoring high-order terms. This means that the number of accurate digits[4] approximately doubles at each iteration.

---

[3]Recall the formal definition given in p. 11.

[4]We say that an iterate is **correct within** p **decimal places** if the error is less than 0.5×10^{-p}.

Convergence of Newton's Method

Theorem 4 (Quadratic Convergence of Newton's Method)

Let f be twice continuously differentiable and $f(r) = 0$. If $f'(r) \neq 0$, then Newton's method is locally and quadratically convergent to r. The error $\epsilon_k = x_k - r$ at step k satisfies

$$\lim_{k \to \infty} \frac{|\epsilon_{k+1}|}{|\epsilon_k|^2} = \left| \frac{f''(r)}{2f'(r)} \right|.$$

Implementation

```
function x = newton(f,dfdx,x1)
% NEWTON   Newton's method for a scalar equation.
% Input:
%   f         objective function
%   dfdx      derivative function
%   x1        initial root approximation
% Output
%   x         vector of root approximations (last one is best)

% Operating parameters.
    funtol = 100*eps;  xtol = 100*eps;  maxiter = 40;

    x = x1;
    y = f(x1);
    dx = Inf;    % for initial pass below
    k = 1;

    while (abs(dx) > xtol) && (abs(y) > funtol) && (k < maxiter)
        dydx = dfdx(x(k));
        dx = -y/dydx;           % Newton step
        x(k+1) = x(k) + dx;

        k = k+1;
        y = f(x(k));
    end

    if k==maxiter, warning('Maximum number of iterations reached.'), end
end
```

Note: Stopping Criteria

For a set tolerance, $\texttt{TOL}$, some example stopping criteria are:

- Absolute error:
$$\left|x_{k+1} - x_k\right| < \texttt{TOL}.$$
- Relative error: (useful when the solution is not too close to zero)
$$\frac{\left|x_{k+1} - x_k\right|}{\left|x_{k+1}\right|} < \texttt{TOL}.$$
- Hybrid:
$$\frac{\left|x_{k+1} - x_k\right|}{\max(\left|x_{k+1}\right|, \theta)} < \texttt{TOL},$$
for some $\theta > 0$.
- Residual:
$$\left|f(x_k)\right| < \texttt{TOL}.$$

Also useful to set a limit on the maximum number of iterations in case convergence fails.

Secant Method

Secant Method

- Newton's method requires calculation and evaluation of $f'(x)$, which may be challenging at times.
- The most common alternative to such situations is the **secant method**.
- The secant method replaces the instantaneous slope in Newton's method by the average slope using the last two iterates.

Secant Method (cont')

Secant Method (Algorithm)

- Begin with two initial iterates x_{-1} and x_0; draw the secant line connecting $(x_{-1}, f(x_{-1}))$ and $(x_0, f(x_0))$:

$$y = f(x_0) + \frac{f(x_0) - f(x_{-1})}{x_0 - x_{-1}}(x - x_0).$$

- Find the x-intercept of the line and call it x_1:

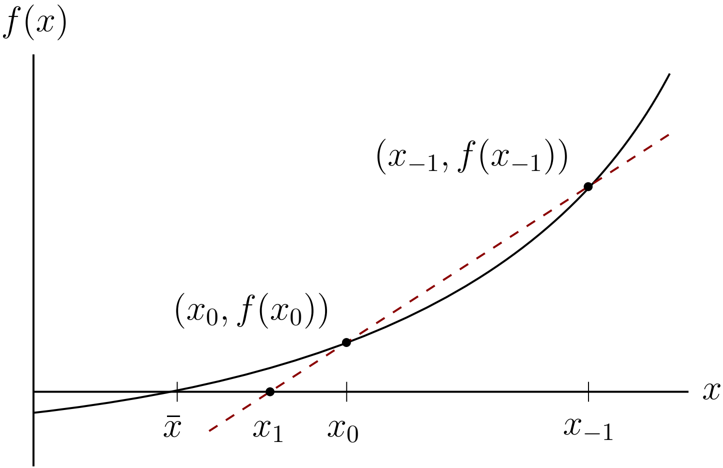
$$x_1 = x_0 - f(x_0)\frac{x_0 - x_{-1}}{f(x_0) - f(x_{-1})}.$$

- Continue this procedure to find $x_2, x_3, \ldots$ until convergence is obtained.

General iterative formula:

$$x_{k+1} = x_k - f(x_k)\frac{x_k - x_{k-1}}{f(x_k) - f(x_{k-1})} \quad \text{for } k = 0, 1, 2, \ldots$$

Secant Method: Illustration



Series Analysis

Assume that the secant method converges to r and $f'(r) \neq 0$. Let $\epsilon_k = x_k = r$ as before.

It can be shown that

$$\left|\epsilon_{k+1}\right| \approx \left|\frac{f''(r)}{2f'(r)}\right| \left|\epsilon_k\right| \left|\epsilon_{k-1}\right|,$$

which implies that

$$\left|\epsilon_{k+1}\right| \approx \left|\frac{f''(r)}{2f'(r)}\right|^{\alpha-1} \left|\epsilon_k\right|^{\alpha},$$

where

$$\alpha = \frac{1+\sqrt{5}}{2} \approx 1.618,$$

the *golden ratio*.

