1 Fichiers CSV

1.1 Enregistrement des données dans un fichier CSV

1.1.1 Exemple 1

```
import csv # le module pour les fichiers csv

en_tetes = ['Composant','Valeur','Unité','Courant','Tension']
lignes = [('R', 1, 'kOhms', 0.02, 1.5),
          ('C', 5, 'uF', 0.005, 0.55),
          ('L', 2, 'mH', 0.001, 0.95),
          ]
with open('csv/MonFichier.csv','w') as f:
    f_csv = csv.writer(f)
    f_csv.writerow(en_tetes)
    f_csv.writerows(lignes)
```

Le fichier csv créé doit ressembler au tableau suivant :

Composant	Valeur	Unité	Courant	Tension
R	1	kOhms	0.02	1.5
C	5	uF	0.005	0.55
L	2	mH	0.001	0.95

1.1.2 Exemple 2

```
import csv

def ecritureCSV(fichier, sep, colonne1, colonne2):
    """
    fichier     <str> : Nom du fichier CSV à créer -> "MonFichier2.csv"
    sep         <str> : Séparateur de colonnes -> ";" pour notre cas
    colonne1    <int> : Première colonne
    colonne2    <int> : Deuxième colonne
    """
    with open('csv/MonFichier2.csv','w') as f2:
        ecriture = csv.writer(f2, delimiter = sep)
        taille1, taille2 = len(colonne1), len(colonne2)
        if taille1 == taille2:
            ecriture.writerow(['t', 'f'])
            for i in range(taille1):
                ecriture.writerow((colonne1[i], colonne2[i]))
        else:
            print("Les tailles des listes sont différentes")
        f2.close()

# Declaration des listes
t = [0, 1, 2, 3, 4, 5]
f = [0.1, 0.4, 0.6, 0.9, 0.7, 0.5]
# Exécution de la fonction
ecritureCSV("MonFichier2.csv", ";", t, f)
```

Le fichier csv créé doit ressembler au tableau suivant :

t	f
0	0.1
1	0.4
2	0.6
3	0.9
4	0.7
5	0.5

1.1.3 Exemple 3

L'exemple ci-dessous montre comment tracer des signaux sinusoïdaux sur une période et sauvegarder les résultats dans un même fichier CSV.

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
'''
* sin et cos sur une période
* Sauvegarde dans un même fichier CSV
'''

a = 2.      # Amplitude des signaux
f = 5.      # Fréquence des signaux
fe = 500.   # Fréquence d'échantillonnage

t = np.arange(start=0,stop=1/f,step=1/fe)
S1 = a*np.sin(2.0*np.pi*f*t)
S2 = a*np.cos(2.0*np.pi*f*t)

# Courbes
plt.figure(figsize=(8,4))
plt.plot(t,S1,label='sin')
plt.plot(t,S2,label='cos')
plt.xlabel('t [s]')
plt.ylabel('S(t)')
plt.legend()
plt.title('Signaux sinusoïdaux')
plt.grid()
#plt.show()

# Données
data = np.zeros((len(t),3))
data[:,0] = t
data[:,1] = S1
data[:,2] = S2

# Sauvegarde
name="csv/SinCos.csv"
np.savetxt(name,data,delimiter=",",header="t,sinus,cosinus",comments="")
```

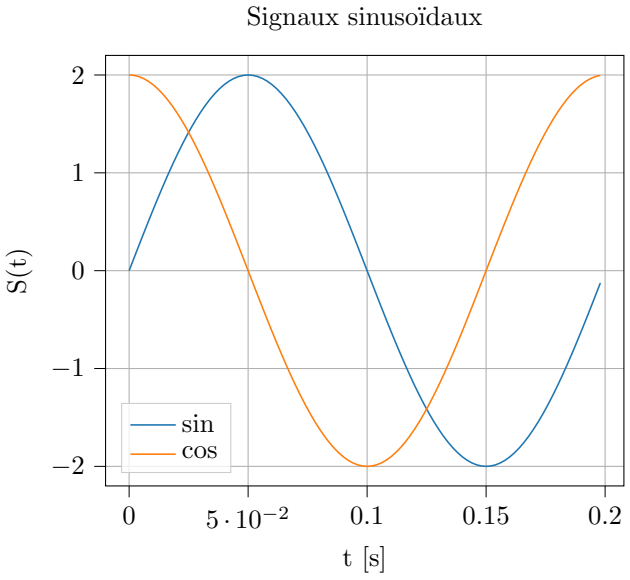


FIGURE 1 – Résultats : Courbes

Pour visualiser le fichier CSV, cliquer ici

1.2 Importation et lecture d'un fichier CSV

1.2.1 Lecture du fichier créé dans l'exemple 1

