# Incertitudes

## Plan du cours



Même en s'appliquant du mieux possible, répéter la même expérience dans les mêmes conditions conduira presque toujours à des résultats différents : on parle de **variabilité** des résultats de l'expérience. La première cause de variabilité est bien souvent l'expérimentateur lui-même, qui ne peut reproduire les mêmes gestes rigoureusement à l'identique. L'environnement et les instruments de mesure sont d'autres causes importantes.

L'**incertitude** est un nombre qui quantifie la variabilité des résultats obtenus lors de différentes répétitions d'une même expérience dans les mêmes conditions.

En physique, on utilise conventionnellement l'**incertitude-type**, c'est-à-dire reliée à un écart-type.

**Attention !** Ne pas confondre incertitude et erreur dans la réalisation du protocole : ce n'est pas parce que les résultats varient d'une réalisation à l'autre que certaines mesures sont correctes et d'autres fausses. Ainsi, sauf si vous avez de bonnes raisons de croire que vous vous êtes trompé dans la réalisation de l'expérience, il n'y a aucune raison d'éliminer une mesure « parce que sa valeur est différente, donc fausse ».

Le résultat final d'une expérience de mesure d'une grandeur $x$ doit nécessairement présenter la valeur expérimentale $X_{\text{exp}}$ **et** l'incertitude-type $u(X_{\text{exp}})$ associée.

**Convention :** l'incertitude-type s'exprime avec deux chiffres significatifs ; la position du deuxième chiffre donne celle du dernier chiffre significatif à garder sur la valeur mesurée.

$$\begin{aligned} X_{\text{exp}} &= 1\ 7,3\ 0\ 9\ 6 \text{ cm} \\ u(X_{\text{exp}}) &= \quad 0,2\ 8\ 7\ 1 \text{ cm} \end{aligned} \quad \text{devient} \quad \begin{cases} x = 17{,}31 \text{ cm} \\ u(x) = 0{,}29 \text{ cm} \end{cases}$$

# I - Introduction : modélisation probabiliste d'un processus de mesure

Formellement, la mesure d'une grandeur physique se modélise par le tirage d'une variable aléatoire $x$, dont on obtient un résultat différent à chaque réalisation de l'expérience (= chaque tirage). « Mesurer la grandeur » revient à estimer du mieux possible l'espérance $\mathbb{E}(x)$ et l'écart-type $\sigma_x$ de la variable aléatoire, appelées dans ce contexte **valeur expérimentale** et **incertitude-type**.

La loi de probabilité de la variable aléatoire $x$ est (et restera !) inconnue : on ne peut que la modéliser. Les deux modèles que nous utiliserons sont la **loi normale** et la **loi uniforme**, représentées figure 1.

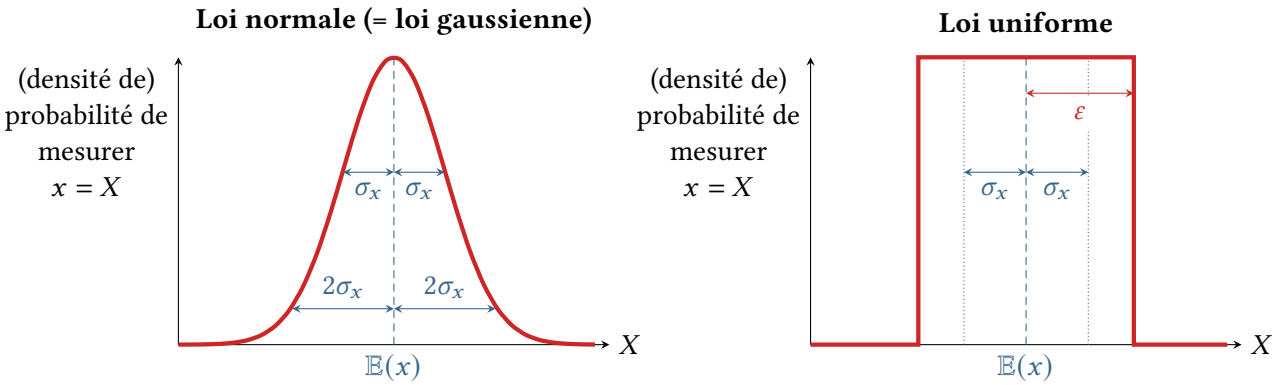


**Figure 1 – Lois de probabilité usuelles.**

- Pour une loi normale, on montre que la probabilité qu'une mesure se trouve dans l'intervalle $[\mathbb{E}(x)-\sigma_x, \mathbb{E}(x)+\sigma_x]$ est de 68 % : on parle d'**intervalle de confiance à 68 %**. De la même façon, $[\mathbb{E}(x)-2\sigma_x, \mathbb{E}(x)+2\sigma_x]$ est l'intervalle de confiance à 95 %.
- Dans le cas d'une loi uniforme, l'écart-type $\sigma_x$ est relié à la **demi-étendue** $\varepsilon$ de la distribution par

$$\sigma_x = \frac{\varepsilon}{\sqrt{3}}.$$

Pour cette loi, $[\mathbb{E}(x)-\sigma_x, \mathbb{E}(x)+\sigma_x]$ est l'intervalle de confiance à 57 % ... et bien sûr $[\mathbb{E}(x)-\varepsilon, \mathbb{E}(x)+\varepsilon]$ l'intervalle de confiance à 100 % !

# II - Estimer l'incertitude sur une grandeur mesurée

## II.A - La grandeur fluctue : estimation de type A

La situation la plus simple à analyser est celle où la variabilité est directement visible sur les résultats expérimentaux. On parle alors d'une **estimation d'incertitude de type A**.

