Table de Matières

REMERCIEMENTS

Nous saisissons cette occasion pour adresser nos sincères remerciements à tous ceux qui ont participé de près ou de loin à l'élaboration de ce projet, en particulier notre bienveillant encadrant le professeur Mr. Maslouhi, et l'ensemble des professeurs et corps administratif de l'ENSA de Kenitra.

Ils nous ont assuré un cadre et un environnement de travail adéquats. Notamment, la formation et les outils nécessaires à la réalisation de ce projet, entre autre : des cours basiques très informatifs présentés dans leurs meilleurs aspects, des travaux dirigés qui explicitent clairement les notions appris, et des travaux pratiques qui mettent en oeuvre notre apprentissage. Tout un bagage dont l'ingénieur aurait besoin pour le reste sa formation.

Une spéciale dédicace à Monsieur Moulay Taïb Belghiti qui nous a été d'une grande aide dans le choix de notre sujet, en nous orientant vers un sujet qui pourrait nous être utile pour nos filières souhaitées.

Je tiens à remercier également nos pères qui nous ont marqué un soutien moral remarquable et à qui nous devons de la fierté et du respect.

Introduction

De nombreuses métaheuristiques permettent de résoudre des problèmes d'optimisation non linéaires à variables continues. Cependant, la plupart des méthodes actuelles ne sont pas conçues pour résoudre efficacement des problèmes dimensionnellement grands, et leurs performances se détériorent rapidement lorsque la dimensionnalité de l'espace de recherche augmente. Ces problèmes particuliers apparaissent pourtant de plus en plus souvent dans des applications telles que *le data* ou *mining*. Ainsi, des méthodes de résolution dimensionnellement robustes sont de plus en plus sollicitées. Nous proposons dans cet article quelques méthodes issues de l'optimisation unidimensionnelle permettant de résoudre rapidement des problèmes unimodaux et multimodaux de grande dimensionnalité.

Notion d'optimisation

L'optimisation est un ensemble de techniques permettant de trouver les valeurs des variables qui rendent optimale une fonction de réponse, appelée aussi **fonction objectif** ou **critère**. Sur le plan mathématique, cela correspond à la recherche des extrémums de fonctions à plusieurs variables.

C'est donc un problème de minimisation en dimension n. Si on cherche $\tilde{x}$ qui maximise f, un changement de signe de f permet de se ramener au problème précédent. Pour simplifier l'écriture, on considère dans la suite uniquement des problèmes de minimisation.

Optimisation unidimensionnelle

L'optimisation unidimensionnelle est l'étude des minimas ou des maximas des fonctions de la variable réelle. Ce n'est pas un exercice gratuit de s'y intéresser uniquement en dimension un. En effet, d'une part on n'y retrouve souvent les mêmes difficultés que dans les dimensions supérieures (existence de minima locaux, non-différentiabilité, coût élevé d'évaluation de la fonction, etc.), d'autre part des méthodes d'optimisation unidimensionnelle sont utilisées sous forme de « procédures de recherche linéaire » dans les algorithmes d'optimisation vectorielle.

Enfin, il est intéressant de se poser au moins une fois le problème de trouver le minimum d'une fonction que l'on ne peut pas représenter : en effet, un minimum, un maximum, un zéro d'une fonction, cela saute aux yeux lorsque l'on trace la fonction. Mais si l'on peut tracer la fonction, il est clair que l'on triche : on dispose de toute l'information en même temps, et il suffit de faire glisser verticalement la règle vers le haut ou vers le bas et de s'arrêter lorsqu'un seul point de la courbe touche la règle. Or, dans la pratique, il est important de comprendre que l'on ne dispose que d'une information locale sur la fonction à minimiser. Cette information peut être un ensemble de valeurs de la fonction en des points (forcément en nombre fini, et donc de mesure nulle) choisis par l'utilisateur. On peut aussi disposer de valeurs de la dérivée de cette fonction ou d'approximations de cette dérivée (différences finies ou valeurs exactes entachées d'un bruit de mesure ou de calcul), ou d'indications sur le comportement à l'infini de la fonction. Pour se représenter la situation dans laquelle se trouve réellement l'optimiseur, il suffit de tracer le graphe d'une fonction (éventuellement avec des discontinuités) au crayon, d'en marquer à l'encre quelques points en tangentes, puis d'effacer le graphe. Il s'agit maintenant d'élaborer une stratégie intelligente pour choisir en quels points on désire plus d'informations, sachant que celle-ci coûte cher. On réalisera alors d'une part que les algorithmes présentés ici ne marchent bien que parce que l'on a supposé une très grande régularité des fonctions considérées (dérivabilité, convexité, unimodalité) , d'autre part que ces « recettes de cuisine numérique » font preuve dans le cadre de ces hypothèses simplificatrices - d'une extrême ingéniosité, tirant parfois le maximum de l'information disponible.

Les méthodes itératives de minimisation font souvent appel, à chaque étape, à une minimisation dans $\mathbb{R}$. Ceci constitue une justification de ce chapitre d'introduction dans lequel on étudie quelques méthodes de minimisation unidirectionnelle, c'est-à dire de fonctions définies de $\mathbb{R}$ dans $\mathbb{R}$, et leurs propriétés.

On s'intéressera dans cette partie à l'optimisation de fonctions objectives monovariables (Fonctions de $\mathbb{R}$ dans $\mathbb{R}$). L'intérêt des algorithmes d'optimisation unidimensionnelle ne vient pas seulement du fait que dans les applications on rencontre naturellement des problèmes unidimensionnels, mais aussi du fait que l'optimisation unidimensionnelle est un composant fondamental de bon nombre de méthodes d'optimisation multidimensionnelle.

Chapitre 1

Vocabulaire

1.1 Minimum global

Soit $f : [a,b] \subset \mathbb{R} \to \mathbb{R}$ une fonction. $f(\tilde{x})$ est le **minimum global** (ou absolu) de f si et seulement si :

$$\forall x \in [a,b], f(x) \geq f(\tilde{x})$$

$\tilde{x}$ est un minimiseur global de f.

- **Minimum strict :** $x \neq \tilde{x} \Rightarrow f(x) > f(\tilde{x})$

1.2 Minimum local

Soit $f : [a,b] \subset \mathbb{R} \to \mathbb{R}$ une fonction. $f(\tilde{x})$ est le **minimum local** de f si et seulement si :

$$\exists \epsilon, \forall x \in]\tilde{x}-\epsilon, \tilde{x}+\epsilon[, f(x) \geq f(\tilde{x})$$

$\tilde{x}$ est un minimiseur local de f.

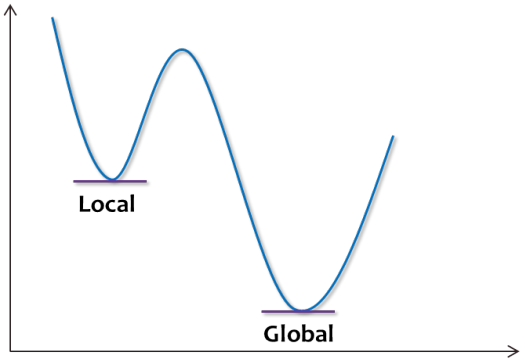
- **Minimum local strict :** $x \in]\tilde{x}-\epsilon, \tilde{x}+\epsilon[\setminus \{\tilde{x}\} \Rightarrow f(x) > f(\tilde{x})$

1.3 Théoréme (Minimalité local)

Soit $f : [a,b] \subset \mathbb{R} \to \mathbb{R}$ une fonction de C^2. Si $\tilde{x} \in \mathbb{R}$ et vérifie les conditions :

$$\begin{cases} f'(\tilde{x}) = 0 \\ f''(\tilde{x}) > 0 \end{cases}$$

Alors, $f(\tilde{x})$ est un minimum local strict de f.



1.4 Fonction unimodale

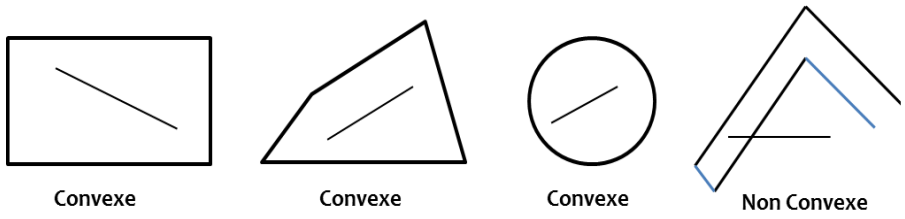
Une fonction f, partout définie sur I, est dite ***unimodale*** sur I :

- Elle admet un unique minimum $\tilde{x}$ dans l'intérieur de I
- Elle est strictement décroissante sur : $I \cap]-\infty, \tilde{x}[$ et strictement croissante sur : $I \cap]\tilde{x}, +\infty[$

1.5 Convexité

Un ensemble S est dit ***convexe***, si pour toute paire de points a, b de S, S contient le segment $[ab]$.

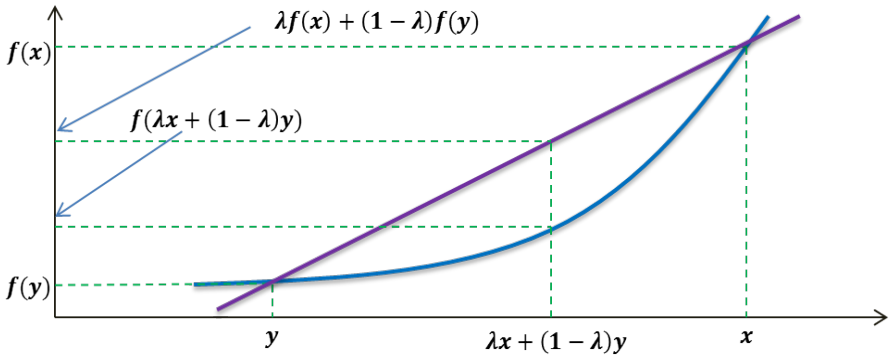
$$a, b \in S \Rightarrow (0 \leq \lambda \leq 1 \Rightarrow a + (1 - \lambda)b \in S$$



1.6 Fonction convexe

Une fonction f est dite ***convexe*** sur un ensemble S si et seulement si :

$$\forall x, y \in S, 0 \leq \lambda \leq 1 \Rightarrow f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda)f(y)$$



1.7 Intérêt de la convexité

Théorème

Si f est convexe dans un ensemble convexe S, alors tout minimum local de f est un minimum global.

