TP0 – Mesures et incertitudes

Correction

Mesure de résistance

Ohmmètre analogique

1. N'ayant qu'une mesure, on ne peut procéder qu'à une estimation de type B de l'incertitude.
2. Sur l'ohmmètre, on estime $R_{\text{ana}} \in [84\,\Omega, 85\,\Omega]$:
   - la valeur retenue est donc $R_{\text{ana}} = 84{,}5\,\Omega$;
   - l'incertitude-type se calcule par

$$u(R_{\text{ana}}) = \frac{0{,}5}{\sqrt{3}} \approx 0{,}3\,\Omega.$$

   Le résultat de la mesure s'écrit

$$\boxed{R_{\text{ana}} = (84{,}5 \pm 0{,}3)\,\Omega,}$$

   où ce qui suit le symbole $\pm$ est l'incertitude-type.

Ohmmètre numérique

3. Avec cette série de mesure, on peut réaliser une estimation statistique de l'incertitu-type, c'est-à-dire une estimation de type A.
4. On complète le code de manière à calculer les quantités demandées.

```
R_num = np.mean(mesures)            # Moyenne
print(R_num)

s_R_num = np.std(mesures, ddof=1) # Écart-type expérimental
print(s_R_num)

n = len(mesures) # nombre de valeurs
u_R_num = s_R_num / np.sqrt(3)     # Incertitude-type sur la moyenne
print(u_R_num)
```

   Le programme renvoie :

$$R_{\text{num}} \approx 82{,}5\,\Omega, \quad s(R_{\text{num}}) \approx 0{,}4\,\Omega \quad \text{et} \quad u_A(R_{\text{num}}) \approx 0{,}2\,\Omega.$$

   Le résultat de la mesure s'écrit

$$\boxed{R_{\text{num}} = (82{,}5 \pm 0{,}2)\,\Omega,}$$

   où ce qui suit le symbole $\pm$ est l'incertitude-type.

5. On procède à une estimation de type B : on calcule l'incertitude $\Delta(R_{\text{num}})$ d'après les données indiquées par le fabriquant :

$$\Delta(R_{\text{num}}) = \frac{0{,}9}{100} \times 82{,}5 + 1 \times 0{,}1 \approx 0{,}8\,\Omega,$$

d'où

$$u_B(R_{\text{num}}) = \frac{\Delta(R_{\text{num}})}{\sqrt{3}} = \frac{0{,}8}{\sqrt{3}} \approx 0{,}5\,\Omega.$$

L'incertitude-type totale vaut donc

$$u(R_{\text{num}}) = \sqrt{u_A(R_{\text{num}})^2 + u_B(R_{\text{num}})^2} \approx 0{,}5\,\Omega.$$

Le résultat de la mesure s'écrit donc finalement

$$\boxed{R_{\text{num}} = (82{,}5 \pm 0{,}5)\,\Omega,}$$

où ce qui suit le symbole $\pm$ est l'incertitude-type.

Loi d'Ohm

6. De la même façon que pour la question 5, en utilisant la ligne *Current DC* (courant continu) de la notice, on obtient

$$\Delta(I) = \frac{1}{100} \times 146{,}5 + 3 \times 0{,}01 \approx 2\,\text{mA}, \quad \text{d'où} \quad \boxed{u(I) \approx 0{,}9\,\text{mA}.}$$

7. D'après la loi d'Ohm, on a

$$R_{\text{ohm}} = \frac{U}{I} = 82{,}6\,\Omega.$$

L'incertitude-type totale se calcule par propagation des incertitudes sur un quotient :

$$u(R_{\text{ohm}}) = R_{\text{ohm}} \sqrt{\left(\frac{u(U)}{U}\right)^2 + \left(\frac{u(I)}{I}\right)^2} \approx 0{,}9\,\Omega.$$

$$\boxed{R_{\text{ohm}} = (82{,}6 \pm 0{,}8)\,\Omega,}$$

où ce qui suit le symbole $\pm$ est l'incertitude-type.

8. Attention : le programme `monte-carlo.py` utilise l'incertitude Δ et non l'**incertitude-type** (u) sur U et I, car la simulation utilise une distribution uniforme de probabilité. Il faut donc recalculer l'incertitude en multipliant l'incertitude-type par $\sqrt{3}$.
On peut aussi utiliser directement l'incertitude-type mais il faut alors utiliser une loi normale pour simuler les mesures.
Les deux solutions sont proposées ci-dessous, elles donnent bien le même résultat que précédemment.

```python
import numpy as np

def r(u,i):     # loi d'Ohm
    return u/i

# MESURES
u   = 12.1      # tension en volts
u_u =  0.1      # incertitude-type sur u en volts
i   = 146.5e-3 # intensité en ampère
u_i =   0.8e-3 # incertitude-type sur i en ampère

# MONTE-CARLO : simulation de 10000 mesures
# avec np.random.uniform()
d_u = u_u * np.sqrt(3) # incertitude sur u en volts
d_i = u_i * np.sqrt(3) # incertitude sur i en ampère
u_sim = np.random.uniform(u-d_u, u+d_u, 10000)
i_sim = np.random.uniform(i-d_i, i+d_i, 10000)
r_sim = r(u_sim, i_sim)
print("Valeur           :", r(u,i))
print("Incertitude-type :", np.std(r_sim, ddof=1))

# avec np.random.normal()
u_sim = np.random.normal(u, u_u, 10000)
i_sim = np.random.normal(i, u_i, 10000)
r_sim = r(u_sim, i_sim)
print("Valeur           :", r(u,i))
print("Incertitude-type :", np.std(r_sim, ddof=1))
```

