

Rapport projet en ingénierie (2021–2022)

Modélisation de la ventilation et mesure du dioxyde de carbone dans des espaces clos

Erdi ÇAN, Baptiste BRAUN-DELVOYE, Axel OUGA, Enzo TONATI

Encadré par José Maria FULLANA

23 mai 2022

Remerciements

Nous tenons tout d'abord à remercier M. FULLANA, qui était toujours là pour nous aider et nous guider dans notre projet. Nous lui sommes reconnaissants pour la confiance qu'il a eue en nous et d'avoir donné de son temps.

Un grand merci aussi à M. BÔNE, M. DARINGE, M. CHASSAING et M. WUNENBURGER pour avoir répondu à toutes nos questions rapidement.

Nous remercions également toute l'équipe du CMI qui nous a également aidés durant ce projet et qui nous a laissé travailler sur ce sujet.

Table des matières

Résumé	iii
Abstract	iii
Introduction	1
1 Modèle théorique	3
1.1 Objectifs	3
1.2 Protocole	3
1.3 Résultats	5
1.4 Influence d'un apport en flux sur une particule fine	7
2 Le système de mesure	8
2.1 Carte Arduino	8
2.2 Capteur de CO_2	9
2.3 Les <i>shields</i> de carte SD et de base	9
2.4 Protocole des mesures	10
2.5 Difficultés rencontrées	10
3 Mesures	12
3.1 Étalonnage des capteurs	12
3.2 Dynamique des particules	14
3.2.1 Espace vide	14
3.2.2 Espace avec n personnes	15
3.2.3 Espace avec n personnes et purificateur d'air	18
3.3 Aération d'une pièce	19
Conclusion	24
Bibliographie	25
Annexes	27
Nos Codes	27
Codes Arduino	27
Codes pour les graphs et données	32

Résumé

Nous nous sommes intéressés à la dynamique des particules fines dans un écoulement. Notre objectif fut de comprendre la manière d'optimiser la ventilation d'une salle ou d'un couloir pour avoir une qualité d'air dans les normes. Pour cela nous avons analysé la concentration en CO_2 présent dans l'air avec une valeur de 440 ppm à l'extérieur. Nous avons commencé par étudier la dynamique de suspension des particules dans l'air puis nous l'avons relié avec la ventilation ambiante. On a analysé un modèle simplifié avec une bille tombant dans un liquide et nous avons vu si ce modèle concordait avec la théorie.

Nous avons construit des capteurs de CO_2 autonomes avec des cartes électroniques Arduino et fait une série de mesures avec différentes conditions.

Enfin, nous avons montré comment une particule fine arrive à rester dans l'air plusieurs heures, voire plusieurs jours et l'intérêt d'aérer une pièce correctement pour le renouvellement de l'air.

Abstract

This paper analyses the dynamic of Carbon dioxide (CO_2) in a closed space with flow and what we can do to keep CO_2 levels safe. To answer this question first we made a simplified model of CO_2 particles using glass marbles in glycerol and then we studied the dynamic suspension of fine particles in the air and associated our results in a room with air flow.

For the measurements, we are going to do various measures in different conditions using Arduinos equipped with CO_2 sensors.

Our results showed how a fine particle stays in the air for long time periods and how important it is to ventilate the rooms that we spend time in.

Introduction

Dans le cadre de notre Projet En Ingénierie (PEI), nous nous intéressons à la concentration de particules fines dans une pièce. Les particules fines sont des particules en suspension. Elles sont portées par l'eau ou l'air et sont quantifiables par plusieurs procédés physiques. Elles peuvent avoir plusieurs origines[1] :

- Origines naturelles : par éruptions volcaniques ou par le phénomène d'érosion par le vent;
- Origines anthropiques : par la pollution du trafic routier ou même du chauffage résidentiel.

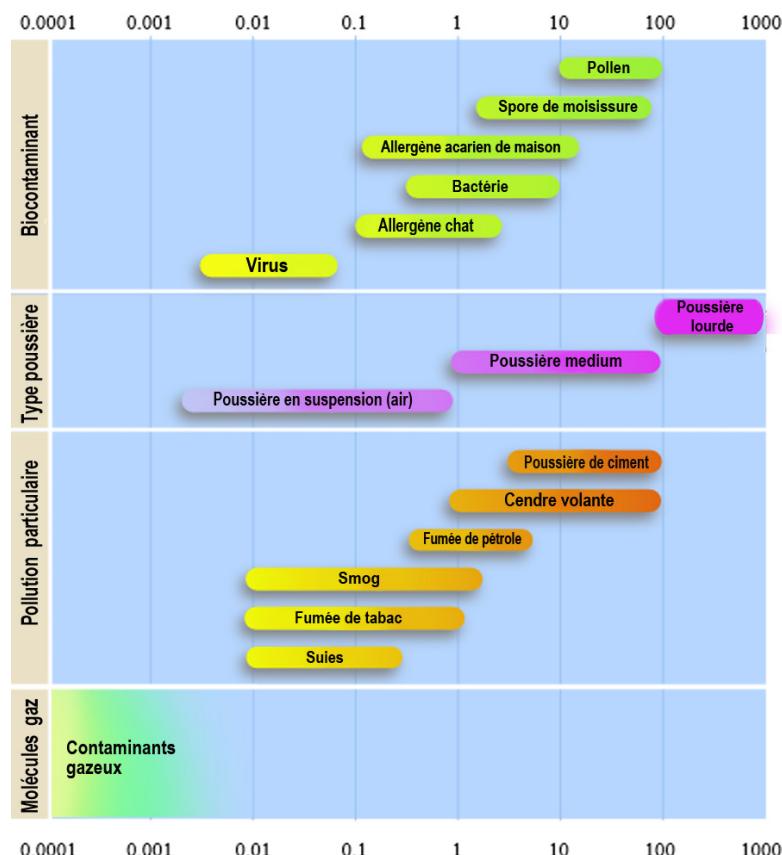


FIGURE 0.1 – Visualisation des catégories de particules en suspension dans l'air ou aéroportées (contaminants biologiques, ou particulaires minéraux ou organiques, ou gazeux)... par nature et taille (en micromètres ; μm)[2]

Durant notre étude nous nous sommes intéressés au dioxyde de carbone. Le dioxyde de carbone (CO_2) est une molécule d'eau et d'oxygène. C'est un gaz à effet de serre majeur augmentant par l'utilisation d'énergies fossiles.

Cependant, le dioxyde de carbone peut également être produit par l'organisme humain lorsqu'il expire de l'air. De ce fait, si une salle n'est pas aérée souvent, la quantité de particules de CO_2 augmente. La concentration de CO_2 est mesurée en ppm (partie par million) ou $\mu\text{g} \cdot \text{m}^{-3}$ et peut avoir des effets nocifs pour la santé si la concentration de CO_2 que l'on inhale est trop importante :

Selon l'ANSES (Agence nationale de sécurité sanitaire de l'alimentation, de l'environnement et du travail), l'air intérieur d'un bâtiment possède un taux de CO_2 compris généralement entre 350 et 2500 ppm. De plus, on peut observer des effets nocifs lorsque la concentration en CO_2 dépasse 1000 ppm. A partir de ce seuil, la respiration des particules de CO_2 peut

entraîner des signes de fatigue et des maux de tête. Mais récemment il fut montré qu'une légère augmentation de la concentration de CO_2 était déjà suffisante pour dégrader nos capacités cognitives[3].

C'est pourquoi il est très important d'aérer une salle plus ou moins longtemps en fonction du nombre de personnes présentes, surtout dans une période de pandémie. En tant qu'étudiants, nous nous intéresserons au cas du bureau ou de notre salle de projet.

Afin de réaliser ces expériences, nous nous sommes basés sur les connaissances en dynamique des fluides acquises cette année. Mais nous avons également dû faire un travail de recherche et d'apprentissage sur le sujet des écoulements et des capteurs.

1 Modèle théorique

1.1 Objectifs

Le but de notre expérience est de comprendre comment se comportent les particules de CO_2 dans l'air. Pour cela nous avons choisi un modèle grossier comportant une particule en chute libre dans l'air que l'on étendra par la suite à des fluides quelconques.

1.2 Protocole

Pour établir notre protocole expérimental, nous nous sommes appuyés sur la loi de Stokes. Selon cette dernière on pourrait prédire la vitesse de chute stabilisée ou la vitesse limite d'une sphère soumise à la pesanteur dans un fluide.

La loi de Stokes est une loi donnant la force de traînée hydrodynamique s'exerçant sur une sphère en déplacement dans un fluide. Pour que la loi puisse être exploitée il faut respecter certaines conditions :

- Le nombre de Reynolds (Re) doit être très inférieur à 1 (une viscosité dynamique très élevée, $\mu >> \rho v D$);
- La sphère est suffisamment loin de tout autre corps, de tout obstacle ou paroi latérale.

$$Re = \frac{\rho_{sphère} v D}{\mu}$$

Avec :

- $\rho_{sphère}$: la masse volumique de la particule en $kg \cdot m^{-3}$;
- v : la vitesse de la particule en $m \cdot s^{-1}$;
- D : le diamètre de la particule en m ;
- μ : la viscosité dynamique du milieu en $Pa \cdot s^{-1}$.

L'expérience se fera en régime laminaire avec une viscosité dynamique élevée pour respecter les conditions et ainsi observer la relation décrite par la loi de Stokes. On utilisera le nombre de Stokes (St) comme indicateur permettant de savoir à l'avance si la particule suivra l'écoulement du fluide.

$$St = \frac{E_c}{frottement} = \frac{\rho_{air} D^2 v_{air}}{18 \mu_{air} L}$$

Avec :

- ρ_{air} : la masse volumique de l'air en $kg \cdot m^{-3}$;
- v_{air} : la vitesse de l'écoulement en $m \cdot s^{-1}$;
- L : la longueur caractéristique en m .

Le bilan des forces exercées sur les particules en chute libre : suivant l'axe \vec{e}_y on a :

- Poids (P) = $-mg$;
- La poussée d'Archimède (A) = $\rho_{air} V g$;
- La traînée hydrodynamique de l'air (T) = $3\pi \mu_{air} v_l D$.