Démonstration

Soit a un minimum local, et V l'ouvert contenant a dans lequel :

$$x \in V \Rightarrow f(x) \geq f(a)$$

Si on suppose qu'il existe un point $b \in S$ tel que $f(b) < f(a)$, alors on a :

$$f(\lambda a + (1-\lambda)b) \leq \lambda f(a) + (1-\lambda)f(b)$$

Il est possible de trouver un λ suffisamment proche de 1 pour que $x = \lambda a + (1 - \lambda b)$ soit dans V . Contradiction en ce point :

$$f(x) \leq \lambda f(a) + (1-\lambda)f(b) \leq f(a)$$

1.8 Convexité et unimodalité

Théorème

Si f est strictement convexe sur I et atteint son minimum sur I en un point $\tilde{x}$ dans l'intérieur de I, elle est ***unimodale***

$$Strictement\ \ convexe \Rightarrow Unimodale$$

Démonstration

Si $m < x < \tilde{x}$, il existe α dans l'intervalle $]0,1[$ tel que : $x = as + (1-\alpha)\tilde{x}$, d'où :

$$f(x) < \alpha f(m) + (1-\alpha)f(\tilde{x}) \leq f(m)$$

Donc f est strictement décroissante sur $I \cap]-\infty, \tilde{x}[$. On montre de même que f est strictement croissante sur $I \cap]\tilde{x}, +\infty, [$.

Corollaire

Supposons que f strictement convexe sur $[0, +\infty[$, f dérivable en 0, $f'(0) < 0$, et : $f(x) \to +\infty$ lorsque $x \to +\infty$. Alors f est unimodale sur $[0, +\infty[$.
Si f est strictement convexe sur $\mathbb{R}$, elle y est unimodale.

Chapitre 2

Méthode de Dichotomie

2.1 Principe

Si l'on dispose de a et b éléments de I tel que $f(a).f(b) < 0$, alors le théorème des valeurs intermédiaires nous assure que l'équation $f(x) = 0$ admet une unique solution $\tilde{x}$ dans $]a, b[$.
Pour approcher la solution, on construit des suites $(a_n)_n$ et $(b_n)_n$ tel que : $a_0 = a$ et $b_0 = a$, pour $n \in \mathbb{N}$:

- Si $f(a_n).f(\frac{a_n+b_n}{2}) \leq 0$, on est sûr qu'il y a une zéro de f entre a_n et $\frac{a_n+b_n}{2}$, on pose ainsi : $a_{n+1} = a_n$ et $b_{n+1} = \frac{a_n+b_n}{2}$.

- Sinon, on pose $a_{n+1} = \frac{a_n+b_n}{2}$ et $b_{n+1} = b_n$

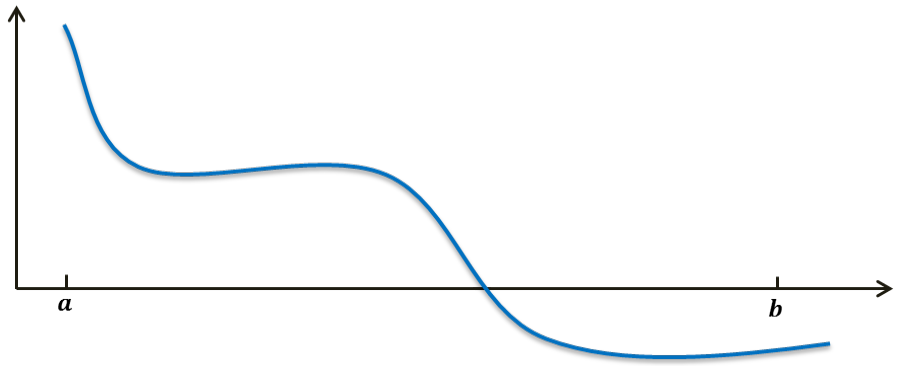
On construit de cette manière deux suites, convergentes vers une même limite l qui vérifient de plus :

$$\forall n \in \mathbb{N}; f(a_n).f(b_n) \leq 0$$

2.1.1 Schématisation

Une fois que l’on connait l’intervalle sur lequel la fonction est croissante ou décroissante (strictement monotone) et qui passe par zéro, il nous reste à le localiser. Pour cela, on utilise la méthode la plus élémentaire : **la méthode dichotomique**.

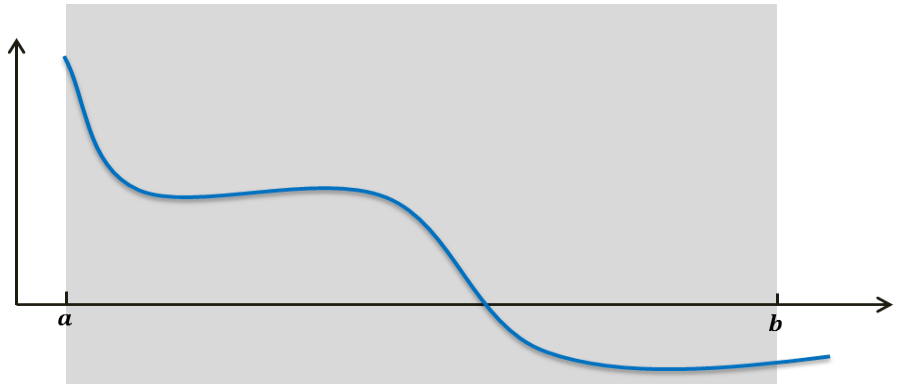
Prenons donc une fonction définie entre deux nombres a et b comme ceci :



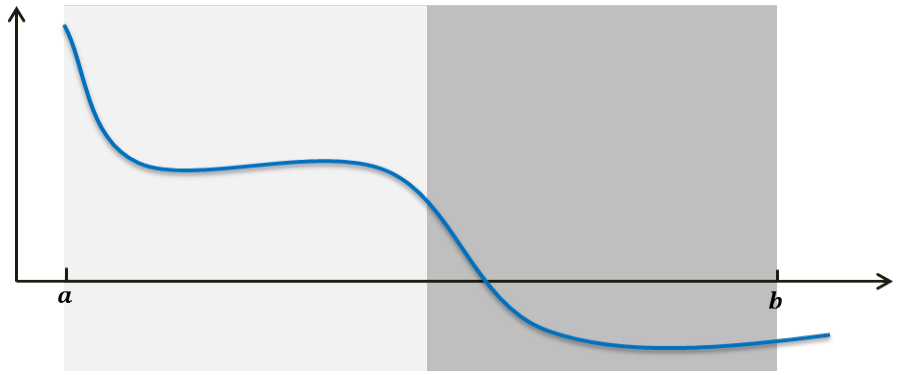
Ici la fonction est décroissante. Cependant, ce qui va suivre s’applique aussi bien pour une fonction croissante.

Au début, tous ce qu’on sait c’est que la solution de notre équation est comprise entre a et b.

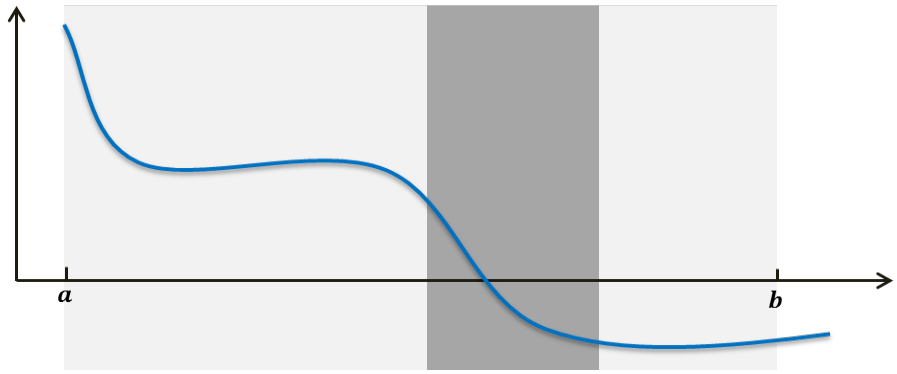
Schématisons ceci par une zone grise que nous allons réduire pas à pas :



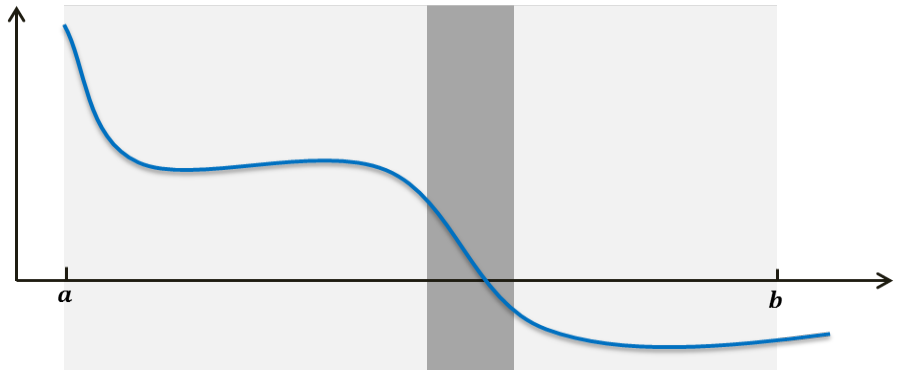
On regarde la valeur de la fonction au milieu de l’intervalle, c’est-à-dire pour la valeur moyenne $\frac{a+b}{2}$. Après avoir testé la négativité du produit des bornes des deux intervalles, on suppose que :



La zone dans laquelle notre zéro se trouve donc divisée par deux. Il n’y a plus qu’à répéter le même processus avec le nouvel intervalle pour se rapprocher encore plus de notre zéro :



Et voici la troisième étape :



2.1.2 Convergence

Pour ce qui est de la convergence, il suffit de remarquer que $(a_n)_n$ et $(b_n)_n$ sont des suites adjacentes : (a_n) est croissante, (b_n) est décroissante.

