République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université 8 Mai 1945 Guelma

Faculté : Sciences et Technologies
Département : Génie Electrotechnique et Automatique

THÈSE

EN VUE DE L'OBTENTION DU DIPLOME DE DOCTORAT EN SCIENCE

Filière : Génie électrique

Présentée par

SAIDANI Samir

Intitulée

Sur l'égalisation adaptative dans les systèmes de communications numériques

Soutenue le : 09/07/2020

Devant le Jury composé de :

Mr BOUDJEHEM Djalil	Professeur	Univ. 8 Mai 1945 - Guelma	Président
Mr MOUSSAOUI Abdelkrim	Professeur	Univ. 8 Mai 1945 - Guelma	Rapporteur
Mr BAHI Tahar	Professeur	Univ. Badji mokhtar - Annaba	Examinateur
Mr ARBAOUI Faicel	MC-A	Univ. Badji mokhtar - Annaba	Examinateur

Année Universitaire : 2019/2020

Remerciement

Cette thèse s'est déroulée au sein du laboratoire de contrôle avancé (LABCAV) de l'Université 8 Mai 1945 Guelma.

Avant tout, Merci à Dieu le tout puissant qui m'a donné le courage et la force pour réaliser ce modeste travail, et à qui j'adresse mes remerciements par sa grâce infinie pour moi.

Mes remerciements s'adressent également, à mon directeur de thèse, Monsieur Abdelkrim MOUSSAOUI Professeur à l'Université 8 Mai 1945 Guelma, pour m'avoir confié le sujet de cette thèse qu'il a dirigé avec intérêt. Et aussi pour la confiance qu'il m'a accordée tout le long de mes travaux de recherche.

Je remercie Monsieur Djalil BOUDJEHEM, Professeur à l'Université 8 Mai 1945 - Guelma, pour l'honneur qu'il m'a fait en acceptant de présider ce jury. Je remercie également Monsieur Tahar BAHI, Professeur à l'Université Badji Mokhtar - Annaba, ainsi que, Monsieur Faicel ARBAOUI, Maître de Conférences à l'Université Badji Mokhtar - Annaba, de m'avoir fait l'honneur de bien vouloir participer au jury de cette thèse.

Je remercie aussi, mes amis et ma famille qui m'ont apporté à un moment ou à un autre leur aide pendant cette thèse. Je remercie tous ceux que j'ai oublies, et qui de près ou de loin, ont contribué à cette thèse.

Avant-propos

Prénom et Nom de l'auteur de la thèse : Samir Saidani

Intitulé de la thèse : Sur l'égalisation adaptative dans les systèmes de communications numériques.

Prénom et Nom du directeur de la thèse : Pr. Abdelkrim Moussaoui (Professeur à la Faculté des Sciences et de la Technologie, Université 8 Mai 1945 Guelma).

Lieu de réalisation du travail : Laboratoire de contrôle avancé (LABCAV), Faculté des Sciences et de la Technologie, Université 8 Mai 1945 Guelma.

Publication Internationales

1. **Samir Saidani**, A. Moussaoui, M. Ghadjati, "Channel Equalization Based On SFLA and DSO Trained Artificial Neural Network", Telecommunications and Radio Engineering, vol. 78, no. 17, pp. 1589-1600, 2019.

2. **Samir Saidani**, Abdelkrim Moussaoui, Mohamed Ghadjati, "A New Training Strategy for DFE-MLP Using Modified BP Algorithm and Application to Channel Equalization", WSEAS Transactions on Signal Processing, vol. 13, pp.115-120, 2017.

Communications Internationales

1. **Samir SAIDANI**, Issam TIFOUTI, Mohamed GHADJATI and MOUSSAOUI Abdelkrim, "Performance of the Multilayer Perceptron based DFE for Linear and Nonlinear Channels", Proceeding of the International Conference on Automatic Control, Telecommunications and Signals(ICATS'2015), 16-18 November 2015, Annaba, Algeria.

2. **SAIDANI Samir**, TIFOUTI Issam, GHADJATI Mohamed and MOUSSAOUI Abdelkrim, "Training Artificial Neural Networks with fast Adaptive Algorithms for Channel Equalization", International Conference on Pattern Analysis and Intelligent Systems(PAIS-2015), 26-27 October 2015, Tebessa, Algeria.

3. **Samir SAIDANI**, Abdelkrim MOUSSAOUI, Mohamed GHADJATI and Issam TIFOUTI, "Channel equalization with artificial neural network", Proceeding of the 1st International conference on applied automation and industriel diagnostics ICAAID2015, March 29-30th, 2015 Djelfa, Algeria.

Résumé

L'égalisation des canaux de transmission est une fonction principale dans les systèmes de communications numériques, elle consiste à la restitution du signal transmis, après être distordu, par l'effet du canal et du bruit.

Les égaliseurs classiques sont basés simplement sur des filtres numériques (Filtre à réponse impulsionnelle finie et Filtre à réponse impulsionnelle infinie) ce qui limite leurs performances dans le cas des canaux de transmission non linéaires. Afin d'améliorer les performances de l'égaliseur, les réseaux de neurones artificiels ont été largement utilisés, en particulier dans certains cas de canaux non linéaires. Plusieurs algorithmes sont utilisés pour optimiser les poids des réseaux de neurones. Certains utilisent les techniques basées sur le gradient. Bien que ces algorithmes soient très utiles pour l'optimisation des poids des réseaux de neurones, ils souffrent du problème de minimum local. Afin de résoudre ce problème, de nombreux algorithmes de recherche stochastiques connus sous le nom d'algorithmes méta-heuristiques ont été proposés.

Dans le présent travail, on se propose l'étude des égaliseurs adaptatifs tout en testant leurs performances. Dans ce contexte, nous avons focalisé nos recherches sur les algorithmes d'apprentissage pour améliorer la fonction d'égalisation.

Mots-clés : Egalisation, Filtrage adaptatif, canal de transmission, communication numérique, algorithmes d'apprentissage.

Abstract

Adaptive channel equalizers play an important role in digital communication systems, its main function is returning the transmitted signal after being distorted by the channel effect and noise.

Classic Equalizers are based simply on digital filters (Finite impulse response filter and Infinite impulse response filter), which limits their performance in the case of non-linear channels. In order to enhance the performance of the equalizer, artificial neural network were broadly used especially in some cases of nonlinear channels. Several algorithms are used to optimize the weight of the neural networks. Some of them use the gradient-based techniques. Although these algorithms are very helpful for training the neural networks, they suffer from the local minimum problem. In order to solve this problem, many stochastic search algorithms which are known as meta-heuristic algorithms have been proposed.

In the present work, we propose a study of adaptive equalizers and testing their performances. In this context, we focused our research on learning algorithms to improve the equalization function.

Keywords: Equalization, adaptive filtering, transmission channel, digital communication, learning algorithms.

مـلخــص

تعــد مسويــات قنــوات الإتصــال مــن الوظــائــف الرئــيسيــة فــي أنظمــة الاتصــالات الرقميــة، فهــي تعمــل علــى إسترجــاع الإشــارة المرسلــة بعــد تشــويههــا بسبب تــأثيــر القنــاة والضجيــج.

تعتمــد المسويــات التقليديــة ببساطــة علــى المرشحــات الرقميــة (مرشــح ذو استجابــة نبضيــة محــددة و مرشــح ذو استجابــة نبضيــة غيــر محــددة) والتــي تحــد مــن أدائــها فــي حالــة قنــوات الإتصــال اللاخطيــة.

مــن أجــل تحسيــن أداء المسويــات، تــم استخــدام الشبكــات العصبيــة الإصطنــاعيــة على نطــاق واســع وخاصــة فــي بــعض القنــوات اللاخطيــة.

لتحسيــن أوزان الشبكــات العصبيــة، تــم استخــدام العديــد مــن الخوارزميــات، بعضهــم يستخــدم التقنيــات القائمــة علــى التــدرج. علــى الرغــم مــن أن هــذه الخوارزميــات مفيــدة جــدًا فــي تدريــب الشبكــات العصبيــة، إلا أنــها تعانــي مــن مشكلــة الحــد الأدنــى المحليــة، لحــل هــذه المشكلــة، تــم إقتــراح العديــد مــن خوارزميــات البحــث العشوائــي المعروفــة باســم خوارزميــات الاستــدلال الفوقيــة. فــي العمــل الحالــي ، نقتــرح دراســة المعــادلات أو المسويــات التكيفيــة و إختبــار أدائــها. فــي هــذا السيــاق ركزنــا بحثنــا علــى خوارزميــات التعلــم لتحسيــن وظيفــة المعادلــة.

الكلمــات المفتاحيــة: التسويــة ،الترشيــح التكيفــي، قنــاة الإرســال ، الاتصــالات الرقميــة ، خوارزميــات التعلــم.

Table des matières

Chapitre 1

INTRODUCTION AUX COMMUNICATIONS NUMERIQUE

Chapitre 2

ALGORITHMES D'OPTIMISATIONS : ÉTAT DE L'ART

Chapitre 3

TECHNIQUES D'EGALISATION

Chapitre 4

ÉGALISATION Á BASE DES RÉSEAUX DE NEURONES ARTIFICIELS

Chapitre 5

ALGORITHMES MÉTA-HEURISTIQUE POUR L'ÉGALISATION : HYBRIDATION ET MÉTHODE PROPOSÉE

LISTE DES FIGURES

Chapitre 1

Chapitre 2

Chapitre 3

Chapitre 4

Chapitre 5

LISTE DES TABLEAUX

LISTE DES ABRÉVIATIONS

ABC	Artificial bee colony
ACO	Ant Colony Optimization
ACROA	Artificial Chemical Reaction Optimization Algorithm
ANN	Artificial Neural Network
AWGN	Additive White Gaussian Noise
BBAG	Bruit Blanc Additif Gaussien
BBO	Biogeography-Based Optimizer
BER	Bit Error Rate
BP	Back Propagation
BPSK	Binary Phase Shift Keying
CFO	Central Force Optimization
CS	Cuckoo Search
CSS	Charged System Search
DFE	Decision-Feedback Equalizer
DSO	Directed Search Optimization
DSP	Densité Spectrale de Puissance
EQMM	Erreur Quadratique Moyenne Minimale
ES	Evolution Strategy
GA	Genetic Algorithms
GbSA	Galaxy-based Search Algorithm
GLS	Guided Local Search
GLSA	Gravitational Local Search Agorithm
GP	Genetic Programming
GSA	Gravitational Search Algorithm
GSO	Group Search Optimizer

HCH	High-level Co-evolutionary Hybrid
HRH	High-level Relay Hybrid
HS	Harmony Search
IES	Interférence Entre Symboles
ILS	Iterated Local Search
I-Q	In phase–Quadrature phase
ISI	Inter–Symbol Interference
LE	Linear Equalizer
LCH	Low-level Co-evolutionary Hybrid
LMS	Least Mean Square
LRH	Low-level Relay Hybrid
MLP	Multi-Layer Perceptron
MLSE	Maximum Likelihood Sequence Estimation
MMSQ	Minimum Mean Square Error
PBIL	Probability-Based Incremental Learning
PDF	Probability Density Function
PSK	Phase Shift Keying
PSO	Particle Swarm Optimization
QPSK	Quadrature Phase Shift Keying
RBF	Radial Basis Fonction
RII	Réponse Impulsionnelle Infinie
RIF	Réponse Impulsionnelle Finie
RLS	Recursive Least Square
RNN	Recurrent Neural Network
SA	Simulated Annealing
SFLA	Shuffled Frog-Leaping algorithm
SNR	Signal–to–Noise Ratio
TEB	Taux d'erreur binaire
TLBO	Teaching Learning Based Optimization
VNS	Variable Neighbourhood Search
ZF	Zero-Forcing.

INTRODUCTION GÉNÉRALE

De nos jours, les systèmes de télécommunications font partie des technologies qui ont révolutionné notre mode de vie. Du télégraphe à l'Internet, de la transmission sans fil au téléphone cellulaire, les progrès établis sont spectaculaires. Afin de développer de tels systèmes, une parfaite connaissance des propriétés du canal de communication est nécessaire. Les performances d'un système de transmission sont en effet directement liées aux conditions de propagation entre l'émetteur et le récepteur. Ceux-ci doivent donc être dimensionnés pour tirer le meilleur parti des caractéristiques du canal et atténuer ses effets négatifs. Les systèmes de communications numériques nécessitent généralement la transmission de quantités importantes d'information dans des bandes de fréquence les plus étroites possible **[1]**. Cependant l'augmentation des besoins en débit se heurte à la nature des canaux eux-mêmes. En effet, dans des applications telles que la télédiffusion à grande échelle ou un réseau informatique radio à l'intérieur d'un bâtiment, le canal est de type multi-trajet. Le signal est réfléchi en plusieurs endroits, et des échos apparaissent et créent des perturbations telles que l'interférence entre symboles créée par la sélectivité en fréquence dont l'influence augmente avec le débit de transmission **[2]**. Pour lutter contre la sélectivité en fréquence des canaux de transmission plusieurs techniques sont possibles parmi lesquelles on peut citer : les transmissions multi-porteuses, les techniques d'étalement de spectre et l'égalisation **[1]**. Cette thèse est consacrée à l'égalisation.

La technique d'égalisation apparait comme une technique de traitement de l'interférence entre symboles efficace lorsque les canaux de transmission sont sélectifs en fréquence et invariants ou variants dans le temps. Les égaliseurs les plus utilisés en pratique sont les égaliseurs adaptatifs. Les égaliseurs adaptatifs jouent un rôle important dans les systèmes de communication numériques, sa fonction principale est de restituer le signal transmis après avoir été déformé par l'effet du canal et le bruit. Les égaliseurs adaptatifs les plus simples sont construits à partir de filtres linéaires transverses dont les coefficients sont généralement optimisés à partir d'un algorithme des moindres carrés moyens (LMS: Least Mean Square) ou des moindres carrés récursifs (RLS: Recursive Least Square). L'égalisation linéaire à coefficients fixes remonte aux travaux pionniers de Nyquist **[3]**, pendant les années 1928. Cependant, le concept d'égalisation automatique n'a été formalisé qu'en 1965, par Lucky **[4]**, tandis que les analyses de performance et convergence des égaliseurs adaptatifs basés sur la minimisation de l'erreur quadratique ont été faites entre 1969 et 1984, par des chercheurs renommés du domaine comme Proakis **[5]** et Macchi **[6]**. Pendant cette période, plusieurs types de structure linéaires d'égalisation adaptative ont été développés, notamment les égaliseurs par filtrage de Kalman, par filtrage en treillis (latice filters) et les égaliseurs fractionnés. D'autre part, dans une voie parallèle de recherche, des structures d'égalisation

adaptatives non linéaire ont été aussi développées, comme l'égaliseur à retour de décision (DFE: Decision-Feedback Equalizers) **[7]**, en 1967.

Dans les années 1970, grâce à l'apparition d'une grande modalité de système de communication numérique, y compris certains qui ne transmettaient pas des séquences d'apprentissage (une séquence préliminaire de données, connue du récepteur), une nouvelle branche de l'égalisation à été créée, à savoir, l'égalisation aveugle (ou autodidacte). Aujourd'hui, l'égalisation non supervisée comme l'égalisation aveugle ou supervisée ont des rôles centraux dans les systèmes de communication numériques.
Pour améliorer les performances de l'égaliseur dans certains canaux non linéaires, les réseaux de neurones artificiels (ANN : Artificial Neural Network) **[8]** ont été largement utilisés dans le domaine de l'égalisation des canaux **[9-14]**. Les réseaux de neurones constituent un ensemble des techniques de traitement du signal qui s'inspirent du système nerveux humain. Ces techniques trouvent une place dans le domaine des télécommunications, en particulier dans les systèmes comportant des non-linéarités. Deux architectures d'égaliseurs à base des réseaux de neurones sont mises en œuvre pour l'égalisation du canal de communication, l'égaliseur direct, qui est la version neuronale de l'égaliseur linéaire et l'égaliseur à retour de décision (DFE : Decision-Feedback Equalizers).
L'optimisation d'un réseau de neurone implique l'utilisation d'un algorithme adaptatif pour optimiser les poids des réseaux neuronaux. Bien que plusieurs algorithmes se trouvent dans la littérature pour optimiser les poids des réseaux de neurones. Certains utilisent les techniques basées sur le gradient comme l'algorithme de rétro-propagation du gradient (BP : pour Back Propagation) **[15]**. Cependant, ces algorithmes, sont sensibles au problème de minimum local. Afin de résoudre ce problème, de nombreux algorithmes de recherche stochastiques connus sous le nom d'algorithmes méta-heuristiques ont été proposés. Par conséquent, un algorithme d'apprentissage est souvent modifié, voire hybridé avec un autre afin d'améliorer leurs performances. Actuellement, les méta-heuristiques hybrides sont devenues plus populaires car les meilleurs résultats trouvés pour plusieurs problèmes d'optimisation ont étés obtenus avec des algorithmes hybrides. Dans ce contexte, Nous allons focaliser nos recherches sur les algorithmes d'apprentissage destinés à résoudre des problèmes d'égalisation des canaux.

Cette thèse est organisée en 5 chapitres (figure 1), suivis d'une conclusion générale et des perspectives. Elle est présentée comme suit :

- Le premier chapitre représente un récapitulatif des outils des communications numériques, les nécessaires, pour avancer dans cette thèse, parmi lesquels on cite quelques modèles des canaux de transmission et les perturbations qui les accompagnent, telles que le bruit et l'interférence entre symboles, et les outils de mesure de la qualité de transmission. A la fin de ce chapitre, on donne un aperçu sur les filtres numériques.

- Le deuxième chapitre dresse un état de l'art sur les algorithmes d'optimisation utilisés dans le filtrage adaptatif en accordant une importance particulière aux algorithmes utilisés dans nos travaux de recherche. Il aborde aussi les définitions générales des méthodes d'optimisation qui se divisent en deux volets déterministes et non déterministes.