```
import csv
file = open("csv/MonFichier.csv","r")       # ouvrir le fichier
reader = csv.reader(file, delimiter = ",") # initialisation d'un lecteur de fichier
for row in reader : # parcours du lecteur avec une boucle
    print(row)      # affichage ligne par ligne
file.close()        # fermeture du fichier
```

```
['Composant', 'Valeur', 'Unité', 'Courant', 'Tesnion']
['R',         '1',      'kOhms', '0.02',    '1.5']
['C',         '5',      'uF',    '0.005',   '0.55']
['L',         '2',      'mH',    '0.001',   '0.95']
```

1.2.2 Lecture du fichier créé dans l'exemple 2

```
import csv
file = open("csv/MonFichier2.csv","r")      # ouvrir le fichier
reader = csv.reader(file, delimiter = ";") # initialisation d'un lecteur de fichier
for row in reader : # parcours du lecteur avec une boucle
    print(row)      # affichage ligne par ligne
file.close()        # fermeture du fichier
```

['t', 'f']
['0', '0.1']
['1', '0.4']
['2', '0.6']
['3', '0.9']
['4', '0.7']
['5', '0.5']

1.3 Importation et exploitation des données récupérées d'un fichier CSV

Remarque :

Ces données sont destinées à être traitées et tracées avec Python, mais :

- On remarque que chaque ligne du fichier CSV est placée dans une liste Python et que les valeurs d'une même grandeur (Tension par exemple) n'appartiennent pas à la même liste.
- On remarque également que toutes les valeurs sont considérées comme des chaînes de caractères.

Il est alors nécessaire d'écrire une fonction de lecture des fichiers CSV un peu plus évoluée afin de tenir compte de ces remarques.

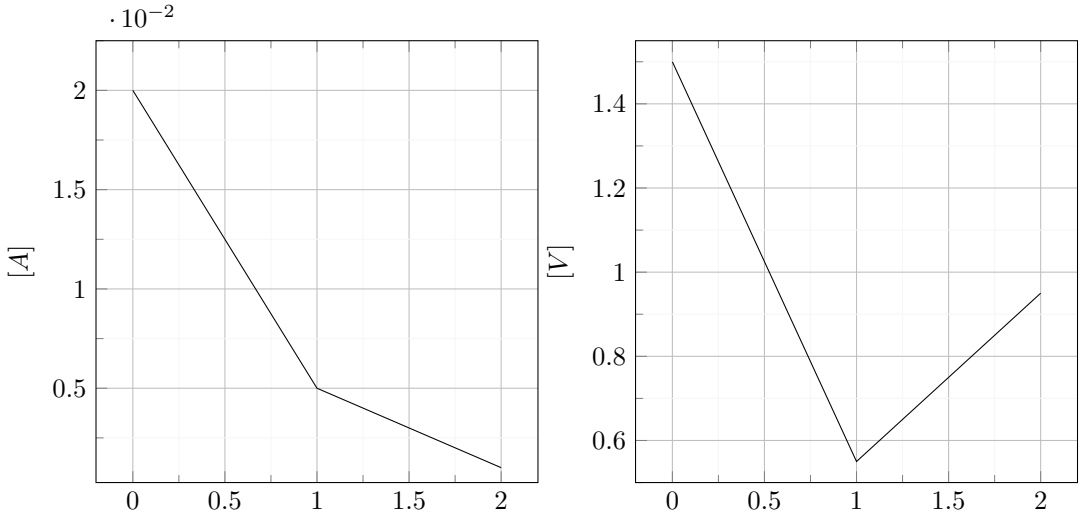
1.3.1 Exemple 3.1

```
import csv

def lectureColCSV(fichier, sep, n) :
    """ Pour les paramètres fichier et sep il faut les écrire entre les
    guillements car la fonction attend des chaines de caractères.
    fichier <str> : Le nom du fichier -> "MonFichier.csv"
    sep     <str> : Le séparateur de colonnes -> "," pour notre cas
    n       <int> : Le numéro de la colonne à lire
    """
    file = open( fichier, "r")
    reader = csv.reader(file, delimiter = sep)
    col = []
    for row in reader:
        """Dans notre cas le séparateur décimal est un "."
        Cette boucle n'est pas nécessaire
        Elle est nécessaire dans le cas où le séparateur décimal est "," """
        try:
            sep_decimal = row[n].replace(",", ".")
            col.append(float(sep_decimal))
        except:
            pass
    file.close()
    return col