Par une simple récurrence on peut démontrer que :

$$b_n - a_n \leq \frac{b-a}{2^n}$$

Preuve

Donc $b_n - a_n \to 0$. Ainsi (a_n) et (b_n) converge vers la même limite l.
Supposons que $f(l) \neq 0$. Puisque f est continue en l, f garde un signe constant au voisnage de l : $]l-\epsilon, l+\epsilon[$ tel que $\frac{b-a}{2^N} \leq \epsilon$. Puisque $l \in [a_N, b_N]$, on a $[a_N, b_N] \subset]l-\epsilon, l+\epsilon[$. Donc $f(a_N)$ et $f(b_N)$ sont de même signe, ce qui est absurde par construction.
Ainsi on a bien la convergence.

Exercice

On considère l'équation : $(E) : x^3 + x^2 + 2x - 6 = 0$

1. Démontrer que cette équation admet une unique solution réelle α et que α appartient à l'intervalle $]1,2[$.
2. Déterminer par la méthode de dichotomie, une valeur approchée à 10^{-4} près de la solution de l'équation (E).

Corrigé

1. On pose $f(x) = x^3 + x^2 + 2x - 6$
   f est dérivable sur $\mathbb{R}$, donc $f'(x) = 3x^2 + 2x + 2$

   on a $f'(1) > 0$ et $f'(2) > 0$ donc f est strictement croissante sur $]1,2[$. D'où :
   $$\exists! \alpha \in]1,2[, f(\alpha) = 0$$
2. **Etape 1** : On a $f(1).f(2) < 0$ car : $f(1) = -2$ et $f(2) = 10$

   $$|1 - 2| > 10^{-4}$$

   Etape 2 : On a $f(1).f(\frac{3}{2}) < 0$ car : $f(1) = -2$ et $f(\frac{3}{2}) = \frac{21}{8}$

   $$|1 - \tfrac{3}{2}| > 10^{-4}$$

   Etape 3 : On a $f(1).f(\frac{5}{4}) < 0$ car : $f(1) = -2$ et $f(\frac{5}{4}) = \frac{1}{64}$

   $$|1 - \tfrac{5}{4}| > 10^{-4}$$

   Et ainsi de suite jusqu'à ce que $|a - b| < 10^{-4}$. Donc,

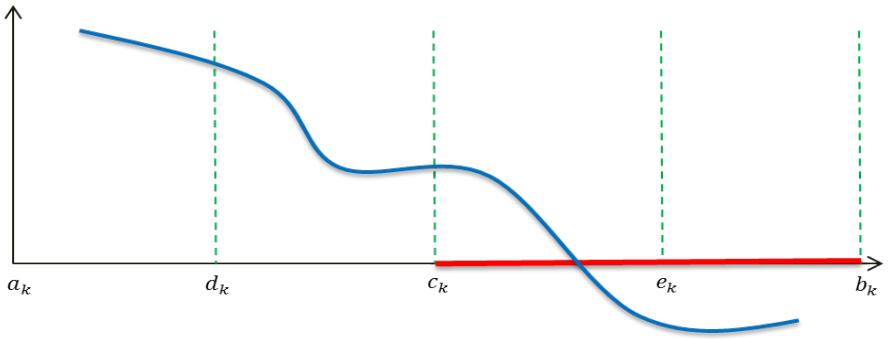
   $$1.248291016 \leq \alpha \leq 1.248352051$$

2.2 Dichotomie sans dérivation

On suppose que f est unimodale. On considère les suites $(a_n)_{n\in\mathbb{N}}$, $(b_n)_{n\in\mathbb{N}}$, $(c_n)_{n\in\mathbb{N}}$, $(d_n)_{n\in\mathbb{N}}$ et $(e_n)_{n\in\mathbb{N}}$

$$\begin{cases} c_k = \frac{a_k+b_k}{2} \\ d_k = \frac{3a_k+b_k}{4} \\ e_k = \frac{a_k+3b_k}{4} \end{cases}$$

- Si $f(c_k) > min(f(e_k), f(d_k))$ alors :
  - Si $min(f(e_k), f(d_k)) = f(e_k)$ donc : $\begin{cases} a_{k+1} = c_k \\ b_{k+1} = b_k \end{cases}$
  - Si $min(f(e_k), f(d_k)) = f(d_k)$ donc : $\begin{cases} a_{k+1} = a_k \\ b_{k+1} = c_k \end{cases}$
- Si $f(c_k) < min(f(e_k), f(d_k))$ alors : $\begin{cases} a_{k+1} = d_k \\ b_{k+1} = e_k \end{cases}$



2.3 Dichotomie d'une fonction dérivable

On suppose que f est unimodale et dérivable.

On considére les suites $(a_n)_{n\in\mathbb{N}}$ et $(b_n)_{n\in\mathbb{N}}$ tel que : $a_0 = a$ et $b_0 = b$.
Si $f'(a_k).f'(b_k) < 0 \Rightarrow \tilde{x} \in [a_k, b_k]$

On calcule $f'(\frac{a_k+b_k}{2})$.

- Si $f'(\frac{a_k+b_k}{2}).f'(a_k) < 0$ donc : $\tilde{x} \in [a_k, \frac{a_k+b_k}{2}]$
- Si $f'(b_k).f'(\frac{a_k+b_k}{2}) < 0$ donc : $\tilde{x} \in [\frac{a_k+b_k}{2}, b_k]$
- Si $f'(\frac{a_k+b_k}{2}) = 0$ donc : $\tilde{x} = \frac{a_k+b_k}{2}$

2.4 Dichotomie : Application

La procédure traduisant la méthode de dichotomie est présentée comme suit :

```
restart;
dichotomie :=proc(f, a, b, epsilon)
local an, bn, cn;
an := evalf(a);
bn := evalf(b);
while bn − an > epsilon do
        cn := (an + bn)/2;
        if f(cn) = 0 then
            return cn;
        else
            if f(an)·f(cn) < 0 then
                bn := cn;
            else
                an := cn;
            end if;
        end if;
end do;
return an;
end proc;
```

Exemple :

$$f := x \to \sin(x);$$

$$x \to \sin(x) \quad (2)$$

$$dichotomie(f, 3, 4, 10^{-8});$$

$$3.141592650 \quad (3)$$

$$identify(3.141592650);$$

$$\pi \quad (4)$$

$$g := x \to \exp(2 \cdot x) - x - 2;$$

$$x \to e^{2x} - x - 2 \quad (5)$$

$$dichotomie(g, 0, 1, evalf(10^{-8}));$$

$$0.4475421607 \quad (6)$$

$$fsolve(g(x) = 0, x = 0..1);$$

$$0.4475421606 \quad (7)$$

Chapitre 3

Méthode de la section dorée

La méthode de la section dorée ou section d'or est tardive et due à l'allemand Zeising,au milieu du **XlX** ème siècle.
Les plus anciennes définitions, et construction géométrique de la section d'or remonte au **III** ème siècle avant JC. Elle est due au mathématicien grec Euclide, dans son ouvrage « les Eléments ».
Cette méthode est presque la même que la précédente, à la différence près qu'au lieu de découper l'intervalle d'incertitude en 4, on le découpe en 3, se qui ne côute, à chaque itération, qu'une seule évaluation supplémentaire de la fonction.

Notons qu'il est impossible de découper l'intervalle courant en trois morceaux égaux à chaque itération. En effet, découpons S en trois intervalles de longeur égales : $S = [a,b] = [x_0, x_{N+1}] = [x_0, x_1] \cup [x_1, x_2] \cup [x_2, x_{N+1}]$.

La connaissance des valeurs de la fonction f en x_1 et x_2 (avec $x_1 < x_2$), permet de réduire l'intervalle d'incertitude:

- Si $f(x_1) > f(x_2)$, on conserve l'intervalle $[x_1, x_{N+1}]$.

- Sinon on conserve l'intervalle $[x_0, x_2]$.

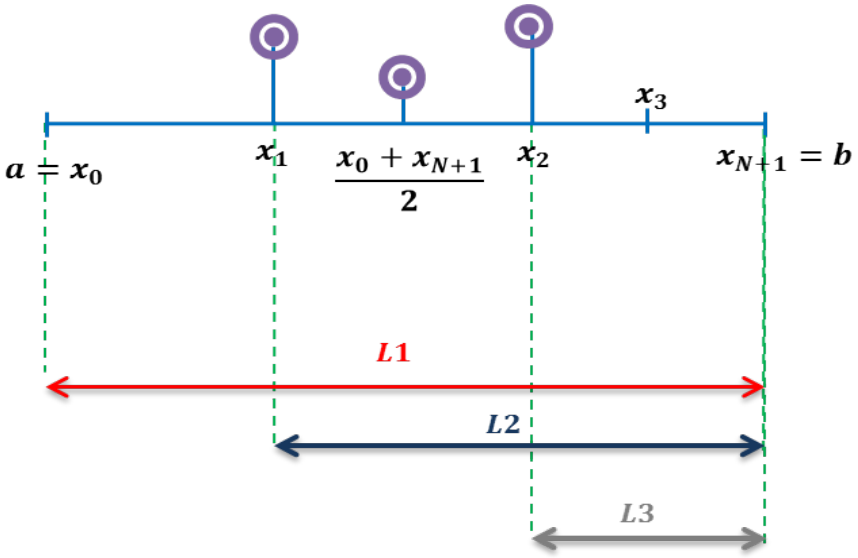
Supposons qu'après avoir comparé comme ci-dessus les valeurs $f(x_1)$ et $f(x_2)$ on ait éliminé le premier sous-segment $[x_0, x_1]$, donc l'intervalle $[x_1, x_{N+1}]$ est retenu. Une stratégie naturelle consiste à choisir le prochain point d'évaluation symétriquement à x_2 par rapport à x_1 et x_{N+1}. Le nouveau point x_3 où évaluer la fonction serait forcément dans l'intervalle complémentaire $[x_1, x_{N+1}]$, d'un côté ou de l'autre du milieu x_2, rompant ainsi la symétrie : les trois nouveaux intervalles

ne peuvent donc être égaux. Bien que cela semble étonnant à première vue, il est possible d'obtenir une réduction (relative) constante de l'intervalle d'incertitude à chaque itération, tout en découpant l'intervalle en trois morceaux inégaux! Et donc, on aura quand même une vitesse de convergence linéaire.

