TP20 – Filtrage numérique

Objectifs

→ **Simuler, à l'aide d'un langage de programmation, l'action d'un filtre d'ordre 1 ou 2 sur un signal périodique dont le spectre est fourni. Mettre en évidence l'influence des caractéristiques du filtre sur l'opération de filtrage.**

Convertisseur alternatif/continu

On souhaite obtenir une tension continue à partir de la tension du secteur, sinusoïdale à 50 Hz. On considère pour cela un convertisseur de tension alternatif/continu, constitué de trois parties :

- un transformateur qui abaisse la tension du secteur et fournit une tension $u(t)$ sinusoïdale de même fréquence :

$$u(t) = U_0 \sin(2\pi f_0 t),\ \text{avec } f_0 = 50\,\text{Hz et } U_0 = 10\,\text{V};$$

- un circuit redresseur, appelé pont de Graëtz, qui permet d'obtenir le signal redressé $e(t)$ (Fig. 1) :
- un filtre qui agit sur le signal $e(t)$ de manière à obtenir, en sortie, une tension continue.

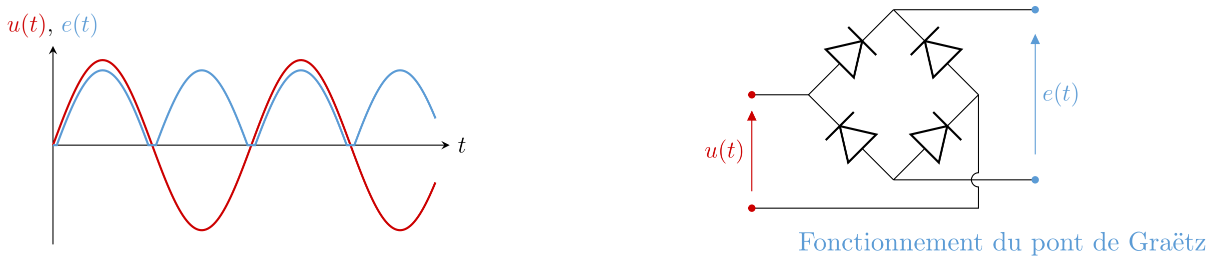


Figure 1 – Représentation temporelle des signaux $u(t)$ et $e(t)$, à l'entrée et à la sortie du pont de Graëtz, représenté à droite.

Synthèse spectrale approchée du signal $e(t)$

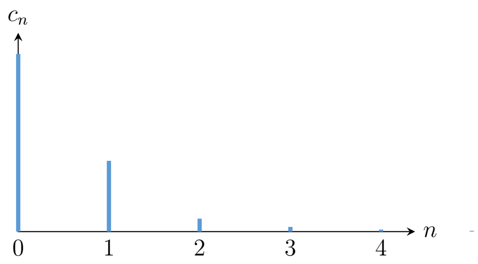
On admet que le signal $e(t)$ à la sortie du pont de Graëtz est de la forme :

$$e(t) = \begin{cases} 0 & \text{si } |u(t)| < 2V_s; \\ |u(t)| - 2V_s & \text{sinon,} \end{cases}$$

où $V_s = 0{,}6\,\text{V}$ désigne la tension de seuil des diodes.

Comme le montre la figure 1, le signal $e(t)$ est périodique. Il peut donc être décomposé en série de Fourier :

$$e(t) = \frac{c_0}{2} + \sum_{n=1}^{+\infty} c_n \cos(2\pi n f_e t + \varphi_n).$$



n	c_n (V)	φ_n (rad)
0	10,424 2	
1	4,152 8	π
2	0,758 8	π
3	0,275 9	π
4	0,117 2	π

FIGURE 2 – Spectre du signal $e(t)$. Les amplitudes des différentes harmoniques ont été obtenues numériquement à l'aide d'une FFT du signal $e(t)$.

L'allure du spectre de $e(t)$ est donnée ci-dessus (Fig. 2). On constate que les valeurs des coefficients c_n décroissent très rapidement avec n : on se restreindra aux composantes de rang $n \leqslant 3$. Dans toute la suite, le signal $e(t)$ sera donc approché par

$$e(t) \approx \frac{c_0}{2} + \sum_{n=1}^{3} c_n \cos(2\pi n f_e t + \varphi_n). \tag{1}$$

Filtrage du signal $e(t)$

L'objectif est de faire apparaitre les contraintes sur le dimensionnement du filtre situé après le pont de Graëtz.

