Information, Calcul et Communication: Cours 9

Résumé de la dernière leçon:

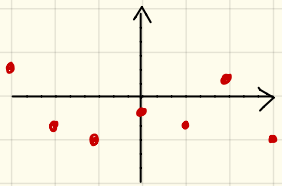
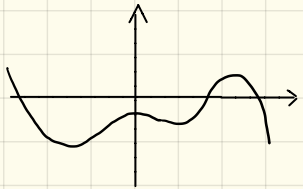
- Signaux:
  - Tout signal est une somme de sinusoïdes
  - Bande passante: $B = \max(f_1, \ldots, f_n)$
- Echantillonnage: condition nécessaire $f_e > 2B$

Plan de cette leçon:

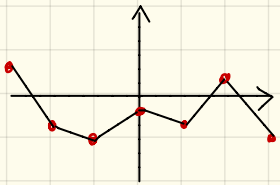
- Reconstruction de signaux, théorème d'échantillonnage
- Entropie ($\longrightarrow$ compression de données)

Algorithme de reconstruction de signaux

Comment reconstruire un signal $(X(t), t \in \mathbb{R})$ à partir de sa version échantillonnée $(X(nT_e), n \in \mathbb{Z})$? C'est-à-dire, comment passer

de  à  ?

Relier les points avec des droites ? pas une bonne idée !

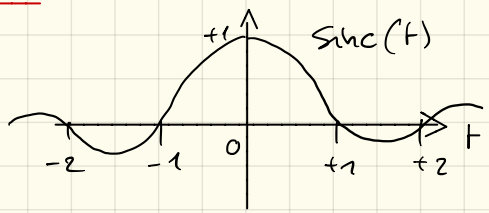


Un signal avec des "coins"

Sonne très mal en vrai ...

Une fonction magique : la fonction sinc !

$$\mathrm{sinc}(t) = \begin{cases} 1 & \text{si } t = 0 \\ \dfrac{\sin(\pi t)}{\pi t} & \text{si } t \neq 0 \end{cases}$$



Remarquez : $\mathrm{sinc}(0) = 1$, $\mathrm{sinc}(n) = 0$ $\forall n \in \mathbb{Z}$ différent de 0

Aussi : $$\mathrm{sinc}(t) = 2\int_0^{1/2} \cos(2\pi f t)\, df$$

- sinc est donc une somme (au intégrale) de sinusoïdes de fréquences f allant de 0 à $\frac{1}{2}$
- mieux que ça : c'est un signal qui contient toutes les fréquences f de 0 à $\frac{1}{2}$!

De même, $\underbrace{\text{sinc}(f_e t)}_{} = \frac{2}{f_e} \int_0^{f_e/2} \cos(2\pi f t)\, df$

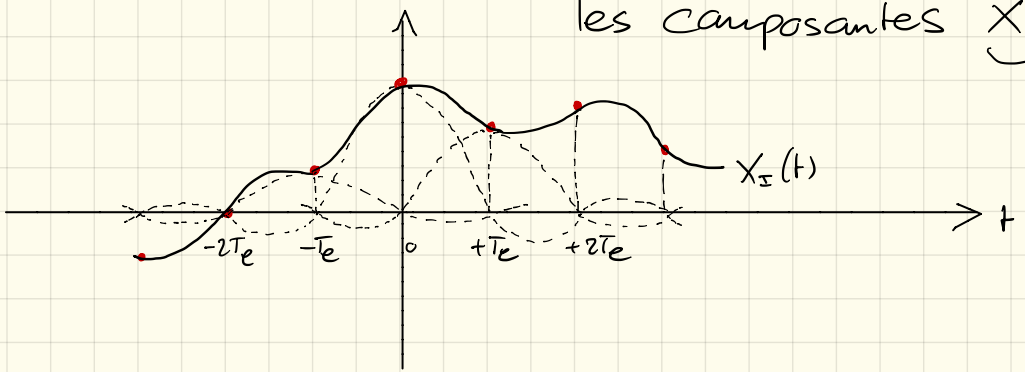
ce signal contient toutes les fréquences de 0 à $\frac{f_e}{2}$;
c'est donc un bon candidat pour reconstruire
un signal de bande passante $B < \frac{f_e}{2}$!

Formule d'interpolation

Soient $(X(nT_e), n \in \mathbb{Z})$ les valeurs échantillonnées à fréquence $f_e = \frac{1}{T_e}$ d'un signal X. On définit :

$$X_I(t) = \sum_{n \in \mathbb{Z}} \underbrace{X(nT_e)\, \text{sinc}(f_e t - n)}_{\begin{cases} = X(nT_e) & \text{si } t = nT_e \\ = 0 & \text{si } t = mT_e,\ m \neq n \end{cases}}, \quad t \in \mathbb{R}$$

Illustration: La fonction $X_I(t)$ est la somme de toutes les composantes $\underbrace{X(nT_e)\,\mathrm{sinc}(f_e t - n)}_{\text{centrée en } t = nT_e}$.



