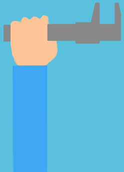
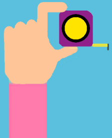
Méthode d'Euler

Renaud Costadoat
Lycée Dorian



DORIAN

Schéma d'Euler

Soit une équation différentielle écrite sous la forme :

$$\forall x \in I, u'(x) = f(x, u(x)), \text{ avec } u(x_0) = y_0$$

Objectif

L'objectif de l'utilisation de la méthode d'Euler est de calculer une approximation de la fonction u sur l'intervalle I. Cette approximation peut être:

- **locale**: si seule la valeur en un point nous intéresse,
- **globale**: calcul des valeurs de u à intervalles réguliers.

Schéma d'Euler

D'après le calcul du taux d'accroissement:

$$\forall x \in I, u'(x) = \lim_{x \to a} \frac{u(x) - u(a)}{x - a}$$

Ce qui peut s'écrire, après un changement de variable:

$$\forall x \in I, u'(x) = \lim_{dx \to 0} \frac{u(x + dx) - u(x)}{dx}$$

En reprenant l'équation différentielle du 1er ordre vu précédemment, on obtient:

$$\forall x \in I, f(x, u(x)) = \lim_{dx \to 0} \frac{u(x + dx) - u(x)}{dx}$$

Démarche

En discrétisant le problème précédent, il est alors possible d'écrire:

C'est la récurrence suivante qui va nous permettre de résoudre l'équation différentielle:

Pseudo code:

1. Soit x, un vecteur représentant la variable de dérivation,
2. Soit y_0, un réel représentant la condition initiale $u(x_0) = y_0$
3. $u(x_1) = (x_1 - x_0) * f(x, u(x_0)) + u(x_0)$
4. Récurrence: le réel x_n étant construit, calculer $u(x_{i+1}) = (x_{i+1} - x_i) * f(x, u(x_i)) + u(x_i)$

Démarche

En discrétisant le problème précédent, il est alors possible d'écrire:

$$f(x, u(x_i)) = \frac{u(x_{i+1}) - u(x_i)}{x_{i+1} - x_i}$$

C'est la récurrence suivante qui va nous permettre de résoudre l'équation différentielle:

$$u(x_{i+1}) = (x_{i+1} - x_i) * f(x, u(x_i)) + u(x_i)$$

Pseudo code:

1. Soit x, un vecteur représentant la variable de dérivation,
2. Soit y_0, un réel représentant la condition initiale $u(x_0) = y_0$
3. $u(x_1) = (x_1 - x_0) * f(x, u(x_0)) + u(x_0)$
4. Récurrence: le réel x_n étant construit, calculer $u(x_{i+1}) = (x_{i+1} - x_i) * f(x, u(x_i)) + u(x_i)$

Code python

Proposer le code python de la fonction `methode_Euler` .

Code python

Proposer le code python de la fonction `methode_Euler` .

```
def methode_euler(F,y0,t):
    y = [0]*len(t)
    y[0] = y0
    for i in range(len(t)-1):
        y[i+1] = y[i]+(t[i+1]-t[i])*F(y[i],t[i])
    return y
```

Application

On considère une équation différentielle d'ordre 1 avec condition initiale (problème de Cauchy) : $\begin{cases} y'(t) = y(t) \\ y(0) = 1 \end{cases}$

Résoudre cette équation à l'aide de la méthode d'Euler.

Application

On considère une équation différentielle d'ordre 1 avec condition initiale (problème de Cauchy) : $\begin{cases} y'(t) = y(t) \\ y(0) = 1 \end{cases}$

Résoudre cette équation à l'aide de la méthode d'Euler.

```
def F(y,t):
    return y

def methode_euler(F,y0,t):
    y = [0]*len(t)
    y[0] = y0
    for i in range(len(t)-1):
        y[i+1] = y[i]+(t[i+1]-t[i])*F(y[i],t[i])
    return y
```

Application: Le circuit R.C.

L'objet de l'étude suivante est le calcul de la tension aux bornes du condensateur dans un circuit R.C. lorsqu'il est en charge.

Donner les équations électriques qui régissent son comportement.

Donc,

Données:

- $R = 300\Omega$,
- $e(t) = 6V$,
- $C = 10mF$.

Application: Le circuit R.C.

L'objet de l'étude suivante est le calcul de la tension aux bornes du condensateur dans un circuit R.C. lorsqu'il est en charge.

Donner les équations électriques qui régissent son comportement. $\begin{cases} e(t) = U_R(t) + U_C(t) \\ U_R(t) = R.i(t) \\ i(t) = C.\dfrac{dU_C(t)}{dt} \end{cases}$

Donc, $\begin{cases} \dfrac{dU_C(t)}{dt} = \dfrac{e(t) - U_C(t)}{R.C} \\ U_C(0) = 0 \end{cases}$

Données:

- $R = 300\Omega$,
- $e(t) = 6V$,
- $C = 10mF$.

Application: Le circuit R.C.

Résoudre cette équation à l'aide de la méthode d'Euler.

Application: Le circuit R.C.

Résoudre cette équation à l'aide de la méthode d'Euler.

```
e=6
R=300
C=10*10**(-3)

def F(y,t):
    return (e-y)/(R*C)

def methode_euler(F,y0,t):
    y = [0]*len(t)
    y[0] = y0
    for i in range(len(t)-1):
        y[i+1] = y[i]+(t[i+1]-t[i])*F(y[i],t[i])
    return y
```