Questions préliminaires

✍ 1. En exploitant la figure 1, donner au moins deux arguments qui attestent de la non-linéarité du pont de Graëtz. Exprimer la fréquence f_e du signal $e(t)$ en fonction de f_0.

✍ 2. Compléter le code `tp20-filtrage_numerique.py` en écrivant la fonction `e3(t)` qui évalue le signal $e(t)$ à l'instant t à l'aide des premiers termes de sa série de Fourier (Éq. 1). Vérifier graphiquement la pertinence de l'approximation réalisée.

✍ 3. Quel type de filtre doit-on utiliser pour obtenir une tension continue en sortie du convertisseur ? Faire un schéma du montage à réaliser pour obtenir un tel filtre avec une résistance et un condensateur.

✍ 4. Établir l'expression de sa fonction de transfert $\underline{H}_1(j\omega)$, puis donner les expressions du gain linéaire $G_1(\omega)$ et du déphasage introduit par le filtre $\phi_1(\omega)$ en fonction de la pulsation de coupure ω_c.

✍ 5. Indiquer la valeur de la tension $s(t)$ à la sortie du filtre dans le cas d'un filtre idéal, dont le gain dans la bande coupée est nul.

Filtre passe-bas d'ordre un

Pour extraire la composante continue du signal $e(t)$, on utilise tout d'abord un filtre passe-bas d'ordre un dont la fonction de transfert s'écrit :

$$\underline{H}_1(jf) = \frac{1}{1 + j\frac{f}{f_c}},$$

où f_c désigne la fréquence de coupure du filtre. Le gain linéaire et le déphasage introduit par le filtre s'expriment donc

$$G_1(f) = \frac{1}{\sqrt{1 + \left(\frac{f}{f_c}\right)^2}} \quad \text{et} \quad \phi_1(f) = -\arctan\left(\frac{f}{f_c}\right).$$

La sortie $s(t)$ se calcule en appliquant le principe de superposition :

$$s(t) = \frac{c'_0}{2} + \sum_{n=1}^{+\infty} c'_n \cos(2\pi n f_e t + \varphi'_n) \quad \text{avec} \quad \begin{cases} c'_n = c_n \times G_1(nf_e) \\ \varphi'_n = \varphi_n + \phi_1(nf_e) \end{cases}$$

À nouveau, on se restreindra aux quatre premières composantes.

6. Quelle propriété du filtre permet d'appliquer le principe de superposition ?
7. Écrire[1] les fonctions `G1(f, fc)` et `phi1(f, fc)` qui évaluent le gain et le déphasage introduit par le filtre de fonction de transfert $\underline{H}_1$ à la fréquence f.
8. Écrire la fonction `s1(t)` qui évalue le signal $s(t)$ à l'instant t. Vérifier graphiquement, pour quelques valeurs pertinentes de f_c, que le filtre a bien le comportement attendu et commenter la valeur de la tension obtenue quand le filtre remplit bien son rôle.

APPEL PROF 1

Effet de la fréquence de coupure

Pour caractériser l'ondulation résiduelle du signal $s(t)$, on introduit le taux d'ondulation Ω : pour un signal $s(t)$ de valeur moyenne S_{moy} et de valeur efficace S_{eff}, le taux d'ondulation est défini par :

$$\Omega = \frac{S_{\text{eff}}}{S_{\text{moy}}} - 1.$$

En particulier, $\Omega = 0$ pour un signal continu et $\Omega \to +\infty$ pour un signal purement sinusoïdal.

9. Justifier que, si l'on dispose de suffisamment d'échantillons du signal $s(t)$ sur une période $T = 1/f_e$, on peut approcher sa valeur moyenne par la moyenne arithmétique des échantillons de $s(t)$.
10. Écrire la fonction `Omega(fc)` qui évalue le taux d'ondulation du signal $s_1(t)$ en fonction de la fréquence de coupure du filtre. Dans une nouvelle fenêtre, représenter graphiquement $\Omega(f_c)$. Commenter l'allure de la courbe obtenue.
11. En pratique, pourquoi ne peut-on pas choisir une fréquence de coupure f_c arbitrairement basse ?

1. On pourra aussi utiliser les fonction `np.abs` et `np.angle` pour obtenir le module et l'argument de la fonction de transfert complexe. En Python, le nombre complexe j s'écrit `1j`.