Quand on dispose d'une série de mesures $X_1, ..., X_N$, la valeur expérimentale est la moyenne

$$X_{\text{exp}} = \langle X_n \rangle = \frac{1}{N}\sum_{n=1}^{N} X_n$$

et l'incertitude-type sur cette valeur est reliée à l'écart-type $\sigma$ de la série de mesure par

$$u(X_{\text{exp}}) = \frac{\sigma}{\sqrt{N}}$$

Il existe plusieurs conventions de définition pour l'écart-type. Celle utilisée pour l'incertitude-type est

$$\sigma = \sqrt{\frac{1}{N-1}\sum_{n=1}^{N}\Big(X_n - \langle X_n\rangle\Big)^2} \qquad \text{où } \langle X_n\rangle \text{ désigne la moyenne.}$$

➽ **Pour approfondir :** Ces résultats s'établissent de manière mathématiquement rigoureuse dans le cas où la distribution de probabilité de la grandeur mesurée suit une loi normale. La distribution de probabilité exacte étant de toute façon inconnue et inaccessible, on les utilise dans tous les cas : on parle d'*approximation normale*, dont on peut montrer qu'elle est de mieux en mieux vérifiée que le nombre de mesures réalisées est élevé. Ce résultat très utilisé en statistiques porte le nom de *théorème central limite*.

Notons que la convention de définition de l'écart-type n'est pas la même qu'en mathématiques, où il est généralement défini avec une division par $N$ au lieu de $N-1$. On peut qualitativement constater qu'en physique, l'interprétation comme une incertitude est plus cohérente avec une division par $N-1$. En effet, si on ne réalise qu'une seule mesure ($N=1$), diviser par $N$ conduirait à une incertitude-type $u(x)=0$ ... alors que le résultat n'est absolument pas certain ! Au contraire, diviser par $N-1$ donne une incertitude-type non définie (0 divisé par 0), ce qui est davantage conforme au sens physique. L'argument optionnel `ddof=1` de la fonction `np.std` indique cette division par $N-1$. ■

**Code clés en main – exemple 1**

```python
import numpy as np

x = np.array([ ... ]) # à compléter

x_exp = np.mean(x)
u_x_exp = np.std(x,ddof=1)/np.sqrt(len(x))

print('x =', x_exp)
print('u(x) =', u_x_exp)
```

## II.B - La grandeur ne fluctue pas : estimation de type B

De nombreuses situations ne permettent pas d'observer directement de variabilité :

- la grandeur elle-même ne varie pas dans les conditions de l'expérience (p.ex. distance entre deux repères) ;
- l'expérience ne peut être réalisée qu'une seule fois, car trop longue ou destructive pour l'échantillon ;
- la variabilité est inférieure à la résolution de l'instrument de mesure utilisé ;
- la valeur est donnée par un constructeur (p.ex. composant électrique).

Dans tous les cas, on ne dispose que d'un seul et unique résultat ... mais cela ne veut absolument pas dire qu'il est certain ! On parle dans ce contexte d'**estimation d'incertitude de type B**.

L'idée consiste à estimer un intervalle dans lequel il est (raisonnablement) certain que le résultat de la mesure se trouve, noté sous la forme $[x_0-\varepsilon, x_0+\varepsilon]$, dont $x_0$ est appelé le centre et $\varepsilon$ la **demi-étendue**, et à assimiler la distribution de probabilité de la grandeur mesurée à une loi uniforme.

Pour un résultat ne fluctuant pas, compris avec « certitude » dans l'intervalle $[x_0-\varepsilon, x_0+\varepsilon]$,

$$X_{\text{exp}} = x_0 \qquad \text{et} \qquad u(X_{\text{exp}}) = \frac{\varepsilon}{\sqrt{3}}$$

Selon les situations, la demi-étendue est estimée ou bien qualitativement « avec bon sens », ou bien en se reportant à la notice de l'instrument ou du composant utilisé.

**Un exemple pour comprendre :**


Imaginons avoir l'idée farfelue de mesurer la longueur $L$ d'une carte bancaire avec la règle jaune du tableau. On constate que la longueur de la carte est comprise avec certitude dans l'intervalle $[8\,\text{cm}; 9\,\text{cm}]$. La demi-étendue vaut 0,5 cm, d'où une incertitude-type $0{,}5/\sqrt{3} = 0{,}2887$ cm. En étant vigilant aux chiffres significatifs, on conclut $L = 8{,}50$ cm et $u(L) = 0{,}29$ cm.

> ***Remarque :*** *J'ai ici considéré le cas le plus simple où la demi-étendue $\varepsilon$ correspond à la moitié d'une graduation, mais ce n'est pas une règle générale : tout dépend de la situation et de l'appréciation de l'expérimentateur. Par exemple, mesurer avec la même règle jaune la largeur d'une feuille A4 donnerait 21 cm « presque tout pile » : compte tenu de l'observation visuelle de la règle, on pourrait alors par exemple prendre comme intervalle [20,8 cm ; 21,2 cm].*

## III - Estimer l'incertitude sur une grandeur calculée

Sauf exception, l'incertitude sur une grandeur calculée ne se détermine pas de façon immédiate en raison de la « compensation des erreurs » : une fluctuation sur une valeur intervenant dans le calcul peut être compensée par une autre, au moins partiellement. L'incertitude est appelée **incertitude composée**, et on parle de **composition des incertitudes** sur les différentes grandeurs.

**Attention !** Avant de se lancer dans un calcul d'incertitude-type composée, il faut toujours commencer par comparer entre elles les incertitudes relatives $u(y)/y$, $u(z)/z$, etc, des différentes grandeurs d'entrée et ne considérer que celles qui sont prépondérantes. Très souvent, une de ces grandeurs possède une incertitude relative très supérieure aux autres, ce qui évite un calcul fastidieux.

