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

Calcul symbolique

(Sympy)

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

Cas des équations différentielles

Trucs et Astuces

- Affichage propre type Latex

```
from IPython.display import *

display(expression)
```

Intéressant avec Sympy

Déjà intégré dans Jupyter

display	$-x(t) - \sin(t) + \frac{d^2}{dt^2}x(t)$
print	`-x(t) - sin(t) + Derivative(x(t), (t, 2))`

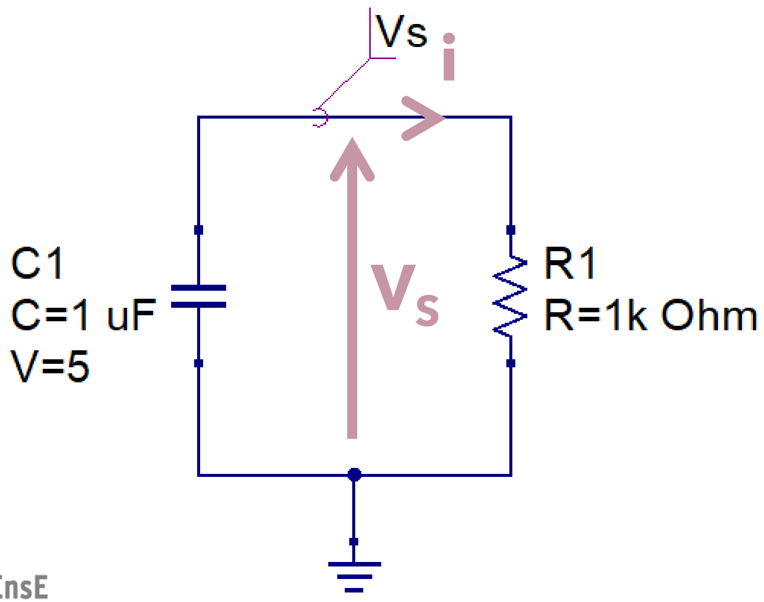
Trucs et Astuces

- Autres paramètres / TO DO

  - Onglet IPython Console => Options > Special consoles > New Sympy Console
  - Onglet Plot => Options > Décocher Mute InLine Plotting

Calcul symbolique (ou formel)

- Approche analytique



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

Equation différentielle d'ordre 1

dont une solution est

$$V_s = K . e^{-a\,t}$$

Calcul symbolique (ou formel)

- Calcul formel

> Le **calcul formel**, ou parfois **calcul symbolique**, est le domaine des mathématiques et de l'informatique qui s'intéresse aux algorithmes opérant sur des objets de nature mathématique par le biais de représentations finies et exactes.

Wikipedia / Calcul formel

SymPy

SymPy pour le calcul formel

- Premier exemple

```
import math
math.sqrt(9)
math.sqrt(8)
```

???

```
import sympy
math.sqrt(9)
math.sqrt(8)
```

???

SymPy pour le calcul formel

- Premier exemple

```
import math
math.sqrt(9)
math.sqrt(8)
```

```
3.0
2.828427…
```

```
import sympy
math.sqrt(9)
math.sqrt(8)
```

3

$2\sqrt{2}$

SymPy pour le calcul formel

- Premier exemple

```
import math
math.sqrt(9)
math.sqrt(8)
```

```
3.0
2.828427…
```

```
m = 3/2                    print(m)
```

```
???
```

```
import sympy
math.sqrt(9)
math.sqrt(8)
```

```
3
```

$2\sqrt{2}$

```
k = sympy.Rational(3,2)          print(k)
```

```
???
```

INSTITUT d'OPTIQUE GRADUATE SCHOOL ParisTech

LEnsE Laboratoire d'Enseignement Expérimental

SymPy pour le calcul formel

spyder jupyter

- Premier exemple

```
import math
math.sqrt(9)
math.sqrt(8)
```

```
3.0
2.828427…
```

```
m = 3/2                    print(m)
```

```
1.5
```

```
import sympy
math.sqrt(9)
math.sqrt(8)
```

```
3
2√2
```

```
k = sympy.Rational(3,2)            print(k)
```

```
3/2
```

