INSTITUT d'OPTIQUE GRADUATE SCHOOL ParisTech
LEnsE
Laboratoire d'Enseignement Expérimental

Intégration Numérique

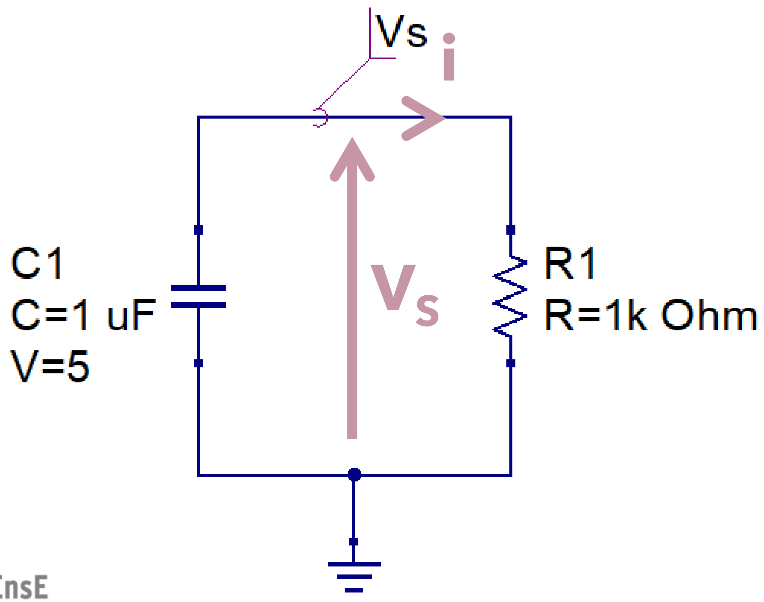
(Euler)

Outils Numériques / Semestre 5 / Institut d'Optique / B1_4

Cas des équations différentielles

Approche analytique

- Approche analytique



$$\Rightarrow \quad V_s = -R_1 . C_1 . \frac{d\,V_s}{dt}$$

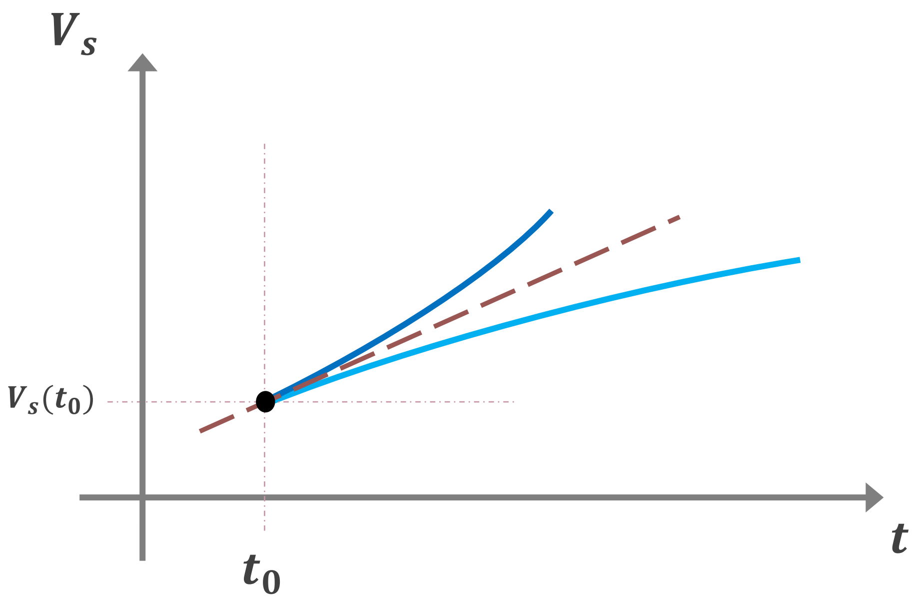
Equation différentielle d'ordre 1

$$\Rightarrow \quad \frac{d\,V_s}{dt} = -\frac{1}{R_1 . C_1} . V_s$$

INSTITUT d'OPTIQUE GRADUATE SCHOOL ParisTech | LEnsE Laboratoire d'Enseignement Expérimental