Therefore, the convergence of the secant method is **superlinear**; it lies between linearly and quadratically convergent methods.

Exercise. Confirm the statements in the previous page. Namely, show that

1. The error ϵ_k satisfies the approximate equation

$$|\epsilon_{k+1}| \approx \left|\frac{f''(r)}{2f'(r)}\right| |\epsilon_k| \, |\epsilon_{k-1}| \, .$$

2. If in addition $\lim_{k\to\infty} |\epsilon_{k+1}| / |\epsilon_k|^\alpha$ exists and is nonzero for some $\alpha > 0$, then

$$|\epsilon_{k+1}| \approx \left|\frac{f''(r)}{2f'(r)}\right|^{\alpha-1} |\epsilon_k|^\alpha \, , \quad \text{where } \alpha = \frac{1+\sqrt{5}}{2}.$$

Implementation

```
function x = secant(f,x1,x2)
% SECANT   Secant method for a scalar equation.
% Input:
%   f         objective function
%   x1,x2     initial root approximations
% Output
%   x         vector of root approximations (last is best)

% Operating parameters.
    funtol = 100*eps;  xtol = 100*eps;  maxiter = 40;

    x = [x1 x2];
    dx = Inf;  y1 = f(x1);
    k = 2;  y2 = 100;

    while (abs(dx) > xtol) && (abs(y2) > funtol) && (k < maxiter)
        y2 = f(x(k));
        dx = -y2 * (x(k)-x(k-1)) / (y2-y1);   % secant step
        x(k+1) = x(k) + dx;

        k = k+1;
        y1 = y2;    % current f-value becomes the old one next time
    end

    if k==maxiter, warning('Maximum number of iterations reached.'), end
end
```

Other Methods

Inverse Interpolation

The **inverse quadratic interpolation** (IQI) is a generalization of the secant method to parabolas.

- Instead of using two most recent points (to determine a straight line), use three and obtain an quadratic interpolant.
- The parabola of the form $y = p(x)$ may have zero, one, or two x-intercept(s). So use the form $x = p(y)$, a parabola open sideways.

Algorithm.

- Begin with three initial iterates x_{-2}, x_{-1}, x_0; find the parabola of the form $x = p(y)$ passing through the three points $(x_{-2}, f(x_{-2}))$, $(x_{-1}, f(x_{-1}))$, and $(x_0, f(x_0))$.
- Find the x-intercept of the parabola and call it x_1.
- Continue the procedure to find $x_2, x_3, \ldots$ until convergence is obtained.

Inverse Interpolation (cont')

General iterative formula:

$$x_{k+1} = x_k - \frac{r(r-q)(x_k - x_{k-1}) + (1-r)s(x_k - x_{k-2})}{(q-1)(r-1)(s-1)}, \quad \text{for } k = 0, 1, 2, \ldots,$$

where

$$q = \frac{f(x_{k-2})}{f(x_{k-1})}, \quad r = \frac{f(x_k)}{f(x_{k-1})}, \quad s = \frac{f(x_k)}{f(x_{k-2})}.$$

Rather than deriving and implementing the formula, try using `polyfit` to perform the interpolation step.

Bisection Method: Bracketing a Root

The following is a corollary to the intermediate value theorem.

Theorem 5 (Existence of a Root)

Let f be a continuous function on $[a, b]$, satisfying $f(a)f(b) < 0$. Then f has a root between a and b, that is, there exists a number $r \in (a, b)$ such that $f(r) = 0$.

Bisection Method (cont')

Algorithm.

- Start with an interval $[a, b]$ where $f(a)f(b) \leqslant 0$.
- Bisect the interval into $[a, m] \cup [m, b]$ where $m = (a + b)/2$ is the midpoint.
- Select the subinterval in which $f(x)$ changes signs, *i.e.*, calculate $f(a)f(m)$ and $f(m)f(b)$, choose the nonpositive one, and update the values of a and b.
- Repeat the process until you get close enough to the solution.

Notes

Let $[a, b]$ be the initial interval and let $[a_k, b_k]$ be the interval after k bisection steps.

- The length of $[a_k, b_k]$ is $(b-a)/2^k$.
- Using the midpoint $x_k = (a_k + b_k)/2$ as an estimate of the root r, note that

$$|\epsilon_k| = |x_k - r| < \frac{b-a}{2^{k+1}}.$$

- This accuracy is obtained by $k+2$ function evaluations.

Bisection Method: Pseudocode

```
while <a NOT CLOSE ENOUGH TO b>
  m = (a + b)/2;
  fm = f(m);
  if sign(fa) ~= sign(fm)
    b = m;
    fb = fm;
  else
    a = m;
    fa = fm;
  end
end
x_zero = .5*(a + b);
```