SymPy pour le calcul formel

- Expressions

```
x, y = sympy.symbols('x y')
expr = x**2 - 4 * x + 5
expr
```

```
???
```

```
expr.subs(x, 1)
```

```
???
```

SymPy pour le calcul formel

- Expressions

```
x, y = sympy.symbols('x y')
expr = x**2 - 4 * x + 5
expr
```

$x^2 - 4x + 5$

```
x*expr
```

???

```
expand_exp = sympy.expand(x*expr)
expand_exp
```

???

```
factor_exp = sympy.factor(expand_exp)
factor_exp
```

???

SymPy pour le calcul formel

- Expressions

```
x, y = sympy.symbols('x y')
expr = x**2 - 4 * x + 5
expr
```

$x^2 - 4x + 5$

```
x*expr
```

$x(x^2 - 4x + 5)$

```
expand_exp = sympy.expand(x*expr)
expand_exp
```

$x^3 - 4x^2 + 5x$

```
factor_exp = sympy.factor(expand_exp)
factor_exp
```

$x(x^2 - 4x + 5)$

SymPy pour le calcul formel

- Déclarer des fonctions (au sens mathématique)

```
f = sympy.Function('f')
f = x**2 + y
f
```

$x^2 + y$

```
f = sympy.Function('f')(x, y)
```

```
f.subs(x, 4)
f.subs(y,1)
```

```
f.subs({x:1, y: 2})
```

SymPy pour le calcul formel

- Vectoriser une fonction

```
g = sympy.Function('g')(x)
g = sympy.sin(x/2 + sympy.sin(x))
```

```
xlin = np.linspace(-np.pi, np.pi, 21)

result = g.subs(x, xlin)
display(result)
```

```
???
```

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

SymPy pour le calcul formel

- Vectoriser une fonction - ***lambdify***

```
g = sympy.Function('g')(x)
g = sympy.sin(x/2 + sympy.sin(x))
```

```
xlin = np.linspace(-np.pi, np.pi, 21)

result = g.subs(x, xlin)
display(result)
```

???

```
from sympy.utilities.lambdify import lambdify
func = lambdify( [x] , g)

yres = func( xlin )
```

???

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

SymPy pour le calcul formel

- Déclarer une expression avec des dérivées

```
t, tau = sympy.symbols('t tau')
vs = sympy.Function('V_s')(t)
```

```
dvs = sympy.Derivative( vs , t )
exp = vs + tau * dvs
display( exp )
```

```
???
```

SymPy pour le calcul formel

- Résoudre une équation différentielle

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

sympy.**dsolve**(equation, fonction, ics=cond_init)

fonction = vs(t)

init_conds = {vs.subs(t,0): 5}

```
exp = vs + tau * dvs
```

```
init_conds = {vs.subs(t,0): 5}
result = sp.dsolve(exp, vs, ics=init_conds)
vs_t = result.rhs
```

+ Afficher *vs_t* **avec** et **sans conditions initiales**
+ De quel type d'objet s'agit-il ?

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

Calcul symbolique (ou formel)

S'ENTRAINER

- Résolution formelle

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

Equation différentielle d'ordre 1

+ Donner la solution analytique

+ Tracer la solution en fonction du temps pour R = 100kΩ et C = 1 µF

sympy.**dsolve**(equation, fonction, ics=cond_init)