### III.A - Formules de composition des incertitudes

Les formules de composition se démontrent par des raisonnements probabilistes, valables lorsque les différentes variables impliquées dans le calcul sont indépendantes les unes des autres. En toute rigueur, elles ne sont valables que dans la limite d'un nombre infini de mesures suivant une loi de probabilité gaussienne. Il existe également des formules plus générales, valables pour n'importe quelle fonction $x = f(y, z, ...)$, mais elles conduisent à des calculs souvent fastidieux et peu parlants, si bien qu'elles ne figurent pas aux programmes de CPGE.

| | | |
|---|---|---|
| Multiplication par une constante | $x = \lambda y \quad (\lambda \in \mathbb{R})$ | $u(x) = \lvert\lambda\rvert\, u(y)$ |
| Somme ou différence | $x = y + z \quad$ ou $\quad x = y - z$ | $u(x) = \sqrt{u(y)^2 + u(z)^2}$ |
| Produit ou quotient | $x = y \times z \quad$ ou $\quad x = \dfrac{y}{z}$ | $\dfrac{u(x)}{x} = \sqrt{\left(\dfrac{u(y)}{y}\right)^2 + \left(\dfrac{u(z)}{z}\right)^2}$ |
| Puissances et racines | $x = y^p \quad (p \in \mathbb{R})$ | $\dfrac{u(x)}{x} = \lvert p\rvert\, \dfrac{u(y)}{y}$ |

**Pour approfondir :** Contrairement à ce que l'on pourrait croire, la dernière ligne $y^p$ ne peut pas se déduire de l'avant-dernière $y \times z$. Pour que ces relations s'appliquent, les deux variables $y$ et $z$ doivent être indépendantes l'une de l'autre ... ce qui n'est évidement pas le cas de $y$ avec lui-même. Ce faisant, il y a une compensation partielle des incertitudes sur $y$ et $z$ : de manière très schématique, si une fluctuation rend $y$ un peu plus grand, il se peut que $z$ soit rendu un peu plus petit, ce qui donne une compensation dans le calcul de $x$. Un tel effet est bien sûr impossible dans le calcul de $y^p$. ■

### III.B - Estimation numérique par simulation Monte-Carlo

On cherche à estimer une grandeur $y$ reliée à un ensemble de $K$ grandeurs expérimentales $x_1, x_2, ..., x_K$ par une relation écrite formellement $y = f(x_1, x_2, ..., x_K)$, avec $f$ une fonction quelconque mais connue. Chaque grandeur expérimentale $x_k$ $(1 \leq k \leq K)$ est caractérisée par sa valeur et son incertitude-type $u(x_k)$. La méthode de Monte-Carlo consiste à estimer l'incertitude-type $u(y)$ par simulation d'un grand nombre $N$ d'expériences.

> ***Remarque culturelle :*** *Ces méthodes par tirage aléatoire étaient (sont ?) utilisées par les joueurs de casino pour estimer leurs gains potentiels, d'où leur nom générique de « méthodes de Monte-Carlo ».*

**Principe :** Pour chaque réalisation $n$ de l'expérience, on tire aléatoirement une valeur $x_{k,n}$ pour chaque $x_k$ et on en déduit la valeur correspondante $y_n$. L'incertitude-type composée $u(y)$ correspond à l'écart-type de la série des $y_n$ simulés, sans division par $\sqrt{N}$.

> ***Remarque :*** *Pourquoi prendre $u(y) = \sigma$ et non pas $u(y) = \sigma/\sqrt{N}$ ? Constatons d'emblée que dans le second cas plus on simulerait d'expériences, plus l'incertitude-type diminuerait ... alors que l'on dispose toujours des mêmes informations sur le système expérimental. Réduire l'incertitude sans affiner la connaissance du système serait contradictoire. L'explication vient du fait que la simulation permet d'estimer l'incertitude sur* une seule *mesure de $y$, celle qui a été réalisée « pour de vrai » dans l'expérience, et non pas sur une moyenne des résultats de* plusieurs *mesures.*

**Choix de la distribution de probabilité :** on préférera utiliser pour les tirages aléatoires une loi de probabilité gaussienne si la valeur centrale est plus probable que les autres, et une loi de probabilité uniforme si toutes les valeurs de l'intervalle sont équiprobables. Cependant, le choix de la distribution impacte peu les résultats en pratique, et on rencontre de nombreux cas où les deux modèles ont de solides arguments à défendre : on peut alors utiliser l'une ou l'autre sans grande conséquence.

**Implémentation :** `numpy` permet de simuler les $N$ expériences sans faire appel à des boucles. Les fonctions ci-dessous renvoient directement un tableau de $N$ valeurs :

- `np.random.uniform(mini,maxi,N)` renvoie un tableau de `N` valeurs aléatoires suivant une loi uniforme dans un intervalle [`mini`, `maxi`] ;
- `np.random.normal(moy,sigma,N)` renvoie un tableau de `N` valeurs aléatoires suivant une loi normale de moyenne `moy` et d'écart-type `sigma`.

La valeur expérimentale est directement la valeur mesurée « en vrai », estimer la valeur moyenne des simulations n'a pas d'intérêt ... et conduit de toute façon au même résultat.

**Code clés en main – exemple 2**

```python
import numpy as np

x1 = ...
u_x1 = ... # on a l'incertitude (c'est un exemple !)

x2 = ...
eps_x2 = ... # on a la demi-étendue (c'est un exemple !)

y_exp = f(x1,x2) # remplacer f par son expression !

N = 10000 # nombre de simulations
x1_sim = np.random.normal(x1, u_x1, N)
x2_sim = np.random.uniform(x2-eps_x2, x2+eps_x2, N)
y_sim = f(x1_sim, x2_sim) # calcul terme à terme car np.array
   # remplacer f par son expression !

u_y = np.std(y_sim,ddof=1)
# il est inutile de calculer np.mean(y_sim) !

print('y =', y_exp)
print('u(y) =', u_y)
```

# IV - Estimer les paramètres d'un modèle : régression

Une méthode de régression consiste à déterminer les meilleures valeurs des paramètres d'un modèle physique dont la forme est connue afin que la fonction obtenue reproduise au mieux les points expérimentaux. En pratique, pour deux grandeurs $x$ et $y$ supposées reliées par une relation de la forme $y = f(x)$, on réalise $N$ expériences donnant accès à des valeurs $X_n$ et $Y_n$ ($1 \leq n \leq N$) et on en déduit les valeurs des paramètres intervenant dans la fonction $f$ qui permettent d'approcher au mieux les points expérimentaux.