A l'équilibre translationnel, on peut écrire :

$$T = 3\pi\mu_{air}v_l D = \frac{\pi D^3}{6} \cdot (\rho_{sphère} - \rho_{air})g$$

On en déduit une vitesse limite v_l :

$$v_l = \frac{D^2(\rho_{sphère} - \rho_{air})g}{18\mu_{air}}$$

$$\text{Or, on sait que } v_l = \frac{d}{t}$$

Nous pouvons donc en tirer le temps nécessaire à la chute de ce solide.

On observe une relation,

$$v_l = \frac{D^2(\rho_{sphère} - \rho_{air})g}{18\mu_{air}} = \frac{d}{t}$$

$$\Rightarrow t_{chute} = \frac{18\mu_{air}d}{D^2(\rho_{sphère} - \rho_{air})g}$$

Nous avons alors mis en œuvre ce modèle théorique de sorte à pouvoir observer une vitesse limite et ainsi déterminer le temps de chute d'une particule en fonction de sa masse et de son diamètre dans un fluide.

Pour ce faire nous nous sommes munis :

- D'une éprouvette graduée (1L) remplie de glycérine, un liquide à forte viscosité (1600 fois plus visqueux que l'eau);
- D'une règle graduée permettant d'obtenir à l'aide du chronomètre la position, avec une incertitude de 0.05 cm;
- D'une potence;
- D'un chronomètre permettant d'obtenir le temps de chute;
- Et enfin de 4 billes en verre mais de diamètres et de masses qui diffèrent, représentant les particules du modèle théorique (particules de CO_2 en chute libre).

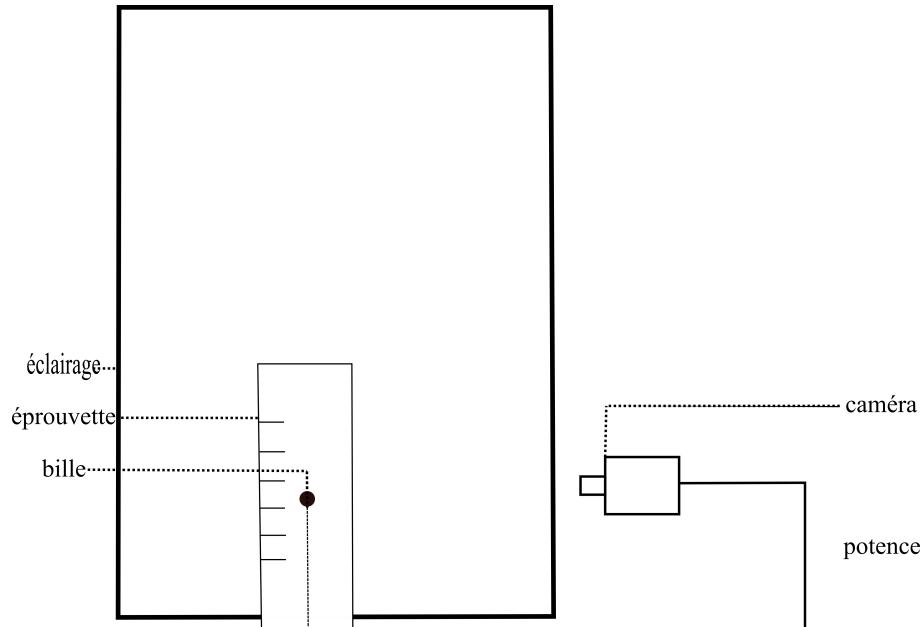


FIGURE 1.1 – Schéma de l'expérience

Pour mesurer et peser les billes, nous avons utilisé un pied à coulisse avec une incertitude de 0.005 mm, et une balance de Roberval avec une incertitude de 0.05 g. Nous avons au final :

- BilleS : Bille Standard avec $d_{BilleS} = 16.27 \pm 0.31 \text{ mm}$, $m_{BilleS} = \frac{89.0 - 60.7}{5} = 5.66 \pm 0.1 \text{ g}$;
- BilleG : Bille Grande avec $d_{BilleG} = 7.70 \pm 0.11 \text{ mm}$, $m_{BilleG} = \frac{95.7 - 60.7}{100} = 0.35 \pm 0.1 \text{ g}$;
- BilleM : Bille Moyenne avec $d_{BilleM} = 4.96 \pm 0.005 \text{ mm}$, $m_{BilleM} = \frac{77.2 - 60.7}{100} = 0.165 \pm 0.1 \text{ g}$;
- BilleP : Bille Petite avec $d_{BilleP} = 3.97 \pm 0.04 \text{ mm}$, $m_{BilleP} = \frac{69.1 - 60.7}{100} = 0.084 \pm 0.1 \text{ g}$.

Nous avons également mesuré l'écart entre les traits de l'éprouvette. Pour ce faire, nous avons d'abord mesuré la hauteur $h_{100 \rightarrow 1000 \text{ mL}}$ entre le trait de 100 mL et le trait de 1000 mL :

$$h_{100 \rightarrow 1000 \text{ mL}} = 30.75 \pm 0.005 \text{ cm}$$

Puis nous avons divisé par le nombre de traits qu'il y avait entre eux, ce qui nous donne :

$$h_{entredeuxtraits} = \frac{30.75}{90} = 0.34 \pm 0.005 \text{ cm}$$

1.3 Résultats

On lâche chaque bille en haut de l'éprouvette graduée, on chronomètre le temps caractéristique entre 2 points pour chaque bille, nous permettant de déterminer «manuellement» la vitesse limite ou le temps de chute. Voici le graphique comparant la vitesse théorique et expérimentale des billes :

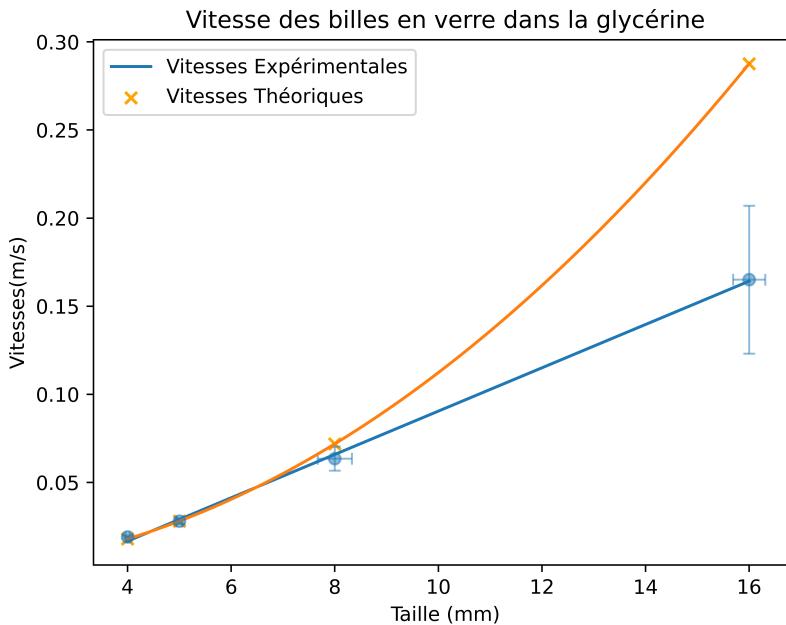


FIGURE 1.2 – La vitesse des billes en fonction de leur taille

Nous nous apercevons que les deux modèles ne suivent pas la même courbe. Le modèle théorique suit une parabole, tandis que les valeurs expérimentales suivent une fonction affine. Cependant nous nous rendons compte que les valeurs des 2 premières billes sont identiques et l'incertitude de la 3ème bille touche la valeur théorique. Nous pouvons donc en conclure que la droite formée par les valeurs expérimentales est la tangente de la courbe du modèle théorique. Nous pouvons donc en conclure que le modèle est vérifié pour les objets de très petite taille, donc il est également vérifié pour des particules fines.

En conclusion, grâce aux expériences on a pu démontrer que la loi de Stokes était vérifiée mais uniquement pour des objets de très petite taille. Nous pouvons donc utiliser l'équation trouvée précédemment pour t_{chute} avec les caractéristiques d'une particule de CO_2 qui sont :

- $D_{CO_2} = 10 \cdot 10^{-6} m$;
- $\rho_{CO_2} = 1.87 \text{ kg} \cdot m^{-3}$.

Et nous avons aussi :

- $\rho_{air} = 1.22 \text{ kg} \cdot m^{-3}$ pour 15 degrés Celsius;
- $\mu_{air} = 18.5 \cdot 10^{-6} Pa \cdot s$;
- $g = 9.81 \text{ m} \cdot s^{-2}$;
- nous prenons une hauteur de $d = 1.70 \text{ m}$.

Nous avons donc :

$$t_{chute} = \frac{18\mu_{air}d}{D_{CO_2}^2(\rho_{CO_2}\rho_{air})g}$$

$$\text{Analyse numérique : } t_{chute} = \frac{18 \times 18.5 \cdot 10^{-6} \times 1.70}{(10 \cdot 10^{-6})^2 \times (1.87 - 1.22) \times 9.81}$$

$$t_{chute} = 88791.1 \text{ s} \approx 10 \text{ jours}$$

Si nous reprenons nos données, nous avons aussi :

$$\nu_l = \frac{D_{CO_2}^2 (\rho_{CO_2} - \rho_{air}) g}{18 \mu_{air}} \approx 2 \cdot 10^{-6} \cdot m \cdot s^{-1}$$

Ce qui nous donne un nombre de Reynolds équivalent à :

$$Re = \frac{1.87 \times 2 \cdot 10^{-6} \times 10 \cdot 10^{-6}}{18.5 \times 10^{-6}} = 2.02 \times 10^{-6} \ll 1$$

Cela montre encore une fois que notre modèle est vérifié.

1.4 Influence d'un apport en flux sur une particule fine

Lorsque nous ouvrons une ouverture d'une pièce, la différence de pression entre la salle et l'extérieur créé un courant d'air. Ici nous considérerons un courant d'air allant de l'intérieur vers l'extérieur. Le nombre de Stokes va nous permettre de savoir si une particule de CO_2 suivra ou non l'écoulement. Si le nombre de Stokes est très grand comparé à 1, alors la particule ne sera que très peu affectée par l'écoulement de l'air. S'il est très petit comparé à 1 alors la particule sera directement emportée par le vent.

En reprenant les données précédentes avec une vitesse d'écoulement ν quelconque et une longueur $L = 10 \text{ m}$, nous obtenons :

$$St = \frac{1.22 \times 10 \cdot 10^{-6} \times \nu}{18 \times 18.5 \cdot 10^{-6} \times 10}$$

$$St = 3.66 \cdot 10^{-6} \nu \ll 1$$

Le nombre de Stokes reste très petit comparé à 1, la particule de CO_2 sera donc emportée par le courant d'air, permettant le renouvellement de l'air dans la salle.