Théorème d'échantillonnage (Nyquist 1928, Shannon 1949, & al.)

Si la bande passante B du signal d'origine X est plus petite que $\frac{f_e}{2}$, alors $X_I(t) = X(t)\ \forall t \in \mathbb{R}$, i.e., la formule d'interpolation permet de reconstruire parfaitement le signal d'origine X.

Sous-échantillonnage

Et si $B \geq \frac{f_e}{2}$, que faire pour éviter l'effet stroboscopique?

Solution 1 (coûteuse, parfois impossible !)

Augmenter f_e jusqu'à ce que $B < \frac{f_e}{2}$.

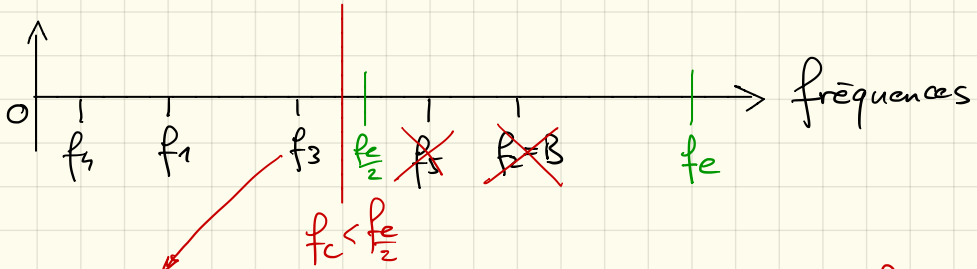
Mais il arrive qu'on ne puisse pas changer f_e, à moins de changer d'appareil de mesure; il arrive aussi que $B = +\infty$! (ou B très grande en pratique)

Solution 2

Réduire B jusqu'à ce que $B < \frac{f_e}{2}$!

Comment ? En filtrant le signal d'origine X avant de l'échantillonner, i.e., en supprimant toutes les composantes du signal de fréquence plus grande qu'une fréquence de coupure $f_c < \frac{f_e}{2}$.

"En image" :



Inconvénient :

On perd ainsi les plus hautes fréquences du signal X (ceci dit, celles-ci contiennent souvent du bruit...)

Avantage :

On évite ainsi l'effet stroboscopique, source de distorsion (par rappel, lorsque $B > \frac{f_e}{2}$, le signal reconstruit ne contient pas les mêmes fréquences que le signal d'origine).

Illustration avec un morceau de jazz (cf. Moodle)

Compression de données

Buts : { réduire l'espace de stockage des données
réduire le temps de transmission des données

