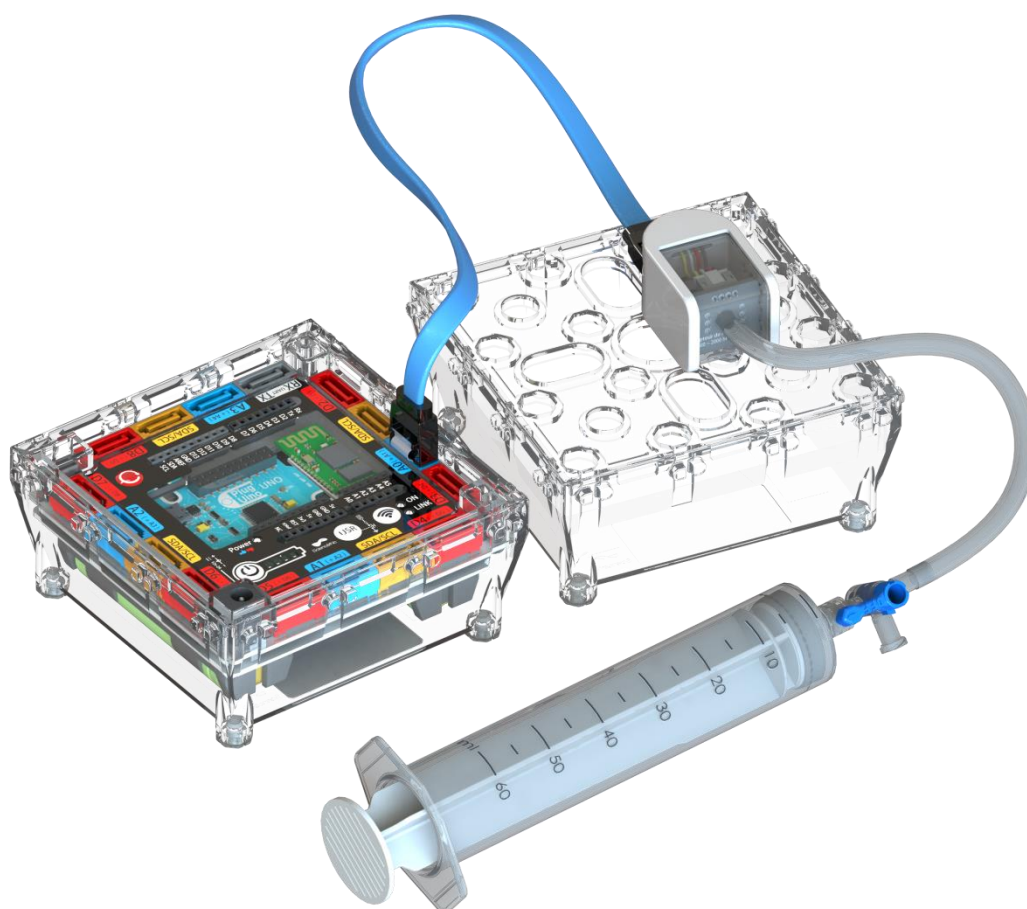


Niveau : première enseignement de spécialité	Mouvement et interactions partie 2. Description d'un fluide au repos
Modèle de comportement d'un gaz : loi de Mariotte.	Utiliser la loi de Mariotte. <i>Tester la loi de Mariotte, par exemple en utilisant un dispositif comportant un microcontrôleur.</i>
Variabilité de la mesure d'une grandeur physique.	Discuter de l'influence de l'instrument de mesure et du protocole.

Dossier TP Plug'Uino® Loi de Mariotte Programmation en langage Python



1. Résumé de l'activité

Cette activité expérimentale a pour but de confronter le modèle de la loi de Mariotte à des résultats expérimentaux obtenus à l'aide d'un capteur de pression relié à un microcontrôleur.

La programmation est réalisée en langage Python et les traitements avec un tableur grapheur

2. Thème du programme abordé

Niveau : première enseignement de spécialité	Mouvement et interactions partie 2. Description d'un fluide au repos
Modèle de comportement d'un gaz : loi de Mariotte.	Utiliser la loi de Mariotte. <i>Tester la loi de Mariotte, par exemple en utilisant un dispositif comportant un microcontrôleur.</i>
Variabilité de la mesure d'une grandeur physique.	Discuter de l'influence de l'instrument de mesure et du protocole.

