L'ENTROPIE DE SHANNON

La quantité d'information moyenne d'une source de données

Niveau : Terminale générale, Maths complémentaires. Approfondissement.
Lien avec le programme : Thème « **Fonction logarithme népérien** » et « **Variables aléatoires : espérance** ». Ce thème peut s'étendre à la loi binomiale (correction d'erreurs), aux probabilités conditionnelles (arbres de décision) et à la programmation en Python.
Lien avec Les maths au quotidien : Codage/IA/Société/Santé/Loisirs

1. Le problème que Shannon voulait résoudre

En 1948, l'ingénieur américain Claude Shannon travaille pour les laboratoires Bell, qui cherchent à transmettre des messages (téléphone, télégraphe) le plus efficacement possible. Il se pose une question qui paraît simple mais qui n'avait jamais été posée aussi précisément :

> *Comment mesurer, avec un nombre, la quantité d'information contenue dans un message ?*

Avant Shannon, « information » était une notion floue, presque littéraire. Shannon en fait une grandeur mathématique, aussi précise que la masse en kilogrammes ou la température en degrés : c'est la naissance de la théorie de l'information. Sa réponse tient en une idée : **un message est informatif dans la mesure où il lève une incertitude**. Apprendre qu'un événement très improbable s'est produit nous apprend beaucoup ; apprendre qu'un événement certain s'est produit ne nous apprend rien. On mesure donc l'information apportée par un événement par sa « surprise », d'autant plus grande que l'événement était improbable. Pour une source qui produit des messages au hasard, l'incertitude levée **en moyenne** à chaque message reçu est une **espérance** : c'est ce que Shannon appelle l'**entropie**.

> **À la fin de ce document, tu sauras :** mesurer la quantité d'information d'un événement ; calculer l'entropie d'une source et l'interpréter comme une espérance ; construire un code de Huffman ; relier l'entropie à la compression de fichiers, au débit d'un câble, à la solidité d'un mot de passe et à l'apprentissage automatique.

2. Première idée : informer, c'est lever une incertitude

Imaginons deux phrases qu'on vous annonce :

- Phrase A : « Le soleil se lèvera demain matin. »
- Phrase B : « Il a neigé au Sahara aujourd'hui. »

La phrase A ne vous apprend presque rien : vous le saviez déjà, c'est certain. La phrase B, en revanche, vous surprend énormément : c'est rare, improbable. Shannon part de cette idée toute simple :

> *Un événement rare apporte beaucoup d'information. Un événement certain n'en apporte aucune.*

2.1 Mesurer la quantité d'information d'un événement

Il faut donc associer à un événement de probabilité p un nombre $I(p)$, sa quantité d'information, qui vérifie trois propriétés :

- elle diminue quand la probabilité augmente (rare = beaucoup d'information) ;
- elle vaut 0 quand la probabilité vaut 1 (un événement certain n'apporte rien) ;
- elle est **additive** pour des événements indépendants : si deux événements indépendants ont pour probabilités p_1 et p_2, la probabilité que les deux se produisent est le produit $p_1 \times p_2$, et l'on veut que les quantités d'information s'additionnent : $I(p_1 \times p_2) = I(p_1) + I(p_2)$.

Or la propriété $\ln(ab) = \ln a + \ln b$ du logarithme népérien transforme justement les produits en sommes. Shannon définit donc la quantité d'information d'un événement de probabilité p par :

$$I(p) = -\log_2(p) = \log_2\left(\frac{1}{p}\right)$$

où $\log_2$ est le **logarithme en base 2**, que l'on calcule avec le logarithme népérien (touche « ln » de la calculatrice) :

$$\log_2(x) = \frac{\ln(x)}{\ln(2)} \qquad \text{donc} \qquad I(p) = -\frac{\ln(p)}{\ln(2)} = \frac{\ln\left(\frac{1}{p}\right)}{\ln(2)}$$

Vérifions les propriétés. Pour $p = 1$: $I(1) = -\frac{\ln 1}{\ln 2} = 0$. Pour l'additivité :

$$I(p_1 p_2) = -\frac{\ln(p_1 p_2)}{\ln 2} = -\frac{\ln p_1 + \ln p_2}{\ln 2} = I(p_1) + I(p_2)$$

La base 2 est choisie pour que la quantité d'information se mesure en **bits**, l'unité de l'informatique (0 ou 1). Comme p est compris entre 0 et 1, $\ln(p)$ est négatif : le signe « moins » rend la quantité d'information positive, et l'écriture avec $\frac{1}{p}$ montre bien que plus p est petit, plus la quantité d'information est grande.

