TD IMSP 0 : Le bruit résiduel de la mesure

Introduction :

Nous avons vu en TP quelques éléments de calcul de propagation des incertitudes de mesure (formules quadratiques en particulier) et on a décrit à cette occasion une distribution statistique fondamentale : la **distribution normale** ou **gaussienne** (interprétant alors l' « écart de mesure » comme le cumul de microévénements contraires équiprobables). *(Le théorème central limite lui confère une importance considérable en statistiques)*

Nous allons donc profiter de l'outil informatique (langage Python) pour :

1- Analyser des dispersions de mesures (« bruits ») de types différents (distribution et spectre)

2- Confirmer les « formules » de propagation de l'incertitude (LPU) par la méthode statistique de Monte Carlo (MMC) *(qui nous permettra de simuler diverses distributions de variables (uniformes,triangulaires ou gaussiennes) ainsi que celle de la grandeur fonction de ces variables pour mesurer son écart-type et sa moyenne et comparer à la prédiction LPU)*

Deux documents annexes sont joints :

- un tableau comparatif du déroulement des deux méthodes d'évaluation LPU et MMC
- un article scientifique illustrant la MMC sur deux exemples types de physique simples (mesures de puissance thermique et de masse conventionnelle) *(Le second exemple montrant la supériorité de la MMC dans le cas de défauts de linéarité où les calculs de LPU doivent être menés à un degré supérieur : LPU2 contre LPU1)*

I. 50 nuances de bruits

Si l'origine sonore du terme est intuitive, il est couramment utilisé dans le cadre de la mesure en électronique : vous avez eu l'occasion d'observer du « bruit »sur les oscilloscopes numériques lors d'observations de tensions continues à des calibres particulièrement faibles (et donc sensibles).
Lors de la mesure d'une grandeur, on distingue l'erreur ***aléatoire*** *de l'erreur* ***systématique :*** *l'erreur aléatoire est de moyenne nulle. Mais on complète parfois cette description de l'écart à la valeur moyenne des mesures comme le cumul (somme) d'un nombre important de micro-événements de même amplitude, opposés et équiprobables. Dans ce cas, l'histogramme des écarts « instantanés » mesurés doit tendre vers une loi normale : courbe de Gauss.*

Bruit blanc gaussien

« Un bruit blanc est un processus stochastique qui possède la même densité spectrale de puissance à toutes les fréquences » *source : Wikipedia*
On interprétera cette définition comme un **spectre plat continu** : totalement à l'opposé d'une DSF.
Un bruit **blanc** sera dit « **gaussien »** si les valeurs d'amplitudes suivent une **loi normale** de moyenne et de variance données.
(Les bruits blancs sont utilisés par exemple pour la relaxation ou le masquage des acouphènes)

Dans cette partie, nous allons utiliser des fichiers sonores au format .wav correspondants donc véritablement à des bruits (souffles) de 10 ou 30 secondes avec des fréquences d'échantillonnage différentes . Certains sont générés « informatiquement » d'autres sont des enregistrements réels.

[fichiers sonores présents sur vos espaces de travail : B1.wav, B2.wav, B3.wav, B4.wav]

Vous devez les importer et récupérer les valeurs (entiers relatifs) par **scipy.io.wavfile.read**
(L'import du module **scipy** ne servira qu'à cela)
Vous récupérez alors la fréquence d'échantillonnage fe et la liste des valeurs d'amplitude.
Vous affichez la fréquence d'échantillonnage et la durée de l'enregistrement.
Vous utilisez **numpy** pour calculer la moyenne et l'estimateur de l'écart-type de cet échantillon.