Effet de l'ordre du filtre

En prenant les précautions nécessaires,[2] on peut obtenir un filtre d'ordre p en mettant en cascade p filtres d'ordre un. La fonction de transfert $\underline{H}_p$ du filtre obtenu est alors simplement le produit des fonctions de transfert $\underline{H}_1$ de chacun des filtres, soit :

$$\underline{H}_p(jf) = \underline{H}_1^n(jf) = \left(\frac{1}{1 + j\frac{f}{f_c}}\right)^n.$$

12. En s'inspirant du travail réalisé précédemment, écrire la fonction `sp(t, fc, p)` qui évalue le signal $s(t)$ à la sortie du filtre de fonction de transfert $\underline{H}_p$.

13. À l'aide d'une représentation graphique pertinente, mettre en évidence l'effet de l'ordre du filtre sur l'ondulation résiduelle du signal de sortie.

14. Sur un même graphe, représenter les diagrammes de Bode en amplitude associés à $\underline{H}_p$ pour quelques valeurs de p. Commenter la valeur de la pente de l'asymptote en haute fréquence et justifier l'intérêt d'utiliser un filtre d'ordre élevé.

Lien avec la représentation temporelle

15. À partir de l'expression de la fonction de transfert $\underline{H}_1$, établir l'équation différentielle reliant $s(t)$ et $e(t)$ dans le cas du filtre passe-bas d'ordre un.

16. Écrire la fonction `s1odeint(t)` et/ou `s1euler(t)` qui évalue la sortie $s(t)$ à l'instant t en intégrant numériquement l'équation différentielle. Comparer la solution obtenue à `s1(t)`.

2. En pratique, ces précautions permettent de s'assurer que l'ajout d'un filtre ne modifie pas le comportement du précédent.

Document 1 – Quelques fonctions utiles

`np.mean(tab)` : moyenne des valeurs contenues dans la liste ou le tableau `tab`.

`np.log(x)` : logarithme népérien de x.

`np.log10(x)` : logarithme en base 10 de x.

`np.abs(z)`, `np.angle(z)` : module et argument d'un nombre complexe. Le complexe $5+j$ s'écrit sous la forme `5 + 1j`.

`plt.semilogx`, `plt.semilogy`, `plt.loglog` : remplacent la commande `plt.plot` pour tracer un graphe en échelle logarithmique (abscisse seulement, ordonnée seulement ou les deux).

`scipy.integrate.odeint(F, V0, t)` : intègre un système d'équations différentielles du premier ordre de la forme :

$$\frac{\mathrm{d}V}{\mathrm{d}t} = F(V,t).$$

Paramètres :

`F` : fonction qui calcule la dérivée de V en t ;

`V0` : conditions initiales sur V ;

`t` : liste des instants auxquels calculer V.

Renvoie :

`V` : tableau contenant `len(t)` vecteurs V, calculés aux instants de `t`.

Annexes

Annexe 1 – `tp20-filtrage_numerique.py`

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

# CONSTANTES
U0 = 10   # amplitude du signal u(t) en volts
f0 = 50   # fréquence du signal u(t) en hertz
Vs = 0.6  # tension de seuil des diodes en volts

cn   = [10.4242, 4.1528, 0.7588, 0.2759] # coefficients cn en volts
phin = [      0,   np.pi,  np.pi,  np.pi] # coefficients phin en radians

def u(t):
    """
    params: temps t en secondes
    returns: tension u(t) à la sortie du transformateur en volts
    """
    return U0 * np.sin(2*np.pi*f0 * t)

def e(t):
    """
    params: temps t en secondes
    returns: tension e(t) à la sortie du pont de Graëtz en volts
    """
    output = np.abs(u(t)) - 2*Vs
    if hasattr(t, "__len__"):
        for i in range(len(output)):
            output[i] = max(output[i],0)
    else:
        output = max(output,0)
    return output

###########################
# REPRÉSENTATIONS GRAPHIQUES
###########################
plt.figure(1)
plt.clf()
t = np.linspace(0,2/f0, 1000) # instants t en secondes pour les graphes
plt.title("Évolution temporelle des différents signaux")
plt.plot(t, u(t), label="$u(t)$")
plt.plot(t, e(t), label="$e(t)$")
plt.xlabel("Temps (s)")
plt.ylabel("Signal (V)")
plt.legend()
plt.grid(which="both")
plt.show()
```