Méthodes numériques

Régression linéaire

Objectif : effectuer des régressions linéaires avec le langage Python pour exploiter des données.

I Régression linéaire pour valider un modèle

Exemple introductif

On part d'une expérience que l'on souhaite modéliser, c'est-à-dire qu'on souhaite décrire ses résultats par un modèle physique et/ou par l'utilisation d'une loi physique. Une fois la modélisation effectuée, il faut la *tester* en la confrontant aux mesures. Ceci peut être fait via une régression linéaire.

On prend l'exemple de la loi de Snell-Descartes : $n_1 \sin i_1 = n_2 \sin i_2$. Pour la tester, on réalise des mesures d'angles de réfraction à une interface air (milieu 1) – plexiglas (milieu 2).

i_1 (rad)	0	0,17	0,35	0,52	0,70	0,87	1,05	1,22
i_2 (rad)	0	0,12	0,22	0,35	0,42	0,51	0,67	0,71

On écrit alors la loi de Descartes sous la forme $y = ax + b$. Il suffit de poser $y = \sin i_2$ et $x = \sin i_1$.

1 - Quelle est alors l'expression de a et celle de b ?

2 - Entrer les valeurs ci-dessus de i_1 et de i_2 dans deux listes `I1` et `I2`.

Puis créer des listes x et y en accord avec leurs définitions. On peut faire par exemple `x = [sin(i1) for i1 in I1]` et de même pour `y`.

Puis tracer y en fonction de x (utiliser l'argument `'o'` de `plt.plot` pour afficher des points et non pas une ligne).

On effectue ensuite la régression linéaire à l'aide de la fonction polyfit.

Polyfit

`a,b = np.polyfit(x,y,1)` permet d'effectuer une régression linéaire sur les listes x et y.
Elle retourne le meilleur couple a, b, tel que la droite d'équation $y = ax + b$ soit au plus près des points (x_i,y_i).

Remarque : le troisième argument (1 ici) signifie que le modèle est en $ax + b$. Mettre 2 ferait un modèle en $ax^2 + bx + c$, etc...

3 - Utiliser la fonction polyfit pour faire une régression linéaire sur les valeurs x et y. Vous devez obtenir les valeurs de la pente a et de l'ordonnée à l'origine b.

Tracer ensuite la droite $y = ax + b$ sur le même graphique que précédemment. On peut utiliser `plt.plot(x, [a*x+b for x in x])`.

Conclure : les points semblent-ils alignés ? La loi de Snell-Descartes semble-t-elle vérifiée ? Si oui, en déduire une estimation de l'indice optique du plexiglas utilisé.

Un second exemple

On veut vérifier la loi qui donne la vitesse du son dans un gaz parfait. Deux modèles sont possibles :

- Un où les compressions et détentes des volumes d'air lors du passage de l'onde sont adiabatiques. La vitesse est alors $c = \sqrt{\dfrac{\gamma RT}{M}}$, avec T la température, R la constante de gaz parfaits, γ l'indice adiabatique, et M la masse molaire. Pour l'air on a $\sqrt{\dfrac{\gamma R}{M}} = 20{,}1\,\mathrm{m \cdot s^{-1} \cdot K^{-0{,}5}}$.
- Un où les compressions et détentes sont isothermes (modèle initialement établi par Newton). On a alors $c = \sqrt{\dfrac{RT}{M}}$. Pour l'air on a $\sqrt{\dfrac{R}{M}} = 16{,}9\,\mathrm{m \cdot s^{-1} \cdot K^{-0{,}5}}$.

On a réalisé les mesures suivantes :

T (K)	240	260	280	300	320	340	360
c (m/s)	310	325	331	342	358	370	378

4 - Que faut-il poser pour x (en fonction de T) et pour y (en fonction de c) afin qu'un tracé de y en fonction de x permette de vérifier une des deux lois ?

Sous Python, créer une liste `T` avec les valeurs de T et une liste `c` avec celles de c. Comme tout à l'heure, créer les listes `x` et `y`, puis faire le tracé de x en fonction de y, faire une régression linéaire et l'afficher sur le même graphique.

