Reporting Quatre Quadrants n°5

Projet Minuto - FISE A1

Groupe B11

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

3 décembre 2024

Constitution de l'équipe

Groupe/Equipe : B11
Chef de projet : Emilien WOLFF

Actualisation de la fiche à la date du : 03/12

▷ Ce que nous avions prévu de faire aujourd'hui

- Dernière mesure de c_T à l'aide de la relation :

$$c_{\text{laiton}} = \frac{E}{m_{\text{laiton}} \times \Delta T} = \frac{(V \times I) \times \Delta t}{m_{\text{laiton}} \times (T_{\text{f}} - T_{\text{i}})}$$

  Or on remarque que :

$$\frac{\Delta T}{\Delta t} = \frac{P}{mc_t}$$

  Ce rapport est donc constant et nous allons donc réaliser plusieurs mesures de températures sur des intervalles $\Delta t = 30$ s puis réaliser une régression linéaire.

- Calcul des incertitudes associées

$$\frac{u(c_T)}{c_T} = \sqrt{\left(\frac{u(V)}{V}\right)^2 + \left(\frac{u(I)}{I}\right)^2 + \left(\frac{u(\Delta T)}{\Delta T}\right)^2 + \left(\frac{u(m)}{m}\right)^2 + \left(\frac{u(\Delta t)}{\Delta t}\right)^2}$$

- Suivi du projet dans *GanttProject*
- Continuer le compte-rendu final sur *Overleaf*
- Calcul des incertitudes de la mesure expérimentale de C_T
- Écriture d'un protocole précis pour les mesures d'irradiance
- Feuille `Python` pour faire des tirages de Monte Carlo

▷ Ce que nous avons réalisé effectivement

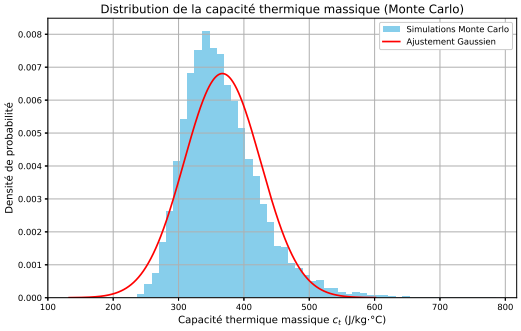
- Calcul de l'incertitude associée à la mesure de c_T :

$$\frac{u(c_T)}{c_T} = 0,02 = 2\%$$

  On trouve une incertitude très acceptable, à partir de celles des instruments de mesure :

$$\frac{u(V)}{V} = 0,01 \qquad \frac{u(I)}{I} = 0,01 \qquad u(\Delta t) = 0,5s \qquad u(m) = 1g \qquad u(\Delta T) = 0,05°C$$

- On faire alors une simulation de Monte Carlo




```python
// Generation des echantillons
    aleatoires
I_samples = np.random.normal(I_mean,
    I_std, n_simulations)
U_samples = np.random.normal(U_mean,
    U_std, n_simulations)
delta_T_samples = np.random.normal(
    delta_T_mean, delta_T_std,
    n_simulations)
m_samples = np.random.normal(m_mean,
    m_std, n_simulations)
```