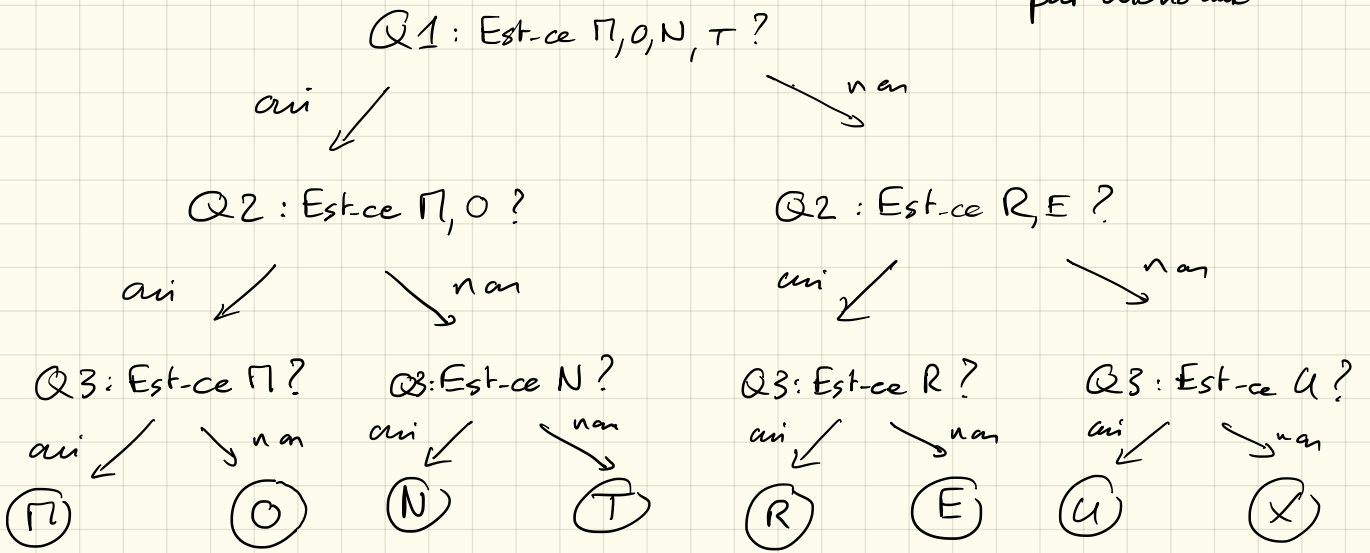
Préliminaire : entropie

Introduction : jeu des questions

- Je tire une lettre uniformément au hasard dans une séquence de lettres (un mot).
- Votre but est de deviner quelle lettre a été tirée en posant un minimum de questions binaires (i.e., de questions auxquelles je ne peux répondre que par oui ou par non).

Exemple 1 : MONTREUX (8 lettres)

Ici, toutes les lettres sont différentes → algorithme de recherche par dichotomie !

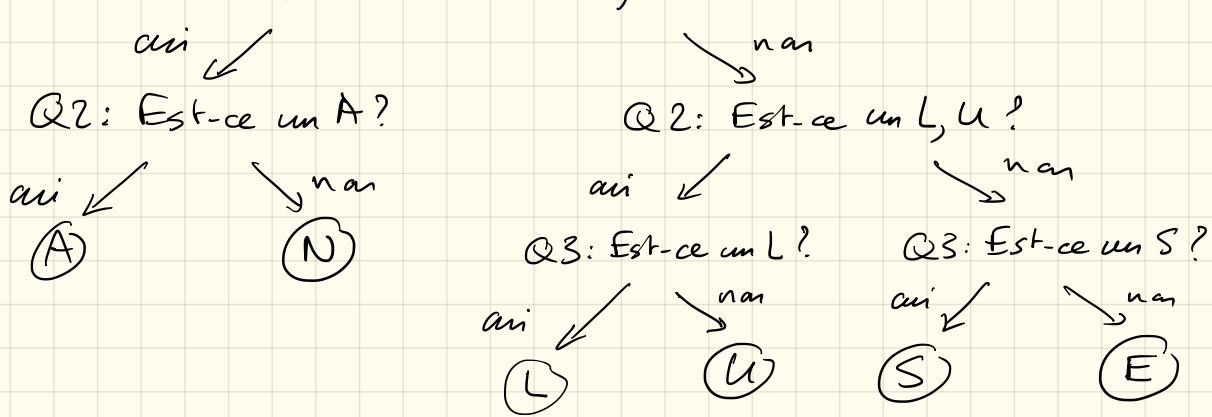


Dans tous les cas, 3 questions suffisent pour deviner la lettre.

On dit que l'entropie de cette séquence vaut 3 (notez que $3 = \log_2 8$).

Exemple 2 : L AUSANNE (encore 8 lettres)

Ici, des lettres se répètent (A et N). Notez que vous n'avez pas besoin de deviner la position de la lettre, mais juste sa valeur. Voici les questions optimales à poser :



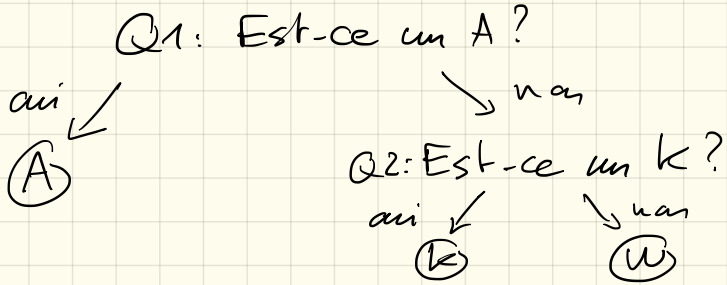
Pour les lettres plus fréquentes, on a besoin de moins de questions !

Plus précisément : A et N apparaissent chacune 2 fois sur 8, i.e., 1 fois sur 4, et on a besoin de 2 questions pour deviner ces lettres ; notez que $2 = \log_2(4)$. Tandis qu'on a toujours besoin de 3 questions pour deviner les lettres L, S, U, E, qui apparaissent 1 fois sur 8 (et $3 = \log_2 8$)

En moyenne (rappelez-vous que le dé tire une lettre uniformément au hasard parmi les 8 lettres que forment le mot), on a donc besoin ici de $\frac{1}{2} \cdot 2 + \frac{1}{2} \cdot 3$ $= 2.5$ questions pour deviner une lettre.
On dit que l'entropie de cette séquence vaut 2.5.

Exemple 3 : KAWAKAWA (toujours 8 lettres, ville en Nouvelle-Zélande)

Voici les questions optimales à poser ici :



Ici, le A apparaît 4 fois sur 8, i.e., 1 fois sur 2, et on a besoin d'une question pour deviner cette lettre (remarquez à nouveau que $1 = \log_2(2)$ ...).

