Avant-propos

Les forts développements des technologies de l'information et de la communication (TIC) ont naturellement amené les universités et les écoles d'ingénieurs à faire évoluer leur formation type EEA (électronique, électrotechnique et automatique, *Electrical Engineering*). C'est particulièrement le cas dans le secteur des communications sans fil ! En effet, la transmission de données, type parole et vidéo, trouve des applications de plus en plus nombreuses et variées. Il est devenu nécessaire aux futurs diplômés de comprendre et maîtriser les problèmes liés à la mise en place d'une liaison radio en fonction de l'environnement, de la mise en forme et du débit des données source, de la puissance disponible à l'antenne, de la sélectivité et de la sensibilité du récepteur.

Cet ouvrage ne demande qu'un niveau de propédeutique, assimilé, en mathématiques. Il ne cherche pas à jouer le rôle d'une somme, mais plutôt à se convaincre de la richesse du domaine, de son avenir, les bons fascicules de base faisant, par ailleurs, florès. Les « applications notes » des industriels, concises, semblent aussi incontournables pour tout chercheur-ingénieur.

L'innovation technologique a un rôle très important dans le domaine des TIC. Il apparaît alors nécessaire aujourd'hui que les formations apportent des propositions adaptées et innovantes en matière de pédagogie et d'outils associés, tout en maîtrisant au mieux le caractère fondamental des enseignements, seul garant d'une formation solide et pérenne.

Cet ouvrage s'adresse aux étudiants de BTS/IUT, licences professionnelles, ingénieurs et masters, mais peut-être aussi aux chercheurs des domaines connexes, tels que ceux du *hardware*, avec, par exemple, les boucles à verrouillage de phase et leurs éléments centraux : les oscillateurs commandés en tension et le fameux bruit de phase associé. Il y a sûrement tout un domaine lié à ce que l'on appelle le *firmware*, à instruire, mais aussi, des outils mathématiques déjà utilisés, par exemple, pour la

relativité ou la cryptographie, voire d'anciens codages à revisiter, tels certains de Claude Shannon lui-même.

Remerciements (non exhaustifs)

Chafia Yahiaoui de l'École supérieure d'informatique d'Alger, et mes collègues « Télécom » de l'INSA Lyon, entre autres : Guillaume Villemaud, Jean-Marie Gorce, Hugues Benoit Cattin, Attila Baskurt, Stéphane Frenot, Thomas Grenier, Jacques Verdier, Gérard Couturier, Patrice Kadionic, Alexandre Boyer, Carlos Belaustegui Goitia, pour leurs observations scrupuleuses, ainsi que leurs commentaires pertinents ; une amicale pensée à Omar Gaouar, mon très sympathique « coturne » de l'INSA FES, plutôt homme de réseaux, mais cependant aussi féru de modulations musicales.

Ce travail est supporté par l'UpM (Union pour la Méditerranée). Il a été réalisé au Centre d'intégration en télécommunication et intelligence artificielle (CITIA), INSA/UEMF/FES.

Introduction

Le mot communication est aujourd'hui un mot fourre-tout dans nos sociétés ; *ab initio*, cela permet de partager de l'information. Un département de telle université ou telle école d'ingénieurs pouvait s'appeler historiquement « humanités » (à la fin des années 1960, orienté en particulier gestion des ressources humaines, voire sociologie) ; puis il tendit à se ramener à : « communication et humanités », les deux termes ayant été permutés entretemps. Ne va-t-il rester que le seul terme dénaturé de communication ?

Il ne faudrait pas amalgamer ce mot à d'autres termes comme information, codage, voire l'utiliser dans une acception sémantique, polémique ; l'expression « propager/partager de l'information » intervient dans cet ouvrage, *stricto sensu*, dans son sens technique, loin de toute interprétation moderniste.

I.1. Pourquoi le numérique ?

Pour les communications haut débit, les transmissions sont limitées par des contraintes physiques telles que le bruit ou les parasites, dus aux imperfections des systèmes et la physique des composants modifiant la transmission du signal émis. La déformation du signal au cours de la propagation est, de même, un souci. D'où la nécessité d'une claire séparation temporelle des signaux émis, pour, en pratique, qu'ils restent effectivement distincts à la réception.

La transmission d'un train de symboles subit des dispersions des données, dans le temps, d'où des interférences inter-symboles. Les signaux réfléchis par les bâtiments, le sol ou les véhicules induisent celles-ci, selon la longueur des chemins parcourus. L'importance de ce phénomène est fonction de la fréquence (surtout en haute fréquence), pouvant varier de façon stochastique, *via*, par exemple, les phases du signal dans le temps (après réflexion sur des obstacles : échos). Elles engendrent souvent

des signaux s'ajoutant de façon soit destructive, en réception. Le signal résultant sera alors très faible, parfois quasi nul. Ces signaux peuvent aussi s'ajouter de manière constructive ; le signal final sera alors plus puissant que celui du trajet direct. Remarquons que les multitrajets n'ont pas que des inconvénients puisqu'ils permettent que la communication soit possible, même lorsque l'émetteur et le récepteur ne sont pas en vis-à-vis (cas des communications transcontinentales).

Un signal est souvent corrompu lorsqu'il parcourt différents trajets entre l'émetteur et le récepteur : les bits de données qui arrivent au récepteur subissent des retards. Ce signal, déformé, sera mal interprété par le récepteur.

Dans les communications à haut débit, ces derniers sont limités par des contraintes : les erreurs de transmission sont atténuées si l'on numérise le signal. Par exemple, pour la voix, l'amplitude du signal est mesurée typiquement 8 000 fois par seconde et sa valeur est codée par une séquence de 8 bits (des 0 ou des 1) – des échantillons (*samples*). Le récepteur décode la séquence du signal d'origine, reconstruisant ainsi le signal émis. Utiliser uniquement des 0 ou des 1 induit une faible probabilité d'erreur (mais non nulle). Le canal de propagation peut être modélisé *via* une réponse impulsionnelle (voir système linéaire, peigne (*comb*) de Dirac) ; le signal reçu r (t) n'est alors autre chose que le filtrage du signal émis x (T) par le canal de propagation c (t) et peut donc être écrit en bande de base, *via* une convolution à laquelle s'ajoute souvent un bruit additif (voir terme de Langevin) modélisant les imperfections du système. Il est fait référence aux canaux sélectifs en fréquence lorsque le signal transmis x (t) occupe une bande de fréquences [-W/2, W/2], cette dernière étant plus large que la bande de cohérence du canal de propagation, c (t), canal de propagation – définie comme l'inverse du temps de retard (*delay spread*) – maximum du canal de propagation Tr.

Dans ce cas, les composantes fréquentielles de x(t) séparées de la bande de cohérence subissent des atténuations différentes. Dans les systèmes de transmission numériques haut débit, les symboles sont souvent transmis à intervalle de temps régulier T, à temps de retard maximum des trajets Tr ; le signal reçu à un instant t peut s'exprimer comme une somme pondérée (affectée des atténuations des trajets) du signal transmis simultanément (le temps de propagation des ondes électromagnétiques est souvent négligé, ces dernières se propageant à la vitesse de la lumière) et les signaux émis aux instants précédents, un multiple de la période d'échantillonnage (*sampling*).

