Devoir Maison n° 5 - Correction

Partie 1 - Identification expérimentale des paramètres

1. On transforme :

$$\begin{aligned} & \nu_i = \frac{\nu_{max} s_0}{K_M + s_0} \\ \Leftrightarrow \quad & \frac{1}{\nu_i} = \frac{K_M + s_0}{\nu_{max} s_0} \\ \Leftrightarrow \quad & \frac{1}{\nu_i} = \frac{K_M}{\nu_{max} s_0} + \frac{s_0}{\nu_{max} s_0} \\ \Leftrightarrow \quad & \frac{1}{\nu_i} = \frac{K_M}{\nu_{max}} \times \frac{1}{s_0} + \frac{1}{\nu_{max}} \end{aligned}$$

$\boxed{\text{On a une relation de la forme } \nu_i^{-1} = \alpha s_0^{-1} + \beta \text{ en posant } \alpha = \frac{K_M}{\nu_{max}} \text{ et } \beta = \frac{1}{\nu_{max}}}$.

2. La démarche est la suivante :
   - On commence par collecter des couples de valeurs (s_0^{-1}, ν_i^{-1}) et on place les points associés sur un graphique.
   - Graphiquement toujours, on trace au mieux de nos capacités la droite de régression de ν_i^{-1} en s_0^{-1}. On lit graphiquement son ordonnée à l'origine β et son coefficient directeur α.
   - On résout alors :

$$\begin{cases} \alpha = \dfrac{K_M}{\nu_{max}} \\ \beta = \dfrac{1}{\nu_{max}} \end{cases} \Leftrightarrow \begin{cases} \nu_{max} = \dfrac{1}{\beta} \\ K_M = \dfrac{\alpha}{\beta} \end{cases}$$

3. (voir annexe)
4. Si on joue le jeu de tout faire uniquement graphiquement, on peut raisonnablement estimer l'ordonnée à l'origine à environ 250 et la pente à environ 4,7.

$\boxed{\text{Graphiquement, on obtient } \nu_{max} \simeq 4.10^{-3} \text{ mol.L}^{-1}\text{.min}^{-1} \text{ et } K_M \simeq 1,8.10^{-2} \text{ mol.L}^{-1}.}$

Partie 2 - Etude informatique des données expérimentales

1. (a)

```python
def inv(L):
    return [1/x for x in L]
```

(b)

```python
def inv_ex(L):
    resultat = [ ]
    for x in L:
        if x == 0:
            return False
        else :
            resultat.append(1/i)
    return resultat
```

(c)

```python
plt.plot(inv(Ls), [x*1000 for x in inv(Lv)] ,'o')
plt.show()
```

2. (a) Pour une série statistique $x = (x_1, \dots, x_n)$, la moyenne est : $\overline{x} = \frac{1}{n} \sum_{k=1}^{n} x_k$.

   On écrit donc une fonction qui calcule la somme de tous les éléments de **X** puis divise cette somme par la longueur de **X**.

```python
def moyenne(X):
    s = 0
    for x in X:
        s += x
    return s/len(X)
```

(b) Pour une série statistique $x = (x_1, \dots, x_n)$, la variance est : $\mathrm{Var}(x) = \overline{x^2} - (\overline{x})^2$ (formule de Huygens).
On écrit donc une fonction qui calcule (en une seule boucle) la somme des x_k et la somme des x_k^2 puis calcule la variance via la formule de Huygens.

```python
def variance(X):
    s = 0
    c = 0
    n = len(X)
    for x in X:
        s += x
        c += x**2
    V = c/n - (s/n)**2
    return V
```

3. (a) Vue la structure du programme, on est obligé d'utiliser la définition de la covariance de X et Y (plutôt que la formule de Huygens), à savoir ; pour deux séries statistiques $x = (x_1, \dots, x_n)$ et $y = (y_1, \dots, y_n)$ de même longueur, $\mathrm{Cov}(x, y) = \frac{1}{n} \sum\limits_{k=1}^{n} (x_k - \overline{x})(y_k - \overline{y})$.

```python
def cov(X, Y) :
    """Entree: X,Y (liste)."""
    nx = len(X) ; ny = len(Y)
    if nx != ny or nx == 0 :
        return(False)
    else :
        mx = moyenne(X) ; my = moyenne(Y) ; S = 0
        for k in range(0,nx) :
            S = S + (X[k] - mx)*(Y[k] - my)
        y = 1/nx*S
        return(y)
```

(b) La fonction **cov** peut renvoyer soit **False**, qui est de type **bool** soit un nombre de type **float**. Or :
- **True** est de type **bool**, mais ce n'est pas le booléen **False**.
- **[1, 2]** est de type **list**.
- **"False"** est de type **str** (c'est une chaîne de caractères).
- **-0.5** est de type float.

Conclusion : $\boxed{\text{Les valeurs i., ii. et iii. ne peuvent pas être renvoyées par } \textbf{cov}.}$

(c) i. On complète :

```python
def Trace(X, Y) :
    [a, b] = Coef(X, Y)
    xmin = min(X) ; xmax = max(X)
    plt.plot(X, Y, "*")
    plt.plot([xmin, xmax],[a*xmin+b, a*xmax+b])
    plt.plot([moyenne(X)], [moyenne(Y)], "s")
    plt.grid()
    plt.show()
```

ii. Cette droite a pour équation $\boxed{\text{y=ax+b}}$.
Vue la manière dont a et b sont obtenus, on peut aussi répondre que cette droite a pour équation :
$y = \frac{\mathrm{cov}(X,Y)}{\mathrm{Var}(X)} x + \overline{y} - \frac{\mathrm{cov}(X,Y)}{\mathrm{Var}(X)} \overline{x}$.

En simplifiant, on trouve $\boxed{y = \frac{\mathrm{cov}(X,Y)}{\mathrm{Var}(X)} (x - \overline{x}) + \overline{y}}$.

iii. Cette droite est la droite de régression de y en x.

iv. - Dans le tracé n°2, le segment tracé ne passe pas par le point représenté par un carré.
Or : le segment est porté par la droite de régression de y en x ; le point représenté par un carré est le point moyen du nuage de points ; la droite de régression d'un nuage de points passe toujours par le point moyen.
Le tracé n°2 ne peut pas être le résultat de **Trace**.

- Dans le tracé n°3, 4 segments sont tracés.
Or : la fonction **Trace** trace un seul segment.
Le tracé n°2 ne peut pas être le résultat de **Trace**.

- Dans le tracé n°1, il y a un nuage de cinq points non reliés représentés par des étoiles, un segment qui relie les points extrémaux, un point (qui peut être le point moyen vues les valeurs et vu le fait qu'il est sur le segment) représenté par un carré et une grille.
Tout ces éléments prouvent que le tracé n°1 peut être le résultat de **Trace**.

4(a)

```
X = inv(Ls)
Y = [x*1000 for x in inv(Lv)]
[a,b]=Coef(X,Y)
KM = a/b
vmax = 1/b
```

(b) Ce coefficient de corrélation linéaire est très proche de 1.
On en déduit que l'ajustement est très bon.

Annexe 2