Résolution d'équation $f(x) = 0$

Informatique pour tous

Équation $f(x) = 0$

Étant donné une fonction $f : [a, b] \longrightarrow \mathbb{R}$, on veut approximer une solution x_0 de l'équation:

$$f(x) = 0$$

Équation $f(x) = 0$

Étant donné une fonction $f : [a, b] \longrightarrow \mathbb{R}$, on veut approximer une solution x_0 de l'équation:

$$f(x) = 0$$

Par exemple, si $f : x \longmapsto x^2 - 2$, cela revient à approximer une solution de $x^2 = 2$, c'est à dire de $\sqrt{2}$ ou $-\sqrt{2}$.

Équation $f(x) = 0$

Étant donné une fonction $f : [a, b] \longrightarrow \mathbb{R}$, on veut approximer une solution x_0 de l'équation:

$$f(x) = 0$$

Par exemple, si $f : x \longmapsto x^2 - 2$, cela revient à approximer une solution de $x^2 = 2$, c'est à dire de $\sqrt{2}$ ou $-\sqrt{2}$.

Nous allons voir deux méthodes qui approximent x_0 par une suite $(u_n)_n$ telle que $\lim_{n \to \infty} u_n = x_0$.

Méthode par dichotomie

La **méthode par dichotomie** suppose que:

1. f est **continue**
2. les signes de $f(a)$ et $f(b)$ sont différents:

Méthode par dichotomie

La **méthode par dichotomie** suppose que:

1. f est **continue**
2. les signes de $f(a)$ et $f(b)$ sont différents: $\boxed{f(a)f(b) < 0}$

Méthode par dichotomie

La **méthode par dichotomie** suppose que:

1. f est **continue**
2. les signes de $f(a)$ et $f(b)$ sont différents: $\boxed{f(a)f(b) < 0}$

Par le théorème des valeurs intermédiaires, f s'annule entre a et b.

On remplace alors soit a soit b par $\frac{a+b}{2}$ de façon à ce que $f(a)$ et $f(b)$ soient toujours de signes différents.

Méthode par dichotomie

La **méthode par dichotomie** suppose que:

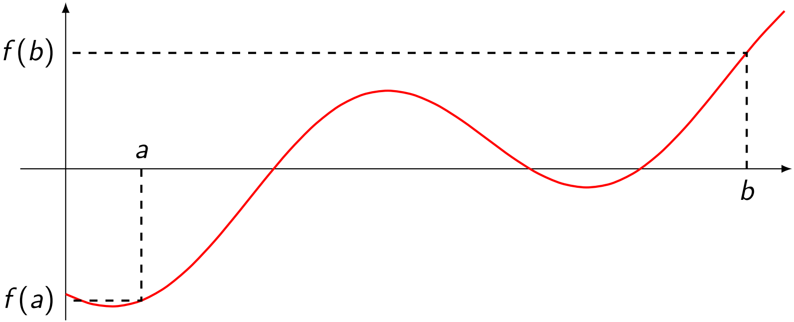
1. f est **continue**
2. les signes de $f(a)$ et $f(b)$ sont différents: $\boxed{f(a)f(b) < 0}$

Par le théorème des valeurs intermédiaires, f s'annule entre a et b.

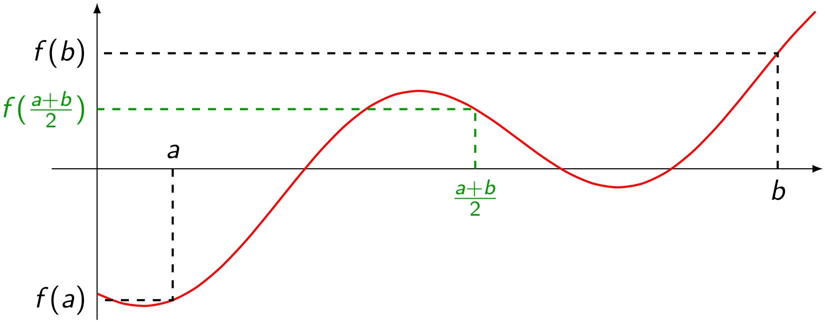
On remplace alors soit a soit b par $\frac{a+b}{2}$ de façon à ce que $f(a)$ et $f(b)$ soient toujours de signes différents.

