Beer-Lambert smartphone

Philippe Larcher
larcher.philippe@gmail.com

Lycée Sud-Médoc
Le Taillan-Médoc - Gironde - FRANCE

April 11, 2019

1 Utilisation de 2 smartphones pour analyser la composition d'une solution - Loi de Beer-Lambert

On cherche à déterminer la concentration en masse C_m d'une solution coloréee.

1.1 Pré-requis :

- Connaitre la définition de la concentration en masse C_m ;
- Savoir réaliser une solution par dilution ;
- Connaitre la "formule de la dilution" : $C_m \times V_m = C_f \times V_f$;
- Savoir qu'une image est un ensemble de pixels composés chacun de 3 sous-pixels, codé chacun sur 8 *bits*, correspondant à 2^8 soit 256 teintes comprises entre 0 et 255 pour un total de $\left(2^8\right)^3$ couleurs par pixel.

1.2 Capacités mises en œuvre :

- Savoir réaliser une dilution ;
- Savoir concevoir et réaliser un protocole expérimental ;
- Savoir exécuter un programme Python pour tracer et exploiter une courbe d'étalonnage ;
- Comprendre et exploiter les composantes RGB d'un signal lumineux.

1.3 Matériel par groupe :

- 1 solution mère de sulfate de cuivre (Cu^{2+}, SO_4^{2-}) de concentration en masse $C_m = 50\ g \cdot L^{-1}$;
- 1 solution mère de permanganate de potassium (K^+, SO_4^{2-}) de concentrtion en masse $C_m = 20\ mg \cdot L^{-1}$;
- 3 pipettes jaugées de 5, 10 et 20 mL ;
- 1 fiole jaugée de 50 mL ;
- 1 ratelier de 8 tubes à essai ;

- 2 béchers de 50 mL ;
- 2 smartphones avec les applications :
  - pour émettre une lumière colorée :
    - **Physics Toolbox Suite** sur Androïd : https://play.google.com/store/apps/details?id=com.chrystianvieyra.physicstoolboxsuite&hl=en
    - **Physics Toolbox Color Gen** sur iOS : https://itunes.apple.com/us/app/physics-toolbox-color-gen/id1372521401
  - pour lire les composante RGB en un point :
    - **Color Grab** sur Androïd : https://play.google.com/store/apps/details?id=com.loomatix.colorgrab&hl=fr
    - **Color Assist** sur iOS : https://itunes.apple.com/us/app/colour-assist/id1166523317?mt=8

1.4 Marche à suivre :

1.4.1 Réalisation des solutions :

Vous allez travailler par groupes de 4. La solution est une solution de sulfate de cuivre.

Préparez d'abord des solutions de concentrations : 0 - $\frac{C_m}{10}$; $\frac{C_m}{5}$; $\frac{2 \times C_m}{5}$; $\frac{3 \times C_m}{5}$; $\frac{4 \times C_m}{5}$ et C_m et les placer dans 7 tubes à essai.

1.4.2 Préparation des smartphones :

Smartphone émetteur : - vérifier que la luminosité automatique est désactivée ; - vérifier que la composante choisie n'atteint pas la valeur 255 (saturation) pour le tube contenant l'eau (blanc). Adapter la luminosité si besoin.

Smartphone lecteur : - viser une zone sans reflet et bien au centre du tube.

1.4.3 Identification de la couleur de travail :

Placer un bécher contenant la solution mère sur le smartphone posé à plat en mode générateur de couleur et identifier la couleur (R, V ou B) la plus absorbée.

1.4.4 Exécution du programme :

Le programme suivant est un programme en langage Python.

Vous devez exécuter un à un les blocs (grisés) du programme ci-dessous en sélectionnant le bloc puis en cliquant sur l'icône de lecture (▶) ou sur la combinaison de touches : $\Uparrow + \hookleftarrow$

