Capacité numérique 2

Oscillateur à relaxation

I- Représentation temporelle des signaux

A - Position du problème

On souhaite mettre en oeuvre une méthode numérique permettant de représenter, pour les étudier simplement, les signaux produits par un oscillateur à relaxation.
Pour cela on va utiliser la méthode d'étude des oscillateurs vue en cours : on étudie le système bloc par bloc.
Considérons l'oscillateur dont le schéma est donné ci-dessous.

Oscil_relax.png

ALI 2 est un intégrateur inverseur dont la sortie $s(t)$ est reliée à l'entrée $c(t)$ par :

$$s(t) = -\frac{1}{\tau}\int c(t)dt + \frac{v}{\tau}$$

Avec $\tau = R.C$ et v est une tension constante dont on souhaite par la suite comprendre l'effet.

ALI 1 est un comparateur à hystérésis dont la sortie $c(t)$ ne peut prendre que deux valeurs (fonctionnement saturé de ALI 1) :

- $c(t) = +V_{sat}$ saturation haute ;
- $c(t) = -V_{sat}$ saturation basse. Grace au cours sur les oscillateurs on sait que la valeur prise par $c(t)$ est fonction de la valeur de $s(t)$ et des résistance R_1 et R_2.
  u est également une tension constante dont on souhaite encore comprendre l'effet.

Dans un premier temps on comprend que pour résoudre l'équation différentielle donnée par ALI 2 on va utiliser un méthode d'Euler explicite.
ALI 1 imposant alors une limite de l'amplitude de $c(t)$ et des conditions de basculement :

- si $\frac{R_2.s + R_1.c}{R_1 + R_2} > u$ alors $c = +V_{sat}$;
- si $\frac{R_2.s + R_1.c}{R_1 + R_2} < u$ alors $c = -V_{sat}$

Le dernier point de vigilence :
Les tensions s et c sont périodiques, et nous allons mettre en oeuvre un méthode numérique de résolution. Il faut alors particulièrement attention au nombre de point utilisé par période pour représenter les signaux. On utilise le théorème de Shannon.
Notons T_{max} la date maximale à laquelle on représente les tensions. On note T_0 la période des oscillations.
Ces deux dates ne sont pas indépendantes puisqu'elles sont liées entres elles par le nombre de point N et la fréquence d'échantillonnage f_{ech}.

B - Mise en oeuvre

1. Importer les bibliothèques nécessaires à la résolution et à la représentation du problème.

```
import numpy as np
import matplotlib.pyplot as plt
```

2. Définir l'ensemble des paramètres caractéristiques de l'intégrateur inverseur : $R = 1000\Omega$, $F = 1\mu F$ et $v = 0V$.

```
R = 1000.0 #résistance d'entrée en ohm.
C = 1.0e-6 #capacité de rétroaction en farad.

tau = R*C #constante de temps d'intégration.

v = 0.0 #tension de référence constante à l'entrée + de l'intégrateur.
```

3. Définir l'ensemble des paramètres caractéristiques du comparateur à hystérésis : $R_1 = 5000\Omega$, $R_2 = 10000\Omega$, $V_{sat} = 15V$ et $u = 0$.

```
Vsat = 15.0 #tension de saturation en volt.

R1 = 5000.0 #résistance d'entrée en ohm.
R2 = 10000.0 #résistance de rétroaction en ohm.

u = 0.0 #tension de référence à l'entrée - du comparateur.
```

4. Définir la période des oscillations en fonction des paramètres de l'intégrateur inverseur et du comparateur à hystérésis. On souhaite que la fréquence d'échantillonnage soit 1000 fois plus importante que la fréquence d'oscillation. Que la durée maximale d'intégration soit 10 fois plus importante que la période.
   Exprimer alors le nombre de point d'intégration.

```
T0 = 4*tau*R1/R2 #période des oscillations en seconde.

f0 = 1.0/T0 #fréquence des oscillations.

print(f0)

fech = 1000*f0 #critère de Shannon. Modifier ce paramètre modifie la qualité de représentation
               # des commutatrice de la sortie du comparateur et donc la qualité de représentation
               #du cycle d'hystérésis.
```

```
Tmax = 10.0*T0 #durée de simulation

N = int(Tmax*fech) #nombre de points de calcul.
```

```
500.0
```