Quand la longueur de $[a, b]$ devient inférieur à ϵ, on obtient un encadrement d'un zéro de f à ϵ près.

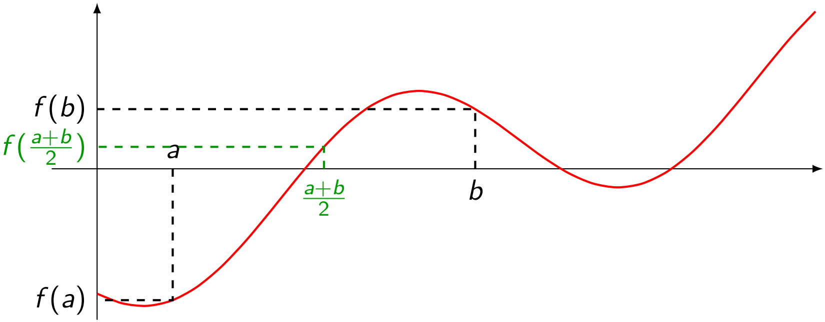
Méthode par dichotomie



Méthode par dichotomie



Méthode par dichotomie



Méthode par dichotomie

La fonction suivante suppose que `f` continue et `f(a)f(b) < 0`:

```
def dichotomie(f, a, b, epsilon):
    while abs(b - a) >= epsilon:
        if f(a) * f((a + b)/2) < 0:
            b = (a + b)/2
        else:
            a = (a + b)/2
    return [a, b]
```

Méthode par dichotomie

La fonction suivante suppose que `f` continue et `f(a)f(b) < 0`:

```
def dichotomie(f, a, b, epsilon):
    while abs(b - a) >= epsilon:
        if f(a) * f((a + b)/2) < 0:
            b = (a + b)/2
        else:
            a = (a + b)/2
    return [a, b]
```

Alors `f(a)f(b) < 0` est un invariant de boucle et les valeurs renvoyées encadrent un zéro de `f` à `epsilon` près.

Exemple

```python
def f(x):
    return x**2 - 2
```

Exemple

```python
def f(x):
    return x**2 - 2
```

```
In [8]: dichotomie(f, 1, 2, 0.1)
Out[8]: [1.375, 1.4375]

In [9]: dichotomie(f, 1, 2, 0.01)
Out[9]: [1.4140625, 1.421875]

In [10]: dichotomie(f, 1, 2, 0.001)
Out[10]: [1.4140625, 1.4150390625]

In [11]: 2**0.5
Out[11]: 1.4142135623730951
```

Méthode par dichotomie

Question

Pourquoi obtient t-on une boucle infinie si on essaie d'exécuter `dichotomie(f, 1, 2, 1e-17)`?

Méthode par dichotomie

Question

Pourquoi obtient t-on une boucle infinie si on essaie d'exécuter `dichotomie(f, 1, 2, 1e-17)`?

A cause du nombre de chiffres après la virgule limité à 52 bits des flottants!

Complexité de dichotomie

```python
def dichotomie(f, a, b, epsilon):
    while abs(b - a) >= epsilon:
        if f(a) * f((a + b)/2) < 0:
            b = (a + b)/2
        else:
            a = (a + b)/2
    return [a, b]
```

Question

Quelle est la complexité dans le pire des cas de `dichotomie`?

Complexité de dichotomie

Soient a_n, b_n les valeurs de a, b après n itérations.
Alors:

$$b_n - a_n \leq \frac{b_{n-1} - a_{n-1}}{2} \leq ... \leq \frac{b_0 - a_0}{2^n}$$

Complexité de dichotomie

Soient a_n, b_n les valeurs de a, b après n itérations.
Alors:

$$b_n - a_n \leq \frac{b_{n-1} - a_{n-1}}{2} \leq ... \leq \frac{b_0 - a_0}{2^n}$$

Si on souhaite obtenir un zéro de f à ε près, il suffit que:

$$\frac{b_0 - a_0}{2^n} \leq \varepsilon$$

Complexité de dichotomie

Soient a_n, b_n les valeurs de a, b après n itérations.
Alors:

$$b_n - a_n \leq \frac{b_{n-1} - a_{n-1}}{2} \leq ... \leq \frac{b_0 - a_0}{2^n}$$