Simulation de Monte Carlo sur Python

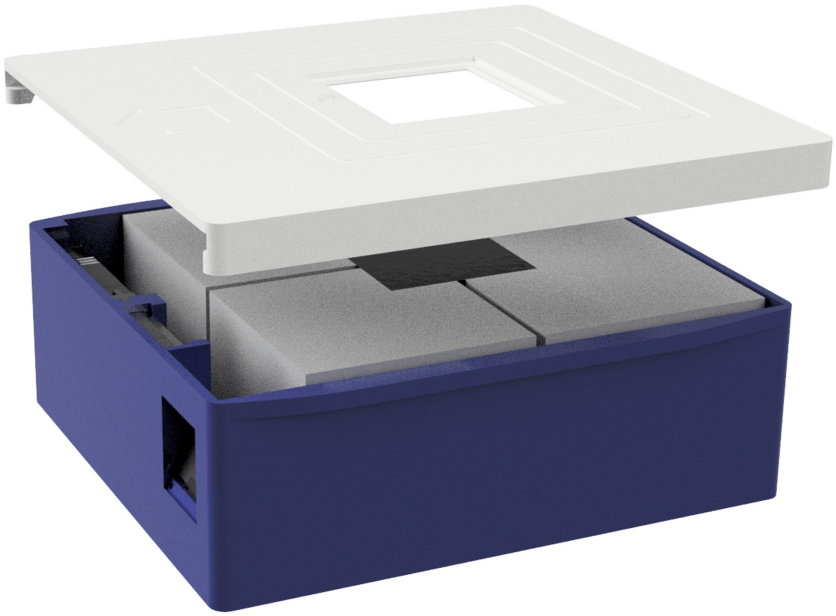


FIGURE 1 – Notre boitier, intégrant le polystyrène, l'électronique et le bloc de laiton

▷ Ce que nous prévoyons de faire les prochains jours

- Test de notre boîtier quand il y aura du soleil (bientôt ?)

▷ Problèmes rencontrés et solutions mises en œuvre

- Valeurs incohérentes affichées sur l'afficheur OLED, code à revoir ...

Annexes : codes de la session

Code pour le tirage de Monte Carlo

```
1
2  import numpy as np
3  import matplotlib.pyplot as plt
4
5  // Donn es  exp rimentales
6  I_mean = 0.510          # Intensit  moyenne (A)
7  I_std = 0.0102         # Incertitude sur l'intensit  (A)
8
9  U_mean = 4.955         # Tension moyenne (V)
10 U_std = 0.04955           # Incertitude sur la tension (V)
11
12 delta_T_mean = 22.59-22.25   # Variation de temp rature moyenne ( C )
13 delta_T_std = 0.05     # Incertitude sur la variation de temp rature ( C )
14
15 m_mean = 0.622          # Masse moyenne (kg)
16 m_std = 0.001        # Incertitude sur la masse (kg)
17
18 delta_t = 30          # Dur e de l'exp rience (s), suppos e sans incertitude
19
20 // Nombre de simulations Monte Carlo
21 n_simulations = 10000
22
23 // G n ration des  chantillons  al atoires
24 I_samples = np.random.normal(I_mean, I_std, n_simulations)
25 U_samples = np.random.normal(U_mean, U_std, n_simulations)
26 delta_T_samples = np.random.normal(delta_T_mean, delta_T_std, n_simulations)
27 m_samples = np.random.normal(m_mean, m_std, n_simulations)
28
29 // Calcul de la capacit  thermique massique pour chaque simulation
30 ct_samples = (U_samples * I_samples * delta_t) / (m_samples * delta_T_samples)
31
32 // Calcul des statistiques
33 ct_mean = np.mean(ct_samples)
34 ct_std = np.std(ct_samples)
35
36 // Trac  de la distribution
37 plt.figure(figsize=(10, 6))
38 plt.hist(ct_samples, bins=50, density=True, alpha=0.6, color='skyblue', label='
       Simulations Monte Carlo')
39
40 // Trac  de la gaussienne ajust e
41 x = np.linspace(ct_mean - 4*ct_std, ct_mean + 4*ct_std, 500)
42 gaussian = (1 / (ct_std * np.sqrt(2 * np.pi))) * np.exp(-0.5 * ((x - ct_mean) /
       ct_std)**2)
43 plt.plot(x, gaussian, color='red', linewidth=2, label='Ajustement Gaussien')
44
45 // Titres et l gendes
46 plt.title("Distribution de la capacit  thermique massique (Monte Carlo)",
       fontsize=14)
47 plt.xlabel("Capacit  thermique massique $c_t$ (J/kg   C)", fontsize=12)
48 plt.ylabel("Densit  de probabilit ", fontsize=12)
49 plt.legend()
50 plt.grid()
51 plt.show()
52
53 // Affichage des r sultats
54 print(f"Capacit  thermique massique moyenne: {ct_mean:.2f} J/(kg   C)")
55 print(f"Incertitude ( cart -type): {ct_std:.2f} J/(kg   C)")
```

Remarques : Le code fonctionne correctement et mesure de températures sont bonnes