3. Matériel mis en œuvre

- Une interface Plug'Uino® Réf. 650 003
- Un capteur de pression absolue (-1000;+2000 hPa) Réf. 651 055
- Un dispositif permettant de faire varier le volume d'une quantité de matière constante d'un gaz.

4. Présentation de l'expérience réalisée

A l'aide d'un microcontrôleur, on mesure la valeur renvoyée par le capteur de pression pour différents volumes de la même quantité d'air contenue dans le dispositif.

Les mesures successives sont traitées à l'aide d'un tableur-grapheur, l'objectif étant de montrer la proportionnalité entre P et 1/V (Loi de Mariotte : $P \cdot V = \text{cste}$)

Pour lisser les fluctuations, les valeurs prises en compte résultent d'une moyenne obtenue à partir de 20 valeurs mesurées tous les dixièmes de seconde.

Le microcontrôleur est particulièrement adapté pour réaliser de nombreuses mesures en un temps limité, ce qui permet d'obtenir une mesure qui moyenne les légères fluctuations des valeurs fournies par le capteur.

5. Programmation du microcontrôleur

Pour la mise en place de la connexion du microcontrôleur et la configuration initiale Python, se référer à la fiche : *Python et microcontrôleur.pdf*

Le script Python fonctionnel, permettant d'effectuer les mesures du signal aux bornes du capteur, est présenté ci-dessous et est disponible en téléchargement : *exemple_pyfirmata.py*

```

1  """ Sciencéthic script exemple de lecture sur une broche analogique d'un microcontrôleur type Arduino
2  capteur branché sur la broche analogique A1, LED branchée sur la broche digitale D2"""
3
4  from pyfirmata import Arduino, util
5  import time                                #importation des bibliothèques
6
7  #----- paramétrage de la communication-----
8  com=input('Saisir le port de communication COM1, COM2, COM3 .... :')
9  port = Arduino(com)                        # la variable port contient le port de communication
10 util.Iterator(port).start()                # itérateur permettant de ne pas saturer la liaison série
11 A1 = port.get_pin('a:1:o')                 # A1 désigne la broche analogique 1 (a:1) utilisée en sortie (:o)
12 time.sleep(0.5)                           # délai d'attente pour la mise en place de la communication
13 port.digital[2].write(1)                   # écriture sur le port digital 2 (Ok pour mesure si LED allumée)
14
15 #----- mesures -----
16 mesures_capteur=[]                         # liste des mesures
17 poursuite=True                             # initialisation de la variable poursuite des mesures
18 while poursuite:                           # tant qu'une mesure est à effectuer
19     m=A1.read()                            # lecture de la valeur sur la broche A1
20     mesures_capteur.append(m)               # ajout de la valeur dans la liste
21     print('mesure : ',m)                   # affichage de la valeur mesurée
22     poursuite=input('Déclencher une nouvelle mesure o/n ? ')
23     if poursuite=='n' or poursuite=='N':
24         poursuite=False
25 print('liste des mesures : ',mesures_capteur) # affichage de la liste des mesures
26 port.digital[2].write(0)                   # écriture sur le port digital 2 (fin de mesure si LED éteinte)
27 port.exit()                               # fermeture de la communication sur le port

```

Un script plus complet permettant de saisir le volume en parallèle avec les mesures et d'effectuer éventuellement un tracé.