Si on souhaite obtenir un zéro de f à ε près, il suffit que:

$$\frac{b_0 - a_0}{2^n} \leq \varepsilon$$

$$\Leftrightarrow \quad \frac{b_0 - a_0}{\varepsilon} \leq 2^n$$

Complexité de dichotomie

Soient a_n, b_n les valeurs de a, b après n itérations.
Alors:

$$b_n - a_n \leq \frac{b_{n-1} - a_{n-1}}{2} \leq ... \leq \frac{b_0 - a_0}{2^n}$$

Si on souhaite obtenir un zéro de f à ε près, il suffit que:

$$\frac{b_0 - a_0}{2^n} \leq \varepsilon$$

$$\Leftrightarrow \quad \frac{b_0 - a_0}{\varepsilon} \leq 2^n$$

$$\Leftrightarrow \quad \boxed{n \geq \log_2\left(\frac{b_0 - a_0}{\varepsilon}\right)}$$

Complexité de dichotomie

```python
def dichotomie(f, a, b, epsilon):
    while abs(b - a) >= epsilon:
        if f(a) * f((a + b)/2) < 0:
            b = (a + b)/2
        else:
            a = (a + b)/2
    return [a, b]
```

En supposant que f a complexité $O(1)$, la complexité dans le pire des cas de `dichotomie(f, a, b, ε)` est donc:

$$\boxed{O(\log_2\left(\frac{b-a}{\varepsilon}\right))}$$

Complexité de dichotomie

En supposant que f a complexité $O(1)$, la complexité dans le pire des cas de `dicho(f, a, b,` ε`)` est donc:

$$\boxed{O(\log_2\left(\frac{b-a}{\varepsilon}\right))}$$

Si on veut p bits significatifs d'un zéro de f, on choisit $\varepsilon = \frac{1}{2^p}$ et le nombre d'itérations est de l'ordre de p.

Méthode de Newton

La **méthode de Newton** trouve un zéro d'une fonction f plus rapidement, mais nécessite des conditions supplémentaires pour converger.

1ère condition: f doit être **dérivable**

Méthode de Newton

La **méthode de Newton** trouve un zéro d'une fonction f plus rapidement, mais nécessite des conditions supplémentaires pour converger.

1ère condition: f doit être **dérivable** (donc continue)

Méthode de Newton

Idée: soit $u_n \in \mathbb{R}$.
Alors, pour x proche de u_n:

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

Méthode de Newton

Idée: soit $u_n \in \mathbb{R}$.
Alors, pour x proche de u_n:

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

Donc, si u_{n+1} est choisi tel que $f(u_n) + (u_{n+1} - u_n)f'(u_n) = 0$, on peut espérer que $f(u_{n+1}) \approx 0$.

Méthode de Newton

Idée: soit $u_n \in \mathbb{R}$.
Alors, pour x proche de u_n:

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

Donc, si u_{n+1} est choisi tel que $f(u_n) + (u_{n+1} - u_n)f'(u_n) = 0$, on peut espérer que $f(u_{n+1}) \approx 0$.
Si $f'(u_n) \neq 0$:

$$\boxed{u_{n+1} = u_n - \frac{f(u_n)}{f'(u_n)}}$$

Méthode de Newton

Méthode:

1. On choisit u_0 quelconque, si possible proche d'un zéro de f.

Méthode de Newton

Méthode:

1. On choisit u_0 quelconque, si possible proche d'un zéro de f.
2. On définit:
$$u_{n+1} = u_n - \frac{f(u_n)}{f'(u_n)}$$

Méthode de Newton

Méthode:

1. On choisit u_0 quelconque, si possible proche d'un zéro de f.
2. On définit:
$$u_{n+1} = u_n - \frac{f(u_n)}{f'(u_n)}$$
3. On prie très fort pour que u_n converge vers un zéro de f.

