Introduction au signal et bruit
Cours

Gabriel Dauphin

September 4, 2025

Contents

Chapter 1

Relations entrées-sorties sans effet mémoire

1.1 Grandeurs étudiées

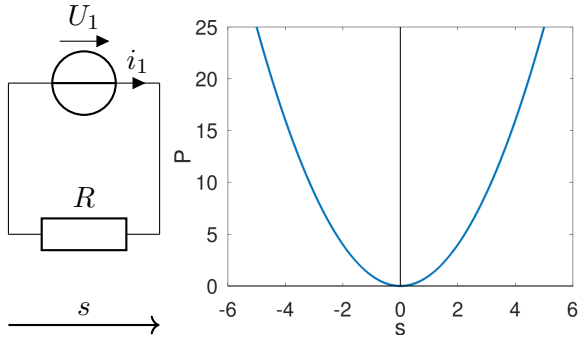

Figure 1.1: À gauche : Circuit générant une tension continue $s = U_1$. À droite : relation entre la puissance d'un signal et la valeur du signal.

Cours signal et bruit :

Dans ce chapitre, on ne s'intéresse pas à la dépendance vis-à-vis du temps. La grandeur étudiée pourrait être l'intensité ou la tension, mais ici on va considérer que c'est la tension et en traitement du signal, on n'indique pas l'unité du signal. On le note s pour signal. Il est à valeurs réelles dans le cadre de ce cours (sauf pour des astuces de calcul).

D'un point de vue électronique :

La figure 1.1 montre un circuit générant une tension $s = U_1$, celle-ci ne dépend pas de R. L'implémentation est dans A.6.

En électronique, la puissance consommée ou fournie est le produit tension aux bornes d'un composant par intensité traversant le composant, elle est notée $\mathtt{P}_s$, son unité est le W. Pour ce montage, $\mathtt{P}_s = s \times \frac{s}{\mathtt{R}} = \frac{s^2}{\mathtt{R}}$, ($s$ est la tension et $\frac{s}{\mathtt{R}}$ est l'intensité. Et en général la puissance instantanée dépend de $u(t)i(t)$ et non uniquement d'un seul signal.

Plus généralement en physique, la notion d'énergie est différente, ce n'est pas une notion qui se déduit de la formule du signal. Le même signal peut représenter suivant le contexte à une énergie finie ou infinie et du coup être décrit plutôt par la notion d'énergie ou de puissance, celle-ci étant aussi la variation de l'énergie au cours du temps. Ainsi quand on considère un système fermé (i.e. sans interaction avec l'extérieur), l'énergie est conservé au cours du temps. Un tel phénomène pourrait être décrit par un signal constant et du point de vue du traitement du signal, on dit qu'on a une énergie infinie, mais pas en physique. Quand on subdivise ce système fermé en deux systèmes `A` et `B` qui interagissent et qui donc ont des transfert d'énergie, lorsque `A` transfert une quantité pendant un laps de temps T vers `B`, on a bien un transfert d'énergie au cours du temps T qui est le résultat d'une intégration au cours du temps d'une puissance instantanée transférée. Et si on imagine qu'en même temps `B` effectue vers `A` le même transfert, on pourrait avoir une puissance instantanée constante au cours du temps qui cumulée sur un temps infini corresponde à une énergie infinie. On aurait ainsi un transfert d'énergie infinie entre deux systèmes, chacun ayant une énergie finie et même constante. Un circuit RC (i.e. formé d'une résistance et d'un condensateur) a un comportement voisin : la source de tension parfois alimente la résistance aidée du condensateur qui se décharge, et parfois la source de tension

alimente à la fois la résistance et le condensateur qui se charge. L'énergie dans le condensateur est finie et vaut $\frac{1}{2}\mathtt{C}u^2(t)$ alors qu'en traitement du signal $u(t)$ en tant que signal périodique a une énergie infinie.

Cours signal et bruit :

En traitement du signal, la puissance est le carré du signal :

$$P_s = s^2 \tag{1.1}$$

Et on observe qu'il y a ici un lien avec $\mathtt{P}_s = \frac{P_s}{\mathtt{R}}$. Dans la suite du cours on désignera ces deux notions par puissance instantanée, car dans la suite on considère des signaux dépendant du temps avec une puissance commune à l'ensemble du signal.

On appelle ici pseudo-code la liste des tâches à accomplir pour calculer les vecteurs utilisés pour représenter une figure.

1.2 Première utilisation de Python

Python :

Pour démarrer l'utilisation de Python, je propose de choisir deux répertoires, le premier que j'appelle `rep_prg` contient les programmes, notamment ceux disponibles sur mon site, et un autre répertoire que j'appelle `rep_tra` où vous mettez le travail effectué notamment les données et les figures.

Je propose ensuite d'effectuer les lignes suivantes qui utilisent un module `seb` que j'ai écrit pour ce cours. `plt` et `np` sont les modules `matplotlib` et `numpy` qui servent à tracer des graphes et à manipuler des vecteurs.

```
import sys
sys.path.append('rep_prg')
import os
os.chdir('rep_tra')
import seb
plt,np,sig=seb.debut()
```

On appelle vecteur un tableau de valeurs composé d'une ligne (on parle alors de vecteur ligne) ou d'une seule colonne (on parle de vecteur colonne). La fonction `linspace` de `numpy` permet de générer un ensemble de valeurs en indiquant la première, la dernière et le nombre de ces valeurs.

```
s=np.linspace(-5,5,100)
```

Cette instruction génère 100 valeurs entre -5 et 5. Cette notion est utile pour générer un graphique, elle permet d'obtenir le graphe à droite 1.1 représentant la puissance instantanée en fonction de la valeur du signal. L'implémentation de $P_s = s^2$ se fait avec

```
P=s**2
```

On obtient alors un vecteur ligne de même taille que `s` et contenant successivement toutes les valeurs de P_s pour chacune des valeurs du vecteur `s`. Pour faire le graphe, on commence par

```
fig,ax = plt.subplots()
```

Ensuite pour la courbe on rajoute [1] Le troisième argument permet de rajouter une légende.

```
ax.plot(s,P_s,label='puissance')
```

Sur le graphe on peut préciser ce que signifie l'axe des abscisses

```
ax.set_xlabel('x')
```

L'affichage de la légende est déclenchée par

[1] Et si on avait plusieurs courbes, il suffit de rajouter une ligne par courbe.

```
ax.legend()
```

La gestion de la taille de la figure est faite avec

```
plt.tight_layout()
```

Je propose de sauvegarde la figure

```
fig.savefig('nom_figure.png')
```

La figure apparaît lorsqu'on exécute cette commane

```
fig.show()
```

1.3 Notion de filtres

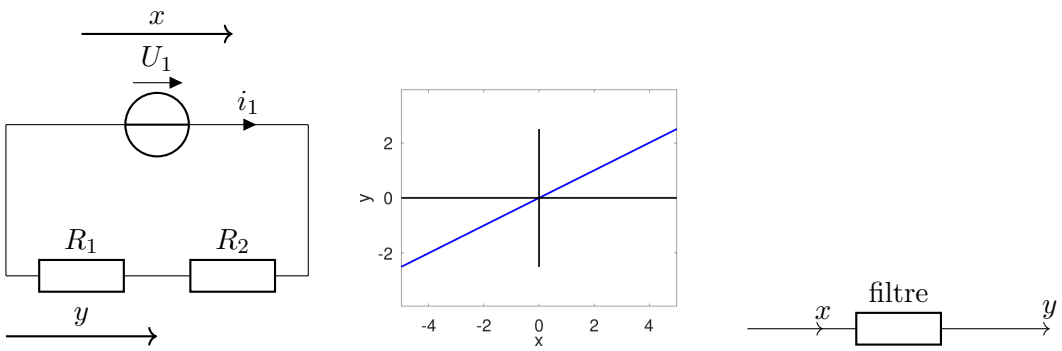

Figure 1.2: À gauche : diviseur de tension. Au centre : graphe de y en fonction de x. À droite : notation pour le filtre transformant x en y.

Cours signal et bruit :

Du point de vue de traitement du signal, on cherche à décrire la relation mais aussi la façon dont on peut utiliser cette relation.

- Une entrée est ce que l'on peut assigner.
- Une sortie est ce que l'on peut mesurer.

La sortie dépend de l'entrée et on utilise dans ce cours la notation $\mathcal{H}$ pour indiquer cettet relation

$$y = \mathcal{H}(x) \tag{1.2}$$

Ce filtre est représenté sur la figure 1.2.

Parmi les filtres, on distingue, les relations linéaires. Dans ce chapitre on dit qu'un filtre est linéaire, s'il existe un coefficient de proportionnalité appelé gain statique G

$$y = Gx \tag{1.3}$$

Lorsque $G > 1$, on dit qu'il y a une amplification. Si ce coefficient vaut $G < 1$ alors on dit qu'il y a une atténuation.

D'un point de vue électronique :

C'est un point de vue différent, les outils semblent être d'abord définis pour simuler un signal au sein d'un montage.

La figure 1.2 présente à gauche un diviseurs de tension et à droite la relation linéaire.

$$y = x\frac{R_1}{R_1 + R_2} \tag{1.4}$$

Une particularité technologique est que y est nécessairement du même signe que x et est inférieur à x.

Le fait que pour chaque valeur de l'entrée, il y ait une unique valeur possible en sortie, c'est justement un effet sans mémoire.

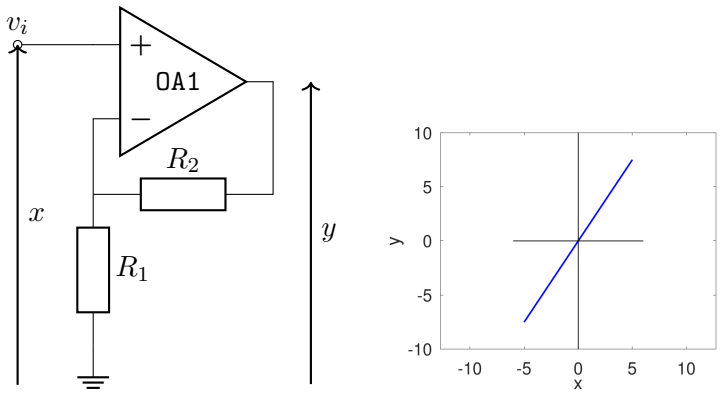

Figure 1.3: À gauche : utilisation d'un amplificateur en boucle fermé. À droite : graphe de y en fonction de x.

D'un point de vue électronique :

La figure 1.3 présente un amplificateur non-inversé en boucle fermé et à droite la relation linéaire correspondante.

$$y = x\left(\frac{R_1 + R_2}{R_1}\right) \tag{1.5}$$

Cette équation vient du théorème de Millman (voir section B.3 p. 93).

Une particularité technologique est que y est nécessairement du même signe que x et est supérieur à x.

D'un point de vue électronique :

On remarque que **ici** quand le gain statique est supérieur à 1, le montage ne peut être obtenu avec des composants passifs et une source extérieure d'alimentation est nécessaire. En réalité ce n'est pas toujours vrai quand on considère un signal dépendant du temps (par exemple un transformateur ou une alimentation à découpage). Les considérations énergiques du traitement du signal ne peuvent avoir un vrai sens physique dans la mesure où en traitement du signal un filtre ne concerne pas à la fois $i(t)$ et $u(t)$.

Cours signal et bruit :

Le rapport entre la puissance en sortie d'une relation entrée-sortie par rapport à la puissance en entrée est

$$\frac{P_y}{P_x} = G^2 \tag{1.6}$$

En terme de visualisation :

Le graphe d'une relation linéaire est une droite passant par l'origine $(0, 0)$. Un graphe ayant les mêmes graduations pour les deux axes est dit orthonormé. Et dans ce cas, il s'agit d'une amplification si l'angle avec l'horizontal est plus grand que $\frac{\pi}{4}$, et il s'agit d'une atténuation si l'angle avec l'horizontal est plus petit que $\frac{\pi}{4}$,

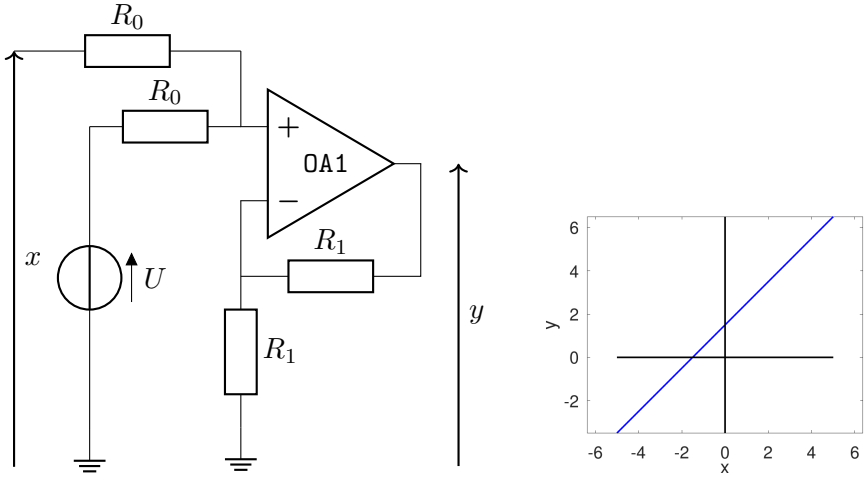

Figure 1.4: À gauche : utilisation d'un amplificateur en boucle fermé. À droite : graphe de y en fonction de x.

Cours signal et bruit :

La figure 1.4 montre à gauche un additionneur de tension et à droite la relation induite qui est l'ajout d'une constante.

$$y = x + U \tag{1.7}$$

Plus généralement lorsqu'une relation s'écrit $y = ax + b$, on dit qu'il s'agit d'une relation affine.

En terme de visualisation :

Le graphe d'une relation affine est une droite.

D'un point de vue électronique :

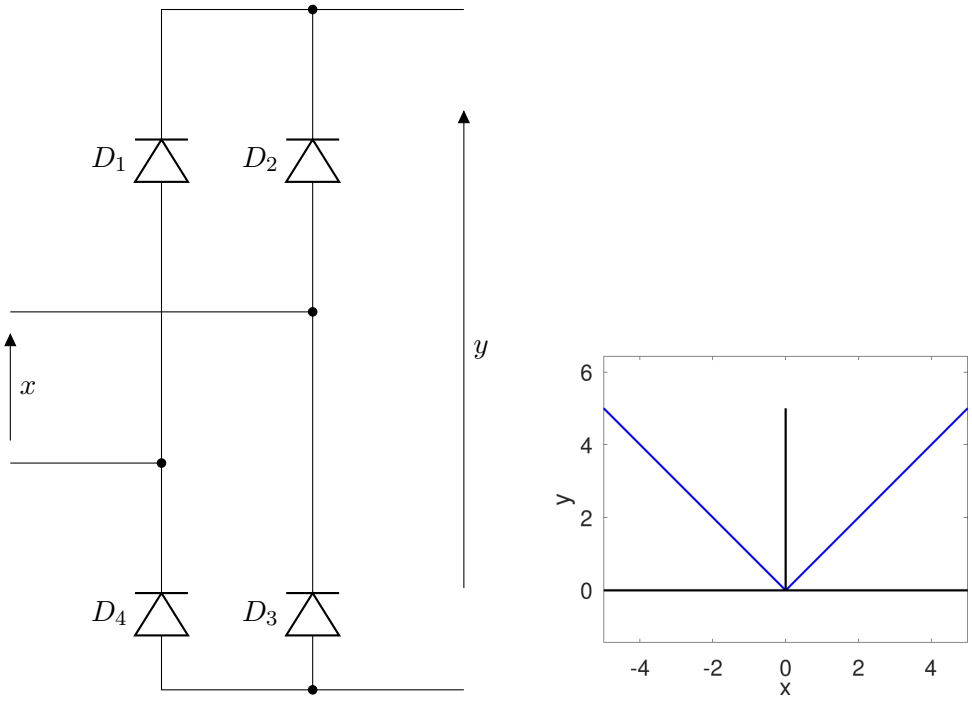

Figure 1.5

La figure 1.5 montre à gauche un pont de diode et à droite la relation entrée-sortie.

D'un point de vue mathématique :

Valeur absolue

$$|t| = \begin{cases} t & \text{si} \quad t \geq 0 \\ -t & \text{si} \quad t < 0 \end{cases}$$

$$y \quad = \quad \begin{cases} x \text{ si } x \geq 0 \\ -x \text{ si } x \leq 0 \end{cases} \quad = \quad |x| \tag{1.8}$$

Deux autres relations non-linéaires sont données par les opérateurs min et max.

$$\min(a, x) = \begin{cases} x \text{ si } x \leq a \\ a \text{ si } x \geq a \end{cases} \quad \text{et} \quad \max(a, x) = \begin{cases} a \text{ si } x \leq a \\ x \text{ si } x \geq a \end{cases} \tag{1.9}$$

En termes numériques :

Python :

La valeur absolue est ainsi implémentée

```
print(np.abs(-3),np.max((-3,3)),np.min((-3,3)),min(-3,3),max(-3,3))
```

L'opération consistant à limiter les valeurs d'une fonction à un certain intervalle au moyen de max et min sont ainsi implémentées

Les relations affines et non-linéaires ne sont pas *linéaires*, c'est-à-dire qu'elles ne vérifient pas

$$\begin{cases} x = x_1 + x_2 & \Rightarrow \quad y = y_1 + y_2 \\ x = \alpha x_1 & \Rightarrow \quad y = \alpha y_1 \end{cases} \tag{1.10}$$

lorsque y_1, y_2, y sont les sorties de x_1, x_2, x.

En terme de visualisation :

L'équation (1.10) signifie que la relation entre x et y n'est pas une droite passant par l'origine.

1.4 Association de filtres

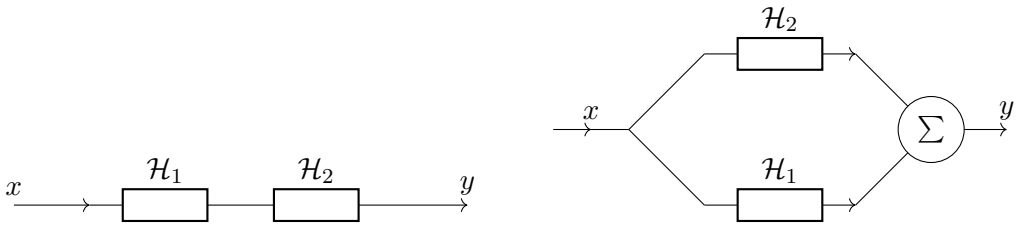

Figure 1.6: À gauche : deux filtres en série.

Cours signal et bruit :

On considère une première façon d'associer deux filtres, en les mettant l'un à la suite de l'autre, on dit aussi en série. Elle est représentée à gauche de la figure 1.6. L'ensemble de ces deux filtres constitue un nouveau filtre noté ici $\mathcal{H}$.

D'un point de vue mathématique :

il s'agit d'une composition de fonctions

$$\mathcal{H} = \mathcal{H}_2 \circ \mathcal{H}_1 \qquad y = \mathcal{H}(x) = \mathcal{H}_2\left[\mathcal{H}_1(x)\right] \tag{1.11}$$

Remarquez que ce qui est dessiné en allant de la gauche vers la droite est représenté dans une équation en allant de la droite vers la gauche.

Cours signal et bruit :

On considère une deuxième façon d'associer deux filtres, en faisant suivre chacun d'eux du même opérateur de somme noté avec $\sum$, on dit qu'on ajoute les sorties des filtres. Elle est représentée à droite de la figure 1.6. L'ensemble de ces deux filtres constitue un nouveau filtre noté ici $\mathcal{H}$.

D'un point de vue mathématique :

il s'agit de la somme de deux fonctions

$$\mathcal{H} = \mathcal{H}_1 + \mathcal{H}_2 \qquad y = \mathcal{H}(x) = \mathcal{H}_1(x) + \mathcal{H}_2(x) \tag{1.12}$$

D'un point de vue électronique :

On observe que la signification des associations est très différente de ce qui se fait en électronique. Avec l'utilisation des quadripôles présentés en annexe B.4, on peut retrouver des idées similaires.

Chapter 2

Signaux temps continu, fonction affine par morceaux

2.1 Des exemples de signaux

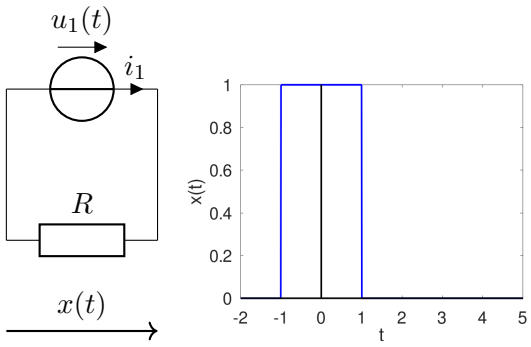

Figure 2.1: Illustration l'équation (2.6) pour $u_1(t) = 1$ pour $t \in [-1, 1]$.

Cours signal et bruit :

Un signal dépendant du temps et ayant une valeur définie pour chaque instant est dit analogique. Il est noté $x(t)$, $y(t)$, $s(t)$... en fonction de la lettre, x,y ou z associée au signal.

Le signal décrit à droite de la figure 2.2 est définie par

$$x(t) = \begin{cases} 0 \text{ si } t < -1 \\ 1 \text{ si } t \in [-1, 1[\\ 0 \text{ si } t \geq 1 \end{cases} \tag{2.1}$$

D'un point de vue mathématique :

On appelle discontinuité les sauts que l'on observe en $t = -1$ et en $t = 1$ sur la droite de la figure 2.1. Dans le cadre de ce cours, on ne s'intéresse pas à la valeur effective du signal sur le saut, en l'occurrence, le signal vaut-il 0, 1 ou une valeur intermédiaire au moment où exactement $t = -1$. En revanche si au cours d'un calcul on utilise des fonctions avec des discontinuités et qu'à la fin du calcul il n'y a plus de discontinuités, la valeur en ces points nous intéresse.

D'un point de vue électronique :

Une discontinuité pour $u(t)$ (ou pour $i(t)$) est effectivement quelque chose qui se produit physiquement : la tension est discontinue lorsqu'on connecte et l'intensité est discontinue lorsqu'on déconnecte. Lorsqu'on met sous tension entre $t = -1$ et $t = 1$, on mesure avec le voltmètre un signal dans le circuit décrit à gauche de la figure 2.1. L'explication

est que quand la source de tension est nulle, il n'y a pas de courant et la tension est nulle. Quand il y a une source de tension, cela créé un courant qui est tel que $x(t) = U_1$.

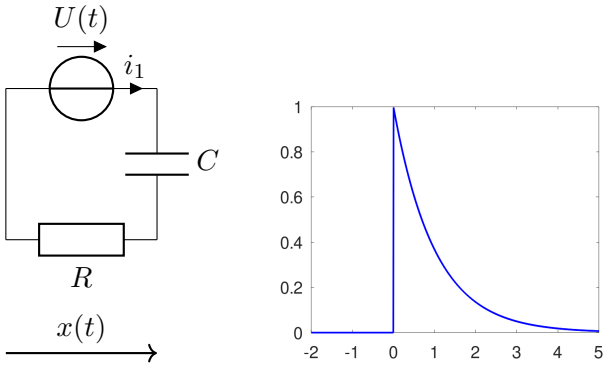

Figure 2.2: Illustration l'équation (2.8) avec $U_1 = 1$V et $RC = 1$s

Cours signal et bruit :

Le signal décrit à droite de la figure 2.2

$$x(t) = \begin{cases} 0 \text{ si } t \leq 0 \\ \\ e^{-t} \text{ si } t > 0 \end{cases} \tag{2.2}$$

Python :

Pour réaliser la droite de la figure 2.2, la première étape est de générer un grand nombre de valeurs entre -2 et 5 qui soient également réparties (ici c'est 10^3).

```
t=np.linspace(-2,5,10**3)
```

Pour chacune de ces valeurs, on calcule la valeur du signal $x(t)$. Cette première instruction n'est pas correcte car elle est non-nulle pour $t < 0$.

```
x=np.exp(-t)
```

Pour la rendre nulle pour $t < 0$, on multiplie par `(t>=0)` qui vaut 1 quand $t \geq 0$ et 0 quand $t < 0$.

```
x=(t>=0)*np.exp(-t)
```

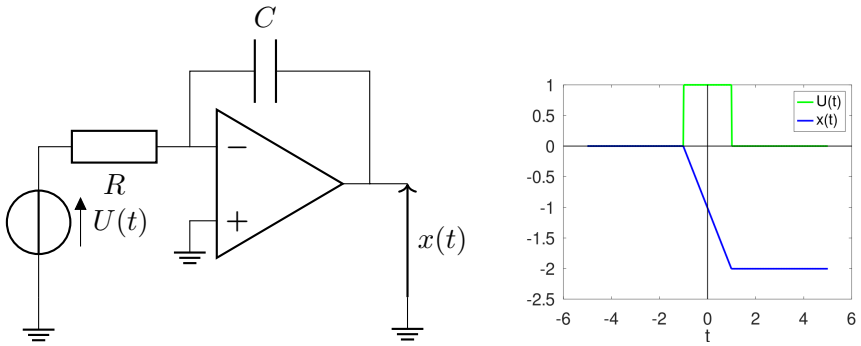

Figure 2.3: À gauche : montage intégrateur avec $RC = 1$. À droite : graphe de $x(t)$ en fonction de t.

Cours signal et bruit :

Le signal décrit à droite de la figure 2.3 est

$$x(t) = \begin{cases} 0 \text{ si } t \leq -1 \\ \\ -(t+1) \text{ si } t \in [-1, 1[\\ \\ -1 \text{ si } t > 1 \end{cases} \tag{2.3}$$

D'un point de vue mathématique :

La formule $-(t+1)$ est l'équation d'une droite, une technique générale pour trouver cette équation de droite est d'utiliser deux points, ici $(-1,0)$ et $(1,-2)$. En notant (t_1,x_1) et (t_2,x_2), l'équation de la droite est définie par

$$\frac{x-x_1}{t-t_1}=\frac{x_2-x_1}{t_2-t_1} \tag{2.4}$$

En l'occurrence,

$$\text{à partir de } \frac{x-0}{t+1}=\frac{-2-0}{1-(-1)},\ \text{on a } x=(-2)/2\times(t+1)=-(t+1) \tag{2.5}$$

2.2 Utilisation du crochet d'Iverson

Le crochet d'Iverson est une façon pratique de coder les équations (2.1), (2.2), (2.3).

$$[\![\mathcal{P}]\!]=\begin{cases}1 & \text{si la proposition } \mathcal{P} \text{ est vraie}\\ 0 & \text{si } \mathcal{P} \text{ est fausse}\end{cases}$$

$$x(t)=[\![|t|\leq 1]\!]=[\![-1<=t<=1]\!] \tag{2.6}$$

$[\![\ldots]\!]$ permet de définir l'échelon d'Heaviside défini par

$$x(t)=[\![t\geq 0]\!] \tag{2.7}$$

Il permet aussi de définir un signal exponentiellement décroissant pour $t\geq 0$ et nul pour $t<0$.

$$x(t)=e^{-t}\,[\![t\geq 0]\!] \tag{2.8}$$

$[\![\ldots]\!]$ permet aussi de représenter des fonctions affines par morceaux (i.e. des segments de droites joints).
Le signal $x(t)$ est une portion de rampe suivi d'un échelon.

$$x(t)=-(t+1)[\![-1\leq t<1]\!]-[\![t\geq 1]\!] \tag{2.9}$$

D'un point de vue mathématique :

La notation utilisant le crochet d'Iverson devient ambigu si on y met deux variables à l'intérieur, parce qu'alors il n'est plus clair à laquelle de ces deux variables dans le crochet, la valeur entre parenthèse signifie. Par exemple

$$[\![\tau<t]\!](0)\ \text{signifie-t-il}\ \begin{cases}[\![0<t]\!]=\begin{cases}0 \text{ si } t\leq 0\\ 1 \text{ si } t>0\end{cases}\\ ou\\ [\![\tau<0]\!]=\begin{cases}0 \text{ si } \tau\geq 0\\ 1 \text{ si } \tau<0\end{cases}\end{cases} \tag{2.10}$$

Dans le cadre de ce cours, on n'utilisera pas cette notation avec deux variables à l'intérieur.

2.3 Présentation de quatre fonctions de base

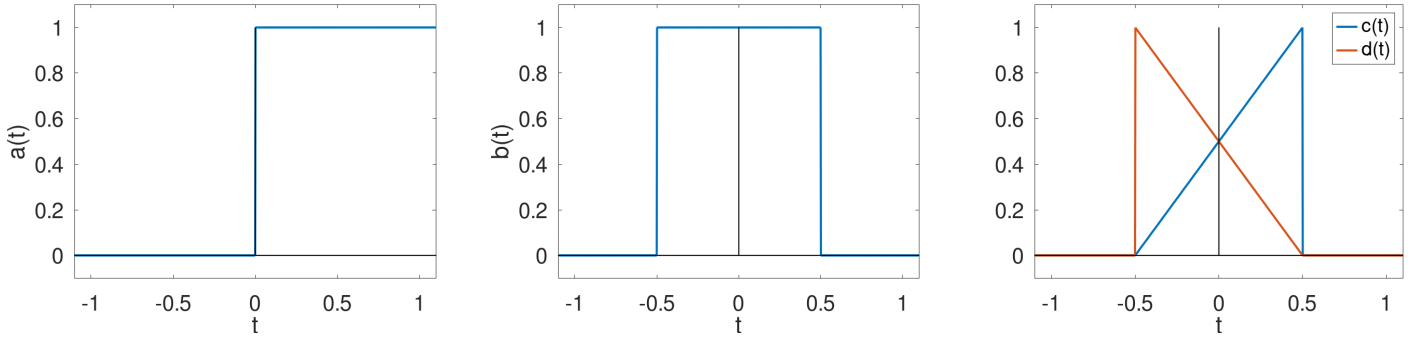

Figure 2.4: Représentation de $\mathbb{H}(t)$, $\Pi(t)$, $\mathbb{C}(t)$, $\mathbb{D}(t)$.

Cours signal et bruit :

L'échelon d'Heaviside est représenté à gauche de la figure 2.4 est noté ici $\mathbb{H}(t)$, il est défini par

$$\mathbb{H}(t) = \begin{cases} 0 \text{ si } t < 0 \\ 1 \text{ si } t \geq 0 \end{cases} \tag{2.11}$$

La porte centrée et de largeur 1 est représentée en haut au milieu de la figure 2.4, elle est notée ici $\Pi(t)$, elle est définie par

$$\Pi(t) = \begin{cases} 0 \text{ si } t < -0.5 \\ 1 \text{ si } t \in [-0.5, 0.5[\\ 0 \text{ si } t \geq 0.5 \end{cases} \tag{2.12}$$

Le demi-triangle croissant de largeur 1 est représenté en haut à droite de la figure 2.4, il est notée ici $\mathbb{C}(t)$, il est définie par

$$\mathbb{C}(t) = \begin{cases} 0 \text{ si } t < -0.5 \\ t + 0.5 \text{ si } t \in [-0.5, 0.5[\\ 0 \text{ si } t \geq 0.5 \end{cases} \tag{2.13}$$

Le demi-triangle décroissant de largeur 1 est représenté en haut à droite de la figure 2.4, il est notée ici $\mathbb{D}(t)$, il est définie par

$$\mathbb{D}(t) = \begin{cases} 0 \text{ si } t < -0.5 \\ 0.5 - t \text{ si } t \in [-0.5, 0.5[\\ 0 \text{ si } t \geq 0.5 \end{cases} \tag{2.14}$$

Le triangle de largeur 2 est représenté en haut à droite de la figure 2.4, il est notée ici $\mathbb{T}(t)$, il est définie par

$$\mathbb{T}(t) = \begin{cases} 0 \text{ si } t < -1 \\ 1 - |t| \text{ si } t \in [-1, 1[\\ 0 \text{ si } t \geq 1 \end{cases} \tag{2.15}$$

Le crochet d'Iverson se prête à une implémentation très simple, parce que une expression conditionnelle utilisant des vecteurs de mêmes tailles vaut 1 si cette expression est correcte et 0 sinon.