Approche graphique

- Intégration numérique



$$V_s = -R_1.C_1.\frac{d\,V_s}{dt}$$

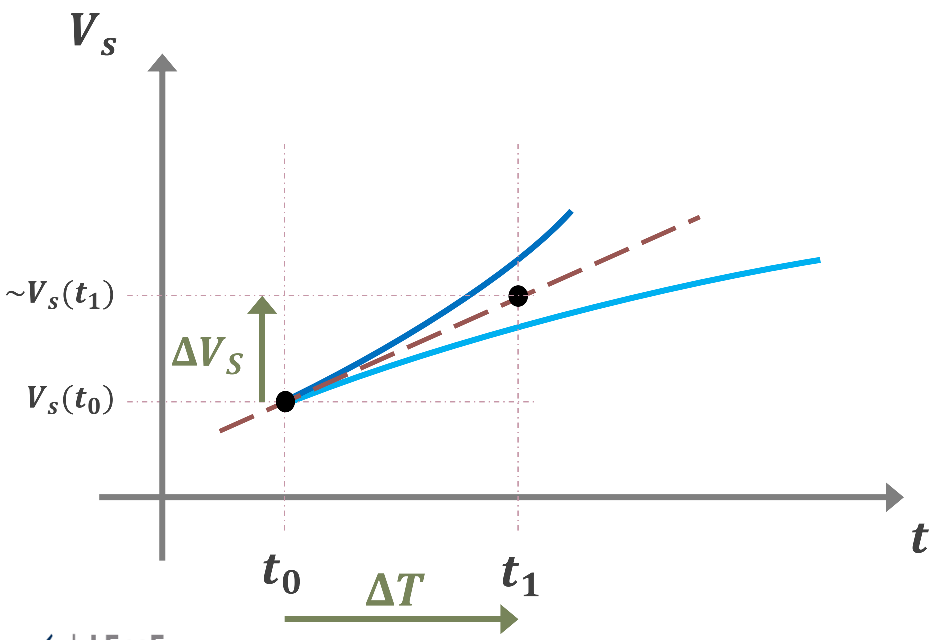
Equation différentielle d'ordre 1

$$\frac{d\,V_s}{dt} = -\frac{1}{R_1.C_1}.V_s$$

Coefficient directeur de la tangente à la courbe en un point donné

Approche graphique

- Intégration numérique



$$V_s = -R_1 . C_1 . \frac{d\,V_s}{dt}$$

Equation différentielle d'ordre 1

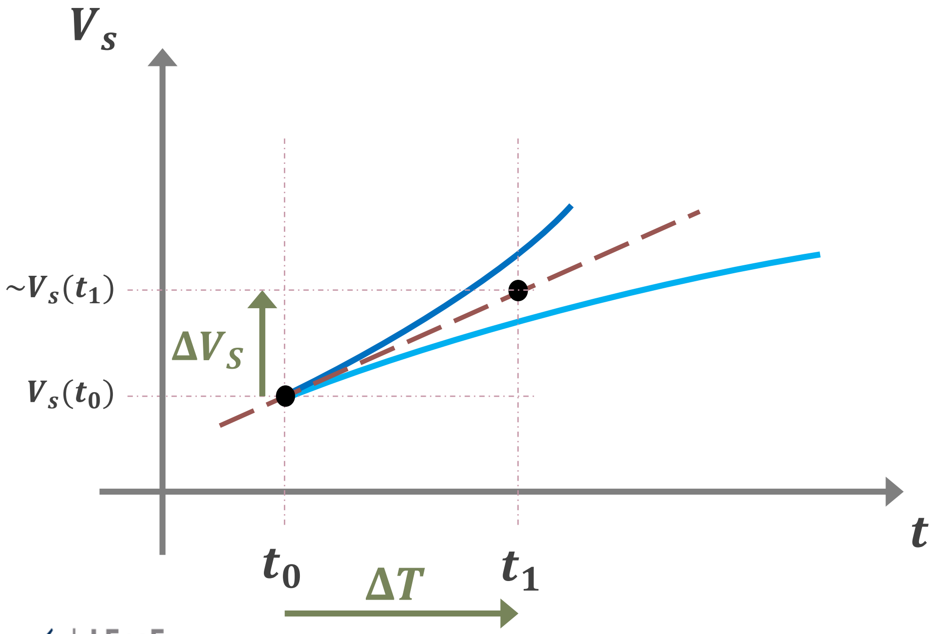
$$\frac{d\,V_s}{dt} = -\frac{1}{R_1.C_1} . V_s$$

Coefficient directeur de la tangente à la courbe en un point donné

Laboratoire d'Enseignement Expérimental

Approche graphique

- Intégration numérique



$$\frac{d\,V_s}{dt} = -\frac{1}{R_1.C_1}.V_s$$

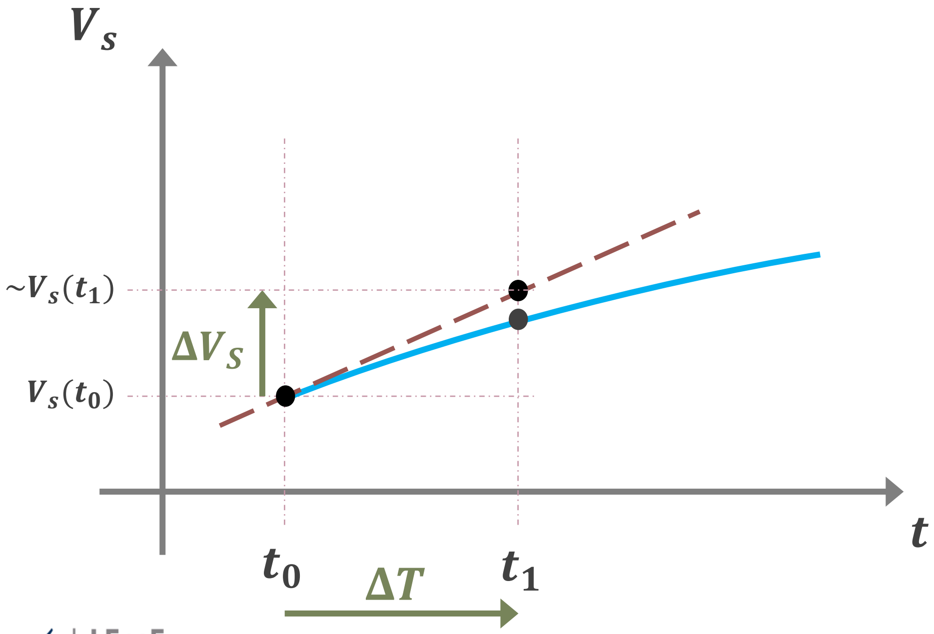
Coefficient directeur de la tangente à la courbe en un point donné

$$\sim V_s(t_1) = V_s(t_0) + \Delta V_S$$

avec $$\Delta V_S = \Delta T \,.\, \frac{d\,V_s}{dt}\bigg|_{t=t_0}$$

Approche graphique

- Intégration numérique



$$\frac{dV_s}{dt} = -\frac{1}{R_1.C_1}.V_s$$

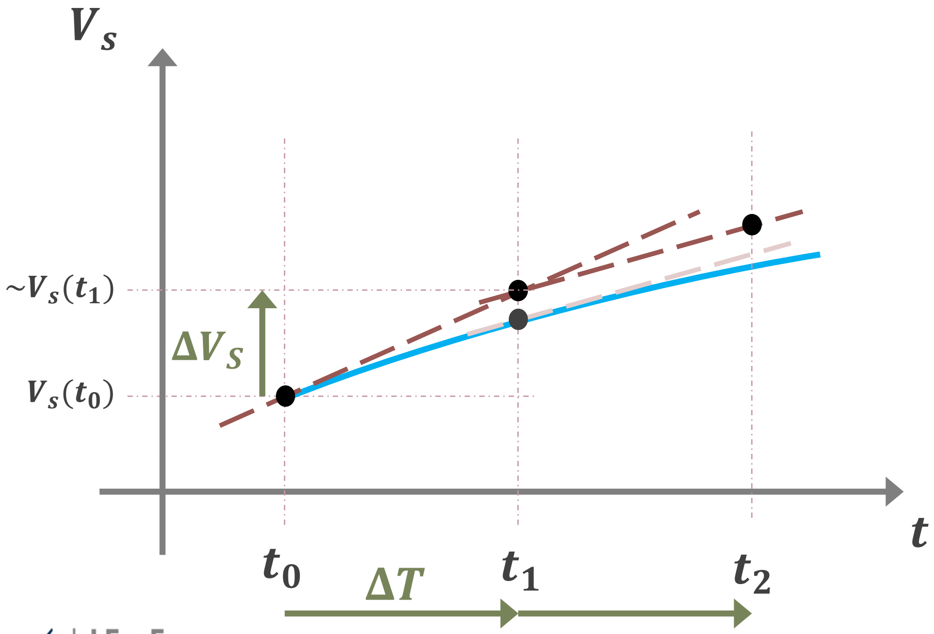
Coefficient directeur de la tangente à la courbe en un point donné