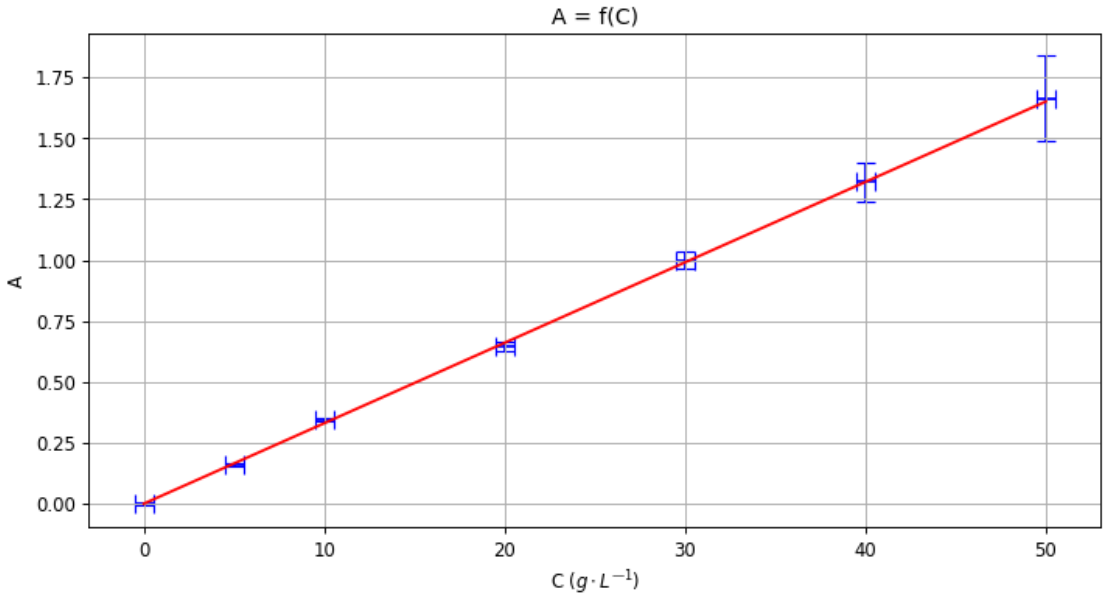
On vous demande d'abord d'entrer : - le nombre de solutions de l'échelle de teinte, y compris l'eau qui servira de référence (blanc) ; - l'incertitude de lecture que vous estimez sur la valeur de la composante RVB choisie ; - les valeurs des différentes concentrations des solutions constituants l'échelle de teinte ; - la valeur de la composante R, V ou B correspondante.

On vous demandera enfin de mesurer la valeur de la composante R, V ou B de la solution inconnue.

In [78]:
```
import numpy as np
import scipy.optimize as spo
import matplotlib.pyplot as plt
%matplotlib inline

C=[]
RGB=[]
N = int(input('Entrer le nombre total de solutions de l'échelle de teinte (eau comprise
U_RGB = float(input("Entrer la précision estimée sur la lecture de la composante RGB :

for i in range(N):
    Ci = np.float(input('Entrer la valeur de Cm en g/L :'))
    RGB_i = np.float(input('Entrer la valeur de la composante choisie du signal RVB :')
    RGB.append(RGB_i)
    C.append(Ci)
U_C=[0.5]*N
U_RGB=[U_RGB]*N

T = []
A = []
U_T = []
U_A = []

for i in range(N):
    Ti = RGB[i]/max(RGB)
    U_Ti = U_RGB[i]/max(RGB)
    Ai = -np.log10(Ti)
    U_Ai = U_Ti/(np.log(10)*Ti)
    T.append(Ti)
    A.append(Ai)
    U_T.append(U_Ti)
    U_A.append(U_Ai)

x = np.array(C)
y = np.array(A)
ux = np.array(U_C)
uy = np.array(U_A)

##########################################################################################

def f(x,p):
    a, b = p
    return a * x + b
def derivx_f(x,p):
    a, b = p
    return a
def residual(p, y, x):
    return (y - f(x,p)) / np.sqrt(uy**2 + (derivx_f(x,p)*ux)**2)
```

```
        p0 = np.array([0,0])
        result = spo.leastsq(residual, p0, args=(y, x), full_output=True)
        popt = result[0];
        pcov = result[1];
        upopt = np.sqrt(np.abs(np.diagonal(pcov)))
        chi2r = np.sum(np.square(residual(popt,y,x))) / (x.size-popt.size)
        plt.figure(num=None, figsize=(10,5), dpi=84, facecolor='w', edgecolor='k')
        plt.errorbar(x,y, xerr = ux, yerr = uy, fmt = 'none', capsize = 5, ecolor = 'blue', zor
        a, b = popt
        ua, ub = upopt
        y_mod = a * x + b

        ##############################################################################################

        plt.plot(x,y_mod,'r-')
        plt.title("A = f(C)")
        plt.xlabel('C ($g\cdot L^{-1}$)')
        plt.ylabel('A')
        plt.grid()
        plt.show()
        print(f"L'équation de la droite modélisant A = f(C) est : A = ({a:1.3f} ± {ua:1.3f}) x
```