2 Le système de mesure

Pour notre système de mesure on a choisi 3 Arduinos Leonards, équipés chacun d'un *shield* de carte SD[4], d'une base *shield*[5] et d'un capteur SDC30[6] (qui est un capteur de CO_2 , de température et d'humidité).

2.1 Carte Arduino

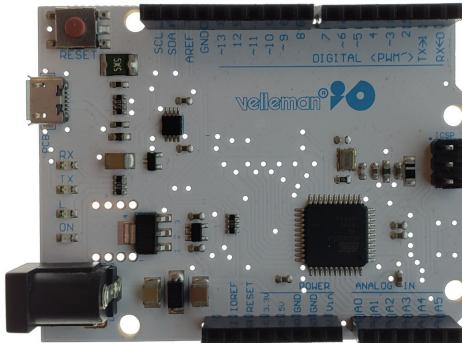


FIGURE 2.1 – Carte Arduino Leonardo

Comme carte Arduino on a utilisé une carte équivalente à une carte Leonardo[7] équipée d'un micro-contrôleur : ATmega32u4 avec 20 broches numériques d'entrée et de sortie et de 12 entrées analogiques. La carte a une mémoire flash de 32 ko , 2.5 ko SRAM, 1 ko EEPROM et une vitesse d'horloge de 16 Mhz. Ses spécifications nous suffisent amplement pour nos travaux. Les utilisateurs d'Arduino forment une très grande communauté à travers le monde. Il fut donc aisément de trouver des guides et ressources pour nous familiariser avec ces micro-processeurs.

2.2 Capteur de CO_2

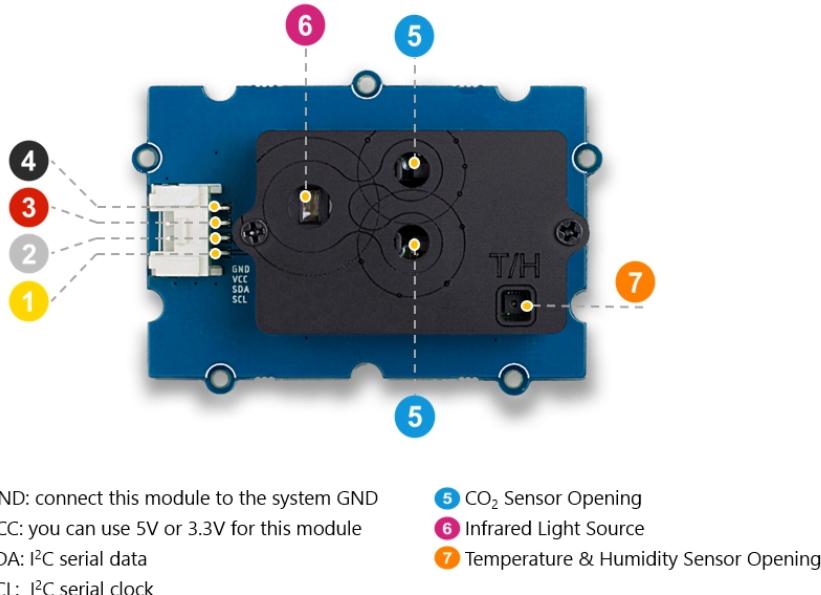


FIGURE 2.2 – Capteur de CO_2

Comme capteur de CO_2 on a utilisé le capteur Grove fait par Seeed Studio[6] qui est basé sur le capteur Sensirion SCD30. Le capteur est qualifié de NDIR (pour "nondispersive infrared spectroscopy", "spectroscopie infrarouge non dispersive"). Il mesure l'absorption de la lumière infrarouge par les molécules de l'air. La bande de radiation infrarouge produite par la lampe du capteur est très proche de celle d'absorption du CO_2 . Le CO_2 a un spectre infrarouge assez unique (une bande d'absorption de 4.26 microns[8]. De ce fait, en mettant une lumière infrarouge qui émet des ondes de longueur assez proche de 4.26 microns, la lumière va se faire absorber par le CO_2 . À la fin nous mesurons la lumière non absorbée pour connaître la quantité de ppm de CO_2 dans le lieu voulu[9]. Selon le constructeur[6], l'incertitude du capteur est de 30 + 3% ppm. De plus, le capteur mesure la température et le pourcentage d'humidité dans l'air. Cela permet une meilleure précision du capteur [10]. Nous préciserons la température à chaque séance de mesure.

2.3 Les *shields* de carte SD et de base

Avec notre Arduino on a utilisé le *Shield* de carte SD pour pouvoir stocker nos données durant nos expériences. Avec ce *Shield*[4] mis directement sur l'Arduino et en téléversant un code (voir annexe) dans celui-ci, nous pouvons enregistrer nos données dans un fichier .txt dans la carte SD.

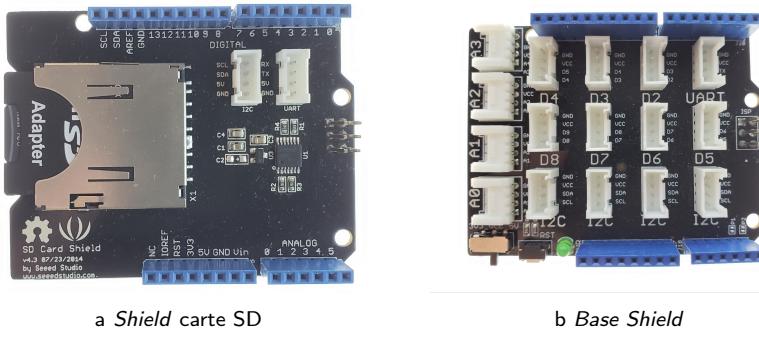


FIGURE 2.3 – *Shield* carte SD et *Base*

Nous avions commandé les *Shields Base* initialement dans le but de faciliter la connexion de capteur LoRa[11] sur les Arduinos, afin de recevoir les données directement sur nos téléphones ou ordinateurs. Au final nous avons utilisé ces *Shields* afin de connecter plus facilement les capteurs sur les micro-processeurs.

2.4 Protocole des mesures

Pour pouvoir mesurer à l'aide des capteurs, nous avons tout d'abord repris le code des anciens élèves sur le projet. Nous l'avons modifié pour que les fichiers .txt soient plus pratiques à lire. L'analyse des données s'est faite en utilisant Python et les bibliothèques Matplotlib, NumPy et Sys. Nous avons codé nous-mêmes plusieurs fonctions permettant de lire les données du fichier .txt, calculer les incertitudes des valeurs des capteurs et les afficher sur un graphique. Nos codes sont dans l'annexe et une partie plus extensive avec nos données sur GitHub [12]. Le placement des capteurs varie en fonction de nos expériences. Leurs positions seront précisées dans chaque protocole dans la partie suivante.

Voici donc nos capteurs de CO_2 installés avec leurs *shields* :

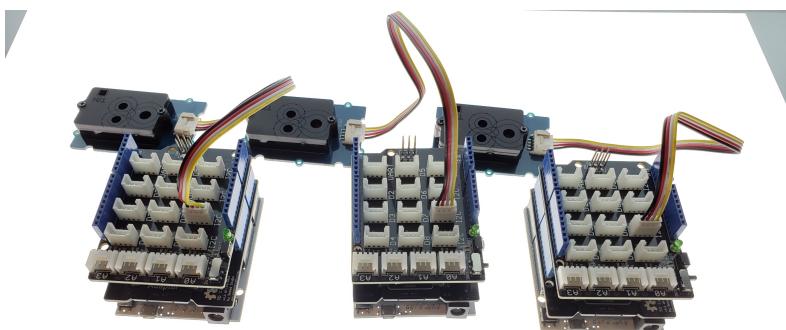


FIGURE 2.4 – Capteur de CO_2 montés

2.5 Difficultés rencontrées

Nous avons rencontré de nombreuses difficultés avec les LoRa (Long Range) qui sont des émetteurs et récepteurs à basse fréquence qui permettent la communication entre deux modules avec un débit faible et utilisant peu d'énergie, ces modules ont la capacité d'envoyer des données dans un rayon de plusieurs kilomètres.

La technologie LoRa est très prometteuse mais il n'y a actuellement pas assez de ressources à ce sujet et encore moins avec Arduino. Nous avons sollicité l'aide de chercheurs qui ont bien voulu répondre à nos questions.

Nous avons également décidé de ne pas utiliser d'autres systèmes de communication comme le Bluetooth ou le WiFi. Les modules Bluetooth d'Arduinos n'avaient pas une assez grande portée et les modules WiFi ont du mal à fonctionner dès qu'un mur les sépare.

3 Mesures

3.1 Étalonnage des capteurs

Nous devons, pour pouvoir utiliser nos capteurs, les étalonner. Pour utiliser nos capteurs, il fallait vérifier leur précision. Pour cela, nos capteurs ont été mis en extérieur à l'abri du vent pendant 30 minutes. Sachant que la quantité de CO_2 dans l'atmosphère est d'environ 440 ppm, nous avons pu vérifier si nos capteurs étaient proches ou non de cette valeur. Le graphique suivant montre les données prises par chaque capteur lors de cette séance :