En effet, notons L_k la longeur de l'intervalle d'incertitude à l'itération k, c'est-à-dire $L_k = L([x_1^k, x_{N+1}^k])$. On désire avoir :

$$\frac{L_{k+1}}{L_k} = \gamma < 1$$

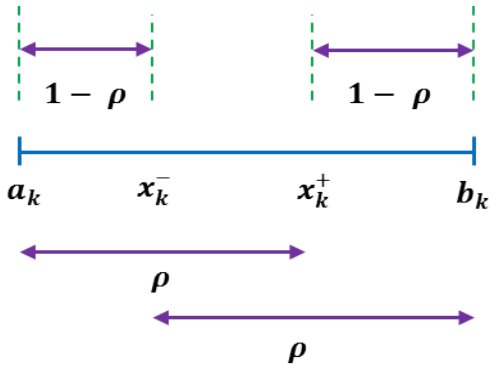
Pour que cette relation soit vraie, il faut que les intervalles $[x_1^k, x_2^k]$ et $[x_3^k, x_{N+1}^k]$ soient symétriques par rapport au milieu de $[x_1^k, x_{N+1}^k]$, puisqu'on ne sait pas lequel d'entre eux va être éliminé.



On a : $L_k = L_{k+1} + L_{k+2}$.
On divise par : L_{k+1}, on aura :

$$\frac{L_k}{L_{k+1}} = 1 + \frac{L_{k+2}}{L_{k+1}} \Rightarrow \frac{1}{\gamma} = 1 + \gamma$$

Cet équation admet comme unique solution positive $\gamma = \frac{1}{2}(\sqrt{5}-1)$.
Précisément :



Soit $[a_k, b_k]$ l'intervalle de recherche à l'instant k, on pose :

$$\frac{x_k^+ - a_k}{b_k - a_k} = \rho$$

Donc :

$$x_k^- - a_k = b_k - x_k^+ = (b_k - a_k) - (x_k^+ - a_k) = (b_k - a_k) - \rho(b_k - a_k) = (1-\rho)(b_k - a_k)$$

D'où:

$$\begin{cases} x_k^- = a_k + (1-\rho)(b_k - a_k) = \rho a_k + (1-\rho)b_k \\ x_k^+ = a_k + \rho(b_k - a_k) = (1-\rho)a_k + \rho b_k \end{cases}$$

Avec $0.5 < \rho < 1$. A l'itération suivante, si le minimum est à droite de x_k^- donc: $a_{k+1} = x_k^-$ et $b_{k+1} = b_k$.
On souhaite réutiliser x_k^+, c'est-à-dire que l'on veut :

$$x_{k+1}^- = x_k^+$$

Donc :

$$\rho a_{k+1} + (1-\rho)b_{k+1} = (1-\rho)a_k + \rho b_k$$
$$\rho(\rho a_k + (1-\rho)b_k) + (1-\rho)b_k = (1-\rho)a_k + \rho b_k$$
$$\rho^2 + \rho - 1 = 0$$

C'est une équation du second degré. On a :

$$\delta = 1^2 - 4 \times 1 \times (-1) = 1 + 4 = 5 > 0$$

Donc l'équation admet deux solutions : $\begin{cases} \rho = \frac{-1-\sqrt{5}}{2} \\ \rho = \frac{-1+\sqrt{5}}{2} \end{cases}$.
Comme $0,5 \prec \rho \prec 1$, on obtient :

$$\rho = \frac{-1+\sqrt{5}}{2}$$

.

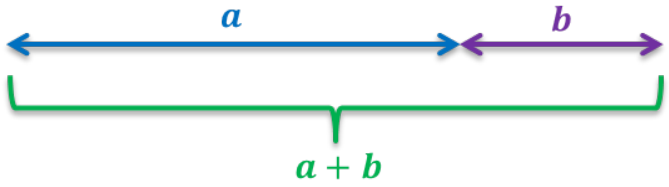
ρ est l'inverse du nombre d'or $\varphi = \frac{1+\sqrt{5}}{2}$. Ce qui explique le nom de la méthode (*Golden section search*).

La convergence de cette méthode est linéaire avec un taux $\rho \approx 0,61803$, ce qui est moins bien que la recherche dichotomique ou la méthode de Newton. Le principal avantage de cette méthode est qu'elle ne nécessite pas de calculer la dérivée.
Cette méthode est souvent améliorée en utilisant, quand cela est judicieux, **une interpolation parabolique** ou cubique.

Note : Nombre d'or

Le nombre d'or est une proportion, définie initialement en géométrie comme l'unique rapport $\frac{a}{b}$ entre deux longueurs a et b tel que le rapport de la somme des deux longueurs $(a+b)$ sur la plus grande a (par exemple) soit égal à celui de la plus grande a sur la plus petite b c'est-à-dire lorsque :

$$\frac{(a+b)}{a} = \frac{a}{b}$$



Le nombre d'or est maintenant souvent désigné par la lettre φ (phi). Ce nombre irrationnel est l'unique solution positive de l'équation : $x^2 = x + 1$. Il vaut exactement : $\frac{1+\sqrt{5}}{2}$, soit approximativement $1,6180339887$.

3.1 Algorithme

L’algorithme de la section dorée peut s’écrire sous les deux formes :
Sur Maple :

Forme 1 :

```
Section := proc(f, a, b, epsilon)
local an, bn, rho, x, y, f1, f2;
an := evalf(a); bn := evalf(b);
rho := (sqrt(5) - 1)/2;
x := a + (1 - rho)·(b - a);
y := a + rho·(b - a);
f1 := f(x); f2 := f(y);
while b - a > epsilon do
    if f1 > f2 then
        an := x; x := y;
        f1 := f2;
        y := a + rho·(b - a);
        f2 := f(y);
    else
        b := y; y := x;
        f2 := f1;
        x := a + (1 - rho)·(b - a);
        f1 := f(x);
    end if;
end do;
return (an + bn)/2;
end proc;
```

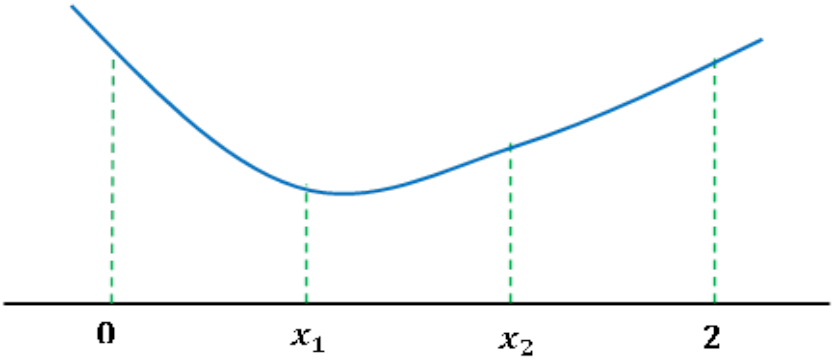
Forme 2 :

```
restart;
section :=proc(f, a, b, epsilon)
 local c, d, rho;
rho := (sqrt(5) - 1)/2;

while b - a > epsilon do
    c := rho·a + (1 - rho)·b;
    d := a + b - c;
    if (f(c) < f(d)) then
        b := d;
    else
        a := c;
     end if;
end do;
return c;
end proc;
```

Exemple

Utilisation de la méthode de la section dorée pour minimiser : $f(x) = 0.5 - xe^{-x^2}$



x_1	$f(x_1)$	x_2	$f(x_2)$
0.764	0.074	1.236	0.232
0.472	0.122	0.764	0.074
0.764	0.074	0.944	0.113
0.656	0.074	0.764	0.074
0.584	0.085	0.652	0.074
0.652	0.075	0.695	0.071
0.695	0.071	0.721	0.071
0.679	0.072	0.695	0.071
0.695	0.071	0.705	0.071
0.705	0.071	0.711	0.071

Chapitre 4

Méthode de Newton

4.1 Optimisation

En analyse numérique, la méthode de Newton ou **la méthode de Newton-Raphson** est, dans son application la plus simple, un algorithme efficace pour trouver numériquement une approximation précise d'un zéro (ou racine) d'une fonction réelle d'une variable réelle. Cette méthode doit son nom aux mathématiciens anglais **Isaac Newton** (1643-1727) et **Joseph Raphson** (peut-être 1648-1715), qui furent les premiers à la décrire pour la recherche des zéros d'une équation polynomiale. On n'oubliera pas Thomas Simpson (1710-1761) qui élargit considérablement le domaine d'application de l'algorithme en montrant, grâce à la notion de dérivée, comment on pouvait l'utiliser pour calculer un zéro d'une équation non linéaire, pouvant ne pas être un polynôme.

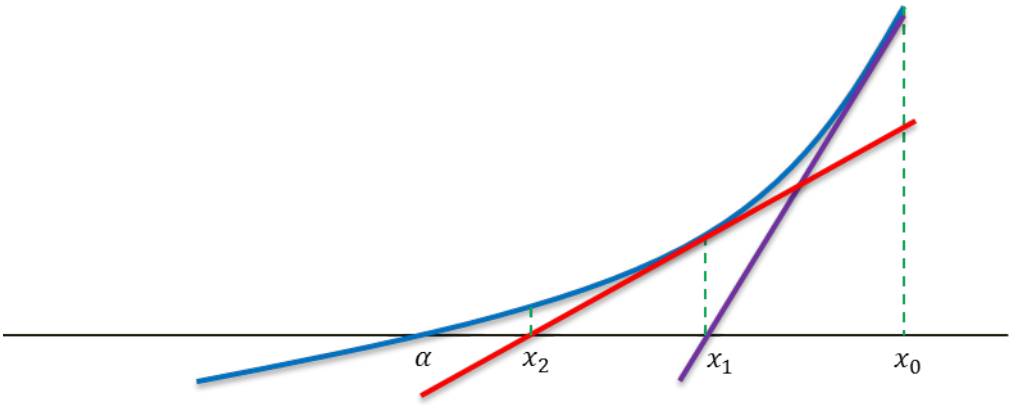
On va donc chercher à construire une bonne approximation d'un zéro de la fonction d'une variable réelle $f(x)$ en se basant sur son développement de Taylor au premier ordre. Pour cela, partant d'un point x_0 que l'on choisit de préférence proche du zéro à trouver. On approche la fonction au premier ordre, autrement dit, on la considère à peu près égale à sa tangente en ce point :

$$f(x) \approx f(x_0) + f'(x_0)(x - x_0)$$

Partant de là, pour trouver un zéro de cette fonction d'approximation, il suffit de calculer l'intersection de la droite tangente avec l'axe des abscisses, c'est-à-dire résoudre l'équation affine :

$$f(x_0) + f'(x_0)(x - x_0) = 0$$

On obtient alors un point x_1 qui, en général, a de bonnes chances d'être plus proche du vrai zéro de f que le point x_0 précédent. Par cette opération, on peut donc espérer améliorer l'approximation par itérations successives : on approche à nouveau la fonction par sa tangente en x_1 pour obtenir un nouveau point x_2, etc.