# On récupère les deux dernières colonnes (4ème et 5ème) du fichier
# Rappel : Python commence l'indexation à 0
x = lectureColCSV("csv/MonFichier.csv", ",", 3)
y = lectureColCSV("csv/MonFichier.csv", ",", 4)
print("Courant = ", x)
print("Tension = ", y)
```

```
Courant =  [0.02, 0.005, 0.001]
Tension =  [1.5, 0.55, 0.95]
```

1.3.2 Exemple 3.2

```
import matplotlib.pyplot as plt
%matplotlib inline
plt.subplot(1, 2, 1)
plt.plot(x, '-', lw=2)

plt.ylabel('[A]')
plt.title('Courant')
plt.grid(True)


plt.subplot(1, 2, 2)
plt.plot(y, '-', lw=2)

plt.title('Tension')
plt.ylabel('[V]')
plt.grid(True)

plt.tight_layout()
plt.show()
```



1.3.3 Exemple 4

```
import csv

def lectureColCSV(fichier, sep, n) :
    """
    fichier <str> : Le nom du fichier -> "SinCos.csv"
    sep     <str> : Le séparateur de colonnes -> ","
    n       <int> : Le numéro de la colonne à lire
    """
    file = open( fichier, "r")
    reader = csv.reader(file, delimiter = sep)
    col = []
    for row in reader:
        try:
            sep_decimal = row[n].replace(",", ".")
            col.append(float(sep_decimal))
        except:
            pass
    file.close()
    return col

temps = lectureColCSV("csv/SinCos.csv", ",", 0)
x = lectureColCSV("csv/SinCos.csv", ",", 1)
y = lectureColCSV("csv/SinCos.csv", ",", 2)

