TP3 - INF2163

Génération de signaux, échantillonnage

1 Le signal « impulsion unitaire » est défini par :

$$\delta(n) = \begin{cases} 1, & \text{for } n = 0 \\ 0, & \text{for } n \neq 0 \end{cases}$$

1.1 Écrire une fonction *GenSig* qui permet de générer une séquence de ce signal et prenant en paramètre, la longueur de la séquence *N* et l'offset (entier *k*) spécifiant la localisation dans le temps de l'impulsion unité.

1.2 Tracer une séquence « impulsion unitaire » de 20 échantillons, pour un offset de 5 échantillons. On utilisera la fonction *stem*.

2 Le signal « saut unité », ou escalier, est défini par :

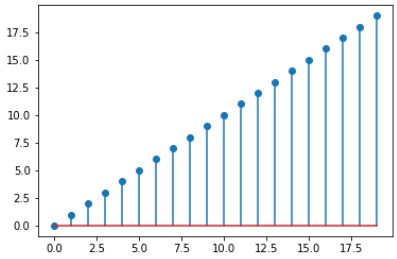
$$u(n) = \begin{cases} 1, & \text{for } n \geq 0 \\ 0, & \text{for } n < 0 \end{cases}$$

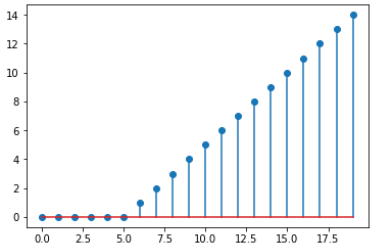
Modifier la fonction *GenSig* en introduisant un paramètre supplémentaire (*type*) permettant de spécifier la séquence à générer ('I' pour impulsion, ''S' pour saut unitaire), et en développant le code spécifique pour le signal « saut unité ». Tracer une séquence « saut unité » de 20 échantillons, pour un offset de 5.

3 Le signal « rampe unité » est défini par :

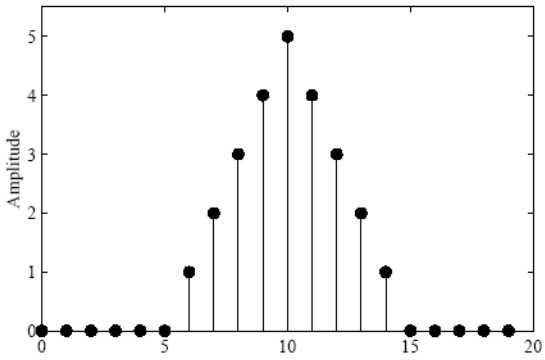
$$u_r(n) = \begin{cases} n, & \text{for } n \geq 0 \\ 0, & \text{for } n < 0 \end{cases}$$

Cette fonction a pour pente 1 et démarre à 0. Modifier la fonction *GenSig* en développant le code spécifique pour le signal « rampe unité ». Vous pourrez prendre en compte un décalage temporel de k comme sur la figure ci-dessous.
Tracer une séquence « saut unité » de 20 échantillons, pour un offset de 5. Le paramètre type prendra 'R' pour valeur.





4 Générer le signal ci-dessous par combinaison linéaire des trois signaux précédents.



5 Synthèse d'un signal sinusoïdal

5.1 Ecrire un programme Python *genSin (N,f,fs)* qui permet de synthétiser un signal de fréquence *f*, de fréquence d'échantillonnage *fs* sur une plage temporelle de *N* échantillons. Vous pourrez par exemple créer un signal sinusoïdal de 50 Hz et prendre un vecteur de 512 échantillons.

Visualiser ce signal à l'aide de la fonction *plot* de *matplotlib* pour différentes fréquences *f* et *fs*.

5.2 Remplacez la fonction sinusoïdale par une autre fonction périodique (fonction *carré* par exemple).

5.3. Déterminez *N* à partir du nombre *k* de périodes à visualiser.

6 Signaux exponentiels complexes

Le signal exponentiel est de la forme :

$$x(n) = a^n, \qquad \text{for all } n$$

Lorsque *a* est réel, *x*(*n*) est réel. Lorsque *a* est complexe, on utilise également la représentation suivante :

$$x(n) = r^n e^{j\theta n} = r^n(\cos\theta n + j\sin\theta n)$$

6.1 Ecrire une fonction Python *expsig* permettant de générer des signaux exponentiels, en prenant en paramètre la longueur de la séquence à générer et le paramètre *a*.

6.2 Tracer à l'aide de la fonction *stem* de matplotlib les séquences de 20 échantillons correspondantes aux paramètres suivants :

$$a = -0.95 \quad a = 0.95e^{j\pi/10}$$

7. Question subsidiaire - Interpolation de matrices : application au « morphing » d'images

Choisissez deux images (portraits en noir et blanc) de votre choix au format jpeg. Écrire une fonction qui charge ces deux fichiers dans deux matrices X et Y respectivement, puis construit une séquence de matrices d'interpolation Z= a.X+(1-a)Y, a variant de 0 à 1 par pas de 0.01. On utilisera les fonctions : *imread, getframe, movie*.