Comparaison des résultats

9. On calcule, pour les trois valeurs déterminée précédemment, l'écart normalisé avec la méthode quatre fils.

Méthode	analogique	numérique	loi d'Ohm
E_n	6,6	0,058	0,089
Compatible avec la méthode 4 fils	non	oui	oui

Seule la méthode avec l'ohmmètre analogique présente un écart normalisé avec la méthode quatre fils supérieur à 2 : ces deux mesures sont incompatibles.

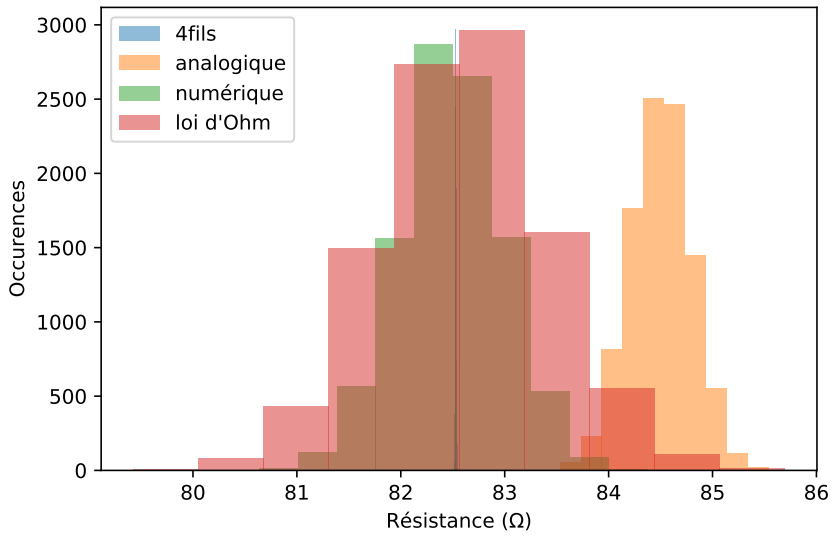
10. Cf. ci-dessous.

11. Le code ci-dessous permet de simuler et représenter un grand nombre de mesures réalisées avec les différents protocoles.

```python
import numpy as np
import matplotlib.pyplot as plt

########################
# SIMULATION DES MESURES
```

```
##########################
R_4fils_sim = np.random.normal(82.529,0.005,10000) # quatre fils
R_ana_sim   = np.random.normal(84.5,0.3,10000)     # analogique
R_num_sim   = np.random.normal(82.5,0.5,10000)     # numérique
R_ohm_sim   = np.random.normal(82.6,0.8,10000)     # loi d'Ohm

############################
# REPRÉSENTATIONS GRAPHIQUES
############################
plt.hist(R_4fils_sim, alpha=.5, label="4fils")
plt.hist(R_ana_sim, alpha=.5, label="analogique")
plt.hist(R_num_sim, alpha=.5, label="numérique")
plt.hist(R_ohm_sim, alpha=.5, label="loi d'Ohm")
plt.xlabel("Résistance ($\Omega$)")
plt.ylabel("Occurences")
plt.legend() # affichage de la légende

plt.show()
```

On obtient un graphique semblable à celui représenté ci-dessous.



On remarque que lorsque les résultats sont compatibles, les mesures simulées donnent lieu à des histogrammes qui se recouvrent, au moins partiellement.

12. Pour simuler une erreur systématique importante, on peut changer la moyenne de la distribution, par exemple :

```
R = np.random.normal(120,0.5,10000)
```

Pour simuler un appareil peu fidèle, il faut augmenter la dispersion des valeurs, par exemple :

```
R = np.random.normal(82.5,25,10000)
```

Choix du calibre

13. La méthode est semblable à celle employée à la question 5.

```
r_num = 85.534
d_cons_100 = 0.014 * r_num / 100 + 0.007 * 100 / 100
u_100          = d_cons_100 / np.sqrt(3)
d_cons_10k = 0.014 * r_num / 100 + 0.001 * 1e4 / 100
u_10k          = d_cons_10k / np.sqrt(3)
d_cons_1M  = 0.014 * r_num / 100 + 0.001 * 1e6 / 100
u_1M           = d_cons_1M / np.sqrt(3)
print("incertitude-type (calibre 100 ohm) =", u_100, "ohm")
print("incertitude-type (calibre 10 kohm) =", u_10k, "ohm")
print("incertitude-type (calibre 1 Mohm)  =", u_1M, "ohm")
```

La précision s'obtient simplement en divisant l'incertitude-type par la valeur mesurée.

Calibre	**Incertitude-type (Ω)**	**Précision (%)**
100 Ω	0,01	$\sim 0{,}01$
10 kΩ	0,06	$\sim 0{,}07$
1 MΩ	6	~ 7

Il apparait que la précision est meilleure quand le calibre est faible : il faut choisir le plus petit calibre, supérieur à la valeur mesurée pour obtenir la meilleure précision lors de la mesure.