# Courbes
plt.figure(figsize=(8,4))
plt.plot(temps,x,label='sin')
plt.plot(temps,y,label='cos')
plt.xlabel('t [s]')
plt.ylabel('S(t)')
plt.legend()
plt.title('Signaux sinusoïdaux')
plt.grid()
plt.show()
```

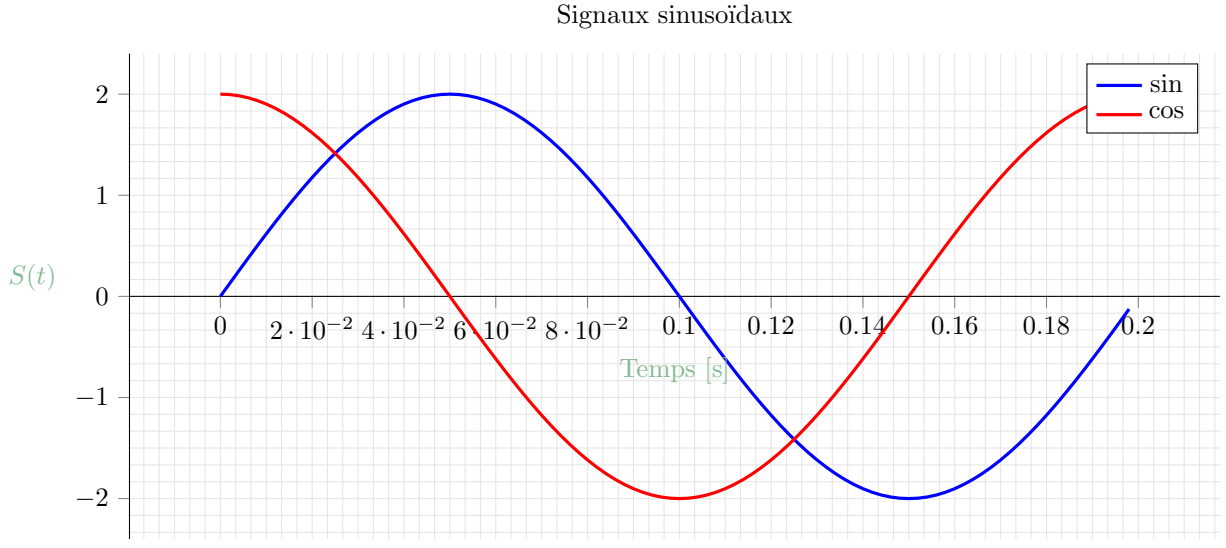


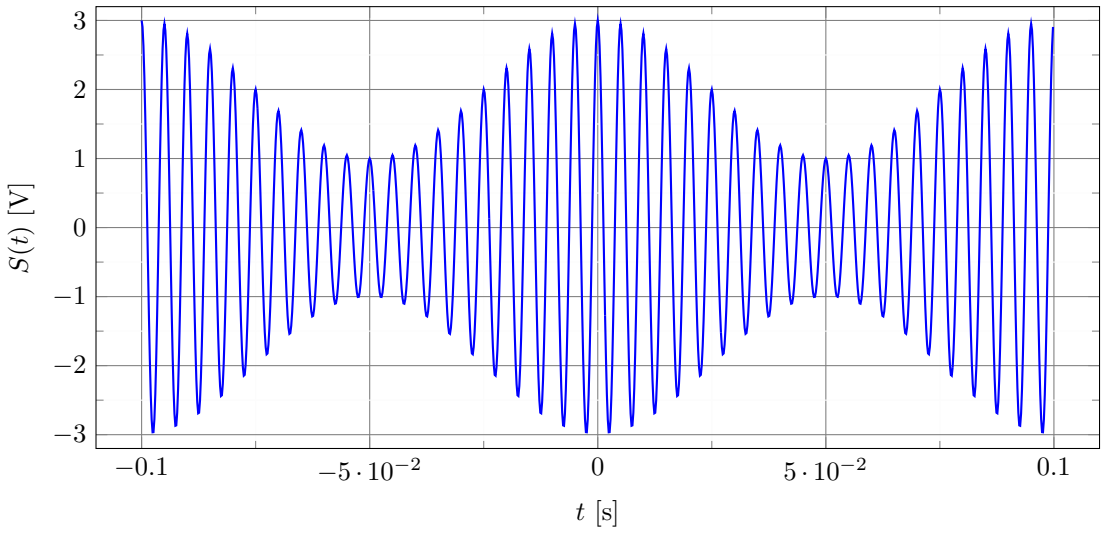
FIGURE 2 – Résultats : Courbes

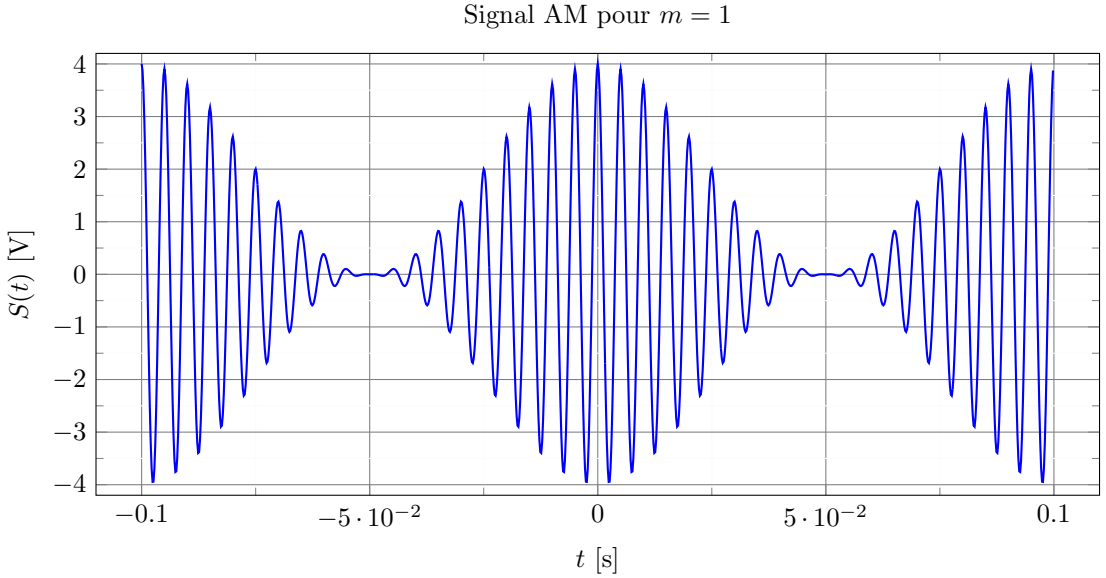
1.4 Enregistrement des données dans plusieurs fichiers CSV

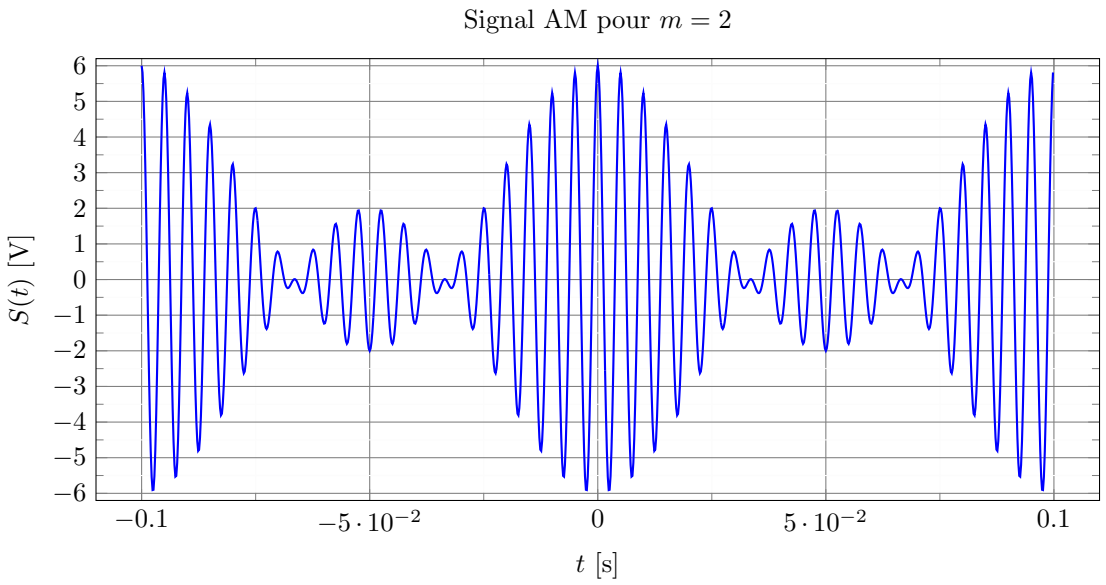
Le programme exporte les données vers trois fichiers CSV (`AM_0.csv, AM_1.csv et Am_2.csv`).

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

'''
Modulation d'amplitude pour différentes valeurs de l'indice de modulation 'm'
'''

m = [0.5, 1, 2] # Indice de modulation
fm = 10         # Fréquence du message (signal modulant)
fp = 200        # Fréquence de la porteuse
Te = 0.0002     # Période d'échantillonnage

t = np.arange(start=-.1,stop=.1,step=Te)

for i in range(len(m)):
    S = 2*(1+m[i]*np.cos(2.0*np.pi*fm*t))*np.cos(2.0*np.pi*fp*t)

    # Courbes
    plt.figure(figsize=(8,4))
    plt.plot(t,S)
    plt.xlabel('t [s]')
    plt.ylabel('S(t)')
    plt.title('Signal AM pour m = {}'.format(m[i]))
    plt.grid()

    # Données
    data = np.zeros((len(t),2))
    data[:,0] = t
    data[:,1] = S

    # Sauvegarde
    name="csv/AM_{}.csv".format(i)
    np.savetxt(name,data,delimiter=",",header="t,S",comments="")

plt.show()
```

