Math 471. Numerical methods
Root-finding algorithms for nonlinear equations

overlap Section 2.1 – 2.5 of Bradie

Our goal in this chapter is to find the root(s) for $f(x) = 0$.

§2.1 Bisection Method

Intermediate value theorem. $f(x)$ is continuous on $[a, b]$. If $f(a)f(b) < 0$, then there must exist at least one root of $f(x) = 0$ in the interval (a, b).

• Example. Let $f(x) = x^3 - 5x + 1$. Then, $f(-3) = -11, f(-2) = 3 \to x_1 \in (-3, -2)$. Likewise $x_2 \in (-1, 1)$ and $x_3 \in (2, 3)$.

Bisection method.

Q: find root $p \in (a, b)$ for $f(x) = 0$.

A: generate a sequence of root enclosing intervals $(a, b) = (a_0, b_0) \supset (a_1, b_1) \supset (a_2, b_2)$....

The process is iterative, that is, (a_{n+1}, b_{n+1}) is determined by (a_n, b_n)....(a_0, b_0) (in simple cases, (a_n, b_n) is sufficient). Each (a_{n+1}, b_{n+1}) is either the left half or right half of (a_n, b_n), depending on the signs of $f(a_n), f\left(\frac{a_n+b_n}{2}\right), f(b_n)$. More precisely, if $f(a_n)f\left(\frac{a_n+b_n}{2}\right) < 0$, then $(a_{n+1}, b_{n+1}) = (a_n, \frac{a_n+b_n}{2})$ — (left half). Otherwise, $(a_{n+1}, b_{n+1}) = (\frac{a_n+b_n}{2}, b_n)$ — (right half).

Note. In case $f(a_n) = 0$ or $f(b_n) = 0$, the algorithm should stop, yielding an exact root.

If this process goes on "infinitely", then by *squeezing theorem*, $\lim_{n\to\infty} a_n = \lim_{n\to\infty} b_n = \lim_{n\to\infty}(a_n + b_n)/2 = p$. But, in a computer algorithm, it must stop at certain point n (so that we simply take a_n or b_n or $(a_n+b_n)/2$ as the appproximate root) → stopping condition

1. $|b_n - a_n| < \varepsilon$.
2. $|f(a_n)| < \varepsilon$ or $|f(b_n)| < \varepsilon$.
3. $n = n_{max}$.
4. ...

Here, ε and n_{max} are prescribed as input parameters from outside the algorithm.

Error bound. Let p be the exact root that the algorithm converges to. Since it is always true that $p \in (a_n, b_n)$, we know

$$|p - (a_n + b_n)/2| < |b_n - a_n|/2 = 2^{-n-1}|b_0 - a_0|.$$

So the Bisection method converges at rate $O(2^{-n})$. Also, by the above estimate, if we

use stopping condition 1, then $\varepsilon/2$ is indeed an error bound for $p-(a_n+b_n)/2$, giving us some confidence on the accuracy of the approximation.

• Example. . $f(x)=x^2-3, f(1)=-2, f(2)=1 \rightarrow$ there is a root $p \in (1,2)$. (theorectically, $p=\sqrt{3}=1.732...$).

n	a_n	b_n	$\frac{a_n+b_n}{2}$	$f(\frac{a_n+b_n}{2})$
0	1	2	1.5	-0.75
1	1.5	2	1.75	0.0625
2	1.5	1.75	1.625	-0.3594
3	1.625	1.75	1.6875	-0.1523
4	1.6875	1.75	...	...

A sample code with stoppping condition 1. The input parameters are initial guess (a,b) and ε.

```
function c=bisection(a,b,ep)


a0=a; b0=b;
while ((b-a)>=ep)
   c=(a+b)/2;
   if (f(c)==0) % root is at the middle point
      return;
   end
   if (f(a)*f(c)<0) % root is in left half
      a0=a; b0=c;
   else % root is in right half
     a0=c; b0=b;
   end
   a=a0; b=b0;
end


c=(a+b)/2;
```

Note that an "f.m" file containing the definition of $f(x)$ should be stored in the same directory as "bisection.m".

A shorter but less readable code can be the following.

```
function c=bisection_short(a,b,ep)
while ((b-a)>=ep)
   c=(a+b)/2;
   if (f(a)*f(c)<0)
      b=c;
   elseif (f(c)*f(b)<0)
      a=c;
   else
      return;
   end
end

c=(a+b)/2;
```

$2.2 False Position method

The Bisection method belongs the family of enclosure methods that use a sequence of enclosing intervals $(a_0, b_0) \supset (a_1, b_1) \supset (a_2, b_2)...$ to gradually squeeze onto the exact root. The underlying principle of an enclosure method is the Intermediate Value Theorem.