Cette méthode requiert que la fonction possède une tangente en chacun des points de la suite que l'on construit par itération, par exemple il suffit que f soit dérivable. Formellement, on part d'un point x_0 appartenant à l'ensemble de définition de la fonction et on construit par récurrence la suite :

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$

Où f' désigne la dérivée de la fonction f. Le point x_{k+1} est bien la solution de l'équation affine :

$$f(x_k) + f'(x_k)(x_{k+1} - x_k) = 0$$

Il se peut que la récurrence doive se terminer, si à l'étape (k, x_k), x_k n'appartient pas au domaine de définition ou si la dérivée $f'(x_k)$ est nulle ; dans ces cas, la méthode échoue.

Exemple

Pour illustrer la méthode, cherchons le nombre positif x vérifiant :

$$\cos(x) = x^3$$

On recherche le zéro positif (la racine) de $f(x) = cos(x) - x^3$.
La dérivation donne $f(x) = -\sin(x) - 3x^2$.

Comme pour tout x et $x^3 > 1$ pour $x > 1$, nous savons que notre zèro se situe entre 0 et 1. Nous essayons une valeur de dèpart de $x_0 = 0,5$.

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)} = 0,5 - \frac{cos(0,5) - 0,5^3}{-\sin(0,5) - 3(0,5)^2} \approx 1,1121416371$$

$$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)} \approx 0,909672693736$$

$$x_3 \approx 0,866263818209$$

$$x_4 \approx 0,865477135298$$

$$x_5 \approx 0,865474033111$$

$$x_6 \approx 0,865474033101$$

$$x_7 \approx 0,865474033102$$

Les 7 premiers chiffres de cette valeur coincident avec les 7 premiers chiffres du vrai zèro.

4.2 Convergence

La vitesse de convergence d'une suite x_n obtenue par la méthode de Newton peut être obtenue comme application de la formule de Taylor-Lagrange. Il s'agit d'évaluer une majoration de $\log|x_n - \alpha|$.

f est une fonction définie au voisinage de a et deux fois continument différentiable. On suppose que a se trouve être un zéro de f qu'on essaie d'approcher par la méthode de Newton. On fait l'hypothèse que a est un zéro d'ordre 1, autrement dit que $f'(a)$ est non nul. La formule de Taylor-Lagrange s'écrit :

$$f(\alpha) = f(x) + f'(x)(\alpha - x) + \frac{f''(\xi)}{2}(\alpha - x)^2 = 0$$

Avec : $x < \xi < \alpha$

Partant de l'approximation x, la méthode de Newton fournit au bout d'une itération :

$$N_f(x) - \alpha = x - \frac{f(x)}{f'(x)} - \alpha = \frac{f''(\xi)}{2f'(x)}(c - \alpha)^2$$

Pour un intervalle compact I contenant x et α inclus dans le domaine de définition de f, on pose : $m_1 = \min_{x\in I} |f'(x)|$ ainsi que $M_2 = \max_{x\in I} |f''(x)|$. Alors, pour tout $x \in I$:

$$\left|N_f(x) - \alpha\right| \leq \frac{M_2}{2m_1} |x - \alpha|^2$$

Par récurrence immédiate, on aura :

$$K |x_n - \alpha| \leq (K |x_0 - \alpha|)^{2^n}$$

Où $K = \frac{M_2}{2m_1}$.

En passant au logarithme :

$$\log |x_n - \alpha| \leq 2^n \log(K |x_0 - \alpha|) - \log(K)$$

La convergence de x_n vers α est donc quadratique, à condition que : $|x_0 - \alpha| < \frac{1}{K}$

Exemple : Racine carré

Un cas particulier de la méthode de Newton est l'algorithme de Babylone, aussi connu sous le nom de méthode de Héron : il s'agit, pour calculer la racine carrée de a, d'appliquer la méthode de Newton à la résolution de : $f(x) = x^2 - a$
On obtient alors, en utilisant la dérivée : $f'(x) = 2x$
La méthode d'approximation de la solution $\sqrt{a}$ est donnée par la formule itérative suivante :

$$x_{k+1} = x_k - \frac{x_k^2 - a}{2x_k} = \frac{1}{2}(x_k + \frac{a}{x_k})$$

Cette méthode converge pour tout $a \geq 0$ et tout point de départ $x_0 > 0$. On peut l'étendre au calcul de toute racine nième d'un nombre a avec la formule :

$$x_{k+1} = x_k - \frac{x_k^n - a}{nx_k^{n-1}} = x_k[1 + \frac{1}{n}(\frac{a}{x_k^n} - 1)]$$

La convergence de la suite $(x_n)_n$ se démontre par récurrence : pour k donné, on peut montrer que si : $0 \leq \sqrt{a} \leq x_k$ alors : $0 \leq \sqrt{a} \leq x_{k+1} \leq x_k$.
De plus, si : $0 < x_k \leq \sqrt{a}$, alors $\sqrt{a} \leq x_{k+1}$. La suite est donc décroissante au moins à partir du second terme. Elle est également bornée, donc elle converge. Reste à montrer que cette limite l est bien égale à $\sqrt{a}$: on obtient ce résultat en montrant qu'il est nécessaire que $l = \sqrt{a}$ pour que $x_{k+1} - l$ tende vers 0 lorsque k tend vers $+\infty$.

Remarques

La méthode de Newton permet de résoudre une équation $f(x) = 0$ dans $\mathbb{R}$. C'est une méthode locale (il faut donc au préalable localiser la racine) d'ordre deux (2).

4.3 Algorithme associé

```
newton := proc(f, u0, n)
local u, df, i;
u := evalf(u0);
df := D(f);
for i from 1 to n do
    u := u - f(u)/df(u);
end do;
return u;
end proc;
```

4.3.1 Exemples

Sur Maple :

$Digits := 20;$

$$20 \quad (2)$$

$f := x \to x^2 - 2;$

$$x \to x^2 - 2 \quad (3)$$

$newton(f, 2, 5);$

$$1.4142135623730950488 \quad (4)$$

$evalf(\mathrm{sqrt}(2));$

$$1.4142135623730950488 \quad (5)$$

$g := x \to \exp(-2 \cdot x) - x;$

$$x \to e^{-2x} - x \quad (6)$$

$newton(g, 0.2, 5);$

$$0.42630275100686274567 \quad (7)$$

$fsolve(g(x) = 0, x = 0..1);$

$$0.42630275100686274567 \quad (8)$$

$Digits := 50;$

$$50 \quad (9)$$

```
newton(f, 2, 1);
newton(f, 2, 2);
newton(f, 2, 3);
newton(f, 2, 4);
newton(f, 2, 5);
newton(f, 2, 6);
evalf(sqrt(2));
```

```
1.5000000000000000000000000000000000000000000000000
1.4166666666666666666666666666666666666666666666667
1.4142156862745098039215686274509803921568627450981
1.4142135623746899106262955788901349101165596221157
1.4142135623730950488016896235025302436149819257762
1.4142135623730950488016887242096980785696718753772
1.4142135623730950488016887242096980785696718753769
```

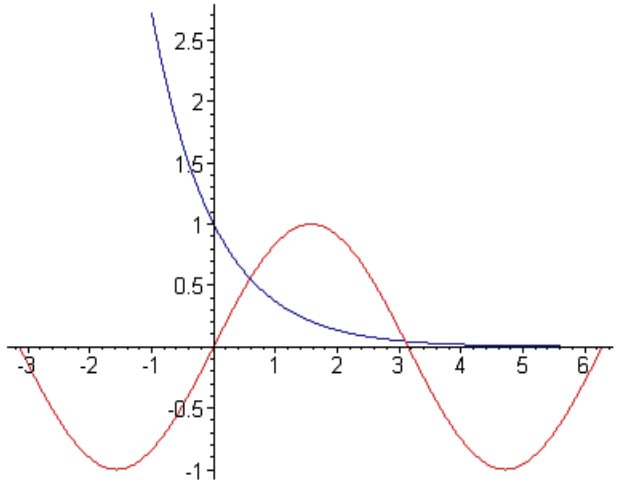
(10)

Sur Matlab :

Appliquons cette formule de récurrence à la résolution de l'équation .

$$\sin(x) = e^{(-x)}$$

```
> Courbe_1:=plot([x,sin(x),x=-Pi..2*Pi],color=orange):
Courbe_2:=plot([x,exp(-x),x=-1..2*Pi],color=navy):
display([Courbe_1,Courbe_2],xtickmarks=6):
```



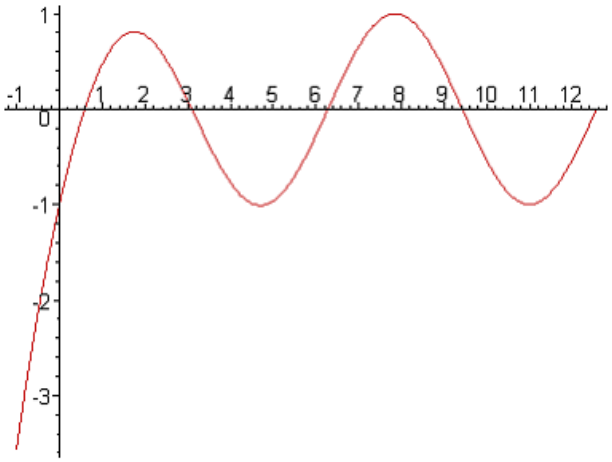
>

Puisque les fonctions $x \to \sin(x)$ et $x \to e^{(-x)}$ sont continues sur tous les réels **R** , il est graphiquement évident que l'équation

$$\sin(x) = e^{(-x)}$$

possède une infinité de racines et donc que la fonction f définie par $f(x) = \sin(x) - e^{(-x)}$ possède une infinité de zéros. Obtenons le tracé de la fonction f sur l'intervalle $[-1, 4\pi]$.

```
> f:=x->sin(x)-exp(-x):
plot([x,f(x),x=-1..4*Pi],color=orange,xtickmarks=10);
```



>

Estimons le zéro de la fonction f dans l'intervalle [0,1]. Afin d'appliquer la formule de Newton-Raphson plus efficacement, créons une fonction "newton" définie par

$$newton(x) = x - \frac{f(x)}{f'(x)}$$

De cette manière, il appert que $x_{k+1} = newton(x_k)$.

