TD3 - Statistique

October 21, 2020

1 Statistiques

Dans ce TP, nous utiliserons plusieurs fonction de numpy et scipy. En particulier :

- Distribution aléatoire : package numpy.random, fonctions rad, normal. En général ces fonctions ont un paramètre size qui permet de faire un grand nombres de tirages. Il est aussi possible de leur envoyer un tableau de paramètre.
- Histogramme : on pourra utiliser la fonction numpy.histogram ou la fonction matplotlib.pyplot.hist. Il est important d'utiliser l'argument optionel bins en mettant une séquence. Par exemple numpy.histogram(data, bins=(a,b)) calcule le nombre de point entre a et b.

2 Estimateur

2.1 Loi normale

On considère une variable aléatoire distribuée selon une loi normale d'écart type σ et de centre μ:

$$P(x|\mu,\sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

On considère un échantillon de N réalisations de la loi : x_i.

La vraissemblance est donnée par :

$$\mathcal{L}(\mu,\sigma) = \prod_i P(x_i|\mu,\sigma)$$

Plutôt que de maximiser la vraissemblance, on va minimiser $-\ln\mathcal{L}$. Démontrer que la maximum est obtenu avec :

$$\hat{\mu} = \frac{1}{N}\sum_i x_i$$

et

$$\hat{\sigma} = \frac{1}{N}\sum_i (x_i - \hat{\mu})^2$$

Les estimateurs que l'on a obtenus pour l'espérance μ et pour l'écart type σ de la loi sont donnés par la moyenne et l'écart type des points.

1. Écrire les fonctions $\hat{\sigma}$ et $\hat{\mu}$
2. Un estimateur est caractérisé par son biais et son écart type. Calculer numériquement le biais et l'écart de ces estimateurs pour $N = 10$ et $N = 100$ (pour $\mu = 0$ et $\sigma = 1$). Ce calcul se fera en simulant un nombre M de fois l'estimateur (typiquement M=100000). Sont ils biaisé ? Semblent-ils convergent ?

2.2 Loi uniforme

On considère une loi uniforme entre a et b ($b > a$). L'espérance de cette loi est $\mu = (a + b)/2$. On note $L = b - a$ (largeur de l'intervale). L'écart type est proportionnel à L et on peut montrer qu'il vaut $\frac{L}{2\sqrt{3}}$.

3. Caractériser pour cette loi l'estimateur de μ donné par la moyenne et de L donné, au facteur $2\sqrt{3}$ près par l'écart type. On prendra $N = 10$ et $N = 100$.
4. Démontrer que $\mathcal{L}(a, b) = \frac{1}{(b-a)^N}$ si $a \leq \min x_i$ et $b \geq \max x_i$ et 0 sinon. En déduire que le maximum de vraissemblance conduit à :

   $$\hat{a} = \min x_i$$

   et

   $$\hat{b} = \max x_i$$

   soit

   $$\hat{\mu} = (\max x_i + \min x_i)/2$$

   et

   $$\hat{L} = (\max x_i - \min x_i)$$

   .
5. Caractériser ces deux estimateurs.
6. Comparez graphiquement l'écart type des deux estimateurs de μ pour N entre 10 et 1000
7. A l'aide d'un histogramme, comparez pour $N = 20$ la distribution de ces deux estimateurs.

3 Ajustement d'une courbe

On va utiliser la fonction curve_fit du packagescipy.optimize. Cette fonction permet aussi d'obtenir l'incertitude des paramètres sous forme d'une matrice de corrélation. Cette fonction s'utilise de la façon suivante