In each step, the choice of the dividing point c_n doesn't have to be the middle point of (a_n, b_n). What can be an alternative? For the False Position method, it "pretends" that $f(x)$ is a linear function and uses the x-intercept of the line that connects $(a_n, f(a_n))$ and $(b_n, f(b_n))$ to be the candidate of the dividing point c_n. Then, check the sign condition and decide the next interval (a_{n+1}, b_{n+1}) to be either (a_n, c_n) or (c_n, b_n).

Note. This method would actually reach the exact root in one step if $f(x)$ is indeed a linear function on (a_n, b_n). Otherwise it only gives an approximation and hence the name "false position".

To find c_n, we establish the equation for the line that connects $(a_n, f(a_n))$ and $(b_n, f(b_n))$. By the point-slope formula, it should be

$$y - f(a_n) = \frac{f(b_n) - f(a_n)}{b_n - a_n}(x - a_n).$$

To find the x-intercept, set $y = 0$ in the above equation and solve for x to get the value of the dividing point c_n

$$c_n = a_n - \frac{f(a_n)(b_n - a_n)}{f(b_n) - f(a_n)}$$

or, in a more symmetric way,

$$c_n = \frac{a_n f(b_n) - b_n f(a_n)}{b_n - a_n}.$$

Convergence and error bound. The text book gives a long but elementary discussion on the error bound. Here, we only give the result: on the n-the step,

$$|p - c_n| \approx \frac{(c_n - c_{n-1})^2}{c_n - 2c_{n-1} + c_{n-2}},$$

which can serve as part of the stopping condition.

§2.3 Fixed-point iteration. This is a family of schemes that solve the fixed-point problem

$$g(x) = x \leftrightarrow g(x) - x = 0.$$

So, there is equivalence between fixed-point problem and root-finding problem.

- Example. $f(x) = x^2 - 3 = 0$ can be re-written as fixed-point problems

$$g_1(x) = \frac{3}{x} = x \text{ or } g_2(x) = x - x^2 + 3 = x \text{ or } g_3(x) = x - \frac{x^2 - 3}{2} = x.$$

The (iterative) scheme is simply

$$x_{n+1} = g(x_n).$$

(Initialize: $x_0 = 1.5$)

n	g_1 $x_n = g_1(x_{n-1})$	g_2 $x_n = g_3(x_{n-1})$	g_3 $x_n = g_3(x_{n-1})$
0	1.5	1.5	1.5
1	2	2.25	1.875
2	1.5	0.1875	1.6172
3	2	3.1523	1.8095
4	1.5	-3.7849	1.6723
5	2	-15.1106	1.7740

If the limit $\lim_{n\to\infty} x_n = y$ shall exist, we must have

$$y = \lim_{n\to\infty} x_n = \lim_{n\to\infty} g(x_{n-1}) = g(\lim_{n\to\infty} x_{n-1}) = g(y).$$

However, not every choice of g will lead to convergence $\lim_{n\to\infty} x_n = p$. For example, from the above table, we see that the numerical schemes are not working for g_1 and g_2.

Theorem. If $|g'(x)| \le k$ for $x \in (a,b)$ and if (by some argument) $x_n \in (a,b)$, $n = 0,1,2,...$ and $p \in [a,b]$, then

$$|p - x_n| \le k^n |p - x_0|.$$

In particular, if $k < 1$, then convergence.

Proof. Mean value theorem. Or, as we did in class, Taylor series

$$g(x) - g(p) = (x - p)g'(\xi)$$

for some $\xi \in (x,p)$ or $\xi \in (p,x)$. A useful trick in studying error estimates for fixed point iteration

$$\begin{aligned} p &= g(p) && \text{def. of fixed point} \\ x_{n+1} &= g(x_n) && \text{numerical scheme} \\ p - x_{n+1} &= g(p) - g(x_n) && \text{subtract the above two} \end{aligned}$$

Corollary. If $\max\limits_{x\in[a,b]} |g'(x)| < 1$ and **if** g **maps** $[a,b]$ **onto** $[a,b]$ and we start with $x_0 \in [a,b]$, then obviously all $x_n \in [a,b]$ and thus convergence.

