Zeros of functions

Francesco Marchetti

Fundamentals of Computational Mathematics

Summary

Introduction to the problem

We are interested in dealing with the equation

$$f(x) = 0$$

where $f : \mathbb{R} \longrightarrow \mathbb{R}$ is a real-valued function.
Analytical solutions might be hard or even impossible to be found!
In fact, our aim is not to find **exact** solutions, but good **approximations**. In the next slides, we describe three well known **iterative** approaches, that is, we construct a sequence of approximations that are expected to get closer and closer to the real exact solution.

- The solution may be non unique (i.e. multiple roots).
- Analytical solutions might be hard or even impossible to be found.

Our focus here

The three methods we are going to discuss are: bisection (binary search), Newton-Raphson and fixed-point iteration. The first two approaches have been already discussed in previous lectures, but here (and in the lab session) we will put a focus on:

- The stopping criterion.
- The fact that the solution may be non unique (i.e. multiple roots).
- Convergence order.

Bisection

The **bisection** method iteratively restricts the interval where a **continuous** function presents a zero, it is based on the following idea.
Let $a < b$, $a, b \in \mathbb{R}$. If a continuous function f is such that $f(a)f(b) < 0$, then there exists $z \in [a, b]$ such that $f(z) = 0$.

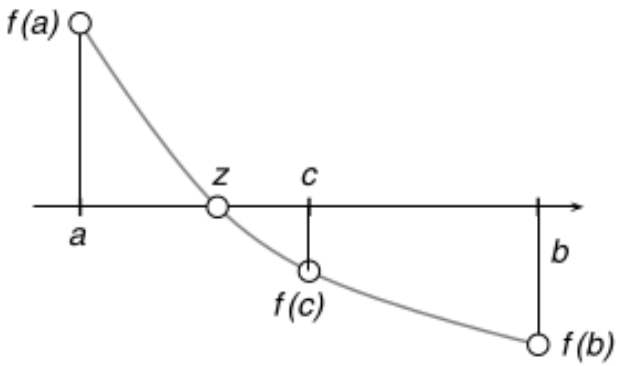
Then, we consider the center of the interval $c = (a + b)/2$. For sure, $f(a)f(c) < 0$ **or** $f(c)f(b) < 0$. Therefore, we restrict to the semi-interval so that the product is negative, and we iterate!

- The bisection method works on the interval containing the root, providing a limited region where the root lies.
- In case of multiple roots, the bisection method may fail. It is important to figure out the overall behavior of the function, and to start the procedure not so far away from the sought root.

Bisection - the algorithm

Algorithm 30 Bisection.

```
1: define error tolerance η
2: if sign f(a) = sign f(b) then stop
3: while |a − b| > η do
4:     c = a + (b − a)/2
5:     if sign f(a) ≠ sign f(c) then
6:         b = c          # (z left of c)
7:     else
8:         a = c          # (z right of c)
9:     end if
10: end while
```



... what about the error tolerance η and the stopping condition $|a - b| > \eta$? We'll come back to this in minutes.

Newton-Raphson

The Newton (-Raphson) method is based on the first order Taylor expansion of f at $z \in \mathbb{R}$: we can write for any x in the domain of f

$$f(x) = f(z) + (x - z)f'(z) + R(z),$$

where R is some remainder function. We are looking for x so that $f(x) = 0$. Therefore

$$0 = f(x) \approx f(z) + (x - z)f'(z),$$

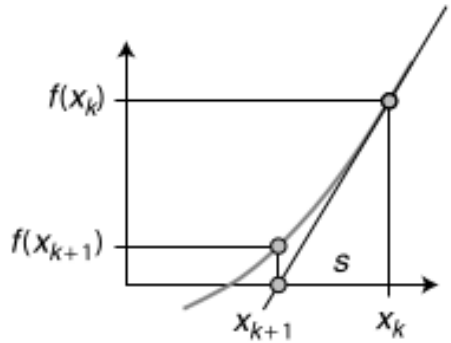
and by substituting $z = x^{(k)}$ and $x = x^{(k+1)}$ we can get the iterative method

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

Newton-Raphson - the algorithm

Algorithm 32 Newton's method for zero finding.

1: initialize starting value x_0
2: **for** $k = 0, 1, 2, \ldots$ until convergence **do**
3: compute $f(x_k)$ and $f'(x_k)$
4: $s = -\frac{f(x_k)}{f'(x_k)}$
5: $x_{k+1} = x_k + s$
6: **end for**



(The book here slightly changed the notation for the iterations)
The starting guess x_0 should be chosen carefully (as the initial interval of bisection), especially when multiple roots are present. Depending on the choice, you may converge to a different root,as we will see in the lab.
When writing **until convergence**... What do we mean?