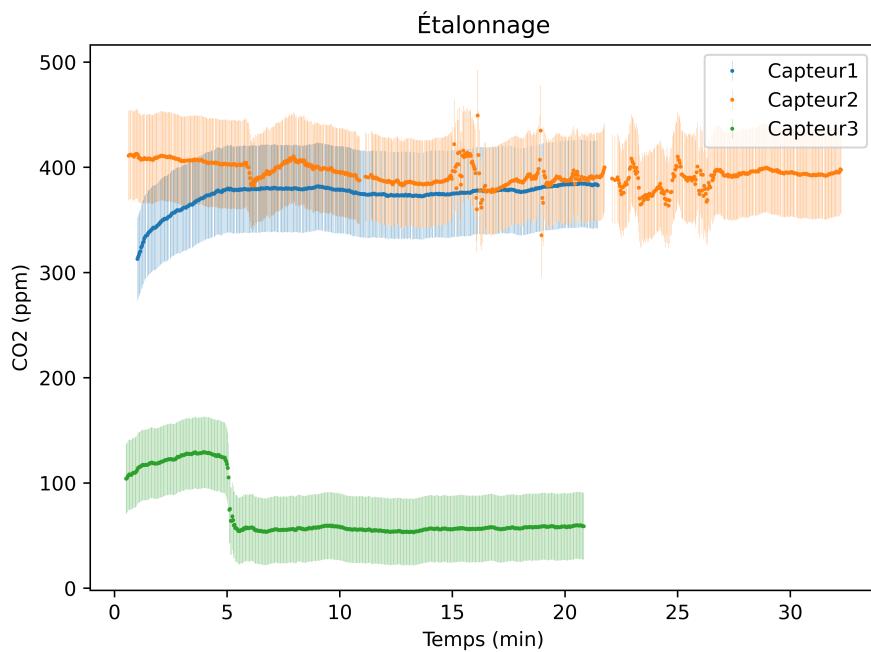


FIGURE 3.1 – Étalonnage des capteurs

Le graphique permet de se rendre compte qu'il faut étalonner les capteurs. Pour cela, la fonction « scd30.forceRecalibrationWithReference(440); » permet au capteur un nouvel étalonnage. Après cela, un nouvel essai est fait, mais cette fois-ci dans la salle de TP pendant la nuit et voici le graphique correspondant :

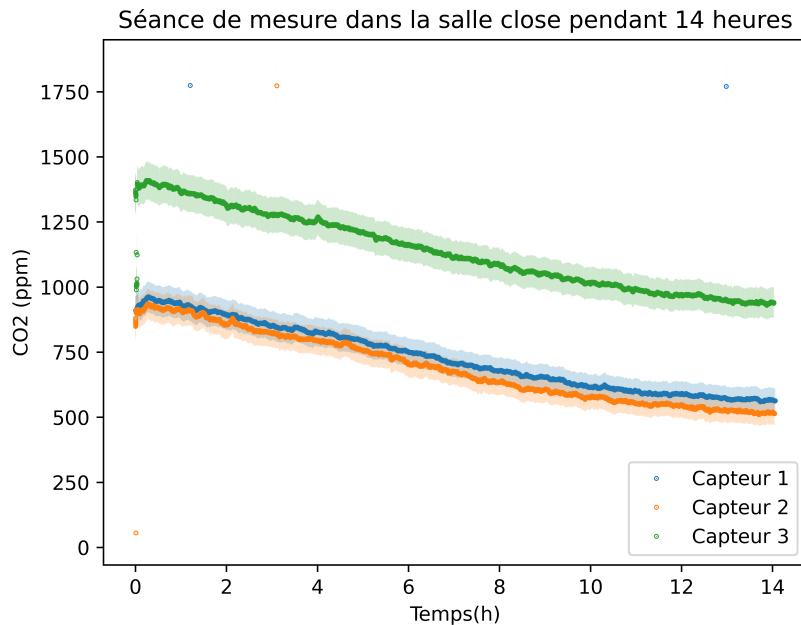


FIGURE 3.2 – 2e étalonnage des capteurs

On remarque cette fois-ci que le capteur 1 et 2 sont bien étalonnés et leurs valeurs sont conformes à l'incertitude. Cependant, le capteur 3 a une erreur systématique que nous allons calculer. Nous avons donc pris la moyenne des valeurs des capteurs 1 et 2 qui est de :

$$m_{1 \text{ et } 2} = \frac{m_1 + m_2}{2} = \frac{730.15 + 691.97}{2} = 711.06$$

Maintenant, nous soustrayons cette valeur à la moyenne du capteur 3 :

$$m_3 - m_{1 \text{ et } 2} = 1137.42 - 711.06 = 426.36$$

Cette valeur de 436.36 sera donc constamment soustraite des valeurs du capteur 3 automatiquement à partir de maintenant. Si nous reprenons donc le dernier graphique avec les valeurs modifiées, nous avons :

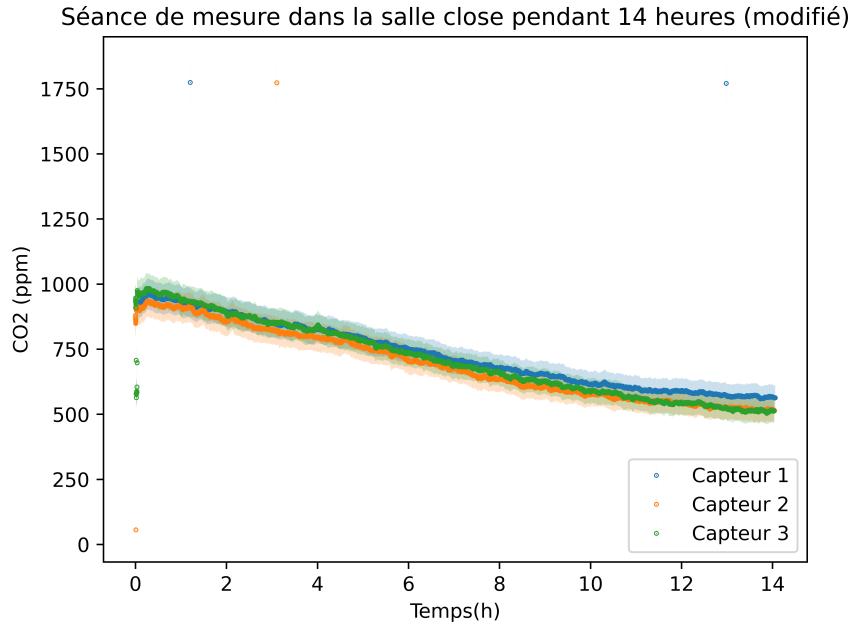


FIGURE 3.3 – Étalonnage final des capteurs

Maintenant que les capteurs sont étalonnés, nous pouvons les utiliser pour nos expériences.

3.2 Dynamique des particules

3.2.1 Espace vide

Nous nous intéressons à la dynamique de particules fines dans l'air d'une pièce aérée représentée par le schéma suivant :

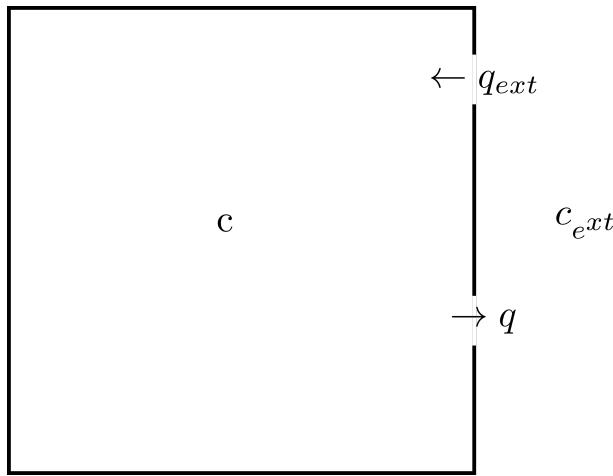


FIGURE 3.4 – Schéma d'une salle vide

Avec :

- c : concentration de particules $[c] = \frac{n_p}{V}$;
- c_{ext} : concentration de particules à l'extérieur;
- V : volume de la pièce;

— n : nombre de particules.

Nous estimons qu'il y a conservation de n , nous pouvons donc écrire :

$$n = \int_V c \, dV$$

$$\begin{aligned} \frac{dn}{dt} &= V \frac{dc}{dt} = \text{flux d'entrée} - \text{flux de sortie} \\ &= q_{ext} \cdot c_{ext} - q \cdot c \end{aligned}$$

Nous faisons l'hypothèse qu'il n'y pas d'écoulement forcé, donc $q_{ext} \approx q$:

$$\begin{aligned} \Rightarrow V \frac{dc}{dt} &= q(c_{ext} - c) \\ \Rightarrow \frac{dc}{dt} &= r(c_{ext} - c), \text{ avec } r = \frac{q}{V} \end{aligned}$$

Sachant que r est le taux de renouvellement de l'air, de dimension :

$$[r] = \frac{[q]}{[V]} = \frac{L^3/T}{L^3} = T^{-1}$$

Nous obtenons une solution de la forme suivante :

$$c(t) = Ae^{-rt} + c_{ext}$$

Déterminons A à partir de la condition initiale :

$$\begin{aligned} c(t=0) &= c_0 \Rightarrow c_0 = A + c_{ext} \\ \Rightarrow A &= c_0 - c_{ext} \\ \Rightarrow c(t) &= (c_0 - c_{ext}) e^{-rt} + c_{ext} \end{aligned}$$

3.2.2 Espace avec n personnes

Faisons maintenant la même étude avec n personnes à l'intérieur :

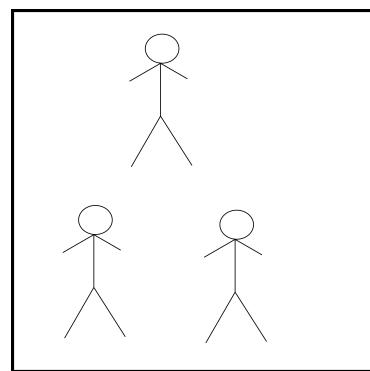


FIGURE 3.5 – Schéma d'une salle remplie

Nous admettons que chaque personne exprime 20L de CO_2 / h^{-1} , soit $0.02 \text{ m}^3 \cdot h^{-1}$

Comme le nombre de personne joue sur la concentration de CO_2 au fil du temps il nous faut ajouter un terme source à l'équation. Ce qui nous donne :

$$\frac{dc}{dt} = r(c_{ext} - c) + n \frac{f}{V}$$

Pour qu'il y ait saturation, il faut que :

$$t \rightarrow \infty, \quad c = c_{ext} + \frac{nf}{rV}$$

Alors nous obtenons :

$$c_{sat} = c_{ext} + \frac{nf}{rV}$$

Nous pouvons réécrire l'équation précédente :

$$\frac{dc}{dt} = r(c_{sat} - c), \text{ avec } c_{sat} = c_{ext} + \frac{nf}{rV}$$

Nous obtenons au final la même solution :

$$c(t) = (c_0 - c_{sat}) e^{-rt} + c_{ext}$$

À partir des mesures de CO_2 on peut alors évaluer r . Pour cela, nous allons nous mettre dans une salle close sans fenêtre. Nous serons 3 dans la pièce de volume $V = 40m^3$. Les capteurs serons mis à 3 hauteurs différentes [demander hauteur] pour qu'on puisse visualiser si la concentration en CO_2 est différente en fonction de la hauteur.