I.2. Représentation temporelle d'un canal

Les coefficients du canal de propagation sont donnés par les valeurs prises pour les divers moments multiples de T : [|c(0)|, |c(T)|, |c(2T)|,|c(3T)|, |c4T)|, |c(5T)|]. Si

l'on s'intéresse à la radio mobile intra-bâtiments, à 5 GHz, T est de l'ordre de 50 ns ; Tr vaut 450 ns.

Les concepteurs doivent réduire les interférences causées par les réflexions multiples du signal et extraire le signal. L'égalisation consiste à équilibrer les effets des distorsions dues à ces trajets multiples. Pour cela, il est nécessaire d'identifier les coefficients d'atténuation qui modélisent l'effet du canal de propagation c (t).

Les techniques actuelles, utilisées dans les applications industrielles, font appel à des séquences d'apprentissage ; on envoie régulièrement une « séquence choisie », connue de l'expéditeur et du destinataire. Cette méthode permet de connaître les différents déphasages et retards du canal, donnant de bons résultats dans la pratique. En revanche, si la période d'échantillonnage est trop petite comparée au délai Tr (comme c'est le cas avec des transferts à flux élevé ; le nombre de coefficients c (iT), (typiquement : $0 \leq i \leq 5$) à déterminer peut être très grand (voir l'inversion de matrices). Ainsi, la transmission de forts débits en présence de plusieurs chemins peut rapidement augmenter la complexité et donc le coût des terminaux.

La sélectivité en fréquence d'un canal se comprend de la manière suivante : le signal à transmettre a des composantes de fréquences atténuées différemment par le canal de propagation. Ce phénomène se produit lorsque le signal a une bande de fréquence plus large que la bande cohérente du canal de propagation. La bande de cohérence d'un canal est définie comme la bande passante minimale pour laquelle les pertes de deux canaux sont indépendantes. Ce phénomène est l'un des principaux obstacles à la fiabilité des transmissions : il faut en effet estimer le canal (ce qui provoque une perte du débit dans les environnements mobiles) et égaliser (ce qui accroît la complexité des récepteurs).

La complexité d'un égaliseur numérique dépend du nombre de chemins du canal de propagation (déterminé par la relation entre la durée d'égalisation Tr et la période d'échantillonnage T), mais également du type de constellations (voir le diagramme de Fresnel) émises : les bits sont réellement transmis sous forme de symboles plutôt que tels quels. Le nombre de bits contenus dans chaque symbole indique la taille de la constellation. Plus cette taille est grande, plus le débit est élevé. L'amplitude moyenne de ces constellations est généralement fixée à un seuil en raison des limites de puissance aux terminaux.

Pourquoi ne peut-on pas augmenter indéfiniment les débits en augmentant la taille de la constellation ? On peut augmenter le débit de transmission en augmentant la constellation. Mais, si l'on parle de débit en tant que nombre de bits par seconde arrivant sans erreur au récepteur, alors ceci n'est pas possible ; plus la taille de la constellation est grande (à puissance fixée, cette dernière est toujours normalisée pour

des questions de coût de transmission), plus les symboles émis ont des valeurs proches. Il n'est alors pas aisé au récepteur de trancher entre deux valeurs entachées d'erreurs dues au bruit. Nous pouvons réellement augmenter le débit (en tant que vitesse de transmission) en augmentant la constellation. Le débit a donc un seuil appelé capacité du canal. Une transmission exempte d'erreur n'était guère admise par des scientifiques avant la fin des années 1950. À cette époque, il était naturel de réduire la probabilité d'erreur de transmission en réduisant le débit binaire, définissant ainsi la capacité du canal. Ce n'est qu'avec les travaux de Claude Shannon au début des années 1920, que l'encodage a émergé pour sortir de ce dilemme.

I.3. Nécessité du codage

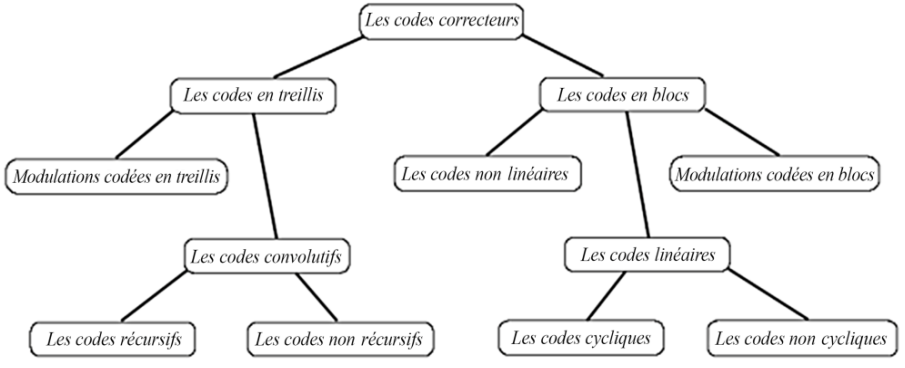

Figure I.1. *Différents types de codes*

Pour que le destinataire comprenne le message transmis, il faut que ce dernier soit le plus proche possible du message de départ. Quel que soit le principe de transmission, des perturbations viennent s'ajouter à l'information et la déforment. Il est donc nécessaire d'éliminer ces parasites, et c'est là l'intérêt premier du codage.

I.4. Bases synoptiques sur la théorie de l'information

La figure I.2 rappelle les principes même de la théorie de l'information selon Shannon.

La source est l'élément qui intéresse le destinataire et le canal est le siège du phénomène de propagation, mais aussi des perturbations.

On considèrera un canal discret sans mémoire. Le mot « discret » fait référence au fait que l'on a déjà converti le signal réel, si analogique, en signal numérique binaire qui n'est donc plus continu.

« Sans mémoire » signifie que le bruit est modélisé *via* une probabilité conditionnelle de B sachant que A est une probabilité indépendante du temps.

On approximera pour la partie théorique ce canal par un canal gaussien blanc, c'est-à-dire que tous les bits ont la même probabilité d'émission, quelle que soit leur position.

L'entropie H définit la quantité d'information apportée par la source et elle dépend de la probabilité d'apparition du 0 ou du 1. Si un seul message est possible, l'entropie est nulle.

L'entropie permet de mesurer la quantité d'information perdue après transmission bruitée ou chiffrement.

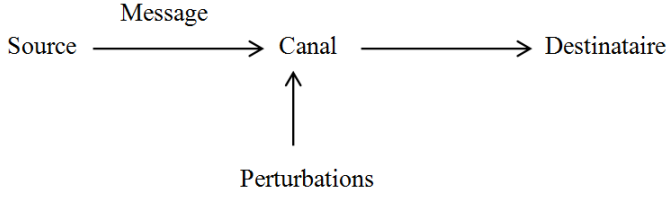

Figure I.2. *Schéma de Shannon*

I.4.1. *Théorème de Shannon-Hartley*

Il y a une quantité d'informations théoriques maximale pouvant être transmise par le canal.

Pour tout canal, il existe un algorithme de codage tel que le message envoyé par la source soit reçu avec un taux d'erreur arbitrairement faible.