Python :

- L'équation (2.11) devient
  `x=(t>=0)` | `x=seb.fonction_H(t)`
- L'équation (2.12) devient
  `x=((t>=-1/2)&(t<=1/2))` | `x=seb.fonction_P(t);`
- L'équation (2.13) devient
  `x=(0.5+t)*((t>=-1/2)&(t<=1/2))` | `x=seb.fonction_C(t);`
- L'équation (2.14) devient
  `x=(0.5-t)*((t>=-1/2)&(t<=1/2))` | `x=seb.fonction_D(t);`
- L'équation (2.15) devient
  `x=(1-np.abs(t))*(np.abs(t)<=1)` | `x=seb.fonction_T(t);`

2.4 Effet d'un retard/avance sur un signal

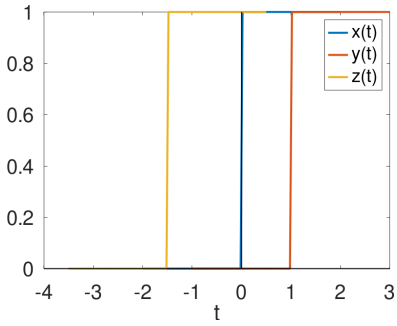

Figure 2.5: À gauche : visualisation en fonction du temps, de trois signaux $x(t) = \mathbb{H}(t)$ au centre, $y(t) = x(t-1)$ à droite et $z(t) = x(t+1.5)$ à gauche.

Cours signal et bruit :

On appelle avancer un signal le fait de transformer $x(t)$ en $x(t+t_0)$. On appelle retarder un signal le fait de transformer $x(t)$ en $x(t-t_0)$.

D'un point de vue électronique :

Introduire un retard ou une avance peut se voir comme retarder ou avancer le moment précis où l'expérience commence. Mais en général, ce qu'on appelle un retard est un décalage dans le temps entre deux signaux. On nomme les composants qui réalisent cette modification une ligne à retard. Il n'est pas physiquement possible de réaliser un composant qui ferait une avance, parce que cela amènerait à prédire l'avenir.

En terme de visualisation :

Lorsqu'on modifie un signal $y(t) = x(t-t_0)$, t_0 étant une valeur fixe, le graphe $t \mapsto x(t)$ est translatée vers la **droite** d'une valeur t_0 alors qu'il y a un signe moins. La gauche de la figure 2.5 illustre où la courbe du milieu est $x(t)$, celle de droite est $x(t-1)$ et celle de gauche est $x(t+1.5)$.
Il existe une façon simple de visualiser un signal retardé ou avancé, cela consiste à modifier l'échelle de temps en lui ajoutant ou en retranchant une valeur au vecteur.
Je considère un premier signal $x(t) = \mathbb{H}(t) = [\![t \geq 0]\!]$. On obtient la gauche de la figure 2.5 avec l'algorithme 1. Celui-ci est implémentée dans A.4. Remarquez que cette fois-ci le retard se code avec un signe plus.

Algorithm 1 générant la figure 2.5.

Créer une échelle de temps `tx` entre -2 et 2 avec 100 points
Calculer `x` un échelon associé à `tx`
Calculer `ty=tx+1`
Calculer `tz=tx-1`
Visualiser `(tx,x,ty,x,tz,x)`.

Python :

```
tx=np.linspace(-2,2,10**2)
x=seb.fonction_echelon(tx)
```

Je considère deux signaux $y(t)$ et $z(t)$, le premier en retardant $x(t)$ et le deuxième en l'avançant.

$$y(t) = x(t-1) \text{ et } z(t) = x(t+1.5) \tag{2.16}$$

Les trois signaux peuvent alors être ainsi représentés

```
ty=tx+1
tz=tx-1.5
fig,ax = plt.subplots()
ax.plot(tx,x,label='x(t)')
ax.plot(ty,y,label='y(t)')
ax.plot(tz,z,label='z(t)')
ax.set_xlabel('t')
ax.legend()
plt.tight_layout()
fig.savefig('nom_figure.png')
fig.show()
```

On peut aussi simuler la gauche de la figure 2.5 en utilisant la même échelle de temps mais cette fois faisant une opération dessus avant de calculer les signaux correspondant.

Algorithm 2 générant aussi la figure 2.5.

Créer une échelle de temps `t` entre -2 et 2 avec 100 points
Calculer `x` un échelon associé à `t`
Calculer `y` un échelon associé à `t-1`
Calculer `z` un échelon associé à `t+1.5`
Visualiser `(t,x,t,y,t,z)`.

D'un point de vue mathématique :

$$x(t) = [\![t_1 \leq t \leq t_2]\!] \quad \Rightarrow \quad x(t-t0) = [\![t_0 + t_1 \leq t \leq t_0 + t_2]\!] \tag{2.17}$$

En terme de visualisation :

On définit une fonction triangle notée $\mathbb{T}(t)$ en avançant $\mathbb{C}(t)$ et en retardant $\mathbb{D}(t)$

$$\begin{aligned} \mathbb{T}(t) &= \mathbb{C}(t+0.5) + \mathbb{D}(t-0.5) = ((t+0.5)+0.5)\,[\![|t+0.5| \leq 0.5]\!]\,(t) + (0.5-(t-0.5))\,[\![|t-0.5| \leq 0.5]\!]\,(t) \\ &= (1-|t|)\,[\![|t| \leq 1]\!]\,(t) \end{aligned} \tag{2.18}$$

$\mathbb{T}(t)$ est représentée à gauche de la figure 2.6. Cette figure est implémentée dans A.3.

2.5 Effet d'une dilatation/concentration de l'échelle des abscisses

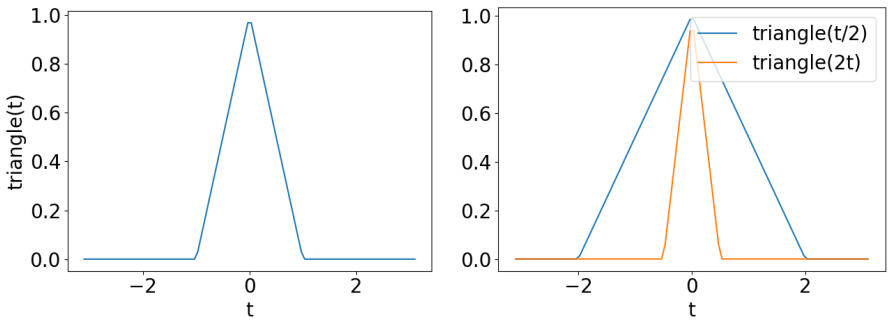

Figure 2.6: À gauche : fonction triangle $\mathbb{T}(t)$. À droite : superposition de $\mathbb{T}(t/2)$ à l'extérieur et de $\mathbb{T}(2t)$ à l'intérieur.

Cours signal et bruit :

On appelle dilater un signal $x(t)$ le fait de le transformer en $x\left(\frac{t}{a}\right)$ lorsque $a > 1$. On appelle concentrer un signal $x(t)$ le fait de le transformer en $x(at)$ lorsque $a > 1$.

En terme de visualisation :

Lorsqu'on modifie un signal $y(t) = x\left(\frac{t}{a}\right)$, $a > 0$ étant une valeur fixe, le graphe $t \mapsto x(t)$ est dilatée si $a > 1$ et concentrée si $a < 1$. La droite de la figure 2.6 montre à l'extérieur la fonction triangle dilatée $\mathbb{T}(t/2)$ et à l'intérieur la fonction triangle concentrée $\mathbb{T}(2t)$.

En termes numériques :

De la même façon que pour les algorithmes 1 et 2, on peut utiliser deux algorithmes différent pour générer le graphe à droite de la figure 2.6. Il importe de remarquer que la multiplication dans le premier algorithme devient une division dans le deuxième et vice-versa.

Algorithm 3 générant la figure 2.6 en modifiant les échelles de temps.

Créer une échelle de temps `tx` entre -3.1 et 3.1 avec 100 points
Calculer `x` un échelon associé à `tx`
Calculer `ty=2*tx`
Calculer `tz=tx/2`
Visualiser à gauche `(tx,x)`
Visualiser à droite `(ty,x,tz,x)`

Algorithm 4 générant aussi la figure 2.6 en modifiant la façon de calculer les fonctions triangle.

Créer une échelle de temps `t` entre -3.1 et 3.1 avec 100 points
Calculer `x` avec fonction_T associé à `t`
Calculer `y` avec fonction_T associé à `t/2`
Calculer `z` avec fonction_T associé à `2t`
Visualiser à gauche `(t,x)`
Visualiser à droite `(t,y,t,z)`

D'un point de vue mathématique :

$$x(t) = [\![t_1 \leq t \leq t_2]\!] \quad \Rightarrow \quad x\left(\frac{t}{a}\right) = [\![at_1 \leq t \leq at_2]\!] \tag{2.19}$$

2.6 Illustration sur les fonctions de base

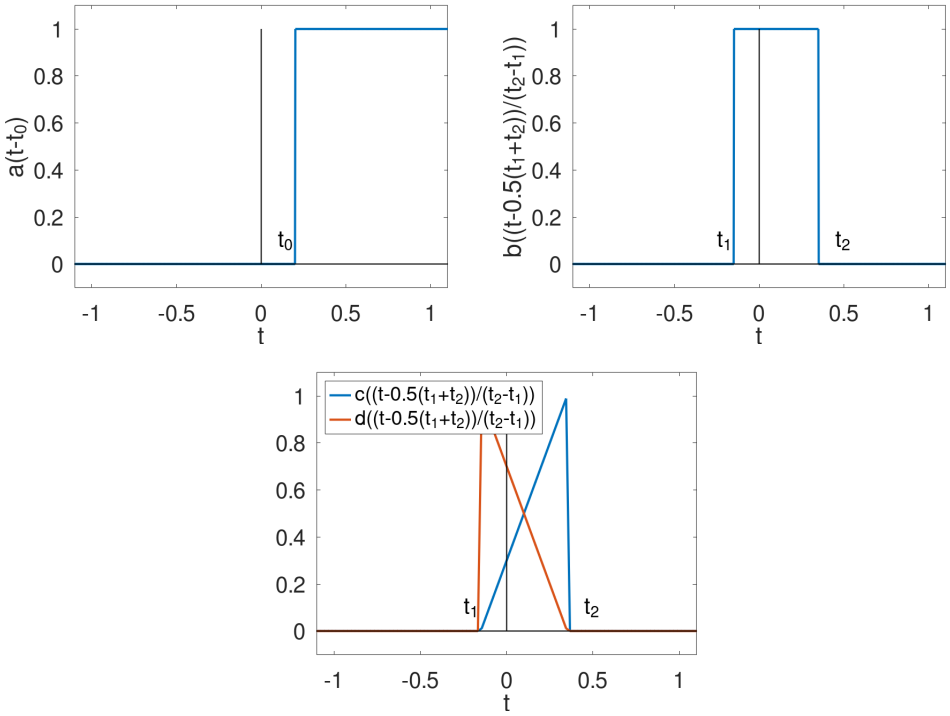

Figure 2.7: visualisation de $\mathbb{H}(t-t_0)$, $\Pi\left(\frac{t-(t_1+t_2)/2}{t_2-t_1}\right)$, $\mathbb{C}\left(\frac{t-(t_1+t_2)/2}{t_2-t_1}\right)$, $\mathbb{D}\left(\frac{t-(t_1+t_2)/2}{t_2-t_1}\right)$.

D'un point de vue mathématique :

Un intervalle $[t_1, t_2]$ décrit par son début t_1 et sa fin t_2 peut aussi être décrit par $\texttt{centre} = \frac{t_1+t_2}{2}$ et $\texttt{longueur} = t_2 - t_1$

$$t \in [t_1, t_2] \quad \Leftrightarrow \quad \frac{|t - \texttt{centre}|}{\texttt{longueur}} \leq 0.5 \tag{2.20}$$

Cours signal et bruit :

On peut adapter une fonction définie sur $[-0.5, 0.5]$ en une fonction définie sur $[t_1, t_2]$ en lui appliquant lui appliquant d'abord un retard de centre $= \frac{t_1+t_2}{2}$ puis une dilatation de longueur $= t_2 - t_1$. La figure 2.7 illustre les fonctions $\mathbb{H}(t)$, $\mathbb{C}(t)$ et $\mathbb{D}(t)$ adaptées à des nouveaux intervalles de temps.

En termes numériques :

De la même façon que pour les algorithmes 1 et 2, on peut utiliser deux algorithmes différent pour générer la droite de figure 2.7. Il importe de remarquer que l'addition, la multiplication et l'ordre sont échangés avec la soustraction, la division et l'ordre inverse.

Algorithm 5 générant la figure 2.7 en modifiant les échelles de temps.

Require: `t0, t1,t2`

Créer une échelle de temps `t` entre -1.1 et 1.1 avec 100 points
Calculer `x` un échelon associé à `t`
Calculer `tx=t+t0`
Visualiser à gauche `(tx,x)`
Calculer `centre=(t1+t2)/2`
Calculer `longueur=t2-t1`
Calculer `y` une porte associée à `t`
Calculer `ty=tx*longueur+centre`
Visualiser au milieu `(ty,y)`
Calculer `z1` avec fonction_C associée à `t`
Calculer `z2` avec fonction_D associée à `t`
Visualiser à droite `(ty,z1,ty,z2)`

Algorithm 6 générant aussi la figure 2.7 en modifiant la façon de calculer les fonctions.

Require: `t0, t1,t2`

Créer une échelle de temps `t` entre -1.1 et 1.1 avec 100 points
Calculer `x` un échelon associé à `t-t0`
Visualiser à gauche `(t,x)`
Calculer `centre=(t1+t2)/2`
Calculer `longueur=t2-t1`
Calculer `y` une porte associé à `(t-centre)/longueur`
Visualiser au milieu `(t,y)`
Calculer `z1` avec fonction_C associé à `(t-centre)/longueur`
Calculer `z2` avec fonction_D associé à `(t-centre)/longueur`
Visualiser à gauche `(t,z1,t,z2)`

Chapter 3

Utilisation de la transformée de Fourier

3.1 Somme, énergie, moyenne et puissance

Cours signal et bruit :

Les signaux déterministes non-périodiques à temps continu que l'on étudie dans ce cours, font en fait partie d'une catégorie appelée signaux à énergie finie et sommable que l'on résume en disant qu'ils sont non-périodiques.

D'un point de vue mathématique :

Ces signaux vérifient que $\int_{-\infty}^{+\infty} x^2(t)\,dt$ et $\int_{-\infty}^{+\infty} |x(t)|\,dt$. Ce n'est pas une condition que l'on ne va pas chercher à tester. Mais il y a des difficultés qui apparaissent quand cette condition n'est pas vérifiée et par exemple $\frac{1}{\sqrt{t}}[\![0 < t \leq 1]\!](t)$ et $\frac{1}{\sqrt{t^2+1}}[\![t \geq 0]\!](t)$ ne la vérifient pas. En fait il est rare (mais possible) qu'à la fois le signal en temps $x(t)$ et sa transformée de Fourier vérifient ces conditions. Du coup le sens que l'on donne à la transformée de Fourier d'un signal est soit la transformée du signal si celui vérifie la condition soit d'être un signal égal à la transformée de Fourier inverse d'une autre fonction qui est d'énergie finie et sommable.

Somme

$$\mathrm{A}_x = \int_{-\infty}^{+\infty} x(t)\,dt \tag{3.1}$$

C'est la cummulation de $x(t)$ au cours du temps, ici il y a une interprétation avec la transformée de Fourier quand le signal est non-périodique, $\mathrm{A}_x = \widehat{X}(0)$.
En particulier

$$x(t) = a[\![t \in [t_1, t_2]]\!] \quad \Rightarrow \quad \mathrm{A}_x = a(t_2 - t_1) \tag{3.2}$$

Le retard ne modifie pas la somme.

$$y(t) = x(t - t_0) \quad \Rightarrow \quad \mathrm{A}_y = \mathrm{A}_x \tag{3.3}$$

La dilatation de l'échelle des abscisses modifie la somme.

$$y(t) = x\left(\frac{t}{a}\right) \quad \Rightarrow \quad \mathrm{A}_y = a\mathrm{A}_x \tag{3.4}$$

La moyenne temporelle est dans le cadre de ce cours défini comme nulle.

$$\mathrm{M}_x = 0 \tag{3.5}$$

D'un point de vue mathématique :

On peut donner un sens à la moyenne temporelle pour des signaux non-périodiques, qui sort du cadre de ce cours

$$\mathrm{M}_x = \lim_{T \to +\infty} \int_{-T/2}^{T/2} x(t)\,dt \tag{3.6}$$

et on peut démontrer que pour les signaux considérés cela vaut 0.

D'un point de vue électronique :

L'énergie et la puissance en traitement du signal sont définies avec $x^2(t)$ au lieu de pour chaque composant le produit de $u(t)$ par $i(t)$, le premier étant la tension aux bornes et le deuxième est l'intensité traversant.

Énergie

$$\mathrm{E}_x = \int_{-\infty}^{+\infty} x^2(t)\,dt \tag{3.7}$$

En particulier

$$x(t) = a[\![t \in [t_1, t_2]]\!] \quad \Rightarrow \quad \mathrm{E}_x = a^2(t_2 - t_1) \tag{3.8}$$

Le retard ne modifie pas l'énergie

$$y(t) = x(t - t_0) \quad \Rightarrow \quad E_y = E_x \tag{3.9}$$

La dilatation de l'échelle des abscisses modifie l'énergie.

$$y(t) = x\left(\frac{t}{a}\right) \quad \Rightarrow \quad E_y = aE_x \tag{3.10}$$

La Puissance des signaux considérés est définie comme nulle.

$$\mathrm{P}_x = 0 \tag{3.11}$$

D'un point de vue mathématique :

Ce sont les même considérations que pour la somme qui permettent de définir

$$\mathrm{P}_x = \lim_{T \to +\infty} \int_{-T/2}^{T/2} x^2(t)\,dt \tag{3.12}$$

qui vaut 0 pour les signaux considérés.

En terme de visualisation :

Pour calculer la somme d'un signal d'énergie finie, il est possible d'utiliser les formules de géométrie classiques. Ainsi la somme de $x(t) = \mathbb{T}(t)$ est la surface d'un triangle de base 2 et de hauteur 1, c'est-à-dire $\mathrm{A}_x = \frac{2\times 1}{2} = 1$.

Cours signal et bruit :

La transformée de Fourier en la fréquence nulle est la somme

$$\mathrm{TF}\left[x(t)\right](0) = A_x \tag{3.13}$$

En appliquant cette formule aux signal $x^2(t)$ on trouve que cela vaut l'énergie.

$$\mathrm{TF}\left[x^2(t)\right](0) = E_x \tag{3.14}$$

En termes numériques :

Il est possible numériquement de trouver une approximation de A_x et E_x en utilisant la fonction proposée `TF` qui lorsqu'appliquée à la fréquence nulle fournit une simple intégrale.

Créer une échelle de temps `t` entre -3 et 3 avec 10000 points
Calculer `x` avec fonction_T associé à `t`
Afficher "la somme est" suivi de TF utilisant `t,x` pour `f=0`.
Afficher "l"energie est" suivi de TF utilisant `t,x*x` pour `f=0`.

Algorithm 7: calculant la somme et l'énergie de $x(t) = \mathbb{T}(t)$.

Cette expérience donne ici le résultat exact pour A_x et une approximation à 10^{-8} pour E_x.

Python :

On créé une échelle de temps avec

```
t=np.linspace(-3,3,10**4)
```

On génère la fonction triangle à l'aide de `fonction_T` défini dans `seb.py`

```
x=seb.fonction_T(t)
```

Le calcul de la A_x et E_x peut se faire avec les équations (3.13) et (3.14) en utilisant `TF` de `seb.py`. On rajoute la partie réelle parce qu'on sait que le résultat est réel. Cette partiel réelle est implémentée avec `np.real`. On fait l'affichage avec `print` en mettant entre accolades l'expression à évaluer lors de l'affichage et en faisant précéder la chaîne de caractère à afficher de la lettre `f`. Les caractères `:.2f` signifient que l'affichage utilise 2 chiffres après la virgule. On peut utiliser aussi `:.2e` pour avoir un affichage avec un exposant.

```
print(f"Ax={np.real(seb.TF(t,x,0)):.2f} Ex={np.real(seb.TF(t,x**2,0)):.2f}")
```

3.2 Transformée de Fourier : définition

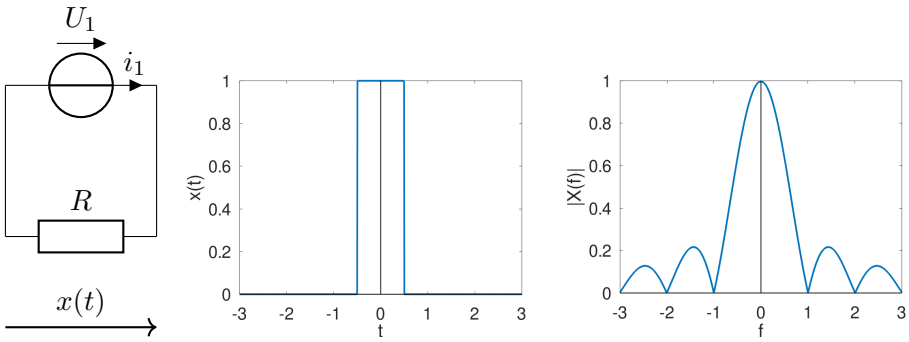

Figure 3.1: Au milieu fonction porte et à droite module de la transformée de Fourier de la fonction porte.

Cours signal et bruit :

Transformée de Fourier pour un signal temps continu non-périodique

$$\widehat{X}(f) = \int_{-\infty}^{+\infty} x(t) e^{-j2\pi ft}\, dt \tag{3.15}$$

f étant une fréquence en Hertz (Hz).

D'un point de vue mathématique :

Cette définition a un sens lorsque $\int_{-\infty}^{+\infty} |x(t)|\, dt < +\infty$ Par exemple, elle n'a pas de sens si $x(t) = 1$ ou si $x(t) = [\![t \geq 0]\!](t)$.

Cours signal et bruit :

Transformée de Fourier inverse d'un spectre défini sur toutes les fréquences et non-périodique

$$x(t) = \int_{-\infty}^{+\infty} \widehat{X}(f) e^{+j2\pi ft}\, dt \tag{3.16}$$

Cette définition suppose que cette intégrale a un sens $\int_{-\infty}^{+\infty} |\widehat{X}(t)|\, dt < +\infty$
Une deuxième définition de la transformée de Fourier inverse est que si on a pu calculer $\widehat{X}(f)$ à partir de $x(t)$ alors par définition $\mathrm{TF}^{-1}\left[\widehat{X}(f)\right](t) = x(t)$.
De même si

En terme de visualisation :

La transformée de Fourier est un complexe, $f \mapsto \widehat{X}(f)$ ne peut donc être représentée sur un graphique 2D car pour chaque valeur de fréquence il y a une partie réelle et une partie imaginaire. On représente généralement le module $|\widehat{X}(f)|$ et parfois aussi la phase $\arg\left(\widehat{X}(f)\right)$.

Python :

Pour tracer la droite de la figure 3.1, on créé d'abord une échelle de temps

```
t=np.linspace(-3,3,10**3)
```

On utilise la fonction porte dans `seb.py` pour générer le signal $x(t)$

```
x=seb.fonction_P(t)
```

On créé une échelle de fréquence f

```
f=np.linspace(-3,3,10**3)
```

On calcule la transformée de Fourier notée $\widehat{X}(f)$

```
X=seb.TF(t,x,f)
```

On en déduit le module `np.abs(X)` que l'on trace en utilisant aussi `f`.
La transformée de Fourier inverse s'utilise de la même façon

```
x2=seb.TFI(f,X,t)
```

Attention il n'est pas possible de retrouver $x(t)$ en utilisant seulement le module de $\widehat{X}(f)$

D'un point de vue mathématique :

Pour un complexe $z = a + jb$,

- $a = \mathrm{Re}(z)$ s'appelle la partie réelle.
- $b == \mathrm{Im}(z)$ s'appelle la partie imaginaire.
- $\sqrt{a^2 + b^2} = |z|$ s'appelle le module.
- $\arg(z)$ s'appelle l'argument de z, on $z = |z|e^{j\phi}$ et $a = |z|\cos(\arg(z))$ et $b = |z|\sin(\arg(z))$. La phase d'un nombre complexe nul n'est pas défini.
- $\overline{z} = a - jb$ s'apelle le conjugué de z.
- Si $b = 0$, alors z est un réel. Et si $a = 0$, alors z est un imaginaire pur.

Python :

On peut aussi en déduire l'argument `np.angle(X)` à partir du complexe X et donc la phase de la transformée de Fourier quand X est la transformée de Fourier de t, x. On obtient le conjugué d'un complexe `z` avec `z.conjugate()`

Cours signal et bruit :

La figure 3.1 montre à gauche un montage très simple avec une tension continue imposée par un générateur de tension pendant les instants $t \in [-0.5, 0.5]$. Au milieu c'est la visualisation du signal $x(t)$ et à droite c'est le module de la transformée de Fourier $|\widehat{X}(f)|$. L'implémentation est en A.9.

$$\widehat{X}(f) = \int_{-\infty}^{+\infty} x(t)e^{-j2\pi ft}\,dt = \frac{\sin(\pi f)}{\pi f} = \mathrm{sinc}(f) \tag{3.17}$$

On définit **ici** $\mathrm{sinc}(x) = \frac{\sin(\pi x)}{\pi x}$, mais attention, cette définition est celle plutôt utilisée dans la communauté des physiciens, celle des mathématiciens utilisent plutôt la définition $\mathrm{sinc}(x) = \frac{\sin(x)}{x}$. Ces fonctions sont appelées sinus cardinal.

Python :

On peut vérifier que Python utilise la même définition

```
assert np.abs(np.sinc(1))<1e-12
```

L'équation (3.1) permet de confirmer le résultat : $\mathrm{A}_x = 1 = \widehat{X}(0)$.

Cours signal et bruit :

On obtient le tableau suivant :

• $\mathrm{TF}\left[e^{-t}[\![t \geq 0]\!]\right](f) = \frac{1}{1+j2\pi f}$	• $\mathrm{TF}^{-1}\left[\frac{1}{1+j2\pi f}\right](t) = e^{-t}[\![t \geq 0]\!] = e^{-t}\mathbb{H}(t)$
• $\mathrm{TF}\left[[\![t \in [-0.5, 0.5]]\!]\right](f) = \frac{\sin(\pi f)}{\pi f}$	• $\mathrm{TF}^{-1}\left[\frac{\sin(\pi f)}{\pi f}\right](t) = [\![t \in [-0.5, 0.5]]\!] = e^{-t}\Pi(t)$

Table 3.1: de transformées de Fourier

Dans ces deux exemples, l'écriture intégrale de TF^{-1} n'a pas de sens.

3.3 Simuler numériquement des intégrales de fonctions non-sommables

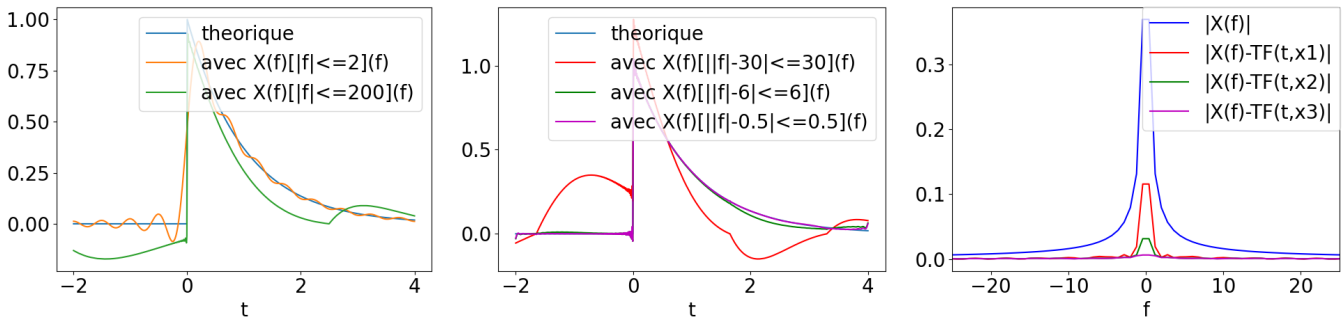

Figure 3.2: À gauche et au milieu approximation de e^{-t}. À droite visualisation en terme de spectre.

L'estimation de $\mathrm{TF}^{-1}\left[\frac{1}{1+j2\pi f}\right](t)$ ou de $\mathrm{TF}^{-1}\left[\frac{\sin(\pi f)}{\pi f}\right](t)$ pose le même genre de problème. Pour avoir une bonne précision, il faut à la fois décrire finement le spectre et considérer aussi des grandes fréquences parce que leur contribution n'est pas négligeable.
La gauche de la figure 3.2 montre en une courbe orange relativement oscillante estimant $x(t)$ en utilisant les valeurs de $\widehat{X}(f) = \frac{1}{1+j2\pi f}$ en utilisant 1000 points entre -2 et 2Hz. La courbe verte avec des ondulations plus amples estime $x(t)$ en utilisant $\widehat{X}(f)$ avec aussi 1000 points entre -200 et 200Hz. La courbe bleue donne le résultat théorique de $x(t) = [\![t \geq 0]\!]e^{-t}$. La première courbe n'est pas correcte car elle ne décrit pas correctement le comportement haute fréquence. La deuxième n'est pas correcte parce qu'elle n'utilise pas assez de points.
La droite de la figure 3.2 utilise des approximations successives du spectre. La courbe orange $x_3(t)$ est obtenue avec 100 points régulièrement répartis dans l'intervalle $[-42, 42]$. Cette approximation est moins bonne que les deux approximations montrées à gauche. C'est la différence entre $\widehat{X} - \mathrm{TF}[x_3(t)](f)$ qui est approchée encore avec 100 points sur l'intervalle $[-3.1, 3.1]$ et qui ajoutée à $x_3(t)$ donne une bonne approximation de $x(t)$.
L'implémentation de la figure 3.2 est dans A.1.

Algorithm 8 Algorithme de la figure 3.2

Créer l'échelle de temps `t` souhaitée
Créer une échelle de fréquence `f` très large
Initialiser
Calculer $\widehat{U}(f) = \mathrm{TF}[u(t)](f)$
Calculer $\widehat{X}(f) = \frac{1}{j2\pi fRC}\widehat{U}(f)$
Calculer $x(t) = \mathrm{TF}^{-1}\left[\widehat{X}(f)\right](f)$
Retrancher la première valeur de $x(t)$ au vecteur `x`

3.4 Propriété de la transformée de Fourier de la dilatation de l'échelle des temps

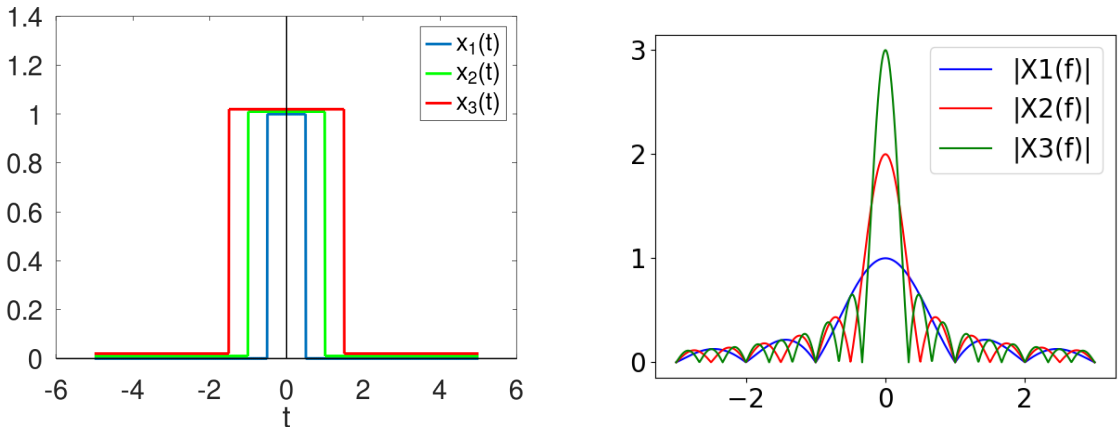

Figure 3.3: À gauche : fonction porte plus ou moins dilatée. À droite : module des transformées de Fourier correspondantes.