Méthode de Héron

Si $a \in \mathbb{R}^+$ et $f(x) = x^2 - a$, on obtient la **méthode de Héron** (1er siècle après J-C.) pour approximer $\sqrt{a}$:

Théorème

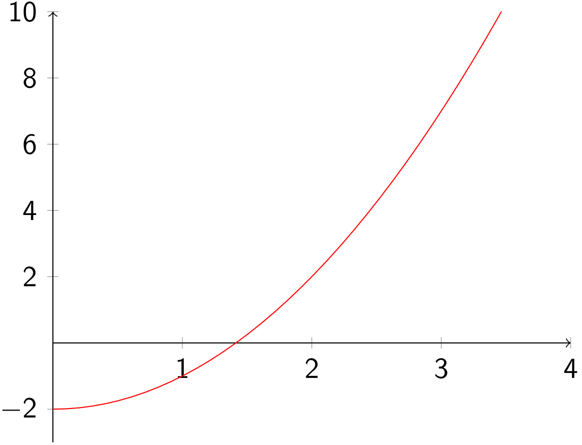
Soit (u_n) telle que:

$$\begin{cases} u_0 > 0 \\ u_{n+1} = \frac{1}{2}(u_n + \frac{a}{u_n}) \end{cases}$$

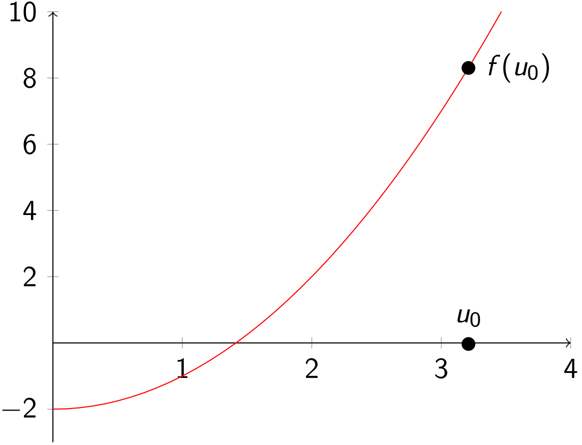
Alors:

