

Régression linéaire

Au programme



Savoirs

- ◇ Analyser les résultats obtenus à l'aide d'une procédure de validation : analyse graphique intégrant les barres d'incertitude ou analyse des écarts normalisés.



Savoir-faire

- ◇ Utiliser un logiciel de régression linéaire afin d'obtenir les valeurs des paramètres du modèle.
- ◇ Simuler, à l'aide d'un langage de programmation ou d'un tableur, un processus aléatoire de variation des valeurs expérimentales de l'une des grandeurs – simulation Monte-Carlo – pour évaluer l'incertitude sur les paramètres du modèle.



Sommaire

I Principe	2
II Réalisation logicielle	2
A À la calculatrice	2
B En Python	3
III Pertinence du modèle	4
IV Cas non linéaires	5

I Principe

La pratique de la science passe par deux pivots inséparables : la *théorie* et la *mesure*. La première pose un cadre de travail pour aboutir à des conclusions sur le fonctionnement du monde physique sous la forme de relations mathématiques, et la seconde relève des valeurs expérimentales du monde physique pour en archiver les propriétés effectives.

L'une et l'autre se nourrissent conjointement : une théorie physique pose une loi et la mesure permet de la tester, ou un relevé expérimental particulier permet de mettre en évidence un point sombre des cadres théoriques actuels. Il reste que les sciences physiques ont pour unique objectif de rendre compte du réel ; ainsi, si l'expérience est bien contrôlée, **c'est toujours la mesure qui clôt la discussion**.

Dans ce cadre, il est commun de disposer de deux jeux de données réelles représentant des **grandeurs reliées** entre elles par une loi physique (disons x et y), et normalement compatibles avec une **relation linéaire** mathématique :

$$y = ax + b$$

avec a et b des coefficients propres à la relation :

- a le *coefficient directeur* ;
- b l'*ordonnée à l'origine*.

La régression linéaire¹ vise à **tester l'accord entre mesure et théorie**, et le cas échéant à **estimer les coefficients** et leurs incertitudes. Elle repose sur la détermination de la meilleure droite qui serait susceptible de représenter le nuage de points donné en deux dimensions par x et y , en l'occurrence en minimisant la somme des distances au carré entre chacun des points et la droite. Cet **écart aux points mesurés** est caractérisé par un coefficient de corrélation, noté r , ou de régression, noté r^2 , et compris entre 0 (nul) et 1 (parfait).

Régression linéaire

Une régression linéaire est une **opération mathématique** entre des points de mesure x_i et y_i qui consiste à trouver les meilleurs coefficients a et b tels que $ax_i + b$ soient les plus proches en moyenne des points de mesures y_i .

Cette régression peut être faite à la main, mais la détermination des coefficients ne sera pas rigoureuse. Nous allons ici détailler le processus pour une régression *via* un logiciel.

II Réalisation logicielle

A À la calculatrice

TI 83+

- Appuyer sur **stats**.
- Dans le menu EDIT choisir 1 : **Editer**.
- Entrer les deux séries de données : l'abscisse dans la première colonne L1, et l'ordonnée dans la seconde colonne L2.
- Dans le menu **graph stats**, choisir la première ligne.
- Choisir **Aff** pour afficher, puis le premier type, la liste L1 pour **ListeX** et la liste L2 pour **ListeY**.

1. Il faudrait en toute rigueur parler de régression *affine* et non linéaire.

- Appuyer sur **graphe** pour afficher le nuage de points.
- Pour déterminer les coefficients a , b et r ou r^2 , retourner dans **stats** puis aller dans le menu **CALC** et choisir 4 : **RegLin(ax+b)**. Appuyer sur **ENTER**.

r non indiqué

- Appuyer sur **2nde** puis **catalog**.
- Choisir **Diagnostic On** ou **corelAff** selon la langue, appuyer deux fois sur **ENTER**.

Tracer la courbe

- Aller dans **f(x)**, appuyer sur **var**.
- Choisir 5 : **Statistiques**.
- Dans le menu **EQ** choisir 1, puis **ENTER**.
- Aller dans **graphe** pour voir la droite.

Casio 35+

- Aller dans le mode **STAT**.
- Entre les deux séries de données dans **List1** et **List2**.
- Appuyer sur **GRAPH** puis **SET**.
- Choisir le type de graphique **GraphType** : **Scatter**. Mettre **List1** dans **XList** pour l'abscisse et **List2** dans **YList** pour l'ordonnée.
- Appuyer sur **ENTER**, et appuyer sur **GRAPH** 1 pour visualiser le nuage de points.
- Appuyer sur **CALC** puis **X** puis sur **ax+b** pour obtenir une régression linéaire $y = ax + b$.
- Une fenêtre **LinearReg** affiche la pente a , l'ordonnée à l'origine b et le coefficient de corrélation r^2 .
- Pour visualiser la droite, choisir **DRAW** en bas à droite.

Numworks

Travail en cours...

Attention !

Il est plus que courant d'inverser les deux listes !