Cours signal et bruit :

Vis-à-vis de la dilatation de l'échelle des temps

$$y(t) = x\left(\frac{t}{a}\right) \quad \Rightarrow \quad \widehat{Y}(f) = a\widehat{X}(af) \tag{3.18}$$

D'un point de vue mathématique :

L'équation (3.18) montre par exemple que s'il est possible de donner un sens à $\mathrm{TF}\,[1]\,(f)$, remarquant que la fonction constante $x(t) = 1$ reste inchangée lorsqu'on dilate l'échelle des temps $x\left(\frac{t}{a}\right) = x(t)$, la transformée de Fourier de $\mathrm{TF}\,[1]\,(f)$ doit rester identique à elle-même lorsqu'on la multiplie par n'importe quelle coefficient a.

En terme de visualisation :

La figure 3.3 est obtenue à gauche en introduisant une dilatation de l'échelle des temps à partir de la figure 3.1. D'une façon générale, une dilatation d'un signal amplifie le spectre et le concentre.

Cours signal et bruit :

$$x(t) = \Pi\left(\frac{t}{a}\right) \quad \Rightarrow \quad \widehat{X}(f) = a\frac{\sin(\pi af)}{\pi af} = \frac{\sin(\pi af)}{\pi f} = a\,\mathrm{sinc}(af) \tag{3.19}$$

3.5 Propriété de la transformée de Fourier d'un retard ou d'une avance

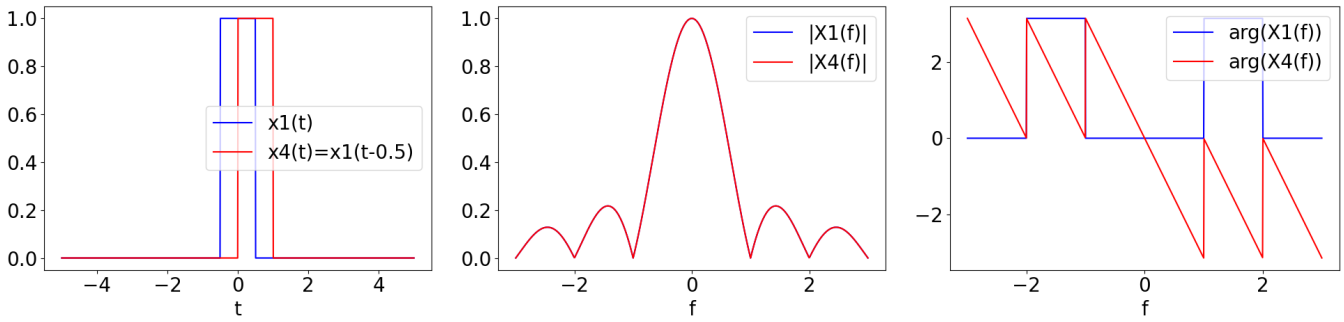

Figure 3.4: À gauche : Fonction porte sans retard et avec retard. Au milieu : module des transformées de Fourier correspondantes. À droite : argument des transformées de Fourier correspondantes (phase).

Cours signal et bruit :

La transformée de Fourier d'un signal retardé de t_0 a le même module et sa phase est diminuée par $-2\pi f t_0$.

$$y(t) = x\left(t - t_0\right) \quad \Rightarrow \quad \widehat{Y}(f) = \widehat{X}(f) e^{-j2\pi f t_0} \tag{3.20}$$

En terme de visualisation :

La figure 3.4 est obtenue à partir de la figure 3.1 en introduisant un retard. Les implémentations se trouvent dans A.7. Un retard ne modifie pas le module de la transformée de Fourier, en revanche il introduit un déphasage. Le graphe 3 de la figure 3.4 indique la phase associée à $\mathrm{TF}\left[\Pi(t)\right](f)$. Il se trouve que $\mathrm{TF}\left[\Pi(t)\right](f)$ est réelle soit positive dans ce cas la phase vaut 0 soit négative et dans ce cas sa phase vaut π. Ceci explique le signal carré bleu variant entre 0 et π. Du fait du retard, $\mathrm{TF}\left[\Pi(t-0.5)\right](f) = \mathrm{TF}\left[\Pi(t)\right](f) e^{-j\pi f}$ a une phase égale à $0 - \pi f$ ou à $\pi - \pi f$. Dans les deux cas elle est décroissante avec une pente de $-\pi$, et des discontinuités permettant aussi de rester entre $-\pi$ et π.

D'un point de vue électronique :

Un signal retardé est aussi appelé un signal avec un retard de phase.

3.6 Symétrie et transformée de Fourier

En terme de visualisation :

Quand on représente le module de la transformée de Fourier d'un signal, le graphe est symétrique par rapport à l'axe des
C'est ce que l'on observe sur les figures 3.3 et 3.4.

D'un point de vue mathématique :

- Un signal pair est défini par

$$x(t) = x(-t) \tag{3.21}$$

  Visuellement il est symétrique par rapport à l'axe des ordonnées.

- Un signal impair est défini par

$$x(t) = -x(-t) \tag{3.22}$$

  Visuellement il a une symétrie centrale par rapport au point $(0, 0)$.

Cours signal et bruit :

Quand $x(t)$ est réel (ce qui est toujours le cas dans ce cours), on a

- $|\mathrm{TF}\left[x(t)\right](f)|$ est pair
- $\arg\left(\mathrm{TF}\left[x(t)\right](f)\right)$ est impair

On peut rajouter que si $x(t)$ est pair, alors sa transformée de Fourier est réelle. Et si $x(t)$ est impair alors sa transformée de Fourier est imaginaire pur.

Chapter 4

Diracs

4.1 Introduction à la notion de distribution de Dirac

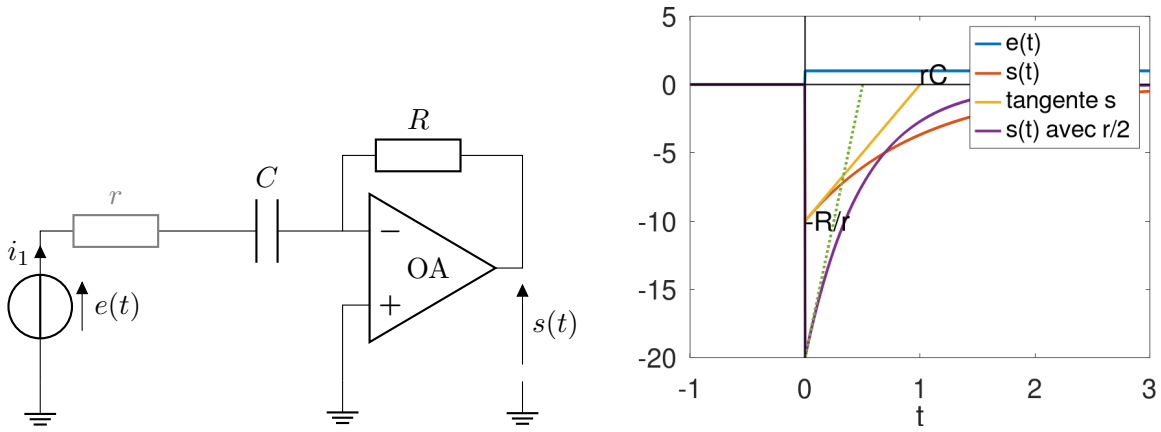

Figure 4.1: À gauche : montage dérivateur avec une résistance r faible et inconnue. À droite réponse dépendant de la résistance r et illustrée avec deux valeurs de r.

D'un point de vue électronique :

Le montage décrit à gauche de la figure 4.1 est un montage dérivateur alimenté par un générateur de tension continue. Ce montage décrit une saute de tension. À droite de la figure 4.1, $e(t)$ en rouge est la tension d'entrée, c'est un échelon. $s(t)$ en bleu est la tension de sortie, c'est un signal négatif en exponentiel et croissant. $\frac{ds}{dt}$ est la dérivée représentée par la courbe en verte. On peut noter que la surface dans le triangle est $\frac{1}{2}\frac{R}{r}rC = \frac{1}{2}RC$ et la surface sous l'exponentielle est RC. Les deux ne dépendent pas de r. On a ainsi une description de cette saute de tension qui ne dépend pas de r.
Ce qu'on appelle un Dirac est la limite théorique de cette saute de tension quand r tend vers 0, plus précisément on écrit que $s(t) = -RC\delta(t)$.

4.2 Règles de calcul pour le Dirac

Cours signal et bruit :

Intégration :

$$\int_{-\infty}^{+\infty} s(t)\delta(t)\,dt = s(0) \tag{4.1}$$

Dérivation d'une discontinuité en $t = 0$:

$$\frac{d}{dt}[\![t \geq 0]\!] = \delta(t) \tag{4.2}$$

Multiplication par une fonction :

$$s(t)\delta(t) = s(0)\delta(t) \tag{4.3}$$

D'un point de vue mathématique :

En général la multiplication d'une fonction avec une distribution n'est pas définie.

D'un point de vue électronique :

Le théorème de Millman appliqué au montage 4.1 en utilisant les impédances et les transformées de Fourier des signaux donne

$$\frac{\widehat{E}(f)}{\frac{1}{Cj2\pi f}} + \frac{\widehat{S}(f)}{R} = 0 \quad \Rightarrow \quad \widehat{S}(f) = -jRC2\pi f\widehat{E}(f) \tag{4.4}$$

En remplaçant la multiplication $j2\pi f$ par la dérivation

$$s(t) = -RC\frac{d}{dt}e(t) = -RC\frac{d}{dt}[\![t \geq 0]\!] = -RC\delta(t) \tag{4.5}$$

À l'inverse on peut interpréter l'équation (4.4) avec une intégration

$$\widehat{E}(f) = -\frac{1}{jRC2\pi f}\widehat{S}(f) \quad \Rightarrow \quad e(t) = -\frac{1}{RC}\int_{-\infty}^{t} s(\tau)\,d\tau \tag{4.6}$$

En remplaçant $s(t)$ par $\delta(t)$ on obtient l'échelon pour $e(t)$.

$$e(t) = \int_{-\infty}^{+\infty} [\![\tau \leq t]\!](\tau)\delta(\tau)\,d\tau = \left\{ \begin{array}{l} 0 \text{ si } t < 0 \\ \\ 1 \text{ si } t > 0 \end{array} \right\} = [\![t \geq 0]\!](t) \tag{4.7}$$

Cours signal et bruit :

La transformée de Fourier d'un Dirac est 1

$$\mathrm{TF}\left[\delta(t)\right](f) = 1 \qquad\qquad \mathrm{TF}^{-1}\left[1\right](t) = \delta(t)$$

On rajoute cela à la table des transformations de Fourier 3.1.

Cours signal et bruit :

On définit aussi Dirac retardé ou avancé dont l'intégrale vaut :

$$\int_{-\infty}^{+\infty} s(t)\delta(t-t_0)\,dt = s(t_0) \text{ et } s(t)\delta(t-t_0) = s(t_0)\delta(t-t_0) \tag{4.8}$$

4.3 Règles de calcul pour dériver un signal discontinu

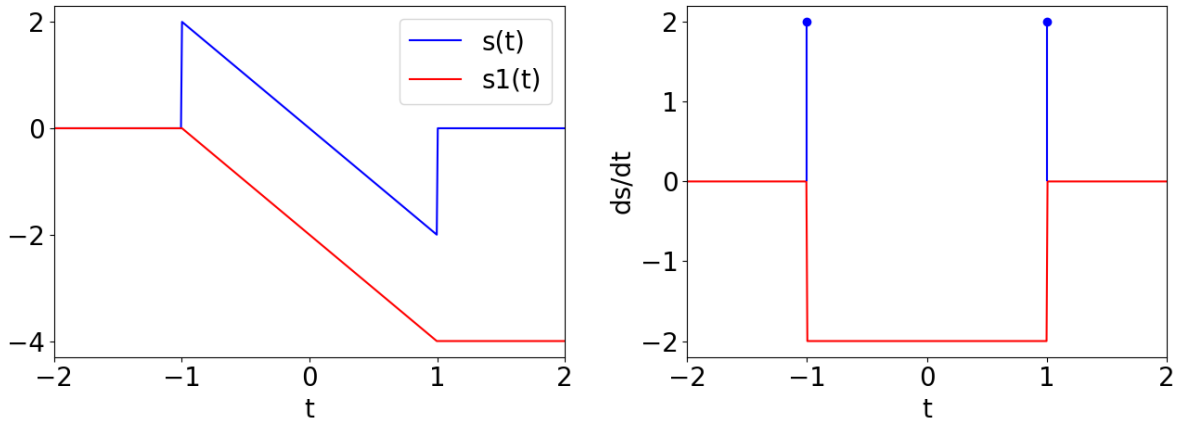

Figure 4.2: Exemple de signal $s(t)$ et de sa dérivée. La courbe rouge sur le premier graphe est $s(t)$ sans ses discontinuités. Celle sur le deuxième graphe est la dérivée de $s_1(t)$.

On définit la discontinuité d'un signal en un instant particulier t_0 par

$$s(t_0^+) - s(t_0-) = \lim_{\substack{t \to t_0 \\ t > t_0}} s(t) - \lim_{\substack{t \to t_0 \\ t < t_0}} s(t) \tag{4.9}$$

La gauche de la figure 4.2 montre une discontinuité positive en $t = -1$ où la courbe passe de la valeur 0 à la valeur 2 et une discontinuité positive en $t = 1$ où la courbe passe de de la valeur -2 à la valeur 0. On a les valeurs suivantes

$$s(-1^-) = 0, \quad s(-1^+) = 2, \quad s(1^-) = -2, \quad s(1^+) = 0 \tag{4.10}$$

D'un point de vue mathématique :

Dérivation d'une courbe discontinue en $t_0, t_1, \ldots$

$$\frac{d}{dt}s(t) = \sum_n \delta(t - t_n)(s(t_n^+) - s(t_n^-)) + \frac{d}{dt}\tilde{s}(t) \tag{4.11}$$

où $\tilde{s}(t)$ est la courbe sans les discontinuités et $s(t_n^+) - s(t_n^-)$ est la discontinuité en t_n.
Cette partie du cours est différente du cours sur les équations différentielles n'utilisant pas le Dirac parce qu'un cours résolvant l'équation différentielle pour $t \geq 0$, n'a pas absolument besoin de dériver une discontinuité. Mais dans ce cours, les signaux sont supposés avoir un sens pour $t < 0$ et il y a donc forcément des discontinuités.

En terme de visualisation :

Graphiquement on représente un Dirac avec une flèche verticale orientée vers le haut si le signe est positif et vers le bas sinon. La longueur de la flèche dépend du coefficient devant le Dirac.
Le signal à gauche de la figure 4.2 est

$$s(t) = -2t[\![t \in [-1, 1]]\!](t) = \begin{cases} 0 \text{ si } t < -1 \\ -2t \text{ si } t \in [-1, 1] \\ 0 \text{ si } t > 1 \end{cases} \tag{4.12}$$

Le signal a deux discontinuités une en $t = -1$ vers le haut d'une amplitude de 2 et une vers le haut en $t = 1$ d'une amplitude de 2. Sans considérer ces discontinuités la dérivée est de -2 pour $t \in [-1, 1]$ et 0 sinon. On obtient ainsi le signal représenté à droite de la figure 4.2.

$$\frac{ds}{dt} = -2[\![t \in [-1, 1]]\!] + 2\delta(t + 1) + 2\delta(t - 1) \tag{4.13}$$

À l'inverse si on considère le signal à droite de la figure 4.2, $x(t) = -2[\![t \in [-1, 1]]\!] + 2\delta(t+1) + 2\delta(t-1)$, son intégrale vaut

$$\begin{aligned} y(t) = \textstyle\int_{-\infty}^{t} x(\tau)\, d\tau &= \begin{cases} 0 & +0 \quad +0 \quad \text{si} \quad t < -1 \\ \int_{-1}^{t} -2\, d\tau & +2 \quad +0 \quad \text{si} \quad t \in]-1, 1[\\ -2\times 2 & +2 \quad +2 \quad \text{si} \quad t > 1 \end{cases} \\ &= \left\{ \begin{array}{lll} 0 & \text{si} & t < -1 \\ (-2t - 2) + 2 = -2t & \text{si} & t \in]-1, 1[\\ 0 & \text{si} & t > 1 \end{array} \right\} = -2t[\![t \in [-1, 1]]\!](t) = s(t) \end{aligned} \tag{4.14}$$

qui est le signal à gauche de la figure 4.2.

Python :

L'évaluation du signal représenté à gauche sur la figure 4.2 s'implémente de façon similaire au crochet d'Iverson, une fois l'échelle de temps créée.

```
s=(t>=-1)*(t<=1)*(-2*t)
```

En ce qui concerne la droite de la figure 4.2, je note `dsdt` la dérivée du signal au sens classique, `t_` les instants associés aux Dirac et `dsdt_`, les pondérations des Diracs. `array` est une façon de définir un vecteur, elle est différente de [-1,1] qui est une liste de nombre et ne relève pas du module `numpy` notée ici `np`.

```
dsdt=(t>=-1)*(t<=1)*(-2);
t_,dsdt_=np.array([-1,1]),np.array([2,2])
```

Pour représenter les Diracs visible à droite de la figure 4.2, je propose d'utiliser `stem`. L'option `basefmt` dans `stem`, sert ici à supprimer l'axe horizontal.

```
fig,ax = plt.subplots()
ax.plot(t,dsdt,'b-')
ax.stem(t_,dsdt_,'b-',basefmt=" ")
```

`seb.py` donne un outil pour approcher la dérivée, mais cet outil suppose qu'il n'y a pas de discontinuité. Tout d'abord on détecte les discontinuités avec la fonction `diff` de `numpy` qui calcule la différence termes à termes d'un signal on cherche les indices pour la valeur absolue de cette différence dépasse un seuil. Ici le seuil est choisi à 0.1.

$$n \text{ est un indice de discontinuite si } |s_{n+1} - s_n| \geq \text{seuil} \tag{4.15}$$

```
ind = np.argwhere(abs(np.diff(s))>0.1)
```

On trouve ensuite les instants associés et les valeurs de cette différence en considérant d'une part `t` l'échelle en temps et d'autre part le signal `s` En appelant $\mathcal{I}$ l'ensemble des indices associés à l'équation (4.15) et t_n et $s(t_n)$ les instants et les valeurs du signal à ces instants. t_n est l'instant associé à la discontinuité et $s(t_{n+1}) - s(t_n)$ est la valeur de cette discontinuité.

$$\text{Pour } n \in \mathcal{I}, \quad t_n \text{ et } s(t_{n+1}) - s(t_n) \tag{4.16}$$

```
t_app_=(t[ind]+t[ind+1])/2
dsdt_app_=np.diff(s)[ind]
```

Pour chaque Dirac, on retranche au signal un échelon ayant une discontinuité au même instant et de la même longueur. Le signal sans discontinuités est ici noté `s1`.

$$s_{\text{sans discontinuités}}(t) = s(t) - \sum_{n \text{ discontinue}} (s(t_{n+1}) - s(t_n))\mathbb{H}(t - t_n) \tag{4.17}$$

```
s_ech1 = dsdt_app_[0][0]*seb.retarder(t,seb.fonction_H(t),t_app_[0][0])
s_ech2 = dsdt_app_[1][0]*seb.retarder(t,seb.fonction_H(t),t_app_[1][0])
s1 = s-s_ech1-s_ech2
```

Pour obtenir la dérivée classique du signal $s(t)$, on applique la fonction `seb.deriver` à `s1`.

```
dsdt_app=seb.deriver(t,s1)
```

Le résultat obtenu contient un pic là où il y a une discontinuité.
L'implémentation de la figure 4.2 est dans A.8.

4.4 Condition initiale et Dirac

D'un point de vue électronique :

La notion de condition initiale est liée à la possibilité de décrire complètement l'état d'un système à un instant donné, c'est ce qu'on fait en automatique. Plus précisément choisir une condition initiale c'est considérer que la sortie du système ainsi qu'éventuellement certaines de ses dérivées décrivent complètement le système.
Cependant lorsqu'on ne connaît un système que par sa relation entrée-sortie, la description de l'état du système à un instant donné revient à imaginer une entrée passé. C'est ce point de vue qui est utilisé dans ce cours.

Cours signal et bruit :

Il n'y a pas de notion de condition initiale dans ce cours, cependant il est parfois possible rajouter un Dirac dans une relation entrée-sortie pour modéliser un comportement similaire au fait d'utiliser une condition initiale.

Exemple pour illustrer :

Ainsi dans l'équation différentielle

$$\frac{d}{dt}y(t) + y(t) = x(t) + y(0)\delta(t) \tag{4.18}$$

est équivalente à l'équation différentielle $\frac{d}{dt}y(t) + y(t) = x(t)$ pour $t \geq 0$ avec $y(0)$ non-nul.
En effet dans le cadre de ce cours, on vérifie que

$$y(t) = e^{-t}\int_{-\infty}^{t} e^{\tau}x(\tau)\,d\tau + y(0)[\![t \geq 0]\!](t)e^{-t} \tag{4.19}$$

est bien solution de l'équation (4.18) :

$$\frac{d}{dt}\left[e^{-t}\int_{-\infty}^{t} e^{\tau}x(\tau)\,d\tau + y(0)[\![t \geq 0]\!](t)e^{-t}\right] = -e^{-t}\int_{-\infty}^{t} e^{\tau}x(\tau)\,d\tau + e^{-t}e^{t}x(t) + y(0)\delta(t) - y(0)e^{-t}[\![t \geq 0]\!](t) \tag{4.20}$$

Et dans le cadre du cours sur la résolution des équations différentielles, on a d'une part une solution générale $y(t) = Ae^{-t}$ pour $t \geq 0$ qui pour coïncider avec la valeur en $t = 0$ amène à $A = y(0)$ On a aussi une solution particulière obtenue en remplaçant A par une fonction $A(t)$ Constatant que

$$\frac{d}{dt}y(t) + y(t) = \left(\frac{d}{dt}A(t)\right)e^{-t} - A(t)e^{-t} + A(t)e^{-t} = x(t) \tag{4.21}$$

on définit

$$A(t) = \int_{0}^{t} e^{\tau}x(\tau)\,d\tau \tag{4.22}$$

On obtient ainsi une expression identique pour $t \geq 0$

$$y(t) = e^{-t}\int_{0}^{t} e^{\tau}x(\tau)\,d\tau + y(0)e^{-t} \tag{4.23}$$

4.5 Dérivées d'un Dirac

Cours signal et bruit :

On définit $\delta'(t)$ et par suite les dérivées successives de $\delta(t)$. Mais dans le cadre de ce cours, on n'utilise que leur définition en tant que dérivée successive de $\delta(t)$.

D'un point de vue mathématique :

Les distributions dérivées de $\delta(t)$ ont des propriétés surprenantes, ainsi

$$\int_{-\infty}^{+\infty} s(t)\delta'(t - t_0)\,dt \quad \neq \quad s'(t_0) \text{ et } s(t)\delta'(t - t_0) \quad \neq \quad s'(t_0)\delta(t - t_0)$$

Chapter 5

Transformées de Fourier, dérivation et équations différentielles

5.1 Introduction

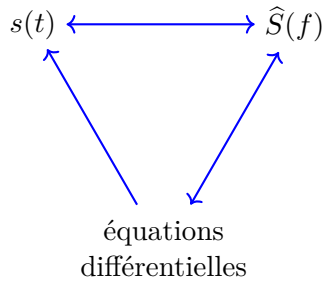

Figure 5.1: Représentation schématique des transformations qu'il est possible d'effectuer parfois avec un calcul parfois avec une simulation

Cours signal et bruit :

La figure 5.1 donne trois façons de définir un signal, $s(t)$ indique la relation en fonction du temps (voir chapitre 2). $\widehat{S}(f)$ indique la dépendance vis-à-vis de la fréquence (voir chapitre 3). Vous avez déjà eu un cours présentant les équations différentielles. Ce cours montrait aussi la façon analytiquement de trouver $s(t)$ à partir de ces équations différentielles rappelée très brièvement en section 5.4 pour le type d'équations différentielles qui nous intéresse ici, c'est la flèche diagonale contre-oblique [1]. Numériquement il est aussi possible de simuler les solutions des équations différentielles, ceci est présenté en section 5.5. Enfin pour une certaine catégorie de spectres, il est très simple d'identifier équations différentielles et spectres, ceci est présenté en section 5.2, ceci correspond à la flèche diagonale oblique [2]

5.2 Transformer la définition d'un spectre en une équation différentielle définissant un signal

D'un point de vue mathématique :

Tout inverse d'un polynôme de variable $j2\pi f$ est la transformée de Fourier d'une équation différentielle.

Considérant

$$\widehat{Y}(f) = \frac{1}{a_0 (j2\pi f)^N + a_1 (j2\pi f)^{N-1} + \ldots + a_N} \tag{5.1}$$

$\widehat{Y}(f)$ vérifie aussi

$$a_0 (j2\pi f)^N \widehat{Y}(f) + a_1 (j2\pi f)^{N-1} \widehat{Y}(f) + \ldots + a_N \widehat{Y}(f) = 1 \tag{5.2}$$

[1] direction correspondant à \
[2] direction correspondant à /.

On remplace chaque occurrence de $j2\pi f$ par $\frac{d}{dt}$ et 1 par $\delta(t)$ pour obtenir

$$a_0\frac{d^N}{dt^N}y(t)+a_1\frac{d^{N-1}}{dt^{N-1}}y(t)+\ldots+a_Ny(t)=\delta(t) \tag{5.3}$$

Plus généralement, quand on a, non pas juste l'inverse d'un polynômes en $j2\pi f$ mais un quotient de polynômes, il suffit d'abord d'interprétant l'inverse du dénominateur avec une équation différentielle et ensuite considérer le numérateur opur en déduire une deuxième relation. Considérant

$$\widehat{Y}(f)=\frac{b_0\left(j2\pi f\right)^M+\ldots+b_M}{a_0\left(j2\pi f\right)^N+a_1\left(j2\pi f\right)^{N-1}+\ldots+a_N} \tag{5.4}$$

On pose $\widehat{Y}_1(f)=\frac{1}{a_0(j2\pi f)^N+a_1(j2\pi f)^{N-1}+\ldots+a_N}$ de sorte que $\widehat{Y}(f)=\left(b_0\left(j2\pi f\right)^M+\ldots+b_M\right)$ Cela conduit à

$$\begin{cases} a_0\frac{d^N}{dt^N}y_1(t)+a_1\frac{d^{N-1}}{dt^{N-1}}y_1(t)+\ldots+a_Ny_1(t)=\delta(t) \\ y(t)=b_0\frac{d^M}{dt^M}y_1(t)+\ldots+b_My_1(t) \end{cases} \tag{5.5}$$

Exemple pour illustrer :

L'équation (5.13) permet d'en déduire une équation différentielle en posant d'abord $y_1(t)$ tel que $y(t)=\mathtt{RC}\frac{d}{dt}y_1(t)$ ce qui amène à

$$\begin{cases} \widehat{Y}(f)=\mathtt{RC}j2\pi f\widehat{Y}_1(f) \\ \widehat{Y}_1(f)=\frac{1}{j2\pi f\mathtt{RC}+1} \end{cases} \tag{5.6}$$

Et finalement on obtient

$$\begin{cases} y(t)=\mathtt{RC}\frac{d}{dt}y_1(t) \\ \mathtt{RC}\frac{d}{dt}y_1(t)+y_1(t)=\delta(t) \end{cases} \tag{5.7}$$

5.3 Propriété de la transformée de Fourier vis-à-vis de la dérivation

Cours signal et bruit :

Une propriété importante des transformées de Fourier est de transformer une dérivation en produit par $j2\pi f$.

$$y(t)=\frac{d}{dt}x(t) \quad\Leftrightarrow\quad \widehat{Y}(f)=\widehat{X}(f)\,j2\pi f \tag{5.8}$$

De même $\frac{1}{j2\pi f}$ correspond à un intégration.
Dans la mesure où la dérivée d'un échelon est un Dirac, on aboutit à une propriété de la transformée de Fourier de l'échelon :

$$j2\pi f\mathrm{TF}\left[\mathbb{H}(t)\right](f)=\mathrm{TF}\left[\frac{d}{dt}\mathbb{H}(t)\right](f)=\mathrm{TF}\left[\delta(t)\right](f)=1 \tag{5.9}$$

La transformée de Fourier de l'échelon n'est pour autant pas définie.
On rajoute à la table 3.1,

$j2\pi f \quad\Leftrightarrow\quad \frac{d}{dt}$	$\frac{1}{j2\pi f} \quad\Leftrightarrow\quad \int_{-\infty}^{t} d\tau$
$j2\pi f\mathrm{TF}\left[\mathbb{H}(t)\right](f)=1$	

Table 5.1: additionnelle de transformées de Fourier

D'un point de vue mathématique :

Formellement identifier un $j2\pi f\text{TF}[x(t)](f)$ avec $\text{TF}\left[\frac{d}{dt}x(t)\right](f)$ pose de nombreux problèmes théoriques, le cadre généralement utilisé est la transformée de Laplace qui est définie seulement pour $t \geq 0$. Un autre cadre théorique est celui des distributions tempérées. La présentation qui est faite ici est donc une simplification qui sans surprise pose des problèmes numériques.

D'un point de vue électronique :

Les transformées de Fourier de $v(t)$ et $i(t)$ sont notées ici $\widehat{V}(f)$ et $\widehat{I}(f)$. L'impédance généralise la relation entre tension et résistance pour un condensateur et une bobine.

$$\widehat{V}(f) = Z(f)\widehat{I}(f) \tag{5.10}$$

Ainsi pour un condensateur,

$$\widehat{V}(f) = \frac{1}{Cj2\pi f}\widehat{I}(f) \quad \Leftrightarrow \quad \frac{d}{dt}v(t) = \frac{1}{C}i(t) \tag{5.11}$$

et pour une bobine,

$$\widehat{V}(f) = Lj2\pi f\widehat{I}(f) \quad \Leftrightarrow \quad v(t) = L\frac{d}{dt}i(t) \tag{5.12}$$

On peut utiliser les impédances et la notion de transformée de Fourier pour trouver la transformée de Fourier de signaux à partir de certains montages électroniques.

Exemple pour illustrer :

L'utilisation de la transformée de Fourier permet de retrouver l'équation (2.8) à partir du circuit 2.2 en considérant $u_1(t) = [\![t \geq 0]\!]$.

- En utilisant la notion de diviseur de tension, on trouve

$$\widehat{X}(f) = \frac{\mathtt{R}}{\mathtt{R} + \frac{1}{j\mathtt{C}2\pi f}}\widehat{U}_1(f) = \frac{j\mathtt{RC}2\pi f}{j\mathtt{RC}2\pi f + 1}\widehat{U}_1(f) \tag{5.13}$$

- D'un point de vue théorique, l'entrée vérifie $j2\pi f\widehat{U}_1(f) = 1$ d'après (5.9). Aussi

$$\widehat{X}(f) = \frac{\mathtt{RC}}{j\mathtt{RC}2\pi f + 1} = \frac{1}{j2\pi f + 1} \text{ en choisissant } \mathtt{RC} = 1 \tag{5.14}$$

  En utilisant la table 3.1, on obtient finalement

$$x(t) = [\![t \geq 0]\!]e^{-t} \tag{5.15}$$

- D'un point de vue numérique la figure 3.2 montre qu'il est difficile, (mais apparemment possible) de trouver numériquement $x(t)$ à partir de $\widehat{X}(f)$. Remarquez que cette figure ressemble effectivement à la figure 2.2 obtenue avec l'équation (5.15).