## IV.A - Régression linéaire $y = ax$

N'avoir qu'un seul coefficient à estimer permet une exploitation relativement simple : l'idée consiste à traiter chaque réalisation de l'expérience comme une mesure indépendante de la grandeur $a = y/x$.

### IV.A.1 - Estimation de la valeur expérimentale $a_{\text{exp}}$

Il suffit de calculer la valeur $A_n = Y_n/X_n$ pour chaque réalisation de l'expérience et d'en prendre la valeur moyenne $a_{\text{exp}} = \langle A_n \rangle$.

**Code clés en main – exemple 3**

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

x = np.array([ ... ])
y = np.array([ ... ])

### Estimation de a_exp
a = y/x # calcul terme à terme car np.array
a_exp = np.mean(a)

### Tracé final
plt.figure()
plt.plot(x, y, 'o') # points expérimentaux
plt.plot(x, a_exp*x, '-') # droite de régression
   # attention, a_exp et non pas a
plt.show()

print('a =', a_exp)
```

### IV.A.2 - Estimation de l'incertitude $u(a_{\text{exp}})$

- **Méthode rapide sans incertitude sur $x$ et $y$**

Traiter chaque réalisation de l'expérience comme une mesure indépendante de la grandeur $a = y/x$ permet de dérouler la procédure « classique » d'une estimation de type A : $u(a_{\text{exp}}) = \sigma/\sqrt{N}$, comme décrit au paragraphe II.A. Cela donne une estimation simple et rapide de l'incertitude $u(a_{\text{exp}})$, qui ne fait pas appel aux incertitudes $u(x)$, $u(y)$ sur les valeurs mesurées.

**Code clés en main – exemple 4**

Ce code s'insère à la suite de l'exemple 3.

```python
u_a_exp = np.std(a,ddof=1)/np.sqrt(len(a))
print('u(a) = ', u_a_exp)
```

- **Méthode Monte-Carlo prenant en compte les incertitudes $u(x)$ et $u(y)$**

Si on dispose des incertitudes sur les valeurs de $x$ et $y$, il est logique (et préférable !) d'en tenir compte pour estimer $u(a_{\text{exp}})$. Il faut alors procéder par une méthode Monte-Carlo, se basant comme au paragraphe III.B sur la simulation numérique d'un grand nombre d'expériences :

- Simuler $K$ séries de mesures différentes : pour tout $k$ compris entre 1 et $K$,
  → tirer aléatoirement les valeurs $X_{n,k}$ et $Y_{n,k}$ pour $1 \leq n \leq N$ tenant compte des incertitudes ;
  → estimer le coefficient directeur $a_k = \left\langle Y_{n,k}/X_{n,k} \right\rangle$, différent à chaque simulation.
- L'incertitude-type $u(a_{\text{exp}})$ est égale à l'écart-type des $a_k$ simulés, sans division par $\sqrt{K}$ (revoir l'explication au paragraphe III.B).

Cette fois, je trouve qu'un code traitant directement les $K \times N$ valeurs expérimentales simulées est peu lisible, car cela demande de travailler avec des matrices : utiliser une boucle est nettement plus clair, et donc préférable.

**Code clés en main – exemple 5**

Ce code s'insère à la suite de l'exemple 3.

```
u_x = ... # loi normale et incertitude-type pour x (c'est un exemple !)
eps_y = ... # loi uniforme et demi-étendue pour y (c'est un exemple !)

K = 10000 # nombre de simulation

a_sim = []
for k in range(K):
    X_sim = np.random.normal(x,u_x) # tableau de même longueur que x
    Y_sim = np.random.uniform(y-eps_y, y+eps_y) # tableau de même longueur
        que y
    a_sim.append(np.mean(Y_sim/X_sim))

u_a_exp = np.std(a_sim)
```

## IV.B - Régression affine $y = ax + b$

### IV.B.1 - Estimation des coefficients $a_{\text{exp}}$ et $b_{\text{exp}}$

Coder « de zéro » la régression affine n'est pas envisageable dans un contexte de TP de physique. On utilise la fonction `polyfit` de la bibliothèque `numpy` :

