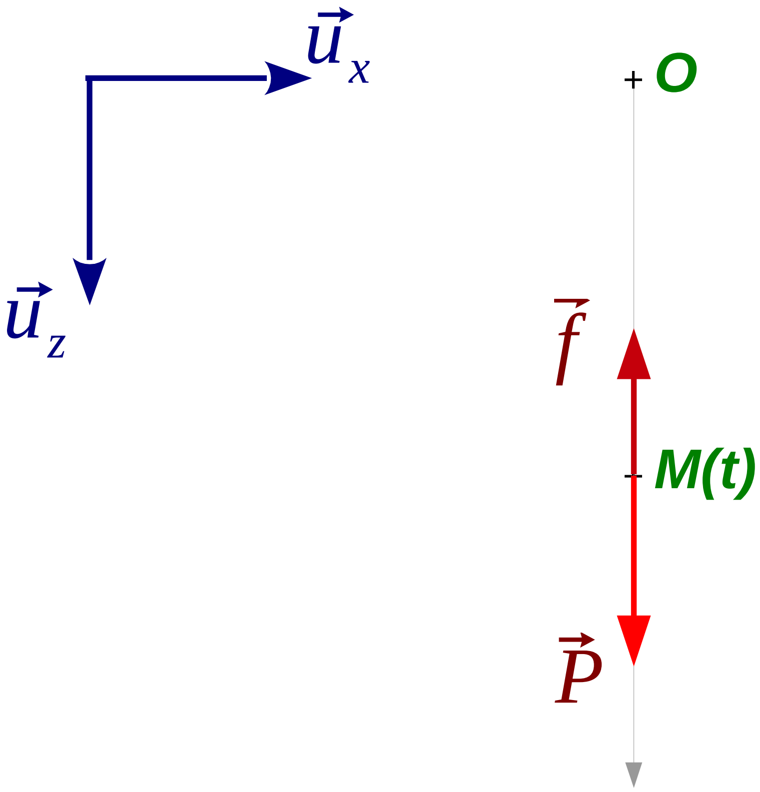
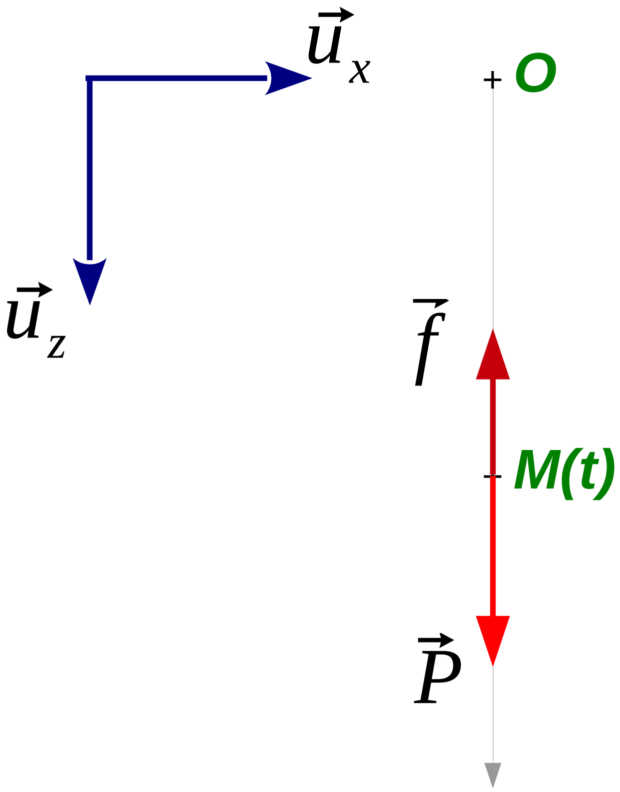
# Un problème de physique

## Chute d'un objet avec des frottements fluides en v²



# Un problème de physique

## Chute d'un objet avec des frottements fluides en v²




- Système : Point matériel M de masse m
- Référentiel : terrestre supposé galiléen
- Bilan des actions mécaniques
  - Poids : $\vec{P}=m\vec{g}=mg\,\vec{u}_z$
  - Frottements fluides : $\vec{f}=-\lambda\, v\,\vec{v}$

# Un problème de physique

## Chute d'un objet avec des frottements fluides en v²

(...)

