Lecture 3
Bisection, Newton's Method, Secant Method

Bisection

The Problem

Suppose we have a continuous function f on some domain $[a, b]$. Find $x^* \in [a, b]$ such that

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

Theorem 1 (Intermediate Value Theorem) Suppose we have a continous function f on some domain $[a, b]$. Then if k is some number between $f(a)$ and $f(b)$ then there exists at least one number c in the interval $[a, b]$ such that $f(c) = k$. That is,

$$\begin{aligned} f(a) < k < f(b) \quad &\text{or} \quad f(a) > k > f(b), \\ \rightarrow \quad \exists c \in [a, b] \quad &\text{s.t.} \quad f(c) = k \end{aligned}$$

Note: The Intermediate Value Theorem can be applied at $k = 0$ to find where $f(c) = 0$ when $f(a)f(b) < 0$.

Algorithm 1 Bisection.

Require: $f \in \{f : \mathbb{F} \rightarrow \mathbb{F}\}$, $a \in \mathbb{R}$, $b \in \mathbb{R}$, $k_{\max} \in \mathbb{Z}^+$, $\epsilon_x \in \mathbb{R}$, $\epsilon_f \in \mathbb{R}$.

```
a ← a_0
b ← b_0
k ← 1
while k ≤ k_max do
    c_k ← (1/2)(a_{k−1} + b_{k−1})
    f_k ← f(c_k)
    if f_k f(a_{k−1}) > 0 then
        a_k ← c_k
        b_k ← b_{k−1}
    else
        a_k ← a_{k−1}
        b_k ← c_k
    end if
    if |b_k − a_k| < ε_x or |f(c_k)| < ε_f then
        k ← k_max + 1
    end if
    k ← k + 1
end while
```

Newton's Method

Theorem 2 (Newton's Method) Given an iterate $x^{(k)}$ approximating a zero of f, the next iterate is:

$$\boxed{x^{(k+1)} = x^{(k)} - \frac{f(x^{(k)})}{f'(x^{(k)})}}$$

- Iterative procedure to locate zeros of f.
- Requires initial iterate $x^{(0)}$ to start.
- Near true zero x^* of f, iteration converges quickly.

Algorithm 2 Newton's Method

Require: f, f', $x^{(0)}$.

$k \leftarrow 0$

while $k = 0, 1, 2, ...$**until** convergence **do**

$r^{(k)} \leftarrow f(x^{(k)})$ ▷ (evaluate nonlinear residual)

$\delta x^{(k)} \leftarrow -[f'(x^{(k)}]^{-1} r^{(k)}$ ▷ (compute Newton step)

$x^{(k+1)} \leftarrow x^{(k)} + \delta x^{(k)}$ ▷ (compute next iterate)

Test for convergence ▷ (break if necessary)

end while

Definition 1 The **residual** $r^{(k)}$ is defined by:

$$r^{(k)} := f(x(^{(k)}))$$

Definition 2 The **Newton step** $\delta x^{(k)}$ is defined by:

$$\delta x^{(k)} := -[f'(x^{(k)})]^{-1} r^{(k)}$$

Example 1 Carry out Newton's method starting from $x^{(0)} = 0.75$ to find $x^{(3)}$ that approximates the real solution of $x = \cos x$.

Example 2 Carry out Newton's method starting from $x^{(0)} = 0.5$ to find $x^{(3)}$ that approximates a zero of the equation $xe^x = 2$.

Now we have to answer these three essential questions:

1. Under what conditions does the algorithm converge?
Bisection converges to some x^* such that $f(x^*) = 0$ in $[a, b]$ if f is continuous and $f(a)f(b) < 0$. If there are two or more solutions, we don't know to which one it will converge. Newton iteration converges if x_0 is *sufficiently close* to x^*. Usually, we do not know a priori how close is close enough and we must resort to trial and error.

2. How accurate will the result be?'
Both methods can give us x^* up to machine precision.

3. How fast does it converge?
In bisection, the error $|x^* - x^{(k)}|$ decreases by a factor of $\frac{1}{2}$ in each iteration. In Newton iterations, the error is approximately squared in each iteration (provided it is small enough).

$$\epsilon_0, \frac{\epsilon_0}{2}, \frac{\epsilon_0}{4}, \frac{\epsilon_0}{8}, \ldots \qquad \text{vs.} \qquad \epsilon_0, \epsilon_0^2, \epsilon_0^4, \epsilon_0^8, \ldots$$

Newton's method converges very quickly, but requires the computation of $f'(x)$. Sometimes, we cannot compute it, for instance if f is shorthand for some complicated procedure. In that case we have two options: (1) bisection or (2) the secant method.

Remark 1 Suppose we have two initial points, x_0 and x_1. Then we can estimate the derivative of f at x_1 as

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

and substitute this in the Newton iteration:

$$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)} \approx x_1 - \frac{f(x_1)(x_1 - x_0)}{f(x_1) - f(x_0)}$$

Theorem 3 (Secant Method) Given iterates $x^{(k)}$ and $x^{(k-1)}$ approximating a zero of f, compute

$$\boxed{x^{(k+1)} = x^{(k)} - \frac{f(x^{(k)})(x^{(k)} - x^{(k-1)})}{f(x^{(k)}) - f(x^{(k-1)})}} \quad (k \geq q)$$

Secant methods need two initial guesses: $x^{(0)}$ and $x^{(1)}$.

Remark 2 Some remarks on secant methods:

- This method uses a *finite difference* approximation to f'.
- Asymptotically (meaning if $|x^{(k)} - x^*|$ is small enough) the secant method converges as fast as Newton's method does.
- The secant method has extensions to problems with more than 1 unknown, but in this case Newton's method tends to be less cumbersome.
- The secant method is a *second order recurrance relation*. It relates the next approximation to the two previous approximations.
- If we can find an a and b such that $x^* \in [a, b]$, then $x_0 = a$ and $x_1 = b$ is a good starting point.

Recursion

Recurrence and iteration really mean procedures in which we repeat the same action over and over. One way to program this is by using `for` and `while` loops. We can also make the recurrent nature of the computation explicit my making the function call itself. This is called recursive programming. See the Python code below for a simple example of recursion to calculate the factorial function:

```python
def fact(k):
        if k == 1:
                return 1
        else:
                return fact(k-1) * k
```