```
a_exp, b_exp = np.polyfit(x,y,1)
```

Le troisième argument « `1` » indique que l'on modélise la courbe par un polynôme d'ordre 1.

**Attention !** La fonction `polyfit` ne permet pas de réaliser une régression linéaire de la forme $y = ax$, car il est impossible de lui imposer $b = 0$. Il faut donc malheureusement retenir deux méthodes différentes selon que la fonction est linéaire ou affine.

**Pour approfondir :** Un algorithme de régression affine détermine les valeurs de $a$ et $b$ qui minimisent « une » distance entre la droite théorique $y = ax + b$ et les points expérimentaux. Toute la difficulté consiste à définir correctement la fonction distance à utiliser. La définition utilisée par `polyfit` est celle des moindres carrés, où la distance est simplement définie par

$$d = \sum_{n=1}^{N} \Big( Y_n - (aX_n + b) \Big)^2 .$$

D'autres définitions sont possibles, notamment la distance des moindres carrés pondérés, qui permet de prendre en compte les incertitudes sur les points expérimentaux, l'idée étant qu'un point de faible incertitude doit avoir plus de poids qu'un point à l'incertitude élevée. ■

### IV.B.2 - Estimation des incertitudes $u(a_{\text{exp}})$ et $u(b_{\text{exp}})$

Si l'on ne dispose pas des incertitudes $u(x), u(y)$ sur les points expérimentaux, il n'existe pas de méthode aussi simple qu'un calcul d'écart-type pour estimer les incertitudes sur $a_{\text{exp}}$ et $b_{\text{exp}}$.

➽ **Pour approfondir :** Bien que ne figurant pas à notre programme, des formules analytiques existent. Elles reposent sur l'écart entre les points expérimentaux et la droite, et certaines hypothèses simplificatrices sur l'incertitude à l'origine de ces écarts. ■

La seule méthode au programme concerne le cas où l'on dispose de l'incertitude $u(y) \neq 0$ sur une des deux grandeurs mesurées alors que l'incertitude $u(x) = 0$ sur l'autre grandeur est supposée nulle[1]. Elle s'appuie sur une simulation Monte-Carlo de $K$ séries de mesures et les régressions associées.

- Simuler $K$ séries de mesures différentes : pour tout $k$ compris entre 1 et $K$,
  → tirer aléatoirement les valeurs $X_{n,k}$ et $Y_{n,k}$ pour $1 \leq n \leq N$ tenant compte des incertitudes ;
  → réaliser la régression affine donnant les coefficients $a_k$ et $b_k$, différents à chaque simulation.
- Les incertitudes-types $u(a_{\text{exp}})$ et $u(b_{\text{exp}})$ sont égales aux écarts-types des $a_k$ et $b_k$ simulés, sans division par $\sqrt{K}$ (revoir l'explication au paragraphe III.B).

**Code clés en main – exemple 6**

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

x = np.array([ ... ])
y = np.array([ ... ])
u_y = ... # valeur unique ou tableau

a_exp, b_exp = np.polyfit(x, y, 1)

### Estimation des incertitudes
K = 10000 # nombre de simulations
a_sim = []
b_sim = []

for k in range(K):
    y_sim = np.random.normal(y, uy) # tableau de même longueur que y
    # pas de tirage pour x, supposé connu exactement
    a,b = np.polyfit(x, y_sim, 1)
    a_sim.append(a)
    b_sim.append(b)

u_a_exp = np.std(a_sim)
u_b_exp = np.std(b_sim)

### Tracé
plt.figure()
plt.plot(x,y,'+')
plt.plot(x,a_exp*x+b_exp,'-') # droite de régression
    # attention, a_exp,b_exp et non pas a,b
plt.show()

print('a =', a_exp, 'u(a) =', u_a_exp)
print('b =', b_exp, 'u(b) =', u_b_exp)
# Inutile de calculer np.mean(a_sim) et np.mean(b_sim)
```

1. Cette hypothèse qui peut sembler grossière reste souvent pertinente, car il est fréquent que l'une des deux grandeurs soit connue avec une incertitude relative très supérieure à l'autre. C'est alors celle-ci qu'il faut placer en ordonnée.

## IV.C - Complément : régression non-linéaire

Dans la très grande majorité des situations que nous rencontrerons, il sera toujours possible de se ramener à une modélisation des points expérimentaux par une fonction linéaire ou affine, éventuellement en identifiant de nouvelles variables $x$ et $y$ permettant de se ramener à une forme affine ou linéaire. Cependant, il existe des situations qui résistent à cette démarche, bien qu'elles soient rares à notre niveau. La fonction `curve_fit` du module `scipy.optimize` permet de réaliser une telle régression. Ses spécifications (simplifiées) sont les suivantes :

*Syntaxe principale :* `popt, pcov = curve_fit(f,x,y,sigma=uy)`

*Entrées :*

- `f` est la fonction supposée modéliser la courbe expérimentale ;
- `x` et `y` sont deux tableaux de même longueur donnant les abscisses et les ordonnées ;
- l'argument optionnel `sigma=uy` est un tableau contenant les incertitudes sur l'ordonnée `y`, l'abscisse étant supposée connue exactement.

*Sorties :*

- `popt` est un tableau contenant les valeurs optimales des paramètres intervenant dans la fonction `f`, dans l'ordre dans lequel ils apparaissent dans la définition de `f` ;
- `pcov` est une matrice appelée matrice de covariance, dont les termes diagonaux `pcov[i,i]` sont égaux au carré de l'incertitude-type sur le paramètre `popt[i]`.

### Code clés en main – exemple 7

```python
import numpy as np
import matplotlib.pyplot as plt
from scipy.optimize import curve_fit

### Mesures
t = np.array([ ... ])
z = np.array([ ... ])
u_z = ... # valeur unique ou tableau

### Régression non-linéaire
# Fonction modèle : p.ex. oscillation amortie
def f(t, A, tau, omega):
    return A*np.exp(-t/tau)*np.cos(omega*t)

# Détermination des paramètres
popt, pcov = curve_fit(f,t,z,sigma=u_z)

# Paramètres optimaux
A, tau, omega = popt
u_A, u_tau, u_omega = np.sqrt(np.diag(pcov)) # racine des termes diagonaux

print('A =', A, 'u_A =', u_A)
print('tau =', tau, 'u_tau =', u_tau)
print('omega =', omega, 'u_omega =', u_omega)

# Ne pas oublier de vérifier par un tracé que la courbe modèle colle bien
   aux points expérimentaux !
```

# V - Tester la compatibilité entre deux valeurs

L'interprétation d'une expérience se termine bien souvent par une comparaison, par exemple entre une valeur expérimentale et une valeur attendue donnée par un fabricant ou prévue par un modèle théorique, ou bien entre deux valeurs issues de deux expériences différentes (p.ex. deux protocoles ou deux binômes).

## V.A - Comparer une valeur expérimentale et une valeur connue sans incertitude

Supposons la valeur attendue $x^\star$ donnée sans incertitude, ou d'incertitude négligeable (p.ex. grandeur tabulée). L'idée est qualitativement simple : la valeur mesurée est compatible avec la valeur attendue si elle n'en est « pas trop » éloignée. Comme la dispersion des valeurs expérimentales est quantifiée par l'incertitude-type, l'écart entre les valeurs mesurée et attendue doit être comparé à cette incertitude-type.

Par convention, la valeur expérimentale $x_{\text{exp}}$ et la valeur attendue $x^\star$ sont dites compatibles si

$$|x_{\text{exp}} - x^\star| \leq 2\,u(x_{\text{exp}}) \quad \Longleftrightarrow \quad x^\star \in [x_{\text{exp}} - 2\,u(x_{\text{exp}});\ x_{\text{exp}} + 2\,u(x_{\text{exp}})]$$

Le facteur 2 est choisi par convention, essentiellement pour des raisons historiques et parce qu'il tombe rond : en pratique, les mesures ne sont pas « beaucoup plus compatibles » si $|x_{\text{exp}} - x^\star| = 1{,}9\,u(x_{\text{exp}})$ ou $2{,}1\,u(x_{\text{exp}})$.

