Pythonによる時系列信号処理

1　Python による時系列信号処理

開発環境には Google Colaboratory，もしくはすでにアカウントを作っている大学の GPU サーバを利用してください。

1.1　バンドパスフィルタを使ってみよう

1) 0Hz から 30Hz まで (1Hz 毎) の正弦波の総和をとった波形を作りなさい。
2) 1 で作った波形に 10Hz 以上を遮断するローパスフィルタを適用して，に対して上記ローパスを適用して，フィルタ適用後の波形から高周波成分がなくなっていることを確認しなさい。ローパスフィルタには `scipy.signal.filtfilt()` と `signal.butter()` を用いること。
3) 透過率を表すグラフ (横軸周波数，縦軸透過率) を作りなさい。

1.2　移動平均と加算平均

1) sin 関数にノイズを加えた波形を生成しなさい。1 周期が 100 タイムステップとする。
2) **移動平均 (平滑化)**: 以下のように，信号 a(t)に対してある窓幅 ΔT で時間平均をとることで，ノイズの影響を小さくする方法を移動平均と呼ぶ。

$$\bar{a}(t) = \frac{1}{\Delta T}\int_{t-\Delta T/2}^{t+\Delta T/2} a(t)dt$$

   a) 1 で生成した信号の移動平均を取りなさい。
   b) 元の波形と，移動平均をとった後の波形について，時間軸の原点をどう対応づけると良いか考え，両者を一つのグラフに重ねて表示しなさい。
   c) 移動平均の窓幅をどの程度にすると，元の sin 関数に近い形状を復元できるか検討しなさい。
3) **加算平均**: (1) の第 i 周期の信号 $a_i(t)$, $(i = 1 \cdots N)$ とする。周期的な信号がノイズに埋もれている場合，以下のように加算平均をとることで原波形に近い波形を取り出すことができる。

$$\bar{a}(t) = \sum_{i=1}^{N} a_i(t)$$

a) (1) で生成した n 周期分の信号を，1 周期毎のデータに各行に格納した 2 次元配列 (ndarray) にしなさい。列数が，1 周期のタイムステップ (100) になる。

b) 加算平均をとり，nを大きくすることで，元の sin 関数に近づくことを確認しなさい。効果が分かりにくい時には，ノイズを大きくしてみなさい。

4) **規格化 + 可算平均**:

a) sin 関数にノイズを加えた波形を生成しなさい。ただし，1 の場合と異なり，1 周期毎に，ランダムに 10% 程度周期が異なる信号とする。

b) 前問で作った波形に対して以下の処理をして，ノイズの影響を減らし，(できるだけ) 元の波形を取り出しなさい

i. 移動平均をとる

ii. 極大値か極小値に注目し，1 周期毎の時間が同じ長さになるようにリサンプリングする。(1 周期毎に元の波形の時間ステップは異なるはずだが，全て 100 ステップになるようにリサンプリングをする)

iii. 加算平均を行う