I.4.1.1. *Un peu de mathématiques*

Un message X est un ensemble d'éléments élémentaires x_i caractérisés par leur probabilité d'occurrence.

La quantité d'information qu'il transmet est une mesure de son imprévisibilité : plus un message est prévisible, moins il apporte de renseignements.

Soit x un message élémentaire et p(x) sa probabilité d'émission.

La quantité d'information h(x) qu'il transmet est définie par h(x) = – log (p(x)).

On remarque que si le message a une probabilité de 1, l'information transmise h est nulle.

On définit l'entropie de la source par la quantité d'information moyenne de la source, ce qui se traduit mathématiquement par l'espérance de la quantité d'information intrinsèque de chaque message élémentaire. H(X) ne dépend donc que de la probabilité d'émission du 0 ou du 1 :

$$H(X) = E(h(x_i)) = -\sum_{i=1}^{k} p(x_i) * \log_2(p(x_i))$$

I.4.1.2. *Unité de H : le shannon*

On voit que l'entropie est maximale pour une probabilité d'émission uniforme, c'est-à-dire pour $p(x_i = 0) = p(x_i = 1) = 1/2$, ce qui est le cas dans un canal binaire symétrique.

On définit aussi H(X/Y), appelée ambiguïté ou entropie conditionnelle, qui est directement liée à la probabilité d'erreur de transmission du canal :

$$H(X/Y) = E(hx_i/y = -\sum_{j=1}^{k} p(y_j) * H(X/Y = y_j$$

avec : $H(X/Y = y_j) = -\sum_{i=1}^{k} p(x_i y_j) * \log_2(p(x_i y_j))$.

En effet, une variable binaire X peut prendre seulement 2 valeurs : 0 ou 1 (voir figure I.3).

Dans la modélisation du canal binaire symétrique, quelle que soit la valeur de départ, il y a une probabilité d'erreur $p = p_e$ pour que le bit soit changé en son opposé, et il y a alors une probabilité $p = 1 - p_e$ pour que le bit soit transmis.

On a alors pour valeurs de l'entropie :

$$\begin{cases} H(X/Y) = -p\log_2(p) - (1-p)\log_2(1-p) \text{ si } 0<p<1 \\ H(X/Y) = 0 \text{ si } p = 0 \text{ ou } 1 \end{cases}$$

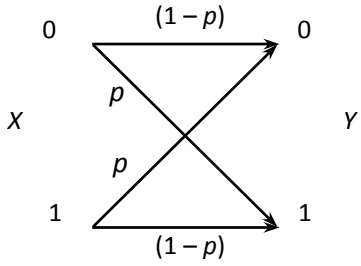

Figure I.3. *Probabilité d'erreur (canal binaire symétrique)*

Le débit d'information Ds correspond au produit de l'entropie par le nombre moyen de symboles transmis par seconde.

Si chaque symbole a une durée moyenne de τ_m, alors $Ds = H(x) / \tau_m$ en Shannons/s.

L'information mutuelle I(X;Y) mesure la quantité d'information apportée sur x, c'est-à-dire l'information correcte transmise par le canal. On a :

$$I(X;Y) = H(Y)-H(Y/X) = H(X)-H(X/Y)$$

Alors que H(X) représente la quantité d'information initiale, H(X/Y) est en quelque sorte la quantité d'information perdue lors de la transmission.

Si I(X;Y) = 0, H(Y) = H(Y/X), c'est-à-dire que si la probabilité de réception de y est indépendante de la probabilité d'émission du message reçu, le canal est mauvais.

En revanche, le canal est considéré comme parfait si I(X;Y) est maximal, c'est-à-dire si le terme négatif -H(X|Y) = 0 soit I(X|Y) = H(X) :

$$<=> H(X/Y) = 0$$

$$<=> -\log\{p(X/Y)\} = 0$$

$$<=> P(X/Y) = 1$$

Ce qui signifie que si l'on reçoit un message Y, on est sûr à 100 % du message d'origine X.

La capacité du canal de transmission CC est définie par le maximum de l'information mutuelle : $CC = \max\{I(X;Y)\}$.

Le but de tout système de transmission est de se rapprocher de cette valeur, en sachant que la présence imprévisible de bruits dégrade le message émis.

I.4.1.3. *Application au canal binaire symétrique*

On cherche à calculer :

$$Cc = \max \{I(X;Y)\} = \max \{ H(Y)-H(Y/X)\}$$

Pour le calcul de H(Y), on cherche à calculer la probabilité p(y = 0) d'obtention de y = 0.

Il y a une probabilité p(x = 0) que x = 0 soit émis et une probabilité p(y = 0/ x = 0) = $(1-p_{e)}$ qu'il soit correctement retransmis. Mais il y a aussi une probabilité p(y = 0/x = 1) = p_e que le bit 0 soit reçu alors que c'est x = 1 qui a été envoyé, avec une probabilité p(x = 1).

Il en résulte donc :

$$p(y = 0) = p(x = 0) * p(y = 0/x = 0) + p(x = 1) * p(y = 0/x = 1) = 1/2 * (1-p_e) + 1/2 * p_e$$

car dans le cas d'un canal binaire symétrique, on sait que : p(x = 0) = p(x = 1) = 1/2 = ½ = p(y = 1).

D'où :

$$H(Y) = -1/2*\log_2(1/2)-1/2*\log_2(1/2) = - 2 * 1/2 * \log_2(1/2) = \log_2(2) = 1$$

Pour le calcul de H(Y/X), on a :

$$H(Y/X) = p(x = 0) * H(Y/x = 0) + p(x = 1)*H(Y/x = 1) = 1/2 * [H(Y/x = 0) + H(Y/x = 1)] \text{ (car } p(x = 0) = p(x = 1) = 1/2)$$

avec :

$$H(Y/x = 0) = - p(y = 0/x = 0) * \log_2 (p(y = 0/x = 0)) - p(y = 1/x = 0) * \log_2 (p(y = 1/x = 0))$$

$$= - (1-p_e) * \log_2 (1-p_{e)} - p_e * \log_2 (p_e)$$

et :

$$H(Y/x = 1) = - p(y = 0/x = 1) * \log_2 (p(y = 0/x = 1)) - p(y = 1/x = 1) * \log_2 (p(y = 1/x = 1))$$

$$= - p_e * \log_2 (p_{e)} - (1-p_{e)} * \log_2 (1-p_e)$$

Nous avons donc obtenu :

$$H(Y/x = 0) = H(Y/x = 1)$$

d'où :

$$H(Y/X) = 1/2 * [H(Y/x = 0) + H(Y/x = 1)]$$

$$= - p_e * \log_2 (p_e) - (1-p_e) * \log_2 (1-p_e)$$

Pour le calcul de I(X;Y), en réunissant les différents éléments précédemment calculés :