Vous devez alors présenter à la verticale trois graphes dans une seule fenêtre graphique (1 colonne et trois rangées donc) :
- La courbe listant les valeurs d'amplitudes : présentation temporelle du bruit
- L'histogramme **normalisé** de ces valeurs d'amplitudes superposé à la loi normale de même moyenne et écart-type pour « tester » le caractère gaussien.
- La Transformée de Fourier Rapide du signal pour « tester » la « couleur » du bruit *:* présentation fréquentielle du bruit *(une représentation en échelles logarithmiques sur les deux axes serait un plus)*

Les bibliothèques de fonctions nécessaires seront :

-**numpy** à renommer np
-**matplotlib.pyplot** à renommer plt
-**scipy** à renommer sc

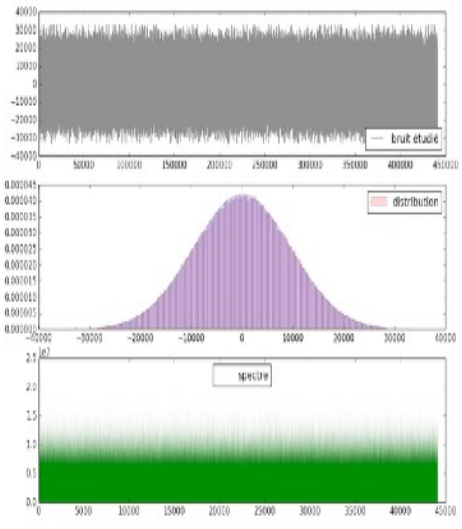
La densité de probabilité de la **loi normale** autour d'une moyenne nulle s'écrit :

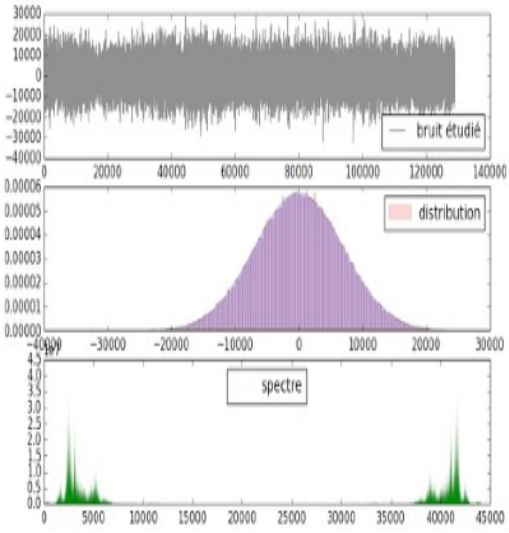
$$\frac{dP}{dx} = \frac{1}{\sigma\sqrt{2\pi}} . e^{-\frac{x^2}{2\sigma^2}}$$

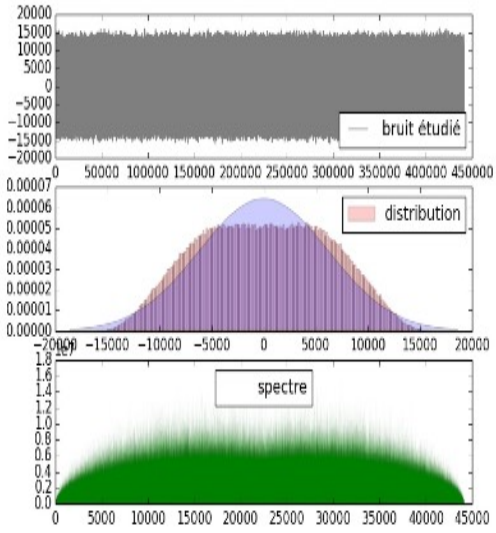
La FFT (Transformée de Fourier Rapide) de **numpy** est obtenue immédiatement par :

abs(np.fft.fft(valeurs))

Ci-dessous trois « bruits » dévoilés :







II. Comparaison LPU et MMC

Les formules quadratiques de propagations de l'erreur nécessitent généralement des calculs de dérivées partielles (premières au moins) de la grandeur finale par rapport à chacune de ses variables d'influence (cf article annexe). On y incorpore également les incertitudes-type évaluées en « choisissant » des distributions diverses pour chaque variable (rectangulaire (ou uniforme), triangulaire, « dérivée d'arcsinus » , gaussienne etc.). La relation (connue) entre demi-largeur et incertitude-type dépend donc du profil de densité de probabilité de la distribution.