Conclure : une loi où c est proportionnelle à $\sqrt{T}$ semble-t-elle bien décrire les données ? Si oui, la valeur de a tend-elle à confirmer le modèle adiabatique ou isotherme ?

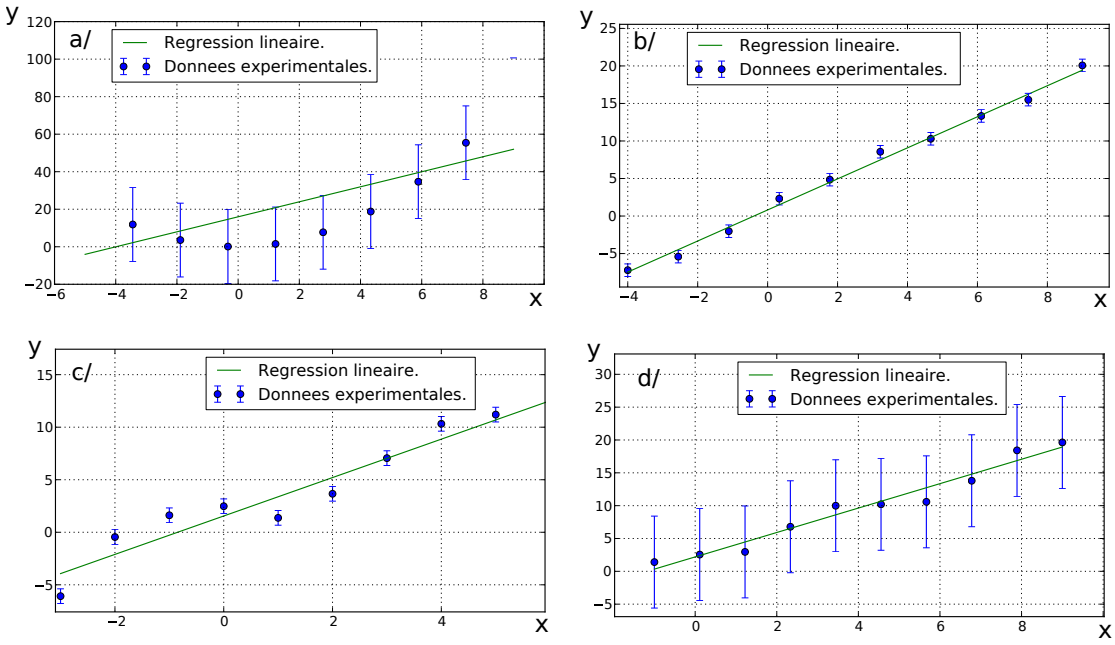
Un meilleur critère de validation nécessite d'introduire les incertitudes sur les points mesurés

On voit sur l'exemple précédent qu'il manque quelque chose pour vraiment valider si le modèle linéaire $y = ax + b$ est compatible avec les données ou non : il manque la prise en compte des incertitudes sur les données.

Critères de validation d'un modèle linéaire

Pour cela, il faut 1/que les points ne suivent pas une tendance nettement non linéaire (une courbure, une oscillation régulière, etc.), et 2/ que la droite de régression passe proche des points, à une distance inférieure à deux fois la barre d'incertitude-type.

Quelques exemples :



- Cas **a** : la droite de régression passe par les barres d'erreur, mais cependant on note une nette courbure : le modèle linéaire n'est donc certainement pas bon.
- Cas **b** : pas de tendance particulière, et la droite de régression passe par les barres d'erreur : on valide le modèle linéaire.
- Cas **c** : la droite de régression ne passe pas par les barres d'erreur : on rejette le modèle. (il faudra vérifier qu'on n'a pas pris des incertitudes trop petites)
- Cas **d** : pas de tendance nette et la droite passe par les barres d'erreur : on valide le modèle. Remarquons toutefois que l'incertitude est grande et que d'autres modèles, non linéaires, pourraient aussi convenir : la conclusion n'est donc pas très forte. Il faut aussi vérifier qu'on n'a pas surestimé les incertitudes.