Exemples. Un événement qui a une chance sur deux de se produire apporte $I\left(\frac{1}{2}\right) = \log_2 2 = 1$ bit : c'est l'information d'une seule question « oui/non ». Un événement de probabilité $\frac{1}{8}$ apporte $\log_2 8 = 3$ bits (3 questions « oui/non »), et un événement de probabilité $\frac{1}{1024}$ apporte $\log_2 1024 = 10$ bits.

Un cas réel. Dans un texte français, la lettre E représente environ 15 % des lettres, et la lettre Z environ 0,2 %. Voir apparaître un E est peu surprenant : $\log_2\left(\frac{1}{0{,}15}\right) \approx 2{,}7$ bits. Voir apparaître un Z l'est beaucoup plus : $\log_2\left(\frac{1}{0{,}002}\right) \approx 9{,}0$ bits.

Pourquoi « n bits » veut dire « n questions oui/non »

Une question « oui/non » dont les deux réponses sont **aussi probables l'une que l'autre** coupe l'ensemble des possibilités en deux moitiés égales : elle divise donc leur nombre par 2. Après n questions bien choisies, on a divisé n fois par 2, c'est-à-dire par 2^n. Pour retrouver un cas parmi $N = 2^n$ cas équiprobables (chacun de probabilité $\frac{1}{N}$), il faut donc exactement n questions, et c'est bien la quantité d'information de cet événement :

$$I\left(\frac{1}{2^n}\right) = \log_2(2^n) = n \text{ bits}$$

Exemple 1 : un nombre entre 1 et 8. Je pense à un nombre entier entre 1 et 8 ; le deviner au hasard réussit avec la probabilité $\frac{1}{8}$, soit $\log_2 8 = 3$ bits d'information. Voici les 3 questions pour le trouver quand il vaut 6 :

Question	Réponse	Nombres encore possibles
Est-il supérieur à 4 ?	oui	5, 6, 7, 8 (4 cas)
Est-il supérieur à 6 ?	non	5, 6 (2 cas)
Est-ce 6 ?	oui	6 (1 seul cas)

Chaque réponse a divisé le nombre de cas par 2 : 8 → 4 → 2 → 1. Écrire les réponses en 0 (non) et 1 (oui) donne « 1 0 1 » : **3 bits**, un par question.

Exemple 2 : une carte d'un jeu de 32 cartes. Tirer au hasard la dame de cœur a la probabilité $\frac{1}{32}$, soit $\log_2 32 = 5$ bits : 5 questions suffisent, chacune coupant le paquet en deux.

Question	Réponse	Cartes encore possibles
La carte est-elle rouge ?	oui	16
Est-ce un cœur ?	oui	8
Est-ce un valet, une dame, un roi ou un as ?	oui	4
Est-ce un valet ou une dame ?	oui	2
Est-ce la dame ?	oui	1 (la dame de cœur)

Exemple 3 : et si le nombre de cas n'est pas une puissance de 2 ? Pour un chiffre de 0 à 9 tiré au hasard (probabilité $\frac{1}{10}$), l'information vaut $\log_2 10 = \frac{\ln 10}{\ln 2} \approx 3{,}32$ bits. Trois questions ne suffisent pas ($2^3 = 8 < 10$ cas) et quatre sont toujours suffisantes ($2^4 = 16 \geq 10$) : le nombre de bits n'est pas un entier, mais il indique ce qu'il faut **en moyenne** si l'on a beaucoup de chiffres à deviner. Pour un code de carte bancaire à 4 chiffres (10^4 possibilités), $\log_2 10^4 \approx 13{,}3$ bits : il faut 14 questions dans le pire des cas.

Attention. Une question mal choisie rapporte moins : demander directement « est-ce 6 ? » dans l'exemple 1 aura presque toujours la réponse « non » (probabilité $\frac{7}{8}$), et apporte donc en moyenne moins d'un bit. Un bit est le maximum qu'une question « oui/non » puisse apporter, atteint seulement si les deux réponses sont équiprobables.

3. Rappel : l'espérance d'une variable aléatoire

Soit Y une variable aléatoire discrète qui prend les valeurs $y_1,\ y_2,\ \ldots,\ y_n$. Son **espérance** est la moyenne de ses valeurs, chacune pondérée par sa probabilité :

$$E(Y) = \sum_{i=1}^{n} y_i \, P(Y = y_i)$$

Exemple : le lancer d'un dé équilibré. Le résultat D prend les valeurs 1 à 6, chacune avec la probabilité $\frac{1}{6}$:

$$E(D) = \sum_{k=1}^{6} k \times \frac{1}{6} = \frac{1+2+3+4+5+6}{6} = \frac{21}{6} = 3{,}5$$

Cela ne veut pas dire qu'on obtient 3,5 en lançant le dé : cela signifie que, si l'on lance le dé un très grand nombre de fois, la moyenne des résultats se rapproche de 3,5 (loi des grands nombres). C'est cette interprétation qui va donner son sens à l'entropie.