En moyenne, le nombre de questions à poser est $\frac{1}{2} \cdot 1 + \frac{1}{2} \cdot 2 = 1.5$

On dit que l'entropie de cette séquence vaut 1.5.

Avec ces 3 exemples (savamment choisis pour que tout fonctionne bien...), on obtient en résumé :

probabilité d'apparition d'une lettre :	nombre de questions à poser pour deviner celle-ci :
$1/8$	$3 = \log_2 8$
$1/4$	$2 = \log_2 4$
$1/2$	$1 = \log_2 2$

Le nombre de questions à poser pour deviner une lettre apparaissant avec probabilité p vaut donc $\log_2\left(\frac{1}{p}\right)$ dans ces 3 exemples.

Définition générale de l'entropie

Soit X une séquence de lettres provenant d'un alphabet $A = \{a_1, \dots, a_n\}$

Soit p_j la probabilité d'apparition de la lettre a_j dans la séquence X $\left(0 \leq p_j \leq 1 , \sum_{j=1}^{n} p_j = 1 \right)$

L'entropie de la séquence X est définie par

$$H(X) = \sum_{j=1}^{n} p_j \log_2 \left(\frac{1}{p_j} \right)$$

Notes:
- l'entropie ne dépend pas des valeurs des lettres a_j
- si $p_j = 0$, alors par convention, on pose $p_j \log_2 \left(\frac{1}{p_j} \right) = 0$
- $\log_2 \left(\frac{1}{p_j} \right)$ n'est pas forcément un nombre entier en général !

Interprétations :

- En physique (Clausius, Boltzmann) : L'entropie est une mesure de la quantité de désordre contenue dans un système.
- En informatique (Shannon) : L'entropie est une mesure de la quantité d'information contenue dans des données.

$$H(\text{KAWAKAWA}) < H(\text{LAUSANNE}) < H(\text{MONTREUX})$$
$$1.5 \qquad\qquad 2.5 \qquad\qquad 3$$

Plus H augmente, plus il y a de "désordre", plus il y a aussi d'informations à enregistrer (et plus il y a aussi de possibilités d'écrire des mots différents).

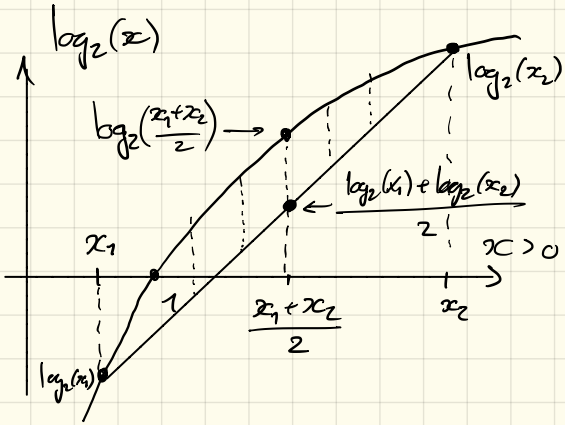
Propriétés :

- $0 \leq p_j \leq 1 \Rightarrow \log_2\left(\frac{1}{p_j}\right) \geq 0 \Rightarrow H(X) = \sum_{j=1}^{n} p_j \log_2\left(\frac{1}{p_j}\right) \geq 0$

- $H(X) \leq \log_2(n)$, où n est la taille de l'alphabet :

  rappel : $\log_2(x)$ est concave

  ie. $\frac{\log_2(x_1) + \log_2(x_2)}{2} \leq \log_2\left(\frac{x_1 + x_2}{2}\right)$



  Plus généralement :

  $p_1 \log_2(x_1) + p_2 \log_2(x_2) \leq \log_2(p_1 x_1 + p_2 x_2)$

  où $p_1, p_2 \geq 0$ et $p_1 + p_2 = 1$

Plus généralement encore : si $p_j \geq 0$ et $\sum_{j=1}^{n} p_j = 1$, alors

$$\sum_{j=1}^{n} p_j \log_2(x_j) \leq \log_2\left(\sum_{j=1}^{n} p_j x_j\right) \qquad \forall x_1 \dots x_n > 0$$

Utilisons cette inégalité avec $x_j = \frac{1}{p_j}$:

$$H(X) = \sum_{j=1}^{n} p_j \log_2\left(\frac{1}{p_j}\right) \leq \log_2\left(\underbrace{\sum_{j=1}^{n} \underbrace{p_j \frac{1}{p_j}}_{=1}}_{=n}\right) = \log_2(n) \qquad \#$$

Notez encore : • $H(X) = 0$ ssi toutes les lettres de la séquence sont identiques

• $H(X) = \log_2 n$ ssi " " différentes

A quoi tout cela peut-il bien servir ???

⟶ semaine prochaine (compression de données)