# Un problème de physique

## Chute d'un objet avec des frottements fluides en $v^2$

Équation différentielle du mouvement :

$$\frac{dv}{dt} + \frac{\lambda}{m} v^2 = g$$

On reconnaît une équation différentielle du premier ordre...

***... non linéaire !***

# Position du problème

## Comment résoudre numériquement une équation différentielle d'ordre 1 ?

On cherche à « résoudre » numériquement l'équation différentielle d'ordre 1 (le problème de Cauchy, en fait...) sur l'intervalle [a,b]

$$\begin{cases} y'(t) = F(y(t),t) \\ y(0) = Y_0 \end{cases}$$

où $y$ est une fonction de R dans R, $y'$ sa dérivée par rapport à t, $F$ une fonction de t et de $y(t)$ et $Y_0$ une condition initiale

Résoudre numériquement cette équation différentielle revient à déterminer une approximation $y_k$ des valeurs $y(t_k)$ de la fonction $y$ en $t_k = a + h \times k$, où $h = \frac{b-a}{n}$ est le pas qu'il conviendra d'ajuster.

# Principe de la méthode

## Discrétisation du problème.

Nous allons approximer la solution de l'équation différentielle par la suite numérique $\left(y_n\right)$ définie par

$$y_n = y\left(t_n\right)$$

# Principe de la méthode

## Approximation par le taux d'accroissement

Nous allons approximer la solution de l'équation différentielle par la suite numérique $(y_n)$ définie par

$$y_n = y(t_n)$$

Cette suite peut être construite par récurrence en procédant à l'approximation suivante

$$y_{n+1} = y(t_{n+1}) = y(t_n + h)$$

# Principe de la méthode

## Approximation par le taux d'accroissement

Nous allons approximer la solution de l'équation différentielle par la suite numérique $(y_n)$ définie par

$$y_n = y(t_n)$$

Cette suite peut être construite par récurrence en procédant à l'approximation suivante

$$y_{n+1} = y(t_{n+1}) = y(t_n + h) \approx y(t_n) + h \frac{dy}{dt}(t_n)$$

# Principe de la méthode

## La méthode d'Euler explicite

Nous allons approximer la solution de l'équation différentielle par la suite numérique $(y_n)$ définie par

$$y_n = y(t_n)$$

Cette suite peut être construite par récurrence en procédant à l'approximation suivante

$$y_{n+1} = y(t_{n+1}) = y(t_n + h) \approx y(t_n) + h\frac{dy}{dt}(t_n)$$

Or, la valeur de la dérivée en $t_n$ est connue. Elle est donnée par l'équation différentielle !

# Principe de la méthode

## La méthode d'Euler explicite

La méthode d'Euler explicite consiste à remplacer le problème de Cauchy par une suite définie par récurrence

$$\begin{cases} y'(t) = F(t, y(t)) \\ y(0) = Y_0 \end{cases} \rightarrow \boxed{\begin{cases} y_{n+1} = y_n + h.F(y_n, t_n) \\ y_0 = Y_0 \end{cases}}$$

Où $h$ est le pas, qu'il conviendra de choisir judicieusement, et $Y_0$ la condition initiale (ou aux limites)

# Proposition d'algorithme

# Proposition d'algorithme

<u>Données</u> : $F, Y_0, a, b, n$

$h \leftarrow (b-a)/n$

$y_0 \leftarrow Y_0$

$t_0 \leftarrow a$

Pour $k$ de $1$ à $n$

- $y'_k \leftarrow F(y_k, t_k)$
- $y_{k+1} \leftarrow y_k + h \times y'_k$
- $t_{k+1} \leftarrow t_k + h$

Résultat $(y_0, \ldots, y_n)$

# Écriture d'une fonction python

# Écriture d'une fonction python

```
def euler1(f,a,b,y0,n):
    h = (b-a)/n # Détermination du pas
    y = [y0]    # Initialisation de la suite par la condition initiale
    t = a       # Initialisation du temps par la borne inf

    for k in range(0,n-1):
        y = y + [y[k] + h*f(y[k],t)] # Ajout de la nouvelle valeur de y
        t = t + h                       # Temps au pas suivant

    return y
```