- Le troisième chapitre explique la notion d'égalisation, et présente une étude des égaliseurs conventionnels, l'égaliseur linéaire LE (Linear Equalizer) et l'égaliseur à retour de décision

DFE (Decision Feedback Equalizer), et les différents critères d'optimisation. Ce chapitre présente aussi une comparaison entre les deux égaliseurs LE et DFE en termes de performances, et montre leurs limitations face aux non linéarités du canal. Les coefficients de ces deux égaliseurs sont actualisés à partir de l'algorithme des moindres carrés moyens (LMS), et l'algorithme des moindres carrés récursifs (RLS).

- Le quatrième chapitre se concentre sur l'utilisation des réseaux de neurones pour la fonction d'égalisation, c'est-à-dire un traitement non linéaire. Il présente une étude des réseaux de neurones, leurs types, le perceptron multicouche (MLP : Multi Layer Perceptron) et le réseau de fonction à base radiale (RBF : Radial Basis Functions). Les simulations examinent l'efficacité de ces égaliseurs à réduire l'effet du bruit et du canal utilisé, sur un signal BPSK (Binary Phase Shift Keying). Les performances des égaliseurs présentés sont aussi comparées et évaluées avec une méthode proposée et avec différents outils de mesure de performance tels que les courbes de convergence de l'erreur quadratique moyenne (EQM ou MSE : Mean Square Error), et les courbes des taux d'erreurs binaires (TEB ou BER: Bit Error Rate).

- Le cinquième chapitre se concentre sur l'utilisation des algorithmes de recherche stochastiques connus sous le nom d'algorithmes méta-heuristiques dans l'égalisation des canaux. Nous présentons d'abord d'une manière détaillée l'hybridation des algorithmes méta-heuristiques. Ensuite, nous présentons notre méthode d'optimisation qui est basée sur deux algorithmes méta-heuristiques. Enfin, nous présentons les résultats de simulation de la méthode proposés sur le problème de l'égalisation des canaux non linéaires tout en effectuant une étude comparative.

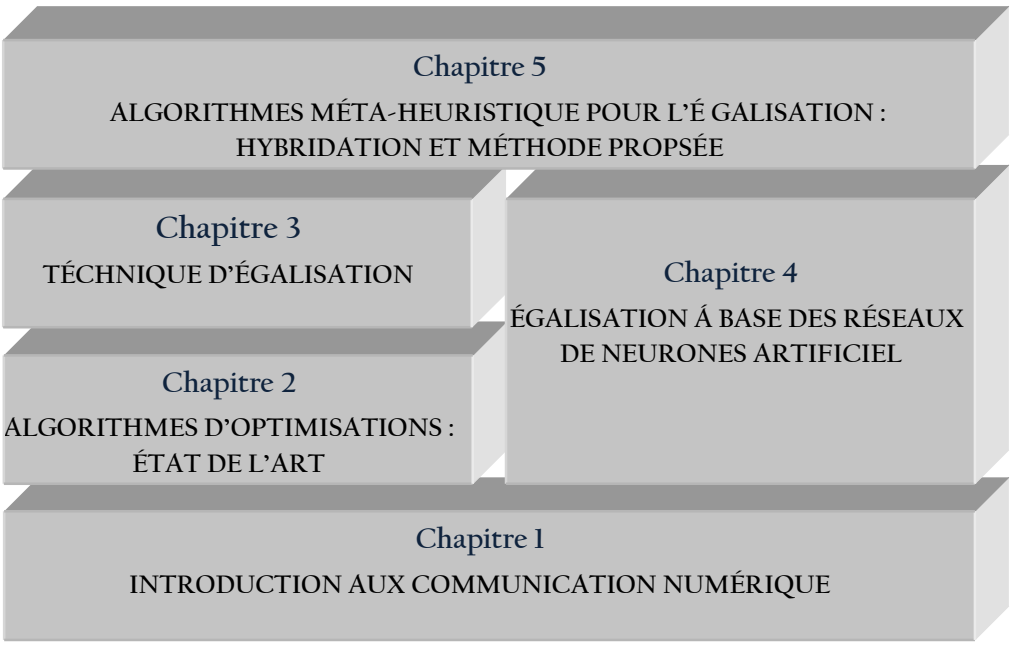


Figure 1 *: Schéma d'organisation de la thèse.*

CHAPITRE 1

INTRODUCTION AUX COMMUNICATIONS NUMERIQUES

1.1 INTRODUCTION

Les systèmes de communication numériques deviennent de plus en plus attrayants en raison de la demande sans cesse croissante pour la communication de données et du fait que la transmission numérique offre des options de traitement de données et des marges de flexibilité non disponibles avec la transmission analogique **[16]**. Les signaux numériques présentent en effet plusieurs propriétés intéressantes pour les télécommunications : souplesse de traitement, signal à états discrets donc moins sensibles aux bruits et simple à régénérer, utilisation de codes correcteurs d'erreur, cryptage de l'information …etc **[17]**. L'objective de cette partie est de présenter les principales notions relatives aux systèmes de communication numérique, qui vont servir à la bonne compréhension de cette thèse. Dans un premier temps, nous allons voir la chaine de communication point-à-point et la plupart de ses modules. Ensuite, nous présenterons la description du canal de transmission dans le cas général, nous allons étudier les différents types de canaux de transmission et des méthodes de transmission adaptées à chacun des types. Les phénomènes perturbateurs qui nuisent aux signaux utiles seront aussi présentés.

1.2 DESCRIPTION D'UN SYSTEME DE COMMUNICATION NUMERIQUE

1.2.1 Chaine de transmission numérique

Les systèmes de transmission numérique véhiculent de l'information sous formes numériques entre une source et un ou plusieurs destinataire en utilisant un support physique comme le câble, la fibre optique ou encore la propagation sur un canal radioélectrique **[18]**. Les signaux transportés peuvent être soit directement d'origine numérique comme dans les réseaux de données, soit d'origine analogique (parole, image…) mais convertis sous une forme numérique. La tâche du système de transmission est d'acheminer le signal de la source vers le destinataire avec le plus de fiabilité possible. Le schéma synoptique d'un système de transmission numérique est donné à la figure 1.1, où l'on se limite aux fonctions de base **[19]** :

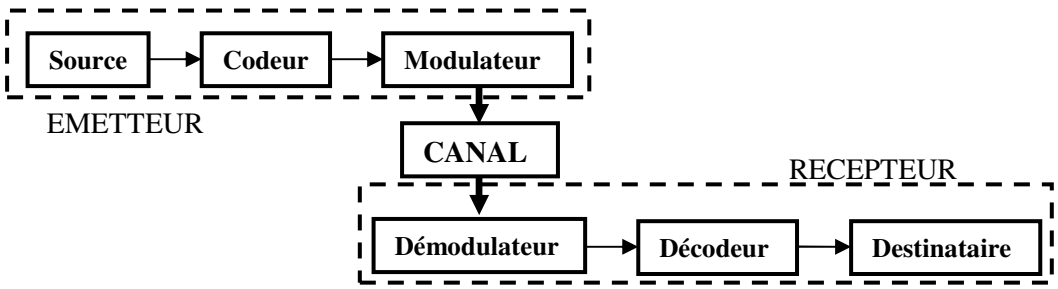


Figure 1.1 : *Schéma d'un système de transmission numérique.*

- **La source** peut être soit de forme analogique ou numérique. Pour réaliser une transmission numérique, le message à transmettre doit être sous forme numérique. Si le message produit par la source est de type analogique tel que le signal de parole (sortie d'un microphone) ou le signal d'image (sortie d'une caméra), il faut converti en une séquence d'élément binaire (numériser), à travers un étage de conversion analogique numérique (Echantillonnage et Quantification). Chaque échantillon quantifié est ensuite codé sur *m* élément binaire (appelés traditionnellement, mais improprement, *bits*) **[20]**.
  La taille du message binaire original ainsi produit est en général très importante et contient de la redondance. Dans le cas idéal, et pour augmenter l'efficacité de la transmission et optimiser l'utilisation des ressources du système, cette séquence doit être la plus courte possible, ce qui constitue la tâche du codeur source.

- **Le codeur** peut éventuellement supprimer des éléments binaires non significatifs (compression de données ou codage de source), ou au contraire introduire de la redondance dans l'information en vue de la protéger contre le bruit et les perturbations présentes sur le canal de transmission (codage de canal).

  Le codage source représente donc la source avec un minimum de bits sans en diminuer la quantité d'information, c'est-à-dire de délivrer une source aussi proche que possible d'une source idéale. Dans le cas idéal, cette séquence doit être la plus courte possible. Pour augmenter l'efficacité de la transmission et optimiser l'utilisation des ressources du système, un codeur de source compresse donc les données en éliminant les éléments binaires non significatifs. Cette séquence binaire en sortie du codeur de source est appelée séquence d'information. Le codage source peut aussi comporter une étape de cryptage dans le cas où l'on souhaite sécuriser le transfert des données.

  Lors du passage dans le canal physique de transmission, le signal est altéré par du bruit et des interférences, induisant parfois le récepteur en erreur. Afin d'augmenter la fiabilité de la transmission, un codeur du canal introduit, de manière parfaitement contrôlée, de la redondance dans la séquence d'information afin de la protéger contre les différentes perturbations. Ce codage est encore appelé codage détecteur et correcteur d'erreurs puisque le récepteur connaît la loi de codage utilisée, et donc capable de détecter puis éventuellement corriger les données binaires erronées. Cependant, cette amélioration de la qualité du message se fait au détriment du débit global de transmission et si l'on se réfère de plus aux travaux conduits par Shannon sur la théorie de l'information, le codage du canal n'est possible que si le débit de la source binaire est inférieur à la capacité du canal de transmission. Le codage canal est réalisé uniquement en bande de base. Une fois le codage est réalisé le signal numérique est transformé en un signal physique capable de transiter sur le canal de transmission, ce qui constitue la tâche du modulateur **[21]**.

- **Le modulateur** a pour rôle d'adapter le spectre du signal au canal (milieu physique) sur lequel il sera émis. Le milieu de transmission physique autorise uniquement la transmission de signaux analogique, il est nécessaire de convertir l'information numérique en un signal analogique avant sa transmission à travers le milieu de transmission. Le modulateur numérique est donc l'interface qui associe l'information numérique à des signaux analogiques adaptés aux caractéristiques du canal. Dans la plupart des systèmes de télécommunication, la bande de fréquence allouée est faible au regard du débit souhaité. Afin d'augmenter le débit sans pour autant augmenter la bande passante, des modulations à plusieurs états ou modulations M-Aire ont été développées **[22]**. Le principe de cette modulation est d'associer à chaque groupe de M symboles binaires de durée T_b un symbole complexe de durée T_s. Généralement, il prend des blocs de $k=\log_2(M)$ bits de la séquence à transmettre et les associe à l'une des $M=2^k$ formes d'ondes (appelés symboles) pour la transmission à travers le canal. M représente le nombre d'états de la modulation. Par exemple, la modulation BPSK qui associe $k=1$ bit à $M=2$ symboles possibles (suivant la valeur du bit) est dite modulation à deux états de phase (ou Biphase). En pratique, il est possible de créer une modulation numérique M-aire en créant des signaux à M états, c'est-à-dire en donnant plus de 2 états possibles au signal modulant. De nombreuses modulations M-aires sont basées sur un modulateur I/Q (Inphase / Quadrature). Celui-ci consiste à séparer le flux binaire en entée en 2 flux parallèle et de les multiplier par deux porteuse de même fréquence mais en quadrature. En les recombinants, on obtient un signal modulé en phase. De manière générale, un signal modulé en phase M-aire peut s'exprimer de la manière suivante :

$$s_i(t) = A_p \cos(2\pi f_c t + \varphi_i), \quad \varphi_i = \frac{2\pi i}{M}, \quad i \in [0; M-1] \tag{1.1}$$

$s_i(t)$: le signal modulé à l'instant t, qui transmet un symbole i, Il existe M symboles possible.
A_p : Amplitude constante du signal modulé en phase.
f_c : Fréquence porteuse.
φ_i : L'angle de la modulation pour le symbole i.

Le signal modulé en 4-PSK ou QPSK (Quadrature Phase Shift Keying) correspond donc aux 4 états de phase de la modulation et est codé par 2 bits.
On distingue donc le débit binaire (D_b) qui est le débit de la séquence à transmettre (nombre de bits par seconde) et le débit symbole (D_s) qui est le nombre de blocs transmis. Le débit symbole est aussi appelé rapidité de modulation et il s'exprime en bauds, c'est le nombre de symboles transmis par unité de temps **[20]**:

$$D_b = \frac{1}{T_b} \quad bit/s \tag{1.2}$$

$$D_s = \frac{1}{kT_b} = \frac{D_b}{\log_2(M)} \quad bauds \tag{1.3}$$

On appelle efficacité spectrale le rapport du nombre de bits transmis par seconde avec la largeur de bande utilisée **[23]**. On pourrait montrer que l'efficacité spectrale d'une modulation M-aire est égale à :

$$\eta_{M-aire} = N, \quad M = 2^N \tag{1.4}$$

L'efficacité spectrale d'une modulation QPSK est donc égale le double de celle d'une modulation BPSK. Ce qui se comprend intuitivement puisqu'on fait passer N bits par symbole. Plus on augmente le nombre de symbole M, meilleure est l'efficacité spectrale. Pour un canal à bande passante donnée, le débit binaire augmentera avec M.

- **Le canal** est un élément important dans la chaine de transmission, il assure le lien entre l'émetteur et le récepteur permettant le transfert de l'information. Donc il constitue le support physique de l'information. Il peut être un support guidé (câble coaxial, paire torsadée ou fibre optique) ou l'espace libre utilisant la propagation d'une onde électromagnétique dans l'atmosphère. Une connaissance fine des mécanismes mis en jeu est indispensable à la conception d'une chaîne de communication et à l'estimation des performances optimales **[18]**. Le canal dont les caractéristiques exactes peuvent être inconnues, introduit généralement des distorsions linéaires ou non linéaires du signal utile et un délai de propagation dus aux perturbations aléatoires en provenance de phénomènes physiques indépendants du signal utile. Ces perturbations plus ou moins importantes sont créés par les conditions de propagation (imperfection des équipements, les propagations multi-trajets, présence de bruiteurs, affaiblissements...). Ces dernières provoquent une dégradation du signal émis qui se traduit par l'apparition d'erreurs de transmission. Ces erreurs peuvent être très gênantes pour la restitution fidèle de l'information au destinataire. Ce canal sera expliqué en détail et on peut résumer son effet sur le signal transmis.

- Enfin, du côté **récepteur**, pour retrouver l'information binaire initiale, les fonctions de démodulation et de décodage sont les inverses respectifs des fonctions de modulation et de codage situées du côté émetteur. L'information binaire n'arrive pas toujours intacte au destinataire et les performances du système de transmission dépendent de très nombreux facteurs, parmi lesquels on peut citer les caractéristiques du canal, la puissance de l'émetteur, la forme d'onde utilisée ou encore le type de codage. Le bruit est le terme générique qui regroupe l'ensemble des perturbations subies par le signal lors de son passage dans le canal de transmission. Afin de mesurer ces perturbations, on appelle donc rapport signal sur bruit (RSB ou SNR pour : Signal Noise Ratio) le rapport entre la puissance totale du signal émis et la puissance du bruit au niveau du récepteur. La fréquence à laquelle les erreurs se produisent constitue une bonne indication de la fiabilité de la communication, pour la quantifier, on définit le taux d'erreur binaire (TEB ou BER pour : Bit Error Rate) comme le rapport entre le nombre de bits erronés et le nombre total de bits émis **[24]**.

1.2.2 Critère de choix entre les techniques de transmissions

Les caractéristiques principales permettant de comparer entre les différentes techniques de transmission sont les suivantes:

- **Le taux d'erreur binaire (TEB),** plus connue sous l'acronyme BER pour : Bit Error Rate, permet d'évaluer la qualité d'un système de transmission. Elle est fonction de la technique de transmission utilisée, mais aussi du canal sur lequel le signal est transmis, il correspond au rapport entre le nombre de bits erronés et le nombre total des bits émis. Ce critère est le plus adapté pour mesurer la performance des algorithmes, puisque le but est de transmettre le maximum d'information avec le minimum d'erreurs possibles. Le BER n'est jamais strictement nulle, mais cela ne signifie pas pour autant que la qualité de transmission est mauvaise, en effet il suffit qu'elle prenne une valeur suffisamment faible pour satisfaire un certain critère de fidélité **[25-27]**.
- **Erreur quadratique moyenne (EQM),** plus connue sous l'acronyme MSE pour : Mean Square Error, est utilisée comme un paramètre de mesure de performance. Le MSE détermine la moyenne du carré de l'erreur entre les symboles émis et les symboles estimés. Il est pratique de chercher à minimiser l'erreur quadratique car c'est une fonction quadratique facilement dérivable **[21]**. Evidemment, plus l'erreur quadratique moyenne sera faible, plus l'estimation sera bonne.
- **L'occupation spectrale** du signal émis doit être connue pour utiliser efficacement la bande passante du canal de transmission. Les différentes modulations se caractérisent par des efficacités différentes, en termes d'occupation spectrale et de puissance émise nécessaire pour obtenir une certaine probabilité d'erreur. On cherche en général à minimiser la probabilité d'erreur pour des conditions de transmission données **[23]**.
- **Le rapport signal sur bruit (RSB),** plus connu sous l'acronyme SNR pour : Signal to Noise Ratio. Le SNR est généralement adopté en transmission numérique comme paramètre d'entrée du récepteur pour lequel on va évaluer la qualité du message numérique restitué, il permet ainsi de qualifier la sensibilité du récepteur aux perturbations subies par le signal lors du passage dans le canal. Le SNR est déterminé par le rapport E_b/N_0 avec N_0 la densité spectrale de puissance du bruit blanc en entrée du récepteur et E_b est l'énergie moyenne par bit du signal modulé. En effet, plus le rapport signal à bruit est faible, plus le signal est dégradé par le bruit et plus il sera difficile de supprimer l'influence du bruit sur le signal. Il nécessaire de garantir un rapport signal à bruit important pour s'assurer que le signal reçu reste une copie fidèle du signal transmis **[21]**.
- **La complexité du récepteur** dont la fonction est de restituer le signal émis est le dernier aspect important d'un système de transmission. Le développement des transmissions numériques s'est appuyé sur les progrès rapides réalisés dans le domaine des circuits intégrés de traitement des signaux. L'utilisation de solutions intégrées devient indispensable au fur et à mesure que le niveau de complexité des systèmes s'accroît et que le prix consenti par le consommateur diminue **[28]**.