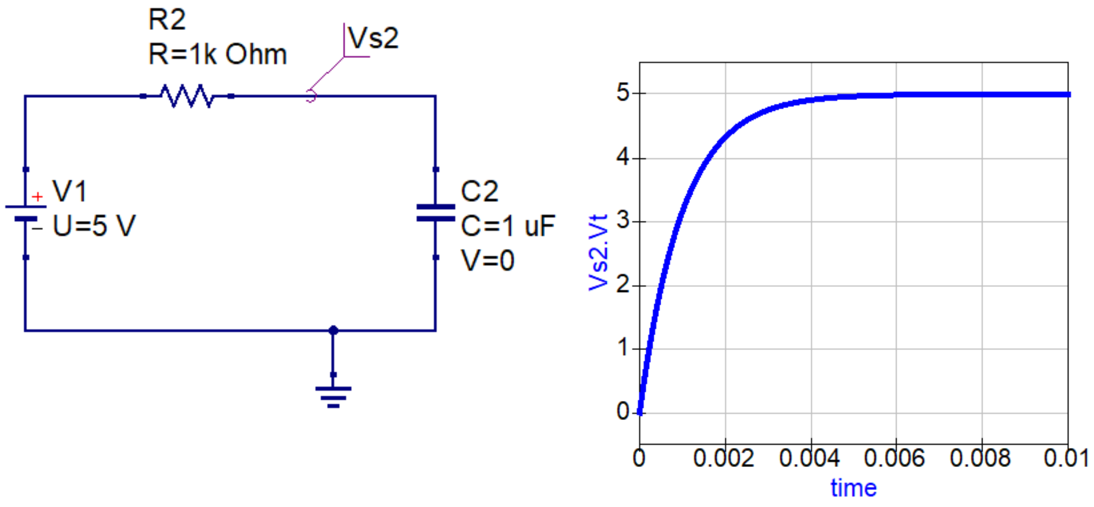
fonction = vs(t)

init_conds = {vs.subs(t,0): 5}

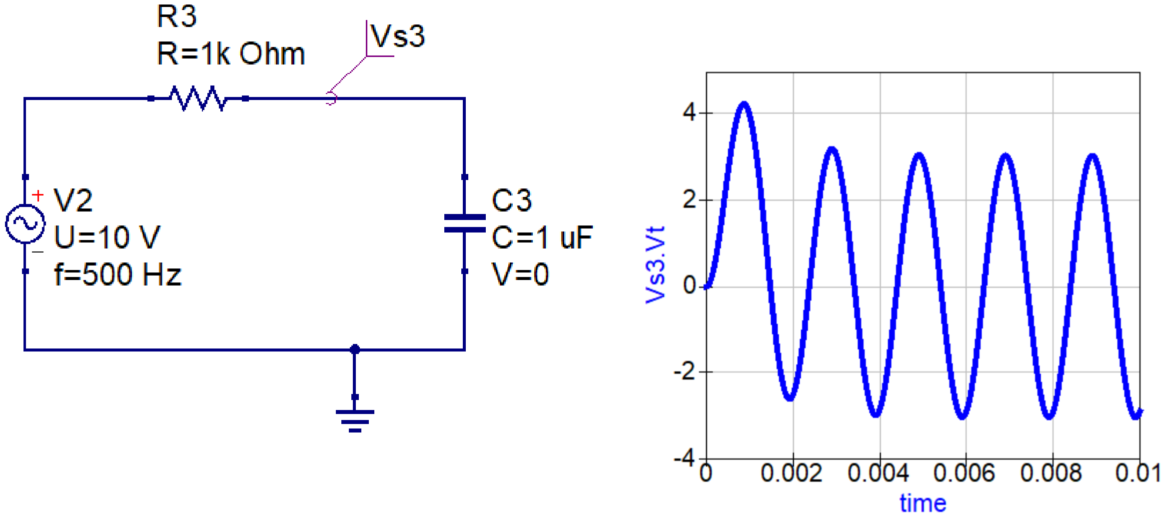
sympy.**lambdify**([params], fonction)

Circuits similaires / Généralisation

- Réponse à un échelon



- Régime forcé



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

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

Calcul symbolique (ou formel)