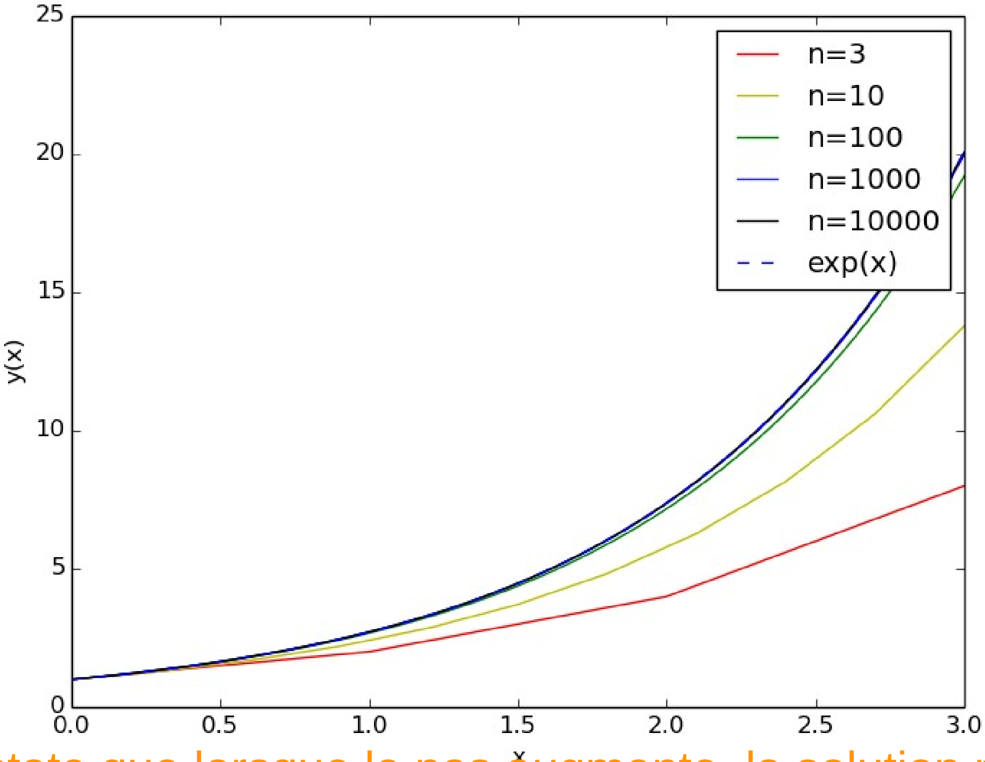
# Vérification sur un cas simple

Résoudre le problème de Cauchy suivant, entre 0 et 3

$$\begin{cases} y' = y \\ y(0) = 1 \end{cases}$$

- On déterminera l'expression de la fonction F
- On cherchera l'approximation de la solution pour différente valeur du pas
- Commenter l'influence du pas

# Vérification sur un cas simple

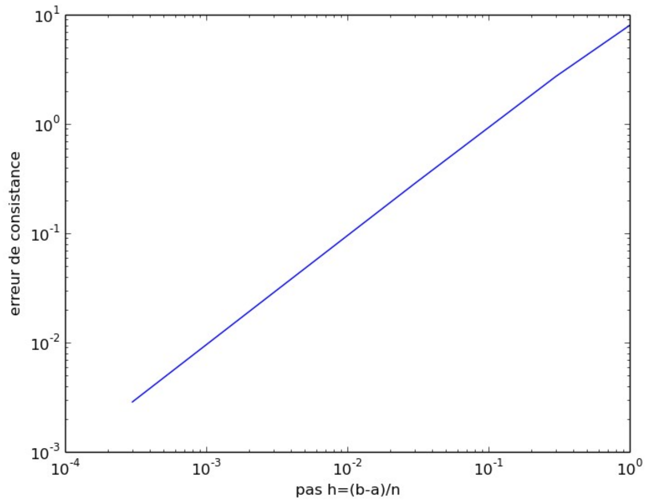



- On constate que lorsque le pas augmente, la solution numérique se rapproche de la solution analytique

# Vérification sur un cas simple

- Déterminons la complexité en temps de calcul. Pour cela, on va s'intéresser à l'écart entre la solution numérique et la solution analytique

$$e(h) = \sum_{i=0}^{n(h)-1} |y(t_{i+1}) - (y(t_i) + hF(y(t_i), t_i))|$$




- On constate que l'erreur croit linéairement avec le pas : la méthode d'Euler est une méthode dite d'ordre 1