Grâce au module « random » de fonctions aléatoires de **numpy**, nous allons pouvoir réaliser la méthode de Monte Carlo et ainsi valider les formules quadratiques utilisées en LPU.

Il s'agit donc de simulations de tirages pour obtenir la distribution empirique de la grandeur de sortie. On simule la PDF (fonction densité de probabilité) de chacune des variables. On génère 1 million de réalisations par tirage au sort dans la distribution de chaque variable. On obtient alors 1 million de valeurs pour la grandeur de sortie qui constituent la distribution empirique du mesurande. Il reste alors à calculer : moyenne, estimateur d'écart-type pour bâtir un intervalle de confiance à 95% et comparer à celui qu'aurait donné la LPU.

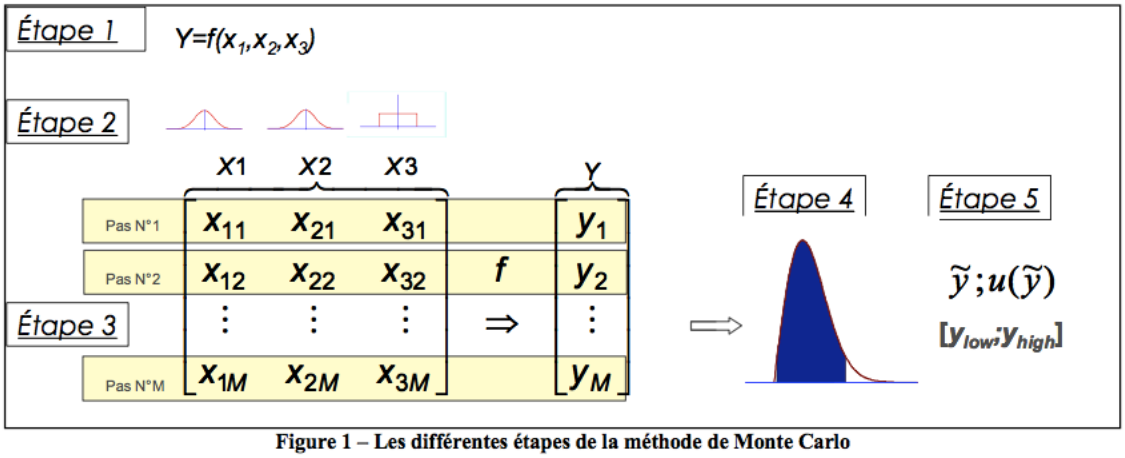
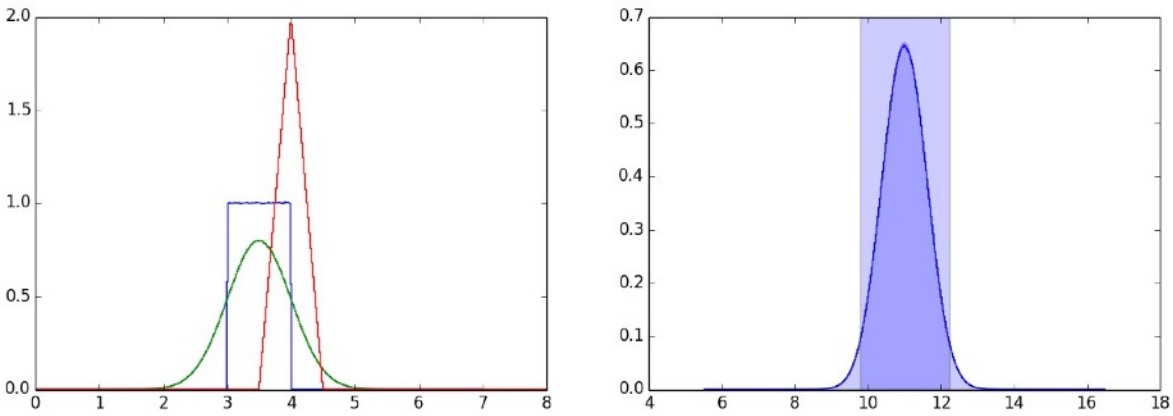