> **newton:=x->evalf(x - f(x)/D(f)(x));**

$$newton := x \rightarrow \text{evalf}\left(x - \frac{f(x)}{D(f)(x)}\right)$$

>

Nous avons imbriqué le calcul dans la macro-commande **evalf** afin que l'image soit un nombre décimal.

Choisissons maintenant l'amorce $x_0 = 1$ et appliquons successivement la fonction newton à chaque résultat.

> **x[0]:=1**

$$x_0 := 1$$

> **x[1]:=newton(x[0]);**

```
> n:=5;
x[0]:=1;
for k from 0 to n do
x[k+1]:=newton(x[k])
od;
n:='n':
```

$$n := 5$$

$$x_0 := 1$$

$$x_1 := .4785277891$$

$$x_2 := .5841570194$$

$$x_3 := .5885251122$$

$$x_4 := .5885327439$$

$$x_5 := .5885327440$$

$$x_6 := .5885327439$$

>

Montrons que le nombre $x_4 = 0{,}5885327439$ est une bonne approximation de ce zéro en évaluant f(x_4).

```
> 'f'(x[4])=f(x[4]);
```

$$f(.5885327439) = -.1\ 10^{-9}$$

Hum! pas si mal... Pour obtenir une meilleur approximation de ce zéro (meilleur dans le sens que $f(x_{k+1})$ soit davantage près de zéro), il suffit d'augmenter la valeur de la variable d'environnement Digits et d'augmenter probablement le nombres d'itérations. En initialisant la variable d'environnement **Digits** avec une valeur plus grande que 10 et en augmentant le nombre d'itérations, nous allons, au fil des approximations successives, établir un plus grand nombre de chiffres exactes du zéro réel de la fonction f.

```
> Digits:=40:
n:=7:

> for k from 0 to n do
x[k+1]:=newton(x[k])
od;
n:='n':
```

$$x_1 := .47852778898031161193902872985009416739 94$$

$$x_2 := .58415701941147088183523486724787545003 31$$

$$x_3 := .58852511220739121884299938218489615470 22$$

$$x_4 := .58853274395854760220351711304796044634 04$$

$$x_5 := .58853274398186107743223448863459261098 64$$

$$x_6 := .58853274398186107743245204570290368853 13$$

$$x_7 := .58853274398186107743245204570290368853 12$$

$$x_8 := .58853274398186107743245204570290368853 13$$

>

Lorsque la variable **Digits** est initialisé à 40, les nombres x_6 et x_7 ne diffèrent qu'à partir de la 40^e décimale, on a donc obtenu 39 chiffres décimaux exacts du zéro de la fonction f..

```
> 'f'(x[6])=f(x[6]);
```

$$f(.58853274398186107743245204570290368853 13) = .1\ 10^{-39}$$

$$x_1 := .4785277891$$

> **x[2]:=newton(x[1]);**

$$x_2 := .5841570194$$

> **x[3]:=newton(x[2]);**

$$x_3 := .5885251122$$

> **x[4]:=newton(x[3]);**

$$x_4 := .5885327439$$

> **x[5]:=newton(x[4]);**

$$x_5 := .5885327440$$

> **x[6]:=newton(x[5]);**

$$x_6 := .5885327439$$

On remarque que les nombres x_4 et x_5 ne diffèrent qu'à partir de la 9^e décimale. Cela nous montre qu'effectivement les différentes valeurs x_k convergent vers un nombre particulier compris dans l'intervalle [0,1]. Et nous en connaissons donc exactement les huit premières décimales de ce nombre.

Au lieu d'exécuter ces dernières requêtes manuellement, automatisons l'exécution de ces requêtes en les incluant dans une boucle **FOR** .

Chapitre 5

Méthode de la sécante

La méthode de la sécante est une méthode dérivée de celle de Newton où l'on remplace $f'(x_n)$ par $f(x_n) - f(x_{n-1})/(x_n - x_{n-1})$. on obtient la relation de récurrence :

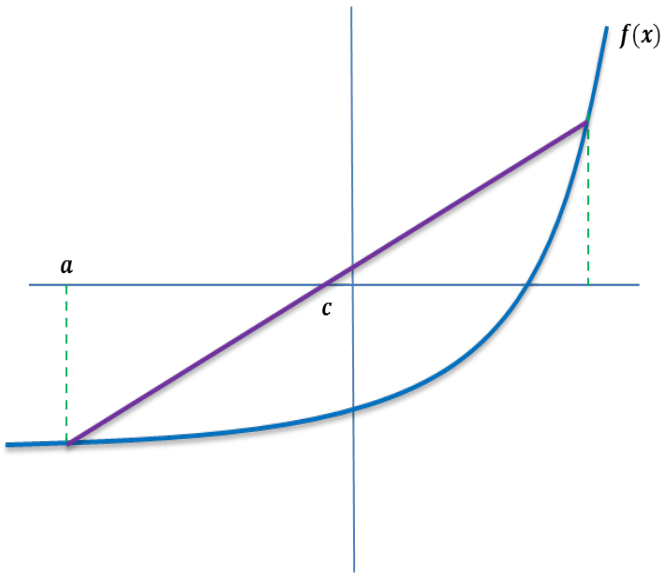
$$x_{n+1} = x_n - \frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})} f(x_n)$$

L'initialisation nécessite deux points x_0 et x_1, proches, si possible, de la solution recherchée. Il n'est pas nécessaire, contrairement à la méthode de la fausse position, que x_0 et x_1, encadre une racine de $f(x)$. La méthode de la sécante peut aussi être vue comme une variante de la méthode de la fausse position, ou les deux valeurs x_n et x_{n+1}, utiles à l'algorithme ne sont pas choisis pour encadrer la racine, mais sont simplement les dernières calculées.

Démonstration

Etant donnés a et b, on construit la droite passant par $(a, f(a))$ et $(b, f(b))$. Son équation est :

$$y - f(b) = \frac{f(b) - f(a)}{b - a}(x - b)$$



On choisit c égal à l'abscisse du point d'ordonnée nulle de cette droite.

$$f(b) = \frac{f(b) - f(a)}{b - a}(c - b) = 0$$

Si on extrait c de cette équation, on retrouve la relation de récurrence citée en haut :

$$c = b - \frac{b - a}{f(b) - f(a)} f(b)$$

Avec : $c = x_{n+1}$, $b = x_n$, $a = x_{n-1}$

5.1 Convergence

Si les valeurs initiales x_0 et x_1 sont suffisamment proches de la solution, la méthode aura un ordre de convergence de

$$\varphi = \frac{1 + \sqrt{5}}{2} = 1,618$$

qui est **le nombre d'or**.
Toutefois, la fonction f doit être deux fois continuement différentiable et la solution doit être une racine simple de f.

Chapitre 6

Méthode d'interpolation quadratique

Comme nous venons de le voir, la plupart des algorithmes nécessitent la résolution du problème intermédiaire « minimum d'une fonction f ». Il existe de nombreuses méthodes de recherche linéaire pour le résoudre. Cette étape constitue souvent la partie la plus coûteuse en temps de calcul, car elle implique une évaluation de la fonction f à chaque itération. Les méthodes de Newton ou de la sécante, ne sont que rarement utilisées. Les méthodes les plus répandues sont la méthode de la section d'or et l'interpolation polynomiale. Nous nous intéresserons ici à cette dernière, et plus particulièrement à la méthode d'interpolation parabolique.

On suppose ici que f est une fonction qui peut ne pas être dérivable ou bien dont on ne connaît pas la dérivée.

La méthode part du principe suivant : on choisit d'abord x_1, x_2 et x_3 tels que : $0 < x_1 < x_2 < x_3$ et $f(x_2) < f(x_1)$ et $f(x_2) < f(x_3)$.
La méthode de l'interpolation parabolique (ou quadratique) consiste à choisir le quatrième point x_4 comme le minimum du polynôme de degré 2 qui passe par les points $(x_i, f(x_i))$ avec $i = 1, 2, 3$.

On note q le polynôme interpolateur, on a d'apès Lagrange :

$$L_j(x) = \frac{\prod_{i=1}^{3}(x - x_i)}{\prod_{i=1}^{3}(x_j - x_i)}$$

avec $j = 1, 2, 3$.