4. L'entropie : l'espérance de la quantité d'information

Considérons une source qui produit un résultat X parmi n résultats possibles $x_1, \dots, x_n$ (les lettres d'un texte, les faces d'un dé, les pixels d'une image…). On note $p_i = P(X = x_i)$ la probabilité du résultat x_i.

La quantité d'information ressentie dépend du résultat obtenu : c'est donc elle-même une **variable aléatoire**, notée $I(X)$, qui prend la valeur $-\log_2(p_i)$ avec la probabilité p_i. L'**entropie de Shannon** est, par définition, l'espérance de cette variable :

$$H(X) = E\big(I(X)\big) = \sum_{i=1}^{n} p_i \times (-\log_2 p_i) = -\sum_{i=1}^{n} p_i \log_2 p_i$$

Avec la fraction vue plus haut, on peut aussi l'écrire sous la forme qui montre le mieux la « quantité d'information moyenne » :

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

> *L'entropie est la quantité d'information moyenne que l'on s'attend à ressentir en observant un résultat de la source. Elle se mesure en bits par symbole.*

Par convention, on pose $0 \times \log_2(0) = 0$: un résultat impossible ne contribue pas à l'entropie.

Exemple 1 : une pièce de monnaie équilibrée

Deux issues, chacune de probabilité $\frac{1}{2}$ et de quantité d'information 1 bit :

$$H = \frac{1}{2}\log_2 2 + \frac{1}{2}\log_2 2 = \frac{1}{2} + \frac{1}{2} = 1$$

Chaque lancer apporte exactement 1 bit d'information : c'est le maximum possible pour une source à deux issues.

Exemple 2 : une pièce truquée

La pièce tombe sur « pile » avec la probabilité $\frac{9}{10}$ et sur « face » avec la probabilité $\frac{1}{10}$:

$$H = \frac{9}{10}\log_2\left(\frac{10}{9}\right) + \frac{1}{10}\log_2(10) \approx 0{,}9 \times 0{,}152 + 0{,}1 \times 3{,}32 \approx 0{,}47$$

L'entropie vaut environ 0,47 bit : elle a baissé. Le résultat « pile » est très prévisible (peu de surprise, de quantité d'information) et le résultat « face », très surprenant, est rare : en moyenne, chaque lancer nous surprend beaucoup moins.

Exemple 3 : une pièce qui tombe toujours sur pile

$$H = 1 \times \log_2(1) = 1 \times 0 = 0$$

Aucune surprise possible : on connaît le résultat avant de lancer. Entropie nulle = aucune incertitude.

Exemple 4 : un dé équilibré

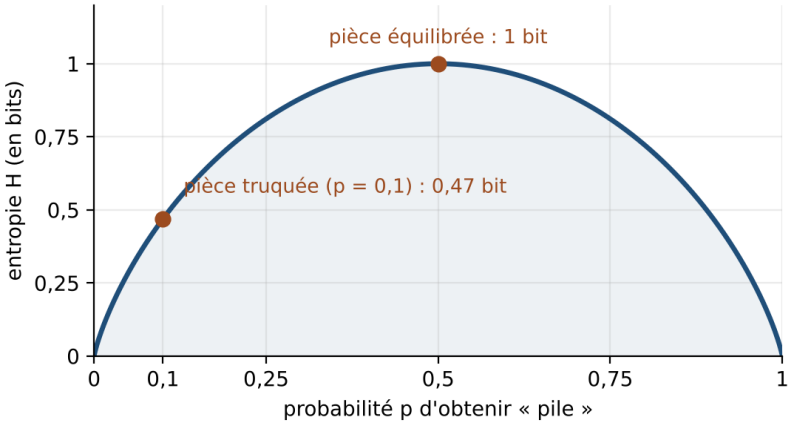
$$H = \sum_{i=1}^{6} \frac{1}{6}\log_2(6) = 6 \times \frac{1}{6}\log_2(6) = \log_2(6) \approx 2{,}58$$

Ici la quantité d'information vaut toujours $\log_2(6)$, quel que soit le résultat : sa valeur moyenne est donc $\log_2(6)$, soit environ 2,58 bits.

Toutes les pièces à la fois : la fonction d'entropie binaire

Pour une pièce qui donne « pile » avec la probabilité p, l'entropie est une fonction de p, notée h :

$$h(p) = -p\log_2 p - (1-p)\log_2(1-p)$$



L'entropie est nulle aux extrémités ($p = 0$ ou $p = 1$: résultat certain) et maximale pour $p = \frac{1}{2}$: c'est quand on ne sait pas du tout quoi prévoir que le résultat est le plus informatif.