$$I(X;Y) = H(Y)-H(Y/X) = 1 + p_e * \log_2 (p_e) + (1-p_e) * \log_2 (1-p_e)$$

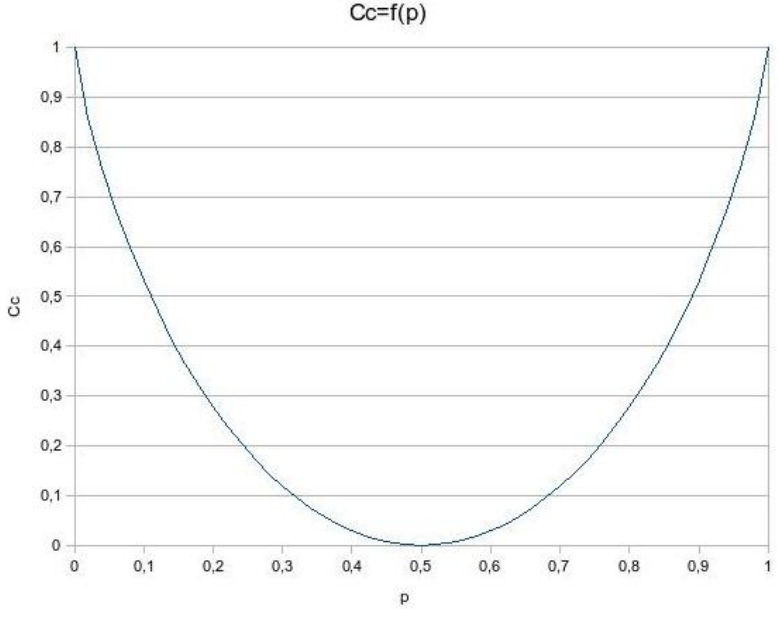

Figure I.4. *Capacité du canal de transmission en fonction de la probabilité d'erreur qu'il induit*

On remarque que la capacité du canal est maximale si la $p_e = 0$ ou 1. Ce résultat paraît logique puisque si le canal n'est pas bruité ($p_e = 0$), on est sûr d'avoir en sortie le message original. Si, en revanche, on a $p_e = 1$, on sait que l'on doit prendre le message inverse de celui de sortie pour retrouver l'information de départ.

D'après le théorème de Shannon, pour toute source d'entropie $H < C$, il existe un code de longueur N tel que la probabilité d'erreur par mot soit majorée par une quantité arbitrairement faible :

$$p_e < 2^{-N^* E(Rb)}$$

avec : $E(R_b)$ fonction de Gallagher ou de décodage aléatoire.

I.5. Codes en bloc linéaires

Pour envoyer une information, on la numérise, c'est-à-dire qu'on la transcrit sous forme d'une séquence de bits. Pour éviter la détérioration du signal par les parasites présents le long du trajet de la source au destinataire, il est nécessaire de le coder. La méthode la plus fréquente consiste à introduire une redondance dans le message d'entrée afin d'être certain de recevoir tous ses éléments en sortie.

Si le canal est très parasité, on triple les bits : si l'on a abc en entrée, on transmet aaabbbccc. Le décodeur sait alors reconnaître les bits du signal initial et les erreurs. Par exemple, s'il reçoit aiabbbocc, il choisit à chaque fois le caractère le plus présent parmi 3 bits consécutifs, donc il lit bien abc. Avec ce type de code, on suppose au départ que le maximum d'erreur est d'une par séquence de 3 bits.

Si le canal est peu parasité, on n'a pas besoin d'introduire une si grande redondance, qui allonge le message et donc le temps de transmission. On choisit alors un codage par bit de parité : le message est découpé en blocs de k bits auxquels on ajoute un bit « de parité » tel que le nombre de 1 qui sont envoyés soit pair. Ainsi, si le décodeur reçoit un nombre impair de 1, il détecte l'erreur. Comme il ne sait pas la corriger, il renvoie le message en entrée, et le cycle recommence jusqu'à ce qu'il n'y ait plus d'erreurs.

I.5.1. *Codes en bloc*

Ici aussi, le message est séquencé en blocs de k bits, traités séparément par le codeur.

On peut alors avoir 2^k messages différents à envoyer. On crée un code de 2^k collections ordonnées de n bits ($n > k$) dont les éléments sont appelés « mots » : on fait correspondre à chaque message potentiel un mot unique de n bits (voir figure I.5).

Pour chaque message reçu par le codeur, s'il ne correspond pas exactement à un mot du code, on mesure sa distance[1] par rapport à chacun des mots et on en déduit le message initial pour la distance la plus petite, inférieure ou égale au nombre e d'erreurs : e = E((d-1)/2) (avec E : fonction partie entière). En effet, si un message se trouve à une distance (d/2) de deux mots, on ne saura pas à quel mot il correspond.

Un code est alors défini par trois paramètres : [n,k,d] avec :

– n : taille des mots ;

– k : dimension du code ;

– d : distance de Hamming[1] minimale entre deux mots.

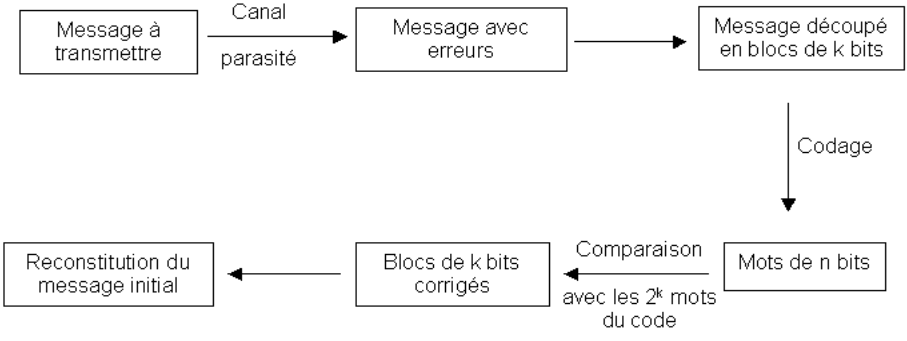

Figure I.5. *Codes en bloc*

On mesure la fiabilité d'un code à l'aide du rapport d/n et le taux du code par $R = k/n$. Plus R est petit, plus la redondance est grande, et donc plus le temps de fonctionnement est long. Il faut alors trouver un bon équilibre entre ces 2 valeurs, pour corriger un maximum d'erreurs en un temps minimal.

Prenons un exemple : on veut envoyer un message sur k = 2 bits, il peut alors prendre $2^2 = 4$ valeurs différentes. On choisit de coder les mots sur 5 bits, et la distance de Hamming sera alors de 3. On a donc un code [5,2,3].

1. Distance de Hamming (d) : nombre de bits par lesquels diffèrent deux mots du code.

Message	Mot correspondant
(0,0)	(0,0,1,1,0)
(0,1)	(0,1,0,1,1)
(1,0)	(1,0,1,0,1)
(1,1)	(1,1,0,0,0)

Tableau I.1. *Codage pour correction d'erreurs*

Si l'on reçoit (1,0,1,1,0), on trouve d = 1 minimale par rapport au mot qui correspond au message (0,0), donc on rectifie l'erreur et on lit (0,0).

Comment est-ce que cela fonctionne concrètement ? Il faut utiliser l'outil mathématique.

Soit G une matrice n*k. Tout mot c du code peut être obtenu à partir de tout message x de l'ensemble des messages par c = Gx, avec dim(c) = n et dim(x) = k, n≥k. G est appelée matrice de codage. Prenons H une matrice (n-k)*n pour laquelle Hc = 0, c'est-à-dire H(Gx) = GHx = 0, donc GH = 0. On peut dire que H est « orthogonale » à G, leur produit, scalaire, est nul. Le mot c envoyé appartient alors au code si et seulement si le produit Hc est nul, sinon il y a une erreur dans la réception du message. H est la matrice de contrôle du code. Il suffit donc de créer une matrice G, puis H orthogonale à celle-ci pour vérifier si le message reçu est erroné ou non. S'il l'est, on peut alors détecter et corriger cette erreur par un simple calcul matriciel.

