Recherche de zéros : l'algorithme de dichotomie

Proposition 1 (*Théorème des valeurs intermédiaires*)

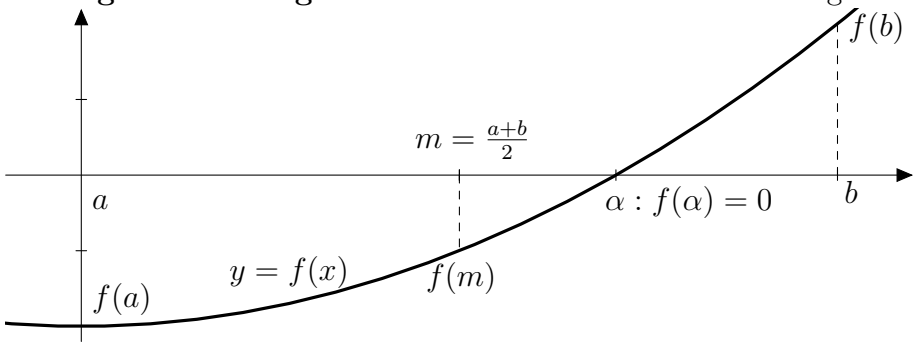
Une fonction continue qui change de signe sur un intervalle s'y annule.

Soit $f : [a, b] \to \mathbb{R}$ une fonction continue ($a < b$) On suppose que $f(a).f(b) < 0$: la fonction f change de signe entre a et b. Par conséquent elle s'annule au moins une fois :

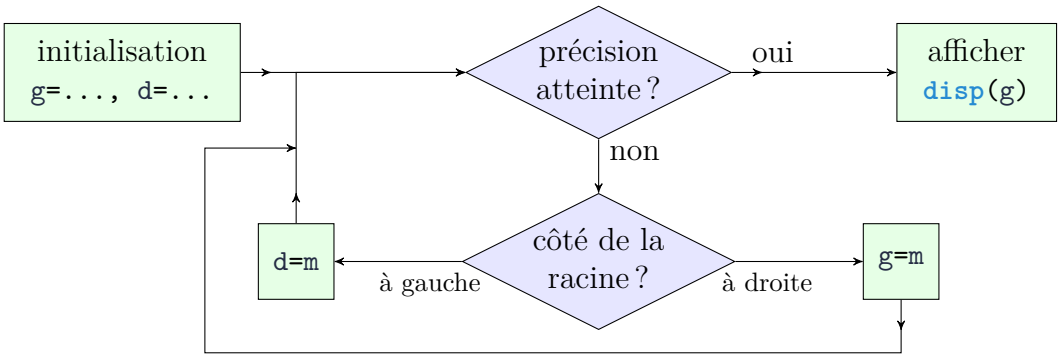
$$\exists \alpha \in]a, b[,\ f(\alpha) = 0.$$

Pour se fixer les idées, on pense à $f : x \mapsto x^2 - 2$, $[a; b] = [0; 2]$, et $\alpha = \sqrt{2}$ est unique.

On **subdivise** le segment $[a, b]$ par le milieu $m = \frac{a+b}{2}$ en deux sous-segments : $[a, m]$ et $[m, b]$, et on trouve dans quel sous-segment (**de quel côté** de m) se trouve α en cherchant dans lequel il y a **changement de signe**. On « zoome » alors sur ce sous-segment.



L'**algorithme de dichotomie** consiste à répéter cette démarche jusqu'à ce qu'on soit « assez proche » de α.



Proposition 2 (*Convergence de la méthode de dichotomie*)

On définit deux suites (a_n) et (b_n) par : $a_0 = a,\ b_0 = b$, et, pour $n \in \mathbb{N}$, on pose $m_n = \frac{a_n+b_n}{2}$: le milieu du segment $[a_n; b_n]$, et :

si $f(a_n)f(m_n) > 0$, alors : $\begin{cases} a_{n+1} &= m_n \\ b_{n+1} &= b_n, \end{cases}$ sinon : $\begin{cases} a_{n+1} &= a_n \\ b_{n+1} &= m_n \end{cases}$

Alors ces deux suite (a_n) et (b_n) sont adjacentes, et elles convergent vers une solution α de $f(\alpha) = 0$.

dichotomie.sce

```
// Constantes : données du problème
GAUCHE_INIT = 0                         // encadrement initial de la racine
DROITE_INIT = 2
PRECISION   = 10^(-3)                   // précision souhaitée
function y = f(x)                       // fonction étudiée
  y = x.^2 - 2
endfunction

// initialisation
gauche = GAUCHE_INIT
droite = DROITE_INIT

// la boucle de l'algorithme
while ( gauche - droite > PRECISION ) // jusqu'à avoir la bonne précision
  milieu = (gauche+droite) / 2
  if ( f(gauche) * f(milieu) > 0 )
    then gauche = milieu                // si la racine est à droite de `milieu`
    else droite = milieu                // si la racine est à gauche de `milieu`
  end // fin du `if`
end   // fin du `while`

// affichage
disp ("Approximation par défaut de sqrt(2) à 10^(-3) près :")
disp (gauche)
```