5.4 Bref rappel pour résoudre analytiquement des équations différentielles à coefficients constant

Cours signal et bruit :

Considérant une équation différentielle définie par

$$a_0\frac{d^N}{dt^N}y(t) + a_1\frac{d^{N-1}}{dt^{N-1}}y(t) + \ldots + a_N y(t) = \delta(t) \tag{5.16}$$

On définit le polynôme de degré N

$$Q(p) = a_0 p^N + a_1 p^{N-1} + \ldots + a_0 \tag{5.17}$$

On cherche les N racines de ce polynôme. On appelle racine les complexes $z_0 \ldots z_{N-1}$ qui vérifient $Q(z_n) = 0$. On appelle degré d'un polynôme, l'exposant le plus élevé utilisé pour définir le polynôme.
La solution de l'équation différentielle est alors de la forme

$$y(t) = \sum_{n=0}^{N-1} c_n e^{jz_n t} [\![t \geq 0]\!](t) \tag{5.18}$$

où $c_0, c_1, \ldots, c_{N-1}$ sont N complexes à déterminer de façon que l'équation différentielle soit juste.
En général $y(t)$ ou certaines de ses dérivées présente des discontinuités et il convient d'appliquer le cours de la section 4.3. Parfois à force de dériver on se retrouve à dériver un Dirac, mais il convient de ne pas faire de calculs avec ces dérivées de Dirac, il suffit de les constater et d'en déduire que les coefficients devant sont nuls.

D'un point de vue mathématique :

Cette présentation simplifiée masque quelques difficultés qui ne font pas l'objet du cours.

- Il n'y a pas de formules générales pour trouver les racines quand Q est un polynôme de degré élevé.
- Comme le polynôme est en fait à valeurs réelles, les racines quand elles ne sont pas réelles, elles vont par paires, l'une étant le conjugué de l'autre.
- Quand il y a moins de racines que le degré du polynôme Q, certaines racines ont une multiplicité plus grande que 1. La multiplicité est un indice entier strictement positif associé à chaque racine z_m notée ν_m. Elle est définie par le fait qu'on peut alors écrire $Q(p)$ sous la forme

$$Q(p) = C(p - z_0)^{\nu_0}(p - z_1)^{\nu_1} \ldots (p - z_M)^{\nu_M} \text{ avec } \sum_{m=0}^{M-1} \nu_m = N \text{ et } C \text{ est une constante} \tag{5.19}$$

  Dans ce cas l'équation (5.18) doit alors être modifiée, au lieu de c_n on met $C_m(t)$ un polynôme de degré $\nu_m - 1$ à déterminer.

Exemple pour illustrer :

Considérons l'équation différentielle

$$\frac{d^2}{dt^2} y(t) + y(t) = \delta(t) \tag{5.20}$$

Le polynôme associé est $Q(p) = p^2 + 1$ dont les racines sont $p = j$ et $p = -j$, ce sont des racines simples ($Q(p) = (p + j)(p - j)$). Donc il existe A et B tel que

$$y(t) = (Ae^{jt} + Be^{-jt})[\![t \geq 0]\!](t) \tag{5.21}$$

Cette courbe a une discontinuité en $t = 0$, aussi

$$\frac{d}{dt} y(t) = (A + B)\delta(t) + \left(jAe^{jt} - jBe^{-jt}\right)[\![t \geq 0]\!](t) \tag{5.22}$$

$\frac{d}{dt} y(t)$ a une discontinuité en $t = 0$ qui vaut $jA - jB$.

$$\frac{d^2}{dt^2} y(t) = (A + B)\delta'(t) + (jA - jB)\delta(t) + \left(j^2 Ae^{jt} + j^2 Be^{-jt}\right)[\![t \geq 0]\!](t) \tag{5.23}$$

J'annule $A + B$ parce que regroupant les termes devant $\delta'(t)$. En regroupant les autres termes, je trouve

$$\delta(t) = \frac{d^2}{dt^2} y(t) + y(t) = (jA - jB)\delta(t) \tag{5.24}$$

De sorte que A et B vérifie

$$\begin{cases} A+B=0 \\ jA-jB=1 \end{cases} \tag{5.25}$$

La solution du système est $A=-j/2=\frac{1}{2j}$ et $B=-\frac{1}{2j}$ et donc finalement

$$y(t)=\sin(t)[\![t\geq 0]\!](t) \tag{5.26}$$

5.5 Simulation numérique des solutions des équations différentielles

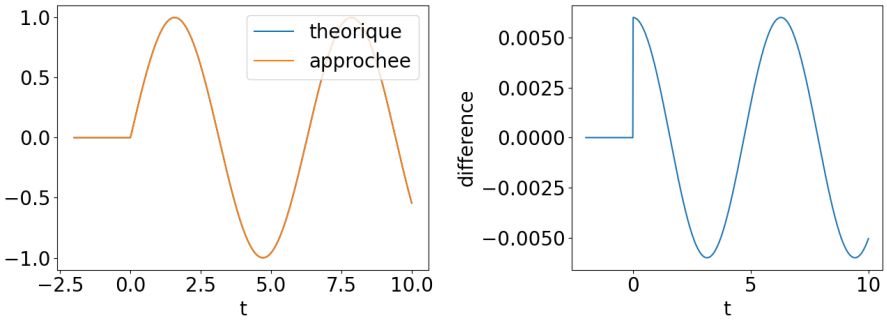

Figure 5.2: À gauche : courbe theorique et expérimentales. À droite : différences entre les deux courbes

Python :

Le module `seb.py` contient un programme `sol_eq_diff` qui permet de simuler numériquement la solution de l'équation différentielle. La première étape est de créer un vecteur contenant les différents instants où l'on veut calculer la solution.

```
t=np.linspace(t_debut,t_final,nombre_de_points)
```

Puis on définit `coef`, un `tuple` contenant les coefficients du polynôme

$$\texttt{coef}=(a_0,a_1,\ldots,a_N)$$

Enfin on calcule le vecteur `y` contenant les valeurs prises par $y(t)$ aux instants indiqués par `t`.

```
y=seb.sol_eq_diff(coef,t)
```

Exemple pour illustrer :

L'implémentation se fait en utilisant `sol_eq_diff` fourni dans `seb.py`. Les coefficients de l'équations différentielles (5.20) sont $1,0,1$.

```
t=np.linspace(-2,10,10**3)
y=seb.sol_eq_diff((1,0,1),t)
```

La figure 5.2 présente à gauche la solution théorique et approchée numériquement. À droite, il s'agit de la différence entre ces deux courbes. L'implémentation est faite dans A.20.

Chapter 6

Filtres et effet mémoire

6.1 Notion de filtre et de relation entrée-sortie

Cours signal et bruit :

Généralement l'entrée est notée $x(t)$ (ou $e(t)$) et la sortie est $y(t)$ (ou $s(t)$). On note les sorties associées à un signal

$$y(t) = \mathcal{H}\left[x(t)\right](t) \tag{6.1}$$

D'un point de vue électronique :

Un filtre généralise la relation entrée-sortie à des signaux dépendant du temps. Les figures 2.1, 2.2 et 2.3 peuvent se voir comme des filtres où l'entrée est $U(t)$ et la sortie est $x(t)$. Mais on pourrait aussi considérer que l'entrée est une intensité et la sortie pourrait aussi être une sortie. Les formules en électronique pour déterminer la tension ou l'intensité n'utilisent pas cette notion d'entrée-sortie par exemple pour les quadripôles, cela peut être l'entrée qui s'exprime en fonction de la sortie (cf : annexe B.4, page 93), parce que l'intention est de faire du calcul. En revanche il est fréquent que pour un circuit donné, on ne puisse intervertir entrée et sortie, par exemple on ne peut imposer la tension de sortie d'un amplificateur opérationnel et lire la tension à son entrée.

Cours signal et bruit :

La définition d'un filtre linéaire est plus complexe que l'équation (1.3) pour une relation entre deux grandeurs.
Un filtre est linéaire si et seulement si étant donnée α, β, $x_1(t)$ et $x_2(t)$ on a

$$\boxed{x(t) = \alpha x_1(t) + \beta x_2(t) \quad \Rightarrow \quad y(t) = \alpha y_1(t) + \beta y_2(t)} \tag{6.2}$$

où $y_1(t)$, $y_2(t)$ et $y(t)$ sont les sorties associées à $x_1(t)$, $x_2(t)$ et $x(t)$.
On peut écrire l'équation (6.2) ainsi

$$\mathcal{H}\left[\alpha x_1(t) + \beta x_2\right](t) = \alpha \mathcal{H}\left[x_1(t)\right](t) + \beta \mathcal{H}\left[x_2(t)\right](t) \tag{6.3}$$

D'un point de vue électronique :

La question de la linéarité se pose peu dans ce cours parce que les exemples considérés sont le plus souvent linéaires. Ce cours ne s'applique que pour des systèmes linéaires. Et dans la pratiques il y a de nombreux systèmes pour lesquels le fonctionnement linéaire est en fait une approximation.
On peut utiliser les simulations numériques pour vérifier si un système est linéaire. Considérons un filtre défini par $\mathcal{H}_1[x(t)](t) = \int_{-\infty}^{t} x(\tau)\,d\tau$ et un deuxième filtre défini par $\mathcal{H}_2[x(t)](t) = \int_{-\infty}^{t} x^2(\tau)\,d\tau$ qui sont implémentés par ces deux fonctions

```
def H1(t,x):
  return seb.integrer(t,x)
def H2(t,x):
  return H1(t,x**2)
```

On génère aléatoirement α et β suivant une loi gaussienne de moyenne nulle et d'écart-type 1.

```
alpha, beta = np.random.normal(0,1,2)
```

On considère une échelle de temps `t`, une fonction porte et une fonction triangle en entrée et on vérifie que l'équation (6.2).

```
t = np.linspace(-3,3,10**3)
x1, x2 = seb.fonction_P(t), seb.fonction_T(t)
assert all(np.abs(H1(t,alpha*x1+beta*x2)-alpha*H1(t,x1)-beta*H1(t,x2))<1e-8)
```

Quand on applique le même test à `H2`, cela ne fonctionne plus.

```
assert all(np.abs(H2(t,alpha*x1+beta*x2)-alpha*H2(t,x1)-beta*H1(t,x2))<1e-8)
```

6.2 Présence d'un effet mémoire dans la relation entrée-sortie

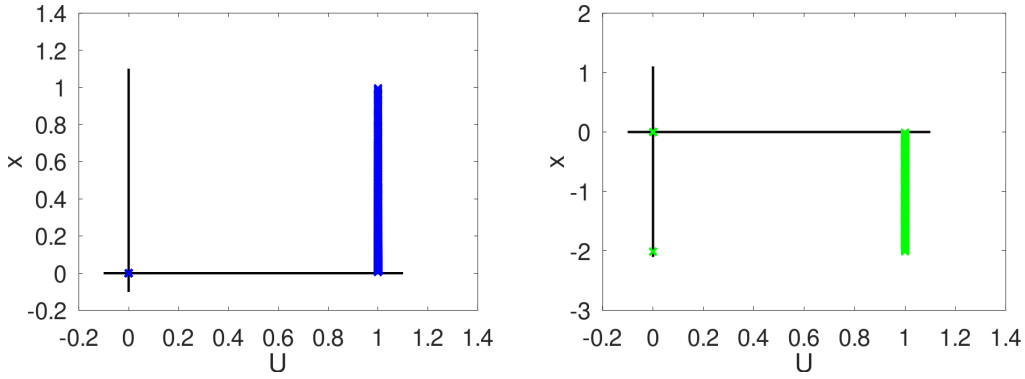

Figure 6.1: Graphe visualisant la sortie $x(t)$ en fonction de l'entrée $u(t)$ pour les montages 2.2 et 2.3.

La figure 6.1 représente les graphes des valeurs des sorties en fonction des valeurs des entrées pour les montages décrits par les figures 2.3 et 2.2. Ce qui montre qu'il y a un effet mémoire, c'est que ce n'est pas une courbe au sens où pour chaque position sur l'axe des abscisse, il n'y a pas qu'une unique position sur l'axe des ordonnées.

6.3 Filtre temps invariant

Cours signal et bruit :

On dit qu'un filtre est temps invariant lorsqu'une entrée retardée produit une sortie retardée.

$$x_2(t) = x_1(t - t_0) \quad \Rightarrow \quad y_2(t) = y_1(t - t_0) \tag{6.4}$$

Lorsqu'un filtre est **linéaire** et **temps invariant** alors on peut le décrire par son comportement fréquentiel et donc parler d'une réponse impulsionnelle et d'une réponse fréquentielle.

D'un point de vue électronique :

La propriété de temps-invariance est généralement vérifié dans les montages électriques. Ce n'est pas le cas si par exemple des composants dépendent fortement de la température et que cette température est influencée par un phénomène extérieur.

Python :

On peut aussi tester par simulation pour chercher à savoir si un filtre est temps invariant. On considère ici deux filtres $\mathcal{H}_1\left[x(t)\right](t) = \int_{-\infty}^{t} x(\tau)\,d\tau$ et $\mathcal{H}_2\left[x(t)\right](t) = \int_{-\infty}^{t} \tau x(\tau)\,d\tau$ qui sont implémentées avec deux fonctions

```
def H1(t,x):
  return seb.integrer(t,x)
def H2(t,x):
  return H1(t,t*x)
```

On génère une échelle de temps et un signal, ici la fonction porte.

```
t = np.linspace(-3,3,10**3)
x = seb.fonction_P(t)
```

On retarde de τ où celui-ci est tiré aléatoirement suivant une loi uniforme

```
tau = np.random.uniform(-1,1)
```

Et on teste l'équation (6.4) pour $\mathcal{H}_1$.

```
assert all(np.abs(H1(t,seb.retarder(t,x,tau))-seb.retarder(t,H1(t,x),tau))<1e-8)
```

En revanche cela ne fonctionne pas pour $\mathcal{H}_2$.

```
assert all(np.abs(H2(t,seb.retarder(t,x,tau))-seb.retarder(t,H2(t,x),tau))<1e-8)
```

6.4 Réponse impulsionnelle

Cours signal et bruit :

On peut définir la réponse impulsionnelle d'un filtre à partir de la réponse fréquentielle

$$h(t) = \mathrm{TF}^{-1}[\widehat{H}(f)](t) \tag{6.5}$$

Python :

On peut utiliser l'équation (6.5) pour justement simuler cette réponse impulsionnelle, (dans la mesure où intégrer ces réponse fréquentielle ne pose pas trop de problèmes).
Considérons l'exemple de $\widehat{H}(f) = \frac{1}{1+j2\pi f}$.
On se donne tout d'abord une échelle en fréquence décrite finement

```
f = np.linspace(-30,30,10**4)
```

puis on calcule la réponse fréquentielle

```
H = 1/(1+1j*2*np.pi*f)
```

On se donne une échelle en temps pour faire de l'affichage

```
t = np.linspace(-1,3,10**2)
```

Et on calcule la réponse impulsionnelle en prenant la partie réelle de l'équation (6.5).

```
h = np.real(seb.TFI(f,H,t))
```

On mesure la précision de la simulation qui ici est de 0.04.

```
err = np.max(np.abs(h-np.exp(-t)*(t>=0)))
print(f"err max = {err}")
```

6.5 Produit de convolution

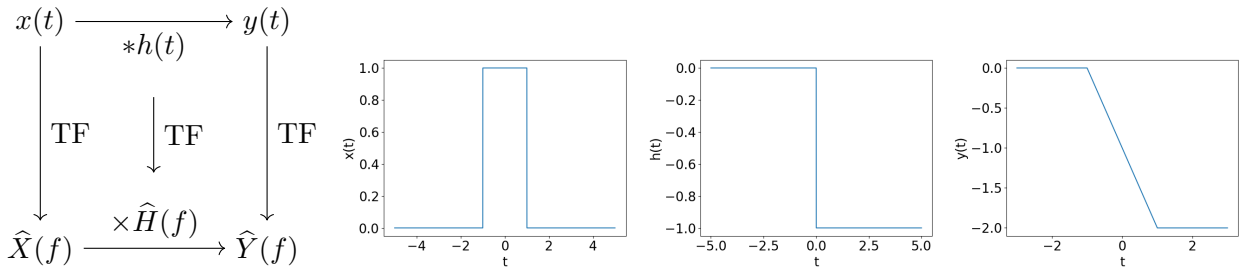

Figure 6.2: À gauche : diagramme filtre et réponse fréquentielle. À droite : simulation d'un filtre déjà illustré sur la figure 2.3 en utilisant la flèche du milieu dans le sens inverse l'algorithme 9.

Cours signal et bruit :

On définit un produit de convolution entre deux signaux par

$$z(t) = x(t) * y(t) = \int_{-\infty}^{+\infty} x(\tau)y(t-\tau)\,d\tau = \int_{-\infty}^{+\infty} x(t-\tau)y(\tau)\,d\tau \tag{6.6}$$

Une propriété importante est ce qui est un produit de convolution dans le domaine temporel devient un simple produit dans le domaine fréquentiel

$$\widehat{Z}(f) = \widehat{X}(f)\widehat{Y}(f) \tag{6.7}$$

Cela donne une relation entre la réponse impulsionnelle et la réponse fréquentielle

$$\widehat{H}(f) = \mathrm{TF}\left[h(t)\right](f) \tag{6.8}$$

Cela donne aussi une deuxième façon d'écrire la relation entrée-sortie en utilisant la définition du produit de convolution (utilisation de l'équation (7.1)).

$$y(t) = h(t) * x(t) = \int_{-\infty}^{+\infty} h(t-\tau)x(\tau)\,d\tau \tag{6.9}$$

En termes numériques :

Algorithm 9: pour la figure 6.2

Créer une échelle de temps `tx` pour $x(t) = [\![t \in [-1, 1]]\!](t)$
Créer une échelle de temps `ty` pour $y(t)$
Déterminer les valeurs de `x` avec `tx`
Déterminer les valeurs de `h` avec `ty`, sachant que $h(t) = [\![t \geq 0]\!]$
Calculer le produit de convolution $y(t) = x(t) * h(t)$

En terme de visualisation :

La droite de la figure 6.2 montre le signal d'entrée et de sortie. L'implémentation est faite dans A.12.
Le produit de convolution est implémenté avec la fonction `seb.convolution`. Cette fonction utilise le premier signal avec son échelle de temps, le deuxième signal et son échelle de temps ainsi que l'échelle de temps du signal résultant. La particularité de cette fonction est qu'elle requière que les échelles de temps aient toutes la même résolution, c'est-à-dire la même période d'échantillonnage. Ainsi si on dispose d'une certaine échelle de temps `tx` pour un signal `x`, qu'on a une fonction `h(t)` qui transforme une échelle de temps en la réponse impulsionnelle, et que cette réponse impulsionnelle est décrite correctement sur l'intervalle $[0, 10]$, on peut implémenter $y(t) = x(t) * h(t)$ avec les étapes suivantes.

- Récupération de la période d'échantillonnage de `tx`

```
Te = tx[1]-tx[0]
```

- Fabrication de l'échelle de temps pour $h(t)$

  ```
  th = np.arange(0,10,Te)
  ```

- Calcul du produit de convolution

  ```
  y=seb.convolution(tx,x,th,h(th),tx)
  ```

Chapter 7

Description fréquentielle des filtres

7.1 Réponse fréquentielle

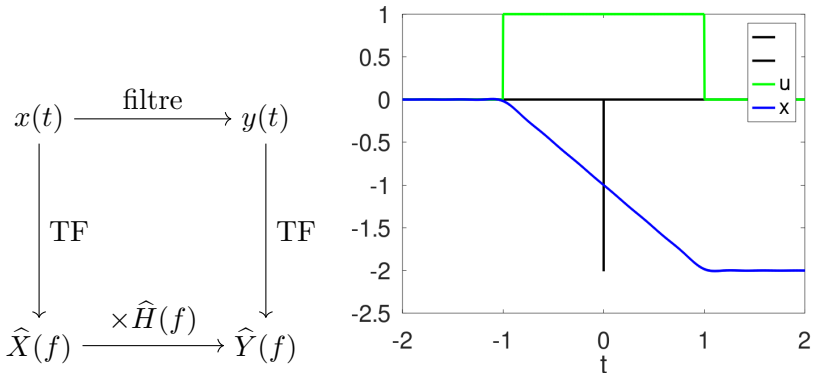

Figure 7.1: À gauche : diagramme filtre et réponse fréquentielle. À droite : simulation d'un filtre déjà illustré sur la figure 2.3 en utilisant l'algorithme 10 (p. 44).

Cours signal et bruit :

Comme illustré sur la figure 7.1, un filtre qui est linéaire et temps invariant peut se mettre sous la forme

$$\widehat{Y}(f) = \widehat{H}(f)\widehat{X}(f) \tag{7.1}$$

$\widehat{H}(f)$ est la réponse fréquentielle du filtre.

D'un point de vue électronique :

On peut voir la relation entre l'intensité qui le traverse et la tension autour du dipôle comme la relation entrée-sortie d'un filtre.

- Pour une résistance, $\widehat{U}(f) = \mathtt{R}\widehat{I}(f)$ et $\widehat{\mathtt{Z}}(f) = \mathtt{R}$ est la réponse fréquentielle.
- Pour un condensateur, $\widehat{U}(f) = \frac{1}{j\mathtt{C}2\pi f}\widehat{I}(f)$ et $\widehat{\mathtt{Z}}(f) = \frac{1}{j\mathtt{C}2\pi f}$ est la réponse fréquentielle.
- Pour un bobine, $\widehat{U}(f) = j\mathtt{L}2\pi f\widehat{I}(f)$ et $\widehat{\mathtt{Z}}(f) = j\mathtt{L}2\pi f$ est la réponse fréquentielle.

En termes numériques :

Je considère le montage présenté sur la figure 2.3 avec l'objectif de retrouver numériquement les courbes à droite en utilisant la réponse fréquentielle. Le calcul indiqué par le diagramme à gauche de la figure 7.1 est

$$x(t) = \mathrm{TF}^{-1}\left[\frac{1}{j2\pi fRC}\mathrm{TF}[u(t)](f)\right](t) \tag{7.2}$$

La simulation utilise l'algorithme 10. Son implémentation est dans la figure A.11.

Algorithm 10 Algorithme de la figure 7.1

Créer une échelle de temps `t`
Créer une échelle de fréquence `f`
Retirer la valeur 0 de l'échelle de fréquence.
Calculer $\widehat{U}(f) = \mathrm{TF}[u(t)](f)$
Calculer $\widehat{X}(f) = \frac{1}{j2\pi fRC}\widehat{U}(f)$
Calculer $x(t) = \mathrm{TF}^{-1}\left[\widehat{X}(f)\right](f)$
Retrancher la première valeur de $x(t)$ au vecteur `x`

7.2 Filtres passe-bas, passe-haut et passe-bande

Cours signal et bruit :

À partir de la visualisation du graphe du module de la réponse fréquentielle, on définit trois types de filtres.

- Un filtre passe-bas a une réponse fréquentielle plus élevée sur les basses fréquences.
- Un filtre passe-bande a une réponse fréquentielle plus élevée sur des fréquences intermédiaires.
- Un filtre passe-haut a une réponse fréquentielle plus élevée sur des fréquences élevées.

Les filtres peuvent tout à fait n'être ni passe-bas, ni passe-bande, ni passe-haut.

D'un point de vue mathématique :

Le calcul des limites du module de la réponse fréquentielle en 0 ou en $+\infty$ peut servir d'indice pour dire si un filtre est susceptible d'être d'un de ces trois types.

7.3 Fréquence de coupure et bande passante d'un filtre

- Un filtre passe-bas vérifie a priori $\lim_{f\to 0}|\widehat{H}(f)| \neq 0$ et $\lim_{f\to+\infty}|\widehat{H}(f)| \approx 0$
- Un filtre passe-bande vérifie a priori $\lim_{f\to 0}|\widehat{H}(f)| \approx 0$ et $\lim_{f\to+\infty}|\widehat{H}(f)| \approx 0$
- Un filtre passe-haut vérifie a priori $\lim_{f\to 0}|\widehat{H}(f)| \approx 0$ et $\lim_{f\to+\infty}|\widehat{H}(f)| \neq 0$

Généralement une réponse fréquentielle se présente comme le quotient de deux polynômes en f à coefficients complexes, donc de cette forme-là

$$\widehat{H}(f) = \frac{b_0 f^M + b_1 f^{M-1} + \ldots + b_M}{a_0 f^N + a_1 f^{N-1} + \ldots + a_N} \tag{7.3}$$

Pour trouver la limite lorsque f tend vers 0, il suffit de remplacer f par 0 si cela conduit à une division par zéro. Mais dans ce cas on met en facteur les termes en f de façon à pouvoir conclure.
Pour trouver la limite lorsque f tend vers $+\infty$, on ne peut pas procéder de la même façon parce que dans le domaine complexe il y a plein de façon de tendre vers l'infini, suivant l'argument du complexe. Par contre on met des termes en f en facteur de façon à ce que les autres termes tendent vers une constante.

Exemple pour illustrer :

Je considère la réponse fréquentielle $\widehat{H}(f) = \frac{jf}{1-f^2+jf}$.
La limite en $f = 0$ est

$$\lim_{f\to 0}\left|\widehat{H}(f)\right| = \lim_{f\to 0}|f|\left|\frac{j}{1-f^2+jf}\right| = 0 \tag{7.4}$$

La limite en $f \to +\infty$ est

$$\lim_{f\to+\infty}\left|\widehat{H}(f)\right| = \lim_{f\to+\infty}\left|\frac{1}{f}\right|\left|\frac{j}{\frac{1}{f^2}-1+\frac{j}{f}}\right| = 0 \tag{7.5}$$

Cours signal et bruit :

La fréquence de coupure est définie par rapport à la valeur maximale du module de la réponse fréquentielle. Dans ce cours, on note $f_{\max}$, la fréquence à laquelle le module est maximale.

$$\widehat{H}(f_{\max}) = \max_f \left|\widehat{H}(f)\right| \tag{7.6}$$

- Pour un filtre passe-bas, $f_{\max} = 0$ et il y a une fréquence de coupure f_1

$$\left|\widehat{H}(f_1)\right| = \frac{\sqrt{2}}{2}\left|\widehat{H}(0)\right| \tag{7.7}$$

- Pour un filtre passe-bande, $f_{\max} = 0$ et il y a deux fréquences de coupure f_1 et f_2

$$\left|\widehat{H}(f_1)\right| = \left|\widehat{H}(f_2)\right| = \frac{\sqrt{2}}{2}\left|\widehat{H}(0)\right| \text{ et } f_1 \leq f_{\max} \leq f_2 \tag{7.8}$$

- Pour un filtre passe-haut, $f_{\max} = 0$ et il y a une fréquence de coupure f_1

$$\left|\widehat{H}(f_1)\right| = \frac{\sqrt{2}}{2}\lim_{f\to+\infty}\left|\widehat{H}(f)\right| \tag{7.9}$$

La bande passante d'un filtre est définie comme un intervalle $[f_1, f_2]$. Parfois on utilise le terme de bande passante pour désigner la longueur de cette intervalle : $f_2 - f_1$.
Le Facteur de qualité, Q est définie à partir de la bande passante et de fréquence maximale.

$$Q = \frac{f_{\max}}{f_2 - f_1} \tag{7.10}$$

Python :

Numériquement il est possible d'évaluer cette bande fréquence.
Considérons l'exemple de réponse fréquentielle

$$\widehat{H}(f) = \frac{j2\pi f}{1 - 4\pi^2 f^2 + \frac{2}{10}j\pi f} \tag{7.11}$$

On cherche tout d'abord la fréquence donnant le module le plus élevé en créant une échelle de fréquence et en récupérant la valeur la plus élevée de ce module.

```
f = np.linspace(0,10,10**5)
H = 1j*2*np.pi*f/(1-4*(np.pi**2)*f*f+1j*2*np.pi*f/10)
ind_max = np.argmax(np.abs(H))
f_max, H_max = f[ind_max], H[ind_max]
print(f"f_max = {f_max} H_max = {H_max}")
```

Pour cet exemple il se trouve que $|\widehat{H}(f)|$ est croissant entre 0 et $f_{\max}$ et décroissant après

```
assert all(np.diff(np.abs(H[:ind_max]))>=0)
assert all(np.diff(np.abs(H[ind_max:]))<=0)
```

Pour trouver les fréquences f_1 et f_2 qui sont respectivement dans les intervalles $[0, f_{\max}]$ et $[f_{\max}, +\infty[$ et qui vérifient approximativement $|\widehat{H}(f_1)| = |\widehat{H}(f_2)| = \frac{\sqrt{2}}{2}|H(f_{\max})|$, on peut utiliser la fonction `seb.where_nearest` qui trouve l'indice dans un vecteur dont la composante est la plus proche d'une valeur cible.

```
ind1 = seb.where_nearest(np.abs(H[:ind_max]),np.abs(H_max)/np.sqrt(2))
ind2 = ind_max + seb.where_nearest(np.abs(H[ind_max:]),np.abs(H_max)/np.sqrt(2))
```

Finalement on en déduit la bande de fréquence et le facteur de qualité.

```
B,Q = f[ind2]-f[ind1], f_max/(f[ind2]-f[ind1])
print(f"B={B}, Q={Q}")
```

Chapter 8

Signaux périodiques

8.1 Définition d'un signal périodique

Cours signal et bruit :

Un signal périodique est un signal qui se répète dans le temps

$$x(t) = x(t + T_x) \tag{8.1}$$

T_x est la période du signal. Sur un graphique du signal en fonction du temps, ce signal déplacé vers la gauche est identique à lui-même et on appelle la période la longueur du déplacement horizontal effectué.

D'un point de vue électronique :

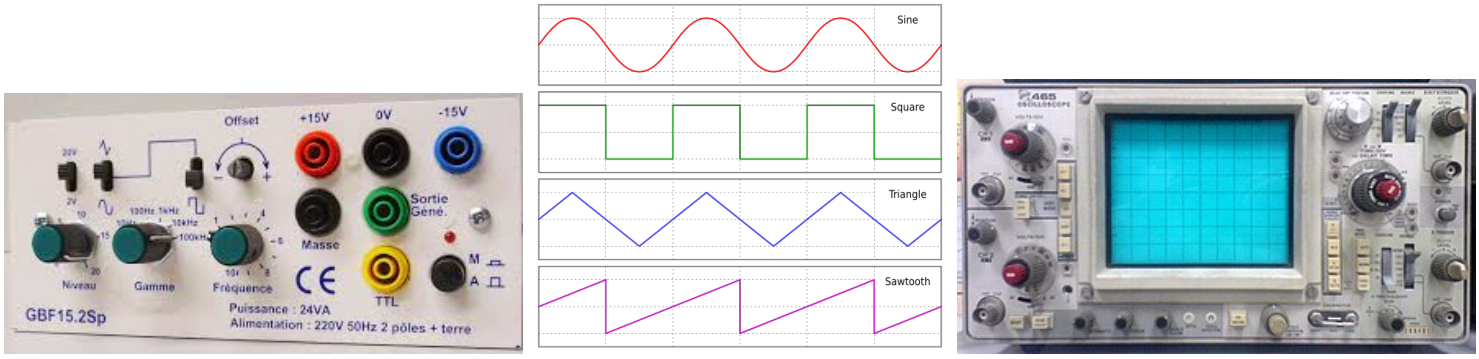

Figure 8.1: Générateur de basse fréquences, signaux et oscilloscope

La figure 8.1 montre à gauche un générateur de basses fréquences qui peut générer des signaux avec une tension qui varie de façon sinusoïdale, triangulaire ou carrée visualisés au milieu. Ces signaux sont périodiques. Il est en général possible de leur ajouter un signal constant et de modifier l'amplitude ou la période. Sur le GBF la fréquence désigne en fait l'inverse de cette période. L'oscilloscope permet de visualiser le graphe de la tension en fonction du temps. C'est adapté aux signaux périodiques, la valeur 0 du temps indiqué correspond non pas à $t = 0$ mais au dépassement d'une valeur seuil par la tension. Il n'est donc pas possible en général de lire φ dans $v(t) = \sin(2\pi f_0 t + \varphi)$ en visualisant un signal avec un oscilloscope.