$$V_s(t_1) \sim V_s(t_0) + \Delta T \,.\, \left.\frac{dV_s}{dt}\right|_{t=t_0}$$

Laboratoire d'Enseignement Expérimental

Approche graphique

- Intégration numérique



$$\frac{dV_s}{dt} = -\frac{1}{R_1.C_1}.V_s$$

Coefficient directeur de la tangente à la courbe en un point donné

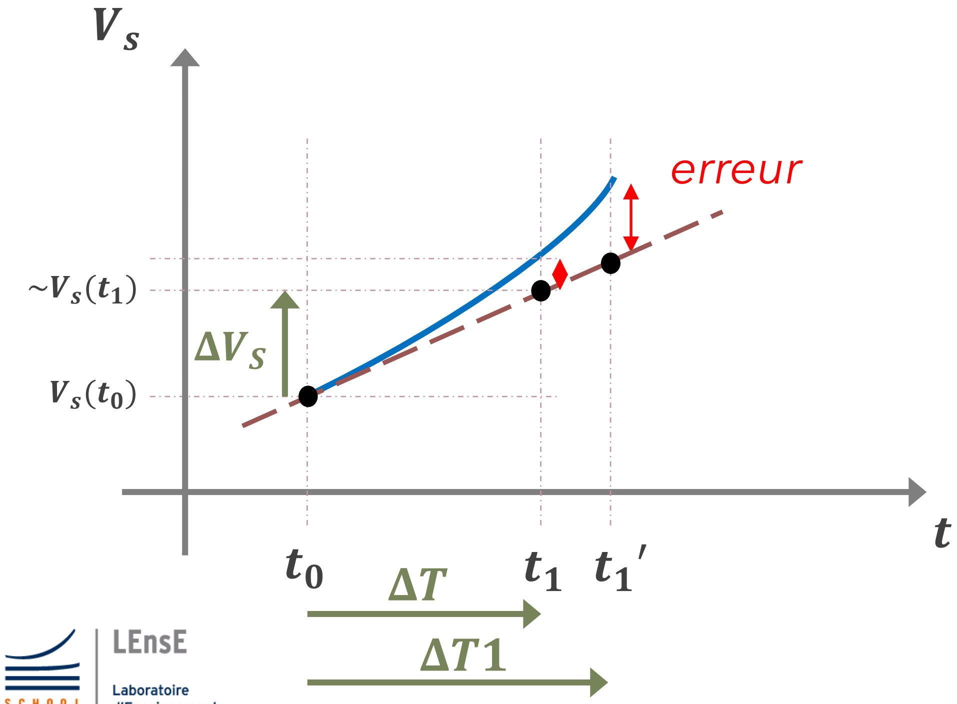
$$V_s(t_1) \sim V_s(t_0) + \Delta T.\left.\frac{dV_s}{dt}\right|_{t=t_0}$$

$$V_s(t_2) \sim V_s(t_1) + \Delta T.\left.\frac{dV_s}{dt}\right|_{t=t_1}$$

INSTITUT d'OPTIQUE GRADUATE SCHOOL ParisTech | LEnsE Laboratoire d'Enseignement Expérimental

Approche graphique

- Méthode d'**Euler** (explicite)



$$\frac{d\,V_s}{dt} = -\frac{1}{R_1.C_1}.V_s$$

Coefficient directeur de la tangente à la courbe en un point donné

$$V_s(t_{n+1}) \sim V_s(t_n) + \Delta T\,.\left.\frac{d\,V_s}{dt}\right|_{t=t_n}$$

APPROXIMATION NUMERIQUE

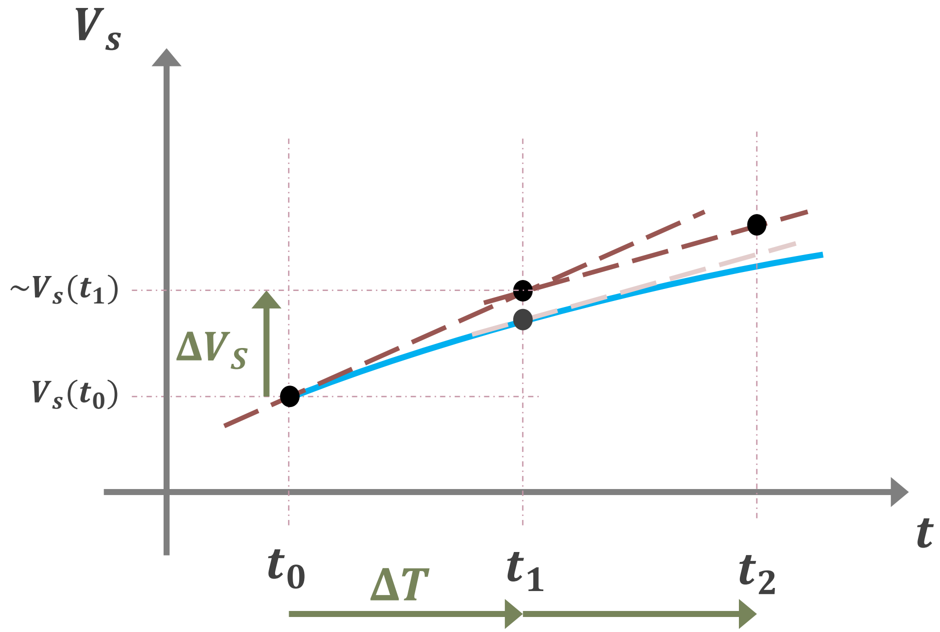
Intégration Numérique

- Intégration Numérique

Calculer des **valeurs approchées** d'une fonction **y(t)**

sur un intervalle de $[t_0, n.\Delta T]$ de **t**

en connaissant sa **dérivée** et un point particulier de la fonction



Implémentation de la méthode d'Euler

- Données d'entrée

$\boldsymbol{F(t, V_S(t))}$: fonction qui définit l'équation différentielle
$\boldsymbol{V_S(t_0)}$: condition initiale
$\boldsymbol{N}$ et $\boldsymbol{\Delta T}$: le nombre de points souhaités et le pas de calcul d'intégration