5. Définir une fonction `integrateur` dont les arguments sont υ, τ et les tensions d'entrée/sortie et traduisant l'équation différentielle de ALI 2 et permettant d'appliquer la méthode d'Euler explicite.

```
def integrateur(x,y,ref_int,t_carac):
    return(ref_int/t_carac - y/t_carac)
```

6. Définir une fonction `bascule` prenant en argument les résistances R_1, R_2, la tension de saturation V_{sat} ainsi que les tensions s, c et u et qui permet de définir les conditions de basculement de ALI 1.

```
def bascule(saturation,s,c,ref_comp,res1,res2):
    if (res2*s+res1*c)/(res1+res2)>ref_comp:
        c = saturation
    if (res2*s + res1*c)/(res1+res2)<ref_comp:
        c = -saturation
    return(c)
```

7. Définir une fonction `Comparateur` dont les arguments permettent d'utiliser les fonctions `integrateur` et `bascule` et mettant en oeuvre la méthode d'Euler explicite permettant de résoudre le problème posé. Cette fonction doit retourner $s(t)$, $c(t)$ et le temps.

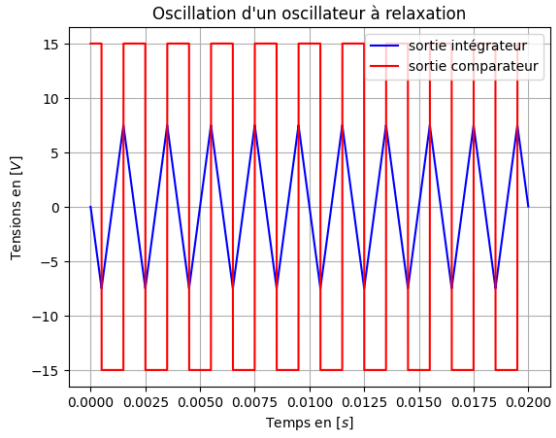
```
def Comparateur(ref_int,ref_comp,saturation,t_carac,res1,res2,Tmax,points,init_s):
    pas = Tmax/points
    S = init_s
    C = Vsat
    sortie = [S]
    comp = [C]
    t = [0.0]
    date = 0.0
    i = 0.0
    while i < N-1:
        S = S + pas*integrateur(S,C,ref_int,t_carac)
        C = bascule(saturation,S,C,ref_comp,res1,res2)
        date = date+pas
        sortie.append(S)
        comp.append(C)
        t.append(date)
        i+=1
    return(t,sortie,comp)
```

8. Ecrire une suite d'instructions permettant de réaliser une représentation graphique des signaux temporels $s(t)$ et $c(t)$. Vous prendrez un soing tout particulier à la mise en forme du graphique.

```
u = 0.0
v = 0.0

t,s,c = Comparateur(v,u,Vsat,tau,R1,R2,Tmax,N,0.0)
```

```
plt.clf()
plt.figure()
plt.plot(t,s,'b-',label='sortie intégrateur')
plt.plot(t,c,'r-',label='sortie comparateur')
plt.title('Oscillation d\'un oscillateur à relaxation')
plt.xlabel('Temps en $[s]$')
plt.ylabel('Tensions en $[V]$')
plt.legend(loc=1)
plt.grid(True)
plt.show()
plt.close()
```

```
<Figure size 640x480 with 0 Axes>
```

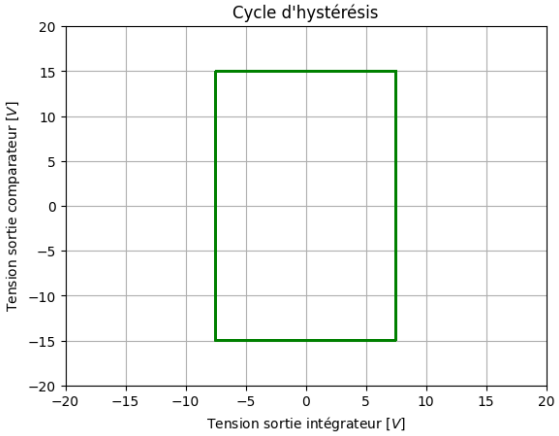


9. Ecrire une suite d'instruction permettant de réaliser une représentation graphique du cycle d'hystérésis.

```
plt.clf()
plt.figure()
```

```
plt.plot(s,c,'g-')
plt.grid(True)
plt.ylim(-20,20)
plt.xlim(-20,20)
plt.xlabel('Tension sortie intégrateur $[V]$')
plt.ylabel('Tension sortie comparateur $[V]$')
plt.title('Cycle d\'hystérésis')
plt.show()
plt.close()
```

