Solution of nonlinear equations

Goal: find the roots (or zeroes) of a nonlinear function:

$$\text{given } f : [a, b] \to \mathbb{R},\ \text{find } \alpha \in \mathbb{R} \text{ such that } f(\alpha) = 0.$$

Various applications, e.g. optimization: finding stationary points of a function leads to compute the roots of f'.

When f is linear (and its graphic is a straight line) the problem is very easy. But when the analytic expression of f is more complicated, even though we have an idea of the location of its roots (with the help of graphics), we are unable to compute them exactly. Even finding the roots of polynomials of higher degree is difficult.

All the methods available in the literature are iterative, that is, they construct a sequence of values converging to the root(s): starting from an initial guess $x^{(0)}$, we construct a sequence of values $x^{(k)}$ such that

$$\lim_{k\to\infty} x^{(k)} = \alpha.$$

Questions/comments regarding iterative methods:

- Does the sequence converge?
- Does convergence depend on the initial guess $x^{(0)}$? (In general, yes)
- How fast is the convergence? (or: what is the *order of convergence*?)
- When to stop the procedure? (how many iterations should we do? need for reliable stopping criteria)

Bisection method

Simplest and robust method, based on the intermediate value theorem:

Theorem

(Bolzano) *Let* $f : [a, b] \to \mathbb{R}$ *be a continuous function that has opposite signs in* $[a, b]$ *(meaning, to be precise, that* $f(a)f(b) < 0$*). Then there exists* $\alpha \in]a, b[$ *such that* $f(\alpha) = 0$.

Remark

Note that the root α does not need to be unique (take $f(x) = cos(x)$ on $[0, 3\pi]$). Hence, under the hypotheses of Bolzano's theorem, we will look for *a root of the equation* essentially without choosing which one.

Bisection method

Idea: to construct a sequence by repeatedly bisecting the interval and selecting to proceed the sub-interval where the function has opposite signs. In the hypotheses of the intermediate value Theorem, we will

- divide the interval in two by computing the midpoint c of the interval: $c = (a+b)/2$;
- compute the value $f(c)$;
- if $f(c) = 0$ we found the root (very unlikely, but can be replaced by a stopping criterion),
- if not, that is, if $f(c) \neq 0$, there are two possibilities:
  - $f(a)f(c) < 0$ (and then f has opposite signs on $[a, c]$),
  - or $f(b)f(c) < 0$ (and then f has opposite signs on $]c, b[$).

The method selects the subinterval where f has opposite signs as the new interval to be used in the next step. In this way an interval that contains a zero of f is reduced in width by $1/2$ at each step. The process is continued until the interval is sufficiently small.

Pseudocode

Input: the function f; the endpoints a, b; the tol $\in \mathbb{R}^+$ and MAXITER $\in \mathbb{N}$

$a^{(1)} \leftarrow a$ and $b^{(1)} \leftarrow b$

for $k = 1, \ldots, \text{MAXITER}$

    $c^{(k)} = (a^{(k)} + b^{(k)})/2$

    if $f(c^{(k)}) = 0$ or $(b^{(k)} - a^{(k)})/2 <$ tol

        Output: $c^{(k)}$

        Stop the algorithm

    end

    if $\mathrm{sign}(f(c^{(k)})) = \mathrm{sign}(f(a^{(k)}))$

        $a^{(k+1)} \leftarrow c^{(k)}$ and $b^{(k+1)} \leftarrow b^{(k)}$

    else

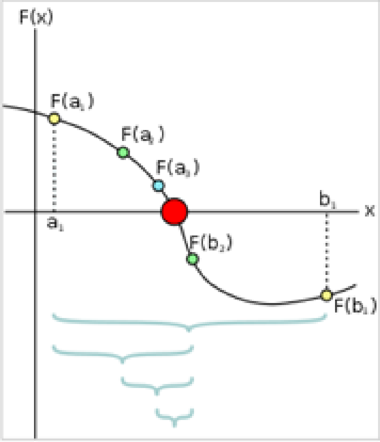
        $a^{(k+1)} \leftarrow a^{(k)}$ and $b^{(k+1)} \leftarrow c^{(k)}$

    end

end

Print "exceeded MAXITER iterations without reaching the tolerance".

Example



A few steps of the bisection method applied over the starting range $[a_1;b_1]$. The bigger red dot is the root of the function.

Analysis

In the hypotheses of Bolzano's theorem (f continuous with opposite signs at the endpoints of its interval of definition) the bisection method **converges always** to a root of f, but it is very slow: the absolute value of the error is halved at each step, that is, the method converges linearly.

If $c^{(1)}$ is the midpoint of [a,b], and $c^{(n)}$ is the midpoint of the interval at the n-th step, the error is bounded by

$$|c^{(n)} - \alpha| \leq \frac{b-a}{2^n}$$

This relation can be used to determine in advance the number of iterations needed to converge to a root within a given tolerance:

$$\frac{b-a}{2^n} \leq TOL \implies n \geq \log_2(b-a) - \log_2 TOL$$

Example: $b - a = 1$, TOL$= 10^{-3}$ gives $n \geq 3\log_2 10$, TOL$= 10^{-4}$ gives $n \geq 4\log_2 10$ and so on. Since $\log_2 10 \simeq 3.32$, to gain one order of accuracy we need a little more than 3 iterations.