ou $\boldsymbol{T_{total}}$ et $\boldsymbol{\Delta T}$

- Algorithme de calcul

$$\boldsymbol{t_{n+1} = t_n + \Delta T}$$

$$\boldsymbol{V_s(t_{n+1}) \sim V_s(t_n) + \Delta T \,.\, F(t_n, V_S(t_n))}$$

INSTITUT d'OPTIQUE GRADUATE SCHOOL ParisTech | LEnsE Laboratoire d'Enseignement Expérimental

Implémentation de la méthode d'Euler

S'ENTRAINER

- Résolution par intégration

$$\frac{dV_s}{dt} = -\frac{1}{R_1.C_1}.V_s$$

spyder

+ Définir la fonction $F(V_S(t_n), t_n)$

+ Implémenter l'algorithme de la méthode d'Euler explicite dans une fonction prenant comme paramètres : $F, V_S(0), T_{total}, N$**, R, C**

+ Tracer l'évolution en fonction du temps pour R = 100 kΩ et C = 1 µF et au moins 3 valeurs de ΔT **différentes (pour un temps total équivalent)**

$$V_s(t_{n+1}) \sim V_s(t_n) + \Delta T \, . \, F(V_S(t_n), t_n)$$

INSTITUT d'OPTIQUE GRADUATE SCHOOL ParisTech | LEnsE Laboratoire d'Enseignement Expérimental

Implémentation de la méthode d'Euler

CORRECTION

- Définition de la fonction ***F***

```
def F(t, Vs, R=1e5, C=1e-6):
    return -Vs/(R*C)
```

- Implémentation Euler

```
def explicit_euler(F, Vs0, Ttot, N, R=1e5, C=1e-6):
    dt = Ttot/N
    t = numpy.zeros(N+1)
    Vs = numpy. zeros(N+1)
    Vs[0] = Vs0

    for n in range(N):
        t[n+1] = t[n] + dt
        Vs[n+1] = Vs[n] + F(t[n], Vs[n], R, C)*dt

    return t, Vs
```

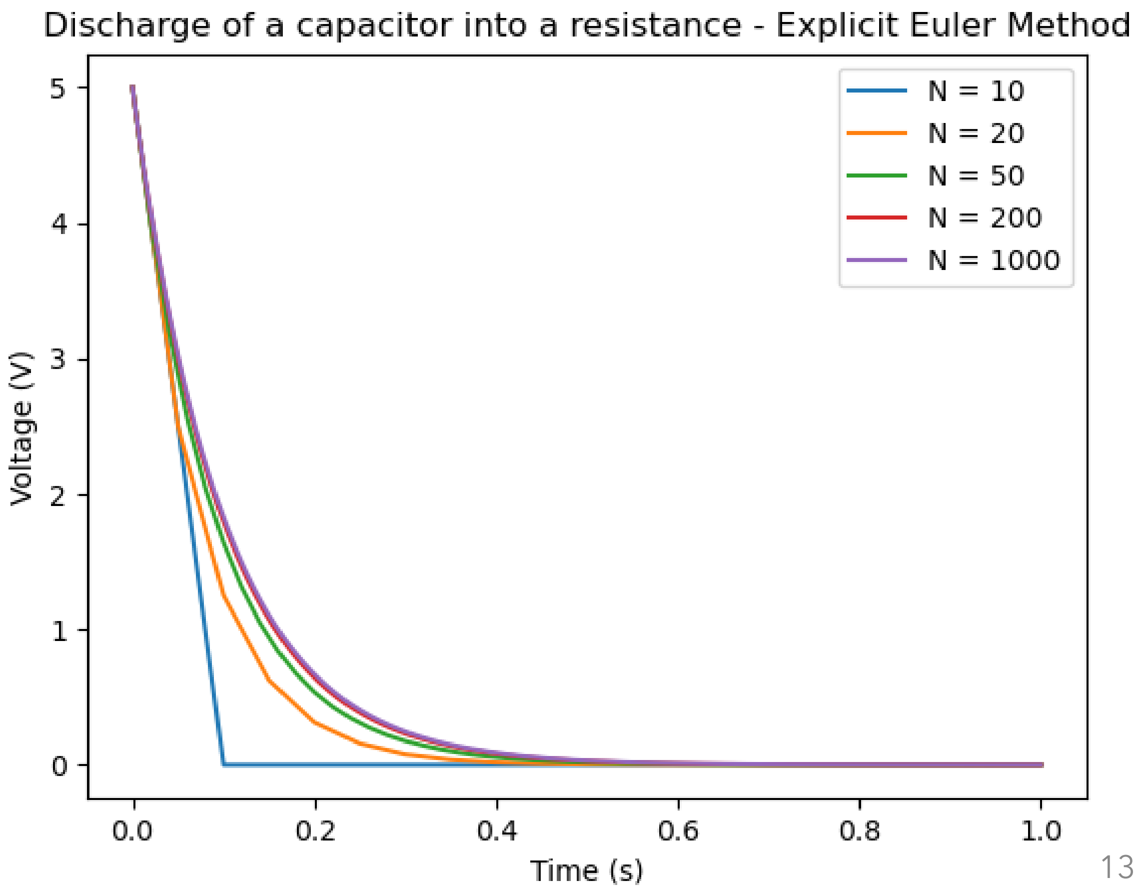
INSTITUT d'OPTIQUE GRADUATE SCHOOL ParisTech | LEnsE Laboratoire d'Enseignement Expérimental