Signal AM pour $m = 0.5$




Signal AM pour $m = 1$
$S(t)$ [V]
4
3
2
1
0
−1
−2
−3
−4
−0.1
$-5 \cdot 10^{-2}$
0
$5 \cdot 10^{-2}$
0.1
t [s]


Signal AM pour $m = 2$
$S(t)$ [V]
6
5
4
3
2
1
0
−1
−2
−3
−4
−5
−6
−0.1
$-5 \cdot 10^{-2}$
0
$5 \cdot 10^{-2}$
0.1
t [s]

2 Fichiers TXT

2.1 Importation et lecture d'un fichier texte (.txt)

```
import matplotlib.pyplot as plt
import numpy as np
%matplotlib inline

f = open("txt/Donnees.txt" , 'r')
f.readline()
data = np.loadtxt(f)
f.close()

plt.figure()
plt.plot(data[:,0],data[:,1],"r-")
plt.plot(data[:,0],data[:,2],"b-")
plt.grid()
plt.show()
```

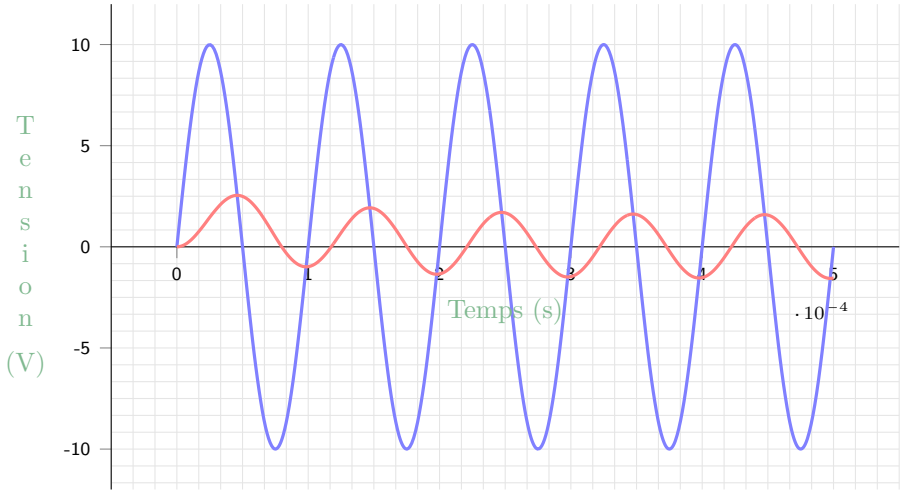
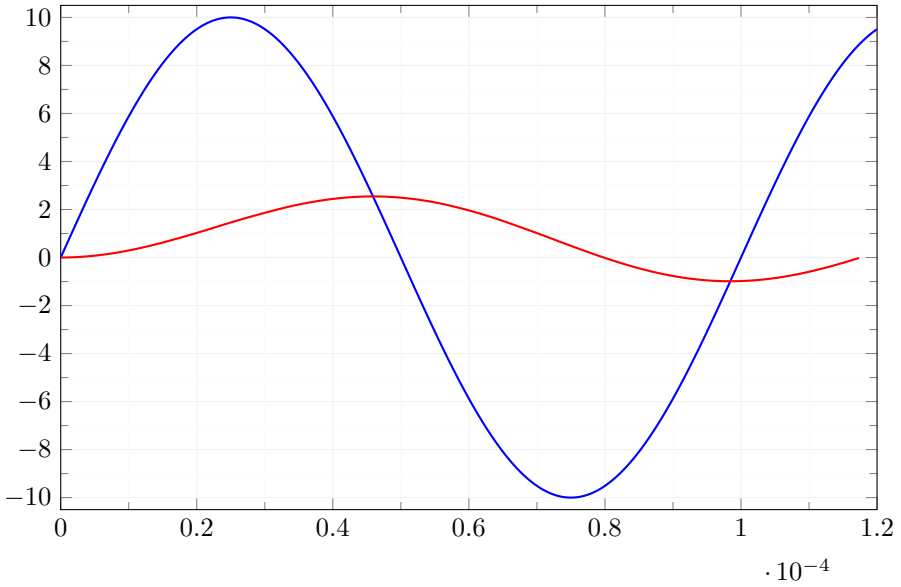