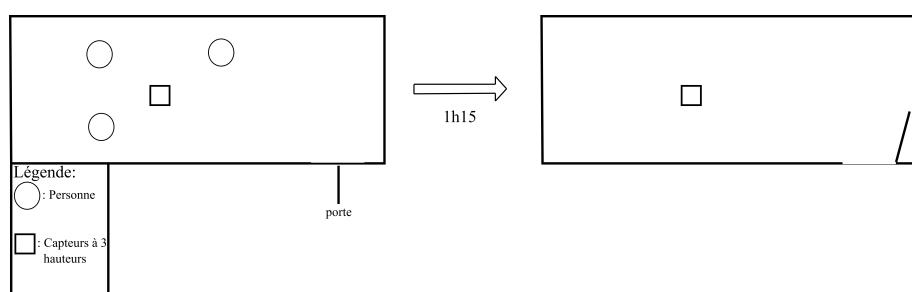


FIGURE 3.6 – Schéma expérience 1

La salle est fermée après 15 minutes pour qu'on puisse visualiser la concentration initiale. Une fois la porte fermée, nous attendons dans la pièce pendant 90 minutes. Au bout de ce temps nous ouvrons la porte, sortons et attendons 30 minutes pour que l'air circule dans la pièce. Le graphique ci-dessous montre les valeurs prises durant l'expérience.

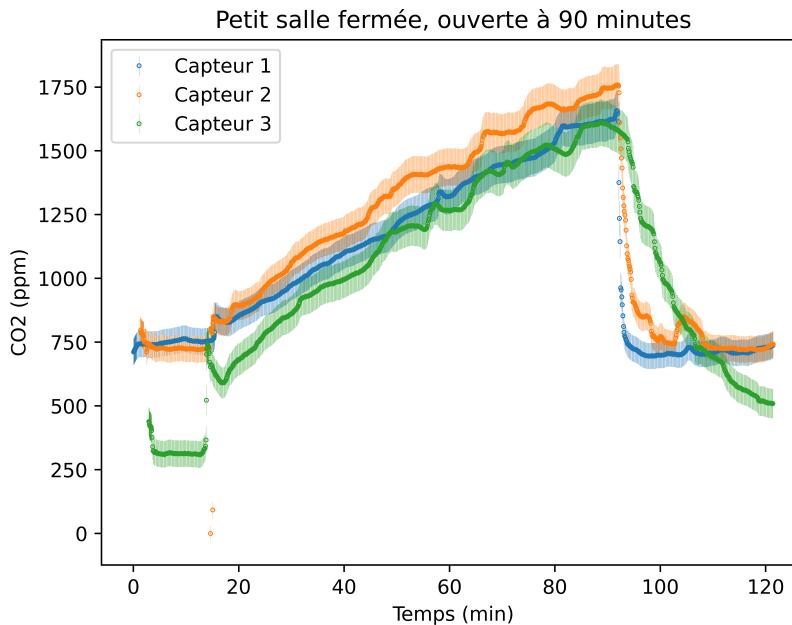


FIGURE 3.7 – Concentration de CO_2 en fonction du temps – Petite salle

Nous voyons que le nombre de particules de CO_2 est initialement de 750 ppm, notée c_{ext} . Nous pouvons déjà remarquer avec cela que le couloir 55/65 au 5e étage est déjà riche en CO_2 par rapport à l'air extérieur. Cela peut s'expliquer par le manque de fenêtre et de systèmes d'aération dans ce couloir, mais aussi par le nombre de personnes y circulant. Nous voyons qu'après la fermeture de la porte les particules de CO_2 augmentent de façon exponentielle jusqu'à atteindre 1750 ppm. Ce résultat sera utilisé dans un premier temps comme valeur de saturation en concentration de CO_2 , notée c_{sat} . Enfin, après 90 minutes et l'ouverture de la porte, la concentration en CO_2 chute très rapidement pour atteindre à nouveau la valeur initiale c_{ext} en 15 minutes. Nous avons donc :

- $c_{ext} = 750 \text{ ppm};$
- $c_{sat} = 1750 \text{ ppm};$
- $V = 40 \text{ m}^3;$
- $n = 3.$

Cela nous donne le calcul suivant :

$$c_{sat} - c_{ext} = \frac{n f}{r V}$$

Comme nous connaissons le volume V et le nombre de personnes dans la salle on peut estimer r :

$$\text{A.N : } r = \frac{3 \times 0.02}{40 \times 0.001} \text{ car } 1000 \text{ ppm} = \frac{1000}{1 \times 10^7} = 0.001 \\ \Rightarrow r = 1.5 \text{ h}^{-1}$$

Nous avons trouvé le taux de renouvellement de l'air, maintenant en calculant son inverse nous trouverons le temps de retour à la concentration initiale si nous étions restés dans la salle :

$$\frac{1}{r} = 0.67 \text{ h} \approx 40 \text{ min}$$

Il aurait donc fallu 40 minutes pour que l'air revienne à la valeur initiale de 750 ppm si nous étions restés. Cela est 3 fois plus long que lorsque nous sommes sortis. En conclusion à cette partie, il est préférable de sortir d'une pièce en l'aérant pour faire redescendre la concentration en CO_2 plutôt que de l'aérer uniquement.

3.2.3 Espace avec n personnes et purificateur d'air

Nous avons voulu vérifier l'efficacité d'un purificateur d'air dans la même pièce que l'expression précédente. Nous avons utilisé les fonctionnalités d'ionisation et de stérilisateur dans les mêmes conditions que précédemment. Ces fonctions permettent, normalement, de faciliter le renouvellement de l'air. Voici les résultats :

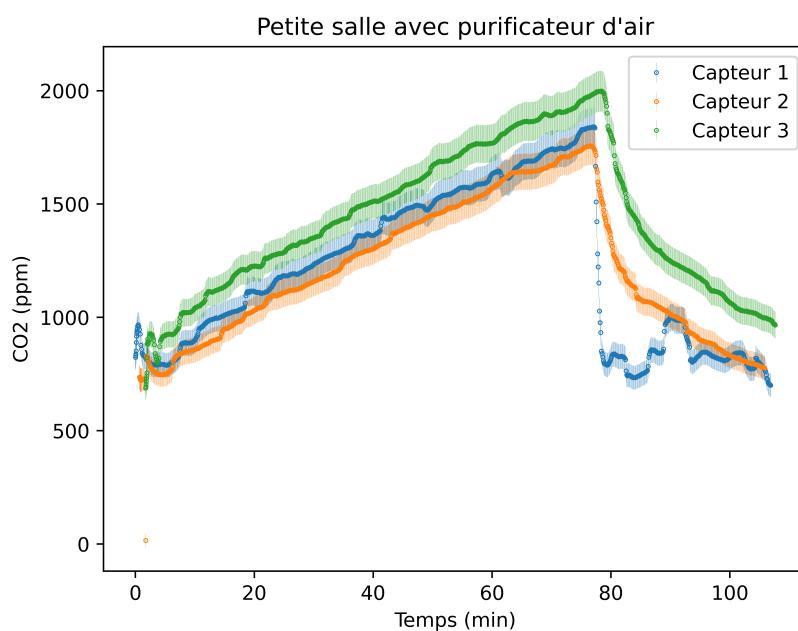


FIGURE 3.8 – Concentration de CO_2 en fonction du temps - Petite salle avec purificateur

La concentration initiale est de 800 ppm. La concentration maximale est de 2000 ppm, et après l'ouverture de la porte et notre sortie, la concentration chute jusqu'à atteindre 700 ppm en 16 minutes. Mais cela, uniquement au niveau du capteur 1, au sol. La concentration en hauteur prend plus de temps pour se renouveler et n'atteint pas un palier stable au bout des 30 minutes. Pour vérifier l'effet du purificateur nous allons comparer directement les 2 derniers graphiques entre eux :

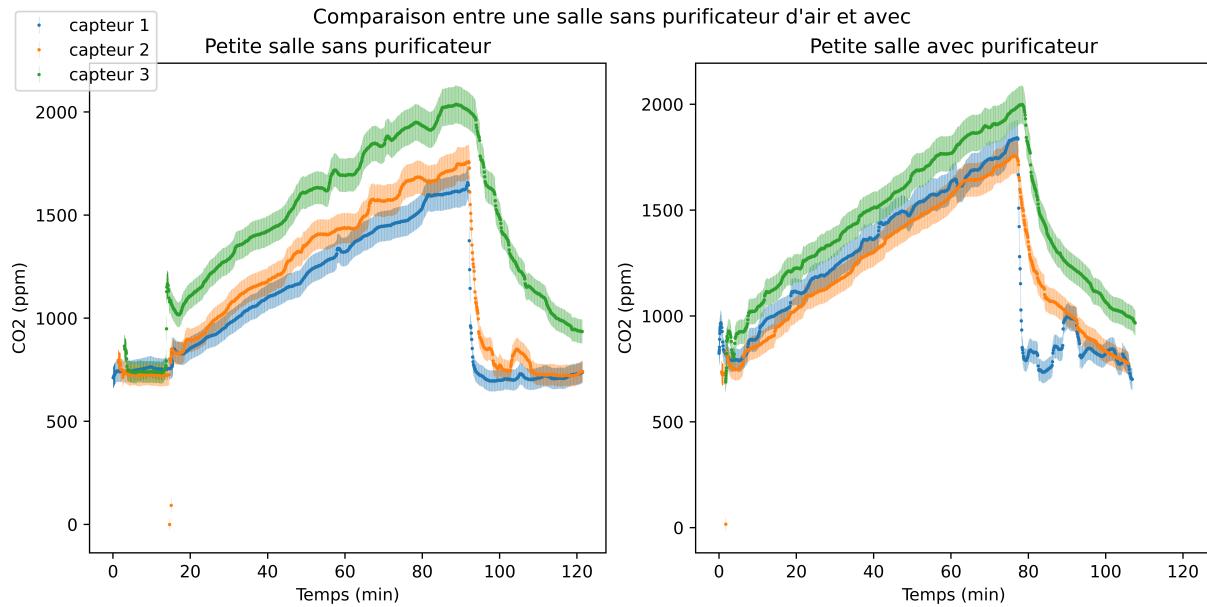


FIGURE 3.9 – Comparaison des graphiques 3.7 et 3.8

Nous pouvons voir avec cette comparaison que le purificateur d'air ne joue pas un rôle dans le renouvellement de l'air. Nous voyons même que la concentration autour du capteur 2 chute plus doucement avec le purificateur et la concentration autour du capteur 1 est beaucoup moins stable au final. Cependant, ces 2 expériences permettent de visualiser que la concentration en CO_2 est plus élevée en hauteur et que l'air s'y renouvelle moins facilement qu'au niveau du sol.