1.3 CARACTÉRISATION ET MODÉLISATION D'UN CANAL DE TRANSMISSION

Chaque système de communication a un modèle de canal approprié. Les modèles de canaux sont utilisés pour introduire des phénomènes physiques (distorsion, bruit, retard…) liés au médium de transmission. Suivant le phénomène physique considéré, le modèle du canal varie. La modélisation du canal de communication peut être simple ou très complexe selon la nature du milieu de propagation, qui peut se comporter comme un simple filtre linéaire de réponse en fréquence *C(f)*, mais aussi être non stationnaire (la réponse *C(f)* est alors fonction du temps) ou présenter des non-linéarités ou encore un effet Doppler **[20, 21]**. Voici quelque définition **[22]** :

- **Canal discret :** l'ensemble des symboles reçus après le passage dans le canal est fini. L'information est donc numérique.
- **Canal sans mémoire :** le symbole reçu à un instant donné dépend uniquement du symbole émis au même instant t (en considérant le retard de transmission nul).
- **Canal avec mémoire :** Si le canal a une mémoire, la sortie dépend aussi des symboles présents en t', t'<t.
- **Canal stationnaire :** ses caractéristiques sont fixes au cours du temps. Une fibre optique est un canal stationnaire, ses caractéristiques étant quasi invariantes au cours du temps, alors qu'une liaison radio correspond à un canal non stationnaire puisque ses caractéristiques dépendent de nombreux facteurs tels que les objets environnants, les conditions atmosphériques ou les perturbations électromagnétiques.
- **Canal sélectif :** le signal à transmettre a des composantes fréquentielles qui sont atténuées différemment par le canal de propagation. Il introduit donc une distorsion dans le signal transmis.

Nous allons maintenant présenter quelques modèles de canaux de transmission prenant en compte l'ajout de bruit par le canal. Dans les chapitres suivants, nous verrons comment inclure les effets temporels induit par le canal.

1.3.1 Canal à bruit additif

Le modèle mathématique le plus simple pour un canal de communication est le canal à bruit additif, illustré à la figure 1.2.
Dans ce modèle, le signal transmis *s(t)* est corrompu par un bruit additif aléatoire *v(t)*.
Le bruit peut être caractérisé de plusieurs manières **[22]** :

- Par sa densité spectrale de puissance (DSP), c'est-à-dire la répartition énergétique en fonction de la fréquence (puissance par hertz). La quantité totale de bruit sur une bande de fréquence donnée (par exemple la puissance) est égale à l'intégrale de la DSP sur cette bande de fréquence
- Par sa fonction de répartition ou densité de probabilité en amplitude (Fig. 1.3), et aussi par différentes valeurs statistiques comme sa valeur moyenne et sa variance.

Le bruit est une perturbation aléatoire dont les origines sont le milieu de transmission (bruit externe), ou les dispositifs électroniques utilisés dans le récepteur (bruit interne). Parmi les sources de bruit externes, on peut citer les rayonnements divers captés par l'antenne (cas des transmissions en espace libre), les interférences éventuelles entre les différents utilisateurs du milieu de transmission ou encore les bruits d'origine industrielle (moteurs, lignes à haute tension, etc.). Le bruit interne a pour origine le mouvement brownien des électrons dans les composants passifs (résistances) et les composants actifs (semi-conducteur) qui constituent les dispositifs du récepteur (amplificateurs, filtres, mélangeurs, etc…). Le bruit engendré par les composants passifs est un bruit blanc (densité spectrale de puissance uniforme), au moins dans le domaine de fréquence utilisé en radiocommunication, qui dépend de la température, d'où son nom de bruit thermique. Les composants actif (semi-conducteurs) sont aussi générateurs de bruit divers, dont le bruit dit de grenaille est sans doute le plus important. Le bruit de grenaille est aussi blanc mais indépendant de la température, c'est donc un bruit non thermique, fonction du courant qui traverse les composants. Compte tenu du fait qu'il existe un grand nombre d'électrons dans la matière, évoluant indépendamment les uns des autres et suivant une même loi, le bruit interne peut être modélisé par un processus gaussien d'après le théorème de la limite centrale. Par conséquent, le modèle mathématique résultant pour le canal est généralement appelé canal a bruit gaussien additif. Parce que ce modèle de canal s'applique à une large classe de canaux de communication physiques et en raison de sa facilité de traitement mathématique, il s'agit du modèle de canal prédominant utilisé dans notre analyse et notre conception de système de communication. L'atténuation du canal est facilement intégrée au modèle. Lorsque le signal subit une atténuation lors de la transmission par le canal, le signal reçu est donnée par la formule suivante **[24]** :

$$r(t) = \alpha s(t) + v(t) \tag{1.5}$$

Où : α est le facteur d'atténuation.
r(t) : Signal à la sortie du canal
v(t) : Bruit additif gaussien.

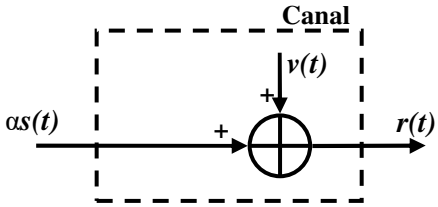


Figure 1.2 : *Modèle d'un canal à bruit additif.*

Un bruit gaussien suit une distribution gaussienne, caractérisée par une moyenne μ et une variance σ^2. La densité de probabilité est donnée par l'équation (1.6). La figure (1.3) illustre la représentation temporelle d'un bruit gaussien et la distribution statistique qui peut en être extrait, dont la densité de probabilité suit une distribution gaussienne. La représentation temporelle ne permet pas d'extraire d'informations sur le signal en raison de sa nature aléatoire (pas de période par exemple), mais la distribution permet d'extraire des éléments statistiques sur la nature du bruit. **[22]**

$$p(x)=\frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{1.6}$$

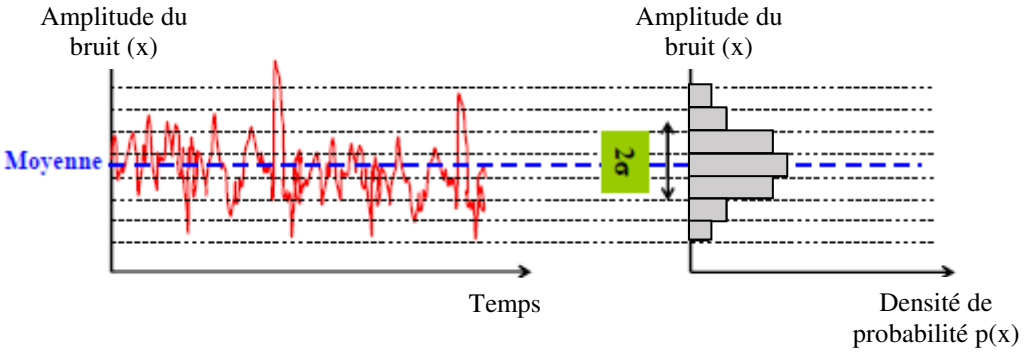


Figure 1.3 : *Représentation temporelle d'un bruit gaussien et distribution statistique de son amplitude.*

Connaître la puissance du bruit P_b n'a un intérêt que si on peut la comparer à celle du signal de sortie P_s et en déduire son impact sur la dégradation du signal. C'est pourquoi on utilise généralement un rapport de puissance appelé rapport signal sur bruit (Signal Noise Ratio).
Le rapport signal sur bruit se rapporte toujours au niveau nominal du signal. Le plus souvent, celui-ci est exprimé en décibel (dB).

$$SNR(dB)=10\times\log\left(\frac{P_s}{P_b}\right) \tag{1.7}$$

Celui-ci va donc permettre d'apprécier la qualité d'un signal et déterminer la sensibilité d'un dispositif pour une densité spectrale du bruit donnée. En effet, plus le rapport signal à bruit est faible, plus le signal est dégradé par le bruit et plus il sera difficile de supprimer l'influence du bruit sur le signal.

1.3.2 Canal à filtre linéaire (modélisé par un filtre linéaire)

Le bruit n'est pas la seule source de perturbations, la fonction de transfert du canal introduit une distorsion au signal lors de sa propagation. Les caractéristiques temporelles du canal tendent à étaler le temps de transmission d'un symbole, augmentant le risque de chevauchement de plusieurs symboles adjacents et limitant le débit de transmission admissible sur ce canal.
Dans certains canaux physiques, tels que les canaux téléphoniques filaires, des filtres sont utilisés pour garantir que les signaux transmis ne dépassent pas les limites de bande passante spécifiées et n'interfèrent donc pas les uns avec les autres. Ces canaux sont généralement caractérisés mathématiquement en tant que canaux de filtrage linéaires avec bruit additif, comme illustré à la figure 1.4. Par conséquent, si l'entrée du canal est le signal *s(t)*, la sortie du canal est donnée par la formule suivante **[24]** :

$$\begin{aligned} r(t) &= s(t)*h(t)+v(t) \\ &= \int_{-\infty}^{+\infty} h(\tau)s(t-\tau)\,d\tau + v(t) \end{aligned} \tag{1.8}$$

Où *h(t)* est la réponse impulsionnelle du filtre linéaire et * désigne la convolution.

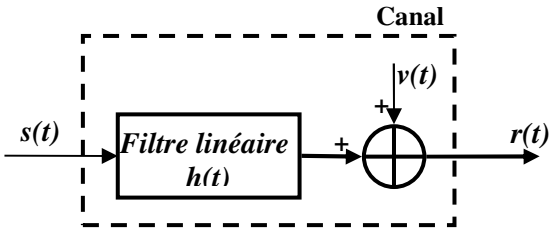


Figure 1.4 : *Modèle d'un canal à filtre linéaire avec bruit additif*

Le modèle discret utilisé pour décrire les effets de distorsion du canal est représenté par la fonction de transfert dans le domaine Z **[24]**, définie par l'équation suivante :

$$H(z)=\sum_{i=0}^{N-1} h_i z^{-i} = h_0 + h_1 z^{-1} + h_2 z^{-2} + ... + h_{N-1} z^{N-1} \tag{1.9}$$

Ce modèle est illustré à la figure 1.5.

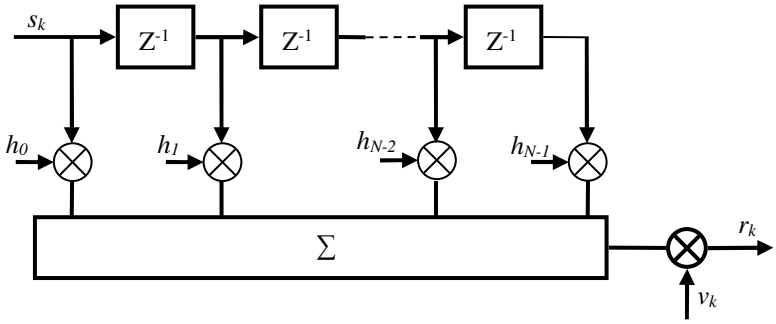


Figure 1.5 : *Modèle d'un canal discret linéaire.*

1.3.3 Canal à filtre linéaire variant dans le temps

Le signal transmis à travers des canaux physiques tels que les canaux acoustiques sous-marins où les canaux radio subit des trajets multiples de propagation variant dans le temps. L'action de ces canaux sur le signal transmis peut être caractérisée mathématiquement en tant que filtres linéaires variant dans le temps (Figure 1.6). Ces filtres linéaires sont caractérisés par une réponse impulsionnelle variante dans le temps $h(\tau;t)$.Où $h(\tau;t)$ est la réponse du canal à l'instant t due à une impulsion appliquée à l'instant $(t-\tau)$. Donc τ représente le temps écoulé (variable).

Le signal de sortie du canal à filtre linéaire variant dans le temps est donné par la formule suivante **[24]** :

$$\begin{aligned} r(t) &= s(t) * h(\tau;t) + v(t) \\ &= \int_{-\infty}^{+\infty} h(\tau;t)\, s(t-\tau)\, d\tau + v(t) \end{aligned} \tag{1.10}$$

Un bon modèle pour la propagation de signaux par trajets multiples via des canaux physiques, tels que les canaux radio mobile cellulaire, est un cas particulier de l'équation (1.10) dans lequel la réponse impulsionnelle variante dans le temps a la forme suivante:

$$r(t)=\sum_{k=1}^{L} a_k(t)\, s(t-\tau_k) + v(t) \tag{1.11}$$

Par conséquent, le signal reçu est constitué de *L* trajets multiples, chaque trajet est atténuée par $\{a_k(t)\}$ et retardée de$\{\tau_k\}$. L'amplitude et la phase du signal reçu apparaissent comme des variables aléatoires qui suivent une loi de Rayleigh. Un canal de Rayleigh permet de prendre en compte ces effets : réflexions multiples, évanouissements, fluctuations à grande et petite échelle et effet Doppler.

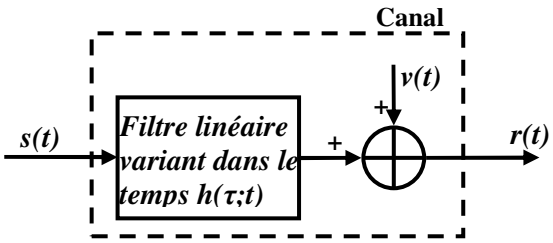


Figure 1.6 : *Modèle d'un canal de filtrage linéaire variant dans le temps avec bruit additif*

1.3.4 Canal à filtre non linéaire

Les canaux non linéaires sont communément rencontrés dans la pratique, les distorsions non linéaires peuvent survenir en raison des phénomènes de saturation des amplificateurs, ou provenir des processus de modulation. L'effet non linéaire du canal introduit des distorsions non linéaires d'amplitude et de phase. Un model introduit par Volterra consiste à déterminer une expression analytique de la réponse d'un système non linéaire à mémoire. La série de Volterra prend la forme d'une série de fonctions intégrales comparables à la convolution des systèmes linéaire.
Soit un signal *x(t)* se présentant à l'entrée d'un système non linéaire, la sortie *z(t)* est donnée par la série de Volterra du nième degré **[30-32]** :

$$z(t)=\int_{-\infty}^{+\infty} h_1(\tau_1)\,x(t-\tau_1)\,d\tau_1+\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} h_2(\tau_1,\tau_2)\,x(t-\tau_1)\,x(t-\tau_2)\,d\tau_1 d\tau_2+$$
$$....+\int_{-\infty}^{-\infty}...\int_{-\infty}^{+\infty} h_n(\tau_1,...,\tau_n)\,x(t-\tau_1)\,x(t-\tau_2)....x(t-\tau_n)\,d\tau_1...d\tau_n \qquad (1.12)$$

Où $h_1(\tau_1), h_2(\tau_1,\tau_2), h_n(\tau_1,...,\tau_n)$ sont appelés les noyaux de Volterra, chaque noyau représente un invariant du système indépendant du signal d'excitation.
La forme discrète de la série de Volterra est donnée par **[30]** :

$$z(n)=h_0+\sum_{m_1=0}^{\infty} h_1(m_1)\,x(n-m_1)+\sum_{m_2=0}^{\infty}\sum_{m_1=0}^{\infty} h_2(m_1,m_2)\,x(n-m_1)\,x(n-m_2)+$$
$$....+\sum_{m_1=0}^{\infty}\sum_{m_2=0}^{\infty}...\sum_{m_p=0}^{\infty} h_p(m_1,...,m_p)\,x(n-m_1)\,x(n-m_2)....x(n-m_p) \qquad (1.13)$$

Où *x(n)* est le signal d'entée, *z(n)* est le signal de sortie, *n* la variable indépendante. $h_p(m_1,....m_p)$ est le noyau de Volterra d'ordre *p*.

Une autre forme d'exprimer l'expression (1.12) est la suivante **[31, 32]** :

$$z(t) = H_1\left[x(t)\right] + H_2\left[x(t)\right] + ... + H_n\left[x(t)\right] \tag{1.14}$$

Où $H_n\left[x(t)\right] = \int_{-\infty}^{+\infty} ..... \int_{-\infty}^{+\infty} h_n(\tau_1, ..., \tau_n) x(t-\tau_1) ....x(t-\tau_n)\, d\tau_1 ....d\tau_n$ est appelé opérateur de Volterra du nième ordre. Une représentation graphique de la série de Volterra peut être donnée par le schéma synoptique de la figure 1.7.

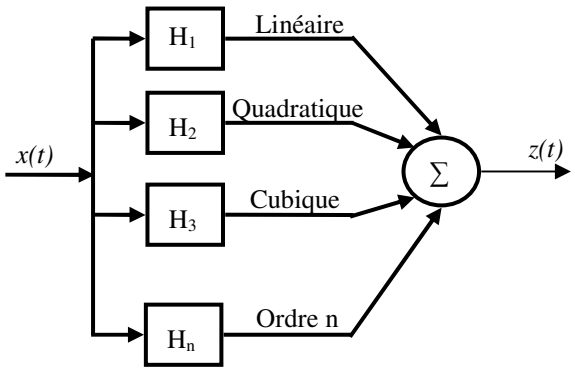


Figure 1.7 : *Représentation d'un système non linéaire par la série de Volterra*

L'intérêt d'un tel formalisme réside dans la possibilité de l'appliquer à tout système non linéaire. Cependant pour un système fortement non linéaire, le nombre de termes à mettre en jeu est important, et la représentation mathématique, trop lourde (intégrales multiples), font que ce formalisme devient impraticable pour des raisons de coût de calcul et de complexité d'extraction des noyaux. Ces difficultés ont limité l'application de la série de Volterra aux systèmes faiblement non linéaires de deux à trois noyaux.

