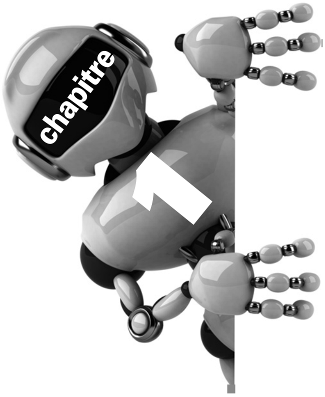

Chapitre 1. Tous les algorithmes de 1^{re} année

Chapitres concernés :

Fonctions, résolution approchée d'équations

Calcul intégral (approché)

Équations différentielles

Arithmétique

Polynômes

Matrices

Déterminants

Systèmes linéaires

Listes

Courbes paramétrées

Probabilités

Résolution d’une équation par dichotomie*

- **Chapitre concerné** : Fonctions, résolution approchée d’équation.
- **Ce que fait l’algorithme** : il résout l’équation $f(x)=0$ de manière approchée avec une certaine précision.
- **L’algorithme** (ici on veut résoudre $f(x)=0$ avec $f(x)=x^2-2$).

File Edit Format Run Options Window Help

```python
def f(x):
    return x**2-2
def dichotomie(f,a,b,précision):
    while b-a>précision:
        m=(a+b)/2
        if f(a)*f(m)<0:
            b=m
        else:
            a=m
    return b
```

- **Exemple** : On veut résoudre $x^2-2=0$ avec une précision au millième sur l’intervalle $[0;2]$.

Saisie :

```
>>> dichotomie(f,0,2,0.001)
```

Affichage :

```
1.4150390625
```

Cela signifie que l’équation $x^2-2=0$ a pour solution approchée 1,415 (au millième).

Résolution d'une équation par la méthode de la fausse position

- **Chapitre concerné** : Fonctions, résolution approchée d'équation.
- **Ce que fait l'algorithme** : il résout l'équation $f(x)=0$ de manière approchée avec une certaine précision, à l'aide de la suite (x_n) définie par $\begin{cases} x_{n+1}=x_0-\dfrac{x_n-x_0}{f(x_n)-f(x_0)}f(x_0) \\ x_0 \end{cases}$.
- **L'algorithme** (ici on veut résoudre $f(x)=0$ avec $f(x)=\frac{1}{4}x^2-x-1$ en partant de $x_0=3$).

```
def f(x):
    return 1/4*x**2-x-1
def Fausse_position(f,Précision):
    x_0=3
    x=x_0
    y=f(x_0)
    while abs(y-x)>Précision:
        x=y
        y=x_0-(y-x_0)/(f(y)-f(x_0))*f(x_0)
    return y
```

- **Exemple** : on veut résoudre $\frac{1}{4}x^2-x-1=0$ avec une précision au millième.

Saisie :

```
>>> Fausse_position(f,0.001)
```

Affichage :

```
4.828722998486295
```

Cela signifie que la valeur approchée (au millième) d'une solution de l'équation $\frac{1}{4}x^2-x-1=0$ est 4,829.

Résolution d'une équation par la méthode de la sécante

- **Chapitre concerné** : Fonctions, résolution approchée d'équation.
- **Ce que fait l'algorithme** : il résout l'équation $f(x)=0$ de manière approchée avec une certaine précision, à l'aide de la suite (x_n) définie par $\begin{cases} x_{n+1}=x_n-\dfrac{b-x_n}{f(b)-f(x_n)}f(x_n) \\ x_0=a \end{cases}$ (où a et b sont tels que $f(a)<0$ et $f(b)>0$) .
- **L'algorithme** (ici on veut résoudre $f(x)=0$ avec $f(x)=\frac{1}{4}x^2-x-1$, $a=3$ [car $f(3)=-1{,}75$] et $b=5$ [car $f(5)=0{,}25$]).

```
File  Edit  Format  Run  Options  Window  Help
def f(x):
    return 1/4*x**2-x-1
def Sécante(f,Précision):
    x=3
    y=x-(5-x)/(f(5)-f(x))*f(x)
    while abs(y-x)>Précision:
        x=y
        y=y-(5-y)/(f(5)-f(y))*f(y)
    return y
```

- **Exemple** : on veut résoudre $\frac{1}{4}x^2-x-1=0$ avec une précision au millième.

Saisie :

```
>>> Sécante(f,0.001)
```

Affichage :

```
4.82842509603073
```

Cela signifie que la valeur approchée (au millième) d'une solution de l'équation $\frac{1}{4}x^2-x-1=0$ est 4,828.