I.5.2. *Exemple*

Soit un mot que l'on code en blocs de k = 3 bits, il y aura alors $2^3 = 8$ possibilités de messages différents (de x_1 à x_8). On fait correspondre à chacun de ces messages un mot (de c_1 à c_8). Pour cela, on crée la matrice de codage G, 6*3 avec laquelle on peut calculer chaque mot : $c_n = Gx_n$.

Les messages x_n envoyés sont :

$$\begin{matrix}1&0&0\\0&1&0\\0&0&1\\1&1&0\\0&1&1\\1&0&1\end{matrix}$$

$$x_1 = \begin{matrix}0\\0\\0\end{matrix} \quad x_2 = \begin{matrix}1\\0\\0\end{matrix} \quad x_3 = \begin{matrix}1\\1\\0\end{matrix} \quad x_4 = \begin{matrix}1\\1\\1\end{matrix} \quad x_5 = \begin{matrix}1\\0\\1\end{matrix}$$

$$x_6 = \begin{matrix}0\\0\\1\end{matrix} \quad x_7 = \begin{matrix}0\\1\\1\end{matrix} \quad x_8 = \begin{matrix}0\\1\\0\end{matrix}$$

On obtient alors les mots du code par $c_n = Gx_n$:

$$c_1 = \begin{matrix}0\\0\\0\\0\\0\\0\end{matrix} \quad c_2 = \begin{matrix}1\\0\\0\\1\\0\\1\end{matrix} \quad c_3 = \begin{matrix}1\\1\\0\\0\\1\\1\end{matrix} \quad c_4 = \begin{matrix}1\\1\\1\\0\\0\\0\end{matrix}$$

$$c_5 = \begin{matrix}1\\0\\1\\1\\1\\0\end{matrix} \quad c_6 = \begin{matrix}0\\0\\1\\0\\1\\1\end{matrix} \quad c_7 = \begin{matrix}0\\1\\1\\1\\0\\1\end{matrix} \quad c_8 = \begin{matrix}0\\1\\0\\1\\1\\0\end{matrix}$$

On remarque que les 3 premiers bits de chaque mot sont identiques aux 3 bits du message envoyé, puis les 4 suivants servent simplement à coder l'information ; ce sont les symboles de parité. Ce type de code est dit « systématique ».

On cherche ensuite une matrice de contrôle H telle que $HG = 0$ et dont tous les vecteurs colonnes sont distincts. On trouve :

$$H = \begin{matrix}0&1&1&1&0&1\\1&1&0&1&0&0\\1&0&1&1&1&0\end{matrix}$$

Si on envoie l'objet 100001, on calcule le syndrome $s = Hc = 111$: c'est la quatrième colonne de H, cela signifie que le quatrième bit du mot envoyé est erroné. Comme on est en binaire, il suffit de remplacer le 1 par un 0, et on retrouve bien le mot correspondant (c_2 ici).

REMARQUE. Il existe encore des configurations d'erreurs indétectables.

I.6. Techniques de codage

I.6.1. *Entrelacement*

L'entrelacement est une technique de codage qui consiste à permuter une séquence de bits pour éloigner le plus possible les erreurs les unes des autres : on répartit les erreurs sur toute une séquence ; le pourcentage d'erreurs à chaque endroit est alors peu élevé et on peut donc les corriger. On remettra ensuite les bits dans l'ordre pour retrouver le message initial.

Concrètement, l'entrelacement est utilisé par exemple sur les CD : s'il y a une rayure, les erreurs sont concentrées au même endroit, on les répartit sur une longue séquence pour pouvoir les détecter et ensuite les corriger. C'est le codage en bloc de Reed-Salomon.

Aujourd'hui, il n'existe aucune règle d'entrelacement, il faut tester différents entrelaceurs pour choisir celui qui a le meilleur résultat.

Pour les turbocodes, l'entrelaceur fait partie intégrante de la conception du code (on choisit l'entrelaceur en fonction du code).

Mais cette technique pose un problème : un entrelaceur est souvent conçu pour une longueur de code précise, il ne fonctionne plus si le paquet d'erreurs s'étend sur une longueur supérieure. En turbocodes, on utilise majoritairement les entrelaceurs Golden car ils ont de bonnes propriétés d'étalement.

REMARQUE. On utilise aussi l'entrelacement pour les codes convolutifs.

I.6.2. *Codes convolutifs*

I.6.2.1. *Généralités*

Le principe des codes convolutifs a été inventé en 1955 par Peter Elias, professeur au MIT. Contrairement aux codes en blocs, qui découpent le message en blocs finis, on considèrera ici une séquence semi-infinie d'informations qui passe à travers plusieurs registres à décalage. Le nombre de ces registres est appelé mémoire du code. Comme exemple, on considère le code convolutif présenté en figure I.6.

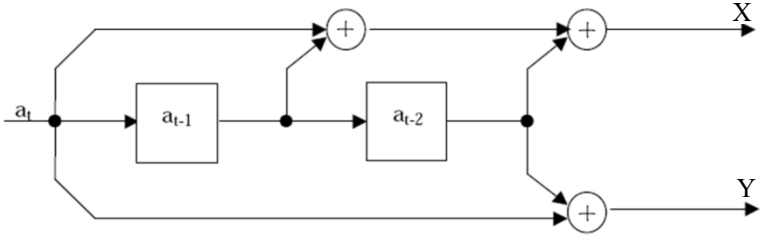

Figure I.6. *Codeur convolutif*

Il possède une mémoire égale à 2. À l'instant t, on considère donc les bits a_t, a_{t-1}, a_{t-2}. On aura en sortie :

$$X_1 = a_t \oplus a_{t-1} \oplus a_{t-2} \qquad X_2 = a_t \oplus a_{t-2}$$

On représente le code par un diagramme de transition (voir figure I.7). Ce schéma décrit, pour chaque combinaison possible des registres à décalage, le message en sortie du codeur en fonction du bit d'entrée. Chaque case du schéma correspond à un état des registres à décalage. Les chiffres à côté des flèches indiquent les bits d'entrée et le bit codé correspondant à la transition. Par exemple, si le codeur, initialisé à 00, reçoit la séquence 101, le message codé sortant sera 11 10 00 (voir figure I.8).