B En Python

```
# =====
#                               Imports                               #
# =====

import numpy as np              # pour les tableaux et la gestion des données
import matplotlib.pyplot as plt # pour les graphiques, l'affichage des données

# =====
#                               Données expérimentales                #
# =====

X = np.array([0, 2, 4, 6, 8, 10]) # À modifier
Y = np.array([0.5, 7.9, 11, 17.5, 26, 31.8]) # À modifier
```

```

# ===== #
#                                     Régression                                     #
# ===== #

# Donne a le coefficient directeur et b l'ordonnée à l'origine
a, b = np.polyfit(X, Y, 1)
# Affiche a et b. .3f pour 3 valeurs après la virgule
print(f"a = {a:.3f}, b = {b:.2f}")

# ===== #
#                                     Utilisation                                    #
# ===== #

# Liste fine des abscisses à tracer
# découpe l'intervalle [min(X), max(X)] en 100 points
xliste = np.linspace(min(X), max(X), 100)

# Fonction qui à x, a, b associe y
def yfunc(abscisse, coeff_dir, ord_ori):
    return coeff_dir*abscisse + ord_ori

# Liste des points y_i obtenus par régression
yliste = yfunc(xliste, a, b)

# ===== #
#                                     Tracé                                       #
# ===== #

plt.figure(figsize=(8, 6))
plt.grid()
plt.xticks(fontsize=20)
plt.yticks(fontsize=20)
plt.xlabel('$grandeur$ en UNITÉ', fontsize=20)
plt.ylabel('$grandeur$ en UNITÉ', fontsize=20)

plt.plot(xliste, yliste,
         'r', label='Régression linéaire')

plt.title('Titre efficace et descriptif', fontsize=20)
plt.legend(fontsize=15)
plt.show()

```

III Pertinence du modèle

Une fois la régression effectuée, il faut juger de sa validité. En effet, **une régression linéaire va toujours « fonctionner »**, mais ceci n'assure en rien qu'elle est bonne.

* Attention !!

La seule façon valable de conclure à la validité d'une régression linéaire est d'observer l'alignement des points avec la droite de régression.

Une erreur des plus basiques est de ne considérer que l'accord numérique *via* la valeur de r ou r^2 . S'il est *nécessaire* d'avoir et d'écrire

$$|r| > 0,999x \quad \text{ou} \quad r^2 > 0,99x$$

où x représente le premier chiffre autre qu'un 9, il est loin d'être *suffisant* et c'est la vérification visuelle qui prévaut. Le modèle sera validé sous deux conditions :

- 1) Les points de mesure sont **bien alignés** ;

- 2) la droite de régression passe le plus proche de tous les points possibles, en incluant leurs incertitudes-type.

Ainsi, dans le cas 1 de la Figure 4.1, le modèle est validé. Dans le second, il ne l'est pas.

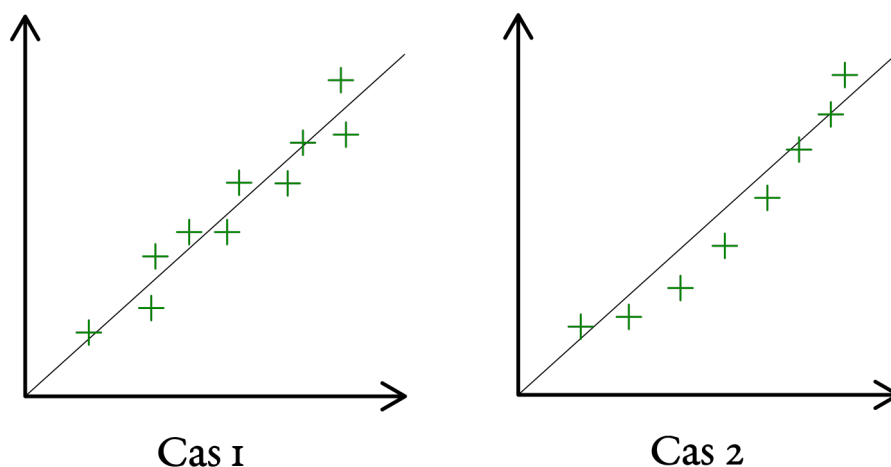


FIGURE 4.1 – Exemples de régressions linéaires. Le cas 1 est un modèle validé. Le second ne l'est pas.

IV Cas non linéaires

Dans de nombreux cas, la loi à vérifier n'est ni affine ni linéaire. C'est le cas de la loi de SNELL-DESCARTES sur la réfraction entre deux milieux d'indice optique n_1 et n_2 :

$$n_1 \sin(i_1) = n_2 \sin(i_2)$$

Avec des mesures de i_1 et de i_2 , il n'est pas possible d'obtenir directement une droite. Il faut dans ce cas tracer

$$y = ax + b \quad \text{avec} \quad y = \sin(i_1) \quad \text{et} \quad x = \sin(i_2)$$

L'adaptation en **Python** est relativement directe, mais il faut maîtriser les opérations sur des listes *via* les calculatrices.

TI 83+

- Aller sur la case L3, puis ENTER
- Entrer la formule (L3=sin(L2) ici), puis ENTER

Casio 35+

- Dans menu RUN, touche OPTN et LIST
- Entrer la formule (sin(List2) → List3 ici), puis EXE

Numworks

Travail en cours...