$$u_n \underset{n \to \infty}{\longrightarrow} \sqrt{a}$$

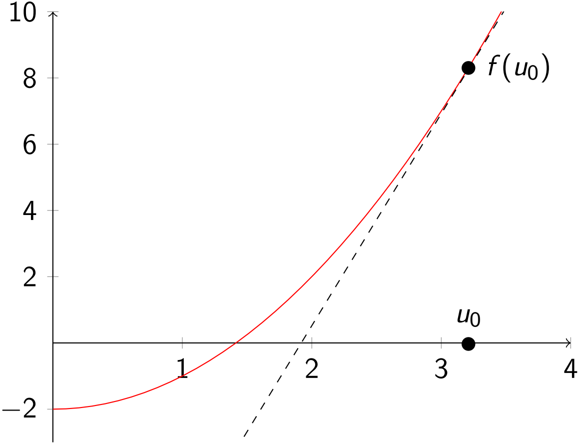
Méthode de Newton



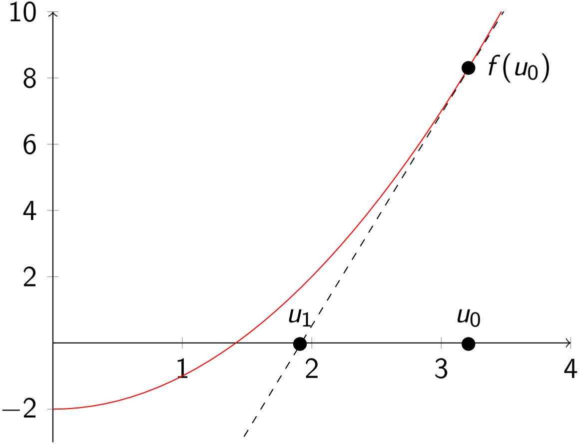
Méthode de Newton



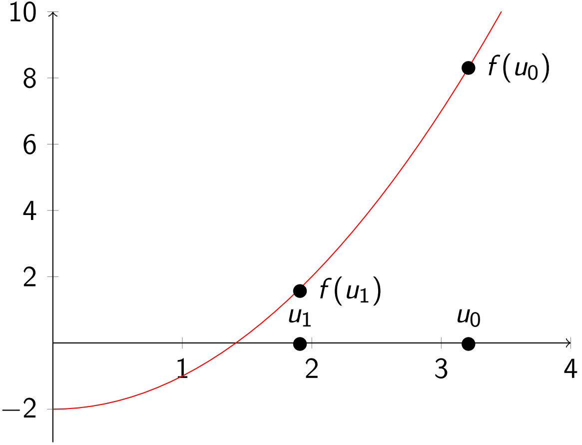
Méthode de Newton



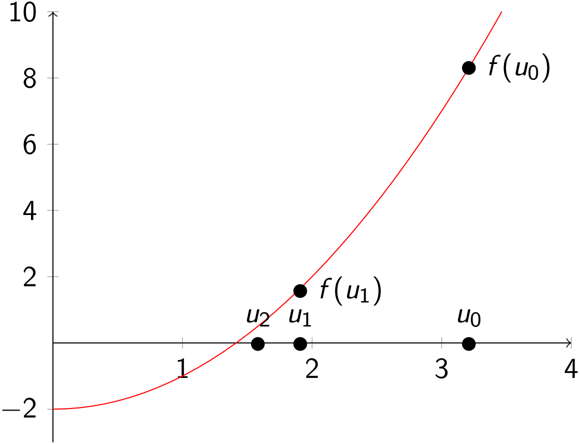
Méthode de Newton



Méthode de Newton



Méthode de Newton



Méthode de Newton

Question

Comment savoir si u_n approxime un zéro de f à ε près?

Question

Comment savoir si u_n approxime un zéro de f à ε près?

Condition **suffisante**: $f(u_n - \varepsilon)$ et $f(u_n + \varepsilon)$ sont de signes différents.

Méthode de Newton

Question

Comment savoir si u_n approxime un zéro de f à ε près?

Condition **suffisante**: $f(u_n - \varepsilon)$ et $f(u_n + \varepsilon)$ sont de signes différents.

Autres conditions d'arrêts envisageables:

1. $f(u_n) \leq \epsilon$
2. $|u_{n+1} - u_n| \leq \epsilon$
3. ...

Méthode de Newton

```
def newton(f, fp, un, epsilon):
    while f(un - epsilon) * f(un + epsilon) > 0:
        un = un - f(un)/fp(un)
    return un
```

- `fp`: dérivée de `f`.
- `un`: premier terme de la suite des approximations.
- `epsilon`: précision de l'approximation d'un zéro de `f`.

Méthode de Newton

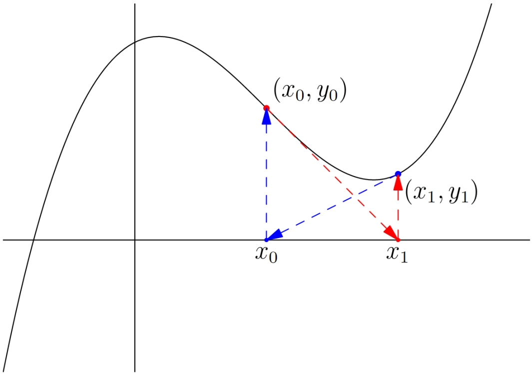
```python
def newton(f, fp, un, epsilon):
    while f(un - epsilon) * f(un + epsilon) > 0:
        un = un - f(un)/fp(un)
    return un
```

- `fp`: dérivée de `f`.
- `un`: premier terme de la suite des approximations.
- `epsilon`: précision de l'approximation d'un zéro de `f`.

`newton` peut ne pas terminer.

Méthode de Newton

`newton` peut ne pas terminer:



Convergence de la méthode de Newton

Théorème

Soit f une fonction $\mathcal{C}^2$ et c tel que $f(c) = 0$, $f'(c) \neq 0$.
Alors, si u_0 est assez proche de c, la suite u_n obtenue par la méthode de Newton converge vers c.
De plus il existe une constante K telle que:

$$|u_n - c| \leq K|u_{n-1} - c|^2$$

Convergence de la méthode de Newton

Théorème

Soit f une fonction $\mathcal{C}^2$ et c tel que $f(c) = 0$, $f'(c) \neq 0$.
Alors, si u_0 est assez proche de c, la suite u_n obtenue par la méthode de Newton converge vers c.
De plus il existe une constante K telle que:

$$|u_n - c| \leq K|u_{n-1} - c|^2$$

Preuve: utiliser l'inégalité de Taylor-Lagrange.