Le développement en série tronqué de Volterra d'ordre trois de l'équation (1.14), pour $h_0=0$ est donné par :

$$\begin{aligned} z(n) &= \sum_{m_1=0}^{N} h_1(m_1)\, x(n-m_1) + \sum_{m_2=0}^{N}\sum_{m_1=0}^{N} h_2(m_1,m_2)\, x(n-m_1)\, x(n-m_2) + \\ &\sum_{m_1=0}^{N}\sum_{m_2=0}^{N}\sum_{m_3=0}^{N} h_3(m_1,m_2,m_3)\, x(n-m_1)\, x(n-m_2)\, x(n-m_3) \end{aligned} \tag{1.15}$$

Le premier modèle de Voltera simplifié d'ordre trois, pour les systèmes non linéaire sans mémoire était proposé par Falconer **[33]** et une version plus simplifiée de ce modèle est utilisée par Biglieri **[34]**, comme c'est représenté à la figure 1.8. Où une réduction significative dans le nombre de paramètre du modèle est obtenue. La sortie z est reliée à l'entrée x à travers la relation suivante :

$$z(n) = D_1 x(n) + D_2 x^2(n) + D_3 x^3(n) \tag{1.16}$$

Les coefficients D_1, D_2, et D_3 sont des scalaires qui contrôlent le degré de non linéarité. Ce modèle et sa version d'ordre deux sont largement exploités dans la littérature pour exprimer la distorsion non-linéaire introduite par les canaux de transmission.

Le signal de sortie du canal à filtre non linéaire est donné par la formule suivante :

$$r(k) = \hat{z}(k) + v(k) \tag{1.17}$$

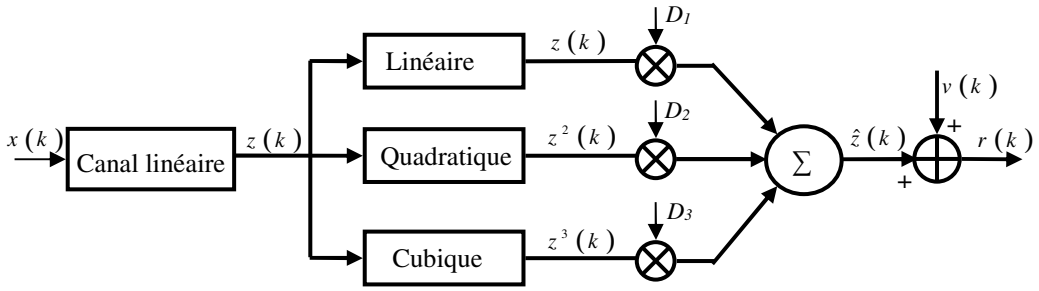


Figure 1.8 : *Modèle simplifié du canal non linéaire*

Les quatre modèles mathématiques décrits ci-dessus représentent la grande majorité des canaux physiques rencontrés dans la pratique. Dans notre travail, nous nous sommes focalisés sur le deuxième modèle et le quatrième modèle de canaux de transmission pour l'analyse et la conception de systèmes de communication. Le deuxième modèle se caractérise par l'action d'un filtre linéaire et l'ajout d'un bruit blanc gaussien. Le quatrième modèle se caractérise par l'action d'un filtre non linéaire et l'ajout d'un bruit blanc gaussien. A travers ces canaux, on récupère au niveau du récepteur plusieurs versions du signal émis avec des amplitudes, des phases différentes et des retards plus ou moins espacés. La présence de toutes ces perturbations ne permet pas de récupérer correctement le message émis sans aucun traitement préalable du signal au niveau du récepteur. D'où la nécessité d'introduire un bloc de traitement nommé égaliseur dans le but de réduire l'effet introduit par le canal de transmission.

1.4 TAUX D'ERREUR ET CAPACITÉ D'UN CANAL DE TRANSMISSION

1.4.1 Capacité d'un canal de transmission

La bande passante (en anglais bandwidth) d'une voie de transmission est l'intervalle de fréquence sur lequel le signal ne subit pas un affaiblissement supérieur à une certaine valeur (généralement 3 dB, car 3 décibels correspondent à un affaiblissement du signal de 50%). On définit la capacité du canal comme étant le nombre de symboles par unité de temps transmissible par le médium (symbole/s). En supposant un filtrage optimal (filtrage de Nyquist), la capacité d'un canal idéal (non bruité) de largeur B est $C=B$. Dans le cas du canal à bruit additif blanc gaussien (BABG, ou AWGN : Additive White Gaussian Noise), la capacité est d'après Shannon donnée comme suit **[35]** :

$$C = B_s \log_2\left(1 + \frac{P_s}{P_b}\right) \tag{1.18}$$

Où P_s *:* La puissance du signal reçu,
P_b *:* La puissance du bruit dans la bande,
C *:* Capacité (en bits par secondes),
B_s *:* Largeur de bande (en Hertz).

A partir de cette équation, deux approches permettant d'augmenter la capacité peuvent être énoncées. La première consiste à utiliser une bande étroite avec un rapport signal sur bruit P_s/P_b important, la seconde permet d'exploiter une bande large avec un rapport P_s/P_b faible. La figure 1.9 représente parfaitement ce cas c'est-à-dire plus la largeur de la bande de fréquence B_s augmente et plus la capacité augmente **[29]**.

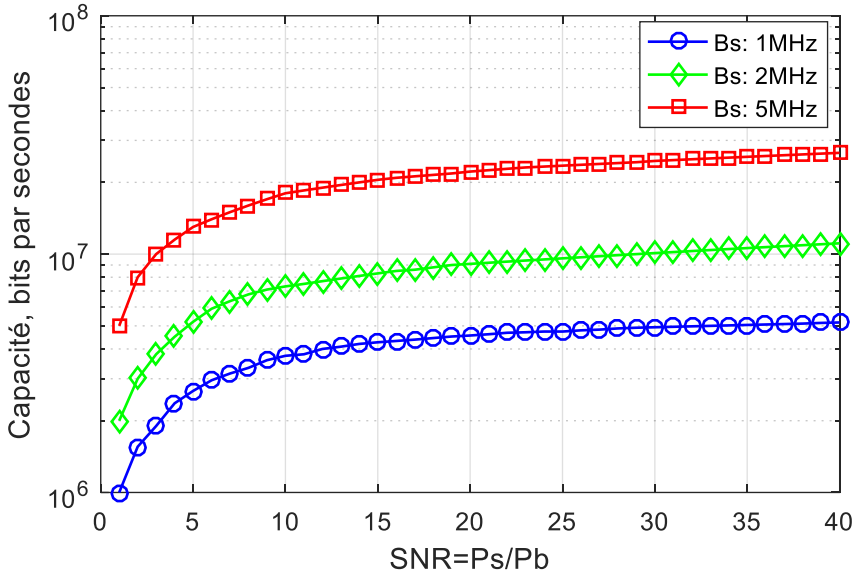


Figure 1.9 : *Capacité en fonction du SNR*

1.4.2 Taux d'erreur binaire (TEB)

Le rapport signal sur bruit P_s/P_b permet de quantifier le niveau de bruit par rapport au signal. Ce rapport est mesurable et il permet d'évaluer les performances d'un système dans un milieu bruité (AWGN). Toutefois, on préfère exprimer les performances d'un système en termes d'énergie du bit transmis (E_b) par rapport au bruit (N_o). Ce rapport est appelé rapport signal à bruit par bit noté E_b/N_o. Il s'agit du rapport entre l'énergie véhiculée par un bit E_b et la densité spectrale en puissance du bruit N_o. Cette grandeur est directement reliée au taux d'erreur binaire, et fixer une contrainte en termes de taux d'erreur binaire revient à fixer une contrainte sur le rapport E_b/N_o.
Décortiquons maintenant les deux termes de ce rapport. E_b représente l'énergie transportée par un bit. L'énergie par unité de temps s'appelle la puissance. Dans le cas d'un signal binaire, l'énergie par bit (en J/bit) est donc reliée à la puissance moyenne du signal P_s (en W) par le débit binaire D_b (en bit/s) comme le montre l'équation (1.19). On remarque que l'énergie par bit est inversement proportionnelle du débit binaire.

$$E_b = \frac{P_s}{D_b} \tag{1.19}$$

N_0 représente la densité spectrale de bruit, c'est-à-dire la puissance P_b transportée par le bruit sur une bande de fréquence de largeur B_s. Son unité est W/Hz, ce qui est équivalent à une énergie.

$$N_0 = \frac{P_b}{B_s} \tag{1.20}$$

En combinant les 2 équations précédentes, on peut relier le rapport signal à bruit P_s/P_b et le rapport E_b/N_o par l'équation suivante :

$$\frac{P_s}{P_b} = \frac{E_b}{N_0} \cdot \frac{D_b}{B_s} \Leftrightarrow \frac{E_b}{N_0} = \frac{P_s}{P_b} \cdot \frac{B_s}{D_b} \tag{1.21}$$

Le rapport E_b/N_o dépend non seulement des puissances du signal et du bruit, mais aussi des propriétés du signal telles que le débit binaire, la modulation (la bande passante nécessaire pour transporter un signal dépend du type de modulation employée). Le rapport E_b/N_o prend donc en compte l'effet des autres paramètres influents sur la robustesse d'une communication numérique. Ce rapport est donc un paramètre plus intéressant pour comparer des systèmes de communication différents.
Afin de quantifier la dégradation subie par un signal numérique ou de spécifier la qualité que doit atteindre une transmission numérique, on utilise la notion de taux d'erreur binaire (TEB ou BER : Bit Error Rate). Il s'agit du taux d'erreur mesuré à la réception d'une transmission numérique, et se calcule à l'aide de cette équation **[22]** :

$$TEB = \frac{\text{nombre de bits erronés}}{\text{nombre total de bits recus}} \tag{1.22}$$

Le taux d'erreur binaire (TEB) est similaire à la probabilité d'apparition d'erreur binaire, donc le TEB est relié au rapport signal à bruit par bit par l'équation (1.23) pour une modulation BPSK (Binary Phase Shift Key). Cette relation est particulièrement importante pour le dimensionnement d'un canal de transmission car il permet de définir une limite en termes de rapport signal à bruit à partir d'une contrainte donnée sur le taux d'erreur binaire maximale. Cette formule reste cependant théorique mais donne une bonne estimation du taux d'erreur binaire pour un rapport signal à bruit donné.

$$TEB = \frac{1}{2} erfc\left(\sqrt{\frac{E_b}{N_0}}\right) \tag{1.23}$$

1.5 PERTURBATIONS APPORTÉES PAR LES CANAUX DE TRANSMISSION

L'objectif d'un système de réception pour les transmissions numériques est de reconstituer le signal émis qui a été perturbé par le canal de transmission. Donc il est nécessaire de connaître la nature des perturbations qu'il a subies afin de dimensionner les fonctions élémentaires de l'émetteur et du récepteur.

1.5.1 Affaiblissement

Par définition, l'affaiblissement ou l'atténuation représente la perte de signal en énergie dissipée dans la ligne. L'affaiblissement se traduit par un signal de sortie plus faible que le signal d'entrée et caractérisée par le rapport de la puissance à la sortie du système P_s sur la puissance à son entrée P_e. On le calcule de la manière suivante **[22]** :

$$A = 10\log\left(\frac{P_s}{P_e}\right) \quad (dB) \tag{1.24}$$

$$A = 10\ln\left(\frac{P_s}{P_e}\right) \quad (Np) \tag{1.25}$$

Suivant la base choisie pour le logarithme, le gain ou l'affaiblissement sont exprimés en décibel (*dB*) ou en Néper (*Np*).
Le canal représente le milieu de transmission ou le lien physique entre l'émetteur et le récepteur, il est pratiquement constitué par l'un des supports suivants **[20]** :

- **Un câble bifilaire,** dont la bande passante est faible et qui est en générale réservé pour les transmissions à bas débit (inférieur à 2 Mbit/s sur le réseau téléphonique).
- **Un câble coaxial,** qui possède une bande passante plus importante que le câble bifilaire et qui permet de réaliser des transmissions avec un débit relativement élevé : plusieurs centaines de Mbit/s (jusqu'à 565 Mbit/s sur le réseau téléphonique). Le câble coaxial est notamment utilisé pour connecter les centraux téléphoniques entre lesquels transite un grand nombre de communications.
- **Une fibre optique,** qui apparaît aujourd'hui, grâce à sa bande passante très élevée (plusieurs dizaines de Gbits/s) et sa faible atténuation (0.2 dB/Km pour une longueur d'onde de 1550 nm), comme un support très intéressant. Les fibres optiques sont de plus en plus utilisées pour les réseaux terrestres à grande capacité, pour les câbles sous-marins, ainsi que pour les réseaux de distribution (c'est-à-dire sur les liaisons entre centraux téléphoniques et abonnés).
- **L'espace libre,** qui utilise la propagation d'une onde électromagnétique dans l'atmosphère. Ce milieu est généralement réservé aux transmissions par satellite ou par faisceaux hertzien ainsi qu'aux radiocommunications avec les mobiles. Les antennes constituent le dispositif de base pour transmettre ou recevoir un signal à travers ce type de canal. Sa bande passante s'étend sur un spectre très large (de plusieurs KHz à plusieurs GHz). Son avantage par rapport aux autres supports de communications précédemment cités est lié principalement au faible coût d'installation d'un réseau à grande échelle. Toutefois, il constitue le milieu le plus soumis aux perturbations extérieures.

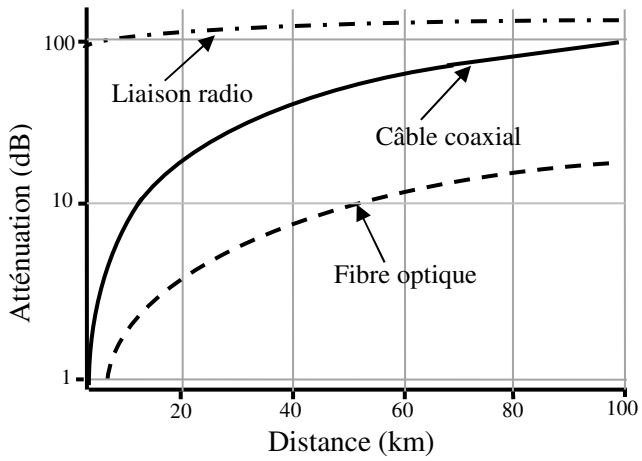


Figure 1.10 : *Comparaison de l'atténuation du signal pour différents supports de communication*

Les liaisons filaires, optiques et radio subissent des atténuations très différentes. La figure 1.10 présente une comparaison des atténuations en fonction de la distance séparant l'émetteur du récepteur pour ces 3 types de canaux de transmission. Le canal radio est celui qui présente l'atténuation la plus importante (Atténuation pour une fréquence égale à 900 MHz), alors que les fibres optiques constituent le support qui introduit le moins d'atténuation (0.2 dB/km à 1550 nm) **[22]**. Le câble coaxial introduit une atténuation égale à 1 dB/km à 900 MHz).
Considérons une propagation en espace libre. Lorsqu'une onde électromagnétique se propage et s'éloigne de la source, la puissance qu'elle transporte par unité de surface décroît avec la distance **[22]**.
L'atténuation en fonction de la distance d et de la fréquence f est donnée par :

$$Atténuation = \left(4\pi \frac{d}{\lambda}\right)^2 = \left(4\pi \frac{f \times d}{c}\right)^2 \tag{1.26}$$

Supposons qu'on est une liaison radiofréquence entre un émetteur E et un récepteur R. La puissance rayonnée par l'antenne de l'émetteur dépend de la puissance électrique P_e et du gain de l'antenne G_e. La puissance électrique reçue P_r dépend de la puissance transportée par l'onde électromagnétique et le gain de l'antenne réceptrice G_r. Le rapport entre la puissance électrique reçue et la puissance électrique émise est donnée par la formule de Friis **[22]** :

$$\frac{P_r}{P_e} = \frac{G_e G_r}{\left(4\pi \frac{d}{\lambda}\right)^2} = \frac{G_e G_r}{\left(4\pi \frac{f \times d}{\lambda}\right)^2} \tag{1.27}$$

1.5.2 Trajets multiples et interférences entre symboles (IES)

1.5.2.1 Trajets multiples

Le canal de propagation radioélectrique est un des moyens de communication les plus variables et les plus incontrôlables. Il est caractérisé par l'existence de trajets multiples.
Les trajets multiples sont également à l'origine de plusieurs problèmes dont les principaux sont :

- **L'obstruction :** apparaît quand un trajet radio est obstrué par un ou plusieurs objets (obstacles naturels ou construits par l'homme). L'onde résultante subit une perte de puissance correspondante au mécanisme de propagation impliqué (qui peut être la réflexion, la diffraction ou la diffusion). La réflexion, elle se produit lorsqu'une onde électromagnétique rencontre des surfaces lisses de très grandes dimensions par rapport à sa longueur d'onde, comme par exemple la surface de la terre, les bâtiments et les murs. La diffraction, elle se produit lorsqu'un obstacle épais et de grande dimension par rapport à sa longueur d'onde obstrue l'onde électromagnétique entre l'émetteur et le récepteur. Dans ce cas, des ondes secondaires sont générées et se propagent derrière l'obstacle. La diffusion, elle se produit lorsque l'onde rencontre un obstacle dont l'épaisseur est de l'ordre de sa longueur d'onde, comme par exemple les lampadaires et les feux de circulation. Dans ce cas, l'énergie est dispersée dans toutes les directions.