> **Pour aller plus loin.** Montrons que le maximum est bien en $p = \frac{1}{2}$. Comme $h(p) = -\frac{p\ln p + (1-p)\ln(1-p)}{\ln 2}$, on dérive : $h'(p) = \frac{-\ln p + \ln(1-p)}{\ln 2} = \log_2\left(\frac{1-p}{p}\right)$. Cette dérivée est positive si $p < \frac{1}{2}$, nulle en $\frac{1}{2}$, négative ensuite : h croît puis décroît, avec un maximum $h(1/2) = 1$.

5. Les propriétés essentielles

L'entropie d'une source à n résultats possibles vérifie :

$$0 \leq H(X) \leq \log_2(n)$$

- $H = 0$ **exactement quand un résultat est certain (aucune incertitude**
- $H = \log_2(n)$ **exactement quand le**n résultats sont équiprobables ($p_i = \frac{1}{n}$). Dans ce cas la quantité d'information est la même pour tous les résultats : c'est une constante, donc son espérance vaut cette constante.

Cela correspond à une intuition naturelle : on est le plus incertain quand on n'a aucune raison de favoriser un résultat plutôt qu'un autre.

Situation	Entropie
Un résultat certain	$H = 0$ (minimale)
Résultats équiprobables	$H = \log_2(n)$ (maximale)
Résultats inégalement probables	entre les deux

> **Démonstration (pour les curieux).** Minorant : chaque terme $p_i \log_2 \frac{1}{p_i}$ est positif car $p_i \leq 1$.
>
> Majorant : comme $\sum p_i = 1$, on a $\log_2 n = \sum p_i \log_2 n$, d'où, en notant k le nombre de probabilités non nulles et en utilisant l'inégalité $\ln x \leq x - 1$ (valable pour tout $x > 0$, par convexité) :

$$H(X) - \log_2 n = \frac{1}{\ln 2}\sum_{p_i>0} p_i \ln\left(\frac{1}{np_i}\right) \leq \frac{1}{\ln 2}\sum_{p_i>0} p_i\left(\frac{1}{np_i} - 1\right) = \frac{1}{\ln 2}\left(\frac{k}{n} - 1\right) \leq 0$$

6. Applications concrètes

6.1 La compression de fichiers (ZIP, PNG, JPEG, MP3…)

Un fichier texte codé en ASCII utilise 8 bits pour chaque caractère, qu'il s'agisse d'un « e » très fréquent ou d'un « z » très rare. Or, on vient de voir que le « e » apporte peu d'information et le « z » beaucoup : l'idée de la compression est d'attribuer **des codes courts aux symboles fréquents et des codes longs aux symboles rares**, comme dans le code Morse où le E, très fréquent, s'écrit avec un seul point.

Si le symbole x_i reçoit un code de ℓ_i bits, la longueur du code est une variable aléatoire dont l'espérance est la **longueur moyenne** du codage :

$$L = E\big(\ell(X)\big) = \sum_{i=1}^{n} p_i \ \ell_i$$

Pour que le décodage soit possible sans ambiguïté, on impose que **aucun code ne soit le début d'un autre** (on parle de code préfixe). Le théorème du codage source de Shannon affirme alors qu'on ne peut pas faire mieux que l'entropie :

$$L \geq H(X)$$

En 1952, David Huffman (alors étudiant) a trouvé un algorithme qui construit un code préfixe de longueur moyenne minimale, et qui vérifie $H(X) \leq L < H(X) + 1$. Il est utilisé dans les formats ZIP et PNG (algorithme DEFLATE), JPEG et MP3. Comme personne ne le rencontre avant le supérieur, détaillons-le pas à pas.

L'algorithme de Huffman

1. **Compter** le nombre d'apparitions de chaque symbole : ce sont les **poids**.
2. Écrire chaque symbole comme un petit arbre réduit à une seule **feuille**, portant son poids.
3. Tant qu'il reste plus d'un arbre : prendre **les deux arbres de plus petits poids** (en cas d'égalité, on choisit au hasard) et les relier à un nouveau **nœud** dont le poids est la **somme** des deux. On étiquette **0** la branche qui mène au plus léger et **1** l'autre.
4. Quand il ne reste qu'un seul arbre, sa **racine** a pour poids le nombre total de symboles. Le code d'un symbole se lit en **descendant de la racine jusqu'à sa feuille** et en notant les étiquettes 0 et 1 rencontrées.