3.3 Aération d'une pièce

Nous allons maintenant analyser l'impact de différentes façons d'aérer une pièce avec plusieurs personnes à l'intérieur. L'expérience se déroule en 3 temps. Nous allons mesurer la quantité de CO_2 dans une pièce fermée, puis nous ouvrirons soit une fenêtre, soit une porte, soit les deux pour analyser l'impact de l'aération d'une pièce sur la teneur en CO_2 . Voici le protocole de l'expérience :

- Utiliser une pièce avec une fenêtre et une porte;
- Placer les 3 capteurs de mesures répartis dans la pièce avec n personnes;
- Lancer les mesures avec les capteurs pendant 3 heures;
- Ouvrir seulement la porte et attendre 30 minutes;
- Recommencer le début de l'expérience;
- Ouvrir uniquement une fenêtre et attendre 30 minutes encore;
- Recommencer une dernière fois l'expérience;
- Ouvrir la porte et la fenêtre et attendre 30 minutes.

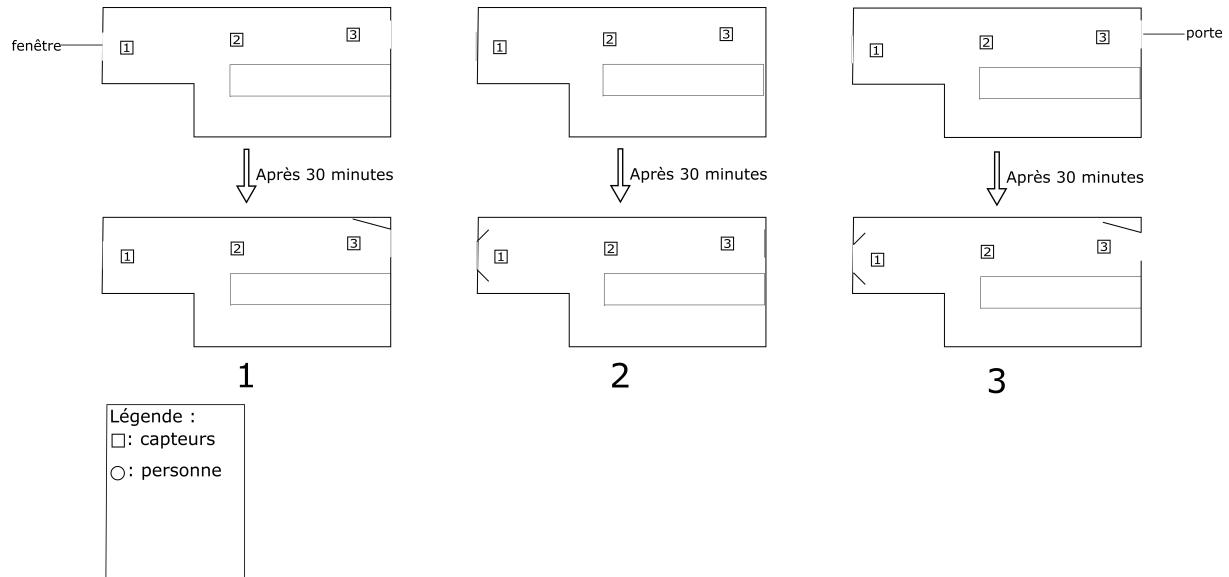


FIGURE 3.10 – Schéma des expériences sur l'aération d'une salle

Nous avons utilisé le bureau de M. FULLANA comme salle d'expérience, son volume, noté V_b est de $50.6m^3$. 3 personnes étaient dans la salle pour chaque mesure. Nous avons commencé par installer les capteurs à la même hauteur dans cet ordre :

- Capteur 1 proche de la fenêtre;
- Capteur 2 au milieu de la salle;
- Capteur 3 proche de la porte.

Cette disposition restera la même pour chaque mesure. Nous avons ensuite lancé les capteurs avec la porte ouverte pour avoir la concentration initiale. Ensuite, nous avons fermé la porte et attendu 3 heures dans le bureau. Au bout de ce temps, nous avons ouvert la porte et avons attendu 30 minutes avant d'éteindre les capteurs. Le graphique suivant montre nos mesures de concentration en CO_2 en fonction du temps :

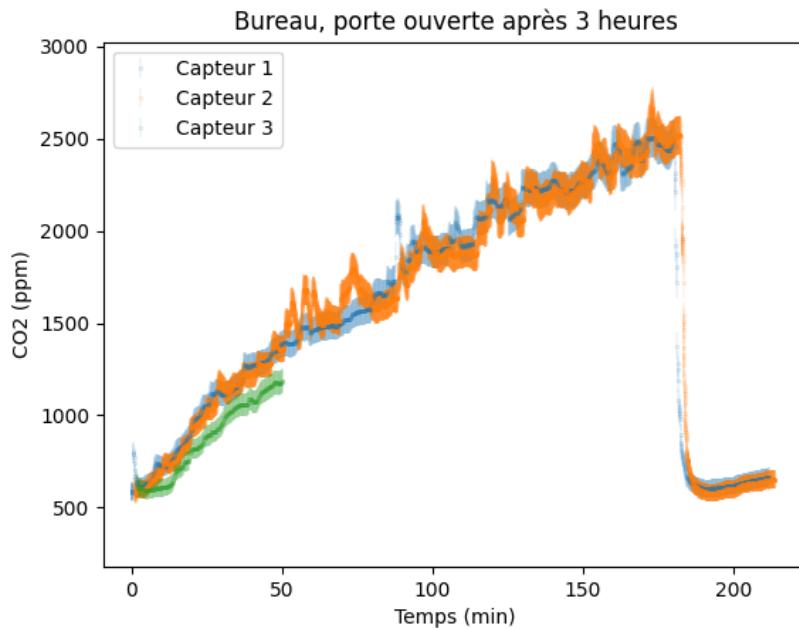


FIGURE 3.11 – Concentration de CO_2 en fonction du temps - bureau, porte ouverte

Nous voyons que le bureau de M. FULLANA est mieux aéré que le couloir 55/65 du 5e étage, puisque sa concentration en CO_2 est de 600 ppm. La concentration en CO_2 augmente fortement durant les 3 heures. Le capteur 2 a mesuré 2700 ppm comme valeur maximale et nous avons une moyenne de 2500 ppm dans les dernières minutes. Après l'ouverture de la porte nous remarquons une chute de la concentration en CO_2 , jusqu'à atteindre à nouveau les environs de 500 ppm au bout de 20 minutes. Nous pouvons aussi visualiser que le capteur 1 mesure la chute de concentration en CO_2 légèrement avant le capteur 2. Cela peut nous aider à visualiser l'écoulement de l'air et sa vitesse, connaissant la distance entre les 2 capteurs. Après cette première séance de mesure, nous avons décidé de diminuer le temps d'attente avant l'aération. La durée d'exposition à une concentration en CO_2 supérieure à 1000 ppm était trop longue pour notre santé.

Nous recommençons l'expérience en ouvrant uniquement la fenêtre cette fois-ci et en attendant juste une heure.

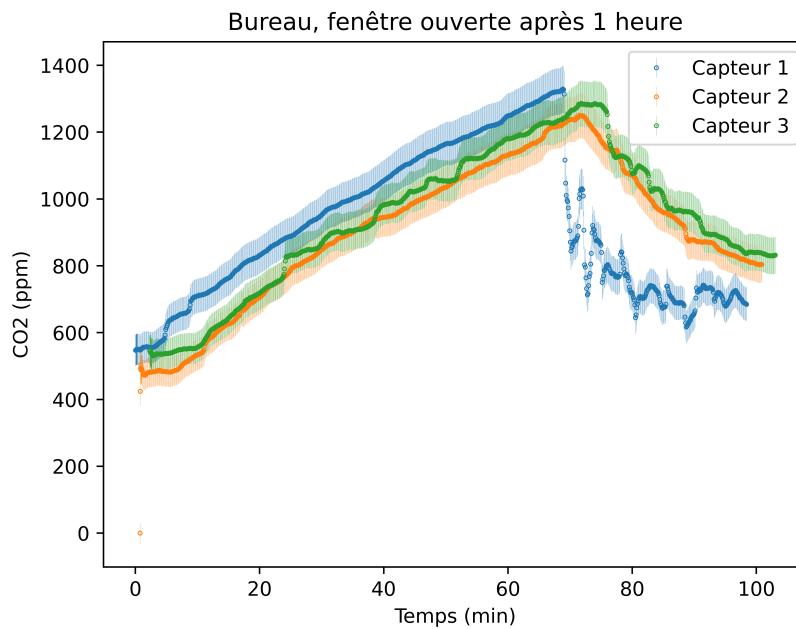


FIGURE 3.12 – Concentration de CO_2 en fonction du temps - bureau, fenêtre ouverte

La concentration initiale est aux alentours de 500 ppm, tandis que la concentration maximale est de 1320 ppm. Après l'ouverture de la fenêtre, la concentration autour du capteur 1 chute beaucoup plus rapidement que celle des deux autres capteurs. La concentration du capteur 1 arrive même à une limite aux alentours de 700 ppm au bout de 16 minutes. La concentration des 2 autres capteurs chute plus lentement et n'a pas encore atteint une valeur limite au bout des 30 minutes. Si nous prolongeons la courbe des capteurs 2 et 3 nous obtenons une concentration limite autour de 750 ppm.

Enfin, pour la dernière expérience, nous ouvrons la porte et la fenêtre au bout d'une heure encore une fois.