- **La dispersion des retards (Delay spread) :** Les trajets réfléchis sont généralement plus longs que le trajet direct c'est-à-dire qu'ils atteignent l'émetteur plus tard que le trajet direct. Les signaux provenant de la même émission arrivent donc au niveau de l'émetteur avec des retards différents.
- **Evanouissements (ou fading) de Rayleigh :** Après réflexion sur un obstacle, l'onde radio peut être altérée en phase et en amplitude. Le phénomène d'évanouissements ou de fading résulte des variations temporelles des phases. Celles-ci peuvent résulter de signaux multiples s'ajoutant de façon destructive au niveau du récepteur. Dans ce cas, le signal reçu résultant sera très faible ou pratiquement nul. Les signaux multiples reçus peuvent également s'additionner de façon constructive et le signal reçu résultant est alors plus puissant que le signal du seul trajet direct.
- **Décalage en fréquence (Doppler shift) :** L'effet Doppler est un phénomène dû au déplacement du récepteur par rapport à l'émetteur. Il entraîne une variation dans la fréquence du signal reçu appelée décalage Doppler. Ce décalage en fréquence dépend essentiellement de deux facteurs : la direction de déplacement et la vitesse du récepteur par rapport à l'émetteur. Chaque trajet possède un décalage Doppler fréquentiel de la forme **[36]** :

$$f_d = f_m \cos\theta \tag{1.28}$$

$$\text{Avec :} \quad f_m = \frac{v}{\lambda} \tag{1.29}$$

Où θ est l'angle entre la direction du récepteur et la direction du trajet considéré, v représente la vitesse du récepteur et λ la longueur d'onde de la porteuse. Dans les environnements multi-trajets, chaque trajet du signal subit un décalage Doppler différent caractérisé par l'angle θ. Par conséquent, le signal reçu est formé de composantes possédant des décalages fréquentiels différents compris entre $f_c - f_m$ et $f_c + f_m$, dont la combinaison crée un élargissement du spectre. Les deux valeurs $+f_m$ ou $-f_m$ sont obtenues lorsque l'onde se propage dans la direction du récepteur ou dans la direction opposée, f_c est la fréquence porteuse. L'étalement Doppler est défini donc comme étant la largeur du spectre de puissance Doppler, obtenu par transformée de Fourier de la fonction d'auto-corrélation de la réponse impulsionnelle du canal. Les effets de l'étalement Doppler sont négligeables tant que la largeur de bande du signal transmis en bande de base est beaucoup plus grande que la largeur du spectre de puissance Doppler.

1.5.2.2 Modèle du canal multi-trajet

En général, un canal multi-trajets aura *L* trajets différents et pour chacun d'eux la réponse impulsionnelle complexe en bande de base est de la forme **[16]**:

$$\alpha_l(t)\,\delta\left(\tau - \tau_l(t)\right) \tag{1.30}$$

Où $\alpha_l(t)$ est l'amplitude associée au retard τ_l, $\tau_l = l\Delta_t$ est le retard de propagation et $\delta(.)$ est une impulsion de Dirac.

La réponse impulsionnelle du canal complexe sera modélisée comme la somme de toutes les réponses des L trajets différents (Figure 1.11).

$$h(\tau,t) = \sum_{l=0}^{L-1} \alpha_l(t)\,\delta\left(\tau - \tau_l(t)\right) \tag{1.31}$$

Et a pour transformée de Fourier :

$$H(f,t) = \sum_{l=0}^{L-1} \alpha_l(t) e^{-j2\pi f \tau_l(t)} \tag{1.32}$$

Avec: $\alpha_l(t) = \rho_l e^{-j2\pi f_0 \tau_l(t)}$, f_0: la fréquence porteuse, ρ_l : amplitude associée au trajet *l*.

$\tau_l(t) = \left(\frac{v_m}{c}\right)t$, avec $f_0.\left(\frac{v_m}{c}\right) = f_d$: la fréquence Doppler.

$\gamma_l(t) = -2\pi f_0 \tau_l(t)$: est le déplacement de la phase introduit par le trajet *l*.

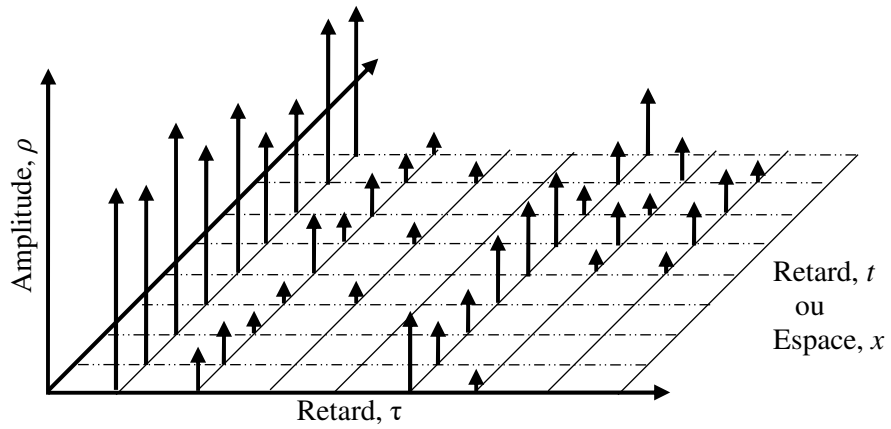


Figure 1.11 : *La réponse impulsionnelle temporelle du canal.*

Cette représentation concerne généralement les variations à petites échelles du signal. Dans ce cas les coefficients $\alpha_l(t)$ représentent les variations rapides du signal et suivent une distribution de Rayleigh ou de Rice selon qu'il y a ou non un trajet direct entre l'émetteur et le récepteur. Tous les paramètres de la réponse impulsionnelle d'un canal multi-trajets sont donc des fonctions aléatoires du temps indépendants dû à un environnement variable dans le temps.

- **Canal de Rayleigh**

  La distribution de Rayleigh est fréquemment utilisée dans le modèle multi-trajets avec l'absence d'un trajet direct entre l'émetteur et le récepteur, sa densité de probabilité est donnée par **[37]** :

$$p(\alpha_l) = \frac{\alpha_l}{\sigma_{\alpha_l}^2} exp^{\left(-\frac{\alpha_l^2}{2\sigma_{\alpha_l}^2}\right)},\ \alpha \geq 0 \tag{1.33}$$

La phase θ de $\alpha_l(t)$ est une variable aléatoire uniformément distribuée sur l'intervalle $[-\pi,\pi]$.

$$p(\theta_{\alpha_l}) = \frac{1}{2\pi}, -\pi \le \theta \le \pi \tag{1.34}$$

$\sigma_{\alpha_l}, \sigma^2_{\alpha_l}$ (Variance) sont respectivement la tension et la puissance moyenne temporelle du signal reçu avant détection.

- **Canal de Rice**

  La distribution de Rice est utilisée dans le modèle multi-trajets avec la présence d'un du trajet direct, sa densité de probabilité est donnée par **[37]** :

$$p(\alpha_l) = \frac{\alpha_l}{\sigma^2_{\alpha_l}} exp^{\left(-\frac{\alpha_l^2+A^2}{2\sigma^2_{\alpha_l}}\right)} J_0\left(\frac{A\alpha_l}{\sigma^2_{\alpha_l}}\right), \ A \ge 0, \alpha \ge 0 \tag{1.35}$$

  Avec *A:* est la puissance du signal reçu ou du trajet direct, $J_0(.)$ est la fonction de Bessel modifiée d'ordre 0. Nous constatons que si *A*=0 nous aurons un canal de Rayleigh.

1.5.2.3 Interférences entre symboles (IES)

Dans un système numérique, particulièrement s'il fonctionne à un débit binaire élevé, la dispersion des retards (delay spread) fait que chaque symbole (ou unité d'information) chevauche le précédent et les subséquents, d'où le phénomène d'interférence entre symboles (IES ou ISI : Inter-Symbol Interference). Contrairement au bruit, l''interférence entre symboles possède une structure particulière qui permet de la combattre, c'est le rôle de l'égalisation. Comme le montre la figure (1.12). Lorsque le temps symbole est inférieur à l'étalement temporel du canal, la valeur du symbole reçu à l'instant *t* est perturbée par les symboles reçus précédemment. Ce phénomène se produit si l'amplitude de l'impulsion soumise à échantillonnage en réception dépend, a l'instant de décision, de symboles voisins. Le symbole reçu peut alors être confondu avec un autre et introduire des erreurs d'interprétation par le récepteur. L'interférence entre symbole est la principale source d'erreur binaire dans les communications numériques.

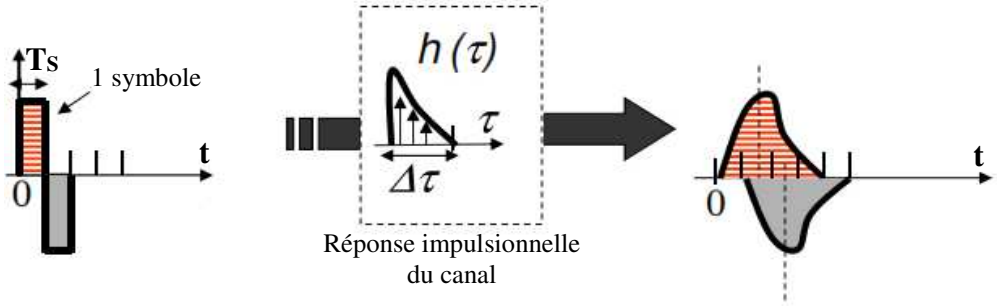


Figure 1.12 : *Etalement d'un signal numérique après transmission*

Les trajets multiples sont à l'origine de la dispersion temporelle et l'effet Doppler provoque la dispersion fréquentielle. On introduit deux paramètres relatifs à ces dispersions **[29]** :

- **La bande de cohérence B_c :** Il s'agit de l'écart fréquentiel minimal sur lequel les caractéristiques du canal sont corrélées. Ainsi, deux sinusoïdes dont l'écart fréquentiel est supérieur à B_c seront différemment affectées par le canal. Cette grandeur est environ l'inverse de l'étalement temporel du canal.
- **Le temps de cohérence T_c :** C'est la durée sur laquelle les caractéristiques du canal de transmission demeurent quasi-constantes. Cette grandeur est environ l'inverse de l'étalement fréquentiel du canal.

Ces paramètres sont utilisés pour classer les canaux. On note B_s la bande occupée par le signal à transmettre et T_s la durée d'un symbole (voir figure 1.13).

- **Si $B_s << B_c$:** toutes les composantes fréquentielles du signal subissent la même atténuation et le canal est dit **non sélectif en fréquence** (quasi pas d'IES).
- **Si $B_s >> B_c$:** les différentes composantes fréquentielles du signal subissent des atténuations différentes et le canal est dit **sélectif en fréquence** (Présence IES).
- **Si $T_s << T_c$:** les caractéristiques du canal ne varient pas pendant la durée de transmission du symbole et le canal est dit **non sélectif en temps**.
- **Si $T_s >> T_c$:** les caractéristiques du canal varient pendant la durée de transmission du symbole et le canal est dit **sélectif en temps**.

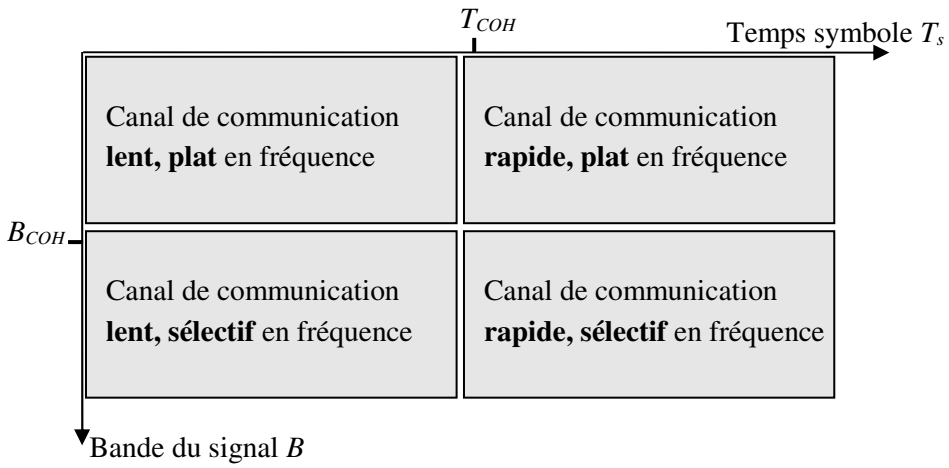


Figure 1.13 : *Classification des canaux de communication*

Le contrôle au niveau temporel du degré d'IES s'effectue de façon très simple sur un oscilloscope par le diagramme de l'œil (figure 1.14). Le diagramme de l'œil est un outil graphique permettant de visualiser la présence d'IES affectant une communication et de qualifier la qualité du signal numérique reçu. En l'absence d'IES, l'œil est complètement ouvert à l' instant de décision : tous les trajets passent par deux points seulement en binaire (par M points en M-air).

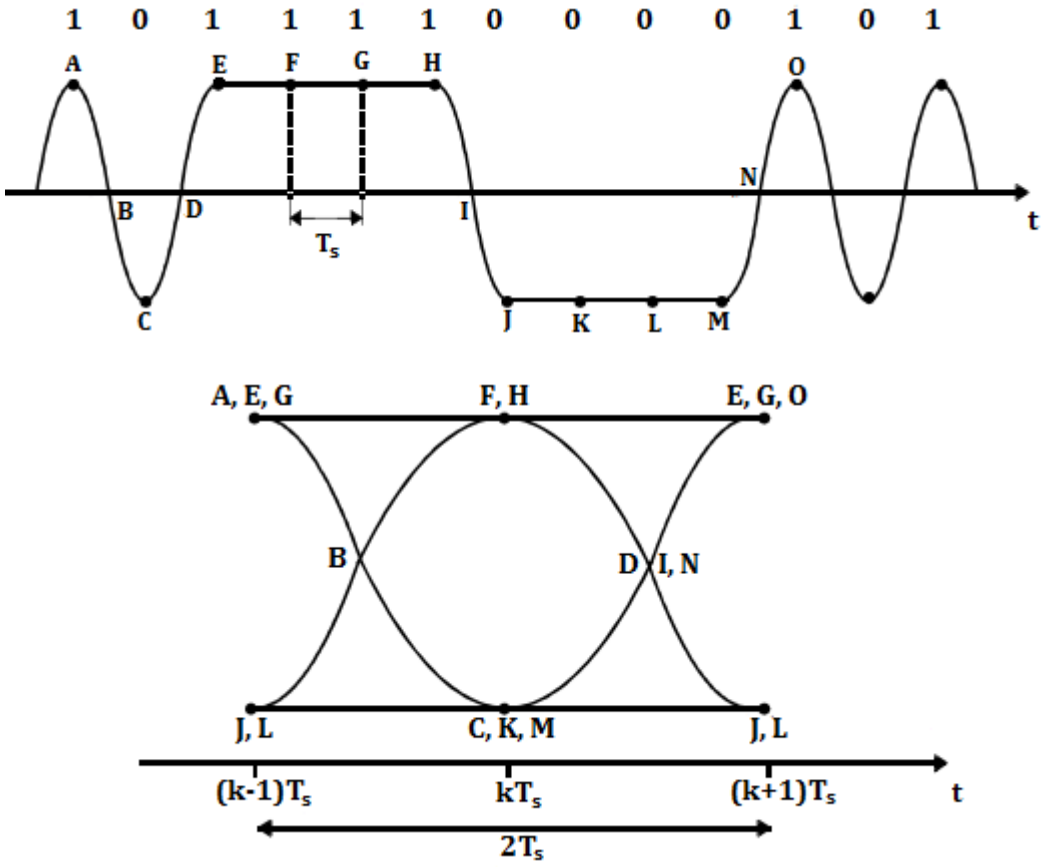


Figure 1.14 : *Diagramme de l'œil*

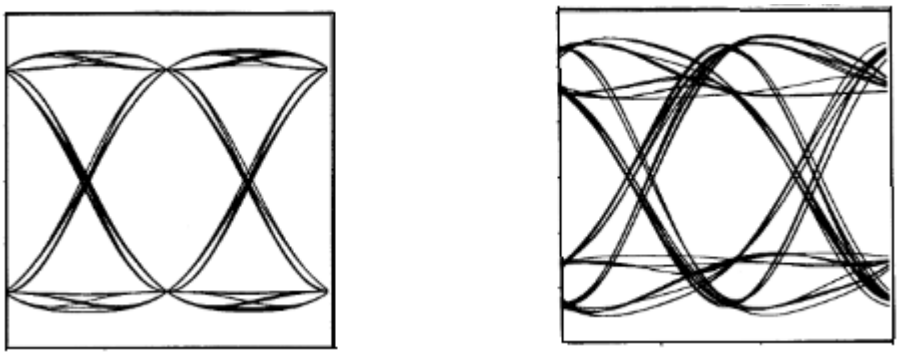


Figure 1.15: *Diagramme de l'œil pour une transmission binaire*
(a) sans IES, **(b)** avec IES

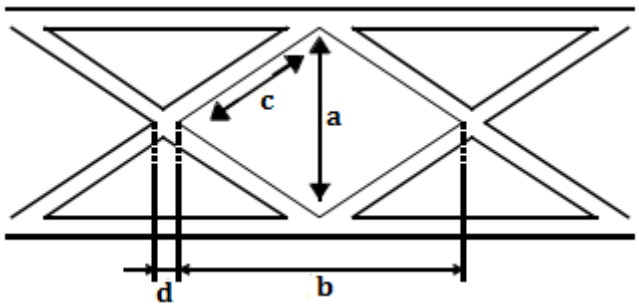


Figure 1.16 : *Caractéristiques du diagramme de l'œil*

La figure 1.15 présente un exemple de diagramme de l'œil (avec et sans interférence entre symboles) pour une transmission binaire. Le diagramme de l'œil met en évidence une ouverture verticale **a** (immunité au bruit), une ouverture horizontale **b** (immunité au déphasage de l'horloge), une pente **c** (immunité à la gigue d'horloge) et une fluctuation **d** (amplitude de la gigue du point de passage par zéro) (Figure 1.16). L'ouverture verticale ou la hauteur de l'œil donne la marge en termes de bruit sur les niveaux. Plus l'ouverture est faible, plus la présence de bruit pourra causer une erreur de décision sur le niveau. L'ouverture horizontale ou la largeur de l'œil donne la marge en termes d'écart temporel entre l'instant d'échantillonnage idéal et tout autre temps d'échantillonnage. L'instant d'échantillonnage idéal, c'est-à-dire le moment où la probabilité d'erreur est minimisée, se situe à l'instant où l'œil présente sa plus grande ouverture. La pente de fermeture ou d'ouverture donne la sensibilité aux erreurs temporelles.