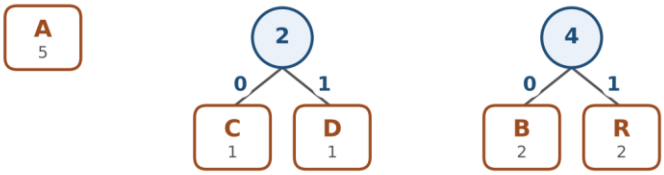
Exemple : le mot ABRACADABRA

Le mot compte 11 lettres : A apparaît 5 fois, B et R 2 fois chacune, C et D une fois chacune. Voici la construction de l'arbre :

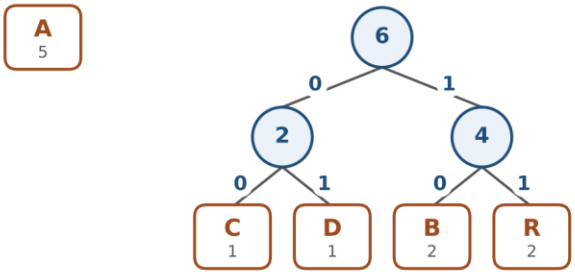
Départ : un « arbre » par lettre, de poids = nombre d'apparitions

Étape 1 : on fusionne les deux plus petits poids, C (1) et D (1) → nœud de poids 2

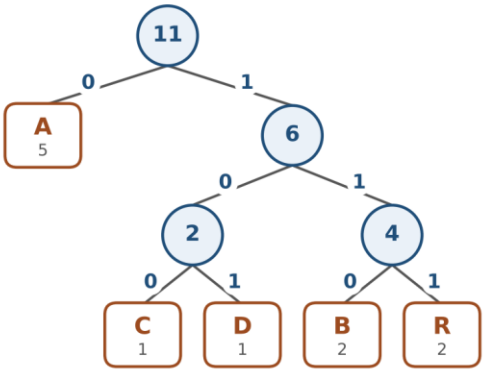
Étape 2 : parmi les poids 2, 2, 2 et 5, on fusionne B et R → nœud de poids 4



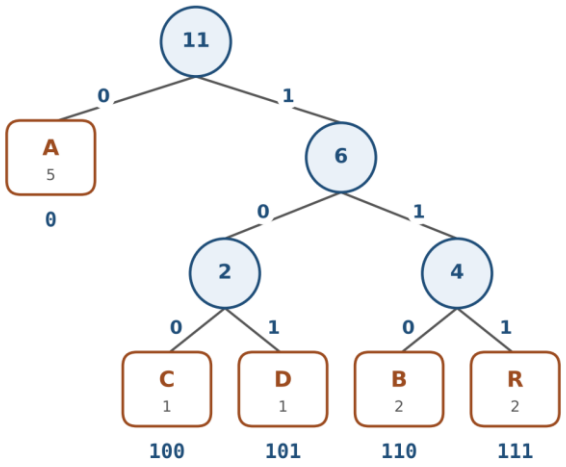
Étape 3 : on fusionne les deux plus petits, 2 et 4 → nœud de poids 6



Étape 4 : il reste 5 et 6, on fusionne → racine de poids 11 (= total des lettres)



À l'étape 2, quatre arbres sont en concurrence (poids 5, 2, 2 et 2) : on prend deux des trois arbres de poids 2, ici B et R. Un autre choix donnerait un arbre différent, mais de même longueur moyenne. Le résultat final se lit sur l'arbre complet :



Lettre	Poids	Code	Longueur ℓ_i	Contribution $p_i\,\ell_i$
A	5	0	1	$\frac{5}{11}\times 1$
B	2	110	3	$\frac{2}{11}\times 3$
R	2	111	3	$\frac{2}{11}\times 3$
C	1	100	3	$\frac{1}{11}\times 3$
D	1	101	3	$\frac{1}{11}\times 3$

Le mot s'écrit **0 110 111 0 100 0 101 0 110 111 0**, soit 23 bits, contre 88 bits en ASCII (8 bits par lettre) ou 33 bits avec un code de longueur fixe (3 bits par lettre, car il y a 5 lettres différentes). Pour **décoder** la suite de bits **01101110100010101101110**, on la lit de gauche à droite et on s'arrête dès qu'on atteint une feuille : 0 → A ; 110 → B ; 111 → R ; 0 → A ; 100 → C ; et ainsi de suite. Comme aucun code n'est le début d'un autre, il n'y a jamais d'hésitation.