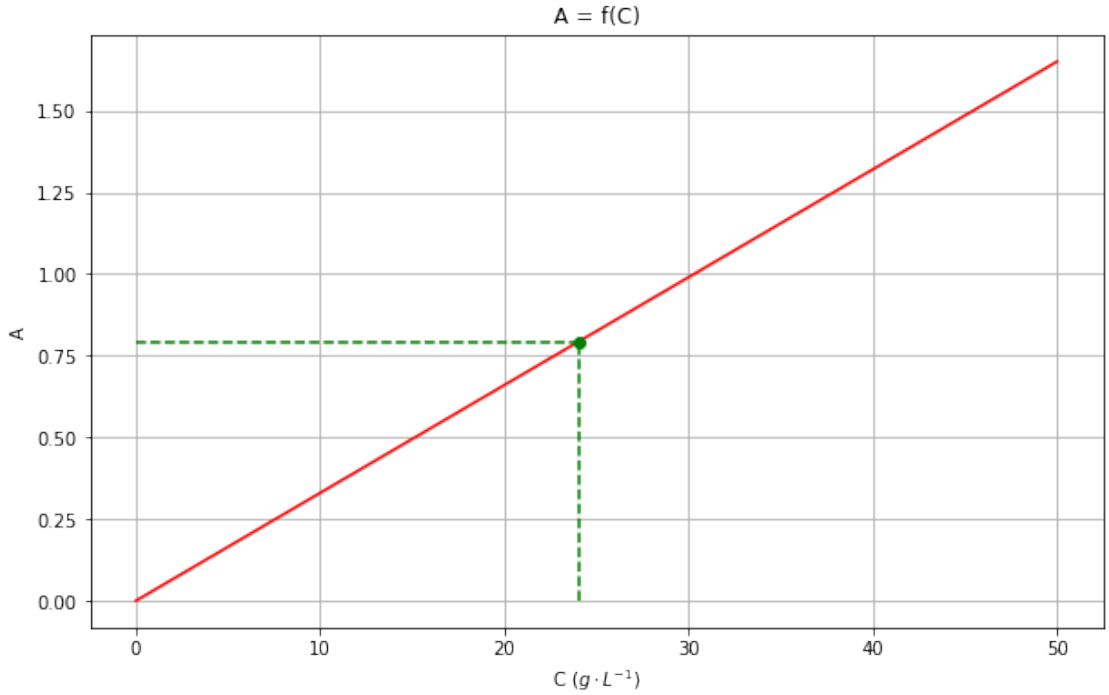
```
Entrer le nombre total de solutions de l'échelle de teinte (eau comprise !) :  7
Entrer la précision estimée sur la lecture de la composante RGB :  2
Entrer la valeur de Cm en g/L : 0
Entrer la valeur de la composante choisie du signal RVB : 230
Entrer la valeur de Cm en g/L : 5
Entrer la valeur de la composante choisie du signal RVB : 159
Entrer la valeur de Cm en g/L : 10
Entrer la valeur de la composante choisie du signal RVB : 104
Entrer la valeur de Cm en g/L : 20
Entrer la valeur de la composante choisie du signal RVB : 52
Entrer la valeur de Cm en g/L : 30
Entrer la valeur de la composante choisie du signal RVB : 23
Entrer la valeur de Cm en g/L : 40
Entrer la valeur de la composante choisie du signal RVB : 11
Entrer la valeur de Cm en g/L : 50
Entrer la valeur de la composante choisie du signal RVB : 5
```



L'équation de la droite modélisant A = f(C) est : A = (0.033 ± 0.001) x C + (0.001 ± 0.012). Qu

In [79]:
```
RGBm = np.float(input('Entrer la valeur de la composante RGB du signal transmis par la
Am = -np.log10(RGBm/max(RGB))

yh=[Am,Am]
xh=[0,(Am-b)/a]
xv=[(Am-b)/a,(Am-b)/a]
yv=[0,Am]

plt.rcParams['figure.figsize'] = [10,6]
plt.title("A = f(C)")
plt.xlabel('C ($g\cdot L^{-1}$)')
plt.ylabel('A')
plt.grid()
plt.plot(x, y_mod, c='r')
plt.plot((Am-b)/a,Am,'go')
plt.plot(xv,yv,'g--')
plt.plot(xh,yh,'g--')
plt.show()

print('La concentration en soluté de la solution inconnue est Cm =',round((Am-b)/a,2),'
```

```
Entrer la valeur de la composante RGB du signal transmis par la solution inconnue : 37
```



La concentration en soluté de la solution inconnue est Cm = 24.03 g/L