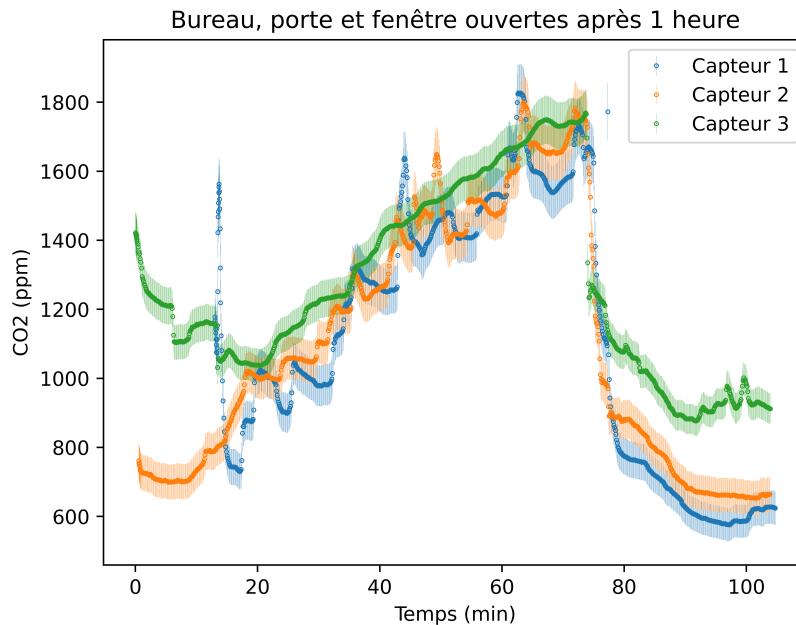


FIGURE 3.13 – Concentration de CO_2 en fonction du temps - bureau, porte et fenêtre ouvertes

Les capteurs ont eu un problème lors de l'initialisation mais nous pouvons estimer la concentration initiale à 700 ppm en fonction du capteur 2. Cette fois-ci nous atteignons 1800 ppm de concentration en CO_2 . Après l'ouverture de la fenêtre et de la porte, nous voyons que les concentrations des capteurs 1 et 2 suivent la même courbe avec des valeurs proches, dont une concentration limite autour de 650 ppm en 14.5 minutes. La concentration du capteur 3 suit également la même allure mais atteint une concentration limite de 1000 ppm, soit 300 ppm de plus que les 2 autres capteurs. Cela peut s'expliquer par le fait que le capteur 3 soit proche de là où nous nous tenons durant l'expérience.

Pour conclure, nous avons vu que les différentes façons d'aérer une pièce ne sont pas toutes aussi efficaces. Le moyen le plus efficace est d'ouvrir la fenêtre et la porte en même temps. Cela s'explique par l'écoulement qui passe dans la pièce par ces ouvertures. L'écoulement emmène les particules de CO_2 avec lui, diminuant ainsi la concentration de la pièce. De plus, sortir de la pièce pendant l'aération diminue aussi le temps de renouvellement de l'air. Pour un bureau sans fenêtre, il est donc conseillé de sortir une quinzaine de minutes pour que l'air se renouvelle entièrement. Pour un bureau avec fenêtre, quinze minutes avec la porte et la fenêtre ouvertes suffisent aussi, mais la personne peut rester à l'intérieur.

Conclusion

En conclusion, nos études sur les particules fines et leurs caractéristiques nous ont permis de ressortir plusieurs points importants.

Nous avons utilisé un modèle simplifié de la chute d'une particule fine à l'aide de billes en verre tombant dans de la glycérine. Ce modèle nous a permis de comprendre comment une particule se comporte dans l'air et comment elle peut sortir d'une salle. Une particule fine peut rester plusieurs jours en suspension dans l'air d'une pièce si elle n'est pas aérée. Alors que si une salle est aérée la particule sera emportée par le fluide, permettant à l'air de se renouveler. Il est donc très important d'aérer très souvent une pièce. En fonction de la taille de la salle et du nombre de personnes à l'intérieur il est préférable de l'aérer toutes les heures.

Nous avons aussi remarqué, à l'aide de nos capteurs placés dans une salle close, qu'il était plus efficace de sortir de la pièce lors de l'aération plutôt que d'y rester. Cela est d'autant plus efficace pour un bureau qui n'aurait pas de fenêtre. Selon nos données, sortir de la pièce permettrait de réduire le temps de renouvellement de 3 fois.

Enfin, notre dernière expérience se déroulait dans un bureau où on faisait augmenter la concentration en CO_2 avant d'aérer de plusieurs façons. Nous avons vu qu'il était plus facile de renouveler l'air en ouvrant porte et fenêtre, plutôt qu'une seule des deux ouvertures.

Ce projet nous a permis de faire un pas vers l'ingénierie et le monde du travail. Cela nous a demandé d'être organisés, sérieux, ponctuels et coopératifs. Nous laisser en autonomie sur notre projet nous a forcés à être matures. Nous avons appris dans plusieurs domaines scientifiques tout en développant notre esprit scientifique, mais nous avons aussi su utiliser nos compétences acquises cette année, notamment en informatique et mécanique.

Bibliographie

- [1] V. RAFFORT, « Modélisation des particules : Participation à Eurodelta et étude au voisinage d'une raffinerie », Theses, Université Paris-Est, juin 2017. adresse : <https://pastel.archives-ouvertes.fr/tel-01625284>.
- [2] L. JISAAC9 Mieszko the first. « Visualisation des catégories de particules en suspension dans l'air ou aéroportées (contaminant biologiques, ou particulaires minéraux ou organiques, ou gazeux)... par nature et taille (en micromètres ; m) ». (21 May 2017), adresse : <https://commons.wikimedia.org/w/index.php?curid=59147824>.
- [3] A. RIHAM JABER, M. DEJAN et U. MARCELLA, « The Effect of Indoor Temperature and CO₂ Levels on Cognitive Performance of Adult Females in a University Building in Saudi Arabia », *Energy Procedia*, t. 122, p. 451-456, 2017, CISBAT 2017 International Conference Future Buildings Districts – Energy Efficiency from Nano to Urban Scale, ISSN : 1876-6102. DOI : <https://doi.org/10.1016/j.egypro.2017.07.378>. adresse : <https://www.sciencedirect.com/science/article/pii/S187661021732982X>.
- [4] seeed STUDIO. « SD Card shield V4.0 ». (s. d.), adresse : https://wiki.seeedstudio.com/SD_Card_shield_V4.0/ (visité le 17/05/2022).
- [5] ——, « Base Shield V2 ». (s. d.), adresse : https://wiki.seeedstudio.com/Base_Shield_V2/ (visité le 17/05/2022).
- [6] « Grove - CO₂ & Temperature & Humidity Sensor (SCD30) - Seeed Wiki ». (s. d.), adresse : https://wiki.seeedstudio.com/Grove-CO2_Temperature_Humidity_Sensor-SCD30/ (visité le 16/05/2022).
- [7] V. GROUP. « CARTE DE DÉVELOPPEMENT ATmega32u4 LEONARDO ». (s. d.), adresse : <https://www.velleman.eu/products/view?id=435502> (visité le 19/05/2022).
- [8] J. H. PARK, « Atlas of Infrared Absorption Lines », nov. 1977. adresse : <https://core.ac.uk/download/pdf/42876018.pdf> (visité le 18/05/2022).
- [9] CO2METER. « How does an NDIR CO₂ Sensor Work? » (19 jan. 2022), adresse : <https://www.co2meter.com/blogs/news/how-does-an-ndir-co2-sensor-work#:~:text=NDIR%20is%20an%20industry%20term,of%20an%20IR%20light%20detector>. (visité le 17/05/2022).
- [10] SPARKFUN. « CO Humidity and Temperature Sensor - SCD30 ». (s. d.), adresse : <https://www.sparkfun.com/products/15112#:~:text=The%20SCD30%20from%20Sensirion%20is,to%20set%20the%20current%20altitude> (visité le 20/05/2022).
- [11] seeed STUDIO. « Grove - LoRa Radio ». (s. d.), adresse : https://wiki.seeedstudio.com/Grove_LoRa_Radio/ (visité le 11/04/2022).
- [12] E. CAN et B. BRAUN-DELVOYE. « PEI ». (s. d.), adresse : <https://github.com/erdicn/PEI> (visité le 20/05/2022).
- [13] « Grove - LoRa-E5 - Seeed Wiki ». (s. d.), adresse : https://wiki.seeedstudio.com/Grove%5C_LoRa%5C_E5%5C_New%5C_Version/ (visité le 17/05/2022).
- [14] « Introduction to LoRa - Send data between two Arduino using LoRa ». (s. d.), adresse : <https://www.electronics-lab.com/project/introduction-lora-send-data-two-arduino-using-lora/> (visité le 17/05/2022).
- [15] J.-M. COURTY, É. KIERLIK et B. SEMIN, « Bien Ventiler Pour Bien Respirer », *Pour la Science*, t. 518, p. 88-90, déc. 2020.
- [16] SENSIRION. « SCD30 ». (s. d.), adresse : <https://sensirion.com/products/catalog/SCD30/> (visité le 16/05/2022).
- [17] SEMTECH. « Applications ». (s. d.), adresse : <https://www.semtech.com/lora/lora-applications> (visité le 19/05/2022).

- [18] L. SLATS. « A Brief History of LoRa®: Three Inventors Share Their Personal Story at The Things Conference ». (8 jan. 2020), adresse : <https://blog.semtech.com/a-brief-history-of-lora-three-inventors-share-their-personal-story-at-the-things-conference> (visité le 19/05/2022).