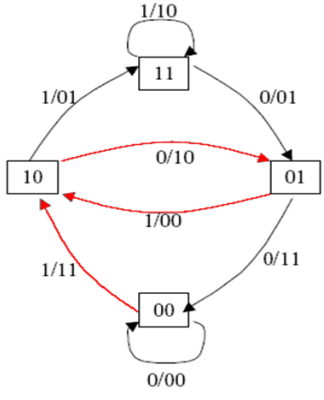

Figure I.7. *Diagramme de transitions*

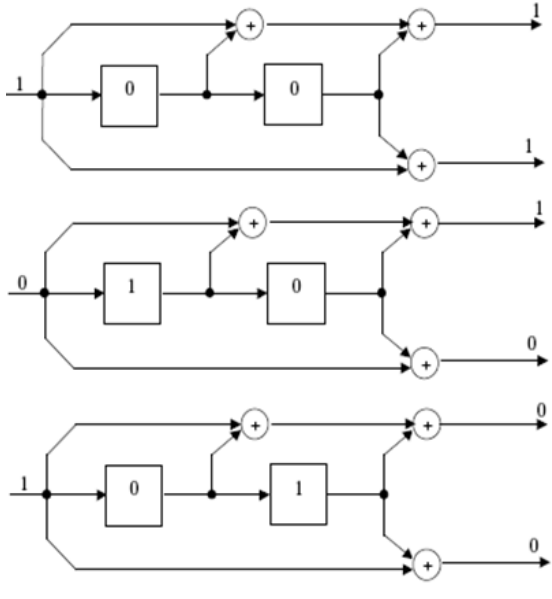

Figure I.8. *Réponse au message 101*

I.6.2.2. *Codes RSC et NSC*

Deux catégories de codes convolutifs sont particulièrement intéressantes à étudier : les codes systématiques récursifs (*Recursive Systematic Convolutional codes*) et les codes non systématiques (*Non Systematic Convolutional codes*).

Un code convolutif est dit systématique si on retrouve à sa sortie le bit d'entrée. De plus, on appelle récursif un code dont les registres à décalage sont « alimentés » par le contenu de ceux-ci.

EXEMPLE.– *Un code RSC est représenté figure I.9 : il est systématique, puisque sa sortie X est identique à l'entrée. Il est aussi récursif, puisque l'on trouve en entrée des registres à décalage des informations qui se trouvent dans ces mêmes registres (l'information circule aussi de la droite vers la gauche).*

On a constaté expérimentalement que seuls les codes RSC sont susceptibles d'atteindre la limite de Shannon.

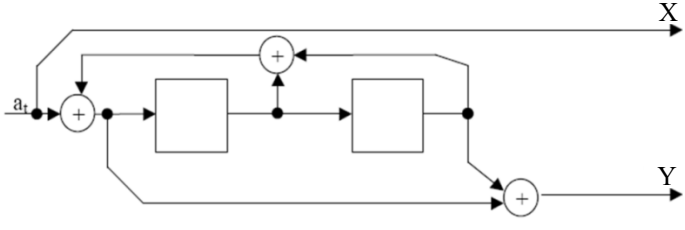

Figure I.9. *Codeur RSC*

Les codes NSC ont l'avantage par rapport aux codes systématiques de fournir plus d'information : tout bit de sortie du codeur renseigne sur plusieurs bits du message source. Le décodeur dispose donc de plus d'éléments et permet de corriger plus d'erreurs. C'est pour cette raison que les codes NSC ont été majoritairement utilisés jusqu'au début des années 1990.

I.6.2.3. *Exemple de décodage : l'algorithme de Viterbi*

L'algorithme le plus utilisé dans le décodage de la séquence a été inventé par Andrew Viterbi, en 1967. Il permet de trouver la séquence d'états la plus probable ayant produit la séquence mesurée. Bien que le message constitue *a priori* une séquence semi-infinie, il est en pratique découpé en blocs de très grande taille (de 100 à 100 000 bits, voire plus) et le codeur est initialisé à 0 entre chaque bloc. Le principe de l'algorithme est d'examiner tous les chemins possibles du message à travers le diagramme des états de transition, en supprimant au fur et à mesure les moins probables.

On code. Et on décode ensuite avec l'algorithme de Viterbi.

Pour une explication simple du fonctionnement de l'algorithme (figure I.10), on utilise la représentation du code en treillis. Ce diagramme est une variante du diagramme de transition qui représente les états possibles successifs des registres à décalage et les transitions entre ces états.

On calcule premièrement, pour chaque mot possible, sa métrique de branche, c'est-à-dire sa distance par rapport au mot reçu. Puis on calcule les métriques cumulées des différents chemins. On ne garde ensuite que les distances minimales et ainsi de suite jusqu'à la fin du bloc. *In fine*, on réalise le parcours le plus probable et on retrouve la séquence d'origine : l'erreur a été corrigée.

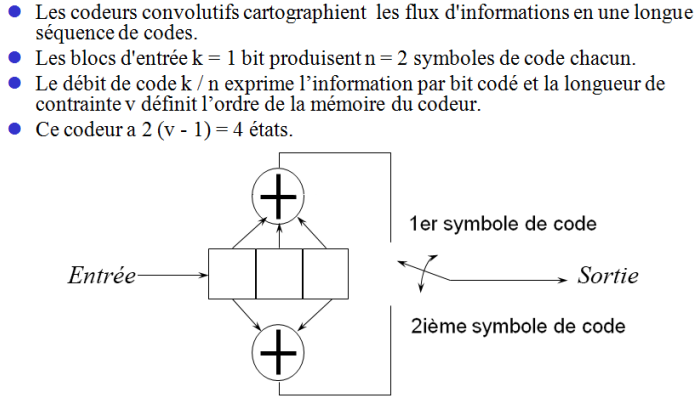

Figure I.10. *Un taux simple ½, v = 3, codeur convolutif*

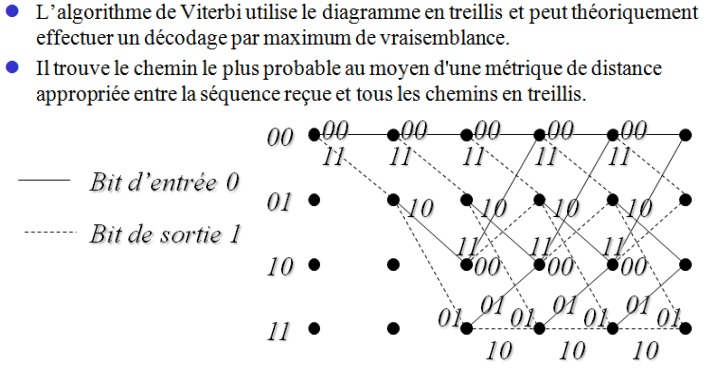

Figure I.11. *Algorithme de Viterbi*

- **BMU**: BM sont calculées à partir des données d'entrée introduites.
- **ACSU (add-compare-select unit)** : les MP de tous les états sont mis à jour conformément à l'équation (1).
- **SMU**: les décisions stockées sont utilisées dans la SMU pour créer une sortie décodée unique.

$$PM[i]_{(t+1)} = \min_{\text{tous les possibles}} \left(PM[k]_{(t)} + BM([k][i])_{(t)} \right) \quad (1)$$

$PM[k]_{(t)}$: métrique de chemin correspondant à l'état k à l'instant t.
$BM([k][i])_{(t)}$: métrique de branche de la transition de l'état k à t à l'état i à t + 1.