> ***Remarque :*** *Ce résultat peut se reformuler avec le z-score, introduit au paragraphe suivant. Puisque $u(x^\star) = 0$, alors*
>
> $$z = \frac{\left|x_{exp} - x^\star\right|}{\sqrt{u(x_{exp})^2 + 0^2}} = \frac{\left|x_{exp} - x^\star\right|}{u(x_{exp})}\,.$$
>
> *Le critère de compatibilité s'écrit ici aussi $z \leq 2$.*

**Interprétation graphique :** voir figure 2. Qualitativement, la valeur attendue est compatible avec les résultats expérimentaux si elle se trouve « à l'intérieur » de l'histogramme des résultats obtenus.

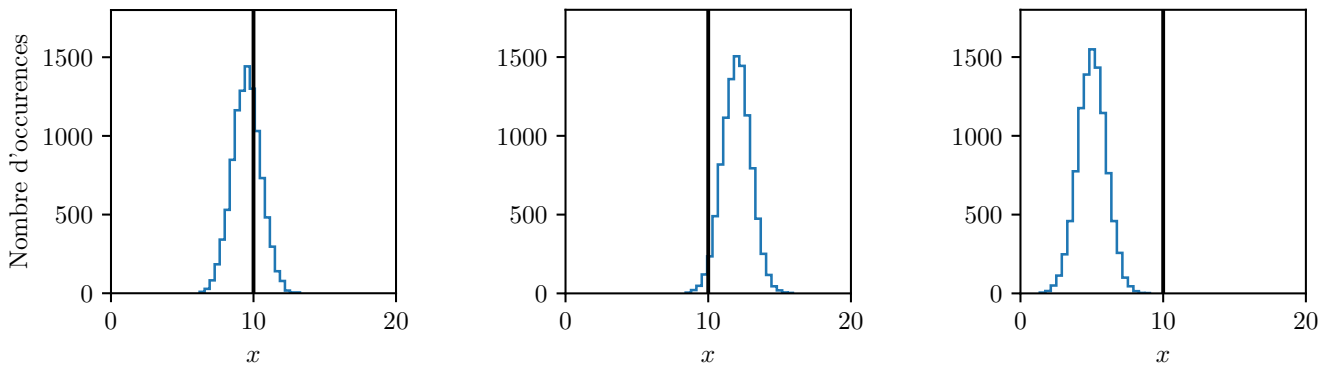


**Figure 2 – Critère de compatibilité entre deux valeurs dont l'une est connue exactement.** On simule 10 000 réalisations d'une même expérience, dont les résultats sont représentés sous forme d'histogramme, à comparer à la valeur attendue représentée par le trait plein. De gauche à droite : $|x^\star - x_{\text{exp}}| = 0{,}5\,u(x_{\text{exp}})$ (valeurs considérées compatibles), $|x^\star - x_{\text{exp}}| = 2\,u(x_{\text{exp}})$ (limite de compatibilité) et $|x^\star - x_{\text{exp}}| = 5\,u(x_{\text{exp}})$ (valeurs incompatibles). Ce nombre de réalisations est bien sûr inatteignable dans une expérience réelle en CPGE, mais permet de bien visualiser l'histogramme.

## V.B - Comparer deux valeurs connues avec incertitude

On souhaite comparer deux valeurs expérimentales $x_1$ et $x_2$, connues avec des incertitudes-types $u(x_1)$ et $u(x_2)$. Ces deux valeurs peuvent par exemple avoir été obtenues par deux binômes ou deux protocoles différents. On se ramène au cas précédent en raisonnant sur l'écart $x_1 - x_2$, dont la valeur attendue est 0 sans incertitude, puisqu'il s'agit de deux mesures de la *même* grandeur physique.

D'après les formules de composition d'incertitudes discutées au paragraphe III.A,

$$u(x_1 - x_2) = \sqrt{u(x_1)^2 + u(x_2)^2}\,.$$

Le critère de compatibilité énoncé au paragraphe précédent s'écrit donc

$$\big|(x_1 - x_2) - 0\big| < 2\,u(x_1 - x_2) \qquad \text{soit} \qquad |x_1 - x_2| < 2\sqrt{u(x_1)^2 + u(x_2)^2}\,.$$

Le $z$-score traduit cette idée de manière compacte.

On appelle **écart normalisé** ou **$z$-score** entre deux valeurs $x_1$ et $x_2$, connues avec une incertitude-type $u(x_1)$ et $u(x_2)$,

$$z = \frac{|x_2 - x_1|}{\sqrt{u(x_1)^2 + u(x_2)^2}}\,.$$

Par convention, les deux valeurs $x_1$ et $x_2$ sont dites **compatibles** si

$$z \leq 2\,.$$

**Interprétation graphique :** On simule 10 000 réalisations des deux expériences (p.ex. travail des deux binômes), voir figure 3, représentées sous forme des deux histogrammes. Si le $z$-score est suffisamment faible, les histogrammes issus des deux expériences se recouvrent « beaucoup », alors qu'ils ne se recouvrent que « très peu » lorsque le $z$-score est élevé.