Ce script est disponible en téléchargement : *loi_Mariotte_complet.py*

```

1 """ Sciencéthic mesures pour loi de Mariotte
2 lecture sur une broche analogique d'un microcontrôleur type Arduino
3 capteur branché sur la broche analogique A1, LED branchée sur la broche digitale D2"""
4 # -----importation des bibliothèques-----
5 from pyfirmata import Arduino, util
6 import time
7
8 #----- paramétrage de la communication-----
9 com=input('Saisir le port de communication COM1, COM2, COM3 .... :')
10 port = Arduino(com) # la variable port contient le port de communication
11 util.Iterator(port).start() # itérateur permettant de ne pas saturer la liaison série
12 A1 = port.get_pin('a:1:o') # A1 désigne la broche analogique 1 (a:1) utilisée en sortie (:o)
13 time.sleep(0.5) # délai d'attente pour la mise en place de la communication
14 port.digital[2].write(1) # écriture sur le port digital 2 (0k pour mesure si LED allumée)
15
16 #----- mesures -----
17 mesure=[] # liste des mesures
18 volume=[] # liste des volumes
19 poursuite=True
20 while poursuite:
21     v=float(input('volume en mL :'))
22     volume.append(v)
23     a=input('taper ENTER pour effectuer la mesure')
24     m=A1.read()
25     mesure.append(m)
26     print('mesure : ',m)
27     poursuite=input('nouvelle mesure o/n : ')
28     if poursuite!='o' and poursuite!='O':
29         poursuite=False
30
31 port.digital[2].write(0) # écriture sur le port digital 2 (fin de mesure LED éteinte)
32 port.exit() # fermeture de la communication sur le port
33
34 print('Volumes : ',volume)
35 print('Mesures : ',mesure)

```

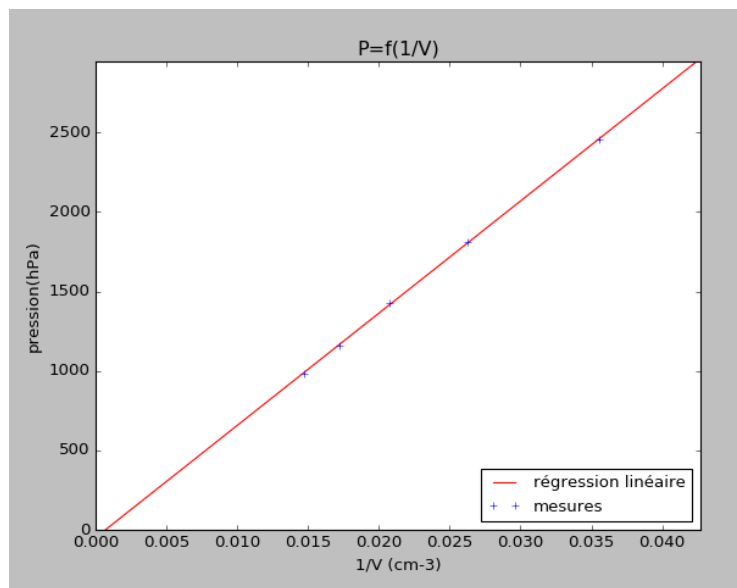
6. Résultats obtenus et traitement :

Exemples de mesures obtenues et traitées :

Attention : il est nécessaire d'ajouter au volume lu sur la seringue, le volume de gaz contenu dans le tube reliant l'embout de la seringue au capteur !

Volumes (cm^3) : [65.0, 55.0, 45.0, 35.0, 25.0]

Mesures : [0.3275, 0.3842, 0.4741, 0.6031, 0.8162]



7. Exploitation

7.1. Programmation

En fonction des capacités des élèves, on peut proposer plusieurs déclinaisons :

- à un niveau **d'initiation**, enlever certains commentaires du programme et demander aux élèves à quoi correspond cette ligne ;
- à un niveau **plus avancé**, effacer certaines lignes et demander aux élèves de programmer eux-mêmes ces lignes (par exemple, la ligne où la mesure est effectuée) ;
- à un niveau **expert**, demander aux élèves de programmer le microcontrôleur : dans un premier temps, pour renvoyer une mesure unique ; dans un second temps, pour acquérir une série de mesures pour le calcul d'une moyenne.

7.2. Résultats des mesures

On peut compléter l'analyse sur les points suivants :

- régression linéaire sur le tracé $P=f(1/V)$
- choix de l'échelle verticale dans le cas du tracé $p*V$
- influence de la variation de la température lors de changements de volume trop rapides

8. Pistes d'exploration possibles

- Observer le changement de température lors d'une variation de volume.
- Modifier le gaz.

9. Points de vigilance

- Évaluer le volume de gaz contenu dans le tube de raccordement du capteur à la seringue pour l'intégrer dans le volume total de gaz.
- Pour le tracé de $P*V$, attention au choix de l'échelle verticale pour ne pas mettre en évidence les fluctuations inhérentes à la précision de la mesure.
- Dans le cas du tracé d'un graphique, le fermer pour relancer l'exécution du script.