PSE : Mesurer la bioluminescence des algues

Crochet Morgane, Bailliet Pierre-Eloi, Provost Matéo

I - Culture des algues

Matériel et produits nécessaires :

- **Algues souches**, commandées sur internet
  pyrocystis fusiformis (PF): https://zoelux.fr/pyrocystis_fusiformis.html
  pyrocystis lunula (PL) : https://bioglow.eu/shop/en/3-bioluminescent-algae
- **Milieu de culture** : https://bioglow.eu/shop/en/4-algae-nutrients
- **Pipettes stériles** (3 par repiquage : 1 pour les PL de 50 mL, 1 pour les PF de 50 mL et 1 pour le milieu de culture 5mL)
- **Lampe** Led ne chauffant pas trop (970 lumen, située à 10-20 cm), respecter les consignes données par le fournisseur : https://bioglow.eu/shop/en/content/7-algae-care
- **Bec Bunsen,** ou réchaud de camping
- **Contenants stériles** de 50mL
- **Pipetboy** classique
- **Prise programmable**
- **Boîte noire** (boîte en carton de 50x50x50 cm étanche à la lumière, scotchée à la paillasse, peinte en noir à l'intérieur et recouverte d'un tissu occultant, noir de préférence)

- ***Maintien d'un cycle jour/nuit artificiel :***

Objectif : Contrôler le **cycle circadien** des algues afin de faire correspondre le début/milieu de leur nuit avec le moment de nos expériences (car les algues bioluminescent la nuit).
Protocole : On construit une **boîte noire** dans laquelle on introduit une lampe. La **lampe** qui permet la photosynthèse est **allumée la nuit et le matin uniquement** (01h-13h) afin d'inverser le cycle circadien : nos expériences commençaient à partir de 14h. Nous utilisons pour cela un timer.
La lampe ne doit pas trop chauffer : respecter les indications ci-dessus.

- ***Repiquage des algues :***

Objectif : Afin que les algues puissent avoir de l'espace et se multiplier, il faut les repiquer. La fréquence conseillée est **une fois toutes les 3 semaines**. Il est conseillé de réaliser un repiquage à 90% de solution d'algues et 10% de milieu de culture nutritif.
Protocole : Pour repiquer une fiole de 50 mL d'algues : On se place sous **atmosphère stérile** (bec bunsen allumé, se placer dans un rayon de 10 cm autour du bec bunsen). On ouvre une **pipette stérile de 50 mL**, on la place sur le **pipetboy** et on prélève 45 mL de la solution mère d'algues (à repiquer) que l'on verse dans une nouvelle fiole stérile de 50 mL. On complète avec 5mL de nutriments.

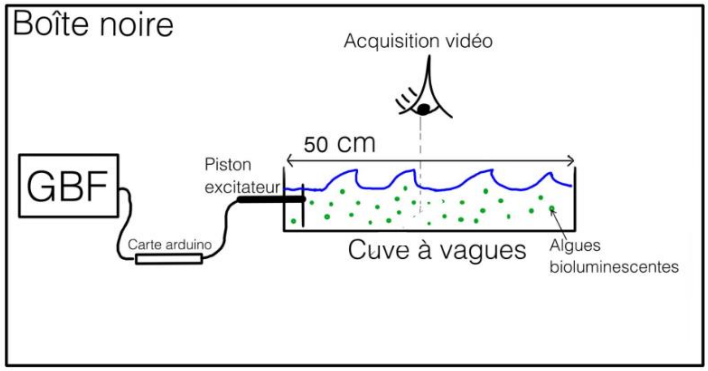
II - Dispositif expérimental et caractérisation

Matériel et produits nécessaires :

- une **boîte noire** pour réaliser les expériences (même principe que dans la partie I)
- une **cuve en plexiglass** (parallélépipède de 30x5x5 cm, assemblée à la colle à plexiglas et étanchéifiée à la colle chaude, plexiglas de 2 mm d'épaisseur)
- de la **résine de silicone souple**
- un **piston électrique** 15V, muni d'une **plaque de plexiglass** (plaque à laquelle on a collé un roulement à bille pour s'affranchir de mouvements parasites du piston. Le roulement à bille est vissé sur le piston directement) qui pousse l'eau dans la cuve et **génère les vagues**.
- une **carte arduino**
- les **solutions d'algues** préparées (partie I)
- **cylindre en plexiglass**
- **pailles**
- **colle chaude**
- **résine de silicone** (avec un module d'Young pas trop grand, prendre la résine orthodontique utilisée pour les moulages par exemple)
- un **capteur de lumière en niveaux de gris** (2000x1000 pixels environ)

- ***Mise en place du dispositif :***

Placer le capteur à 20 cm de la cuve. Penser à masquer toute source de lumière sur les appareils électroniques (avec du scotch noir et des draps noirs pour notre part).