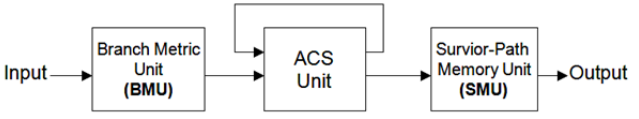

Figure I.12. *Unités de calcul de base dans le décodeur de Viterbi*

I.6.3. *Turbocodes*

I.6.3.1. *Caractéristiques*

Les turbocodes permettent de s'approcher de façon très fine de la limite théorique de Shannon, énoncée dans le théorème de Shannon-Hartley, qui définit la quantité maximale de données non erronées par un canal. On peut calculer la capacité d'un canal gaussien à l'aide de la relation : $Cc = ½ \log_2(1+2R\ E_b/N_0)$ où R représente le rendement du code, et E_b/N_0 l'énergie par bit du signal d'entrée sur la densité de bruit du canal. Plus ce rapport est faible, plus le code correcteur d'erreurs est performant, puisqu'il permet de retrouver avec une probabilité d'erreur faible le message initial malgré un bruit important. On peut tracer Cc en fonction de E_b/N_0 (dB). Sur le graphique de la figure I.13 sont représentées les limites théoriques dans le cas d'un canal continu (*Shannon limit*) et dans le cas d'un canal binaire qui a fait l'objet d'une modulation binaire de phase (*BPSK limit*).

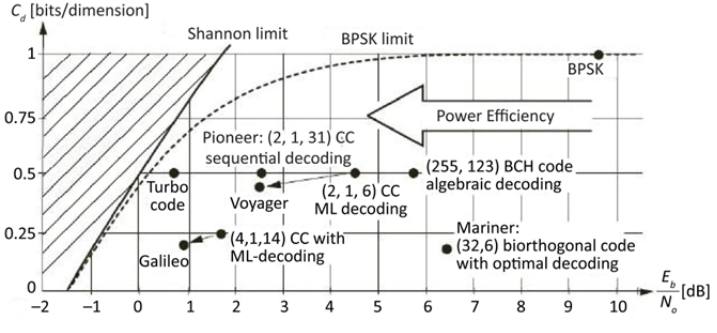

Figure I.13. *Capacité de canal versus énergie/bruit (source : © C. Schlegel, Trellis and Turbo Coding, IEEE Press, 2004)*

L'efficacité d'un turbocode est aussi définie par :

– le taux d'erreur binaire, qui permet de mesurer la proportion d'erreurs que le code ne peut pas corriger ;

– le débit, qui correspond à la vitesse d'exécution du code.

I.6.3.2. *Les différents types de turbocodes*

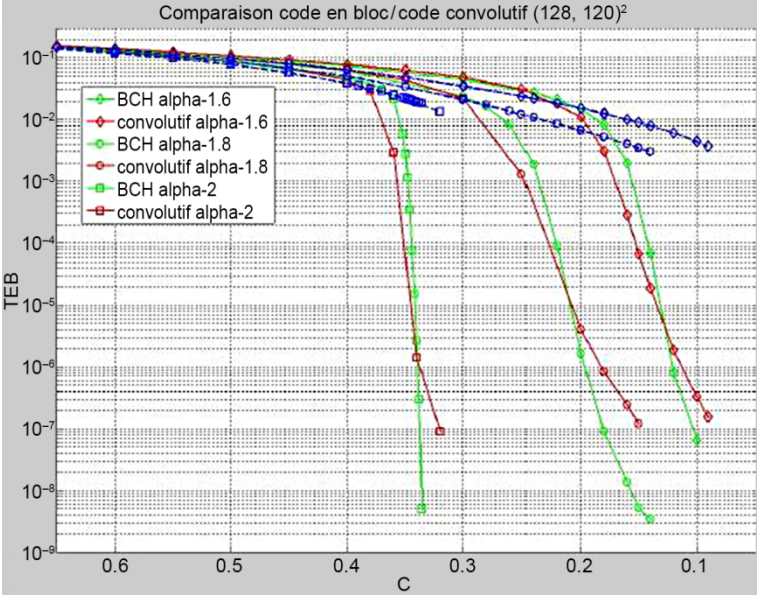

Figure I.14. *Des turbocodes*

I.6.3.2.1. Avantages des turbocodes en blocs (TCB)

Les turbocodes en blocs sont très flexibles en termes de complexité et de rendement du code. Ils supportent des blocs de n'importe quelle taille et une grande variété de rendement entre 1/3 et 0,98 mais propose surtout d'excellentes performances à des rendements élevés. Contrairement aux turbocodes convolutifs, les turbo-codes en blocs n'ont pas de plancher d'erreur. Ils peuvent être utilisés lorsque l'on veut des taux d'erreurs extrêmement faibles. De plus, les TCB possèdent des décodeurs qui peuvent agir à des vitesses très élevées.

I.6.3.2.2. Avantages des turbocodes convolutifs (TCC)

En plus de meilleures performances pour un rapport signal/bruit faible, il est plus facile de passer d'un rendement à un autre avec les turbocodes convolutifs. Le

turbocodage converge plus rapidement avec les TCC ; ils sont donc plus performants pour un taux E_b/N_0 faible. Ils sont par ailleurs plus pratiques lorsque l'on nécessite un envoi de données continu. Théoriquement, les TCC peuvent supporter n'importe quel rendement, mais en pratique, ils sont utilisés pour des rendements de $^1/_2$, $^2/_3$, $^3/_4$. De plus, la mémoire requise par un TCC est moindre que pour un TCB.

I.6.3.3. *Analogie avec les mots croisés*

Une analogie très simple permet de comprendre le principe des turbocodes : il s'agit de la grille de mots croisés avec ses deux dimensions, ligne et colonne. Les lignes, avec leurs définitions respectives, représentent le premier niveau de codage, et les colonnes le second. Les définitions correspondent à la redondance introduite par le code.

La métaphore est aussi valable pour le décodeur : le cruciverbiste utilise les définitions horizontales pour remplir certaines cases, puis les définitions verticales lui permettent de confirmer ou d'infirmer ses choix précédents et de poursuivre le remplissage des cases.

Par itérations successives, il tente ainsi de décoder la grille. Parfois, il peut échouer et des cases restent vides ou erronées : ce sont les très rares erreurs non corrigées par les turbocodes !

Voici comment Claude Berrou (Télécom Rennes) s'exprimait sur ce sujet :

> « Notre codage ajoute de la redondance à la transmission. Si, par exemple, on émet "Blanc et immaculé" en même temps, et que l'on reçoit "Flanc et immaculé", on comprendra quand même ! Avec les turbocodes, on va transmettre une sorte de grille de mots croisés avec des définitions en lignes et en colonnes. Ce qui est vérifié dans un sens est vérifié dans l'autre. C'est cela l'effet turbo, ou décodage itératif : comme dans un moteur turbo, où l'énergie de l'échappement renforce l'admission, *via* une turbine de compression, le codage par colonne renforce le codage par ligne. »

Il rajoute également le fait suivant :

> « La NASA avait créé un décodeur moins efficace, qui coûtait 5 millions de dollars et prenait un volume de deux cantines de marine au lieu de nos chips (puces électroniques) qui tiennent dans la main. »[2]

2. Sources : www.espace-sciences.org et www.institut-telecom.fr.

I.7. Synoptique historique

1969 – ARPANET (*Advanced Research Projects Agency Network*) : le premier système Internet est mis en place aux États-Unis.