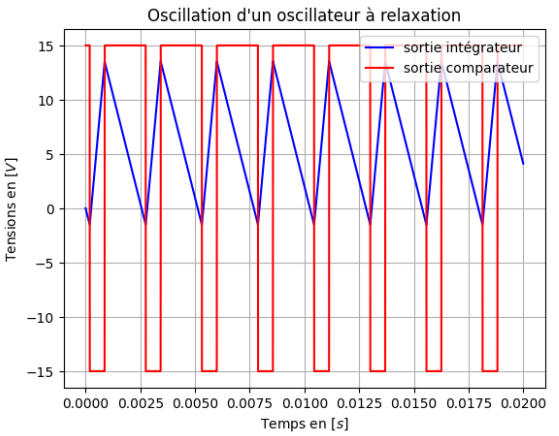
```
<Figure size 640x480 with 0 Axes>
```



10. Conduire une étude simple des effets des tensions u et v sur la forme des signaux. Quels sont leurs effets respectifs ?

```
u = 4.0
v = 7.0

t,s,c = Comparateur(v,u,Vsat,tau,R1,R2,Tmax,N,0.0)

plt.clf()
plt.figure()
plt.plot(t,s,'b-',label='sortie intégrateur')
plt.plot(t,c,'r-',label='sortie comparateur')
plt.title('Oscillation d\'un oscillateur à relaxation')
plt.xlabel('Temps en $[s]$')
plt.ylabel('Tensions en $[V]$')
plt.legend(loc=1)
plt.grid(True)
plt.show()
plt.close()
```

```
<Figure size 640x480 with 0 Axes>
```



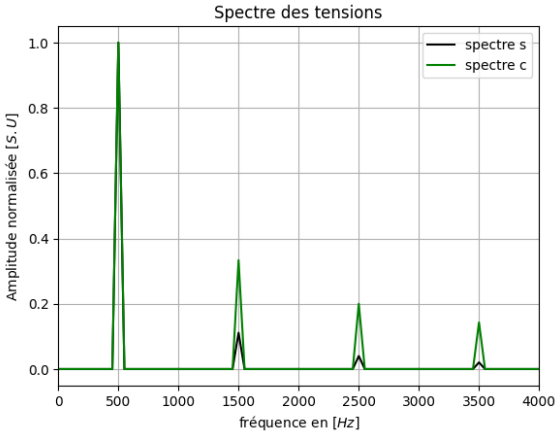
II - Analyse spectrale

On peut conduire une analyse spectrale des signaux précédent et comparer les spectres des tensions $c(t)$ et $s(t)$.

11. Commenter les lignes du code suivant.

```
from scipy.fftpack import fft, fftfreq

u = 0.0
v = 0.0

t,s,c = Comparateur(v,u,Vsat,tau,R1,R2,Tmax,N,0.0)

Tech = 1.0/fech

FenAcq = np.array(s).size

s_FFT = abs(fft(s))
c_FFT = abs(fft(c))
```

```
s_freq = fftfreq(FenAcq,Tech)
c_freq = fftfreq(FenAcq,Tech)

s_FFT = s_FFT[0:len(s_FFT)//2]
s_freq = s_freq[0:len(s_freq)//2]
c_FFT = c_FFT[0:len(c_FFT)//2]
c_freq = c_freq[0:len(c_freq)//2]

plt.figure()
plt.plot(s_freq,s_FFT/max(s_FFT),'k-',label='spectre s')
plt.plot(c_freq,c_FFT/max(c_FFT),'g-',label='spectre c')
plt.xlim(0,4000)
plt.grid(True)
plt.legend(loc='best')
plt.xlabel('fréquence en $[Hz]$')
plt.ylabel('Amplitude normalisée $[S.U]$')
plt.title('Spectre des tensions')
plt.show()
plt.close()
```




```
Commencez à coder ou à générer avec l'IA.
```