[ ]:
```
p_opt, cor_mat = curve_fit(fonction_de_fit, data_x, data_y, p_ini)
```

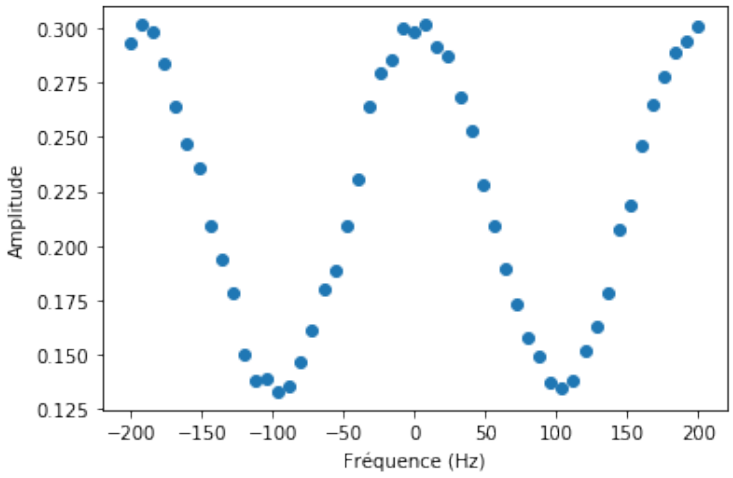
où

- fonction_de_fit(x, p1, p2, ..., pn) est la fonction de fit. Les variables p1, .., pn sont les paramètres de la fonction de fit.
- data_x et data_y sont les points de mesure.

- p_ini sont les paramètres initiaux (sous forme d'une liste/tuple/array).
- p_opt seront les paramètres optimaux et cor_mat est la matrice de correlation entre les paramètres.

4 Fit de franges d'interférence

On souhaite ajuster les franges d'un interféromètre atomique. Les données sont dans le fichier-data/fit_sinus.dat. La première colonne du fichiers (axe x) représente une fréquenceen Hz. La seconde colonne représente la population mesurée pour une fréquence donnée. L'objectif est de trouver la position de la frange centrale.



On ajustera par une fonction cosinus avec une amplitude, un décalage vertical, une position centrale et une largeur ajustable (soit quatre paramètres en plus de fréquence).

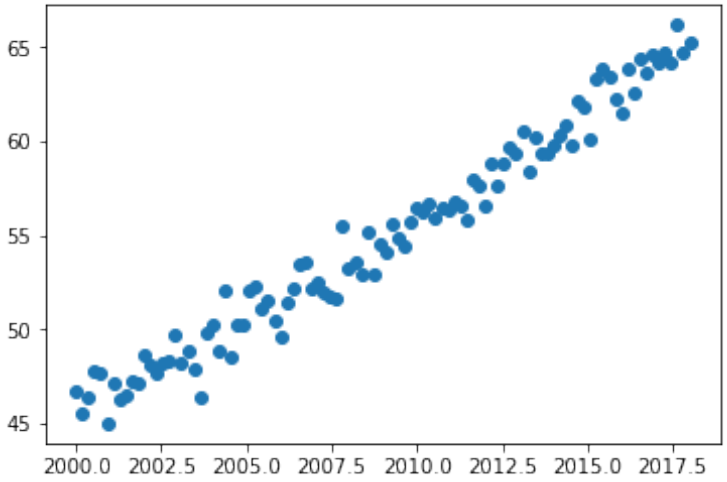
1. Écrivez la fonction de fit qui dépend des paramètres ci dessus. On appellera frange(x, …). Tracez la courbe pour x entre ±220 Hz. On prendra un inter-frange de 200 Hz.
2. Chargez et tracez les données. On représentera les données par des points (plot(…,…, 'o')).
3. Calculez les paramètres optimaux. Quelle est la position la frange centrale ? Représentez les points et la courbe.
4. Quelle est l'incertitude sur la position de la frange centrale ?

5 Corrélation

On simule une jeu de données.

[7]:
```
np.random.seed(0) # pour que le générateur "aléatoire" soit le même pour tout le␣
↪monde
N = 100
x = np.linspace(2000, 2018, N)
y = np.arange(N)*0.2 + 45 + np.random.normal(size=N)
```

```
plt.plot(x, y, 'o');
```



1. Tracez et ajustez les données par une une droite $y = ax + b$.
2. Quel est l'incertitude sur b? Qu'en pensez-vous ? Qu'elle est la signification physique de b ?
3. Calculez la valeur et l'incertitude de votre fit en $x = 2010$.
4. Trouvez une fonction de fit plus pertinente pour ce problème.