Figure 1 – Les différentes étapes de la méthode de Monte Carlo

On en profitera pour remarquer que des distributions différentes donneront souvent une distribution convergeant vers une gaussienne :



« La convolution d'un grand nombre de fonctions de densités de probabilités tend toujours vers une fonction gaussienne. »

Vous aurez donc besoin d'utiliser le module **random** de **numpy**

import **numpy.random** as **npr**

A vous de vous renseigner sur les arguments de distributions :
- Rectangulaire (ou uniforme) : **npr.uniform**()
- Triangulaire : **npr.triangular()**
- Gaussienne (ou Normale) **: npr.normal()**

Je vous propose, dans un premier temps, d'additionner simplement les trois variables des distributions de la figure de la page précédente, de représenter l'histogramme de la grandeur « somme » de sortie, de le superposer à la gaussienne d'incertitude type calculée par LPU et enfin de superposer également la bande d'intervalle de confiance à 95% donnée par l'incertitude élargie.

Vous pouvez vous aider de la succession de commentaires qui ont jalonné ma proposition de solution Python. (fichier MonteCarlomuet.py dans votre espace de travail)

```
#import des modules nécessaires (graphiques et génération aléatoire)
import matplotlib.pyplot as plt
import numpy as np
import numpy.random as npr

#Création d'un échantillon issu d'une population de densité de probabilité uniforme sur un intervalle (rectangulaire)
#Calcul de l'incertitude-type de la distribution rectangulaire
#Calcul des moyenne et estimateur de l'écart-type de l'échantillon rectangulaire
#préparation d'un objet graphique histogramme rectangulaire à présenter en premier dans une fenêtre graphique à un rang et deux colonnes
#Création d'un échantillon issu d'une population de densité de probabilité gaussienne avec écart-type et moyenne fixés
#Calcul des moyenne et estimateur de l'écart-type de l'échantillon gaussien
#préparation d'un objet graphique histogramme gaussien
#Création d'un échantillon issu d'une population de densité de probabilité triangulaire
#Calcul de l'estimateur de l'incertitude-type de la distribution triangulaire par la LPU
#Calcul de la moyenne et de l'estimateur de l'écart-type de l'échantillon triangulaire
#préparation d'un objet graphique histogramme triangulaire
#réalisation de l'échantillon somme des trois variables
#Calcul de la moyenne et de l'estimateur de l'écart-type de la distribution de la variable "somme des trois variables"
#affichage console des moyennes et incertitudes-type pour LPU
#Calcul de l'incertitude-type sur la variable "somme" par la LPU
#affichage console de la moyenne et incertitude-type de la variable "somme" par la LPU
#affichage console de la moyenne et estimateur d'écart-type de la variable "somme" dans la distribution
#affichage console de l'intervalle de confiance à 95% basé sur la LPU
#préparation d'un objet graphique à présenter en second dans une fenêtre graphique à un rang et deux colonnes
#préparation d'un objet graphique histogramme de l'échantillon "somme"
#préparation des abscisses et ordonnées de la densité de probabilité gaussienne de moyenne et incertitude-type calculées par LPU
#préparation de l'objet graphique "densité de probabilité gaussienne" avec remplissage couleur de la surface sous la courbe
#préparation de l'objet graphique "bande rectangulaire de largeur l'intervalle de confiance à 95%"
#affichage de la fenêtre graphique
```

Vous pourrez alors comparer les résultats d'IC par LPU et MMC.
Vous pourrez choisir des distributions, différentes, des lois $y(x_1,x_2,x_3\ldots)$ différentes et même confirmer les valeurs numériques données pour les deux exemples de l'article en annexe.

En conclusion, on peut signaler que la création de distributions statistiques par génération de tirages **pseudo-aléatoires** est communément utilisée (pour simuler des bruits blancs, pour appliquer des méthodes de type Monte Carlo etc.) mais que l'obtention naturelle de la distribution gaussienne grâce à des phénomènes physiques dont l'origine est véritablement l'équiprobabilité exacte de deux micro-états aléatoires opposés est certainement à chercher … du côté de la physique quantique (états de polarisations, spins…)