On the stopping criterion

Every iterative method needs a **stopping criterion**, that is, we need to decide *a priori* when we will be satisfied and thus we could stop the algorithm. The stopping criterion usually relies on a fixed **tolerance** that is chosen by the user.

In the bisection algorithm, the stopping criterion is based on the length of the updated interval where the root lies. Another possibility could have been to check if $|f(c)| > \eta$, which is a condition on the residual. However, this condition fails for *flat* functions!

Another commonly-used stopping criterion is based on the relative error between subsequent iterations, that is, by defining

$$r^{(k+1)} = \frac{|x^{(k+1)} - x^{(k)}|}{|x^{(k+1)}| + 1},$$

we'll stop when $r^{(k+1)} < \eta$.

Convergence order

Let $\{x_k\}$, $k = 1, \ldots,$ be a sequence that converges to $z \in \mathbb{R}$, that is $\lim_{k\to\infty} x_k = z$.
Letting $e_k = |x_k - z|$, we say that $\{x_k\}$ converges to z with order $p \geq 1$ and rate $q > 0$ if

$$\lim_{k\to\infty} \frac{e_{k+1}}{e_k^p} = q.$$

If $p = 1$ (and $0 < q < 1$ here, otherwise we are not converging...) the convergence is **linear**, if $p = 2$ it is **quadratic** and so on.

Convergence of bisection and Newton-Raphson

- The bisection method has convergence order $p = 1$ (linear)
- Generally, the Newton-Raphson method has convergence order $p = 2$ (quadratic). However, it requires knowledge on the derivative f', which may be a strong request.
- If some approximation of f' is used, the Newton-Raphson method is likely to get slower then.
- Newton-Raphson is slow in case of roots with multiplicity greater than 1, but there are strategies to deal with this problem.

Estimating the convergence order

In practice, we may need to estimate the convergence order of a method (besides analyzing it theoretically). A practical way to estimate p is the following. Observe that for *sufficiently large* k, we have $e_{k+1} \approx q e_k^p$, $e_k \approx q e_{k-1}^p$, which leads to

$$\frac{e_{k+1}}{e_k} \approx \left(\frac{e_k}{e_{k-1}} \right)^p,$$

and then

$$p \approx \frac{\log(e_{k+1}/e_k)}{\log(e_k/e_{k-1}|)} = \frac{\log(|x_{k+1} - z|/|x_k - z|)}{\log(|x_k - z|/|x_{k-1} - z|)}.$$

Since usually we do not know z, one considers the expression

$$p \approx \frac{\log(|x_{k+1} - x_k|/|x_k - x_{k-1}|)}{\log(|x_k - x_{k-1}|/|x_{k-1} - x_{k-2}|)}.$$

Fixed point

In the **fixed point** scheme, we need to reorganize

$$f(x) = 0 \Longrightarrow g(x) = x.$$

Then, we choose a starting point $x^{(0)}$ *close* to the roots and for $k = 1, \ldots$ we iteratively compute

$$x^{(k+1)} = g(x^{(k)}).$$

Note that Newton-Raphson is a particular fixed point scheme with the choice $g(x) = x - f(x)/f'(x)$.

Fixed point - the algorithm

Algorithm 31 Fixed point iteration.

1: initialize starting value $x^{(0)}$
2: **for** $k = 1, 2, \ldots$ until convergence **do**
3: $\quad x^{(k)} = g(x^{(k-1)})$
4: **end for**

Concretely, if we are only interested in the last iteration, we may consider the following.

initialize starting value $x0$
while not converged **do**
$\quad x1 = g(x0)$
$\quad x0 = x1$
end while

Fixed point - why and when does it work?

The choice of g is not unique. The effectiveness of this approach relies on the **Ostrowski theorem** (and on the Banach fixed-point theorem).

Let z be the fixed point $g(z) = z$. If there exists a neighborhood I of z such that

- $g \in C^1(I)$,
- $|g'(z)| < 1$,

then there exists a (non unique) starting point $x^{(0)}$ sufficiently close to z so that the method works.

- The speed of convergence depends on g (zeros, regularity, value of derivatives...)...
- If the assumptions of the theorem are not satisfied, the fixed point scheme may fail or find another root in another interval.