Implémentation de la méthode d'Euler

RESULTATS

- Simulation paramétrique

```
N_val = [10, 20, 50, 200, 1000]
plt.figure()
for k in range(len(N_val)):
    t, Vs = explicit_euler(F, 5, 1, N_val[k])
    plt.plot(t, Vs, label=f'N = {N_val[k]}')

plt.legend()
plt.show()
```



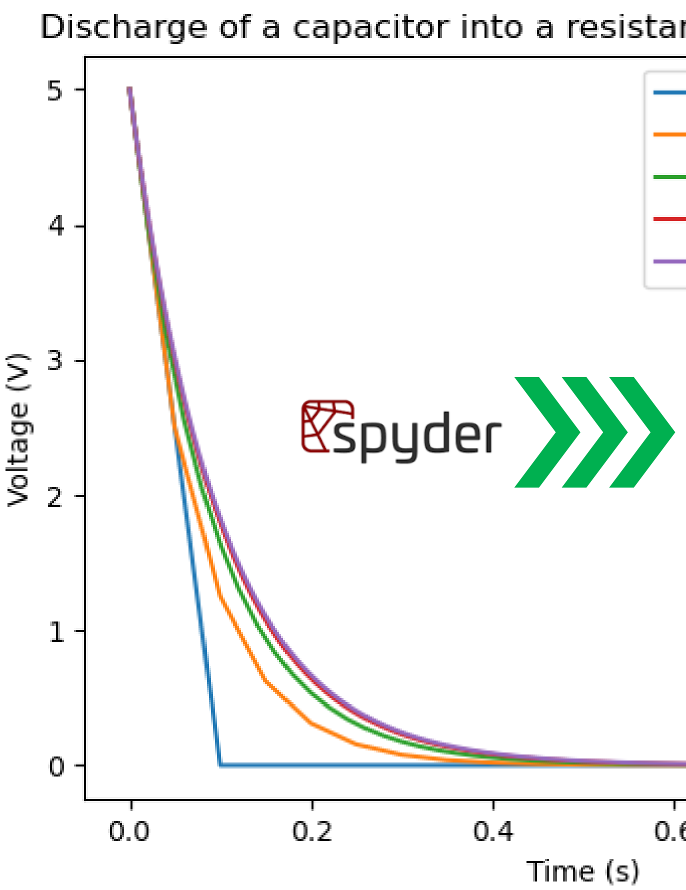
INSTITUT d'OPTIQUE GRADUATE SCHOOL ParisTech | LEnsE Laboratoire d'Enseignement Expérimental

Implémentation de la méthode d'Euler

RESULTATS

- Temps de calcul *vs* N

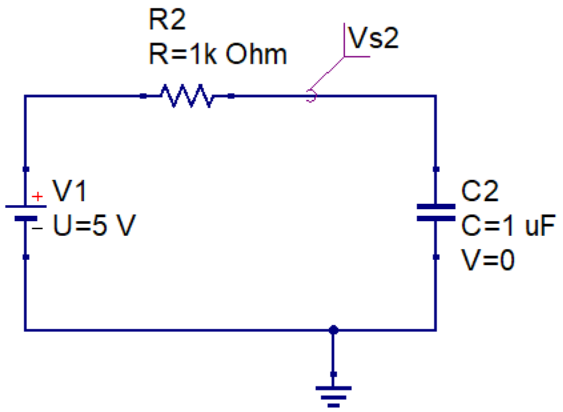
```
import time

st = time.process_time()
[ bloc d'instructions à évaluer ]
et = time.process_time()

vt = int((et-st) * 1e3)
print(f'Execution Time = {vt} ms')
```

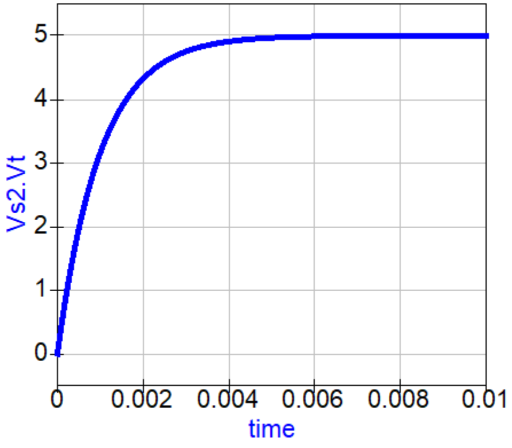


+ Mesurer le temps d'exécution pour N = 2000 et Ttotal = 5s

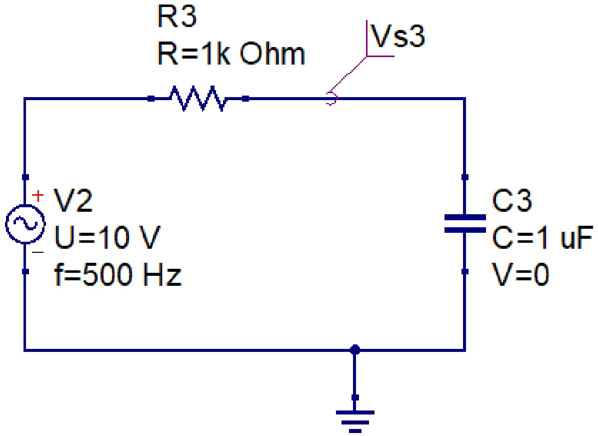
Circuits similaires / Généralisation

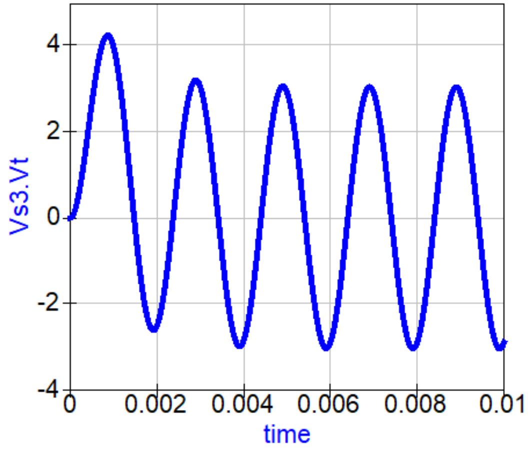
- Réponse à un échelon





- Régime forcé





$$\frac{dV_s}{dt} = -\frac{1}{R_1.C_1}.(V_s - V_e)$$

Simulation réalisée avec QUCS
Quite Universal Circuit Simulator
https://qucs.sourceforge.net/

Implémentation de la méthode d'Euler

OPTIMISATION

- Limite

$\boldsymbol{F(V_S(t), t, R, C, Ve(t))}$: fonction qui définit l'équation différentielle
$\boldsymbol{V_S(t_0)}$: condition initiale
$\boldsymbol{N}$ **et** $\boldsymbol{T_{total}}$: le nombre de points souhaités et le temps total

Comment rendre paramétrable R et C ?