S'ENTRAINER

- Résolution formelle

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

Equation différentielle d'ordre 1

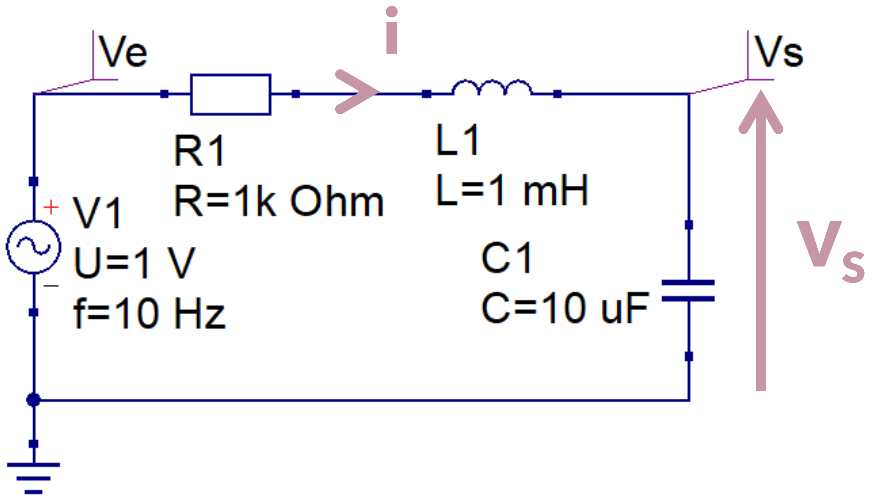
+ Donner la solution analytique à la réponse à un signal sinusoïdal de fréquence f donnée

+ Tracer la solution en fonction du temps pour R = 100kΩ et C = 1 µF pour un signal sinusoïdal à 10 Hz

Autre cas / Equation du second ordre

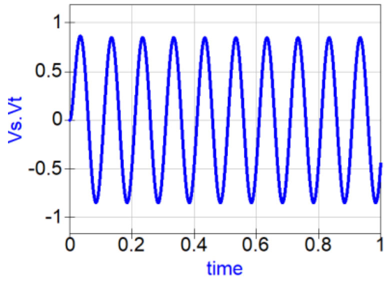
ALLER PLUS LOIN

- 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



+ Donner la solution analytique

+ Tracer la solution en fonction du temps pour R = 1 kΩ, L = 1 mH et C = 1 µF

SymPy pour le calcul formel

- Calculer des limites

```
g = sympy.Function('g')(x)
g = sympy.sin(x/2 + sympy.sin(x))
g
```

```
lg = sympy.limit(g, x, sympy.pi)
lg
```

```
h = 2*sympy.exp(1/x)/(sympy.exp(1/x)+1)
h
lhplus = sympy.limit(h, x, 0, dir='+')
lhplus
```

```
m = (sympy.cos(x)-1)/x
m
lm = sympy.limit(m, x, sympy.oo)
print(f'Limit in +inf = {lm}')
```

SymPy pour le calcul formel

- Caculer des dérivées…

```
f = x**2 + y
f
```

```
dfx = sympy.diff(f, x)
dfx
```

```
dfy = sympy.diff(f, y)
dfy
```

- …ou des intégrales

```
inte_f = sympy.integrate(f, x)
inte_f
```

```
f = sympy.exp(x)/
(sympy.sqrt(sympy.exp(2*x)+9))

inte_f = sympy.integrate(f, (x, 0,
sympy.log(4)))
inte_f
```

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

http://lense.institutoptique.fr/

Création : Avril 2023

Bibliographie

- ***Python pour le calcul symbolique***– *WikiBooks*
  https://fr.wikibooks.org/wiki/Python_pour_le_calcul_scientifique/Calcul_symbolique

- ***Ordinary Differential Equations - SymPy Tutorial 10*** – *TM Quest*
  https://www.youtube.com/watch?v=Z2havWsxa-E

- ***Le calcul symbolique et ses principales applications*** – *Paul LEVY*
  http://www.numdam.org/article/AUG_1945__21__41_0.pdf

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