FIGURE 3 – Courbes des données

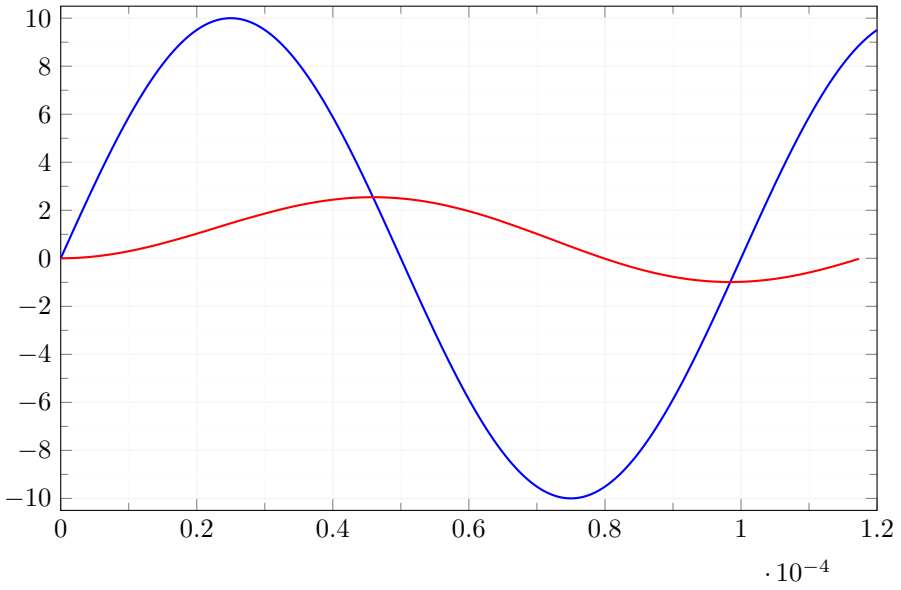
```
Fichier = open('txt/LtPy.txt','r')
# lecture dans le fichier avec la méthode read()
chaine = Fichier.read()
# affichage du contenu du fichier
print('Contenu du fichier :\n' + chaine)
# fermeture du fichier avec la méthode close()
Fichier.close()
```

```
Contenu du fichier :
time    V(cap)  V(source)
0.000000000000000e+00   0.000000e+00    0.000000e+00
9.142857087291697e-08   3.759883e-05    5.744954e-02
1.828571417458339e-07   1.353809e-04    1.148896e-01
2.742857126187509e-07   2.933463e-04    1.723202e-01
3.657142834916679e-07   5.114950e-04    2.297414e-01
4.571428543645849e-07   7.898270e-04    2.871531e-01
           ⋮                ⋮               ⋮
1.168509374961105e-04   -6.253866e-02   8.715511e+00
1.173392187461105e-04   -1.942553e-02   8.861663e+00
```

```
import matplotlib.pyplot as plt
import numpy as np
%matplotlib inline
# Avec diese à l'entête (voir fichier créé)
data = np.loadtxt("txt/LtPy2.txt")
plt.plot(data[:,0],data[:,1],"r-")
plt.plot(data[:,0],data[:,2],"b-")
plt.show()
```



```
import matplotlib.pyplot as plt
import numpy as np
%matplotlib inline
# Sans diese à l'entête (voir fichier créé)
f = open("txt/LtPy.txt" , 'r')
f.readline() # lit la premiere ligne
data = np.loadtxt(f)
f.close()
plt.plot(data[:,0],data[:,1],"r-")
plt.plot(data[:,0],data[:,2],"b-")
plt.show()
```



3 Traitement de données

3.1 Exploitation des données fournies par un capteur

Un programme Arduino récupère les données d'un capteur sous la forme suivante :

```
a1 = ['35\r\n','30\r\n','61\r\n','10\r\n','7\r\n','65\r\n','21\r\n','4\r\n','59\r\n']

type(a1)
```

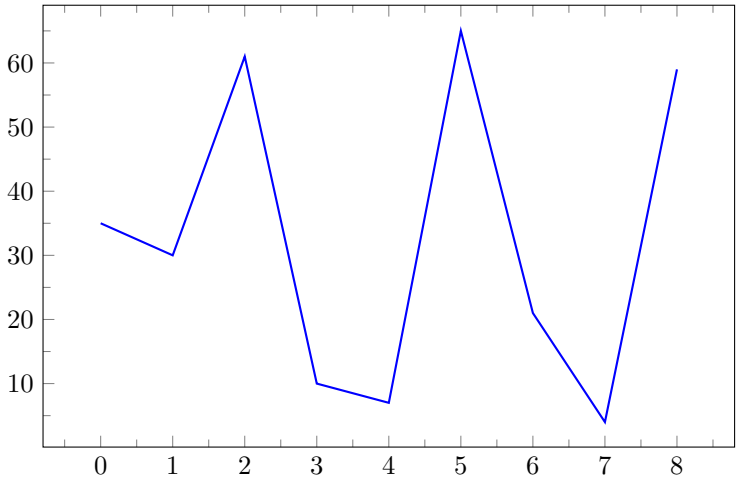
```
list
```

On note que c'est une liste de chaîne de caractères. On ne doit garder que les nombres entiers.

```
x = []
for elt in a1: # elt va prendre les valeurs successives des éléments de la liste
    x.append(int(elt))
    print(elt)
print('x=',x)
```

```
35
30
61
10
7
65
21
4
59
x= [35, 30, 61, 10, 7, 65, 21, 4, 59]
```

```
plt.plot(x);
```




```
a1 = '35\r\n','30\r\n','61\r\n','10\r\n','7\r\n','65\r\n','21\r\n','4\r\n','59\r\n'

type(a2)
```

```
tuple
```

```
a1 = ['35\r\n','30\r\n','61\r\n','10\r\n','7\r\n','65\r\n','21\r\n','4\r\n','59\r\n']
a3 = str(a1)
z1 = a3.replace('"', '')
z2 = z1.replace('\\r', '')
z3 = z2.replace('\\n', '')
print(z3)
```

```
['35', '30', '61', '10', '7', '65', '21', '4', '59']
```

4 Exercices

Exercice 1

Écrire un programme pour enregistrer les données suivantes dans un fichier `CSV`.