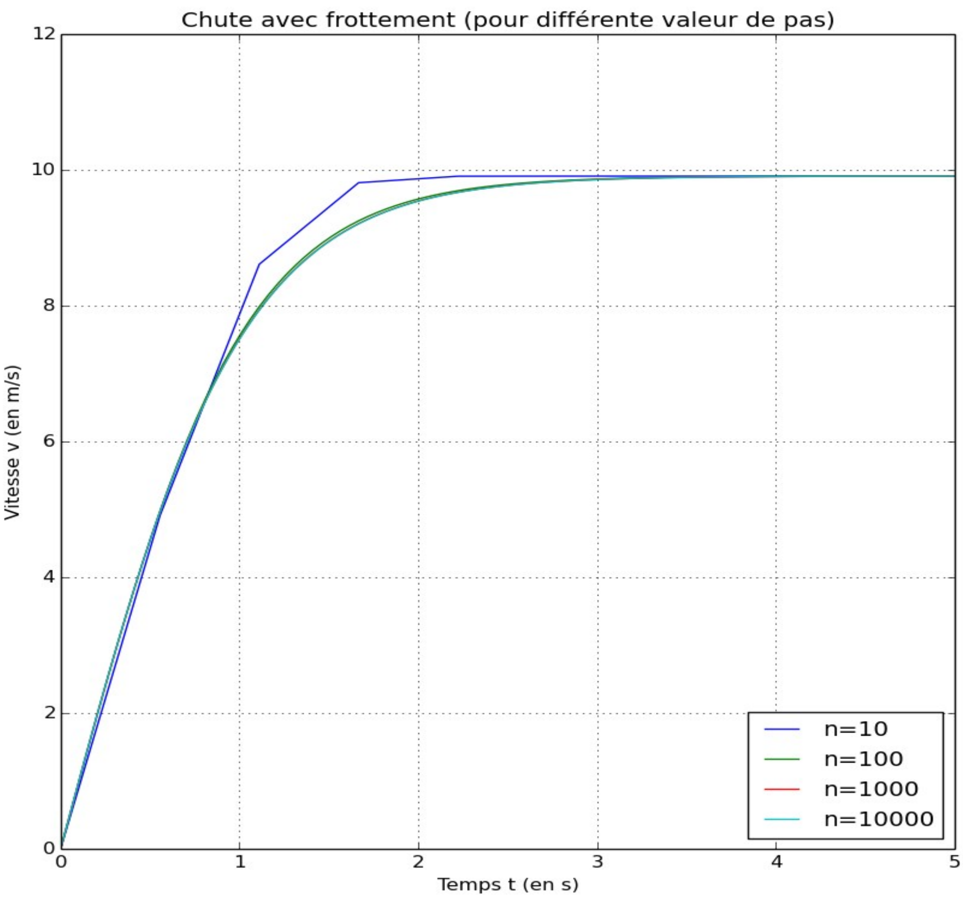
# Retour sur l'exemple du début

Résoudre le problème de Cauchy suivant, entre 0 et 10 s

$$\begin{cases} \dfrac{dv}{dt} = \dfrac{-\lambda}{m} v^2 + g \\ v(0) = 0 \end{cases}$$

- On déterminera l'expression de la fonction F
- On cherchera l'approximation de la solution pour différentes valeurs du pas
- Commenter l'influence du pas
- Déterminer la vitesse limite

# Retour sur l'exemple du début



# Retour sur l'exemple du début

Évolution de la position ?

1. Comment passe-t-on de la vitesse à la position
2. Proposer une fonction python effectuant cette opération
3. En déduire l'évolution de la position (qu'on tracera)

# Une fonction toute faite

Python propose bien-sûr des fonctions toutes faites pour résoudre des ED.

Le module *integrate* de la bibliothèque *scipy* contient la fonction *odeint* qui résout numériquement des ED

```python
from scipy.integrate import odeint

# Initialisations
m=1
g=9.81
l=0.1
v0=0
t=linspace(0,5,1000)

# Défintion de la fonction
def f(y,t):
    return -l/m*y**2+g

# Résolution de l'Ed par odeint
v = odeint(f,v0,t)

# On trace la jôlie courbe
plot(t,v)
legend(loc=4)
grid()
xlabel("Temps t (en s)")
ylabel("Vitesse v (en m/s)")
ylim(0,12)
title("Chute avec frottement (par odeint)")
```