Cours signal et bruit :

$$x(t) = \sin(2\pi f_0 t + \varphi) \text{ et } x(t) = \cos(2\pi f_0 t + \varphi) \text{ sont périodiques de période } \boxed{T_x = \frac{1}{f_0}} \tag{8.2}$$

On décrit un signal périodique en indiquant qu'il est périodique de période T et en décrivant ce signal sur l'intervalle $[0, T]$.

8.2 Somme, moyenne, énergie et puissance

Cours signal et bruit :

La somme est infinie, la moyenne se calcule sur une période.

$$\mathrm{S}_x = +\infty \text{ et } \mathrm{M}_x = \frac{1}{T_x}\int_0^{T_x} x(t)\,dt \tag{8.3}$$

L'énergie est infinie et la puissance se calcule sur une période

$$\mathrm{E}_x = +\infty \text{ et } \mathrm{P}_x = \frac{1}{T_x}\int_0^{T_x} x^2(t)\,dt \tag{8.4}$$

D'un point de vue électronique :

Pour un signal périodique de période T, correspondant à une tension,

- l'amplitude crête à crête est définie par $\max_{t\in[0,T]} v(t) - \min_{t\in[0,T]} v(t)$.
- La tension efficace ou la tension en moyenne quadratique est définie par $\sqrt{\frac{1}{T}\int_0^T v^2(t)\,dt}$ ce qui correspond à $\sqrt{P_v}$.

En termes numériques :

On observe que la définition de la moyenne et de la puissance coïncident avec la définition utilisant une limite

$$\mathrm{M}_x = \lim_{T\to+\infty} \frac{1}{T}\int_{-T/2}^{T/2} x(t)\,dt \text{ et } \mathrm{P}_x = \lim_{T\to+\infty} \frac{1}{T}\int_{-T/2}^{T/2} x^2(t)\,dt \tag{8.5}$$

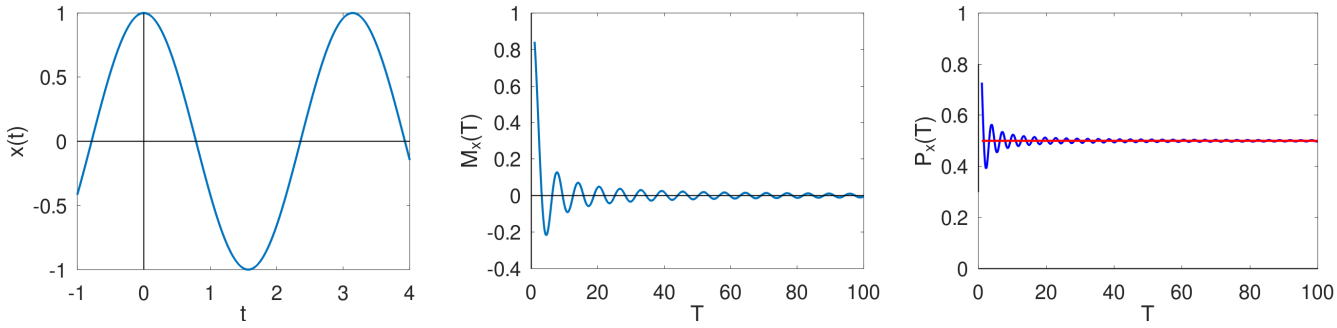

Figure 8.2: Signal $x(t) = cos(2t)$ et moyenne et puissance de $x(t)$ évaluées entre $-T/2$ et $T/2$ en fonction de T.

Python :

On vérifie tout d'abord que le signal construit `t,x` commence bien en $t = 0$.

```
assert x[seb.where_nearest(t,0)]==x[0]
```

`where_nearest` est une fonction que j'ai programmé et placé dans `seb` qui donne l'indice dont la valeur du vecteur ici `t` est la plus proche de la valeur ici 0.
On vérifie ce signal est périodique de période T (du moins que $x(T) = x(0)$).

```
assert x[seb.where_nearest(t,T)]==x[0]
```

On calcule le nombre de composantes correspondant à une période :

```
N=seb.where_nearest(t,T)
```

On calcule la moyenne

`print(f"moyenne={np.mean(x[:N]):.2f}")`

cette instruction utilise les notions suivantes :

- `x[:N]` est un vecteur de N composantes, 0 jusqu'à $N-1$.
- `np.mean(x[:N])` effectue le calcul de la moyenne.
- `f"moyenne=np.mean(x[:N]):.2f"` fabrique une chaîne de caractères dont l'expressoin entre accolades est évaluées et affichées avec deux chiffres significatifs. Attention au `f` précédent l'accolade.
- `print` est l'instruction affichant cette chaîne de caractères à l'écran.

L'implémentation est faite dans A.15.

8.3 Transformée de Fourier

D'un point de vue électronique :

Figure 8.3: Analyseur de spectre

La figure 8.3 montre un analyseur de spectre. L'échelle des abscisses varie entre `fréquence centrale`-$\frac{1}{2}$`span` et `fréquence centrale`+$\frac{1}{2}$`span`.
Pour des raisons technologiques, il est nécessaire de préciser la largeur de bande du filtre de résolution. Plus cette largeur est faible, plus le `sweep time` est long pour faire la mesure et tracer la figure. Ce filtre de résolution transforme un pic en un signal avec une certaine largeur égale à la largeur de bande du filtre. Ceci n'est donc pas un filtre au sens du cours.
L'axe vertical est soit la tension en `mV` ou en `dBmV`, ou la puissance en `mW` ou en `dBm`. En ce qui concerne la puissance, elle est en réalité déduite de la tension en tenant compte de la résistance équivalente de l'analyseur de spectre vue de l'entrée, celle-ci est donc proportionnelle au carré de la tension. Les unités `dBm` et `dBmV` correspondent à des échelles logarithmiques.

$$U_{\texttt{dBmV}} = 10\log_{10}(U_{\texttt{mV}}) \text{ et } P_{\texttt{dBm}} = 10\log_{10}(P_{\texttt{mW}}) \tag{8.6}$$

Cours signal et bruit :

La transformée de Fourier d'un signal temps continu et périodique est toujours un ensemble de raies espacées de $\frac{1}{T}$ (ou u

Les fréquences considérées sont $\boxed{f_k = \frac{k}{T}}$, leur unité est le Hertz (Hz), k est un entier positif ou négatif.
Transformée de Fourier pour un signal temps continu périodique

$$\widehat{X}_k = \frac{1}{T}\int_{-\frac{T}{2}}^{+\frac{T}{2}} x(t)e^{-j2\pi k\frac{t}{T}}\,dt \tag{8.7}$$

Transformée de Fourier inverse d'un spectre défini sur toutes les fréquences et non-périodique

$$x(t) = \sum_{k=-\infty}^{+\infty} \widehat{X}_k e^{\frac{j2\pi kt}{T}} \tag{8.8}$$

On peut noter la transformée de Fourier d'un signal périodique au moyen de la distribution $\delta(f)$.

$$\widehat{X}(f) = \sum_{k=-\infty}^{+\infty} \widehat{X}_k \delta\left(f - \frac{k}{T}\right) \tag{8.9}$$

Cette écriture n'apporte aucune information supplémentaire autre que $\widehat{X}_k$ et $f_k = \frac{k}{T}$.

D'un point de vue mathématique :

On appelle $\widehat{X}_k$ les coefficients de la série de Fourier. La façon dont avec l'équation (8.8), les coefficients $\widehat{X}_k$ permettent de retrouver $x(t)$ est appelé la série de Fourier.

Cours signal et bruit :

La définition de la transformée de Fourier adaptée aux signaux périodiques permet de compléter la table 3.1.

- $\text{TF}\left[e^{j2\pi f_0 t}\right](f) = \delta(f - f_0)$
- $\text{TF}^{-1}\left[\delta(f - f_0)\right](t) = e^{j2\pi f_0 t}$

D'un point de vue mathématique :

Attention quand on écrit $\text{TF}\left[e^{j2\pi f_0 t}\right](f)$ en remplaçant TF par une intégrale de $-\infty$ à $+\infty$, on a une expression qui n'a aucun sens (du moins pas au sens des théories classiques pour l'intégration). Dans ce cours le sens donné à cette expression se limite au fait que si on connaît une distribution (en l'occurence $\delta(f - f_0)$), et qu'on luit applique la transformée de Fourier inverse avec son expression intégrale on aboutit bien à $e^{j2\pi f_0 t}$.

Cours signal et bruit :

Tout signal périodique est la somme de sinusoïdes pondérées et déphasées.
Le calcul de ces sinusoïdes se fait à partir de l'équation (8.8).

$$x(t) = \widehat{X}_0 + 2\sum_{k=1}^{+\infty} \left|\widehat{X}_k\right| \cos\left(\frac{2\pi kt}{T} + \varphi_k\right) \text{ où } \varphi_k = \arg\left(\widehat{X}_k\right) \tag{8.10}$$

En effet en posant $\varphi_k = \arg\left(\widehat{X}_k\right)$ on a

$$\widehat{X}_k = \left|\widehat{X}_k\right| e^{j\varphi_k} \text{ et } \widehat{X}_{-k} = \overline{\widehat{X}_k} = \left|\widehat{X}_k\right| e^{-j\varphi_k} \tag{8.11}$$

L'équation (8.8) s'écrit alors

$$x(t) = \widehat{X}_0 + \sum_{k=1}^{+\infty} |\widehat{X}_k| \left(e^{\frac{j2\pi kt}{T} + j\varphi_k} + e^{-\frac{j2\pi kt}{T} - j\varphi_k}\right) \tag{8.12}$$

et finalement on trouve l'équation (8.10).

En termes numériques :

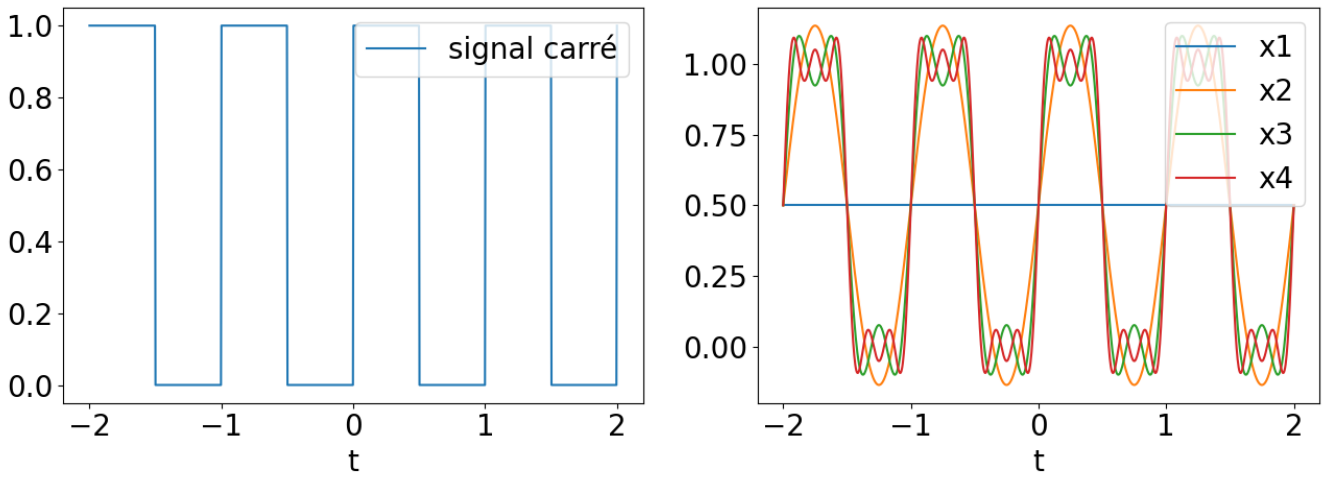

Figure 8.4: À gauche le signal carré périodique d'origine $x(t)$ et à droite sa reconstruction au moyen de sinusoïdes. L'implémentation est dans A.14.

La gauche de la figure 8.4 représente un signal périodique de période 1 qui vaut $\Pi(2t-0.5)$ sur l'intervalle $[0,1]$. Les coefficients de la série de Fourier sont

$$\widehat{X}_k = \int_0^1 x(t)e^{-j2\pi kt}\,dt = \int_0^{0.5} e^{-j2\pi kt}\,dt \tag{8.13}$$

- Si $k=0$, $\widehat{X}_0 = 0.5$.
- Si $k \neq 0$

$$\widehat{X}_k = \int_0^1 x(t)e^{-j2\pi kt}\,dt = \left[\frac{-e^{-j2\pi kt}}{j2\pi k}\right]_0^{0.5} = \frac{1-(-1)^k}{j2\pi k} \tag{8.14}$$

- Si $k \neq 0$ et k pair, $\widehat{X}_k = 0$
- Si k est impair, $\widehat{X}_k = \frac{1}{k\pi}e^{-j\pi/2}$

Finalement

$$x(t) = 0.5 + 2 \sum_{k \text{ impair et positif}} \cos\left(2\pi kt - \frac{\pi}{2}\right) \tag{8.15}$$

8.4 Périodisation

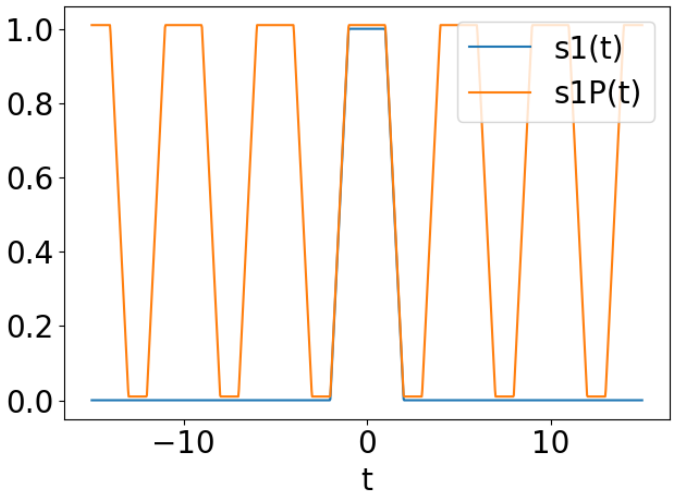

Figure 8.5: Signal $s_1(t)$ en bleu périodisé en $s_1^P(t)$ en répétant le motif défini sur $\left[-\frac{5}{2}, \frac{5}{2}\right]$.

Cours signal et bruit :

À partir de $x(t)$ un signal non-nul seulement dans l'intervalle $[0, T]$, on peut définir $x_P(t)$ un signal périodique de période T appelé périodisation de $x(t)$ en lui ajoutant des copies retardées de T, $2T$, etc... À partir de ce signal périodisé $x_P(t)$, on peut retrouver $x(t)$ par apériodisation en le multipliant par une porte sur $[0, T]$.

$$\begin{array}{ccc} x(t) & \xrightarrow{\text{périodisation}} & x_P(t) = \sum_{k=-\infty}^{+\infty} x(t - kT) \\ x(t) = x_P(t)[\![t \in [0, T]]\!](t) & \xleftarrow{\text{apériodisation}} & x_P(t) \end{array} \tag{8.16}$$

Dans le cadre de ce cours, on note aussi la périodisation

$$x_P(t) = \text{Périodiser}_{T_e}(x_t) \tag{8.17}$$

La figure 8.5 montre $s_1(t)$ et $s_1^P(t)$ périodisés. La courbe de $s_1^P(t)$ est légèrement surélevée pour montrer la différence avec $s_1(t)$. L'implémentation est dans A.21.

Python :

Pour représenter un signal périodique sur une échelle de temps `t`, au moyen d'une définition de ce signal périodique qui n'est valable que sur une période de temps entre t_0 et t_1, on utilise la fonction `periodiser_ech_t` du module `seb.py` qui transforme l'échelle de temps `t` en une échelle de temps `tp` dont les valeurs sont entre t_0 et t_1 mais qui associé à `t` donne un signal périodique. Par exemple pour la figure 8.5, on créé d'abord l'échelle de temps souhaité.

```
t=np.linspace(-15,15,10**3)
```

Puis on périodise cette échelle de temps en lui donnant des valeurs entre -2.5 et 2.5.

```
tp=seb.periodiser_ech_t(t,(-2.5,2.5))
```

Il se trouve que le signal non-périodique considéré est $2\mathbb{T}\left(\frac{t}{2}\right) - \mathbb{T}(t)$, aussi le signal périodisé est

```
sp = seb.fonction_T(tp/2)*2-seb.fonction_T(tp)
```

8.5 Calcul numérique des coefficients de la série de Fourier

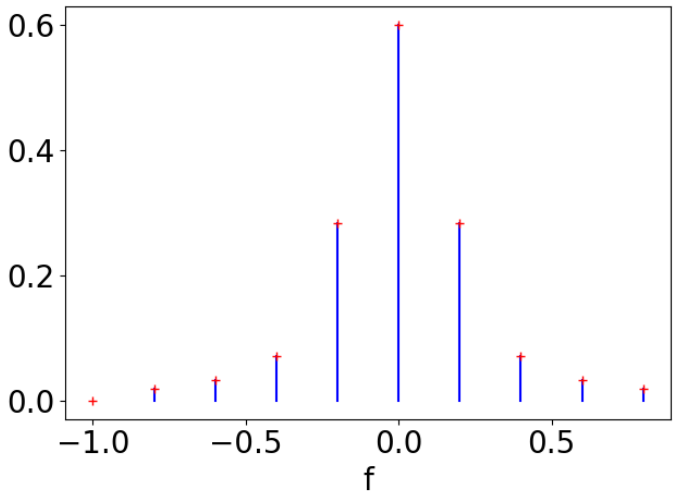

Figure 8.6: Signal $s_1(t)$ en bleu périodisé en $s_1^P(t)$ en répétant le motif $\left[-\frac{5}{2}, \frac{5}{2}\right]$.

La figure 8.6 montre des raies représentant les coefficients de la série de Fourier de $s_1^P(t)$. Comme la restriction de $s_1^P(t)$ sur $[-\frac{5}{2}, \frac{5}{2}]$ coïncide avec $s_1(t)$, ces raies peuvent être obtenues avec $\widehat{S}_1(f)$ pour $f_k = \frac{k}{T}$, ces points sont les plus indiqués en rouge. L'implémentation est dans A.23.
Le module `seb.py` contient la fonction `coef_serie_Fourier` qui approxime le calcul des coefficients de la série de Fourier noté $\widehat{S}_k$. Les arguments sont l'échelle de temps et les valeurs du signal temps continu, l'intervalle sur lequel ce signal est évalué ($-T/2$ et $T/2$ ou 0 et T) et les coefficients k. Cette fonction fournit aussi les fréquences associées aux coefficients k transmis, ce qui forme l'échelle en fréquence.
En l'occurrence pour la figure 8.6, on créé une échelle des indices k entre -5 et 5

```
k = np.arange(-5,5,1)
```

On définit une échelle pour le signal temps continu entre $-T/2$ et $T/2$

```
t=np.linspace(-T/2,T/2,10**3)
```

On évalue le signal temps continu sur cet échelle en utilisant sa définition $s(t) = 2\mathbb{T}(t/2) - \mathbb{T}(t)$.

```
s=2*seb.fonction_T(t/2)-seb.fonction_T(t)
```

Et on utilise la fonction `coef_serie_Fourier`

```
f,SP = seb.coef_serie_Fourier(t,s,(-T/2,T/2),k)
```

8.6 Relation entre transformée de Fourier et calcul des coefficients de la série de Fourier

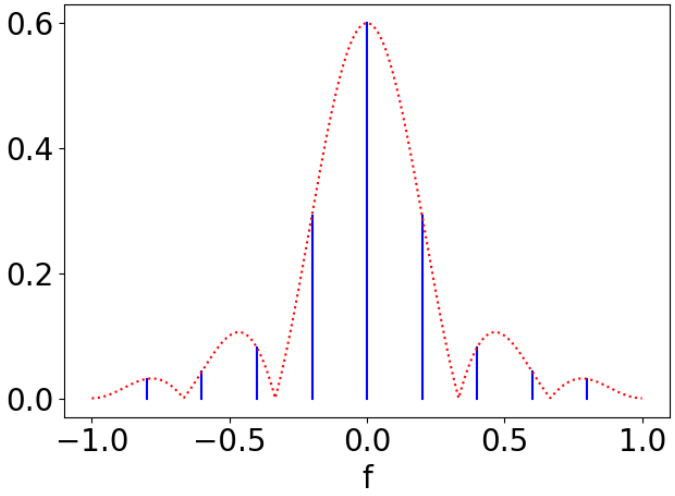

Figure 8.7: L'enveloppe est la transformée de Fourier du signal temps continu divisé par $T = 5$, et les raies sont les coefficients de la série de Fourier.

La ressemblance entre le calcul des coefficients de la série de Fourier défini par (8.8) et le calcul de la transformée de Fourier (**??**) amène à penser qu'il pourrait exister une relation entre les coefficients de la série de Fourier et le résultat de la transformée de Fourier.

Cours signal et bruit :

Les coefficients de la série de Fourier de $x^P(t) = \text{Périodiser}_T(x(t))$ sont des valeurs particulières de la transformée de Fourier, ce sont les valeurs associées à $f_k = \frac{k}{T}$.

$$x^P(t) = \text{Périodiser}_T \quad \Rightarrow \quad \widehat{X}_k^P = \frac{1}{T}\widehat{X}\left(\frac{k}{T}\right) \tag{8.18}$$

Pour représenter la figure 8.7, on représente d'abord la courbe rouge qui est la transformée de Fourier du signal temps continu en considérant d'abord une échelle de fréquence pour afficher,

```
f=np.linspace(-1,1,10**2)
```

une échelle de temps pour la simulation,

```
tx=np.linspace(-2.5,2.5,10**3)
```

une évaluation du signal temps continu,

```
x=2*fonction_T(tx/2)-fonction_T(tx)
```

et finalement le calcul lui-même

```
X=seb.TF(tx,x,f)
```

Pour en extraire les coefficients de la Série de Fourier, on modifie la dernière instruction en utilisant une période de $T = 5$.

```
T=5
k=np.arange(-5,5,1)
XP_k=seb.TF(tx,x,k/T)/T
```

La figure 8.7 est implémentée dans A.22

Chapter 9

Filtres agissant sur des signaux périodiques

9.1 Réponse forcée, réponse transitoire

Cours signal et bruit :

La sortie associée à une entrée périodique de période T est périodique de période T.

D'un point de vue électronique :

Expérimentalement, il n'y a pas de vrais signaux périodiques qui serait défini depuis un temps infini. La sortie observée n'est donc qu'approximativement périodique, elle est la somme d'un signal non-périodique appelé réponse transitoire et d'un signal périodique appelé réponse forcée. Ce dernier signal correspond à la sortie du filtre associée à l'entrée périodique.

Cours signal et bruit :

Pour illustrer la notion de réponse transitoire et réponse forcée, je considère le signal d'entrée $x_1(t) = \cos(\pi t)$ et le filtre de réponse impulsionnelle $h(t) = [\![t \in [0,1]]\!]$ défini par

$$y(t) = \int_{t-1}^{t} x(\tau)\, d\tau \tag{9.1}$$

La sortie associée à $x_1(t)$ est

$$y_1(t) = \int_{t-1}^{t} \cos(\pi\tau)\, d\tau = \left[\frac{\sin(\pi t)}{\pi}\right]_{t-1}^{t} = \frac{2}{\pi}\sin(\pi t) \tag{9.2}$$

Pour voir ce qui se passe quand on commence l'expérience à $t = 0$, je défini $x_2(t) = x_1(t)[\![t \geq 0]\!]$ et je note $y_2(t)$ la sortie correspondante.

- Si $t < 0$, $y_2(t) = \int_{t-1}^{t} x_2(\tau)\, d\tau = 0$.
- Si $t \in [0,1[$, $y_2(t) = \int_{t-1}^{t} x_2(\tau)\, d\tau = \int_{0}^{t} x_2(\tau)\, d\tau = \frac{\sin(\pi t)}{\pi}$
- Si $t \geq 1$, $y_2(t) = \int_{t-1}^{t} x_2(\tau)\, d\tau = y_1(t)$.

La réponse transitoire $y_3(t)$ est définie par

$$y_2(t) = \begin{cases} y_1(t) + y_3(t) \text{ si } t \geq 0 \\ 0 \text{ si } t < 0 \end{cases} \tag{9.3}$$

On en déduit que la réponse transitoire est

$$\begin{aligned} y_3(t) &= (y_2(t) - y_1(t))[\![t \geq 0]\!] = \tfrac{\sin(\pi t)}{\pi}[\![t \in [0,1]]\!] + \tfrac{2}{\pi}\sin(\pi t)[\![t \geq 1]\!] - \tfrac{2}{\pi}\sin(\pi t)[\![t \geq 0]\!] \\ &= -\tfrac{1}{\pi}\sin(\pi t)[\![t \in [0,1]]\!] \end{aligned} \tag{9.4}$$

En terme de visualisation :

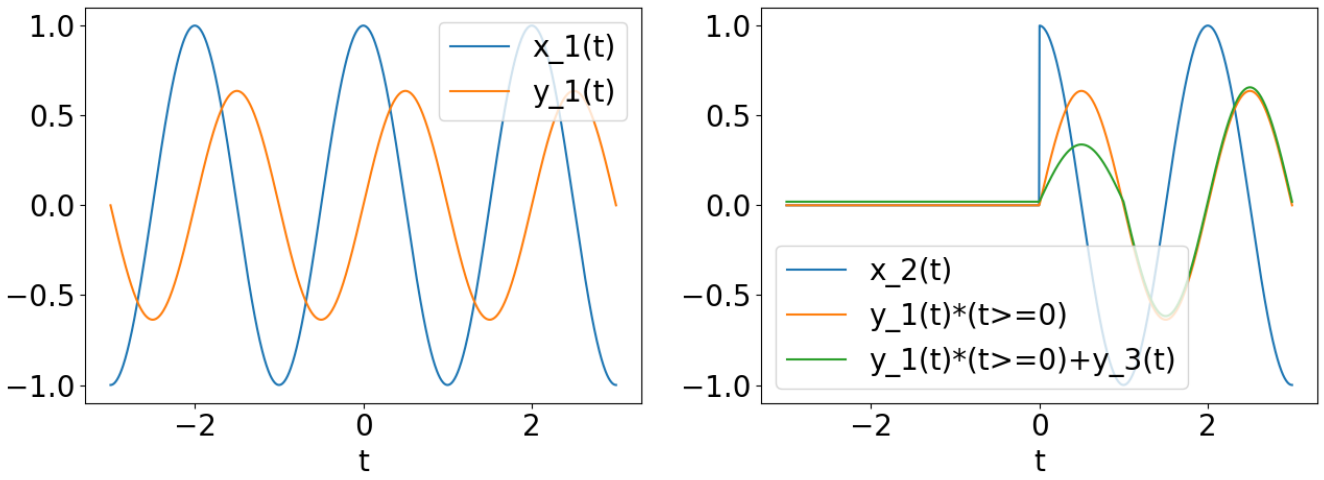

Figure 9.1: À gauche : entrée et sortie périodiques. À droite : entrée et sortie commençant à $t = 0$.

La figure 9.1 montre à gauche un signal sinusoïdal placé en entrée qui est retardé et atténué en sortie. Le retard correspond à un quart de la période, c'est donc un déphasage de $-\frac{\pi}{2}$. Cette figure montre à droite une entrée qui n'est plus périodique puisqu'elle commence en $t = 0$. La sortie est la somme d'une réponse forcée, une sinusoïdale commençant à $t = 0$ (courbe orange) et d'une réponse transitoire, c'est la différence entre la courbe orange et la courbe verte, elle est nulle sauf pour $t \in [0, 1]$. L'implémentation est dans A.13.

9.2 Utilisation de la réponse fréquentielle pour calculer la réponse forcée

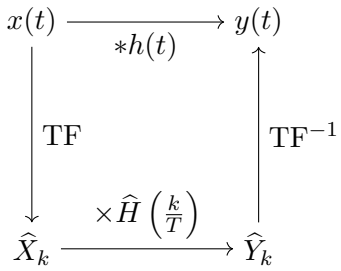

Figure 9.2: Diagramme illustrant la façon de calculer la sortie d'un filtre dont l'entrée est périodique

D'un point de vue mathématique :

Il est tentant d'utiliser la définition du produit de convolution pour calculer la sortie d'un filtre dont l'entrée est périodique. Mais pour une réponse impulsionnelle générale, ceci s'avère en général compliqué.

Cours signal et bruit :

La méthode classique pour calculer la sortie d'un filtre dont l'entrée est périodique est illustré sur la figure 9.2.

- Calcul des $\widehat{X}_k$ de la série de Fourier en fonction de $x(t)$.
- Calcul de $\widehat{Y}_k$ en fonction de $\widehat{X}_k$ et de la réponse fréquentielle évaluée en les fréquences $f_k = \frac{k}{T}$.
- Calcul de $y(t)$ en utilisant la série de Fourier.

Chapter 10

Échantillonnage d'un signal non-périodique

10.1 Signal temps discret

Cours signal et bruit :

Un signal temps discret est décrit par

- des instants régulièrement espacées de T_e appelé période d'échantillonnage ;
- ces instants régulièrement espacées doivent commencer à $t = 0$, aussi ces instants s'écrivent nT_e ;
- à chacun de ces instants nT_e, le signal prend une valeur notée $s_n^{\#}$.

La notation avec # est utilisée dans ce cours pour pouvoir à la fois utiliser la même lettre, ici s pour le signal temps continu et temps discret tout en pouvant les distinguer puisque le signal temps discret a le #. Cette notation n'est pas du tout universelle, elle n'est d'ailleurs utilisée dans ce cours que lorsqu'il y a une ambiguïté .
La période d'échantillonnage est seconde (s).

D'un point de vue mathématique :

Le signal commençant à $n = 0$ et étant composé de N instants, le dernier instant ne peut être NT_e. En effet,

- soit on a $N + 1$ échantillons qui couvrent $0, T_e, \dots NT_e$ et $t_{N+1} = NT_e$;
- soit on a N échantillons qui couvrent $0, T_e, \dots (N-1)T_e$ et $t_N = (N-1)T_e$.

C'est la deuxième convention qu'on utilise dans ce cours.
On appelle fréquence d'échantillonnage

$$f_e = \frac{1}{T_e} \tag{10.1}$$

La fréquence d'échantillonnage est en Hertz (Hz).

Python :

Une fois définies les valeurs de `N,Te,t0`, les commandes suivantes permettent de définir la durée, la fréquence d'échantillonnage et l'échelle de temps.

```
T=N*Te
fe=1/Te
t=np.arange(t0,t0+T,Te)
```

En déclenchant une erreur quand une affirmation est fausse, la commande `assert` est très pratique pour quelqu'un qui relit un code, parce qu'elle lui permet d'être sûr d'une affirmation.

```
assert t[N-1]==t[-1]
assert t[-1]==t0+(N-1)*Te
assert Te==t[1]-t[0]
```

Cours signal et bruit :

Ainsi un signal est décrit par $t = nT_e$ et $s_n^{\#}$. On peut écrire aussi un signal en utilisant des Diracs.

$$s^{\#}(t) = \sum_{n=-\infty}^{+\infty} s_n \delta(t - nT_e) \tag{10.2}$$

Cette autre formulation n'apporte rien de plus que $t = nT_e$ et $s_n^{\#}$.