Convergence de la méthode de Newton

Dans les conditions du théorème, $|u_n - c| \leq K|u_{n-1} - c|^2$ signifie que la suite converge très rapidement (**quadratiquement**).

Convergence de la méthode de Newton

Dans les conditions du théorème, $|u_n - c| \leq K|u_{n-1} - c|^2$ signifie que la suite converge très rapidement (**quadratiquement**).

Si par exemple u_{n-1} est une approximation de c à 10^{-4} près, alors u_n est une approximation de c à 10^{-8} près.

Convergence de la méthode de Newton

Quelques problèmes possibles:

1. Comment trouver u_0 proche de c dans le théorème précédent?

Convergence de la méthode de Newton

Quelques problèmes possibles:

1. Comment trouver u_0 proche de c dans le théorème précédent?
2. Si $f'(u_n) = 0$, la tangente n'intersecte pas l'axe des abscisses.

Convergence de la méthode de Newton

Quelques problèmes possibles:

1. Comment trouver u_0 proche de c dans le théorème précédent?
2. Si $f'(u_n) = 0$, la tangente n'intersecte pas l'axe des abscisses.
3. Si $f'(u_n)$ est proche de 0, u_{n+1} peut partir très loin et s'éloigner de la solution.
4. ...

Convergence de la méthode de Newton

Quelques problèmes possibles:

1. Comment trouver u_0 proche de c dans le théorème précédent?
2. Si $f'(u_n) = 0$, la tangente n'intersecte pas l'axe des abscisses.
3. Si $f'(u_n)$ est proche de 0, u_{n+1} peut partir très loin et s'éloigner de la solution.
4. ...

On peut mélanger Newton et dichotomie.

Calcul de la dérivée

Question

Comment calculer la dérivée requise pour la méthode de Newton?

Calcul de la dérivée

Question

Comment calculer la dérivée requise pour la méthode de Newton?

1ère solution: la calculer à la main ou utiliser **sympy**.

Approximer la dérivée

2ème solution: approximer $f'(x)$.

$$f'(x) \approx \frac{f(x+h) - f(x)}{h}$$

Approximer la dérivée

2ème solution: approximer $f'(x)$.

$$f'(x) \approx \frac{f(x+h) - f(x)}{h}$$

La relation de récurrence devient:

$$u_{n+1} = u_n - f(u_n) \times \frac{h}{f(u_n + h) - f(u_n)}$$

Où h est une constante suffisamment petite.

Méthode de la sécante

On peut aussi approximer la tangente en u_n par:

$$\frac{f(u_n) - f(u_{n-1})}{u_n - u_{n-1}}$$

Méthode de la sécante

On peut aussi approximer la tangente en u_n par:

$$\frac{f(u_n) - f(u_{n-1})}{u_n - u_{n-1}}$$

On a alors:

$$u_{n+1} = u_n - f(u_n) \times \frac{u_n - u_{n-1}}{f(u_n) - f(u_{n-1})}$$

Méthode de la sécante

On peut aussi approximer la tangente en u_n par:

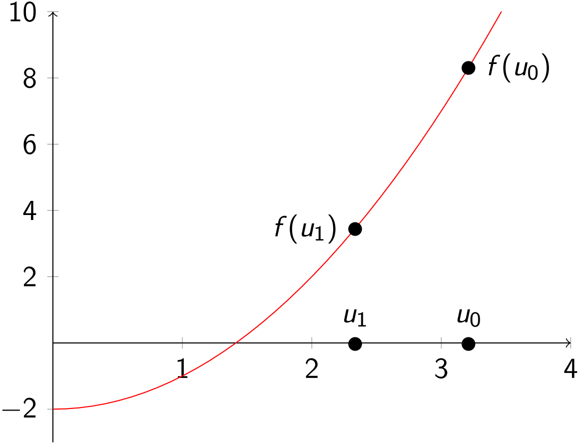
$$\frac{f(u_n) - f(u_{n-1})}{u_n - u_{n-1}}$$

On a alors:

$$u_{n+1} = u_n - f(u_n) \times \frac{u_n - u_{n-1}}{f(u_n) - f(u_{n-1})}$$

C'est la **méthode de la sécante**, qui converge moins vite mais ne demande pas le calcul de f'.

Méthode de la sécante



Méthode de la sécante