Longueur moyenne et entropie. La longueur moyenne du code est :

$$L = \frac{5}{11}\times 1 + \frac{2}{11}\times 3 + \frac{2}{11}\times 3 + \frac{1}{11}\times 3 + \frac{1}{11}\times 3 = \frac{23}{11} \approx 2{,}09 \text{ bits par lettre}$$

L'entropie de la source vaut :

$$H = \frac{5}{11}\log_2\frac{11}{5} + 2\times\frac{2}{11}\log_2\frac{11}{2} + 2\times\frac{1}{11}\log_2 11 \approx 0{,}517 + 0{,}894 + 0{,}629 \approx 2{,}04 \text{ bits}$$

On vérifie bien $H \le L < H + 1$: Huffman est presque optimal. **Astuce de contrôle** : le nombre total de bits est la somme des poids des nœuds internes, ici $2 + 4 + 6 + 11 = 23$.

Quand Huffman atteint exactement l'entropie. Si toutes les probabilités sont des puissances de $\frac{1}{2}$, on obtient $L = H$. Par exemple, pour $p = \left(\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{8}\right)$, les codes 0, 10, 110, 111 donnent :

$$L = \frac{1}{2} \times 1 + \frac{1}{4} \times 2 + \frac{1}{8} \times 3 + \frac{1}{8} \times 3 = \frac{7}{4} = H$$

Les logiciels de compression combinent le codage de Huffman avec la détection des répétitions (algorithme LZ77) : c'est pourquoi un fichier peut être compressé mieux que ne le prévoit l'entropie « lettre par lettre » (voir l'activité 1). À l'inverse, un fichier déjà compressé ou chiffré a une entropie proche du maximum et ne se compresse presque plus.

6.2 Les télécommunications : débit d'un câble et correction d'erreurs

Shannon a aussi démontré que tout canal de communication perturbé par du bruit a un débit maximal. Pour un canal de bande passante B (en hertz) et de rapport signal sur bruit $\frac{S}{N}$ (rapport de puissances, **sans unité**), le débit maximal (en bits par seconde) est donné par la formule de Shannon-Hartley :

$$C = B \log_2\left(1 + \frac{S}{N}\right)$$

Le rapport signal sur bruit est souvent donné en décibels : on le convertit avec $\frac{S}{N} = 10^{\,\mathrm{SNR_{dB}}/10}$.

Exemple réel : une ligne téléphonique transmet les fréquences de 300 Hz à 3 400 Hz ($B = 3\,100$ Hz). Avec un rapport signal/bruit de 35 dB, soit $\frac{S}{N} = 10^{3,5} \approx 3\,162$:

$$C = 3\,100 \times \log_2(1 + 3\,162) \approx 36\,000 \text{ bit/s}$$

Les modems analogiques de l'époque (norme V.34) atteignaient 33,6 kbit/s : très près de la limite de Shannon, qu'aucune technologie ne peut dépasser sans augmenter la bande passante ou le rapport signal/bruit.

Corriger les erreurs. Le bruit inverse parfois des bits. Dans le modèle le plus simple (le canal binaire symétrique), chaque bit est inversé avec la probabilité p. Shannon a démontré qu'on peut transmettre avec une probabilité d'erreur aussi petite qu'on veut, à condition d'ajouter de la redondance de façon astucieuse et de ne pas dépasser le débit maximal :

$$C = 1 - h(p) \text{ bit par bit envoyé}$$

où h est la fonction d'entropie binaire vue plus haut. Pour $p = 0{,}1$: $C = 1 - 0{,}47 \approx 0{,}53$. C'est le principe des codes correcteurs utilisés dans les QR codes (qui restent lisibles même abîmés : jusqu'à 7 %, 15 %, 25 % ou 30 % de données récupérables selon le niveau choisi), les CD et DVD, ou les communications avec les sondes spatiales.

6.3 La sécurité des mots de passe

Un mot de passe de L caractères tirés **au hasard** parmi N caractères possibles a N^L valeurs possibles, équiprobables : son entropie est donc le maximum possible :

$$H = \log_2(N^L) = L \log_2(N)$$

Un mot de passe de 8 lettres minuscules ($N = 26$) a une entropie d'environ 37,6 bits, soit $2^{37,6} \approx 2 \times 10^{11}$ possibilités : une carte graphique actuelle en teste environ 3×10^{11} par seconde (pour une empreinte rapide comme NTLM) et les essaie donc toutes en moins d'une seconde. Avec 16 lettres minuscules, l'entropie monte à 75,2 bits : la longueur L agit comme un **facteur multiplicatif**, alors que la taille N de l'alphabet n'intervient que dans un logarithme.

Attention : cette formule suppose un tirage au hasard. Les humains choisissent mal : dans la fuite de données du site RockYou (2009, environ 32,6 millions de comptes), le mot de passe « 123456 » était utilisé par environ 290 700 comptes, soit 0,9 %. Sa quantité d'information n'est que de $\log_2\left(\frac{32{,}6\times10^6}{290\,700}\right) \approx 6{,}8$ bits : l'entropie réelle des mots de passe humains est bien plus faible que celle d'un tirage aléatoire.