D'un point de vue électronique :

Un signal numérique traité par un processeur électronique est généralement modélisé par signal temps discret, avec une période d'échantillonnage qui est égale ou est un multiple de l'inverse de la fréquence de l'horloge. Et la valeur du signal est codé en binaire.
Un signal temps discret peut aussi être utilisé pour rendre compte de mesures physiques faites a des instants régulièrement répartis dans le temps, par exemple la prise de la température toutes les 30 secondes, ou la mesure de la crue du Nil tous les ans depuis 2000 ans.

10.2 Échantillonnage

Cours signal et bruit :

On appelle échantillonnage le fait de transformer un signal temps continu (i.e. défini à chaque instant) en un signal temps discret. Cela suppose nécessairement le choix d'une période d'échantillonnage.

$$s_e[n] = s(nT_e) \tag{10.3}$$

On peut à nouveau utiliser la notation avec des Diracs

$$s_e(t) = \sum_{n=-\infty}^{+\infty} \delta(t - nT_e) s(t) \tag{10.4}$$

D'un point de vue mathématique :

Le produit d'une fonction avec une distribution n'est pas nécessairement défini pour toutes les fonctions et toutes les distributions. Ici il a un sens $s(t)\delta(t - n_T) = s(nT_e)\delta(t - nT_e)$, en admettant que $s(t)$ tend vers une valeur notée $s(nT_e)$ quand t tend vers nT_e. Dans le cas où cette limite existe à gauche et à droite, on remplace la valeur par la demi-somme de la limite à gauche et à droite.
On appelle peigne de Diracs la succession infinie de Diracs

$$\text{Ш}(t) = \sum_{n=-\infty}^{+\infty} \delta(t - n) \text{ et } \text{Ш}\left(\frac{t}{T_e}\right) = \sum_{n=-\infty}^{+\infty} \delta(t - nT_e) \tag{10.5}$$

Cette notation permet d'écrire l'échantillonnage comme

$$s_e(t) = \text{Ш}\left(\frac{t}{T_e}\right) s(t) \tag{10.6}$$

D'un point de vue électronique :

La notion d'échantillonnage ainsi définie est une situation idéalisée. En réalité la conversion d'un signal temps continu en un signal temps discret correspond plutôt à une moyenne des valeurs du signal pendant une durée donnée. Du coup un échantillonnage réaliste peut être modélisé par un filtre analogique appliqué au signal temps continu dont la sortie est, elle, échantillonnée en un signal temps discret, au sens de prélever une valeur sur une durée infiniment petite.

En termes numériques :

Numériquement les signaux sur ordinateur sont de fait échantillonnés parce qu'ils sont gérés par un tableau de valeurs. Par contre il y a une différence entre simuler un signal échantillonné et simuler un signal temps continu au moyen d'un signal échantillonné. Dans le premier cas, la valeur de la période d'échantillonnage compte beaucoup, dans le second cas, ce qui nous importe est que les calculs soient suffisamment précis on cherche donc une période d'échantillonnage très petite. Cette période d'échantillonnage ne doit pas être trop petite pour éviter que la simulation ne prenne trop de temps. Par exemple j'évite en général de définir un variable contenant plus de 10^7 composantes à stocker en mémoire.

10.3 Signaux temps discret non périodiques : transformées de Fourier à temps discret

Cours signal et bruit :

La transformée de Fourier d'un signal temps discret est un signal périodique de période f_e. Et à l'inverse tout fonction périodique est la transformée d'un signal temps discret dont la période d'échantillonnage est l'inverse de la période de cette fonction périodique.

D'un point de vue mathématique :

Cette idée de périodicité peut se retrouver avec le peigne de Diracs. La transformée de Fourier d'un peigne de Dirac est donnée par

$$\mathrm{TF}\left[\text{Ш}(t)\right](f) = \text{Ш}(f) \text{ et } \mathrm{TF}\left[\text{Ш}\left(\frac{t}{T_e}\right)\right](f) = T_e \text{Ш}\left(\frac{f}{f_e}\right) \tag{10.7}$$

Le coefficient T_e vient de l'application de la formule (3.18) (p. 25) adaptées aux signaux temps continu.
La transformée de Fourier d'un produit est un produit de convolution et le produit de convolution par un peigne de Dirac est une périodisation aussi

$$\widehat{s_e}(t) = \frac{1}{f_e}\text{Ш}\left(\frac{f}{f_e}\right) * \widehat{S}(f) = \frac{1}{f_e}\sum_{k=-\infty}^{+\infty} \widehat{S}(f - kf_e) = \frac{1}{f_e}\text{Périodisation}_{f_e}\left[\widehat{S}(f)\right] \tag{10.8}$$

On retrouve le coefficient T_e maintenant écrit sous la forme $\frac{1}{f_e}$.
On définit une nouvelle formulation de la transformée de Fourier adaptée aux signaux temps discret non périodiques et appelés Transformée de Fourier à Temps Discret, que l'on note ici TFTD.

$$\mathrm{TFTD}\left[s_n\right](f) = \sum_{n=-\infty}^{+\infty} s_n e^{-j2\pi f n T_e} \tag{10.9}$$

10.4 Périodisation du spectre

$$\mathrm{TF}(x_e)(f) = f_e \sum_{k=-\infty}^{+\infty} \widehat{X}(f - kf_e) \tag{10.10}$$

10.5 Interpolation

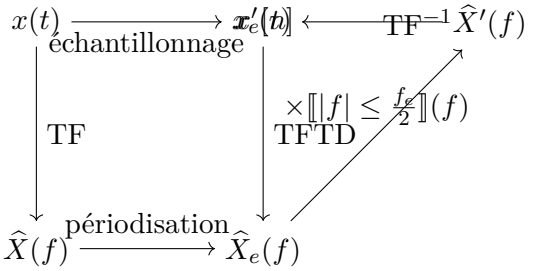

Figure 10.1: Diagramme décrivant l'interpolation

Cours signal et bruit :

L'interpolation signifie estimer la valeur du signal à un instant t connaissant les valeurs des échantillons du signal aux instants nT_e.

En terme de visualisation :

On peut visualiser le problème de l'interpolation en t, en considérant que pour une période d'échantillonnage T_e, il existe un entier n tel que $nT_e \leq t < (n+1)T_e$, cet entier est donné par $n = \lfloor \frac{t}{T_e} \rfloor$, où $\lfloor \ldots \rfloor$ est appelée la partie entière. Du coup l'interpolation peut se voir comment utiliser la valeur du signal en nT_e et en $(n+1)T_e$ pour en déduire celle en t. Le point de vue en traitement du signal est différent parce qu'on s'autorise à utiliser toutes les valeurs du signal en nT_e, quelque soit la valeur de n.

D'un point de vue mathématique :

Il existe de nombreuses techniques pour interpoler un signal, ces techniques reposent sur ce qu'on sait du signal. Par exemple si on sait que le signal est une succession de traits horizontaux, on peut approcher le signal en prolongeant la valeur en nT_e : $s'(t) = s(nT_e)$ où $s'(t)$ est le signal interpolé en t.

Cours signal et bruit :

L'interpolation du signal est décrite sur la figure 10.1.

- On calcule $\widehat{X}_e(f)$ la transformée de Fourier à temps discret du signal échantillonné $x_e[n]$.
- Le spectre périodique de période f_e, $\widehat{X}_e(f)$, est rendu non-périodique en le multipliant par la porte qui vaut 1 pour $f \in [-\frac{f_e}{2}, \frac{f_e}{2}[$ et 0 ailleurs, le résultat est noté $\widehat{X}'(f)$.
- On calcule la transformée de Fourier inverse de $\widehat{X}'(f)$ pour trouver $x'(t)$.

Cours signal et bruit :

Comme le spectre est périodique de période f_e, on représente généralement celui-ci sur un intervalle de longueur f_e.

- On appelle représentation centrée le fait de représenter le spectre sur l'intervalle $[-\frac{f_e}{2}, \frac{f_e}{2}]$.
- On appelle représentation non-centrée le fait de représenter le spectre sur l'intervalle $[0, f_e]$.

Chapter 11

Peigne de Diracs

Chapter 12

Modélisation stochastique du bruit

12.1 Qu'est-ce que le bruit ?

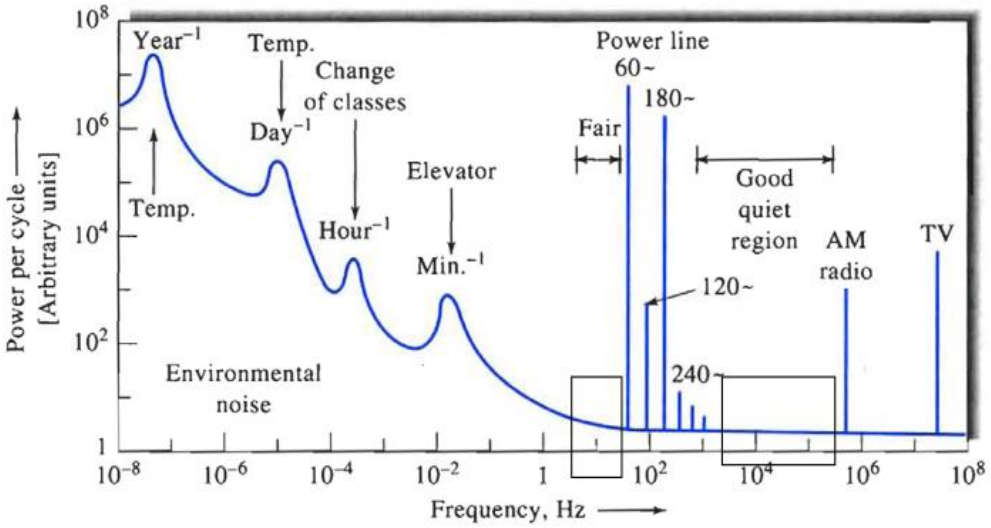

FIGURE 5-3 Some sources of environmental noise in a university laboratory. Note the frequency dependence and regions where various types of interference occur. (From T. Coor, *J. Chem. Educ.*, **1968**, *45*, A540. With permission.)

D'un point de vue électronique :

En électronique, on définit le bruit comme une conséquence de phénomènes physiques identifiés. Ces phénomènes produisent un écart à la valeur théorique, ces écarts sont généralement de moyennes nulles mais le carré de ces écarts n'est pas de moyenne nulle. Les sources du bruit sont :

- bruit thermique
- bruit de grenaille
- bruit en $1/f$
- bruit lié à la quantification d'un signal
- perturbation liée à l'alimentation
- perturbation électromagnétique

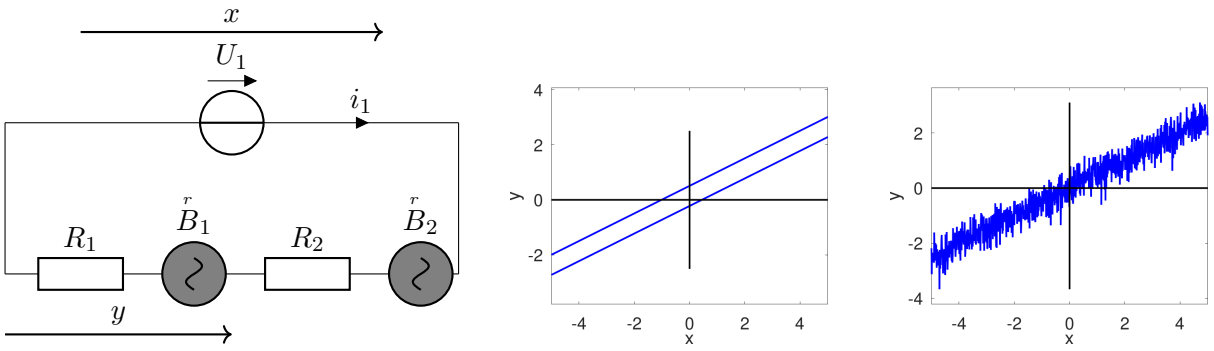

Figure 12.1: À gauche : diviseur de tension. Au milieu : graphe de y en fonction de x pour deux réalisations différentes. À droite : graphe de y en fonction de x lorsqu'on teste successivement pour chaque valeur de x la valeur de y.

Cours signal et bruit :

En théorie du signal, on définit le bruit en fonction de l'application envisagée, on la définit comme tout écart au modèle. Ce bruit est ensuite modélisé par un processus stochastique, c'est-à-dire un signal dont la valeur à chaque instant est lui-même aléatoire. Ainsi ce qui est un bruit pour une application donnée peut s'avérer le signal dans une autre application. De ce point de vue on distingue le bruit modélisé par un processus aléatoire et le bruit modélisé par un signal déterministe qui peut dépendre de certains signaux.

12.2 Bruit modélisé par une variable aléatoire

D'un point de vue électronique :

Un composant ne se comporte pas exactement tel qu'il est modélisé. La modélisation plus précise utilise de multiples composants supplémentaires, elle utilise aussi une source de tension ou d'intensité dont la valeur est modélisée par une variable aléatoire.

Exemple pour illustrer :

La figure 12.1 montre un diviseur de tension où chaque résistance est modélisée par une résistance sans bruit et une source de tension qui génère un bruit gaussien.

$$y = R\frac{x}{2} + \frac{1}{2}\overset{r}{B}_1 - \frac{1}{2}\overset{r}{B}_2 \tag{12.1}$$

D'un point de vue mathématique :

On appelle une réalisation le fait de procéder à un tirage aléatoire.
Une variable aléatoire gaussienne est définie par $\overset{r}{B} \sim \mathcal{N}(\mu, \sigma)$

$$E\left[\overset{r}{B}\right] = \mu \quad \text{et} \quad \text{Var}\left[\overset{r}{B}\right] = \sigma^2 \tag{12.2}$$

En général pour un bruit $\mu = 0$. On dit que la variable aléatoire est centrée.
Les signaux dont on a parlé dans les chapitres précédents ne dépendent pas d'un tirage aléatoire, on les appelle signaux déterministes.
La notation $\overset{r}{B}$ est spécifique à ce cours, elle est pratique parce qu'elle me permet de distinguer la notation d'un signal aléatoire de celle d'un signal déterministe.

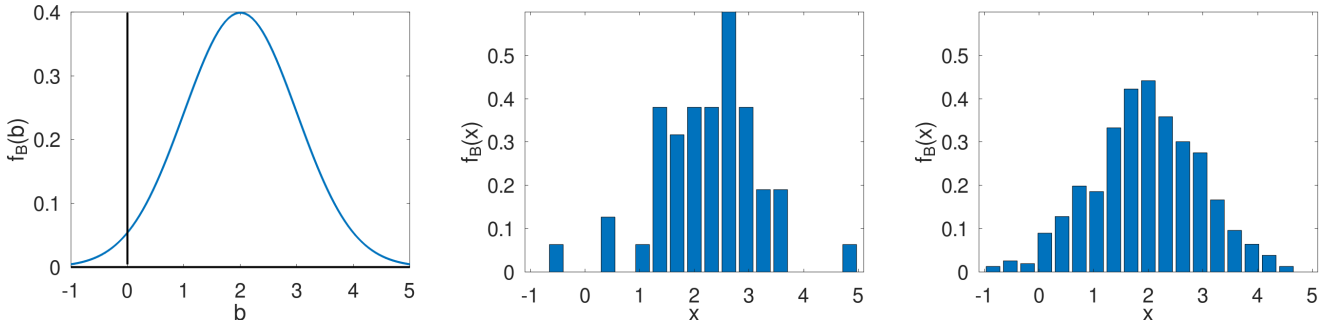

Figure 12.2: À gauche fonction Gaussienne pour $\mu = 2$ et $\sigma = 1$. Au milieu et à droite histogrammes obtenues avec 50 et 500 tirages aléatoires de $\overset{r}{B}$ suivant la loi $\mathcal{N}(2, 1)$.

D'un point de vue mathématique :

La densité de probabilité d'une gaussienne $\overset{r}{B} \sim \mathcal{N}(\mu, \sigma)$ est

$$f_B(b) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}\frac{(b-\mu)^2}{\sigma^2}} \tag{12.3}$$

Cette gaussienne est représentée à gauche de la figure 12.2.

$$E\left[g(\overset{r}{B})\right] = \int_{-\infty}^{+\infty} f_B(b) g(b)\, db \text{ et } \text{Var}\left[\overset{r}{B}\right] = E\left[\overset{r}{B}^2\right] - E\left[\overset{r}{B}\right]^2 = E\left[\left(\overset{r}{B} - E\left[\overset{r}{B}\right]\right)^2\right] \tag{12.4}$$

Pour un tirage d'un certain nombre de valeurs.

- 68 % des échantillons sont entre $\mu - \sigma$ et $\mu + \sigma$.
- 95 % des échantillons sont entre $\mu - 2\sigma$ et $\mu + 2\sigma$.
- 99.7 % des échantillons sont entre $\mu - 3\sigma$ et $\mu + 3\sigma$.

Il est important de distinguer la fonction gaussienne qui en tant que fonction est déterministe d'un tirage aléatoire dont la densité de probabilité est décrit par cette fonction gaussienne.
Les différentes sources de bruit sont généralement supposées indépendantes. Et dans ce cas

$$E\left[g_1(\overset{r}{B}_1) g_2(\overset{r}{B}_2)\right] = E\left[g_1(\overset{r}{B}_1)\right] E\left[g_2(\overset{r}{B}_2)\right] \tag{12.5}$$

où g_1 et g_2 sont deux fonctions quelconques.
Ajouter une constante à $\overset{r}{B}$ c'est modifier sa moyenne statistique.

$$E\left[\overset{r}{B} + \mu'\right] = \mu + \mu' \tag{12.6}$$

Amplifier (ou atténuer) un bruit c'est modifier sa moyenne et son écart-type.

$$\text{E}\left[\lambda \overset{r}{B}\right] = \lambda\mu \quad \text{et} \quad \text{Var}\left[\lambda \overset{r}{B}\right] = \lambda^2\sigma^2 \tag{12.7}$$

12.3 Présentation physique du bruit thermique

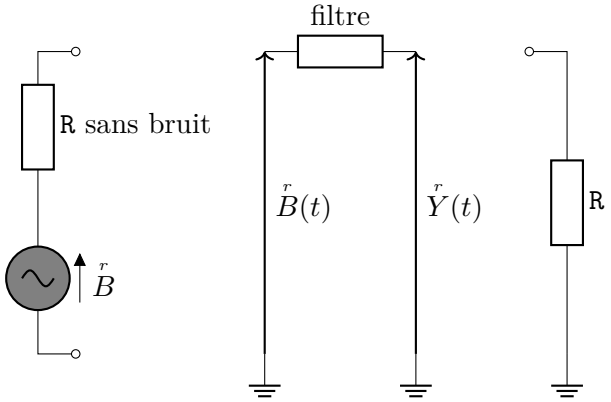

Figure 12.3: Modélisation du bruit thermique par un bruit en tension $\vec{B}$ en série avec une résistance sans bruit amplifié d'un facteur a par un filtre à bande limitée ΔB.

D'un point de vue électronique :

Une résistance R, avec ou sans courant, provoque un bruit thermique dont la **puissance électronique par unité de fréquence** appelé densité spectrale de puissance, son unité est WHz^{-1} qui est homogène à un Joule J mais ne correspond pas du tout à cette notion.

$$\mathtt{P}_{W/Hz} = 4\mathtt{k}_B\mathtt{T} \tag{12.8}$$

- $\mathtt{k}_B$ est la constante de Boltzmann au sens de la physique.
- T est la température en degré Kelvin au sens de la physique.

Ce bruit est aussi appelé *thermal noise* ou *Johnson noise*. Le bruit est modélisé par une nouvelle source de bruit en tension comme le montre la gauche de la figure 12.3. À partir de $\mathtt{P}_{W/Hz}$ et la résistance on récupère la densité spectrale de la puissance qui fournit une telle puissance électronique par unité de fréquence

$$\mathtt{S}_B(f) = \mathtt{P}_{W/Hz}R \tag{12.9}$$

Ce bruit pourrait laisser penser qu'il est infini. En réalité la mécanique quantique amène à donner une autre formulation plus précise utilisant la constante de Planck $\mathtt{h} \approx 6{\times}10{-}34$

$$\mathtt{P}_{W/Hz} = 4\frac{\mathtt{h}f}{e^{\frac{\mathtt{h}f}{\mathtt{k}_B\mathtt{T}}} - 1} \tag{12.10}$$

En utilisant un changement de variable $u = \frac{\mathtt{h}f}{\mathtt{k}_B\mathtt{T}}$ et en utilisant que $\int_0^{+\infty} \frac{u}{e^u-1}\,du = \frac{\pi^2}{6}$, on trouve qu'au total la puissance disponible avec la résistance est de l'ordre de 80nW et celle qu'on pourrait idéalement récupérer serait de $\frac{80}{2} = 40$nW. L'approximation de l'équation (12.9) est valable pour des fréquences inférieures à 10THz. Pour observer ce bruit thermique (et par exemple mesurer la constante de Boltzmann $\mathtt{k}_B$), le signal aléatoire $\vec{B}(t)$ est amplifié et filtré en $\vec{Y}(t)$ comme le montre le milieu de la figure 12.3.

$$\vec{Y}(t) = h(t) * \vec{B}(t) \tag{12.11}$$

Ce filtre est caractérisé par une amplification a et une bande de fréquence $\mathcal{B}$ aussi

$$\mathrm{S}_Y(f) = a^2\mathtt{P}_{W/Hz}R[\![f \in \mathcal{B}]\!] \tag{12.12}$$

À partir de $\mathrm{S}_Y(f)$ et en notant ΔB la largeur de l'intervalle $\mathcal{B}$, on récupère la puissance de $\vec{Y}(t)$, noté P_y

$$\mathrm{P}_y = a^2\mathtt{P}_{W/Hz}R\Delta B \tag{12.13}$$

et on récupère la variance du bruit

$$\mathrm{Var}\left(\overset{r}{Y}(t)\right) = \mathrm{P}_y = a^2 \mathtt{P}_{W/Hz} R \Delta B \tag{12.14}$$

Une résistance est mise en sortie du filtre, elle dissipe une puissance électronique donnée par

$$\mathtt{P}_{\mathrm{W}} = \frac{\mathrm{P}_y}{R} = a^2 \mathtt{P}_{W/Hz} \Delta B \tag{12.15}$$

12.4 Processus aléatoire

12.4.1 Généralités

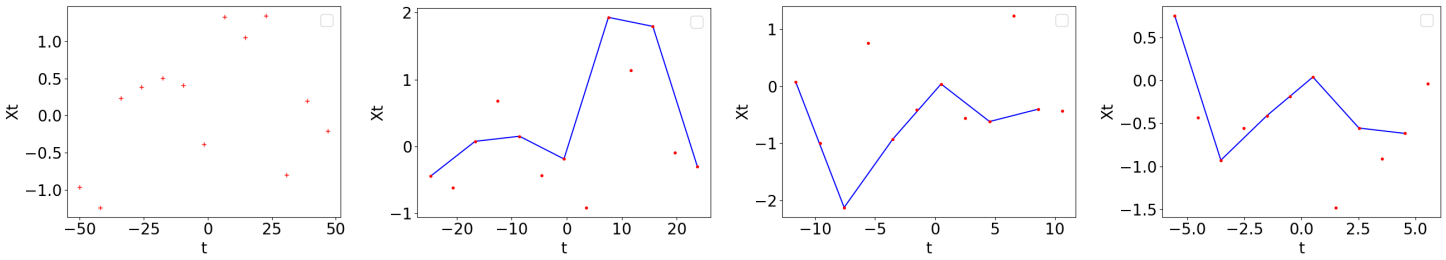

Figure 12.4: Extrait d'une réalisation d'un bruit blanc et zoom successif permettant de compléter les points avec des voisins.

Cours signal et bruit :

Un processus aléatoire peut se voir comme une variable aléatoire dépendant du temps.

- $\overset{r}{X}(t)$ est un processus aléatoire
- $x(t) = \overset{r}{X}(t, \omega)$ est une réalisation particulière de $\overset{r}{X}(t)$.
- $\overset{r}{X}(t_0)$ est une variable aléatoire

La difficulté vient de ce que $\overset{r}{X}(t)$ pourrait dépendre des autres instants.

12.4.2 Processus aléatoire blanc

On définit un processus aléatoire où il n'y a aucune dépendance.
$\overset{r}{X}(t)$ est un bruit blanc signifie que

$$t_0 \neq t_1, \quad \Rightarrow \quad \overset{r}{X}(t_0) \text{ est indépendant de } \overset{r}{X}(t_1) \tag{12.16}$$

En termes numériques :

La simulation d'une partie d'une réalisation d'un processus aléatoire dépend d'un vecteur `t` correspondant à une échelle de temps et caractérisée par un début t_0, une fin $t_0 + T$ et un nombre de points N associé à une période d'échantillonnage T_e et une fréquence d'échantillonnage f_e. Le vecteur de temps est la liste de N valeurs espacées de T_e notées t_n

$$T = NT_e \quad f_e = \frac{1}{T_e} \quad t_n = nT_e + t_0 \quad t_{N-1} = T - T_e + t_0 \quad T_e = t_1 - t_0 \tag{12.17}$$

Il se trouve que pour que la théorie du traitement du signal fonctionne il faut que tous les instants t_n soient des multiples de T_e.
La figure 12.4 montre dans ses quatre graphiques un même signal qui est une réalisation d'un bruit blanc à bande limitée sur $[-10^6, 10^6]$ en Hz. Le premier graphique montre son évolution entre -50 et 50s, le deuxième est un zoom

entre -25 et 25s, le troisième est encore un zoom entre -12.5 et 12.5s, le quatrième est encore un zoom entre -6.25 et 6.25s. Chaque graphique ne montre que 12 points. Pour les graphiques 2,3,4, les traits bleus joints par des pointillés sont les 6 points déjà montrés respectivement dans les graphiques 1,2,3. Les 12 points en bleus sont pour 6 déjà présents dans le graphe à gauche et 6 sont nouveaux. Ceci est implémentée dans A.16.

12.4.3 Filtrage d'un processus aléatoire pour obtenir un processus aléatoire non-blanc

Cours signal et bruit :

D'après l'équation (12.16), un processus aléatoire non-blanc ne vérifie pas à priori $\mathrm{E}\left[\overset{r}{X}(t_1)\overset{r}{X}(t_2)\right] = \mathrm{E}\left[\overset{r}{X}(t_1)\right]\mathrm{E}\left[\overset{r}{X}(t_2)\right]$

L'idée générale pour obtenir un bruit non blanc est de mettre le bruit blanc en entrée d'un filtre. On n'a plus l'indépendance entre les différents instants.

En termes numériques :

Dans le cadre de ce cours, la façon d'obtenir un processus aléatoire gaussien non-blanc consiste à considérer un bruit blanc gaussien numérique à une fréquence d'échantillonnage f_e notée $\overset{r}{B}_n$ et à le filtrer avec un filtre $\mathcal{H}$ de réponse impulsionnelle $h(t)$ et de réponse fréquentielle $\widehat{H}(f)$. Le filtre $\mathcal{H}$ doit être discrétisé pour fonctionner avec $\overset{r}{B}_n$. La sortie notée $\overset{r}{Y}_n$.

$$\overset{r}{Y}_n = \mathcal{H}_n^{\#} * \overset{r}{B}_n \text{ où } h_n^{\#} = T_e h(nT_e) \tag{12.18}$$

Cette méthode permet d'avoir

$$\mathrm{E}\left[\overset{r}{Y}_n\right] = 0 \text{ et } \mathrm{E}\left[\overset{r}{Y}_n^2\right] \approx \frac{\Delta B_E}{f_e} \max_f \left|\widehat{H}(f)\right|^2 \tag{12.19}$$

Ici j'utilise une autre définition de la bande de fréquence ΔB_E qui est un peu différente de celle utilisée dans (**??**) où on se repère avec les fréquences de coupures. Il se trouve qu'approximativement cela donne le même résultat. Ici la définition de la bande de fréquence est choisie pour la relation (12.19) soit la plus précise.

$$\Delta B_E = \int_{-\infty}^{+\infty} \left|\widehat{H}(f)\right|^2 df \tag{12.20}$$

Dans le cadre de ce cours je note $\overset{f}{Y}_n = \sqrt{f_e}\overset{r}{Y}_n$.

D'un point de vue mathématique :

Pour les processus aléatoire, les calculs infinitésimaux posent problème : intégrale, dérivation, produit de convolution, transformée de Fourier. Il existe plusieurs définitions mathématiques, mais ces définitions requièrent plus de régularité pour $\overset{r}{X}(t)$ et elles amènent à renoncer en partie à la notion de densité de probabilité pour le processus aléatoire résultant.
Du coup, il est naturel de chercher à approcher les filtres analogiques par un filtre numérique en considérant une certaine fréquence d'échantillonnage assez élevée. pour des fréquences inférieures à une certaine fréquence maximale.

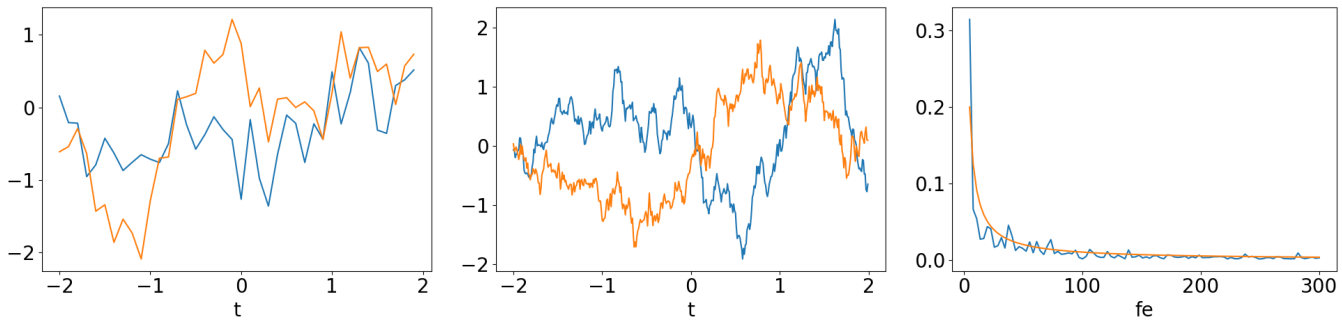

Figure 12.5: Sortie d'un filtre moyenneur en supposant que le signal est échantillonné à 10Hz à gauche et à 100Hz au milieu. À droite : Variance de $\overset{r}{Y}_n$ en fonction de f_e.

Exemple pour illustrer :

Je considère un filtre moyenneur de réponse impulsionnelle $h(t) = [\![t \in [0,1]]\!]$ et de relation entrée sortie

$$y(t) = \int_{t-1}^{t} x(\tau)\, d\tau \tag{12.21}$$

Les notations $\overset{r}{Y}(t)$ et $\overset{f}{Y}(t)$ sont, en un sens, abusives car elles n'ont de sens que pour des valeurs de t multiples de T_e.

$$\overset{f}{Y}(nT_e) = \sqrt{f_e}\overset{r}{Y}(nT_e) = \frac{\sqrt{f_e}}{\lfloor f_e \rceil} \sum_{k=n-\lfloor f_e \rceil+1}^{n} \overset{r}{X}(kTe) \approx \frac{1}{\sqrt{f_e}} \sum_{k=n-\lfloor f_e \rceil+1}^{n} \overset{r}{X}(kTe) \tag{12.22}$$

Ainsi défini, $\overset{f}{Y}(nT_e)$ est pratiquement indépendante de f_e.
La figure 12.5 montre quatre réalisations obtenues en utilisant l'équation (**??**), à gauche avec $f_e = 10$Hz et au milieu avec $f_e = 100$Hz en utilisant $\overset{f}{Y}_n$ (i.e. processus corrigé par $\sqrt{f_e}$). L'implémentation est dans A.17. Ce sont des processus aléatoires qui sont différents d'une réalisation à l'autre et on voit bien qu'au sein de chaque graphe les courbes sont différentes. Je vais montrer que celles de droite sont identiques à celles de gauche à un coefficient de proportionnalité prés corrigé grâce au produit par $\sqrt{f_e}$. Les moyennes des carrés valent 0.3 et 0.7 à gauche et 0.7 et 0.65 à droite.
La droite de la figure 12.5 montre justement la moyenne des carrés en utilisant ici les $\overset{r}{Y}_n$ et non sur chaque trajectoire en fonction de f_e et on voit justement que les valeurs tournent autour de $\frac{1}{f_e}$ et par suite que lorsqu'on considère $\overset{f}{Y}_n = f_e\overset{r}{Y}_n$, cela ne dépend plus de f_e (i.e. pas de manière aussi significative).