- ***Caractérisation des vagues générées par le piston***

Objectif : Nous cherchons à **caractériser le système** afin de savoir quels paramètres appliquer sur le piston excitateur pour avoir une amplitude et fréquence de vague spécifique.

Le piston est contrôlé un par un **programme Arduino** (annexe 1) qui permet de **définir le temps de déplacement du piston** (donc la **fréquence** des oscillations), ainsi que le temps d'arrêt du piston entre chaque battement (que l'on prend nul pour toutes les acquisitions). Il est également relié au GBF. Par expérience, on constate que changer la tension change la

vitesse du piston (et par conséquent son **amplitude de mouvement** pendant un temps donné).

Protocole : On fait des acquisitions vidéos (avec un **téléphone**) des vagues dans la cuve, en dehors de la boîte noire pour plus de facilité et en faisant varier les caractéristiques du piston. Nous mettons une règle dans le même plan que la cuve pour définir l'échelle. Les expériences sont faites avec le cylindre dans la cuve. On place le téléphone de façon fixe parallèlement à la cuve.
On fait varier le voltage en entrée du piston entre **5V et 15V** (valeur maximale du GBF). Faire aussi varier le temps d'un aller-retour du piston à l'aide du programme arduino. On fait varier le temps d'un aller de **100ms à 400ms** (soit l'aller-retour de 200ms à 800ms). Ainsi on réalise une vidéo par couple vitesse/fréquence de déplacement du piston.
Par la suite on fait un pointage vidéo afin d'obtenir l'amplitude de la vague résultante.
→ Nous savons ainsi pour chaque caractéristique expérimentale du piston que nous utilisons, la **caractéristique des vagues** correspondant.

Pointage vidéo pour la mesure de l'amplitude des vagues :

Logiciel utilisé : PyMecavideo
On ouvre une vidéo correspondant à un voltage et un temps précis.

- Cliquer sur définir l'échelle. Pointer 5cm sur la règle placée sur la vidéo pour définir l'échelle.
- Se déplacer image par image pour trouver une vague bien exploitable (pas de vague retour perturbant le système, pas de mouvements extérieurs perturbant la mesure).
- Se stopper sur une image où la vague passe par un maximum.
- Cliquer sur démarrer. Pointer le haut de la vague puis pointer le bas de la cuve (limite cuve-table). Cette mesure définit le **niveau d'eau avec la vague.**
- Faire défiler les images jusqu'à ce que la surface de l'eau redevienne plate devant le piston (i.e plus de vague générée à ce moment). Pointer la surface de l'eau et la frontière cuve-table à nouveau. Cette mesure définit le **niveau normal** d'eau dans la cuve.
- Cliquer sur le tableau de valeurs. Relever les coordonnées y correspondantes aux pointages faits.
- On rentre ensuite nos données dans un tableau Excel. On y insère la **tension** et le **temps** appliqués au piston. **La vitesse résultante** du piston (d'après l'étalonnage précédent).
- On calcule **l'amplitude finale** comme étant l'amplitude de la vague par rapport au niveau normal (**niveau d'eau avec la vague - niveau normal**). On finit par calculer **l'amplitude du piston** A = Vitesse * Temps.

→ Obtention de l'amplitude de la vague en fonction de la fréquence (inverse du temps) et de l'amplitude du piston.

- ***Obtention d'ondes non stationnaires***

Objectif : Afin d'obtenir des vagues propagatives (pour reproduire au mieux les vagues de l'océan) et non une onde stationnaire, on crée un *"atténuateur de vagues"* qui absorbe l'énergie des vagues en bout de leur course. Comme la berge par exemple.

Protocole : Pour cela, on utilise la résine de silicone. On coupe des pailles qui font la hauteur de la cuve dans lesquelles on moule nos tubes de silicone. On les fixe ensemble grâce à une plaque de plexiglas dans laquelle on fait des entailles pour laisser passer les cylindres en résine souple. On place cette "forêt" de résine au bout de notre cuve à onde.

III - Acquisitions

Matériel et produits nécessaires :

- le **dispositif expérimental du II**
- 200 mL **d'algues préparées en I**
- une **mémoire** où stocker les données

- ***Préparation d'une acquisition***

Objectif : Régler tous les paramètres d'acquisition (dispositif + caméra) avec de lancer l'expérience

Protocole : Pour **une acquisition** (*i.e. pour une fréquence et une amplitude de vague données*), on prépare **200 mL d'algues** (PF ou PL) que l'on place dans la cuve en plexiglas puis dans la boîte noire. Il faut ensuite les **laisser reposer dans le noir** (1h est l'idéal, 10-20 min suffisent pour avoir une bonne luminescence au début au moins). Préparer les bons paramètres pour le piston (fréquence, tension aux bornes).