Comment rendre dépendant $\boldsymbol{F}$ *de* $\boldsymbol{Ve(t)}$ *?*

INSTITUT d'OPTIQUE GRADUATE SCHOOL ParisTech | LEnsE Laboratoire d'Enseignement Expérimental

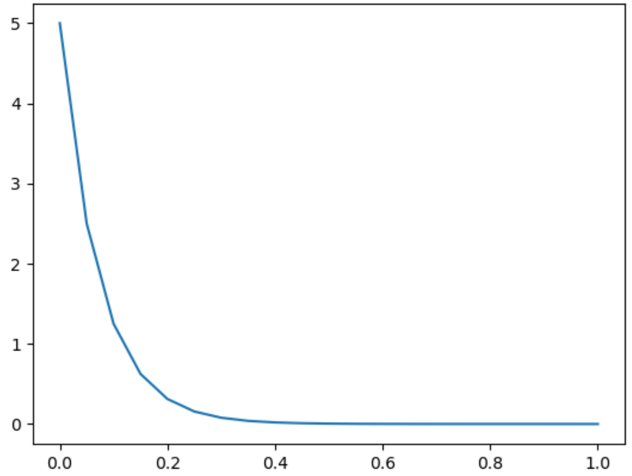
Implémentation de la méthode d'Euler

OPTIMISATION

- Définition d'une classe $\boldsymbol{F}$

```
class F:
    def __init__(self, R, C, Vs0):
        self.C = C
        self.R = R
        self.Vs0 = Vs0

    def __call__(self, t, Vs):
        return -Vs/(self.R*self.C)
```

```
newF = F(1e5, 1e-6, 5)
t, Vs = explicit_euler(newF, F.Vs0, 1, 1000)
```



Implémentation de la méthode d'Euler

ALLER PLUS LOIN | OPTIMISATION

- Définition d'une classe ***F***

```
class F:
    def __init__(self, R, C, Vs0, Ve):
        self.C = C
        self.R = R
        self.Vs0 = Vs0
        self.Ve = Ve

    def __call__(self, t, Vs):
        return (self.Ve-Vs)/(self.R*self.C)
```

```
newF = F(1e5, 1e-6, 0, ?? )
t, Vs = explicit_euler(newF, F.Vs0, 1, 1000)
```

spyder

+ Définir une nouvelle classe *F_sin*, ayant pour paramètre une amplitude et une fréquence
+ Tracer l'évolution d'un régime forcé pour une entrée sinusoïdale

$$\frac{dV_s}{dt} = -\frac{1}{R_1.C_1}.(V_s - V_e)$$

Intégration Numérique

(Scipy)

Outils Numériques / Semestre 5 / Institut d'Optique / B1_4

Cas des équations différentielles

Scientific Python / SciPy

- Boite à outils pour les sciences

SciPy User Guide

SciPy

- Introduction
- Special functions (`scipy.special`)
- Integration (`scipy.integrate`)
- Optimization (`scipy.optimize`)
- Interpolation (`scipy.interpolate`)
- Fourier Transforms (`scipy.fft`)
- Signal Processing (`scipy.signal`)
- Linear Algebra (`scipy.linalg`)
- Sparse eigenvalue problems with ARPACK
- Compressed Sparse Graph Routines (`scipy.sparse.csgraph`)
- Spatial data structures and algorithms (`scipy.spatial`)
- Statistics (`scipy.stats`)
- Multidimensional image processing (`scipy.ndimage`)
- File IO (`scipy.io`)

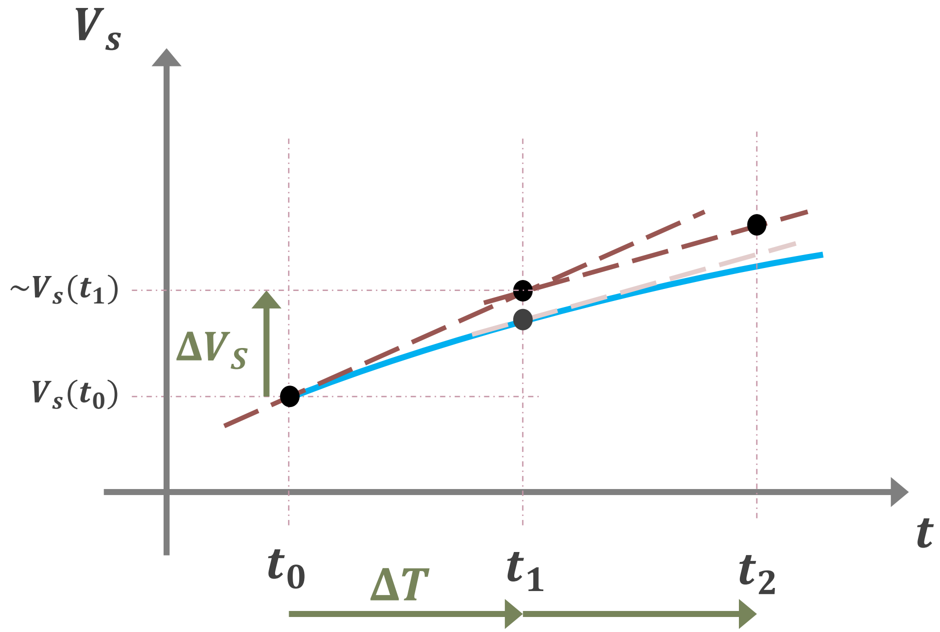
Intégration Numérique / SciPy

- Intégration Numérique

Calculer des **valeurs approchées** d'une fonction **y(t)**

sur un intervalle de $[t_0, n.\Delta T]$ de **t**

en connaissant sa **dérivée** et un point particulier de la fonction



Intégration Numérique / SciPy

S'ENTRAINER

- Résolution par intégration

$$\frac{dV_s}{dt} = -\frac{1}{R_1.C_1}.V_s$$

*scipy.integrate.***solve_ivp**($\boldsymbol{F}$, $[t_0, \boldsymbol{T_{final}}]$, $[\boldsymbol{V_s(t_0)}]$)