Cours signal et bruit :

Les caractéristiques d'un processus aléatoire sont données par

- La moyenne $\mathrm{E}\left[\overset{r}{X}_n\right]$ qui pour un processus centré vaut 0.
- La variance $\mathrm{Var}\left[\overset{r}{X}_n\right]$ qui pour un processus stationnaire de dépend pas de n.
- La corrélation $\mathrm{E}\left[\overset{r}{X}_n\overset{r}{X}_{n-k}\right]$ (définition adaptée à un processus centré)
- La transformée de Fourier $\mathrm{E}\left[\mathbb{TFTD}\left[\overset{r}{X}_n\right](f)\right]$
- La distribution de probabilité de $\overset{r}{X}_n$ qui pour un $\boxed{\text{processus stationnaire}}$ de dépend pas de n.

Exemple pour illustrer :

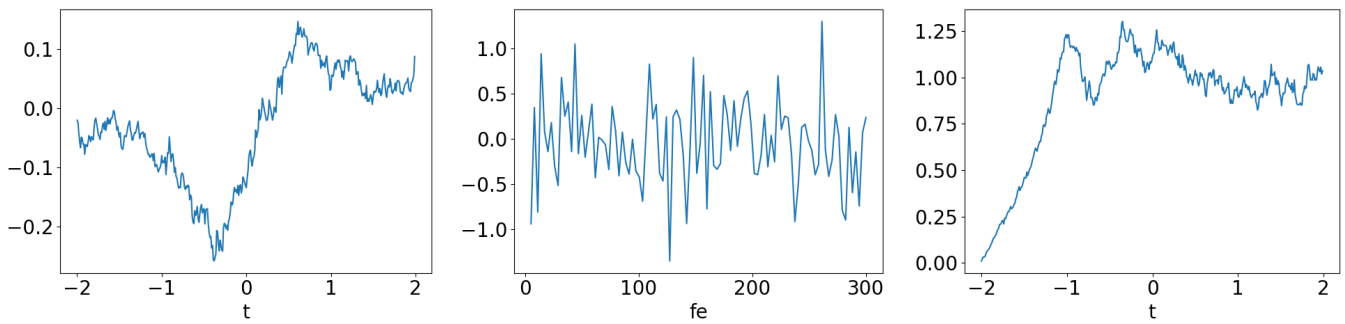

Figure 12.6: À gauche : trajectoire moyenne obtenue en moyennant 100 trajectoires de $\overset{f}{Y}_n$ en fonction du temps pour $f_e = 100$Hz. Au milieu : Valeur moyenne de chaque trajectoire $\overset{f}{Y}_n$ en fonction de f_e. À droite : trajectoire moyenne en moyennant le carré de 100 trajectoires de $\overset{f}{Y}_n$ en fonction du temps pour $f_e = 100$Hz.

La figure **??** montre à gauche une trajectoire obtenue en simulant d'abord 100 trajectoires $\overset{f}{Y}_n(\omega_k)$ pour $k \in \{0 \dots 99\}$ puis en calculant une valeur moyenne $\frac{1}{100}\sum_{k=0}^{99} \overset{f}{Y}_n(\omega_k)$. On voit que cette trajectoire est beaucoup plus proche de la trajectoire nulle que celles de la gauche de la figure 12.5. A droite on voit que toutes les trajectoires simulées ont des moyennes proche de 0 mais ces valeurs sont du même ordre de grandeur quelque soit f_e. L'implémentation est faite dans **??**.

Cours signal et bruit :

Un processus centré est défini par $\mathrm{E}\left[\overset{r}{X}_t\right] = 0$. C'est toujours le cas lorsqu'il est généré avec un filtre en partant d'un processus centré, ce qui est le cas de $\overset{r}{B}_n$.

Exemple pour illustrer :

La droite de la figure 12.6 montre la valeur moyenne du carré des valeurs de chacune des 100 trajectoires simulées. $\frac{1}{100}\sum_{k=0}^{99} \overset{f}{Y}_n{}^2(\omega_k)$. On observe que cela se stabilise très vite autour de 1. Ainsi le processus aléatoire vérifie $\mathrm{E}\left[\overset{f}{Y}_n{}^2\right] = 1$. Au début de la courbe on voit qu'elle part de 0 et monte rapidement à 1, ceci est dû au fait qu'avant $t = 0$, les valeurs étaient nulles et que celles-ci ont un impact pour t petit. En ce sens, le processus simulé n'est pas stationnaire. Cependant le processus aléatoire théorique obtenu en filtrant un bruit blanc gaussien est stationnaire.

Cours signal et bruit :

La puissance d'un processus aléatoire est défini par

$$P_{\overset{f}{Y}} = \mathrm{E}\left[\overset{f}{Y}_n{}^2\right] = \mathrm{Var}\left[\overset{f}{Y}_n\right] + \mathrm{E}\left[\overset{f}{Y}_n\right]^2 \tag{12.23}$$

Exemple pour illustrer :

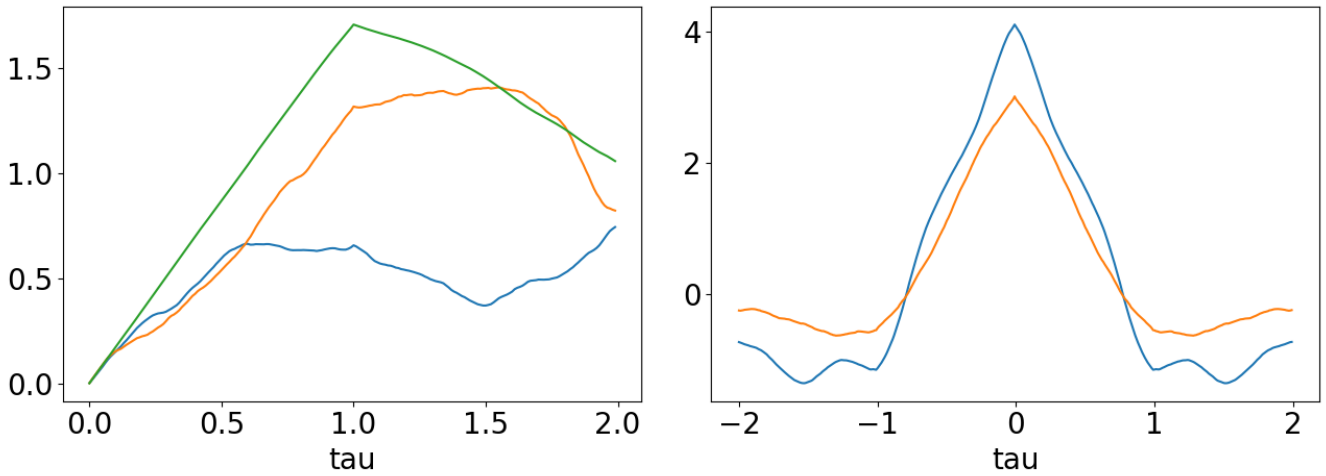

Figure 12.7

La figure 12.7 montre à gauche deux courbes obtenues à partir deux réalisations ω_0 et ω_1. Il s'agit du carré de la différence entre les valeurs en $t = -1 + \tau$ et $t = -1$: $(\overset{f}{Y}_{k+f_e}(\omega_0) - \overset{f}{Y}_{-f_e}(\omega_0)^2$ en fonction de $\tau = \frac{k}{f_e}$ et de même

$(\overset{f}{Y}_{k+f_e}(\omega_1) - \overset{f}{Y}_{-f_e}(\omega_1)^2$ en fonction de $\tau = \frac{k}{f_e}$. Ces deux courbes sont un peu différentes. Quand on les moyenne sur 100 réalisations, on obtient la troisième courbe beaucoup plus raide : $\frac{1}{100}\sum_{l=0}^{99}(\overset{f}{Y}_{k+f_e}(\omega_l) - \overset{f}{Y}_{-f_e}(\omega_l)^2$ en fonction de $\tau = \frac{k}{f_e}$. Ces courbes approchent $\mathrm{E}\left[\left(\overset{f}{Y}_{n+n0} - \overset{f}{Y}_{n0}\right)^2\right]$. Cette courbe part de 0 car quand $n = 0$, les deux quantités sont identiques. Elle augmente car plus n augmente moins les deux quantités se ressemblent. Au delà de $\tau = 1$, c'est-à-dire $n = f_e$, les deux quantités sont très différentes et ensuite elle diminue du fait des effets de bord, le signal étant ensuite composé de valeurs nulles.
La figure 12.7 montre à droite deux courbes obtenues en calculant l'autocorrélation pour deux réalisations ω_0 et ω_1. Cette autocorrélation est définie comme la valeur moyenne du produit d'un signal par lui-même retardé en fonction du retard. Il s'agit de $\frac{1}{4f_e}\sum_{n=-2f_e}^{2f_e-1}\overset{f}{Y}_n(\omega_0)\overset{f}{Y}_{n-k}(\omega_0)$ et de la même expression en fonction de ω_1 en fonction de $\tau = \frac{k}{f_e}$. On observe que la corrélation est approximativement nulle pour $\tau < -1$, qu'elle augmente progressivement jusqu'à $\tau = 0$ parce que plus τ augmente plus les signaux se ressemblent. Ensuite la corrélation diminue jusqu'à $\tau = 1$.
L'implémentation est dans **??**.

Cours signal et bruit :

On définit l'autocorrélation pour un processus aléatoire $\overset{r}{X}_n$

$$\gamma_k = \mathrm{E}\left[\overset{r}{Y}_n\overset{r}{Y}_{n-k}\right] \tag{12.24}$$

Pour un processus aléatoire stationnaire et ergodique, cela coïncide avec

$$\gamma_k = \sum_{n=-\infty}^{+\infty}\overset{r}{Y}_n(\omega)\overset{r}{Y}_{n-k}(\omega) \tag{12.25}$$

$n \mapsto \overset{r}{Y}_n(\omega)$ étant une réalisation du processus aléatoire.

Cours signal et bruit :

On définit la densité spectrale de puissance à partir de l'autocorrélation lorsque celle-ci n'est pas périodique.

$$S_x(f) = \mathbb{TFTD}\left[\gamma_k\right](f) \tag{12.26}$$

Dans le cadre de ce cours, elle est supposée non-périodique.

Cours signal et bruit :

Il existe un autre lien entre la densité spectrale de puissance et

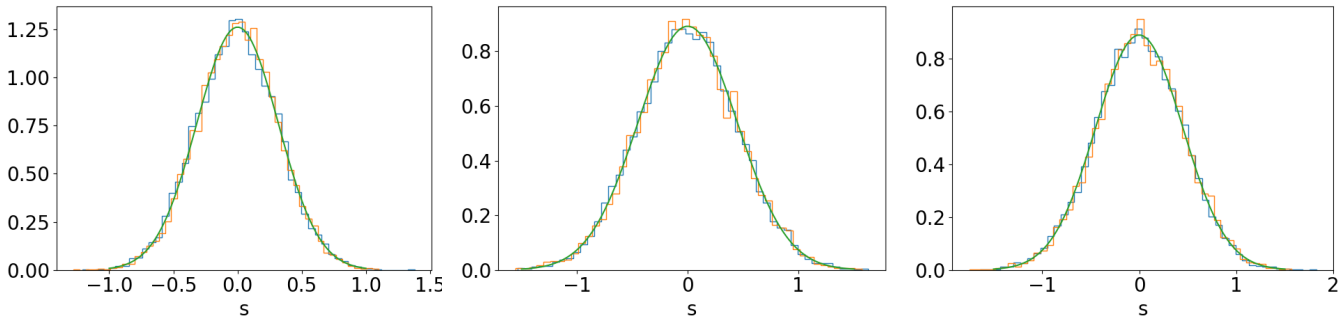

Figure 12.8: Densité de probabilité de 10^4 différentes trafectoires d'une part en utilisant $f_e = 10\mathrm{Hz}$ et $f_e = 100\mathrm{Hz}$; à gauche pour les valeurs de $s(0)$, au milieu pour $s(1) - s(0)$ et à droite pour $s(1) - s(-1)$.

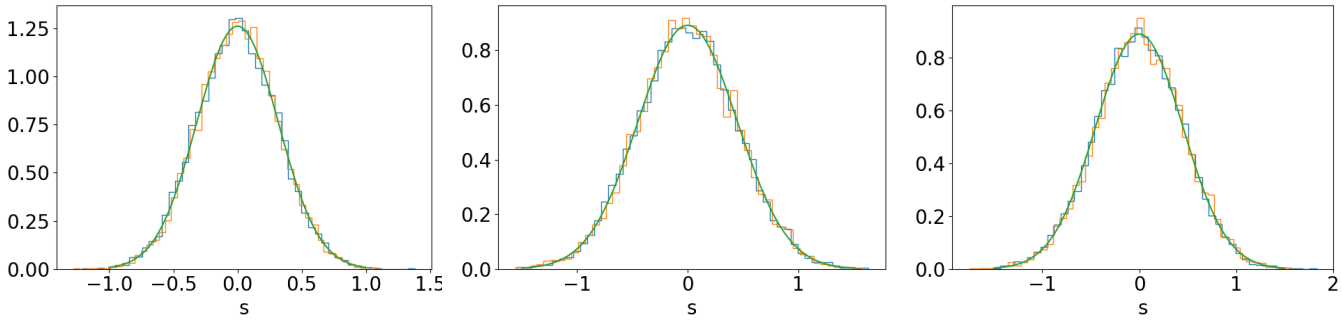

Figure 12.9: Densité de probabilité de 10^4 différentes trafectoires d'une part en utilisant $f_e = 10$Hz et $f_e = 100$Hz ; à gauche pour les valeurs de $s(0)$, au milieu pour $s(1) - s(0)$ et à droite pour $s(1) - s(-1)$.

L'implémentation dans **??**

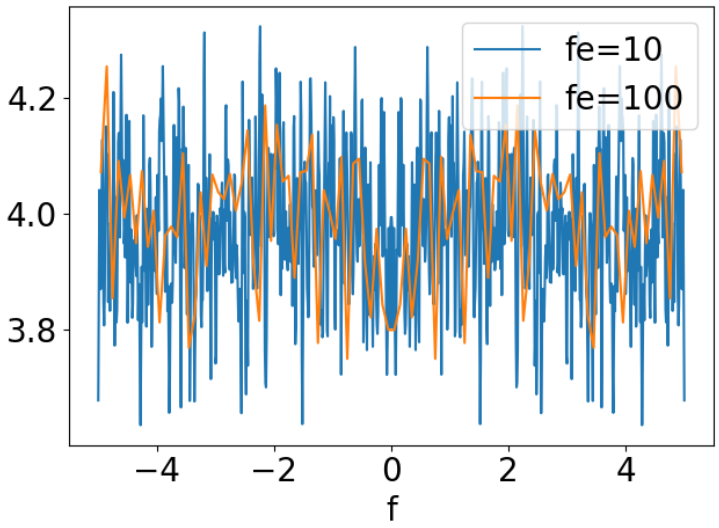

Figure 12.10

L'implémentation dans A.18

Figure 12.11

À partir d'une portion de réalisation $[t_0, t_0 + T]$ et une résolution f_e, on peut retrouver une approximation de la densité de probabilité, la moyenne statistique et la racine carré de la variance. Ceci est implémentée dans A.19.
On observe en effet que sur chaque graphique montré, il n'y a pas de corrélation entre un point et le suivant.
Pour un processus aléatoire $\overset{r}{X}(t)$, on peut calculer la transformée de Fourier de sa version tronquée et normalisée.

$$\widehat{\overset{r}{X}_T}(f) = \frac{1}{T}\mathrm{TF}\left[\overset{r}{X}(t)[\![-T/2 \leq t \leq T/2]\!]\right](f) = \frac{1}{T}\mathrm{TF}\left[\overset{r}{X}(t)\Pi\left(\frac{t}{T}\right)\right](f) = \frac{1}{T}\int_{-T/2}^{T/2} \overset{r}{X}(t)e^{-j2\pi ft}\,dt \tag{12.27}$$

C'est un processus aléatoire dépendant de la fréquence ayant donc des valeurs différentes pour chaque réalisation.
Rapport signal sur bruit

12.5 Bruit thermique (thermal noise)

$$Var(E)(f) = \frac{4kT}{R} \tag{12.28}$$

12.6 Bruit de grenaille (shot noise)

D'un point de vue électronique :

$$I_{\text{bruit}} = \sqrt{2eI_{DC}\Delta f} \tag{12.29}$$

avec eI_{DC} en unité de A^2/Hz.

12.7 Flicker noise

La variance de la tension est

$$Var(E) = \int_f \frac{K_e^2}{f^\alpha}\, df \tag{12.30}$$

avec $\alpha \approx 1$.

Chapter 13

Résumé du cours

13.1 Transformées de Fourier

Appendix A

Code pour simuler les différentes figures

A.1 Code commun

```
import sys
sys.path.append('c:/a/simu/seb/prg')
import os
os.chdir('c:/a/simu/seb/')
import seb
plt,np=seb.debut()
```

A.2 Codes pour simuler des signaux temps continus

Simulation de la figure 3.2

```
t=np.linspace(-2,4,10**3)
x_ex=np.exp(-t)*(t>=0)
f1=np.linspace(-2,2,10**3)
X1=1/(1+1j*2*np.pi*f1)
f2=np.linspace(-200,200,10**3)
X2=1/(1+1j*2*np.pi*f2)
x1=np.real(seb.TFI(f1,X1,t))
x2=np.real(seb.TFI(f2,X2,t))
fig,ax = plt.subplots()
ax.plot(t,x_ex,label='theorique')
ax.plot(t,x1,label='avec X(f)[|f|<=2](f)')
ax.plot(t,x2,label='avec X(f)[|f|<=200](f)')
ax.set_xlabel('t')
ax.legend()
plt.tight_layout()
fig.savefig('./figures/fig_C2_2_4a.png')
fig.show()

X = lambda f : 1/(1+1j*2*np.pi*f)
dX = lambda f,t,x : X(f)-seb.TF(t,x,f)
ech1_f = lambda fa,fb : np.linspace(max(0,fa-fb),fa+fb,10**2)
ech_f = lambda fa,fb : np.concatenate((np.flip(-ech1_f(fa,fb)),ech1_f(fa,fb)))
t=np.linspace(-2,4,10**3)
x_ex=np.exp(-t)*(t>=0)
f0 = np.linspace(-2*10**3,2*10**3,5000)
f0_a = 30; f0_b=30
f1 = ech_f(f0_a,f0_b)
x1 = np.real(seb.TFI(f1,X(f1),t))
```

```
#(f1_a,f1_b)=seb.argmax_fdf(f1,dX(f1,t,x1))
#f1_a = argmax_f(f0,X(f0))
f1_a = 6; f1_b=6
f2 = ech_f(f1_a,f1_b)
x2=np.real(seb.TFI(f2,dX(f2,t,x1),t))+x1
#(f2_a,f2_b)=seb.argmax_fdf(f2,dX(f2,t,x2))
#f2_a = argmax_f(f0,X(f0))
f2_a = 0.5; f2_b=0.5
f3 = ech_f(f2_a,f2_b)
x3=np.real(seb.TFI(f3,dX(f3,t,x2),t))+x2

fig,ax = plt.subplots()
ax.plot(t,x_ex,label='theorique')
str1=f"avec X(f)[||f|-{round(f0_a,1)}|<={round(f0_b,1)}](f)"
ax.plot(t,x1,'r-',label=str1)
str2=f"avec X(f)[||f|-{round(f1_a,1)}|<={round(f1_b,1)}](f)"
ax.plot(t,x2,'g-',label=str2)
str3=f"avec X(f)[||f|-{round(f2_a,1)}|<={round(f2_b,1)}](f)"
ax.plot(t,x3,'m-',label=str3)
ax.set_xlabel('t')
ax.legend()
plt.tight_layout()
fig.savefig('./figures/fig_C2_2_4b.png')
fig.show()


fig,ax = plt.subplots()
ax.plot(f0,abs(X(f0)),'b-',label='|X(f)|')
ax.plot(f0,abs(dX(f0,t,x1)),'r-',label='|X(f)-TF(t,x1)|')
ax.plot(f0,abs(dX(f0,t,x2)),'g-',label='|X(f)-TF(t,x2)|')
ax.plot(f0,abs(dX(f0,t,x3)),'m-',label='|X(f)-TF(t,x3)|')
plt.xlim(-25,25)
fig.legend()
ax.set_xlabel('f')
plt.tight_layout()
fig.savefig('./figures/fig_C2_2_4c.png')
fig.show()
```

Table A.1

Simulation de la figure **??**

```
T1=100; T2=500;  N=1e5;
t=linspace(0,T2,N);
u1=(t>=0).*(t<=T1); u2=(t>=0).*(t<=T2);
f=linspace(-10/T2,10/T2,N);
if ~exist('./donnees/fig_C2_2_3a.mat');
  U1=TF(t,u1,f);
  U2=TF(t,u2,f);
  save('./donnees/fig_C2_2_3a.mat','f','U1','U2');
else
  load('./donnees/fig_C2_2_3a.mat');
end
figure(1); plot(t,u1,t,u2); figure_jolie(1);  xlabel('t'); legend('u1(t)','u2(t)');
```

```
saveas(1,'./figures/fig_C2_2_3a.png');
figure(2); plot(f,abs(U1),f,abs(U2)); figure_jolie(2);  xlabel('t'); legend('U1(t)','U2(t)');
saveas(2,'./figures/fig_C2_2_3b.png');
RC=10;
V1=j*2*pi*f*RC./(1+j*2*pi*f*RC).*U1; V2=j*2*pi*f*RC./(1+j*2*pi*f*RC).*U2;
figure(3); plot(f,abs(V1),f,abs(V2)); figure_jolie(3);  xlabel('t');
saveas(3,'./figures/fig_C2_2_3c.png');
if ~exist('./donnees/fig_C2_2_3b.mat');
  v1=TFI(f,V1,t);
  v2=TFI(f,V2,t);
  save('./donnees/fig_C2_2_3b.mat','t','v1','v2');
else
  load('./donnees/fig_C2_2_3b.mat');
end
figure(4); plot(t,v1,t,v2); figure_jolie(4);  xlabel('t'); legend('v1(t)','v2(t)');
saveas(4,'./figures/fig_C2_2_3d.png');
```

Table A.2

Simulation de la figure 2.6

```
t=np.linspace(-3.1,3.1,10**2);
x=seb.fonction_T(t)
fig, ax = plt.subplots()
ax.plot(t,x)
ax.set_xlabel('t')
ax.set_ylabel('triangle(t)')
plt.tight_layout()
fig.savefig('./figures/fig_C2_5_2a.png');
fig.show()
fig, ax = plt.subplots()
y=seb.fonction_T(t/2)
ax.plot(t,y,label=r'triangle(t/2)')
z=seb.fonction_T(2*t)
ax.plot(t,z,label=r'triangle(2t)')
ax.set_xlabel('t')
ax.legend()
plt.tight_layout()
fig.savefig('./figures/fig_C2_5_2b.png')
fig.show()
```

Table A.3

Simulation de la figure 2.5

```
tx=linspace(-2,2,1e2);
x=fonction_echelon(tx);
ty=tx+1;
tz=tx-1.5;
figure(1); plot(tx,x,ty,x,tz,x); figure_jolie(1);
xlabel('t'); legend('x(t)','y(t)','z(t)');
saveas(1,'./figures/fig_C2_5_1a.png');
```

Table A.4

Simulation de la figure 2.4

```
%Echelon
t=linspace(-1.1,1.1,1e3);
a=(t>=0);
figure(1); plot(t,a); figure_jolie(1);
xlabel('t'); ylabel('a(t)'); axis([-1.1 1.1 -0.1 1.1]);
saveas(1,'./figures/fig_C2_3_1a.png');
%Porte
t=linspace(-1.1,1.1,1e3);
b=(abs(t)<=0.5);
figure(1); plot(t,b); figure_jolie(1);
xlabel('t'); ylabel('b(t)'); axis([-1.1 1.1 -0.1 1.1]);
saveas(1,'./figures/fig_C2_3_1b.png');
%demi-triangle croissant et decroissant
t=linspace(-1.1,1.1,1e3);
c1=(t+0.5).*(t>=-0.5).*(t<0.5);
c2=(0.5-t).*(t>=-0.5).*(t<0.5);
figure(1); plot(t,c1,t,c2); figure_jolie(1);
xlabel('t'); legend('c(t)','d(t)'); axis([-1.1 1.1 -0.1 1.1]);
saveas(1,'./figures/fig_C2_3_1c.png');
```

Simulation de la figure 2.4

```
%Echelon avancé/retardé
t=linspace(-1.1,1.1,1e3);
t0=0.2;
figure(1); plot(t,fonction_echelon(t-t0)); figure_jolie(1);
text(t0-0.1,+0.1,'t_0','fontsize',20);
xlabel('t'); ylabel('a(t-t_0)'); axis([-1.1 1.1 -0.1 1.1]);
saveas(1,'./figures/fig_C2_3_1d.png');
%Porte modifié
t=linspace(-1.1,1.1,1e3); t0=0.1; deltat=0.5;
b=fonction_porte((t-t0)/deltat); %(abs((t-t0)/deltat)<=0.5);
figure(1); plot(t,b); figure_jolie(1);
text(t0-deltat/2-0.1,+0.1,'t_1','fontsize',20);
text(t0+deltat/2+0.1,+0.1,'t_2','fontsize',20);
xlabel('t'); ylabel('b((t-0.5(t_1+t_2))/(t_2-t_1))'); axis([-1.1 1.1 -0.1 1.1]);
saveas(1,'./figures/fig_C2_3_1e.png');
%demi-triangle croissant et decroissant
t0=0.1; deltat=0.5;
c1=@(t)(t+0.5).*(t>=-0.5).*(t<0.5);
c2=@(t)(0.5-t).*(t>=-0.5).*(t<0.5);
t=linspace(-1.1,1.1,1e2);
figure(1); plot(t,fonction_C((t-t0)/deltat),t,fonction_D((t-t0)/deltat)); figure_jolie(1);
xlabel('t');
legend('c((t-0.5(t_1+t_2))/(t_2-t_1))','d((t-0.5(t_1+t_2))/(t_2-t_1))','location','northwest');
axis([-1.1 1.1 -0.1 1.1]);
text(t0-deltat/2-0.1,0.1,'t_1','fontsize',20);
text(t0+deltat/2+0.1,0.1,'t_2','fontsize',20);
saveas(1,'./figures/fig_C2_3_1f.png');
```

Table A.5

Simulation de la figure 1.4.

```
t=linspace(-5,5,500); %-5:1e-2:5;
dt=t(2)-t(1);
U=(t>=-1).*(t<=1);
x=-cumsum(U)*dt;
figure(1); plot(t,U,'g-',t,x,'b-');
figure_jolie(1);
xlabel('t'); legend('U(t)','x(t)');
saveas(1,'./figures/C2_1_fig12.png');
```

Table A.6: Simulation de la figure 1.1.

```
s=linspace(-5,5,100);
P=s.^2;
figure(1); plot(s,P); figure_jolie(1);
xlabel('s'); ylabel('P');
saveas(1,'./figures/fig_C1_2_1b.png');
```

Table A.7: Simulation des figures 3.3.

```
plt,np=seb.debut()
fig, ax = plt.subplots()
t=np.linspace(-5,5,10**3)
x1=seb.fonction_P(t)
ax.plot(t,x1,'b-',label='x1(t)=P(t)')
x2=seb.fonction_P(t/2)
ax.plot(t,x2,'r-',label='x2(t)=P(t/2)')
x3=seb.fonction_P(t/3)
ax.plot(t,x3,'g-',label='x3(t)=P(t/3)')
ax.set_xlabel('t')
ax.legend()
plt.tight_layout()
fig.savefig('./figures/fig_C2_2_fig2a.png');
fig.show()

f=np.linspace(-3,3,10**3)
f=np.delete(f,np.where(abs(f)<10**(-10)))
X1=np.sinc(f); X2=2*np.sinc(2*f); X3=3*np.sinc(3*f);
fig, ax = plt.subplots()
ax.plot(f,np.abs(X1),'b-',label='|X1(f)|')
ax.plot(f,np.abs(X2),'r-',label='|X2(f)|')
ax.plot(f,np.abs(X3),'g-',label='|X3(f)|')
ax.legend()
plt.tight_layout()
fig.savefig('./figures/C2_2_fig2b.png')
fig.show()

fig, ax = plt.subplots()
x4=seb.fonction_P(t-0.5)
ax.plot(t,x1,'b-',label='x1(t)')
ax.plot(t,x4,'r-',label='x4(t)=x1(t-0.5)')
ax.set_xlabel('t')
ax.legend(loc='center right')
plt.tight_layout()
fig.savefig('./figures/C2_2_fig2c.png')
fig.show()
```

```
fig, ax = plt.subplots()
X4=np.sinc(f)*np.exp(-1j*2*np.pi*f*0.5)
ax.set_xlabel('f')
ax.plot(f,abs(X1),'b-',label='|X1(f)|')
ax.plot(f,abs(X4),'r-',label='|X4(f)|')
ax.legend()
plt.tight_layout()
fig.savefig('./figures/C2_2_fig2d.png')
fig.show()

fig, ax = plt.subplots()
ax.set_xlabel('f')
ax.plot(f,np.angle(X1),'b-',label='arg(X1(f))')
ax.plot(f,np.angle(X4),'r-',label='arg(X4(f))')
ax.legend()
plt.tight_layout()
fig.savefig('./figures/C2_2_fig2e.png')
fig.show()
```

Simulation de l'équation (2.8).

```
t=linspace(-2,5,1e3);
x=isin(t,0,6).*exp(-t);
figure(1);
plot(t,x,'b-','linewidth',2);
set(gca,'fontsize',20);
saveas(1,'./figures/C1_1_fig2.png');
```

Simulation de la figure 1.2.

```
x=-5:1e-2:5;
y=x/2;
figure(1); plot(x,y,'b-','linewidth',2,[min(x),max(x)],[0 0],'k-','linewidth',2,[0 0],[min(y),max(y)],
set(gca,'fontsize',20);
axis equal
xlabel('x'); ylabel('y');
saveas(1,'./figures/C1_1_fig10.png');
```

Simulation de la figure 1.4.

```
x=-5:1e-2:5;
y=x+1.5;
figure(1); plot(x,y,'b-','linewidth',2,[min(x),max(x)],[0 0],'k-','linewidth',2,[0 0],[min(y),max(y)],
set(gca,'fontsize',20);
axis equal
xlabel('x'); ylabel('y');
saveas(1,'./figures/C1_1_fig12.png');
```

Simulation de la figure 1.5.

```
x=-5:1e-2:5;
y=abs(x);
figure(1); plot(x,y,'b-','linewidth',2,[min(x),max(x)],[0 0],'k-','linewidth',2,[0 0],[min(y),max(y)],
set(gca,'fontsize',20);
axis equal
xlabel('x'); ylabel('y');
saveas(1,'./figures/C1_1_fig13.png');
```