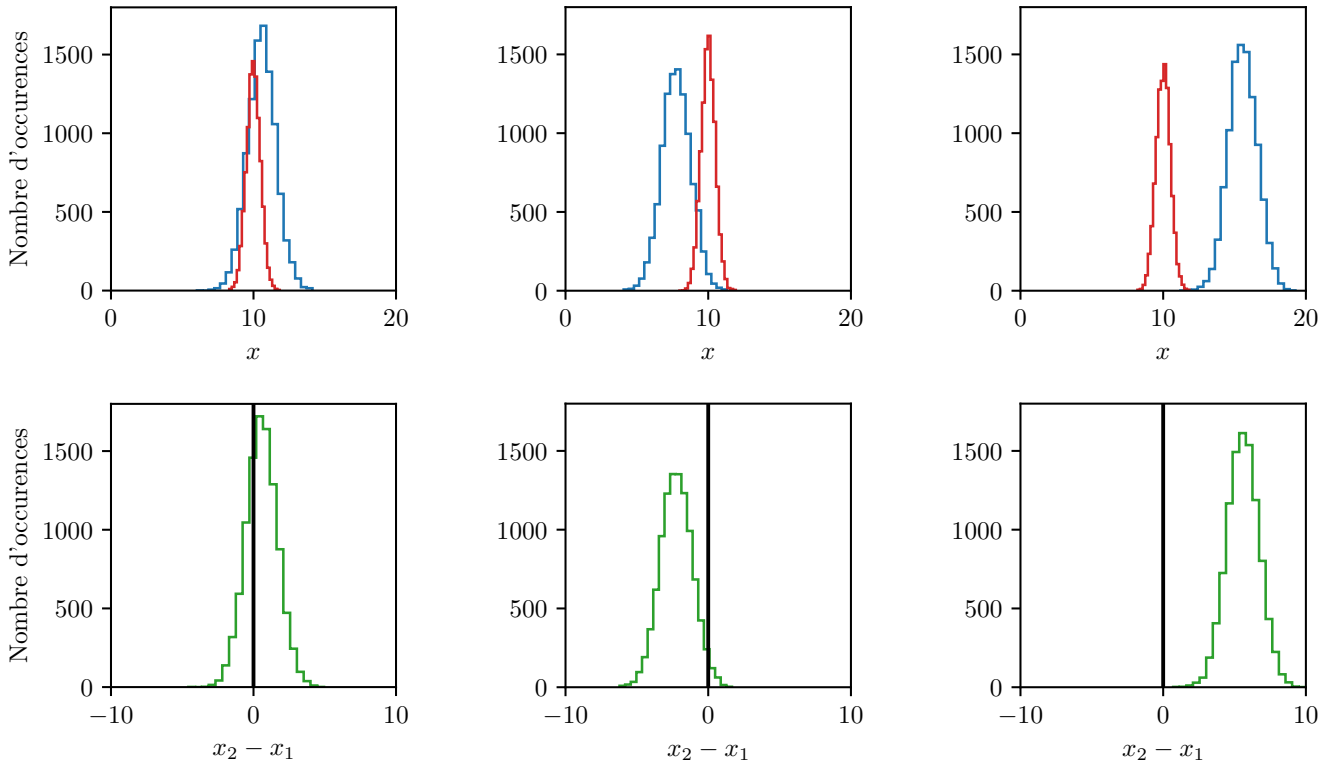


**Figure 3 – Critère de compatibilité entre deux valeurs sur lesquelles l'incertitude-type est connue.** Haut : histogrammes des résultats des deux expériences, bas : histogramme des écarts entre résultats. De gauche à droite : $z = 0{,}5$ (les résultats des deux expériences sont considérées compatibles), $z = 2$ (limite de compatibilité) et $z = 5$ (résultats incompatibles). Ce nombre de réalisations est bien sûr inatteignable dans une expérience réelle en CPGE, mais permet de bien visualiser les histogrammes.

# VI - Valider un modèle théorique

## VI.A - Principe général

Valider un modèle recouvre deux aspects : d'une part, vérifier que les mesures expérimentales sont cohérentes avec la forme fonctionnelle $y = f(x)$ attendue, et d'autre part vérifier que les valeurs des coefficients qui s'en déduisent sont compatibles avec les valeurs prévues par la modélisation. Le plan d'ensemble est le suivant :

❶ Réaliser $N$ mesures indépendantes $(X_n, Y_n)$ et estimer les incertitudes associées.

❷ Valider la forme fonctionnelle du modèle, c'est-à-dire le type de fonction, ce qui ne peut se faire que graphiquement et en se ramenant à une droite, voir paragraphe VI.B.

❸ Si la forme fonctionnelle est validée, déterminer par régression linéaire ou affine les valeurs expérimentales des paramètres intervenant dans le modèle et les incertitudes-types sur ces valeurs. Comparer les valeurs expérimentales aux valeurs attendues par un calcul de $z$-score.

**Attention !** L'étape ❷ est indispensable, déterminer les valeurs des paramètres n'est jamais suffisant. En effet, les calculs calculeront toujours, mais cela ne garantira jamais la pertinence du modèle, qui ne peut être testée à notre niveau *que* graphiquement.

## VI.B - Procédure pour valider la pertinence d'un modèle

### • Linéarisation du modèle

La procédure décrite ici n'est rigoureuse que pour valider un modèle affine dans le cas où l'abscisse est connue sans incertitude. On cherchera donc toujours à s'en approcher le plus possible :

- ▹ les grandeurs $x$ et $y$ ne sont pas forcément les grandeurs directement mesurées, il est fréquent qu'elles s'en déduisent simplement (inverse, racine, etc.) ;
- ▹ la grandeur $x$ portée en abscisse est toujours celle de plus petite incertitude relative : $\dfrac{u(x)}{x} \ll \dfrac{u(y)}{y}$.

On se ramène ainsi à un modèle théorique de la forme $y = ax + b$, pour lequel on dispose de valeurs $(X_n, Y_n)$ et des incertitudes-types $u(Y_n)$ (potentiellement différentes les unes des autres). On commence alors par estimer les paramètres $a$ et $b$ du modèle par régression affine, voir paragraphe IV.B.

### • Validation qualitative

Une première étape de validation est visuelle : les points expérimentaux doivent être répartis de manière régulière de part et d'autre de la droite modèle, sans que les écarts ne fassent apparaître de tendance nette (courbure, regroupement de points, etc.). Cette validation peut être plus simple sur le graphe des résidus, introduit au paragraphe suivant.