1979 : au Japon, premier système de téléphonie cellulaire.

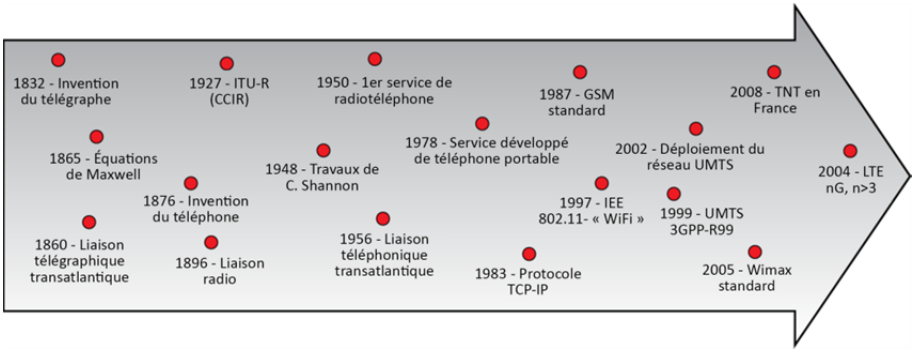

Figure I.15. *Curriculum vitae des communications téléphoniques*

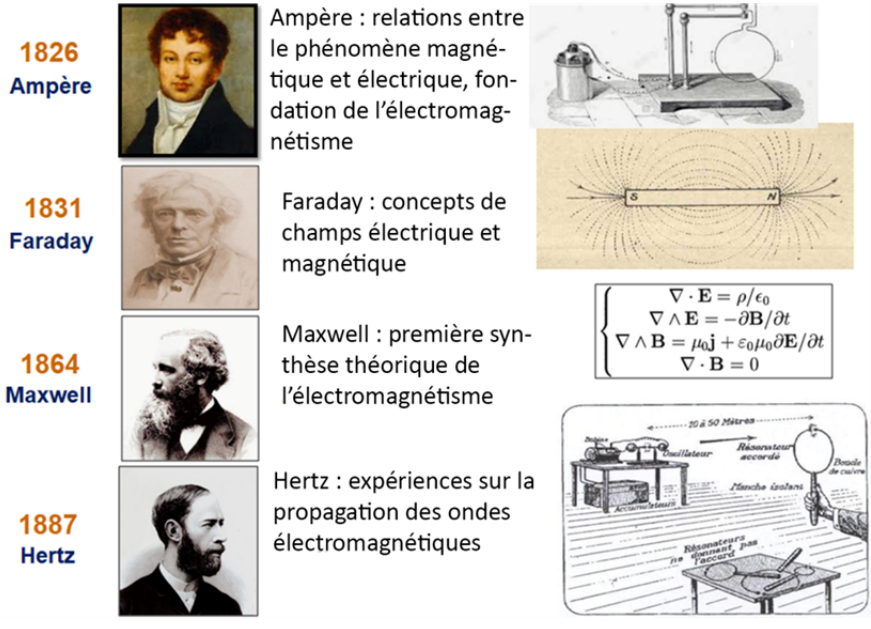

Figure I.16. *Des fondateurs de l'électromagnétisme*

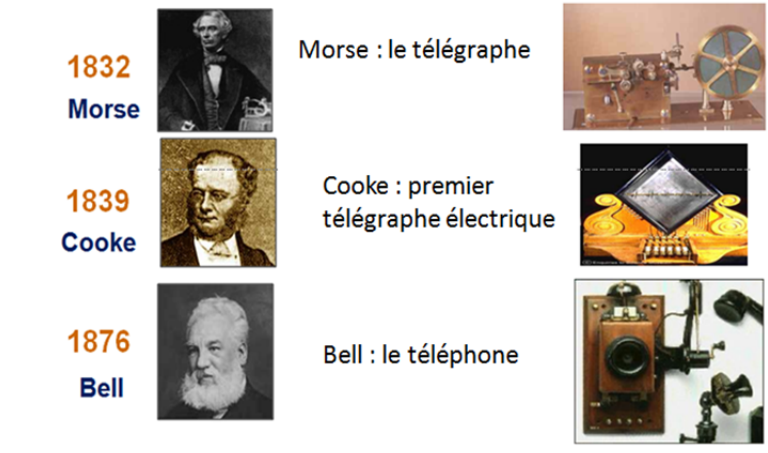

Figure I.17. *Premières communications filaires*

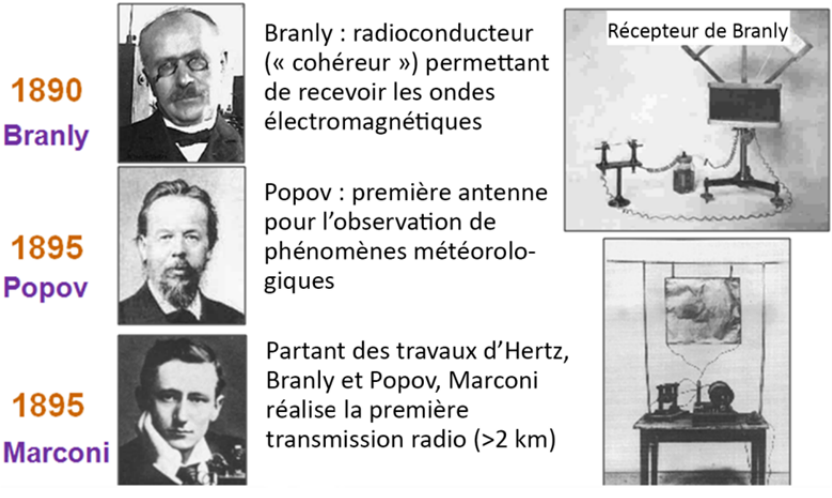

Figure I.18. *Débuts des communications radio*

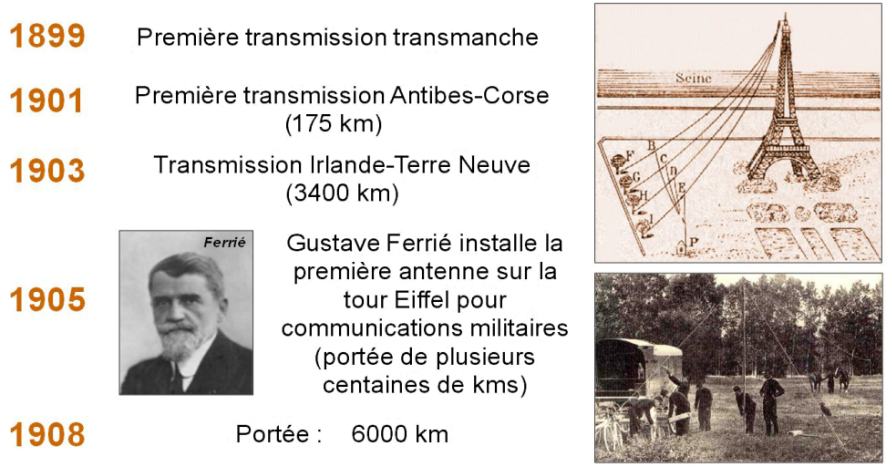

Figure I.19. *Premiers déploiements*

Figure I.20. *L'émetteur de Lyon – La Doua*