Simulation de l'équation (2.6).

```
t=linspace(-2,5,1e3);
x=(t>=-1)&(t<=1);
figure(1);
plot(t,x,'b-','linewidth',2,[min(t),max(t)],...
  [0 0],'k-','linewidth',2,[0 0],[min(x),max(x)],'k-','linewidth',2);
set(gca,'fontsize',20);
xlabel('t'); ylabel('x(t)');
saveas(1,'C:\A\SIMU\SEB\cours\C2_1_fig1.png');
```

Python :

Table A.8: Simulation de la figure 4.2

```
plt.close('all')
t=np.linspace(-2,2,10**3)
s=(t>=-1)*(t<=1)*(-2*t)
fig,ax = plt.subplots()
ax.plot(t,s)
ax.set_xlabel('t')
ax.set_ylabel('s(t)')
plt.tight_layout()
fig.savefig('./figures/C3_1_fig2a.png')
fig.show()

t=np.linspace(-4,4,10**3)
dsdt=(t>=-1)*(t<=1)*(-2);
t_,dsdt_=np.array([-1,1]),np.array([2,2])
fig,ax = plt.subplots()
ax.plot(t,dsdt,'r-')
ax.stem(t_,dsdt_,'b-',basefmt=" ")
ax.set_xlabel('t')
ax.set_xlim(-2,2)
ax.set_ylabel('ds/dt')
plt.tight_layout()
fig.savefig('./figures/C3_1_fig2b.png')
fig.show()



t=np.linspace(-4,4,10**3)
s=(t>=-1)*(t<=1)*(-2*t)
ind = np.argwhere(abs(np.diff(s))>0.1)
t_app_=(t[ind]+t[ind+1])/2
dsdt_app_=np.diff(s)[ind]
s_ech1 = dsdt_app_[0][0]*seb.retarder(t,seb.fonction_H(t),t_app_[0][0])
s_ech2 = dsdt_app_[1][0]*seb.retarder(t,seb.fonction_H(t),t_app_[1][0])
s1 = s-s_ech1-s_ech2
fig,ax = plt.subplots()
ax.plot(t,s,'b-',label='s(t)')
ax.plot(t,s1,'r-',label='s1(t)')
ax.set_xlim(-2,2)
ax.set_xlabel('t')
ax.legend()
```

```
plt.tight_layout()
fig.savefig('./figures/C3_1_fig2c.png')
fig.show()
```

Simulation de la figure 1.3.

```
x=linspace(-5,5,100);
y=x*(1+1/2);
figure(1); plot(x,y,'b-'); figure_jolie(1);
axis('equal');
xlabel('x'); ylabel('y');
saveas(1,'./figures/C1_1_fig11.png');
```

Simulation de la figure **??**.

```
figure(1);
t=linspace(-2,5,1e3);
x=(t>=0).*exp(-t);
figure(1);
plot([-0.1 1.1],[0 0],'k-','linewidth',2,[0 0],[-0.1 1.1],'k-','linewidth',2,...
  (t>=0),x,'bx','linewidth',2);
xlabel('U'); ylabel('x');
set(gca,'fontsize',20);
saveas(1,'./figures/C2_1_fig24a.png');
%%
t=-5:1e-2:5; dt=t(2)-t(1);
U=(t>=-1).*(t<=1);
x=-cumsum(U)*dt;
figure(2); plot([-0.1 1.1],[0 0],'k-','linewidth',2,...
  [0 0],[-2.1 1.1],'k-','linewidth',2,U,x,'gx','linewidth',2);
in(x)),max(max(U),max(x))],'k-','linewidth',2);
set(gca,'fontsize',20);
xlabel('U'); ylabel('x');
saveas(2,'./figures/C2_1_fig24b.png');
```

A.3 Codes pour simuler un spectre

Table A.9: Simulation de la figure 3.1

```
t=linspace(-3,3,1e3);
x=fonction_porte(t);
figure(1); plot(t,x); figure_jolie(1);
xlabel('t'); ylabel('x(t)');
saveas(1,'./figures/C2_2_fig1b.png');
f=linspace(-3,3,1e3);
X=TF(t,x,f);
figure(2); plot(f,abs(X)); figure_jolie(2);
xlabel('f'); ylabel('|X(f)|');
saveas(2,'./figures/C2_2_fig1c.png');
```

A.4 Codes pour simuler une réponse impulsionnelle

Table A.10: Simulation de la figure **??**

```
t=np.linspace(-3,3,10**3)
f1=np.linspace(-300,300,10**2)+10**(-3)
f2=np.linspace(-30,30,10**2)+10**(-4)
f3=np.linspace(-3,3,10**2)+10**(-5)
h_th=(t>=0)
H1=1/(1j*2*np.pi*f1)
H2=1/(1j*2*np.pi*f2)
H3=1/(1j*2*np.pi*f3)
h1=np.real(seb.TFI(f1,H1,t)); h1=h1-h1[0]
h2=np.real(seb.TFI(f2,H2-seb.TF(t,h1,f2),t))+h1; h2=h2-h2[0]
h3=np.real(seb.TFI(f3,H3-seb.TF(t,h2,f3),t))+h2; h3=h3-h3[0]
fig,ax = plt.subplots()
ax.plot(t,h_th,label='theorique')
ax.plot(t,h1,label='h1')
ax.plot(t,h2,label='h2')
ax.plot(t,h3,label='h3')
ax.set_xlabel('t')
ax.legend()
plt.tight_layout()
fig.savefig('./figures/fig_C5_2_fig1a.png')
fig.show()
```

A.5 Codes pour simuler des signaux temps continus et aléatoires

Simulation de la figure 12.1.

```
x=-5:1e-2:5; N=length(x);
sigma=0.5;
y=x/2; epsi1=(randn(1)-randn(1))*0.5*sigma; epsi2=(randn(1)-randn(1))*0.5*sigma;
figure(1); plot(x,y+epsi1,'b-','linewidth',2,x,y+epsi2,'b-','linewidth',2,[min(x),max(x)],...
  [0 0],'k-','linewidth',2,[0 0],[min(y),max(y)],'k-','linewidth',2);
set(gca,'fontsize',20);
axis equal
xlabel('x'); ylabel('y');
saveas(1,'./figures/C1_3_fig10.png');
x=-5:1e-2:5; N=length(x);
sigma=0.5;
y=x/2+0.5*sigma*(randn(1,N)-randn(1,N));
figure(1); plot(x,y,'b-','linewidth',2,[min(x),max(x)],[0 0],'k-','linewidth',2,[0 0],....
  [min(y),max(y)],'k-','linewidth',2);
set(gca,'fontsize',20);
axis equal
xlabel('x'); ylabel('y');
saveas(1,'./figures/C1_3_fig11.png');
```

Simulation de la figure 12.2

```
mu=2; sigma=1;
b_l=linspace(-3*sigma,3*sigma,1e2)+mu;
f_l=1/sqrt(2*pi)/sigma*exp(-0.5*(b_l-mu).^2/sigma^2);
figure(1);
plot(b_l,f_l,'linewidth',2,[min(b_l),max(b_l)],[0 0],'k-','linewidth',2,[0 0],....
  [min(f_l),max(f_l)],'k-','linewidth',2);
set(gca,'fontsize',20);
xlabel('b'); ylabel('f_B(b)');
```

```
saveas(1,'./figures/C1_3_fig12.png');

mu=2; sigma=1;
x=randn(1,50)*sigma+mu;
x_delimite=linspace(-3*sigma,3*sigma,20)+mu;
[f,x_centres]=histogramme(x,x_delimite);
figure(2);
bar(x_centres,f); axis([min(x_delimite)-0.1 max(x_delimite)+0.1 0 0.6]);
set(gca,'fontsize',20);
xlabel('x'); ylabel('f_B(x)');
saveas(2,'./figures/C1_3_fig13.png');


mu=2; sigma=1;
x=randn(1,500)*sigma+mu;
x_delimite=linspace(-3*sigma,3*sigma,20)+mu;
[f,x_centres]=histogramme(x,x_delimite);
figure(3);
bar(x_centres,f); axis([min(x_delimite)-0.1 max(x_delimite)+0.1 0 0.6]);
set(gca,'fontsize',20);
xlabel('x'); ylabel('f_B(x)');
saveas(3,'./figures/C1_3_fig14.png');
```

Simulation de l'équation (2.8).

```
t=linspace(-2,5,1e3);
x=(t>=0).*exp(-t);
figure(1);
plot(t,x,'b-','linewidth',2);
set(gca,'fontsize',20);
saveas(1,'C:\A\SIMU\SEB\cours\C1_1_fig2.png');
```

A.6 Simulation d'un filtrage

Table A.11: Simulation de la figure 7.1

```
t=linspace(-2,2,1e3);
u=isin(t,-1,1);
f=linspace(-3,3,1e3);
f=f(f~=0);
U=TF(t,u,f);
X=-U./(j*2*pi*f);
x=TFI(f,X,t);
x=x-x(1);
figure(1); plot([min(t),max(t)],[0 0],'k-','linewidth',2,[0 0],...
  [min(min(U),min(x)),max(max(U),max(x))],'k-','linewidth',2,...
  t,u,'g-','linewidth',2,t,x,'b-','linewidth',2);
legend('','','u','x');
set(gca,'fontsize',20);
xlabel('t');
saveas(1,'./figures/C4_1_fig2b.png');
```

Table A.12: Simulation de la figure 6.2

```
tx=np.linspace(-5,5,10**4)
```

```
Te=tx[1]-tx[0]
x=(tx>=-1)&(tx<=1)
th=np.arange(-5, 5, Te)
h=-1*(th>=0)
ty=np.arange(-3, 3, Te)
y=seb.convolution(tx,x,th,h,ty)

fig,ax = plt.subplots()
ax.plot(tx,x)
ax.set_xlabel('t')
ax.set_ylabel('x(t)')
plt.tight_layout()
fig.savefig('./figures/C5_3_fig1a.png')
fig.show()

fig,ax = plt.subplots()
ax.plot(th,h)
ax.set_xlabel('t')
ax.set_ylabel('h(t)')
plt.tight_layout()
fig.savefig('./figures/C5_3_fig1b.png')
fig.show()

fig,ax = plt.subplots()
ax.plot(ty,y)
ax.set_xlabel('t')
ax.set_ylabel('y(t)')
plt.tight_layout()
fig.savefig('./figures/C5_3_fig1c.png')
fig.show()
```

Table A.13: Simulation de la figure 9.1

```
t=np.linspace(-3,3,10**3)
x=np.cos(np.pi*t)
y=2/np.pi*np.sin(np.pi*t)
fig,ax = plt.subplots()
ax.plot(t,x,label='x_1(t)')
ax.plot(t,y,label='y_1(t)')
ax.set_xlabel('t')
ax.legend()
plt.tight_layout()
fig.savefig('./figures/fig_C11_1_1a.png')
fig.show()
t=np.linspace(-3,3,10**3)
x2=np.cos(np.pi*t)*(t>=0)
y2=2/np.pi*np.sin(np.pi*t)*(t>=0)
y3=2/np.pi*np.sin(np.pi*t)*(t>=0)-1/np.pi*np.sin(np.pi*t)*(t>=0)*(t<1)
fig,ax = plt.subplots()
ax.plot(t,x2,label='x_2(t)')
ax.plot(t,y2,label='y_1(t)*(t>=0)')
ax.plot(t,y3+0.02,label='y_1(t)*(t>=0)+y_3(t)')
ax.set_xlabel('t')
ax.legend(loc='lower left')
plt.tight_layout()
```

```
fig.savefig('./figures/fig_C11_1_1b.png')
fig.show()
```

Table A.14: Simulation de la figure 8.4

```
t=np.linspace(-2,2,10**3)
t2=seb.periodiser_ech_t(t,1)
x=seb.fonction_P(2*t2-0.5)
fig,ax = plt.subplots()
ax.plot(t,x,label='signal carré')
ax.set_xlabel('t')
ax.legend()
plt.tight_layout()
fig.savefig('./figures/fig_C9_2_2a.png')
fig.show()
x1=0.5*np.ones_like(t)
x2=2/np.pi*np.cos(2*np.pi*t-np.pi/2)
x3=2/np.pi/3*np.cos(6*np.pi*t-np.pi/2)
x4=2/np.pi/5*np.cos(10*np.pi*t-np.pi/2)
fig,ax = plt.subplots()
ax.plot(t,x1,label='x1')
ax.plot(t,x1+x2,label='x2')
ax.plot(t,x1+x2+x3,label='x3')
ax.plot(t,x1+x2+x3+x4,label='x4')
ax.set_xlabel('t')
ax.legend()
plt.tight_layout()
fig.savefig('./figures/fig_C9_2_2b.png')
fig.show()
```

Table A.15: Simulation de la figure 8.2

```
f0=1/pi;
t=linspace(-1,4,1e3);
x=cos(2*pi*f0*t);
figure(1); plot(t,x); figure_jolie(1);
xlabel('t'); ylabel('x(t)');
saveas(1,'./figures/C9_1_1_fig1a.png');
T=linspace(1,100,1e3); m=zeros(size(T)); P=zeros(size(T));
for T_=1:length(T)
  tT=linspace(-T(T_)/2,T(T_)/2,1e3);
  xT=cos(2*pi*f0*tT);
  m(T_)=TF(tT,xT,0)/T(T_);
  P(T_)=TF(tT,xT.^2,0)/T(T_);
end
figure(2); plot(T,m); figure_jolie(2);
xlabel('T'); ylabel('M_x(T)');
saveas(2,'./figures/C9_1_1_fig1b.png');
figure(3); plot(T,P,'b-',T,0.5*ones(size(T)),'r-'); figure_jolie(3);
xlabel('T'); ylabel('P_x(T)');
saveas(3,'./figures/C9_1_1_fig1c.png');
```

Table A.16: Simulation de la figure 12.4

```
t=np.linspace(-50,50,100)
```

```
x=np.random.normal(0,1,len(t))
t1=t[::8]; x1=x[::8]
fig,ax = plt.subplots()
ax.plot(t1,x1,'r+')
ax.set_xlabel('t')
ax.set_ylabel('Xt')
ax.legend()
plt.tight_layout()
fig.savefig('./figures/fig_C8_4_1a.png')
fig.show()

deb2=round(len(t)/4); fin2=round(len(t)*3/4)
t1=t[deb2:fin2:8]; x1=x[deb2:fin2:8]
t2=t[deb2:fin2:4]; x2=x[deb2:fin2:4]
fig,ax = plt.subplots()
ax.plot(t1,x1,'b-')
ax.plot(t2,x2,'r.')
ax.set_xlabel('t')
ax.set_ylabel('Xt')
ax.legend()
plt.tight_layout()
fig.savefig('./figures/fig_C8_4_1b.png')
fig.show()

deb3=round(3*len(t)/8); fin3=round(len(t)*5/8)
t2=t[deb3:fin3:4]; x2=x[deb3:fin3:4]
t3=t[deb3:fin3:2]; x3=x[deb3:fin3:2]
fig,ax = plt.subplots()
ax.plot(t2,x2,'b-')
ax.plot(t3,x3,'r.')
ax.set_xlabel('t')
ax.set_ylabel('Xt')
ax.legend()
plt.tight_layout()
fig.savefig('./figures/fig_C8_4_1c.png')
fig.show()

deb4=round(7*len(t)/16); fin4=round(len(t)*9/16)
t3=t[deb4:fin4:2]; x3=x[deb4:fin4:2]
t4=t[deb4:fin4]; x4=x[deb4:fin4]
fig,ax = plt.subplots()
ax.plot(t3,x3,'b-')
ax.plot(t4,x4,'r.')
ax.set_xlabel('t')
ax.set_ylabel('Xt')
ax.legend()
plt.tight_layout()
fig.savefig('./figures/fig_C8_4_1d.png')
fig.show()
```

Table A.17: Simulation de la figure 12.5

```
fe=10
tx=np.arange(-2,2,1/fe)
x1=np.random.normal(0,1,len(tx))
```

```
x2=np.random.normal(0,1,len(tx))
th=np.arange(0,1,1/fe)
h=np.ones(len(th))/fe
y1=seb.convolution(tx,x1,th,h,tx)*fe
y2=seb.convolution(tx,x2,th,h,tx)*fe
fig,ax=plt.subplots()
ax.plot(tx,y1)
ax.plot(tx,y2)
ax.set_xlabel('t')
plt.tight_layout()
fig.savefig('./figures/fig_C8_4_3a.png')
fig.show()
print(np.sum(y1**2)/len(y1),np.sum(y2**2)/len(y2))

fe=100
tx=np.arange(-2,2,1/fe)
x1=np.random.normal(0,1,len(tx))
x2=np.random.normal(0,1,len(tx))
th=np.arange(0,1,1/fe)
h=np.ones(len(th))/fe
y1=seb.convolution(tx,x1,th,h,tx)*fe
y2=seb.convolution(tx,x2,th,h,tx)*fe
fig,ax=plt.subplots()
ax.plot(tx,y1)
ax.plot(tx,y2)
ax.set_xlabel('t')
plt.tight_layout()
fig.savefig('./figures/fig_C8_4_3b.png')
fig.show()
print(np.sum(y1**2)/len(y1),np.sum(y2**2)/len(y2))
```

Table A.18: Simulation de la figure 12.10

```
import seb; plt,np=seb.debut();
plt.close('all')
fe=10
tx=np.arange(-2,2,1/fe)
x1=np.random.normal(0,1,len(tx))*np.sqrt(fe)
x2=np.random.normal(0,1,len(tx))*np.sqrt(fe)
th=np.arange(0,1,1/fe)
h=np.ones(len(th))/fe
y1=seb.convolution(tx,x1,th,h,tx)*fe
y2=seb.convolution(tx,x2,th,h,tx)*fe
fig,ax=plt.subplots()
ax.plot(tx,y1)
ax.plot(tx,y2)
ax.set_xlabel('t')
plt.tight_layout()
fig.savefig('./figures/fig_C8_4_3a.png')
fig.show()
print(np.sum(y1**2)/len(y1),np.sum(y2**2)/len(y2))

fe=100
tx=np.arange(-2,2,1/fe)
x1=np.random.normal(0,1,len(tx))*np.sqrt(fe)
```

```
x2=np.random.normal(0,1,len(tx))*np.sqrt(fe)
th=np.arange(0,1,1/fe)
h=np.ones(len(th))/fe
y1=seb.convolution(tx,x1,th,h,tx)*fe
y2=seb.convolution(tx,x2,th,h,tx)*fe
fig,ax=plt.subplots()
ax.plot(tx,y1)
ax.plot(tx,y2)
ax.set_xlabel('t')
plt.tight_layout()
fig.savefig('./figures/fig_C8_4_3b.png')
fig.show()
print(np.sum(y1**2)/len(y1),np.sum(y2**2)/len(y2))

def s_traj(fe):
  """realise 1 trajectoires de frequences fe"""
  tx=np.arange(-2,2,1/fe)
  x=np.random.normal(0,1,len(tx))
  th=np.arange(0,1,1/fe)
  h=np.ones(len(th))/fe
  y=seb.convolution(tx,x,th,h,tx)*fe
  return tx,y
K=10**2
fe_l=np.linspace(5,300,K)
std_l=np.zeros(K)
for k in range(K):
  t,s=s_traj(fe_l[k])
  std_l[k]=np.std(s)
fig,ax = plt.subplots()
ax.plot(fe_l,std_l**2)
ax.plot(fe_l,1/fe_l)
ax.set_xlabel('fe')
plt.tight_layout()
fig.savefig('./figures/fig_C8_4_3c.png')
fig.show()
```

Table A.19: Simulation de la figure 12.11

```
if exist('./donnees/data_C8_4_1.mat')
  load('./donnees/data_C8_4_1.mat','t','x');
else disp('Réaliser la simulation~\ref{ver:fig_C8_4_1}'),
end
t1=linspace(-15,-10,1e4+1); fe=1/(t1(2)-t1(1)),
x1=interpoler(t,x,t1);
x_delimite=linspace(-0.1,0.1,30);
[f1,x_centres]=histogramme(x1,x_delimite);
t2=linspace(5,10,1e4+1); fe=1/(t2(2)-t2(1)),
x2=interpoler(t,x,t2);
x_delimite=linspace(-0.1,0.1,30);
[f2,x_centres]=histogramme(x2,x_delimite);
t3=linspace(0,1,1e4+1); fe=1/(t3(2)-t3(1)),
x3=interpoler(t,x,t3);
x_delimite=linspace(-0.1,0.1,30);
[f3,x_centres]=histogramme(x3,x_delimite);
t4=linspace(-20,20,1e4+1); fe=1/(t4(2)-t4(1)),
```

```
x4=interpoler(t,x,t4);
x_delimite=linspace(-0.1,0.1,30);
[f4,x_centres]=histogramme(x4,x_delimite);
figure(1);
plot(x_centres,f1,'b-',x_centres,f2,'r-',x_centres,f3,'g-',x_centres,f4,'m-');
figure_jolie(1);
xlabel('x'); ylabel('f_x(x)');
legend('t_0=-15, T=5, f_e=2000','t_0=5, T=5, f_e=2000',...
  't_0=0, T=1, f_e=10^4','t_0=-15, T=1, f_e=250');
saveas(1,'./figures/C8_4_2a.png');
t0=-10;
T_l=linspace(0.1,20,201);
moy=zeros(size(T_l)); P=zeros(size(T_l));
for k=1:length(T_l)
  t1=linspace(t0,t0+T_l(k),1e4);
  x1=interpoler(t,x,t1);
  moy(k)=TF(t1,x1,0)/T_l(k);
  P(k)=TF(t1,x1.^2,0)/T_l(k);
  disp(['k=',num2str(k)])
end
figure(2); plot(T_l,moy,T_l,P); figure_jolie(2);
xlabel('T'); legend('M_x','P_x');
saveas(2,'./figures/C8_4_2b.png');
t0=-10;
fe_l=linspace(200,10^4,301);
moy=zeros(size(fe_l)); P=zeros(size(fe_l));
for k=1:length(fe_l)
  T=1e4/fe_l(k);
  t1=linspace(t0,t0+T,1e4+1);
  x1=interpoler(t,x,t1);
  moy(k)=TF(t1,x1,0)/T;
  P(k)=TF(t1,x1.^2,0)/T;
  disp(['k=',num2str(k)])
end
figure(3); plot(fe_l,moy,fe_l,P); figure_jolie(3);
xlabel('fe'); legend('M_x','P_x');
saveas(3,'./figures/C8_4_2c.png');
```

A.7 Code pour simuler la solution d'une équation différentielle

Table A.20: Simulation de la figure 5.2

```
t=np.linspace(-2,10,10**3)
y_th=(t>=0)*np.sin(t)
y_num=seb.sol_eq_diff((1,0,1),t)
fig,ax = plt.subplots()
ax.plot(t,y_th,label='theorique')
ax.plot(t,y_num,label='approchee')
ax.set_xlabel('t')
ax.legend()
plt.tight_layout()
fig.savefig('./figures/fig_C13_3_fig1a.png')
fig.show()
fig,ax = plt.subplots()
```

```
ax.plot(t,y_th-y_num)
ax.set_xlabel('t')
ax.set_ylabel('difference')
plt.tight_layout()
fig.savefig('./figures/fig_C13_3_fig1b.png')
fig.show()
```

Python :

Table A.21: Implémentation de la figure 8.5.

```
import seb; plt,np=seb.debut();
def s1(t):
  return seb.fonction_T(t/2)*2-seb.fonction_T(t)
def S1_th(f):
  return 4*(np.sinc(2*f)**2)-(np.sinc(f)**2)
T=5
t1=np.linspace(-3*T,3*T,10**3)
t2=seb.periodiser_ech_t(t1,(-2.5,2.5))
plt.close('all')
fig,ax=plt.subplots()
ax.plot(t1,s1(t1),label='s1(t)')
ax.plot(t1,s1(t2)+0.01,label='s1P(t)')
ax.set_xlabel('t')
ax.legend()
plt.tight_layout()
fig.savefig('./figures/fig_TP2_fig8a.png')
fig.show()
```

Python :

Table A.22: Implémentation de la figure 8.7.

```
import seb; plt,np=seb.debut();
def s1(t):
  return seb.fonction_T(t/2)*2-seb.fonction_T(t)
T=5; N=10; fe=N/T
t=seb.synchroniser(np.arange(-T/2,T/2,1/fe))
assert len(t)==N
assert max(t)<T/2, ('il faut eviter que le signal contienne deux fois la premiere valeur',max(t),T/2)
s1DP=s1(t)
f,S1DP=seb.TFD(t,s1DP,(-T/2,T/2),True)
ta=seb.synchroniser(np.arange(-T/2,T/2,1/fe))
S1e=lambda f:seb.TFTD(ta,s1(ta),f)
f2=np.linspace(-fe/2,fe/2,10**3)

plt.close('all')
t1=np.linspace(-T,T,10**3)
t2=np.arange(-T,T,1/fe)
t3=seb.periodiser_ech_t(t2,(-T/2,T/2))
print('fe=',fe,' S1[0]=',np.sum(s1(t1))*(t1[1]-t1[0]),' S1D[0]=',np.sum(s1(t2)),' S1P[0]=',
  np.sum(s1(seb.periodiser_ech_t(t1,(-T/2,T/2))))/len(t1),' S1DP[0]=',np.sum(s1(t3))/len(t3)
)
fig,ax=plt.subplots()
ax.plot(t1,s1(t1),'r:')
```

```
ax.plot(t1,s1(seb.periodiser_ech_t(t1,(-T/2,T/2))),'g:')
for t_ in range(len(t2)):
  ax.plot([t2[t_],t2[t_]],[0,s1(t1[t_])],'b-')
  ax.plot([t2[t_],t2[t_]],[0,s1(t3[t_])],'g-')
ax.set_xlabel('t')
# ax.legend()
plt.tight_layout()
fig.savefig('./figures/fig_TP2_fig10a.png')
fig.show()

fig,ax=plt.subplots()
for f_ in range(len(f)):
  ax.plot([f[f_],f[f_]],[0,np.abs(S1DP[f_])],'b-')

ax.plot(f2,np.abs(S1e(f2))/N,'r:')
ax.set_xlabel('f')
plt.tight_layout()
fig.savefig('./figures/fig_TP2_fig10b.png')
fig.show()
```

Python :

Table A.23: Implémentation de la figure 8.6.

```
import seb; plt,np=seb.debut();
def s1(t):
  return seb.fonction_T(t/2)*2-seb.fonction_T(t)
def S1_th(f):
  return 4*(np.sinc(2*f)**2)-(np.sinc(f)**2)
T=5
t=np.linspace(-T/2,T/2,10**3)
k=np.arange(-5,5,1)
f,S1P=seb.coef_serie_Fourier(t,s1(t),(-T/2,T/2),k)
plt.close('all')
fig,ax=plt.subplots()
for f_ in range(len(f)):
  ax.plot([f[f_],f[f_]],[0,np.abs(S1P[f_])],'b-')
ax.plot(f,np.abs(S1_th(f))/T,'r+')
ax.set_xlabel('f')
# ax.legend()
plt.tight_layout()
fig.savefig('./figures/fig_TP2_fig9a.png')
fig.show()
```

A.8 Outils

Python :

Lignes à mettre en début de séance

Voir la section 1.2

```
import sys
sys.path.append('rep_prg')
import os
os.chdir('rep_tra')
```

```
import seb
plt,np=seb.debut()
```

`seb.debut`

La fonction `debut` renvoie un tuple `(plt,np,sig)`.

- `plt` sert pour faire des graphes et ses paramètres sont modifiés par `debut` de façon à rendre les graphes plus visibles.
- `np` correspond à `numpy`, c'est utile pour faire des calculs.

A.8.1 Commandes générales

- `help` suivi de parenthèses avec le nom de la fonction. Le contenu affiché est celui entre trois guillemets dans une fonction.
- `assert` provoque une erreur si ce qui suit n'est pas vrai.
- `len` suivi de parenthèses et le nom d'un vecteur ligne ou colonne, cela donne sa longueur.
- `1j` est le complexe imaginaire j.
- `round` qui fournit [1] un entier à partir d'un nombre réel.
- `def` et `return` pour définir une fonction.
- `#` pour mettre une ligne en commentaire.
- `type` pour connaître le type d'une valeur ou d'une variable.
- `min` et `max`

Contenu susceptible d'être utilisé dans `np` pour `numpy`

- `linspace`
- `arange`
- `sqrt` pour racine carré
- `exp` pour exponentielle
- `array` suivi d'une liste entre crochets de valeurs espacées de virgules, pour définir un vecteur de type `nmupy.ndarray`. On peut l'utiliser pour définir une matrice en utilisant deux séries de crochets.
- `real` suivi d'un complexe entre parenthèses pour prendre la partie réelle.
- `abs` suivi d'un complexe ou d'un réel entre parenthèses pour prendre le module ou la valeur absolue.
- `sinc`
- `concatenate`
- `zeros`, `ones`, `zeros_like` et `ones_like`

[1] La fonction `round` de `numpy` fournit un entier de type `float`.

Contenu susceptible d'être utilisé dans `plt` récupéré dans `debut`

Voir la section 1.2

- `subplots`
- `plot`
- `set_xlabel` et `set_ylabel`
- `set_legend`
- `tight_layout`
- `savefig`
- `show`

Contenu susceptible d'être utilisé dans `seb`

- `fonction_echelon`
- `fonction_P`
- `TF` et `TFI` pour transformée de Fourier et transformée de Fourier inverse
-

Contenu susceptible d'être utilisé dans `sympy`

- `Symbol`
- `solve`
- `simplify`
- `matrices.Matrix`

Appendix B

Quelques outils pour calculer l'intensité ou la tension

B.1 Association de résistances

Les résistances et impédances en série sont équivalentes à une résistance ou une impédance, celle-ci étant la somme. Les résistances et impédances en parallèles sont équivalentes à une résistance ou une impédance, celle-ci étant l'inverse de la somme des inverse. Ceci est illustré sur la figure B.1. On remarque à ce propos que

$$\frac{Z_1 Z_2}{Z_1 + Z_2} = \frac{1}{\frac{1}{Z_1} + \frac{1}{Z_2}} \tag{B.1}$$

B.2 Diviseur de tension

La figure B.2 illustre le diviser de tension.

B.3 Théorème de Millman

B.4 Quadripôle

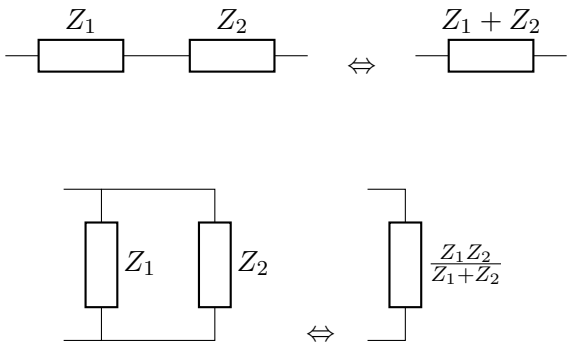

Figure B.1: Illustration des calculs d'impédances en série et en parallèles

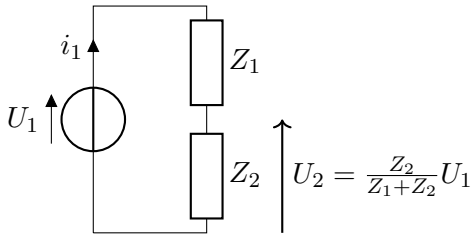

Figure B.2: Diviseur de tension

Appendix C

Notations utilisées

C.1 Notations utilisées dans le cours signal et bruit

- T_x période du signal $x(t)$
- P_x puissance du signal $x(t)$
- E_x énergie du signal $x(t)$
- M_x moyenne du signal $x(t)$
- A_x somme du signal $x(t)$
- $\Pi(t) = [\![t \in [-0.5, 0.5]]\!](t)$ est la porte centrée de largeur 1.

C.2 Notations utilisées dans en ce qui concerne l'électronique

-

Un dipôle est ici un composant électronique ayant deux bornes.