### • Validation quantitative par étude des résidus

Une seconde étape de validation repose sur l'étude des **résidus** $\delta_n$ ou des **résidus normalisés** $r_n$, définis comme les écarts entre les points expérimentaux et la droite modèle, éventuellement divisés par l'incertitude-type :

$$\delta_n = Y_n - (aX_n + b) \qquad \text{et} \qquad r_n = \frac{\delta_n}{u(Y_n)}.$$

Le modèle est considéré valable si tous les résidus normalisés sont inférieurs ou proches de 2 :

$$\forall n,\ r_n = \frac{|\delta_n|}{u(Y_n)} \lesssim 2$$

On retrouve un critère de validation qui évoque celui du $z$-score, même s'il en diffère un peu car les incertitudes $u(a)$ et $u(b)$ ne sont pas considérées.

➽ **Pour approfondir :** $2u(Y_n)$ définit l'intervalle de confiance à 95 %, voir paragraphe I. Ainsi, même pour un modèle valide, la probabilié qu'un résidu normalisé soit supérieur à 2 est de l'ordre de 5 %. Autrement dit, sur une série de 20 mesures, il devient assez probable qu'une d'entre elle ait un résidu supérieur à 2, ce qui explique pourquoi la condition de validité ne peut pas être formulée de manière stricte. On comprend également que, comme pour le $z$-score, le seuil de 2 est conventionnel. ■

Pour visualiser graphiquement les résidus, il est souvent pratique de représenter la courbe modèle et les points expérimentaux accompagnés de **barres d'incertitudes** de hauteur égale au double de l'incertitude-type. On peut aussi représenter directement les résidus normalisés en fonction de $x$.

### Code clés en main – exemple 8

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

x = np.array([ ... ])
y = np.array([ ... ])
uy = ... # valeur unique ou tableau

a,b = np.polyfit(x,y,1)

# Tracé avec barres d'incertitudes :
plt.figure()
plt.errorbar(x,y,yerr=2*uy,fmt='+') # mesures avec barres d'incertitude
plt.plot(x, a*x+b, '-') # courbe modèle

# Graphe des résidus normalisés :
r = (y - a*x+b)/uy # tableau car numpy calcule terme à terme
plt.figure()
plt.plot(x, r, '+')
```

- **Exemples**

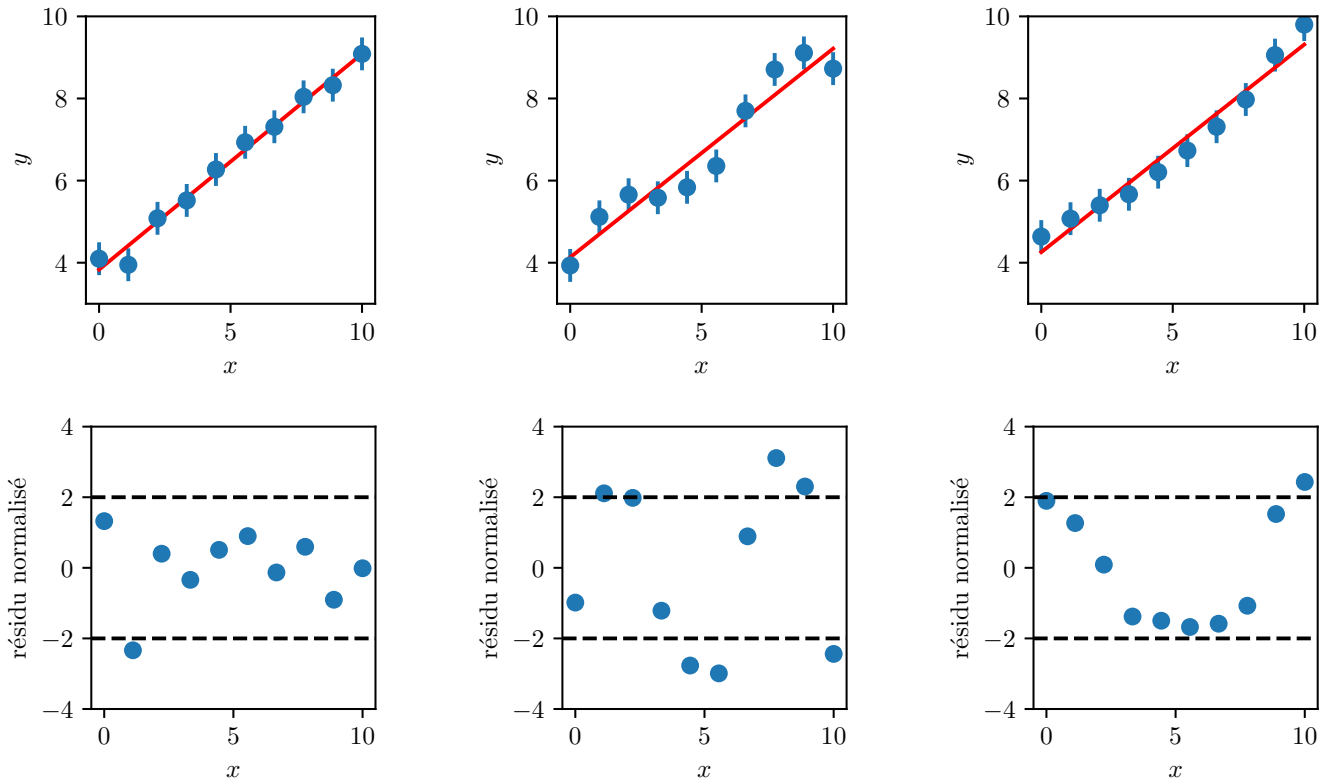


**Figure 4 – Validation graphique d'un modèle.** Gauche : bien qu'un des résidus normalisés soit inférieur à −2, le modèle est valable. Centre : trop de résidus normalisés sont supérieurs à 2 en valeur absolue, le modèle est rejeté. Droite : bien que les résidus soient suffisamment faibles, on constate une nette courbure dans leur répartition, le modèle est donc rejeté.