Les deux séances de bibliographie nous ont permises d'être plus précis dans nos recherches. Nous avons appris à utiliser le navigateur de la Sorbonne en ligne et il fût notre première zone de recherche. Nous avons aussi appris à éviter certaines sources peu fiable, notamment sur internet. De plus, l'outil Zotero nous a permis de recenser nos sources communes et de les exporter facilement pour les utiliser dans notre rapport. Notre démarche nous a fait d'abord regarder les revues, thèses ou rapport scientifique récent. Nous avons ensuite étendu nos recherches à des articles plus anciens. Pour les appareils utilisés nous avons décidé de prendre les informations venant directement des sites des fournisseurs. Nous avons aussi favorisé de prendre nos sources sur internet car les catalogues scientifiques en ligne nous donnent facilement un code BibTex pour citer sur nos fichiers L^AT_EX

Annexes

Nos Codes

Codes Arduino

```

1 //Arduino Data Code
2 #include <SCD30.h> // CO2 sensor header
3 #include "SCD30.h"
4 #include <SPI.h> // Serial Peripheral Interface
5 #include <SD.h> //SD card shield and sd card
6
7 File myFile;
8
9 // defines the arduino architecture type
10 #if defined(ARDUINO_ARCH_AVR)
11     #pragma message("Defined architecture for ARDUINO_ARCH_AVR.")
12     #define SERIAL Serial
13 #elif defined(ARDUINO_ARCH_SAM)
14     #pragma message("Defined architecture for ARDUINO_ARCH_SAM.")
15     #define SERIAL SerialUSB
16 #elif defined(ARDUINO_ARCH_SAMD)
17     #pragma message("Defined architecture for ARDUINO_ARCH_SAMD.")
18     #define SERIAL SerialUSB
19 #elif defined(ARDUINO_ARCH_STM32F4)
20     #pragma message("Defined architecture for ARDUINO_ARCH_STM32F4.")
21     #define SERIAL SerialUSB
22 #else
23     #pragma message("Not found any architecture")
24     #define SERIAL Serial
25 #endif
26
27
28 void setup(){
29     Serial.begin(9600);
30     while(!Serial){
31         ; //wait to connect
32     }
33     Serial.print("Initializing SD card...");
34     if(!SD.begin(4)){
35         Serial.println("initialization failed!");
36         while(1);
37     }
38     Serial.println("initialization done.");
39
40     // open file. note that only one file can be open at a time
41     myFile = SD.open("test.txt", FILE_WRITE);
42
43     //if file opened write in it
44     if(myFile){
45         Serial.print("Writing test.txt...");
46         myFile.println("testing 1, 2, 3."); // test writes on the file
47         myFile.close(); //close the file
48         Serial.println("done.");
49     }
50     else{
51         //if the file didnt open write error
52         Serial.println("error opening test.txt");

```

```

53 }
54
55 //re-open the file for reading
56 myFile = SD.open("test.txt");
57 if(myFile){
58     Serial.println("test.txt");
59
60     //read until there is nothing in it
61     while(myFile.available()){
62         Serial.write(myFile.read());
63     }
64     //close the file
65     myFile.close();
66 }
67 else {
68     // if the file didnt open print an error
69     Serial.println("error opening test.txt");
70 }
71 Wire.begin();
72 SERIAL.begin(115200);
73 SERIAL.println("SD30 Raw Data");
74 scd30.initialize(); //initialises the CO2 sensor
75 }

76 unsigned long myTime;

77 void loop(){
78     float result[3] = {0}; // initialise an array for sensor data
79     myTime = millis(); // gets the time in miliseconds from which we
80     started the arduino
81
82     if(scd30.isAvailable()){
83         scd30.getCarbonDioxideConcentration(result); // puts the values
84         from the sensor to array {ppm, temperature, humidity}
85
86         myFile = SD.open("mesures.txt", FILE_WRITE); // if the file is
87         non existant it creates one and then opens it. If it already exists
88         it only opens it
89         if(myFile){
90             // the serial and file prints are such ppm temperature
91             humidity miliseconds
92             //Serial.print("1 "); // if there are more than one sensor
93             here could be used to give each sensor an identifier
94             Serial.print(result[0]); // prints in serial the ppm
95             Serial.print(" ");
96             Serial.print(result[1]); // prints to serial the temperature
97             Serial.print(" ");
98             Serial.print(result[2]); // prints to serial the humidity
99             Serial.print(" ");
100            Serial.print(myTime); // prints to serial the time from the
101            start as miliseconds
102            Serial.println("\t"); // goes to newline
103
104            myFile.print(result[0]); // writes on the file the ppm
105            myFile.print(" ");
106            myFile.print(result[1]); // writes on the file the
107            temperature
108            myFile.print(" ");

```

```
103     myFile.print(result[2]); // writes on the file the humidity
104     myFile.print(" ");
105     myFile.print(myTime); // writes on the file the time in
106     milliseconds
107     myFile.println("\t");
108     myFile.close(); // closes the file so that way if we unplug
the arduino the unsaved values are not lost and saves them
109
110     } else{ //if the file doesn't open it prints out an error message
111         Serial.println("error opening test.txt");
112     }
113     delay(2000); //the delay between each measurement the minimum is 2000
114 }
```

Listing 1 – Arduino Data Code

```

1 //Arduino Calibration Code
2 #include <Adafruit_SCD30.h>
3
4 Adafruit_SCD30    scd30;
5 int i=1;
6
7 void setup(void) {
8     Serial.begin(115200);
9     while (!Serial) delay(10);      // will pause Zero, Leonardo, etc until
10    serial console opens
11
12    Serial.println("Adafruit SCD30 test!");
13
14    // Try to initialize!
15    if (!scd30.begin()) {
16        Serial.println("Failed to find SCD30 chip");
17        while (1) { delay(10); }
18    }
19    Serial.println("SCD30 Found!");
20
21    Serial.print("Measurement Interval: ");
22    Serial.print(scd30.getMeasurementInterval());
23    Serial.println(" seconds");
24 }
25
26 void loop() {
27    if (scd30.dataReady()){
28        //Serial.println("Data available!");
29        Serial.print(i*2); Serial.println(" seconds");
30        Serial.print("Forced Recalibration reference: ");
31        Serial.print(scd30.getForcedCalibrationReference());
32        Serial.println(" ppm");
33
34        if (!scd30.read()){ Serial.println("Error reading sensor data");
35        return; }
36        // serial prints the sensor values
37        Serial.print("T, ");
38        Serial.print(scd30.temperature);
39        Serial.print("C,");
40
41        Serial.print("RH, ");
42        Serial.print(scd30.relative_humidity);
43        Serial.print("%,");
44
45        Serial.print("CO2, ");
46        Serial.print(scd30.CO2, 3);
47        Serial.print(" ppm,");
48        Serial.println(",");
49    } else {
50        //Serial.println("No data");
51    }
52
53    delay(2000);
54    i = i+1;
55    if (i == 30){ //waits 60 seconds to initialise to the new value
56        Serial.print("Forced Recalibration reference: ");

```

```
57     Serial.print(scd30.getForcedCalibrationReference()); // serial prints  
58     which value we choose  
59     Serial.println(" ppm");  
60     scd30.forceRecalibrationWithReference(440); // here put the value  
61     that we want to take as reference  
62     Serial.print("Forced Recalibration reference: ");  
63     Serial.print(scd30.getForcedCalibrationReference());  
64     Serial.println(" ppm");  
}
```

Listing 2 – Calibration du sensor

Codes pour les graphs et données

```

1 from matplotlib import pyplot as plt
2 #import numpy as np
3
4 def readlist(file) : return list(map(float,file.readline().split()))
5
6 def transformToMinutes(seconds , minutes = 0, hours = 0):
7     return (seconds / 60) + minutes + (hours * 60)
8
9 def transformToHours(seconds , minutes = 0, hours = 0):
10    return (seconds / 3600) + (minutes/60) + (hours)
11
12 def millisToHours(milliseconds): return milliseconds /(1000 * 60 * 60)
13 def millisToMinutes(milliseconds): return milliseconds /(1000 * 60 )
14
15 def calculCO2SensorIncertitude(ppm):
16     # calculates the error of sensors given by the manufacturer
17     #incertitude +- 30 ppm + 3%
18     return 30 + ((3*ppm) / 100)
19
20 def doneesPourGraphs(filename : str)-> list[list]:
21     file = open(filename , 'r') #opens file
22     readedlist = readlist(file) # reades the line
23     ppm, c, h, time= readedlist # puts the readed line to values
24     #initialise the lists
25     list_ppm = []
26     times = []
27     temperatures = []
28     err_ppm = []
29     #entter in the transformToMinutes each Arduino start time
30     startC1 = transformToMinutes(0)
31     startC2 = transformToMinutes(40)
32     startC3 = transformToMinutes(42,1)
33     while readedlist != []: # until we finish the file
34         ppm, celsius, h, time = readedlist
35         time = millisToMinutes(time)
36         temperatures.append(celsius)
37         list_ppm.append(ppm)
38         err_ppm.append(calculCO2SensorIncertitude(ppm))
39         #Adds the time of each sensorc because we start them at
40         #intervals
41         if filename[-5] == "3":
42             times.append(time + startC3)
43         if filename[-5] == "2":
44             times.append(time + startC2)
45         if filename[-5] == "1":
46             times.append(time + startC1)
47     readedlist = readlist(file)
48     return [list_ppm, times, temperatures, err_ppm]
49
50 def plotGraph(nb_graph : int, filename : str,list_ppm,times,
51 temperatures, err_ppm):
52     plt.errorbar(times, list_ppm, yerr = err_ppm, elinewidth=0.1,
53 markeredgecolor='black', markeredgewidth=2, fmt="o", markersize='0.5')
54     plt.xlabel("Temps(m)")
55     plt.ylabel("CO2 (ppm)")
56
57 def main(filenames : list[str]):
```

```

55     # 1 <= len(filenames) <= 9 (Single argument to subplot must be a
56     three-digit integer) plt.subplot(nb_graph)
57     systematicErreurDuCapteur3 = 426.36
58
59     #if we do a calibration we change the systematic error of the sensor
60     nb_3
61     """
62     moyennes = []
63     for i in range(len(filenames)):
64         list_ppm,times, temperatures, err_ppm = donneesPourGraphs(
65             filenames[i])
66         moyennes.append((sum(list_ppm)/len(list_ppm)))
67     moyenneDe1et2 = (moyennes[0] + moyennes[1]) / 2
68     systematicErreurDuCapteur3 = moyennes[2] - moyenneDe1et2
69     print(systematicErreurDuCapteur3)
70     """
71
72     for i in range(len(filenames)):
73         list_ppm,times, temperatures, err_ppm = donneesPourGraphs(
74             filenames[i])
75         if i == 2: # if we are at the saved file with the sensor number
76             3
77             for j in range(len(list_ppm)): #we correct the systematic
78                 error
79                 list_ppm[j] = list_ppm[j] - systematicErreurDuCapteur3 # #
80                 systematicErreurDuCapteur3
81                 plotGraph(i+1, filenames[i],list_ppm,times, temperatures,
82                 err_ppm)
83             else:
84                 plotGraph(i+1, filenames[i],list_ppm,times, temperatures,
85                 err_ppm)
86
87             sensor_legend = ["Capteur 1","Capteur 2","Capteur 3"]
88             plt.legend(sensor_legend)
89             plt.title("Petit Salle Porte ferme et ouverture apres un certain
90             temps Avec purificateur")
91             plt.savefig("Petit salle avec purificateur",format="pdf", dpi = 900)
92             #Saves the plot
93             plt.show()
94
95 filenames = ["CLOSED1.TXT","CLOSED2.TXT","CLOSED3.TXT"] #enter the files
96     with the sensor values
97 main(filenames)
98
99

```

Listing 3 – Prendre les données des fichiers