Régler la caméra avec le logiciel adapté à la caméra (nous utilisons **Pylon Viewer**) :

- régler le ***fps***. Prendre tel qu'il y ait environ un aller de piston par image (on fait ainsi une moyenne des événements sur une demi période des mouvements).
- prendre un ***temps d'exposition*** assez long (pour pouvoir moyenner sur tout ce qu'il se passe dans la cuve pendant une demi-période), mais moins long que le temps entre 2 images
- centrer l'image sur la cuve (ou une partie de la cuve), garder la même taille d'une acquisition à l'autre
- prendre les images en **mono12** (12 bits par pixel,une couleur, noir)
- prendre plutôt une **suite d'images** stockées dans le même dossier qu'une vidéo, c'est plus facile à analyser ensuite. Les enregistrer à un format pris en charge par ImageJ (**.bmp** fonctionne bien)
- prendre un temps d'acquisition de 2 min au total

Afin de pouvoir analyser le bruit environnant, **pour chaque acquisition, démarrer l'acquisition avant l'activation du piston** (une dizaine d'images suffit pour le bruit).

IV - Analyse des données

Matériel et produits nécessaires :

- Logiciel **d'analyse d'images ImageJ**
- Les **images acquises précédemment**
- Un exécuteur Python, ou MatLab pour tracer les résultats obtenus

- ***Traitement des images brutes***

Objectif : Analyser les acquisitions vidéos sur imageJ afin de quantifier l'intensité de bioluminescence observée. Il faut trouver un paramètre clé à mesurer sur imageJ pour cela.
Protocole : On vérifie au préalable sur chaque acquisition si l'on observe de la luminescence, pour tracer un diagramme binaire (oui/non; rouge/vert) d'observation de la luminescence (voir ci-dessous le tracé des diagrammes, et l'Annexe 2).

Puis on analyse les données avec le logiciel **ImageJ**.
On veut un paramètre d'analyse de la luminescence. Pour ça on va utiliser la formule ci-dessous :

$$g = \overline{g_{tot,x,t}} - \overline{g_{blanc,x,0}}$$

Où :

g est **le niveau de gris, le paramètre de luminescence que l'on trace dans nos graphiques** quantitatifs en niveau de bleu
$\overline{g_{tot,x,t}}$ est la moyenne des niveaux de gris sur l'espace (tous les pixels de chaque image) et sur le temps (sur toutes les images obtenues).
$\overline{g_{blanc,x,0}}$ est la **moyenne des niveaux de gris** sur l'espace (tous les pixels de chaque image) et sur les premières images, tant que le piston n'est pas activé. On **repère l'image charnière à l'oeil nu**

Pour ça, **on ouvre les images en *un virtual stack* sur ImageJ**. (*File>Import>Image Sequence*)
Sélectionner la première image de votre acquisition dans la fenêtre qui s'ouvre. Cocher *Use Virtual Stack*, appuyer sur *Ok*.

On **crée ensuite une moyenne des intensités de toutes les images acquises**. (*Image>Stacks>Z-project*)
Sélectionner *Average Intensity* dans le menu déroulant *Projection Type,* cliquer sur *Ok*. Enregistrer l'image obtenue car le processus est long, il faut pouvoir vérifier rapidement les données au cas où.

Pour mesurer le niveau de gris moyen dans l'image obtenue, aller dans *Analyse>Set Measurments* et cocher *Min&Max Gray Value* et *Mean gray value*
Cliquer sur *Ok*

Retourner dans *Analyse* et cliquer sur *Measure.*
On obtient les valeurs voulues.

Recommencer **pour les images correspondant au blanc**.

On obtient **g** en soustrayant les deux valeurs obtenues avec les traitements précédents.

- ***Tracé du diagramme final***

Objectif : Une fois les résultats enregistrés, le but est de tracer un “diagramme de luminescence” : un diagramme où l’on peut connaître les paramètres optimaux afin d’obtenir de la luminescence dans la cuve. On prend pour cela deux axes : **la fréquence** des vagues et leur **amplitude** (ce sont les deux paramètres importants de notre dispositif).

Protocole : Utilisation du programme Python (commenté en annexe 2).