6.4 L'intelligence artificielle : arbres de décision et modèles de langage

Un **arbre de décision** apprend à prédire une réponse (spam ou non, malade ou non…) en posant des questions successives. À chaque étape, il choisit la question qui réduit le plus l'incertitude. Si une variable partage les n individus en groupes de tailles n_k, l'entropie qui reste est l'espérance des entropies de chaque groupe, et le **gain d'information** est la baisse d'entropie obtenue :

$$G = H(Y) - \sum_k \frac{n_k}{n} H(Y \mid k)$$

Exemple réel. Sur les 2 201 personnes à bord du Titanic (passagers et équipage), 711 ont survécu. L'entropie de la survie est $H \approx 0{,}91$ bit. En séparant les femmes (73 % de survie) des hommes (21 % de survie), l'entropie moyenne tombe à 0,77 bit : le gain d'information vaut environ 0,14 bit. Séparer selon la classe (1re classe ou non) ne donne que 0,05 bit : l'algorithme pose donc en premier la question « homme ou femme ? » (voir l'activité 11). Certaines variantes d'arbres utilisent l'indice de Gini à la place de l'entropie.

Les modèles de langage (comme les assistants conversationnels) apprennent à prédire le mot suivant d'un texte. Pendant l'apprentissage, on minimise la **quantité d'information moyenne** du modèle : pour chaque mot réel, on calcule $-\ln(p)$ où p est la probabilité que le modèle lui attribuait. Cette quantité s'appelle l'entropie croisée : c'est exactement la quantité d'information de Shannon, mesurée avec le logarithme népérien.

6.5 La biologie : l'ADN et la diversité du vivant

L'ADN est un texte de 4 lettres (A, T, C, G) : avec 4 bases équiprobables, l'entropie maximale est $\log_2 4 = 2$ bits par base. Mais les génomes réels ne sont pas équilibrés : la proportion g de bases G et C (le « taux de GC ») varie beaucoup d'une espèce à l'autre. D'après la règle de Chargaff, A et T sont aussi fréquents l'un que l'autre, de même que C et G. On obtient :

$$H = -2\left(\frac{1-g}{2}\log_2\frac{1-g}{2} + \frac{g}{2}\log_2\frac{g}{2}\right) = 1 + h(g)$$

Cela donne, pour des génomes réels :

Organisme	Taux de GC	Entropie H (bits)	Gain possible
Parasite du paludisme (*Plasmodium falciparum*)	≈ 20 %	1,72	≈ 14 %
Être humain	41 %	1,98	≈ 1 %
Bactérie *Streptomyces coelicolor*	72 %	1,86	≈ 7 %

Le « gain possible » est l'économie maximale de place par rapport à 2 bits par base. Le génome humain (environ 3,1 milliards de bases) tient dans 775 Mo à 2 bits par base.

En **écologie**, l'indice de Shannon $H' = -\sum p_i \ln p_i$ (avec le logarithme népérien) mesure la diversité d'un milieu à partir des proportions p_i de chaque espèce : il est proche de 0 si une seule espèce domine, et élevé si de nombreuses espèces sont présentes en proportions voisines.

6.6 Les langues et les jeux

La redondance d'une langue. Avec 27 symboles (26 lettres et l'espace) équiprobables, l'entropie serait $\log_2 27 \approx 4{,}75$ bits par symbole. En faisant deviner la lettre suivante à des lecteurs, Shannon a estimé en 1951 que l'entropie réelle de l'anglais écrit est comprise entre 0,6 et 1,3 bit par symbole. La langue est donc très redondante (au moins 73 %) : c'est pour cela qu'on peut deviner un mot tronqué ou corriger une faute de frappe.

Le jeu des 20 questions. Chaque réponse « oui/non » apporte au plus 1 bit. Pour identifier un objet parmi N objets équiprobables, il faut $\log_2 N$ bits, donc au moins $\lceil \log_2 N \rceil$ questions. Avec 20 questions, on distingue au maximum :

$$2^{20} = 1\,048\,576 \text{ objets}$$

Wordle. La liste d'origine contient 2 315 mots-solutions : trouver le mot demande $\log_2(2\,315) \approx 11{,}2$ bits d'information. Chaque essai renvoie un motif de 5 cases de 3 couleurs possibles, donc au plus $3^5 = 243$ motifs, soit au plus $\log_2 243 \approx 7{,}9$ bits : il faut donc au moins 2 essais, et le meilleur premier mot est celui dont les motifs sont les plus « équilibrés », c'est-à-dire d'entropie maximale.