1.5.2.4 Annulation de l'IES : condition de Nyquist

Pour améliorer la fiabilité d'une communication numérique, il convient de minimiser le risque d'apparition d'IES. Il est possible de savoir si un canal pourra présenter un risque important d'apparition d'interférence entre symboles en étudiant sa réponse impulsionnelle discrète. Soit le système de transmission décrit à la figure 1.17, présentant un filtre émetteur, un support de transmission puis un filtre récepteur **[21]**.

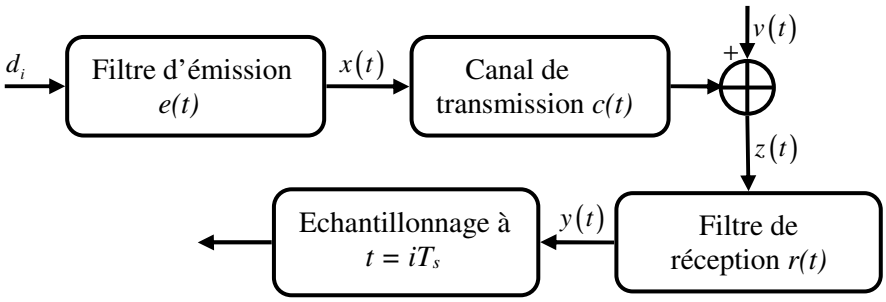


Figure 1.17 : *Modèle d'un système de transmission*

D'après la figure 1.17, la réponse impulsionnelle globale du système de transmission (filtre d'émission, de canal, de réception) s'écrit comme la convolution des trois réponses mises en jeu :

$$p(t) = e(t) * c(t) * r(t) \tag{1.36}$$

Dans le domaine fréquentiel, cette relation devient :

$$P(f) = E(f)\mathrm{C}(f)\mathrm{R}(f) \tag{1.37}$$

- **Condition de Nyquist dans le domaine temporel**

Supposons qu'un émetteur transforme la séquence binaire d_i et produit le signal transmis :

$$x(t) = \sum_i d_i e(t - iT_s) \tag{1.38}$$

Où e*(t)* représente la réponse impulsionnelle du filtre émetteur et T_s la période des symboles binaires.

Le signal est ensuite transmis à travers le canal de transmission, caractérisé par une réponse impulsionnelle *c(t)*. De plus, le canal introduit un bruit additif au signal à l'entrée du récepteur. Le signal en entrée du récepteur $z(t)$ peut s'écrire sous la forme :

$$z(t) = x(t) * c(t) + v(t) \tag{1.39}$$

Ce signal passe ensuite à travers un filtre de réception de réponse impulsionnelle *r(t)*. Le signal en sortie du filtre de réception peut s'écrire :

$$y(t) = r(t) * z(t) \tag{1.40}$$

Finalement, la sortie *y(t)* est échantillonnée de façon synchrone avec l'émetteur tous les $t_i = i.T_s$, ce qui donne **[23]**:

$$y(t_i) = p(0)\, d_i + \sum_{k \neq i}^{\infty} d_k\, p\big((i-k)T_s\big) + w(t_i) \tag{1.41}$$

Où *w(t)* représente le bruit produit à la sortie du filtre de réception par le bruit additif d'entrée du récepteur *v(t)*, et *p(t)* la réponse impulsionnelle du système composé du canal et des filtres d'émission et de réception.
Finalement, un dispositif de décision reconstruit les données symboles de l'entrée à partir de cette sortie échantillonnée. En particulier il doit décider du symbole émis à l'instant t_i, c'est-à-dire fournir $\hat{d}_i$.
Dans l'expression (1.41) :

- Le premier terme $p(0)d_i$ représente la contribution du *i*ème symbole transmis, c'est-à-dire celui qu'on cherche à recevoir sans erreurs.
- Le second terme représente l'effet résiduel de tous les autres symboles transmis sur le décodage du *i*ème symbole. Cet effet résiduel dû aux impulsions arrivant avant et après l'instant t_i est appelé interférence entre symboles.
- Le dernier terme $w(t_i)$ représente le bruit à l'instant t_i.

En l'absence de bruit et d'interférence entre symboles, on ne récupèrerait que le premier terme et aucune erreur d'interprétation ne serait possible.
A partir de l'équation (1.41), l'interférence entre symboles est nulle si l'on vérifie :

$$\sum_{k \neq i}^{\infty} d_k\, p\big((i-k)T_s\big) = 0, \quad \forall d_k \tag{1.42}$$

L'équation (1.42) indique que tous les symboles doivent s'annuler aux instants d'échantillonnage des autres symboles. Une condition nécessaire pour ne pas avoir d'IES est que l'impulsion de base $p(t) = e(t) * c(t) * r(t)$ possède la propriété :

$$p(iT_s) = p(0)\,\delta[i] \tag{1.43}$$

Le filtre *p(t)*, qui représente le canal en entier (filtre d'émission, de canal, de réception) est dit canal de Nyquist s'il vérifie cette condition.

- **Condition de Nyquist dans le domaine spectral**

Si la condition de Nyquist dans le domaine temporel est vérifiée, l'échantillonnage avec une période T_s de l'impulsion de base *p(t)* conduit alors à un seul Dirac en zéro :

$$\sum_i p(t)\delta(t-iT_s) = p(0)\delta(t) \tag{1.44}$$

En prenant la transformée de Fourier (TF) des deux membres, on en déduit la condition dans le domaine spectral **[38]** :

$$\sum_i P\left(f-\frac{i}{T_s}\right) = T_s p(0) \tag{1.45}$$

Compte-tenu des propriétés de la TF, il est impossible d'avoir un support borné à la fois dans les domaines temporel et fréquentiel. En pratique, le choix est imposé : la transmission doit s'effectuer dans un canal à bande passante limitée [-*B*, *B*]. On suppose la TF de l'impulsion de base a un support fréquentiel borné, avec :

$$P(f) = 0 \text{ pour } |f| \geq \frac{1}{T_s} \tag{1.46}$$

On vérifie alors facilement que la condition dans le domaine spectral conduit à la condition ci- dessous, de laquelle découle le critère spectral de Nyquist :

$$P\left(\frac{1}{2T_s}+\Delta f\right)+P\left(\frac{1}{2T_s}-\Delta f\right) = P(0) \tag{1.47}$$

L'IES est nulle si le point $\left(\frac{1}{2T_s}, \frac{P(0)}{2}\right)$ de la réponse en fréquence globale $P(f)$ est un centre de symétrie (voir figure 1.18)

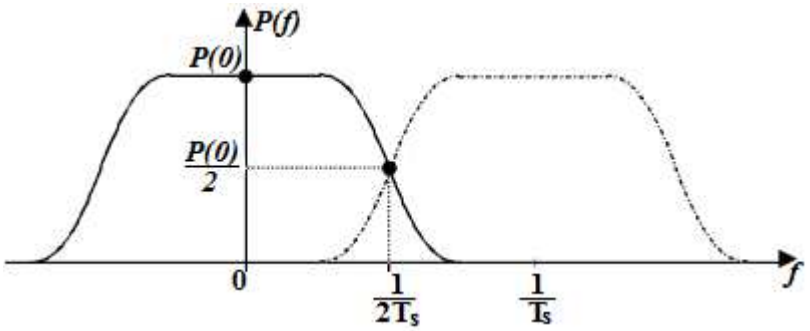


Figure 1.18 : *Démonstration du critère spectral de Nyquist*

Lorsque les conditions de Nyquist sont vérifiées, l'ensemble du système constitue un canal de Nyquist. Notons que la transmission sans IES est impossible si la bande passante *B* du canal est inférieure à la limite appelée la fréquence de Nyquist :

$$B < \frac{1}{2T_s} = \frac{D_s}{2} \tag{1.48}$$

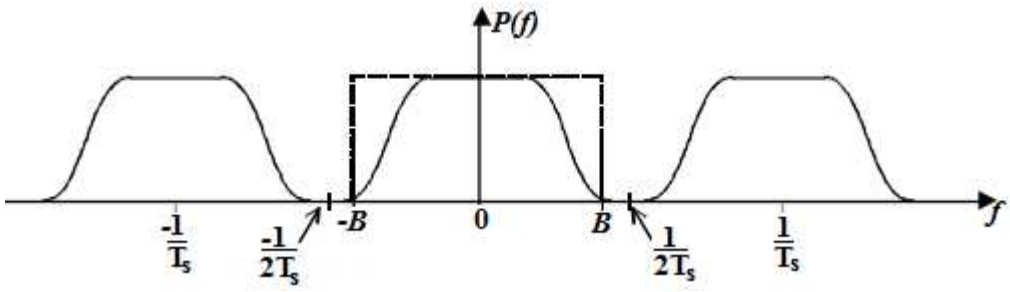


Figure 1.19 : *Bande passante inférieure à 1/2T_s*

En présence du bruit, le filtre de réception, doit corriger, éventuellement de façon adaptative, la distorsion linéaire responsable de l'IES introduite par le canal. Le canal est dit égalisé lorsque la réponse globale vérifie le critère de Nyquist. En pratique on y parvient à l'aide d'un filtre supplémentaire appelé égaliseur placé derrière le filtre d'entrée du récepteur, après échantillonnage (réalisation sous forme numérique). Plus d'information sur les techniques d'égalisation va être présentée dans les chapitres suivants **[38]**.

1.6 FILTRAGE NUMERIQUE

Les filtres numériques sont utilisés pour deux raisons principales, la première est la séparation des signaux qui ont été combinés, la séparation est nécessaire quand un signal a été affecté par des interférences, de bruit, ou d'autres signaux, et la second raison est la restauration des signaux qui ont été déformés. Les filtres analogiques peuvent être utilisés pour ces mêmes tâches, mais, les filtres numériques aboutissent des résultats bien supérieurs. Selon l'implémentation, deux types sont distingués, les filtres récursifs, dits filtres à Réponse Impulsionnelle Infinie (RII), et les filtres non récursifs, dits filtres à Réponse Impulsionnelle Finie (RIF) **[39]**.

1.6.1 Filtre à Réponse Impulsionnelle Finie (RIF)

La sortie d'un filtre numérique à réponse impulsionnelle finie (figure 1.20), peut s'exprimer comme une combinaison linéaire des échantillons présents à son entrée. Les coefficients de la combinaison linéaire constituent la réponse impulsionnelle du filtre.
Le filtre de réponse *H(z)* défini par (1.49) est un filtre de type RIF. Un filtre RIF est stable et causal et donc physiquement réalisable. Pour cette raison les filtres RIF sont très souvent utilisés dans les systèmes d'égalisation **[40]**.

$$H(z)=\frac{Y(z)}{X(z)}=\sum_{i=0}^{N-1}h[i]z^{-i} \tag{1.49}$$

La sortie du filtre est de la forme :

$$y(k)=\sum_{i=0}^{N-1}h[i]x(k-i) \tag{1.50}$$

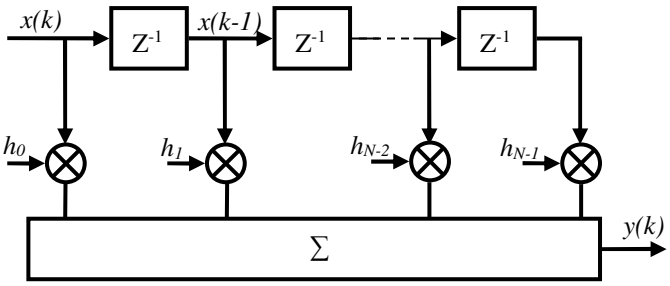


Figure 1.20 : *Structure d'un filtre à réponse impulsionnelle finie (RIF)*

1.6.2 Filtre à Réponse Impulsionnelle Infinie (RII)

Les filtres numériques à réponse impulsionnelle infinie RII (figure 1.21), conservent une trace des échantillons qui leurs ont été appliqués pendant une durée infinie, ils sont donc à mémoire infinie. Une telle mémoire est réalisée en utilisant une boucle de réaction de la sortie sur l'entrée, d'où la dénomination courante de filtre récursif **[40]**. Un filtre RII est décrit par l'équation suivante :

$$y(k)=\sum_{n=0}^{N}a(n)x(k-n)-\sum_{m=1}^{M}b(m)y(k-m) \tag{1.51}$$

Où : $x(k)$: Valeurs successive du signal d'entrée.

a_n, b_m : Coefficients de la fonction de transfert du filtre.

$y(k)$: Valeurs successives du signal de sortie.

N, M : Ordres du numérateur et du dénominateur de $H(z)$

Chaque échantillon de sortie est égal à une combinaison linéaire des échantillons présents à l'entrée du filtre et des échantillons précédemment déterminés en sortie du filtre. La fonction de transfert d'un filtre RII se présente sous la forme suivante **[35].**

$$H(z)=\frac{\sum_{n=0}^{N}a(n)\,z^{-n}}{1+\sum_{m=1}^{M}b(m)\,z^{-m}} \tag{1.52}$$

Les racines du numérateur et du dénominateur sont respectivement appelées les zéros et les pôles de la fonction de transfert, ainsi la fonction de transfert peut s'écrite **[21, 35]**

$$H(z)=A_0\frac{\prod_{n=0}^{N}(1-Z_nZ^{-1})}{\prod_{m=1}^{M}(1-P_mZ^{-1})} \tag{1.53}$$

Z_n et P_m dénotent les zéros et les pôles respectivement, A_0 représente le facteur d'échelle qui définit le gain.

Pour qu'un filtre à réponse impulsionnelle infinie soit réalisable, il est nécessaire qu'il soit causal et stable. Un filtre causal est stable si les pôles de sa fonction de transfert en Z sont à l'intérieur du cercle unité. On notera qu'un filtre possédant tous ses pôles et ses zéros à l'intérieur du cercle unité est appelé filtre à phase minimale, tandis qu'un filtre dont tous les pôles sont à l'intérieur du cercle unité et des zéros à l'extérieur de ce cercle est dit à phase non minimale.

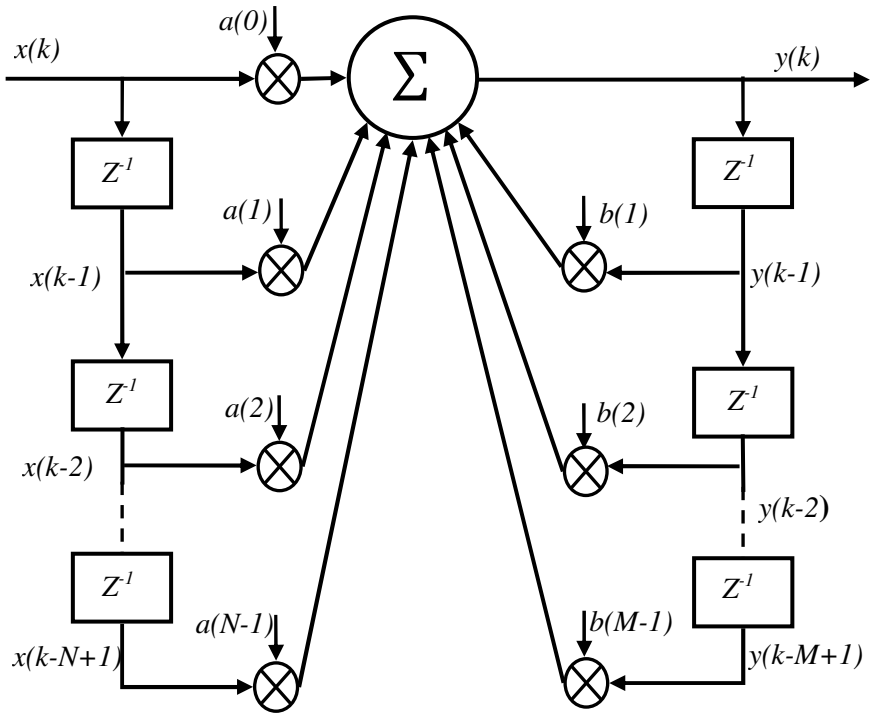


Figure 1.21 : *Structure d'un filtre à réponse impulsionnelle infinie (RII)*

1.6.3 Comparaison entre les filtres RIF et RII

- Un filtre RIF peut atteindre une phase linéaire exacte, il n'introduit pas des distorsions sur la phase du signal. La réponse à phase linéaire est importante dans les applications telles que la transmission des données, et le traitement d'image. Par contre un filtre RII possède généralement une réponse à phase non linéaire en particulier sur les limites de bande.
- La stabilité des filtres RIF est garantie alors qu'un tel garant n'est pas disponible pour les filtres RII.
- Les filtres RII sont plus souhaités lorsqu'un seuil de coupure est exigé. Un tel cas nécessite un filtre RIF d'ordre très élevé, se qui implique un grand retard. Or, un filtre d'ordre supérieur possède plus de coefficients et par conséquent plus de mémoire et plus de coût en termes de calcul.
- Les filtres RII sont plus sensibles aux erreurs d'arrondi et des erreurs de quantification que les filtres RIF.

En conclusion, si un seuil de coupure avec un haut débit (retard faible) sont demandés, les filtres RII sont les plus souhaitables. D'autre part, si la phase linéaire exacte est importante, les filtres RIF sont les plus conseillés. Par conséquent les filtres RIF sont le choix le plus couramment utilisé si le nombre de coefficients n'est pas trop grand en raison de leurs propriétés de calcul supérieures.

1.6.4 Filtrage adaptatif

Dans l'étude de l'égalisation, il semble pertinent d'avoir une idée au sens large sur le « filtrage adaptatif ». Le terme « adaptatif » peut être compris en considérant un système qui tente de lui ajuster pour répondre à certains phénomènes qui avaient lieu autour de lui, en d'autres mots, le système tente d'ajuster leurs paramètres dans le but de répondre à certains objectifs bien définis, qui dépendent de l'état du système ainsi que son environnement. De plus, il est nécessaire d'avoir un ensemble des étapes ou une certaine procédure avec laquelle l'adaptation est accomplie. Le système qui réalise et subit la procédure d'adaptation est appelé « filtre ».