Pour j=1 :

$$L_1(x) = \frac{(x - x_2)(x - x_3)}{(x_1 - x_2)(x_1 - x_3)}$$

Pour j=2 :

$$L_2(x) = \frac{(x - x_1)(x - x_3)}{(x_2 - x_1)(x_2 - x_3)}$$

Pour j=3 :

$$L_3(x) = \frac{(x - x_1)(x - x_2)}{(x_3 - x_1)(x_3 - x_2)}$$

Donc le polynôme interpolateur de f aux points : x_1, x_2 et x_3 est donné par :

$$q(x) = \sum_{i=1}^{3} f(x_i) L_i(x)$$

Donc :

$$q(x) = f(x_1)\frac{(x - x_2)(x - x_3)}{(x_1 - x_2)(x_1 - x_3)} + f(x_2).\frac{(x - x_1)(x - x_3)}{(x_2 - x_1)(x_2 - x_3)} + f(x_3).\frac{(x - x_1)(x - x_2)}{(x_3 - x_1)(x_3 - x_2)}$$

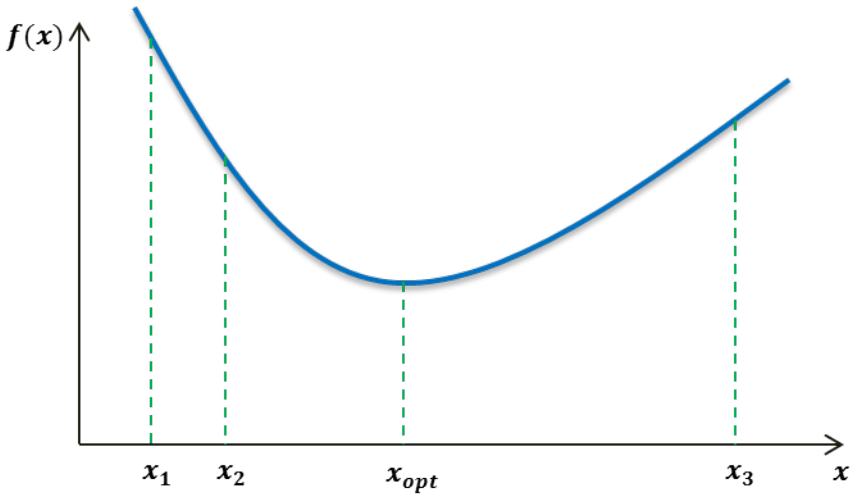
Le minimum de q est atteint sur $[x_1, x_3]$ en un point dont l'abscisse s'exprime facilement en fonction de x_1, x_2 et x_3, $f(x_1)$, $f(x_2)$ et $f(x_3)$. On note x_4 ce point ($q'(x_4) = 0$ nous donne la valeur du minimum) :

$$x_4 = \frac{\frac{(x_2+x_3)f(x_1)}{(x_1-x_2)(x_1-x_3)} + \frac{(x_1+x_3)f(x_2)}{(x_2-x_1)(x_2-x_3)} + \frac{(x_1+x_2)f(x_3)}{(x_3-x_1)(x_3-x_2)}}{2 \times (\frac{f(x_1)}{(x_1-x_2)(x_1-x_3)} + \frac{f(x_2)}{(x_2-x_1)(x_2-x_3)} + \frac{f(x_3)}{(x_3-x_1)(x_3-x_2)})}$$

- $f(x_4) \leq f(x_2)$, alors :
  - Si $x_4 \leq x_2$, le nouveau triplet est (x_1, x_4, x_2)
  - Sinon le nouveau triplet est (x_2, x_4, x_3).

- Si $f(x_4) > f(x_2)$, alors :
  - si $x_4 < x_2$,le nouveau triplet est (x_4, x_2, x_3)
  - Sinon, le nouveau triplet est (x_1, x_2, x_4).

Les trois points qui encadrent le minimum parmi les quatres disponibles sont conservés et le processus est recommencé. Cette méthode, qui a une vitesse de convergence super linéaire (On peut montrer que, si f est assez régulière, la convergence est superlinéaire) nécessite toutefois deux évaluations supplémentaires avant d'être effective ($f(0)$ étant connu initialement).



Chapitre 7

Application 1 : Enérgie rayonnante d'un corps noir

L'énergie rayonnante d'un corps noir dans l'intervalle d'émission $[\lambda, \lambda + d\lambda]$ par unité de surface est donnée par la loi de Planck :

$$M(\lambda) = \frac{2\pi h C_0^2}{n^2 \lambda^5} . \frac{1}{exp(\frac{hC_0}{nkT\lambda}) - 1} (1)$$

Où :

- $C_0 \approx 2,997.10^8 m.s^{-1}$: vitesse de la lumière dans le vide.
- $h \approx 6,625.10^{-34} J.s$: constante de Planck.
- $k = 1,380.10^{-23} J.K^{-1}$: constante de Bolzmann.
- λ= longueur d'onde (m).
- T : température absolue du corps noir (K).
- $n = 1$: indice de refraction du milieu (ici le vide).

1. **Tracer du graphe de la fonction** $M(\lambda)$ pour $\lambda \in [10^{-7}, 2.10^{-5}]$, pour les valeurs de $T \in \{300, 350, 400, 450, 500, 550, 600, 650, 700, 750, 800\}$

En évaluant la fonction $M(\lambda)$ sur Maple, on ontient le graphe suivant :

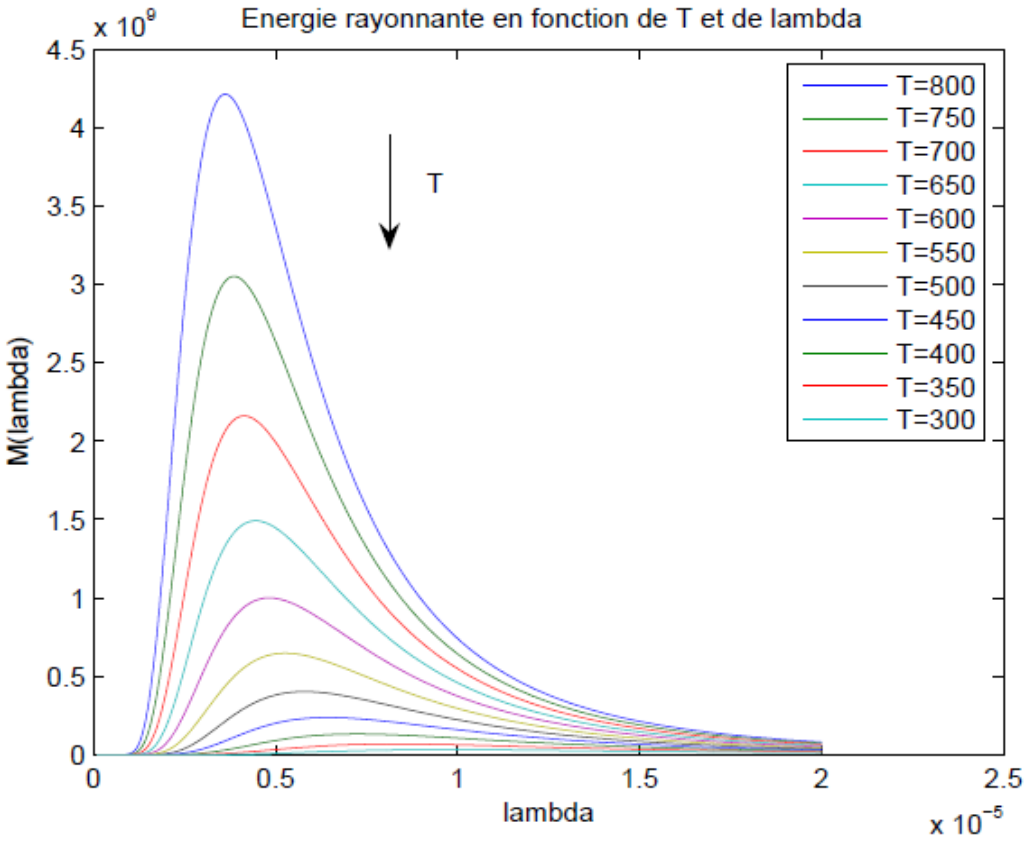


FIGURE 1 – Représentation de $M(\lambda)$

2. On souhaite déterminer la valeur λ^* de λ qui maximise la fonction M(λ) (qui minimise donc M(λ)) en utilisant la méthode de la section dorée, pour une température T donnée ; dans ce cas :
D'après la question précédente, λ^* est dans l'intervalle $[\lambda_1, \mu_1] = [10^{-7}, 2.10^{-5}]$; on peut donc démarer l'algorithme avec cet intervalle avec le taux de réduction $\varphi = \tau = \frac{1+\sqrt{5}}{2}$:

Algorithme de la section dorée :

Pour $i = 1, 2, ...$

$\lambda^{'} = \lambda_i + \frac{1}{\tau^2}(\mu_i - \lambda_i)$

$\mu^{'} = \lambda_i + \frac{1}{\tau}(\mu_i - \lambda_i)$

Si $M(\lambda^{'}) > M(\mu^{'})$, *alors* $\lambda_{i+1} = \lambda_i$ *et* $\mu_{i+1} = \mu^{'}$

Sinon, $\lambda_{i+1} = \lambda^{'}$ *et* $\mu_{i+1} = \mu_i$

Fin Si

Fin pour.

Il faut donc trouver un critère d'arrêt pour λ_i : lorsque λ_i est assez proche de μ_i ;

Fonction *Matlab* de la section dorée :

```
function [Lambda_etoile]=sectionDoree(T,lambda1,mu1,N,tol);

%-------------------------------------------------------
% Entrées :
% T : température du corps noir
% lambda1,mu1 : intervalle de départ
% N : on fixe un nombre limité d'itérations
% tol : test sur la distance entre lambda(i) et mu(i)
%-------------------------------------------------------
tau=(1+sqrt(5))/2;
lambda(1)=lambda1;
mu(1)=mu1;
test =mu(1)-lambda(1)
%for i=1:N-1
Lambda(1)=(lambda(1)+mu(1))/2;
i=1;
%-------------------------------------------------------
while (test>tol && i<N-1)
    format long % pour augmenter la précision des calculs

    lambdaPrime=lambda(i)+(1/tau^2)*(mu(i)-lambda(i));
    muPrime=lambda(i)+(1/tau)*(mu(i)-lambda(i));

    if (1e-9)*M(T,lambdaPrime)>(1e-9)*M(T,muPrime)
        lambda(i+1)=lambda(i);
        mu(i+1)=muPrime;
    else
```

```
            lambda(i+1)=lambdaPrime;
            mu(i+1)=mu(i);
        end
        Lambda(i)=(lambda(i)+mu(i))/2;
        test =mu(i)-lambda(i);

        i=i+1;

    end

    Lambda_etoile=Lambda(i-1);
```

On obtient ainsi pour les différentes valeurs de T, avec $N = 10^3$ et $tol = 10^{-15}$