6.7 La physique : l'entropie de la thermodynamique

Le mot « entropie » vient de la thermodynamique du XIXe siècle. Pour un système dont les états microscopiques i ont pour probabilités p_i, l'entropie de Gibbs est :

$$S = -k_B \sum_i p_i \ln p_i$$

avec la constante de Boltzmann $k_B = 1{,}380649 \times 10^{-23}$ J/K. C'est la même formule que celle de Shannon, à la constante k_B près et au changement de logarithme près : l'entropie mesure l'ignorance qu'on a de l'état microscopique d'un système. Le principe de Landauer en tire une conséquence très concrète : effacer 1 bit d'information dans un ordinateur à la température T dégage au moins $k_B T \ln 2$ de chaleur, soit environ $2{,}9 \times 10^{-21}$ J à 300 K.

7. Ce qu'il faut retenir

- La **quantité d'information** d'un événement de probabilité p est $I(p) = \log_2\left(\frac{1}{p}\right)$ bits : plus il est rare, plus il est informatif.
- L'**entropie** de Shannon est l'**espérance de la quantité d'information** : $H(X) = E\big(I(X)\big) = -\sum_i p_i \log_2 p_i$.
- Elle vaut 0 quand tout est certain et $\log_2(n)$ quand les n résultats sont équiprobables.
- Elle fixe des limites exactes : à la compression ($L \geq H$), à la transmission ($C = B\log_2\left(1 + \frac{S}{N}\right)$) et à la sécurité des mots de passe ($H = L\log_2 N$).
- Le **code de Huffman** s'obtient en fusionnant à chaque étape les deux arbres de plus petits poids ; ses codes sont courts pour les symboles fréquents.

8. Pour s'entraîner

1. Un jeu rapporte 10 € avec la probabilité $\frac{1}{10}$ et 0 € sinon. Calculer l'espérance du gain, puis l'entropie du résultat (« gagné » ou « perdu »). Ces deux quantités ont-elles le même rôle ?
2. Calculer l'entropie du lancer d'un dé à 6 faces équilibré, puis celle d'une pièce équilibrée. Laquelle est la plus grande ? Pourquoi, intuitivement ?
3. Un panier contient 8 boules : 7 rouges et 1 bleue. On tire une boule. Calculer la quantité d'information de chaque couleur, puis l'entropie du tirage, comme espérance de la quantité d'information. Est-elle proche de 0 ou proche du maximum ?
4. Construire un code de Huffman pour la source de probabilités $\left(\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{8}\right)$ et vérifier que $L = H$.
5. Expliquer pourquoi un message totalement aléatoire est impossible à compresser (indice : $L \geq H$ et $H = \log_2 n$).

Réponses

1. $E = 10 \times \frac{1}{10} = 1$ € ; $H = h\left(\frac{1}{10}\right) \approx 0{,}47$ bit. L'espérance mesure la valeur moyenne du gain (en euros), l'entropie mesure l'incertitude sur le résultat (en bits) : ce n'est pas la même chose.

2. $\log_2 6 \approx 2{,}58$ bits pour le dé, 1 bit pour la pièce : le dé a plus d'issues équiprobables, donc plus d'incertitude.

3. Rouge : $\log_2 \frac{8}{7} \approx 0{,}19$ bit ; bleue : $\log_2 8 = 3$ bits. $H = \frac{7}{8} \times 0{,}19 + \frac{1}{8} \times 3 \approx 0{,}54$ bit, très loin du maximum d'1 bit pour 2 issues : le résultat est presque prévisible.

4. On fusionne les deux $\frac{1}{8}$ (nœud $\frac{1}{4}$), puis ce nœud avec $\frac{1}{4}$ (nœud $\frac{1}{2}$), puis avec $\frac{1}{2}$. Codes : 0, 10, 110, 111. $L = H = \frac{7}{4} = 1{,}75$ bit.

5. Pour un message aléatoire, les n symboles sont équiprobables : $H = \log_2 n$ est déjà le nombre de bits d'un code de longueur fixe. Comme $L \geq H$, aucun code ne peut être plus court en moyenne.

Sources

C. E. Shannon, « A Mathematical Theory of Communication », *Bell System Technical Journal*, 1948. — C. E. Shannon, « Prediction and Entropy of Printed English », *Bell System Technical Journal*, 1951. — D. A. Huffman, « A Method for the Construction of Minimum-Redundancy Codes », *Proceedings of the IRE*, 1952. — T. M. Cover et J. A. Thomas, *Elements of Information Theory*, Wiley. — Fréquences des lettres du français : sttmedia.com. — Données du Titanic : jeu de données de R. Dawson (1995). — Fuite RockYou (2009) : décomptes publiés par la presse spécialisée (SC Magazine, 2010). — Taux de GC des génomes : Wikipédia, « GC-content ». — Wordle : liste d'origine de 2 315 solutions.