Un filtre adaptatif est constitué de deux parties (Figure 1.22) :

- Un filtre numérique à coefficients ajustables de nature FIR ou IIR (Wiener, Kalman …etc.),
- Un algorithme de modification de ces coefficients ajustables basé sur un critère d'optimisation **[41]**.

Evidemment, selon le temps nécessaire pour atteindre l'objectif final de l'adaptation, qui s'appelle le temps de convergence, la complexité, et les ressources disponibles pour accomplir l'adaptation, nous pouvons avoir une variété des algorithmes d'adaptation et des structures des filtres.

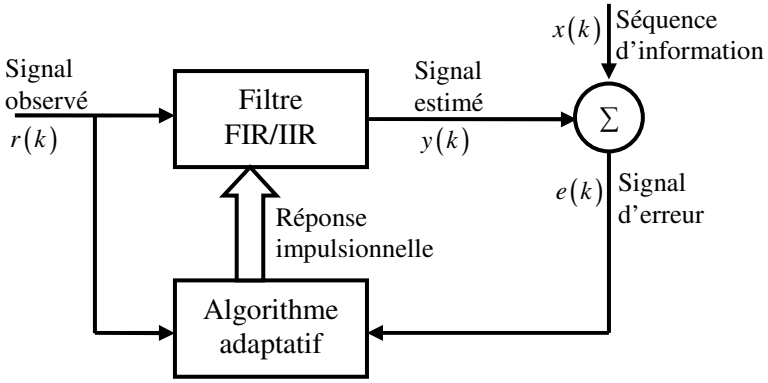


Figure 1.22 : *Schéma d'un système de filtrage adaptatif*

Les filtres adaptatifs, par leur nature même, sont des systèmes auto-concepteurs qui peuvent s'adapter à différents environnements. Par conséquent, ils trouvent des applications dans des domaines aussi divers que le contrôle, les communications, le traitement du signal radar et sonar, l'annulation des interférences, le contrôle actif du bruit, le génie biomédical, etc. La caractéristique commune de ces applications qui les regroupent sous la même formule de base de filtrage adaptatif est qu'elles impliquent toutes un processus de filtrage d'un signal d'entrée pour atteindre une réponse souhaitée. les paramètres du filtre sont mis à jour en faisant un ensemble de mesures des signaux sous-jacents et l'application de cet ensemble à l'algorithme de filtrage adaptatif de telle sorte que l'écart entre la sortie du filtre et la réponse désirée est réduit au minimum soit dans les algorithmes statistiques ou déterministes.

1.7 CONCLUSION

Dans ce chapitre, nous avons présenté les principales notions relatives aux systèmes de communication numérique. Nous avons présenté quelques généralités sur la chaîne de communication numérique. Les différents modèles mathématiques des canaux de transmission les plus rencontrés en pratique, sont également évoqués, avec les distorsions introduites par ces canaux, notamment l'interférence entre symboles qui a été bien détaillé en considérant qu'il constitue un problème dont son élimination fait l'objectif principal de l'égalisation. On a présenté aussi un aperçu sur les filtres numériques, leur structures et leur équation, le filtrage adaptatif est une notion très importante qu'on a évoqué aussi, tous les éléments cités précédemment, constitue les éléments de base qui vont nous aider à avoir une idée précise des égaliseurs adaptatifs, l'objet des chapitres suivants.

CHAPITRE 2

ALGORITHMES D'OPTIMISATION : ETAT DE L'ART

2.1 INTRODUCTION

La résolution de tels problèmes a fait l'objet de nombreux travaux en utilisant diverses méthodes d'optimisation. De nos jours l'optimisation est devenue un domaine indispensable pour résoudre plusieurs problèmes que se soit dans l'industrie ou d'autres secteurs. Les algorithmes d'optimisation utilisés dans le filtrage adaptatif présentent plusieurs difficultés liées aux besoins de l'utilisateur (recherche d'une solution globale, fiabilité et précision de la solution, temps de calculs disponible,…), aux caractéristiques du problème traité (non linéarité, interférence, bruit,…) et aux temps de calculs importants. Nous aborderons dans ce chapitre les définitions générales des méthodes d'optimisation qui se divisent en deux grandes classes : les méthodes déterministes et les méthodes non déterministes. Les caractéristiques principales de chaque classe, leurs points forts et leurs points faibles sont montrés.

2.2 CARACTÉRISTIQUES

2.2.1 Formulation des problèmes d'optimisation

La formulation des problèmes d'optimisation reste très ambigüe à cause de la diversité des vocabulaires et des confusions éventuelles que cela pourrait engendrer. Nous avons convenu d'adopter le vocabulaire qui suit **[42]** :

- Un problème d'optimisation mono-objectif est défini par un ensemble de variables, une fonction objectif et un ensemble de contraintes.
- Un problème d'optimisation multi-objectif est défini par un ensemble de variables, un ensemble de fonctions objectif et un ensemble de contraintes.
- L'espace d'état, appelé aussi domaine de recherche, est l'ensemble des domaines de définition des différentes variables du problème.
- Les variables du problème dite aussi variable de conception ou de décision peuvent être de nature diverse (réelle, entière, booléenne. etc.) et exprimer des données qualitatives ou quantitatives.
- La fonction objectif ou encore (fonction de coût) définit le but à atteindre, on cherche à minimiser ou à maximiser celle-ci.
- Une fonction multimodale présente plusieurs minima (locaux et globaux). Tandis qu'une fonction unimodale n'a qu'un minimum, le minimum global.

Les méthodes d'optimisation recherchent un point ou un ensemble de points dans l'espace de recherche qui satisfont l'ensemble des contraintes, et qui maximisent ou minimisent la fonction objectif.

2.2.2 Optimum local et optimum global

Lorsque l'on veut résoudre un problème d'optimisation, on recherche la meilleure solution possible à ce problème, c'est-à-dire l'optimum global. Cependant, il peut exister des solutions intermédiaires, qui sont également des optimums, mais uniquement pour un sous-espace restreint de l'espace de recherche : on parle alors d'optimums locaux. Cette notion est illustrée dans la figure 2.1 **[43]**.

Soit un problème d'optimisation, avec C l'ensemble des solutions admissibles du problème :

- Si l'on peut prouver que $\forall x \in C,\ f(x_g) < f(x)$, alors on dira que x_g est l'optimum (minimum) global du problème;
- S'il existe un ensemble $V \subset C,$ contenant $x_l,$ tel que $\forall x \in V,\ f(x_l) < f(x)$, avec $x \neq x_l$, alors on dira que x_l est un optimum (minimum) local du problème.

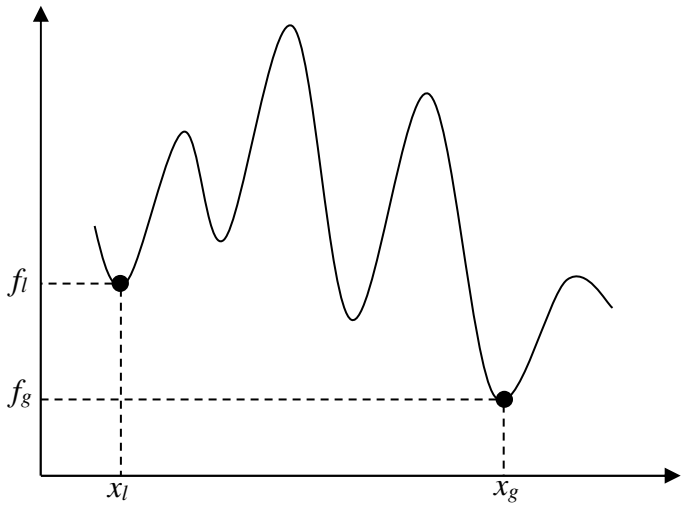


Figure 2.1 : *Optimum local et optimum global.*

2.2.3 Opérateurs de recherche fondamentaux

La recherche de l'optimum d'une fonction est généralement réalisée à l'aide de deux opérateurs fondamentaux : l'exploration et l'exploitation.

- **L'exploration** est la capacité de l'algorithme à explorer le domaine des variables pour rechercher la meilleur vallée, c'est-à-dire celle qui contient l'optimum global.
- **L'exploitation** est la capacité de l'algorithme à converger rapidement vers le minimum d'une vallée donnée à partir d'un point de départ.

Le succès et l'efficacité d'une technique de résolution dépendent la plupart du temps d'un compromis entre l'exploration et l'exploitation.
Tout algorithme d'optimisation doit utiliser ces deux stratégies pour trouver l'optimum global : l'exploration pour la recherche de régions inexplorées de l'espace de recherche et l'exploitation pour exploiter la connaissance acquise aux points déjà visités et ainsi trouver des points meilleurs. Ces deux exigences peuvent paraître contradictoires mais un bon algorithme de recherche doit trouver le bon compromis entre les deux. Une recherche purement aléatoire est bonne pour l'exploration mais pas pour l'exploitation alors que la recherche dans le voisinage est une bonne méthode d'exploitation mais pas d'exploration. Certaines méthodes toutefois n'utilisent qu'un seul de ces opérateurs pour parvenir à l'optimum. Ainsi, les méthodes déterministes, exploitant les dérivées de la fonction objectif pour atteindre rapidement et précisément le minimum local le plus proche du point de départ, privilégient l'exploitation au détriment de l'exploration **[42, 44]**.

2.2.4 Sensibilité et robustesse d'une méthode d'optimisation

L'efficacité d'une méthode d'optimisation est liée à la sensibilité et à la robustesse par rapport aux paramètres de contrôle et aux conditions initiales (valeur initiales des variables de conception, valeur initiales des paramètres de contrôle). Lorsque les variables de conception doivent prendre une valeur bien précise pour que la méthode de résolution converge vers l'optimum d'une fonction donnée, la méthode est dite sensible aux conditions initiales. Une méthode d'optimisation est robuste si pour une même valeur des paramètres de contrôle et des conditions initiales, elle est capable de trouver l'optimum de fonctions très différentes **[44]**.
Une méthode parfaite devrait être totalement insensible aux conditions initiales et aux variables de conception et converger vers l'optimum quelles que soient la fonction objectif et les contraintes.

2.2.5 Ordre d'une méthode de résolution

Les méthodes de résolution peuvent être classées à partir de leur ordre selon qu'elles nécessitent ou non le calcul des dérivées de la fonction objectif et des fonctions contraintes par rapport aux paramètres. Une méthode est dite d'ordre zéro si elle utilise uniquement la connaissance de la fonction elle-même. Elle est d'ordre un si elle requiert le calcul des dérivées première et d'ordre deux s'il lui faut aussi accéder aux dérivées secondes.

Les méthodes d'ordre zéro sont en général peu précises et convergent plus lentement vers l'optimum. En revanche, elles offrent l'avantage d'éviter le calcul du gradient, ce qui est intéressant lorsque la fonction n'est pas différentiable ou que le calcul de son gradient représente un cout important.

Les méthodes d'ordre un permettent d'accélérer la localisation de l'optimum, puisque le gradient donne l'information sur la direction de l'amélioration. Par contre elles sont applicables seulement aux problèmes où les fonctions objectif sont continument différentiables.

2.3 LES MÉTHODES D'OPTIMISATION LOCALE

Les algorithmes de recherche locale passent d'une solution à une autre dans l'espace des solutions candidates (l'espace de recherche) jusqu'à ce qu'une solution considérée comme optimale soit trouvée ou que le temps imparti soit dépassé. En revanche optimiser localement, c'est chercher une solution à un problème qui soit proche d'une solution de départ (optimisation locale), mais qui soit meilleure en terme de coût (fonction objectif). Pour cela, nous recherchons une meilleure solution par itérations successives, cette classe de méthodes peut être déterministe ou non-déterministe.

2.3.1 Optimisation déterministe

Lorsque l'évolution de la méthode de résolution est prévisible et ne laisse aucune place au hasard, celle-ci est qualifiée de déterministe. Les méthodes déterministes s'appuient sur une direction de recherche qui peut être fournie par les dérivées de la fonction objectif. Ces méthodes ont la réputation d'être efficaces lorsque la solution initiale est proche de l'optimum recherché.
La formulation du problème d'optimisation locale s'écrit :

$$\begin{Vmatrix} Trouver\,(x_l, f_l) \in C, \quad ou \quad f_l = f(x_l) \\ telque: \\ \forall x \in V, f(x_l) \le f(x) \\ ou\ V\ est\ un\ voisinage\ de\ x_l \end{Vmatrix} \tag{2.1}$$

En principe la mise à jour se fait selon une direction de recherche d_k par le processus itératif suivant :

$$x_{k+1} = x_k + \mu_k d_k \tag{2.2}$$

Avec : μ_k est un pas de déplacement et x_{k+1} est l'approximation de x_l à l'itération $k+1$.

Les méthodes de types gradient utilisent les gradients et/ou les Hessiens de f (ou leurs approximation) pour choisir des d_k et μ_k appropriés, de façon que la direction choisie soit une direction de descente.
Le principe général est le suivant : x_1 donné, nous construisons un processus x_k satisfaisant :

$$f(x_{k+1}) < f(x_k) \quad avec\ x_{k+1} = x_k + \mu_k d_k \tag{2.3}$$

Où d_k est une direction de descente et μ_k minimise $f(x_{k+1})$ dans la direction d_k.

Pour calculer d_k et μ_k, l'algorithme peut utiliser des informations sur les valeurs du gradient de f au point x_k courant.
Finalement l'algorithme général qui sert à résoudre le problème peut s'écrire comme suit :

Initialisation

x_0, μ_0, d_0

Itération

Si le test d'arrêt n'est pas vérifié faire

$x_{k+1} = x_k + \mu_k d_k$

Fin Si

Le choix de la direction de descente a conduit au développement de plusieurs algorithmes, chacun avec ses propres caractéristiques. Parmi ces algorithmes se trouvent **[42]**:

- **L'algorithme du gradient à pas fixe :** C'est la méthode la plus simple, qui consiste à construire le processus itératif suivant :

$$\begin{cases} x_1 \quad donné \\ x_{k+1} = x_k - \mu \nabla f(x_k) \end{cases} \tag{2.4}$$

  Où μ est appelée pas de descente, est une constante positive donnée, ne dépendant pas de *k*. La direction de descente est celle opposée à la direction du gradient. La convergence de la méthode n'est assurée que pour de bonnes valeurs de μ, c'est-à-dire ni trop grandes ni trop petites. En prenant de petites valeurs de μ, en effet la descente doit être vérifiée, c'est-à-dire que $f(x_{k+1}) < f(x_k)$.

- **L'algorithme du gradient à pas prédéterminé :** Il s'agit d'une amélioration de la méthode précédente. La direction de descente est toujours le gradient et μ_k est choisi tel que :

$$\begin{aligned} &\lim_{k \to +\infty} \mu_k = 0 \\ &\sum_{k=0}^{+\infty} \mu_k = +\infty \end{aligned} \tag{2.5}$$

- **L'algorithme du gradient à pas optimal (Steepest descent)** : Il est également connu sous le nom d'algorithme de la plus forte pente ou de la plus profonde descente. La méthode consiste à choisir le gradient comme direction de descente et à calculer, à chaque itération, la meilleure profondeur de descente. Donc l'algorithme est le suivant :

$$\begin{cases} x_1 \quad donné \\ x_{k+1} = x_k - \mu_k \nabla f(x_k) \end{cases} \tag{2.6}$$

  Où μ_k est calculé de manière à minimiser $f(x_{k+1})$ dans la direction du gradient, à l'étape *k*, la fonction $f(x_{k+1})$ ne dépend que de μ_k car $\nabla f(x_k)$ est fixé.

2.3.2 Optimisation non déterministe ou stochastique

L'optimisation stochastique est une méthode qui génère et utilise des variables aléatoires. L'optimisation stochastique n'est pas une famille de problèmes, de modèles et d'outils différents des autres domaines d'optimisation (programmation linéaire, non-linéaire, dynamique), au contraire, elle constitue un complément de ces familles lorsque la notion d'incertitude intervient. Dans ce cas nous parlerons de programmation stochastique linéaire, non-linéaire ou bien de programmation stochastique dynamique. En effet, le but qui se fixe dans un problème d'optimisation stochastique est de prendre la meilleure décision vis à vis des situations qui comportent de l'incertitude.

L'optimisation stochastique, concerne l'optimisation d'une fonction *f* non directement observable. On se propose de déterminer la valeur x_l du paramètre *x* pour laquelle *f* est minimum. Il existe deux cas : nous pouvons faire des observations bruitées de $\nabla f(x)$, ou des observations bruités de $f(x)$, c'est le cas le plus utilisé. La détermination de x_l se fera par une procédure du type gradient approché stochastique. Dans ce cas, pour estimer x_l, valeur du paramètre réel *x*, pour laquelle *f* est minimum, il suffit de trouver un estimateur du $\nabla f(x)$.

Dans ce même ordre d'idée, plusieurs chercheurs ont mis au point plusieurs processus intéressants, Parmi ces processus se trouvent **[42]** :

- **Le processus de Kiefer-Wolfowitz**

  Soit $\{e_1,....,e_d\}$ une base orthonormée de $\mathbb{R}^d$, et soit $y(x), x \in \mathbb{R}^d$ une famille de variables aléatoires. Soit μ_k et c_k deux suites de réels positifs. Le processus aléatoire multidimensionnel construit X_k (X_k est un vecteur aléatoire de $\mathbb{R}^d$). Le processus de Kiefer-Wolfowitz s'écrit comme suit :

$$\begin{cases} X_1 \quad \textit{vecteur aléatoire donné} \\ X_{k+1} = X_k + \mu_k W_k, \quad \mathrm{k} \geq 1 \end{cases} \tag{2.7}$$

  Où W_k est un vecteur aléatoire, qui apparaît comme une approximation stochastique en différence finie centrée du gradient $\nabla f(x_k)$ et dont la $j^{ème}$ composante est définie par :

$$W_k^j = \frac{y(x_k + c_k e_j) - y(x_k - c_k e_j)}{2c_k}, \; k \geq 1, \; j = 1,....,d \tag{2.8}$$

  Dans cette méthode, quand le nombre de dimension est grand la recherche devient de plus en plus laborieuse et l'algorithme converge en un temps énorme. En effet, si le problème est de dimension *d* alors en une seule itération pour évaluer le gradient il va falloir calculer la fonction objectif 2*d* fois.