Cette fonction retourne des données encapsulées sous la forme de deux vecteurs : ***t*** *et* ***y***

y *doit être transposé pour pouvoir être affiché par rapport à* ***t***

sol = scipy.integrate.**solve_ivp**(F, [0,1], [5])
# résolution entre t = 0 et t = 1, pour Vs(0) = 5

time = sol.t
Vs_t = sol.y.flat

Intégration Numérique / SciPy

S'ENTRAINER

- Résolution par intégration

$$\frac{dV_s}{dt} = -\frac{1}{R_1.C_1}.V_s$$

+ Comparer la méthode d'Euler explicite et la méthode *solve_ivp* implémentée dans Scipy.integrate

- sur la décharge d'un condensateur

- sur le régime forcé d'un filtre RC avec une entrée sinusoïdale

scipy.integrate.**solve_ivp**($\boldsymbol{F}$, $[t_0, \boldsymbol{T_{final}}]$, $[\boldsymbol{V_s(t_0)}]$)

*Cette fonction retourne des données encapsulées sous la forme de deux vecteurs : **t** et **Vs***

Vs** doit être transposé pour pouvoir être affiché par rapport à **t

D’autres méthodes plus optimisées

- Méthodes de Runge-Kutta

Méthode d'Euler → tangente en un point pour trouver le suivant

Méthodes de Runge Kutta

(ordre 2) → création d’un point intermédiaire entre deux points expérimentaux

(ordre 4) → création de 3 points intermédiaires Par défaut / RK45

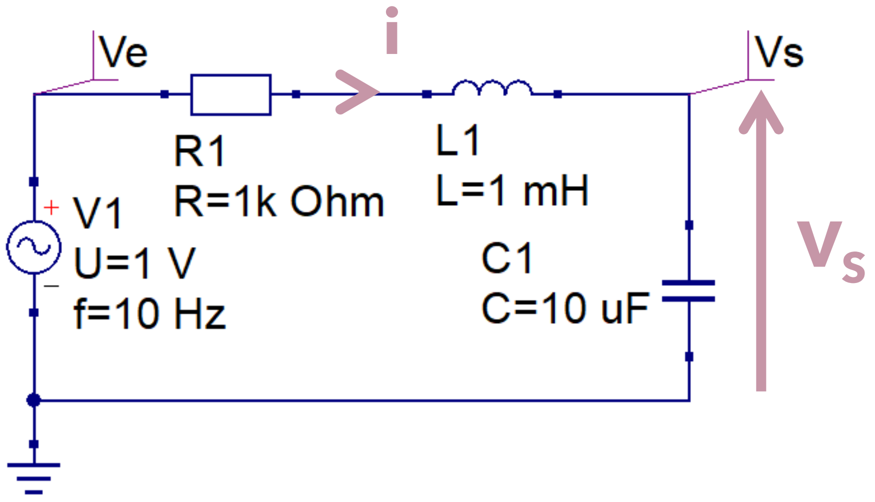
scipy.integrate.**solve_ivp**(… , method=‘RK23’)

scipy.integrate.**solve_ivp**(… , method=‘RK45’)

INSTITUT d'OPTIQUE GRADUATE SCHOOL ParisTech | LEnsE Laboratoire d'Enseignement Expérimental

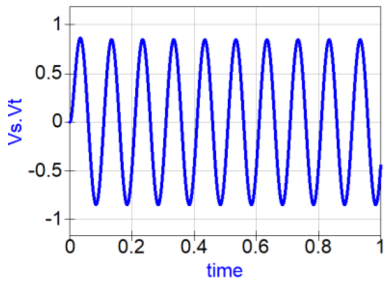
Autre cas / Equation du second ordre

- Circuit RLC



$$V_e = L_1 . C_1 . \frac{d^2 V_s}{dt^2} + R_1 . C_1 . \frac{d\, V_s}{dt} + V_s$$

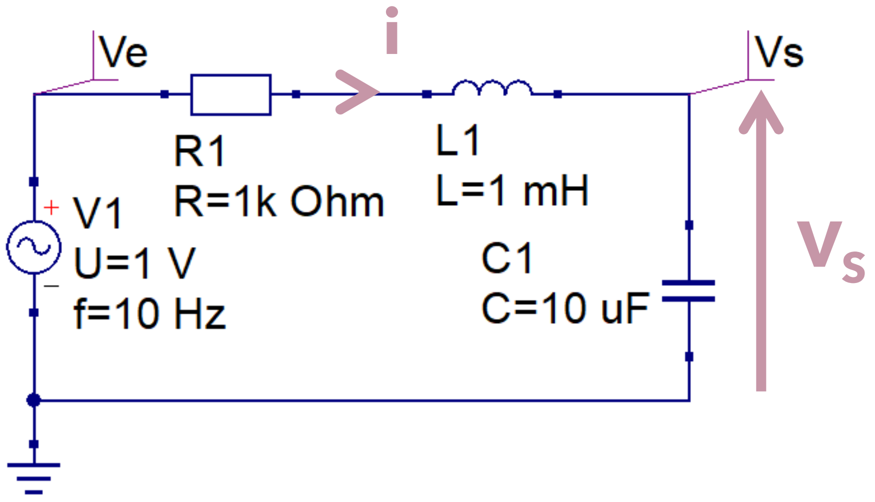
Equation différentielle d'ordre 2



INSTITUT d'OPTIQUE GRADUATE SCHOOL ParisTech
LEnsE Laboratoire d'Enseignement Expérimental