x	$2x$	x^2
0.00000000000000000e+00	0.00000000000000000e+00	0.00000000000000000e+00
1.00000000000000000e+00	2.00000000000000000e+00	1.00000000000000000e+00
2.00000000000000000e+00	4.00000000000000000e+00	4.00000000000000000e+00
3.00000000000000000e+00	6.00000000000000000e+00	9.00000000000000000e+00
4.00000000000000000e+00	8.00000000000000000e+00	1.60000000000000000e+01
5.00000000000000000e+00	1.00000000000000000e+01	2.50000000000000000e+01

Table 1: Exercice `1`

Exercice 2

Écrire un programme pour enregistrer les données suivantes dans un fichier `CSV`.

x	e^x
0.00000000000000000e+00	1.00000000000000000e+00
1.00000000000000056e-01	1.10517091807564771e+00
2.00000000000000111e-01	1.22140275816016985e+00
3.00000000000000444e-01	1.34985880757600318e+00
4.00000000000000222e-01	1.49182469764127035e+00
5.00000000000000000e-01	1.64872127070012819e+00
6.00000000000000888e-01	1.82211880039050911e+00
7.00000000000000666e-01	2.01375270747047663e+00
8.00000000000000444e-01	2.22554092849246787e+00
9.00000000000000222e-01	2.45960311115694985e+00
1.00000000000000000e+00	2.71828182845904509e+00
1.10000000000000089e+00	3.00416602394643339e+00
1.20000000000000178e+00	3.32011692273654813e+00
1.30000000000000044e+00	3.66929666761924444e+00
1.40000000000000133e+00	4.05519996684467543e+00
1.50000000000000000e+00	4.48168907033806452e+00
1.60000000000000089e+00	4.95303242439511493e+00
1.70000000000000178e+00	5.47394739172720079e+00
1.80000000000000044e+00	6.04964746441294654e+00
1.90000000000000133e+00	6.68589444227927032e+00
2.00000000000000000e+00	7.38905609893065041e+00 ;

Table 2: Exercice `2`

Exercice 3

Écrire un programme pour créer les fichiers CSV suivants :

x	$2x$
0.00000000000000000e+00	0.00000000000000000e+00
1.00000000000000000e+00	2.00000000000000000e+00
2.00000000000000000e+00	4.00000000000000000e+00
3.00000000000000000e+00	6.00000000000000000e+00
4.00000000000000000e+00	8.00000000000000000e+00
5.00000000000000000e+00	1.00000000000000000e+01 ;

Table 3: Exercice 3 : Table_1.csv

x	x^2
0.00000000000000000e+00	0.00000000000000000e+00
1.00000000000000000e+00	1.00000000000000000e+00
2.00000000000000000e+00	4.00000000000000000e+00
3.00000000000000000e+00	9.00000000000000000e+00
4.00000000000000000e+00	1.60000000000000000e+01
5.00000000000000000e+00	2.50000000000000000e+01 ;

Table 4: Exercice 3 : Table_2.csv

Exercice 4

— Écrire un programme pour importer et tracer les données du fichier `Table_1.csv`
— Écrire un programme pour importer et tracer les données du fichier `Table_2.csv`

Exercice 5

— Écrire un programme pour importer les données du fichier Exercice5.csv
— Dans des graphes différents, tracer les données de chaque colonne en fonction de la première colonne.

Exercice 6

— Ecrire un programme pour créer le tableau csv suivant :

Titre 1	**Titre 2**	**Titre 3**
Mot 1	0.1	4
Mot 2	0.2	10
Mot 3	0.3	5
Mot 4	0.6	7
Mot 5	0.8	8

— Importer et lire le fichier créé, puis tracer la colonne "Titre 3" en fonction de la colonne "Titre 2"

Exercice 7

— Ecrire un programme pour lire le fichier `Donnees.csv` fourni en pièce jointe
— Afficher le contenu de ce fichier
— Tracer dans la même figure "G1" en fonction de "w" et "P1" en fonction de "w" (utiliser "subplot")
— Tracer dans la même figure "G2" en fonction de "w" et "P2" en fonction de "w"
— Tracer dans la même figure "G" en fonction de "w" et "P" en fonction de "w"

Exercice 8

— Ecrire un programme pour tracer la fonction suivante : $y(t) = 2\sin(\omega t)$ avec $\omega = 10$ et $t \in [0, 1]$.
— Enregistrer les valeurs de t et de y dans un même fichier csv.