• Example. Consider $g(x) = x - \frac{x^2-3}{2}$ with $x_0 = 1.5$. Then, $x_1 = 1.875$ and try if

$$[a,b] = [x_0, x_1] = [1.5, 1.875]$$

will work. Indeed, by Calculus,

$$\left.\begin{aligned} \max_{x\in[1.5,1.875]} g(x) &= g(1.5) = 1.875 \\ \min_{x\in[1.5,1.875]} g(x) &= g(1.875) = 1.617 \end{aligned}\right\} \Rightarrow \ g \text{ maps } [1.5, 1.875] \text{ onto } [1.617, 1.875],$$

assumption of self-mapping satisfied!

Also satisfied

$$k = \max_{x\in[a,b]} |g'(x)| = 0.875 < 1.$$

Convergence.

Note. The concept of *contraction mapping* is often used for function $g(x)$ on domain $[a,b]$ if there exists a constant $k < 1$ such that

$$|g(x) - g(y)| \le k|x - y| \text{ for all } x \in [a,b] \text{ and } y \in [a,b].$$

Function with $\max_{x\in[a,b]} |g'(x)| < 1$ is a special (and common) case of contraction mapping. (Why? Excercise.)

§2.4 Newton's method

Newton's method is a very popular example of fixed-point iteration $x_{n+1} = g(x_n)$ where the goal is to solve $f(x) = 0$ and the iteration function is

$$g(x) = x - \frac{f(x)}{f'(x)} \ \rightarrow \ x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}. \tag{1}$$

Intuition: The tangent line passing $(x_n, f(x_n))$ has slope $f'(x_n)$ and therefore is governed by a linear function

$$y - f(x_n) = f'(x_n)[x - x_n].$$

Use this linear function as an approximation of f so it's x-intercept approximates the root of f. By setting $y = 0$ in the above equation, we get the approximation $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$.

Advantages: Assume $f(p) = 0$ and $f'(p) \neq 0$. Then, calculation shows (do the calculation!)

$$g'(p) = 0 \tag{2}$$

which is in great favor of the smallness condition $\max_{x\in[a,b]} |g'(x)| \leq k < 1$. Indeed, because $g'(x)$ is continuous, there always exist a positive number d such that on the neighborhood $[a, b] = [p - d, p + d]$, the smallness condition is satified. Moreover, the requirement that g maps $[a, b]$ onto itself is true. (Why? For any $x_n \in [a, b] = [p-d, p+d]$, we have $|g(x_n) - p| = |g(x_n) - g(p)| = |g'(\xi)||x_n - p| < d$ and thus $x_{n+1} = g(x_n) \in [a, b]$.)

Disadvantage: if $f'(p) = 0$ or is very close to zero, then one will have a zero or small denomenator in (1). The Newton's method will become unstable (think about the round-off error) and should be avoided.

Order of convergence. The Newton's method converges at second order if the initial guess is choosen to fall in $[p - d, p + d]$, that is

$$\lim_{n\to\infty} \frac{|x_{n+1} - p|}{|x_n - p|^2} = \lambda. \tag{3}$$

Proof. Just like in the general case of fixed-point iteration, subtract the exact equation

$p = g(p)$ from the approximation $x_{n+1} = g(x_n)$.

$$\begin{aligned} x_{n+1} - p &= g(x_n) - g(p) \\ &= g(p) + g'(p)(x_n - p) + g''(\xi)(x_n - p)^2/2 \qquad \text{Taylor expansion to the } \textbf{second} \text{ order!} \\ &= g''(\xi_n)(x_n - p)^2/2 \qquad \text{due to (2).} \end{aligned}$$

Now let $n \to \infty$ and by the convergence of x_n to p and the fact that ξ_n is enclosed by x_n and p, we immediately have $\lim_{n\to\infty} \xi_n = g''(p)/2$. Thus, (3) is proved with $\lambda = |g''(p)|/2$.

Note. Fixed-point iteration methods in general are first order, since the zero derivative condition (2) is not always true.

•. A variation of Newton's method is the secant method. That is, replace the derivative $f'(x_n)$ by yet another approximation $[f(x_n) - f(x_{n-1})]/(x_n - x_{n-1})$. This the slope of the secant line connecting $(x_n, f(x_n))$ and $(x_{n-1}, f(x_{n-1}))$.

Scheme

$$\text{Initialize } x_0 \text{ AND } x_1. \text{ Then, iterate } x_{n+1} = x_n - \frac{f(x_n)}{[f(x_n) - f(x_{n-1})]/(x_n - x_{n-1})}$$

Note. Stoppping condition is usually $|x_{n+1} - x_n| < \varepsilon$ since we, handwavingly, regard x_{n+1} as very close to the exact solution p, which means, $|p - x_n| \approx |x_{n+1} - x_n|$.

Note. Both methods converge after one iteration if $f(x)$ is a linear function!