Résolution d'une équation par la méthode de Newton*

- **Chapitre concerné** : Fonctions, résolution approchée d'équation.
- **Ce que fait l'algorithme** : il résout l'équation $f(x)=0$ de manière approchée avec une certaine précision, à l'aide de la suite (x_n) définie par $\begin{cases} x_{n+1}=x_n-\dfrac{f(x_n)}{f'(x_n)} \\ x_0 \end{cases}$.
- **L'algorithme** : (ici on veut résoudre $f(x)=0$ avec $f(x)=\frac{1}{4}x^2-x-1$

  $[f'(x)=\frac{1}{2}x-1]$ en partant de $x_0=3$).

```
def f(x):
    return 1/4*x**2-x-1
def df(x):
    return 1/2*x-1
def Newton(f,df,Précision):
    x=3
    y=x-f(x)/df(x)
    while abs(y-x)>Précision:
        x=y
        y=y-f(y)/df(y)
    return y
```

- **Exemple** : on veut résoudre $\frac{1}{4}x^2-x-1=0$ avec une précision au millième.

  Saisie :

```
>>> Newton(f,df,0.001)
```

  Affichage :

```
4.828427125049864
```

  Cela signifie que la valeur approchée (au millième) d'une solution de l'équation $\frac{1}{4}x^2-x-1=0$ est 4,828.

Calcul (approché) d'une intégrale par la méthode des rectangles*

- **Chapitre concerné** : Calcul intégral (approché).
- **Ce que fait l'algorithme** : il détermine une valeur approchée de $\int_a^b f(x)dx$ à l'aide de la formule $\int_a^b f(x)dx \simeq \sum_{i=0}^{i=N-1} f(x_i)h$ où $h = \frac{b-a}{N}$ et $x_i = a + ih$ (plus N est grand, meilleure est la précision).
- **L'algorithme** : (ici on veut une valeur approchée de $\int_a^b x^2dx$)

```
File Edit Format Run Options Window Help
def f(x):
    return x**2
def Rectangles(f,a,b,N):
    Pas=(b-a)/N
    Intégrale=0
    for i in range(1,N+1):
        Intégrale=Intégrale+f(a)*Pas
        a=a+Pas
    return Intégrale
```

- **Exemple** : On souhaite une valeur approchée de $\int_0^1 x^2dx$.

Saisie :

```
>>> Rectangles(f,0,1,1000)
```

Affichage :

```
0.33283350000000095
```

Cela signifie que $\int_0^1 x^2dx = \frac{1}{3}$.

Calcul (approché) d'une intégrale par la méthode des trapèzes*

- **Chapitre concerné** : Calcul intégral (approché).
- **Ce que fait l'algorithme** : il détermine une valeur approchée de $\int_a^b f(x)dx$ à l'aide de la formule $\int_a^b f(x)dx \simeq \sum_{i=0}^{i=N-1} \frac{f(x_i)+f(x_i+h)}{2}h$ où $h=\frac{b-a}{N}$ et $x_i = a+ih$ (plus N est grand, meilleure est la précision).
- **L'algorithme** : (ici on veut une valeur approchée de $\int_a^b x^2dx$)

File Edit Format Run Options Window Help

```
def f(x):
    return x**2
def Trapèzes(f,a,b,N):
    Pas=(b-a)/N
    Intégrale=0
    for i in range(1,N+1):
        Intégrale=Intégrale+(f(a)+f(a+Pas))/2*Pas
        a=a+Pas
    return Intégrale
```

- **Exemple** : On souhaite une valeur approchée de $\int_0^1 x^2dx$.

Saisie :

```
>>> Trapèzes(f,0,1,1000)
```

Affichage :

```
0.33333350000000034
```

Cela signifie que $\int_0^1 x^2dx = \frac{1}{3}$.

Calcul (approché) d'une intégrale par la méthode de Simpson

- **Chapitre concerné** : Calcul intégral (approché).
- **Ce que fait l'algorithme** : il détermine une valeur approchée de $\int_a^b f(x)dx$ à l'aide de la formule $\int_a^b f(x)dx \simeq \frac{h}{3}\left[f(x_0)+4\sum_{i=0}^{i=N-1} f(x_{2i+1})+2\sum_{i=1}^{i=N-1} f(x_{2i})+f(x_{2N})\right]$ où $h=\frac{b-a}{2N}$ et $x_i=a+ih$ (plus N est grand, meilleure est la précision).
- **L'algorithme** : (ici on veut une valeur approchée de $\int_a^b x^2dx$)

File Edit Format Run Options Window Help

```
def f(x):
    return x**2
def Simpson(f,a,b,N):
    Pas=(b-a)/(2*N)
    Intégrale=0
    N1=2*N+1
    N2=N1-1
    Somme=0
    for i in range(1,N1,2):
        X=a+i*Pas
        Somme=Somme+f(X)
    Intégrale=Somme*4
    Somme=0
    for i in range(2,N2,2):
        X=a+i*Pas
        Somme=Somme+f(X)
    Intégrale=Intégrale+2*Somme+f(a)+f(b)
    Intégrale=Intégrale*Pas/3
    return Intégrale
```

- **Exemple** : On souhaite une valeur approchée de $\int_0^1 x^2dx$.

Saisie :

```
>>> Simpson(f,0,1,5)
```

Affichage :

```
0.3333333333333333
```

Cela signifie que $\int_0^1 x^2dx=\frac{1}{3}$.