Autre cas / Equation du second ordre

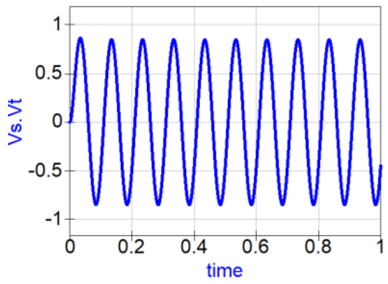
- Circuit RLC



$$V_e = L_1 . C_1 . \frac{d^2 V_s}{dt^2} + R_1 . C_1 . \frac{d\,V_s}{dt} + V_s$$

Equation différentielle d'ordre 2

solve_ivp uniquement pour **ordre 1**



Cas d'une équation différentielle d'ordre 2

- Approche analytique

$$V_e = L_1 . C_1 . \frac{d^2 V_s}{dt^2} + R_1 . C_1 . \frac{d\, V_s}{dt} + V_s$$

$$\frac{d^2 V_s}{dt^2} = -\frac{R_1}{L_1} . \frac{d\, V_s}{dt} - \frac{1}{L_1 . C_1} (V_s - V_e)$$

Système de deux équations différentielles d'ordre 1

$$\frac{d\, V_s}{dt} = u$$

$$\frac{du}{dt} = -\frac{R_1}{L_1} . u - \frac{1}{L_1 . C_1} (V_s - V_e)$$

Cas d’une équation différentielle d’ordre 2

- Méthode d’Euler
  forme matricielle

Système de deux équations différentielles d’ordre 1

$$\boldsymbol{F(t,u) = -\frac{R_1}{L_1}.u - \frac{1}{L_1.C_1}(V_s - V_e)}$$

$$\boldsymbol{\frac{d\,V_s}{dt} = u}$$

$$\boldsymbol{\frac{du}{dt} = F(t,u)}$$

Posons :

$$\boldsymbol{y = \begin{pmatrix} V_s \\ u \end{pmatrix}}$$

$$\boldsymbol{\frac{d\,y}{dt} = \begin{pmatrix} u \\ F(t,u) \end{pmatrix} = F(t, y(t))}$$

Ainsi :

$$\boldsymbol{y_{n+1} = y_n + \Delta T . F(t_n, y_n)}$$

INSTITUT d'OPTIQUE GRADUATE SCHOOL ParisTech | LEnsE Laboratoire d'Enseignement Expérimental

Cas d'une équation différentielle d'ordre 2

- Mise en équation

$$y = \begin{pmatrix} V_s \\ u \end{pmatrix} \qquad \frac{dy}{dt} = \begin{pmatrix} u \\ F(t,u) \end{pmatrix} = F(t, y(t))$$

$$F(t,u) = -\frac{R_1}{L_1}.u - \frac{1}{L_1.C_1}(V_s - V_e)$$

$$y_{n+1} = y_n + \Delta T . F(t_n, y_n)$$

```
class RLC_forced:
    def __init__(self, R, L, C, Ve):
        [...]
    def __call__(self, t, y ):
        Vs = y[0]
        u = y[1]
        return [u, -R/L*u - (Vs - Ve(t)) / (L*C)]
```

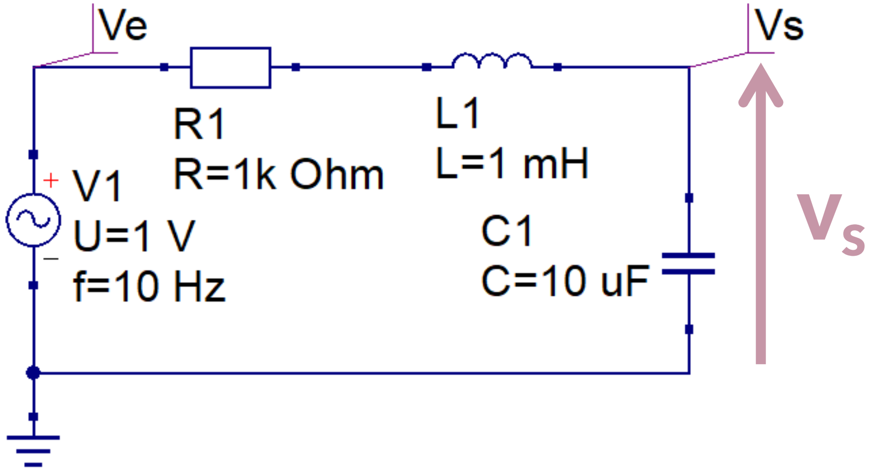
Vecteur
y[0] = Vs
y[1] = u = Vs'

Intégration Numérique / SciPy

ALLER PLUS LOIN

- Résolution par intégration

+ Tracer l'évolution de *Vs(t)* dans un circuit RLC soumis à une tension sinusoïdale de fréquence 20 Hz, avec R = 1 kΩ, L = 1 mH et C = 10 µF



*scipy.integrate.***solve_ivp**($\boldsymbol{F}$, $[t_0, \boldsymbol{T_{final}}]$, $[\boldsymbol{y}(\boldsymbol{t_0}), \boldsymbol{y}'(\boldsymbol{t_0})]$)

Cette fonction retourne des données encapsulées sous la forme de deux vecteurs : ***t*** *et* ***y***

y *doit être transposé pour pouvoir être affiché par rapport à* ***t***

Document rédigé par Julien VILLEMEJANE
LEnsE / Institut d'Optique / France

http://lense.institutoptique.fr/

Création : Avril 2023

Bibliographie

- ***Intégration des équations différentielles : méthode d'Euler*** – *Frédéric LEGRAND*
  https://www.f-legrand.fr/scidoc/srcdoc/numerique/euler/eulers/eulers-pdf.pdf

- ***Introduction à la méthode d'Euler en python*** – *Physique TSI1 Troyes*
  https://www.youtube.com/watch?v=-d7qrNkPDtQ

- ***Cours d'introduction à l'analyse numérique*** – *Femto-Physique*
  https://femto-physique.fr/analyse-numerique/euler.php