T	300	350
$\lambda * (.10^{-6})$	9, 659241719891250	8, 279350281142982

T	400	450
$\lambda * (.10^{-6})$	7, 244431352791985	6, 439494626802787

T	500	550
$\lambda * (.10^{-6})$	5, 795545127391454	5, 268677375507811

T	600	650
$\lambda * (.10^{-6})$	4, 829620981447535	4, 458111654187573

T	700	750
$\lambda * (.10^{-6})$	4, 139675087034757	3, 863696763249309

T	800
$\lambda * (.10^{-6})$	3, 622215743261860

3. Lois de Wien

La valeur λ^* qui maximise M(λ) vérifie $\forall$ T :

$$\frac{\partial M}{\partial \lambda}(\lambda^*) = 0$$

$$\Rightarrow \frac{2\pi h C_0^2}{n^2 \lambda^{*6}} . \frac{1}{exp(\frac{hC_0}{nkT\lambda^*}) - 1} . \left[-5 + \frac{hC_0}{nkT\lambda^*} . \frac{exp(\frac{hC_0}{nkT\lambda^*})}{exp(\frac{hC_0}{nkT\lambda^*}) - 1} \right] = 0$$

Posons $x = \frac{hC_0}{nkT\lambda^*}$, alors :

$$(x-5)e^x + 5 = 0, (x > 0, \Rightarrow x = x^* \approx 4,96511)$$

Donc :

$$\lambda^* T = \frac{hC_0}{nkx^*} = A(C^{te}) \approx 2898 \mu m.K (2)$$

Et $\frac{1}{e^x - 1} = -1 + \frac{5}{x} = -1 + \frac{5nkA}{hC_0}$, donc :

$$M(\lambda^*) = \frac{2\pi h C^0}{n^2 A^5} \left(-1 + \frac{5nkA}{hC_0} \right) T^5 = BT^5 (3)$$

,

$$(Bconstante, B \approx 2.115.10^{-9} Wb.m^{-2}.K^{-5})$$

Avec la formule fermée (2), on peut calculer explicitement λ^* pour les différentes valeurs de T :

T	300	350
$\lambda * (.10^{-6})$	9, 6592500730970	8, 2793572055117

T	400	450
$\lambda * (.10^{-6})$	7, 2444375548228	6, 4395000487314

T	500	550
$\lambda * (.10^{-6})$	5, 7955500438582	5, 2686818580529

T	600	650
$\lambda * (.10^{-6})$	4, 8296250365485	4, 4581154183525

T	700	750
$\lambda * (.10^{-6})$	4, 1396786027559	3, 8637000292388

T	800
$\lambda * (.10^{-6})$	3, 6222187774114

Matlab

Utilisation de la fonction Matlab : *fminsearch* :

```
function [lambda]=fminsearch_Matlab(T,lambda0);

%----------définition des constantes----------------

C0=2.997e+8; %vitesse de la lumière dans le vide
h=6.625e-34; %constante de Planck
k=1.380e-23; %constante de Bolzmann
%lambda :longueur d'onde
%T: température absolue du corps noir
n=1; %indice de refraction du milieu (ici le vide)
%---------------------------------------------------

fct=@(x)(-1).*(2*pi*h*C0^2./(n*n.*x.^5))./(exp(h*C0./(n*k*T.*x))-1);
[lambda,fval]=fminsearch(fct,lambda0);
```

Cependant, cette fonction est très sensible au point de départ λ_0, comme le montre le tableau ci-dessus :

λ_0	$\lambda*$
$0,3.10^{-5}$	$4,139675331115727.10^{-6}$
$0,4.10^{-5}$	$4,139674377441405.10^{-6}$
$0,5.10^{-5}$	$4,1396751403808661.10^{-6}$
$0,6.10^{-5}$	$4,139675331115722.10^{-6}$

Chapitre 8

Application 2 : Etude et optimisation du fonctionnement d'un système photovoltaique

I - Cellule photovoltaïque réel

Le model photovoltaïque précédent ne rendait pas compte de tous les phénomènes présents lors de la conversion d'énergie lumineuse. En effet, dans le cas réel, on observe une perte de tension en sortie ainsi que des courants de fuite. On modélise donc cette perte de tension par une résistance en série R_S et les courants de fuite par une résistance en paralléle R_P

1 - Cherchons I :

D'après la loi des noeuds :

- Considérons le 1^{er} noeud :

$$I_{ph} = I' + T_d$$
$$I' = I_{ph} - I_d(1)$$

- Considèrons le 2^{me} noeud :

$$I' = I_p + I(2)$$

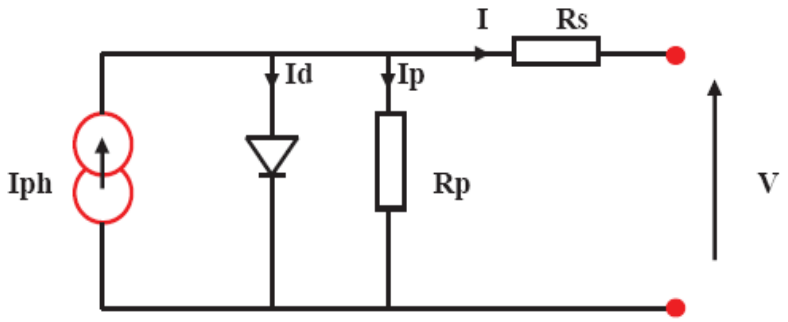


Figure 1.7 : Modèle de la cellule photovoltaïque réel

Donc de (1) et (2) :

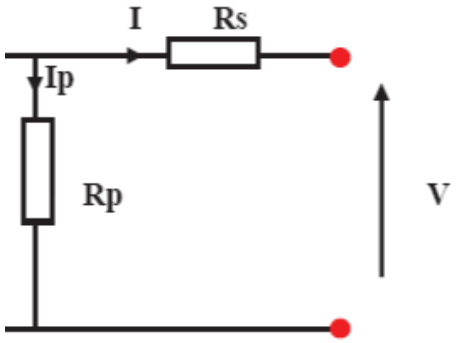
$$I_{ph} - I_d = I_p + I$$

Alors :

$$I = I_{ph} - I_d - I_p$$

2 - Cherchons I_p :

D’après la loi des mailles :



$$V = -R_s I + I_p R_p$$

$$I_p R_p = V + R_s I$$

D' où :

$$I_p = \frac{V + R_s I}{R_p}$$

3 - Cherchons I_d :

On a d'après Fermi-Dirac l'équation caractéristique suivante :

$$I_d = I_0 . \exp\left(\frac{V_d}{nV_T} - 1\right)$$

Avec $V_T = \frac{KT}{q}$

- V_D : Tension aux bornes de la diode (V).
- I_0 : Courant de saturation dépendant de la température.
- T : température absolue de la jonction (k).
- k : constante de Boltzmann $(1,38110^{-23} Joule/Kelvin)$.
- n : Constante empirique qui est égale à 1 pour le germanium et 2 pour le silicium; le facteur de qualité de diode.

$$I_d = I_0 . \exp\left(\frac{V_d}{nV_T} - 1\right) = I_0 . \exp\left(\frac{V + R_s I}{nV_T} - 1\right) = I_0 . \exp\left(\frac{V + R_s I}{nkT} - 1\right)$$

D'où :

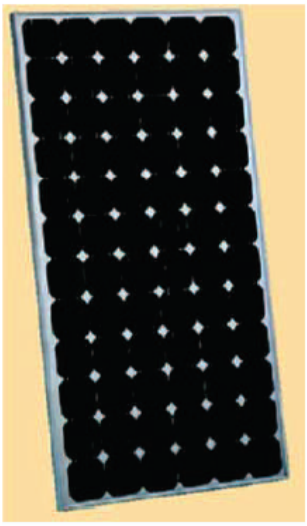
$$I_d = I_0 . exp\left(\frac{V_d}{nV_T} - 1\right)$$

Donc l'expression de I sera :

$$I = I_{ph} - I_0 . \exp\left(\frac{V_d}{nV_T} - 1\right) - \frac{V + R_s I}{R_p}$$

II - Module photovoltaique : Simens

Nous avons choisi une module Simens SP75 composé de 32 cellules en silicium monocristallin connectées en sérer ayant une puissance maximal de $75w$ est considéré dans les conditions standards $G = 1000w/m2$, $T = 25C$.



Puissance maximale	p_{max} : 75 wc
Tension a vide	V_{co} : 21.7 V
Courant de court circuit	I_{sc} : 4.8A
Tension au point de puissance maximale	V_{mpp} : 17 V
Courant au point de puissance maximale	I_{mpp}: 4.4A
Noct	45 ±2 °C
Coefficient de température :	
Tension à vide	φVco : -0.77v/°C
Courant court circuit	φIsc: 2.6 ma / °C

Tableau 1.2 : Caractéristique électrique d'un module photovoltaïque SP75 éventuelle

Le courant de saturation de la diode est donné par :

$$I_0 = \frac{I_{sc}}{\exp\left(q.\frac{V+IR_s}{nkT}\right)} - 1$$

Pour calculer I on considère $R_p = \infty$ donc $I_p = 0$.

$$R_p = \infty \Rightarrow \text{circuit ouvert} \Rightarrow I_p = 0$$

Dans ce cas l'équation de I devient :

$$I = I_{ph} - I_d$$

Dans notre cas :

$$I_d = I_{sc}$$

On a :

$$I_0 = \frac{I_{sc}}{\exp\left(q.\frac{V+IR_s}{nkT}\right)} - 1$$

D'où :

$$I_{sc} = I_0.\exp\left(q.\frac{V+IR_s}{nkT}\right) + 1$$

Donc :

$$I = I_{ph} - I_0.\exp\left(q.\frac{V+IR_s}{nkT}\right) - 1$$

La méthode choisie pour la simulation de ce modèle est la méthode de Newton Raphson qui est décrit comme suit :

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

La fonction de la cellule photovoltaique est donnée comme suit :

$$f(I) = I_{sc} - I - I_0.\exp\left(q \times \frac{V+IR_s}{nkT}\right) - 1$$

Donc :

$$I_{n+1} = I_n - \frac{I_{sc} - I_n - I_0.\left(\frac{V+I_nR_s}{nV_t}\right) - 1}{-1 - I_0.\left(\frac{R_s}{nV_t}.\exp\left(\frac{V+I_nR_s}{nV_t}\right)\right)}$$