Annexes

Annexe 1 : Programme Arduino commenté

```
/*
  Blink

  Turns an LED on for one second, then off for one second, repeatedly.

  Most Arduinos have an on-board LED you can control. On the UNO, MEGA and ZERO
  it is attached to digital pin 13, on MKR1000 on pin 6. LED_BUILTIN is set to
  the correct LED pin independent of which board is used.
  If you want to know what pin the on-board LED is connected to on your Arduino
  model, check the Technical Specs of your board at:
  https://www.arduino.cc/en/Main/Products

  modified 8 May 2014
  by Scott Fitzgerald
  modified 2 Sep 2016
  by Arturo Guadalupi
  modified 8 Sep 2016
  by Colby Newman

  This example code is in the public domain.

  https://www.arduino.cc/en/Tutorial/BuiltInExamples/Blink
*/

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
 // pinMode(7, OUTPUT);
  pinMode(8, OUTPUT);
  pinMode(9, OUTPUT);
  digitalWrite(9, HIGH);   // turn the LED on (HIGH is the voltage level)
  digitalWrite(8, LOW);
  delay(3000);
}
```

Code qui initialise le piston quand on appuye sur reset ou à l’allumage du système. Rentre le piston, le met à x=0 pendant 3sec

```
// the loop function runs over and over again forever
void loop() {
  digitalWrite(8, HIGH);   // turn the LED on (HIGH is the voltage level)
  digitalWrite(9, LOW);    // turn the LED off by making the voltage LOW
  delay(100);                       // wait for a second
  digitalWrite(9, HIGH);   // turn the LED on (HIGH is the voltage level)
  digitalWrite(8, LOW);
  delay(100);                       // wait for a second
}
```

Fait avancer le piston pendant 100ms. Puis fait reculer le piston pendant 100ms.
La pin8 contrôle son mouvement (8=HIGH le piston avance, 8=LOW il recule)

Annexe 2 : Programme Python commenté

```
import numpy as np #Module pour les matrices
import matplotlib.pyplot as plt #Module pour les graphiques

#Graphique Binaire

#On trace d'abord le graphe binaire (qui a 3 couleurs ici) rouge/jaune/vert :
#On range les points dans des tableaux avec leurs coordonnées (fréquence, amplitude) puis l'on utilise
#la fonction scatter pour créer des points plus gros qu'habituellement
#On définit le titre et la légende et on trace le diagramme.
rouge=np.array([[1.667,1.667,1.25,2.5,1.25,3.33,5],[0.001875,0.00125,0.001905, 0.00366,0.002702,0.007143,0.0012]])
orange=np.array([[1.667,1.25,3.33,3.33,2.5, 2.5],[0.0049,0.00743,0.0069,0.0092, 0.00341, 0.0076]])
vert=np.array([[ 2.5,1.667,1.667,1.25,1.25,2.5,1.667,1.25],[0.0064,0.0076,0.0068,0.0082,0.0089, 0.0091,0.0092,0.0083]])
for i in range(len(rouge[1,:])):
  plt.scatter(rouge[1,i], rouge[0,i], c='r', s=100)
for i in range(len(orange[0,:])):
  plt.scatter(orange[1,i], orange[0,i], c='y', s=100)
for i in range(len(vert[0,:])):
  plt.scatter(vert[1,i], vert[0,i], c='g', s=100)

plt.title('Diagramme prévisionnel de Luminescence, Lunula')
plt.xlabel('Amplitude de la vague (m)')
plt.ylabel('Fréquence des vagues (Hz)')
plt.show()
```

2.a. Tracé du diagramme binaire

```
#Graphique quantitatif

#On reprend le même principe, avec une subtilité : on a une donnée non binaire à tracer : on veut un graphique avec des points en niveau
#de bleu pour le tracé
#Pour ça on crée un tableau points à 3 lignes 2 de coordonnées (fréquence, amplitude) et la 3e qui correpsond au "paramètre de luminescence"
#appelé niveau moyen de gris ajouté par les algues ici.
#On utilise la fonction scatter du module plt (comme précédemment) mais avec un jeu de données (data).

points=np.array([[1.667,1.25,1.25,3.33,5,1.25,3.33,3.33,1.667,1.667,1.667,1.667,1.25,1.25,1.667,1.25, 2.5, 2.5, 2.5,2.5,2.5],
                 [0.00125,0.001905,0.002702,0.007143,0.0012,0.00743,0.0069,0.0092,0.001875,0.0049,0.0076,0.0068,0.0082,0.0089,0.0092,0.0083,
                 [162.632,175.105,88.064,194.78,206.91,145.971,211.354,260.302,88.501,98.941,260.799,257.732,228.352,187.392,296.267,267.118,
frequences=points[0]
amplitudes=points[1]
valeurs=points[2]
data = {'x': amplitudes,
        'y': frequences,
        'taille': 100,
        'color': valeurs}

plt.scatter('x', 'y', c='color', s='taille', data=data, cmap='Blues')
#for i in range(len(bleu[0,:])):
  #plt.scatter(bleu[1,i], bleu[0,i], c='w', s=120)
plt.xlabel('Amplitude de la vague (m)')
plt.ylabel('Fréquence des vagues (Hz)')
cbar= plt.colorbar()
cbar.set_label("Niveau moyen de gris ajouté par les algues", labelpad=+1)
plt.title('Diagramme expérimental de Luminescence, Lunula')

plt.show()
```

2.b. Tracé du diagramme quantitatif