→ C'est seulement si on valide le modèle linéaire qu'on peut ensuite exploiter les valeurs de a et de b.
On reprend les mesures de vitesse du son, en supposant que l'incertitude-type relative sur les valeurs de c est de 1 %. On a donc à chaque fois $u(c) = 0{,}01c$.

5 - Comme on a normalement posé $y = c$, l'incertitude sur y est la même que celle pour c. Créer une liste `uy` qui contient les incertitudes-types sur les valeurs de y (par exemple faire `uy = [y*0.01 for y in y]`).

6 - Utiliser `plt.errorbar(x,y,yerr=uy,fmt='o')` pour faire un tracé avec des barres d'erreur (le `'o'` sert à avoir des points non reliés). Le faire sur le même graphique que précédemment.

Vous pouvez alors conclure de façon plus précise sur l'adéquation du modèle !

II Régression linéaire et obtention des incertitudes sur a et b

Il manque une dernière chose : pouvoir obtenir une incertitude sur les valeurs de a et de b retournées par `polyfit`. Ceci permettra alors de conclure plus précisément, par exemple

sur l'accord entre la valeur de a et la pente attendue pour le modèle adiabatique ou isotherme.
Pour faire ceci, on va utiliser une méthode Monte Carlo (de tirage au sort), en générant plusieurs jeux de données (x_k,y_k) et en effectuant à chaque fois une régression linéaire.

7 - Dans le cas de la vitesse du son, seule l'incertitude sur y (qui est la vitesse c) est disponible. Il faudra donc uniquement tirer au sort des valeurs de y.

Traduire en Python les étapes décrites ci-dessous.

- ▶ Fixer un nombre N très grand.
- ▶ Créer deux listes vides `liste_a` et `liste_b` pour stocker les pentes et les ordonnées à l'origine des régressions.
- ▶ Pour chaque i compris entre 0 et $N-1$, réaliser :
  - ▷ Créer une liste vide `ytirage` qui va contenir les valeurs de y tirée au sort.
  - ▷ Pour chaque k compris entre 0 et `len(y)-1`, réaliser un tirage aléatoire d'une valeur de y_k donnée par une loi de probabilité uniforme entre $y_k - \sqrt{3}\,u(y_k)$ et $y_k + \sqrt{3}\,u(y_k)$, et l'ajouter à la liste `ytirage`.
    Rappel : `np.random.uniform(a,b)` génère un nombre aléatoire entre a et b avec une loi uniforme.
  - ▷ Réaliser une régression linéaire sur les listes `x` et `y_tirage`. Puis ajouter dans les listes correspondantes la pente et l'ordonnée à l'origine de cette régression.
- ▶ Calculer la valeur moyenne (`np.mean`) de `liste_a`. Ceci donne la valeur de a. Calculer l'écart-type (`np.std`) de `liste_a`. Ceci donne l'incertitude-type, qui était l'objectif de cette méthode.

  Faire de même avec b.
- ▶ Tracer enfin le même graphique que précédemment : les points expérimentaux, leur incertitude, et la régression linéaire.

8 - Vous devez maintenant avoir la valeur de a, de $u(a)$, et de b et de $u(b)$.

Conclure alors pour a, en indiquant si cette valeur et son incertitude sont compatibles avec la valeur attendue pour le modèle adiabatique (calculer le z-score ou écart normalisé). Faire de même pour la valeur attendue pour le modèle isotherme. Conclusion ?

Pour b, sa valeur est-elle compatible avec la valeur attendue (qui est 0 dans les deux modèles) ?

En conclusion, on voit bien que c'est l'estimation des incertitudes qui permet de conclure 1/ sur la validité d'un modèle linéaire (voir si la droite passe par les barres d'erreurs des données), et 2/ sur la compatibilité entre les valeurs de a et de b avec celles attendues (en exploitant $u(a)$ et $u(b)$, obtenus par méthode Monte Carlo, pour calculer l'écart normalisé à la valeur attendue).

Notons pour finir que certains logiciel, comme Regressi, font automatiquement le calcul de $u(a)$ et $u(b)$.