- **Le processus de James Spall**

  Le processus de James Spall s'écrit comme suit :

$$\begin{cases} X_1 \quad vecteur\ aléatoire\ donné \\ X_{k+1} = X_k + \mu_k W_k, \quad k \geq 1 \end{cases} \tag{2.9}$$

  Où W_k est un vecteur aléatoire, qui peut être approché par l'estimateur suivant :

$$W_k^j = \frac{y(x_k + c_k\Delta_k) - y(x_k - c_k\Delta_k)}{2c_k\Delta_{kj}},\ k \geq 1,\ j = 1,....,d \tag{2.10}$$

  Où Δ_k est un vecteur de d variable aléatoires indépendants symétriquement distribuées autour de zéro. En général nous travaillons avec la distribution symétrique de Bernoulli ±1. Cette méthode ne requiert que deux évaluations de la fonction objectif par itération pour former le gradient et ceux indépendamment des dimensions du problème **[45, 46]**.

Dans les méthodes d'optimisation locale que nous avons présentés, les algorithmes cherchent le minimum d'une fonction en se basent sur la connaissance d'une direction de recherche. Bien sûr, ces méthodes seront toujours applicables et même recommandées lorsque la solution cherchée est réputée proche de la solution connue (point de départ) ou si la fonction objectif est convexe, en particulier, si le calcul n'est pas coûteux **[47]**. Toutefois, elles ne sont pas indiquées pour des problèmes où le risque de rester bloqué dans un optimum local est fort probable, cela suffit pour travailler avec d'autres types : L'optimisation globale.

2.4 LES MÉTHODES D'OPTIMISATION GLOBALE

Durant ces dernières années, l'optimisation globale a fait l'objet de plusieurs études grâce à des résultats théoriques nouveaux, une forte demande dans plusieurs domaines incluant des applications industrielles, et au développement des moyens de calcul. Cependant, les méthodes d'optimisation globale se basent sur une recherche globale sur tout l'espace de recherche. En fait, elles permettent d'échapper au problème de l'optimum local et de déterminer l'optimum global. C'est pourquoi beaucoup d'approches ont été utilisées dans la tentative de résoudre le problème. Il s'agit, de trouver un état de minimum et de ne s'arrêter que s'il est le meilleur (l'optimum global). On distingue deux types d'approches selon qu'elles incluent ou non des processus probabilistes : Les méthodes déterministes et les méthodes non-déterministes.

2.4.1 Méthodes déterministes

Ces méthodes ont tout d'abord été introduites pour résoudre de manière exacte des problèmes particuliers comme par exemple les problèmes continus et linéaires sous contraintes linéaires. La principale qualité des méthodes globales déterministes est qu'elles ne nécessitent pas de point de départ. Ces méthodes permettent de bien gérer les contraintes, contrairement aux méthodes stochastiques et permettent d'obtenir, à la convergence, la solution exacte du problème d'optimisation considéré avec une garantie

absolue, en utilisant l'arithmétique d'intervalles arrondie telle qu'elle a été définie par R. Moore [48]. Aucune erreur numérique insidieuse ne peut écarter de tels algorithmes de la solution optimale, il sera dans le pire des cas seulement ralenti. Ces algorithmes sont appelés : algorithmes de Branch-and-Bound par intervalles. Les plus intéressantes d'entre elles sont :

- Méthodes à Recherche par Quadrillage (Grid Search Methods) ;
- Méthodes des Trajectoires (Trajectory Methods) ;
- Méthodes de séparation-évaluation (Branch-and-Bound).

Cependant, il faut savoir que les méthodes déterministes globales restent utilisables tant que le nombre de variables ne devient pas trop important. Ces méthodes sont connues par le fait qu'elles garantissent l'optimalité de la solution mais elles sont très gourmandes en termes de temps de calcul et de l'espace mémoire nécessaire. C'est la raison pour laquelle, elles sont beaucoup plus utilisées pour la résolution de problèmes faciles. La nécessité de disposer d'une solution de bonne qualité (semi optimale) avec un coût de recherche (temps de calcul et espace mémoire) raisonnable a conduit les chercheurs à proposer un autre type de méthodes de résolution de problèmes, communément connues par les méthodes approchées. Ces dernières constituent une alternative aux méthodes exactes.

2.4.2 Méthodes non-déterministes

Ces méthodes non-déterministes font appel à des tirages de nombres aléatoires. Elles permettent d'explorer l'espace de recherche plus efficacement. Dans la suite on s'intéressera plus particulièrement aux méta-heuristiques.

Le mot méta-heuristique est dérivé de la composition de deux mots grecs : **heuristique** qui vient du verbe heuriskein (euriskein) et qui signifie trouver et **méta** qui est un suffixe signifiant au-delà, dans un niveau supérieur. La particularité qui différencie les méthodes méta-heuristiques des méthodes heuristiques c'est que les méta-heuristiques sont des méthodes générales de recherche applicables sur de nombreux problèmes. Tandis que, les heuristiques sont spécifiques à un problème donné. Elles nécessitent des connaissances du domaine du problème traité **[49].**

Dans la vie pratique, on se retrouve souvent confronté à des problèmes de différentes complexités, pour lesquelles on cherche des solutions qui satisfont deux notions antagonistes: la rapidité et la qualité. Les "méta-heuristiques" d'optimisation sont des algorithmes généraux d'optimisation applicables à une grande variété de problèmes. Elles sont apparues à partir des années 80, dans le but de résoudre au mieux des problèmes d'optimisation [50]. Les méthodes méta-heuristiques, contrairement à la plupart des méthodes déterministes, ne nécessitent ni point de départ, ni à la connaissance du gradient de la fonction objectif pour atteindre la solution optimale. Elles s'appuient sur des mécanismes de transition probabilistes et aléatoire qui explorent efficacement l'espace de recherche et convergent vers l'optimum global. Leur nature aléatoire implique que plusieurs exécutions successives de ces méthodes conduisent à des résultats différents pour une même initialisation du problème d'optimisation. Cependant elles demandent un nombre important d'évaluations de la fonction objectif

en comparaison avec les méthodes déterministes exploitant la dérivée de la fonction objectif.

L'efficacité d'une méta-heuristique dans la résolution d'un problème d'optimisation est liée à sa capacité d'établir un certain équilibre entre l'exploitation des expériences acquises (i.e. les solutions trouvées) au cours de la recherche et l'exploration de l'espace de recherche pour identifier d'autres solutions de très bonne qualité **[49]**. Il est à noter que les termes « exploitation » et « exploration » peuvent être remplacés par les termes « intensification » et « diversification » respectivement (voir figure 2.2).

La différence principale entre ces deux notions réside dans le fait qu'une phase d'intensification vise à forcer une solution donnée à tendre vers l'optimum local de la zone à laquelle elle est attachée **[51]**. Tandis qu'une phase de diversification encourage le processus de recherche à examiner des régions non visitées et à découvrir des solutions différentes des solutions rencontrées afin d'explorer l'espace de recherche dans une échelle globale **[52]**.

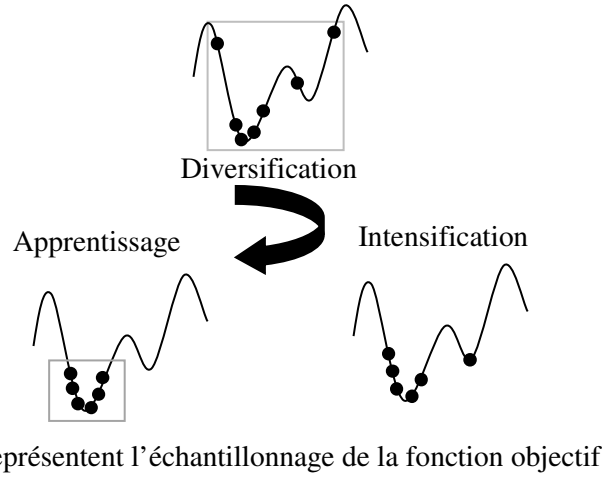


Figure 2.2 *Les trois phases d'une méta-heuristique.*

La majorité des algorithmes méta-heuristiques sont inspirées des systèmes naturels, ils peuvent être regroupés en quatre catégories principales **[53]** (voir figure 2.3) : les algorithmes évolutionnaires, les algorithmes basés sur la physique, les algorithmes basés sur les essaims, les algorithmes basés sur les comportements humains. Les algorithmes évolutionnaires sont inspirés des lois de l'évolution naturelle. Le processus de recherche commence par une population générée de manière aléatoire. Le point fort de ces algorithmes est que les meilleurs individus sont toujours combinés pour former la prochaine génération d'individus, cela permet d'optimiser la population. Les algorithmes évolutionnaires les plus populaires sont : les algorithmes génétiques (GA : Genetic Algorithms) **[54]**, qui simule l'évolution darwinienne. La stratégie d'évolution (ES, pour Evolution Strategy) **[55]**, l'apprentissage incrémental basé sur les probabilités (PBIL : Probability-Based Incremental Learning) **[56]**, la programmation génétique (GP : Genetic Programming) **[57]** et l'optimiseur basé sur la biogéographie (BBO : Biogeography-Based Optimizer) **[58]**.

Les algorithmes basés sur la physique imitent les règles physiques de l'univers. Les algorithmes les plus populaires sont le recuit simulé (SA : Simulated Annealing) [59], la recherche gravitationnelle locale (GLSA : Gravitational Local Search Agorithm) [60], l'algorithme de recherche gravitationnelle (GSA : Gravitational Search Algorithm) [61], la recherche de système chargé (CSS : Charged System Search) [62], l'optimisation de la force centrale (CFO : Central Force Optimization) [63]. Algorithme d'optimisation de la réaction chimique artificiel (ACROA : Artificial Chemical Reaction Optimization Algorithm) [64], Algorithme de recherche basé sur la galaxie (GbSA : Galaxy-based Search Algorithm) [65].

Le troisième groupe comprend les algorithmes basés sur l'intelligence d'essaim qui imitent le comportement social de groupes d'animaux. L'algorithme le plus populaire est l'algorithme d'optimisation par essaim de particules (PSO : Particle Swarm Optimization), développée à l'origine par Kennedy et Eberhart [66]. Autres algorithmes populaires basés sur les essaims sont : Écholocalisation des dauphins (DE : Dolphin Echolocation) [67], L'algorithme de colonie d'abeille artificielle (ABC : Artificial bee colony) [68], Optimisation par colonie de fourmis (ACO : Ant Colony Optimization) [69], La recherche Coucou (CS : Cuckoo Search) [70].

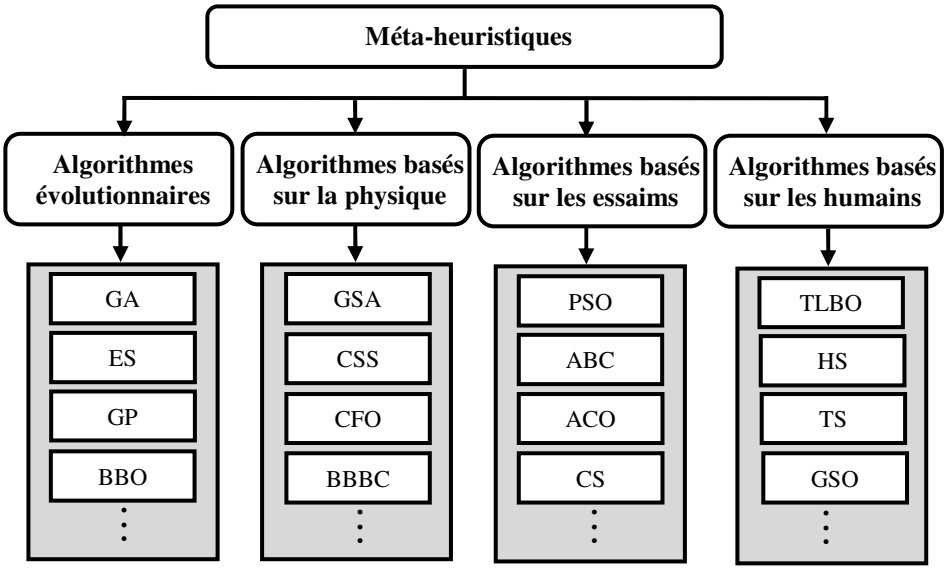


Figure 2.3 *Classification des algorithmes méta-heuristiques.*

Le quatrième groupe comprend les algorithmes basés sur les comportements humains. Les algorithmes les plus populaires sont : l'optimisation basée sur l'enseignement et l'apprentissage (TLBO : Teaching Learning Based Optimization) [71], recherche d'harmonie (HS: Harmony Search) [72], Recherche Tabou (TS: Taboo Search) [73, 74], Optimiseur de recherche de groupe (GSO : Group Search Optimizer) [75, 76].

L'ensemble des méta-heuristiques proposées dans la littérature sont partagées en deux classes: des méta-heuristiques à base de solution unique et des méta-heuristiques à base de solution multiple [49].

- **Les méta-heuristiques à base de solution unique**

  Les méta-heuristiques à base de solution unique sont appelées méthodes de recherche locale ou méthode de trajectoire, commencent la recherche avec une seule solution initiale. Elles se basent sur la notion du voisinage pour améliorer la qualité de la solution courante (voir figure 2.4). En fait, la solution initiale subit une série de modifications en fonction de son voisinage. Le but de ces modifications locales est d'explorer le voisinage de la solution actuelle afin d'améliorer progressivement sa qualité au cours des différentes itérations. Le processus s'arrête si la solution courante ne peut pas être améliorée ou le nombre d'itération maximum est atteint.

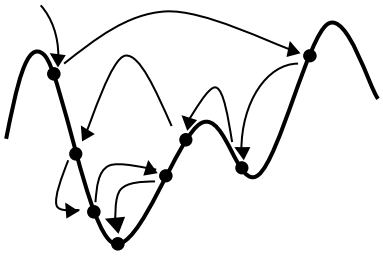


Figure 2.4 *Exemple d'une solution unique.*

La qualité de la solution finale dépend particulièrement des modifications effectuées par les opérateurs de voisinages. En effet, les mauvaises transformations de la solution initiale mènent la recherche vers la vallée de l'optimum local d'un voisinage donné (peut être un mauvais voisinage) ce qui bloque la recherche en fournissant une solution de qualité insuffisante. De nombreuses méthodes à base de solution unique ont été proposées dans la littérature. Parmi lesquelles: La recherche par descente, le recuit simulé, la recherche tabou, la recherche à voisinage variable (VNS: Variable Neighbourhood Search) **[77]**, la recherche locale réitérée (ILS: Iterated Local Search) **[78]**, Recherche locale guidée (GLS: Guided Local Search) **[79]**…etc.

- **Les méta-heuristiques à base de solution multiple**

  Les méta-heuristiques à base de solution multiple débutent la recherche avec un ensemble de points de l'espace de recherche (Fig 2.5). Elles s'appliquent sur une population de solution initiale puis de l'améliorer afin d'extraire la meilleure (l'optimum global) qui représentera la solution du problème traité. L'idée d'utiliser un ensemble de solutions au lieu d'une seule solution renforce la diversité de la recherche et augmente la possibilité d'émergence de solutions de bonne qualité. Une grande variété de méthodes basées sur une solution multiple a été proposée dans la littérature, commençant par les algorithmes évolutionnaires, passant par les algorithmes génétiques et arrivant aux algorithmes à base d'intelligence par essaims (l'algorithme d'optimisation par essaim de particules, l'algorithme de colonies de fourmis, l'algorithme de colonies d'abeilles, la recherche coucou, l'algorithme d'optimisation par

coucou…) qui ont connus une investigation remarquable ces deux dernières décennies.

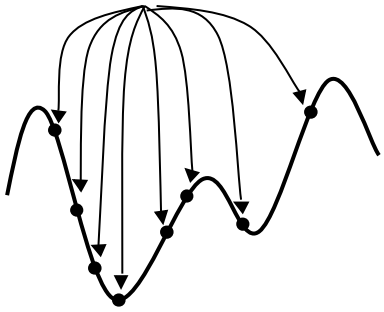


Figure 2.5 *Exemple d'une solution multiple.*

Dans la suite on s'intéressera plus particulièrement aux algorithmes à population de solutions. Les algorithmes à population de solutions simulent le processus d'évolution d'une population. À partir d'une population de N solutions du problème représentant des individus, on applique des opérateurs pour obtenir une population de solutions de mieux en mieux adaptées au problème. Cette adaptation est évaluée grâce à la fonction objectif associée au problème d'optimisation.
Parmi les algorithmes à population de solutions les plus utilisées, nous mettons dans la suite l'accent sur celles que nous avons étudiées dans le cadre de cette thèse : l'algorithme d'optimisation par essaim de particules (PSO : Particle Swarm Optimization) **[66]**, l'algorithme des sauts de grenouilles (SFLA : Shuffled Frog-Leaping Algorithm) **[80, 81]**, et l'algorithme d'optimisation de recherche dirigé (DSO : Directed Searching Optimization Algorithm) **[82]**.

2.4.2.1 L'optimisation par essaim de particules (PSO)

Les algorithmes d'optimisation par essaim ont été introduits comme une alternative aux algorithmes génétiques standards. Ces algorithmes sont inspirés des essaims d'insectes (ou des bancs de poissons ou des nuées d'oiseaux) et de leurs mouvements coordonnés. En effet, tout comme ces animaux se déplacent en groupe pour trouver de la nourriture ou éviter les prédateurs, les algorithmes d'optimisation par essaim recherchent des solutions pour un problème d'optimisation. L'algorithme le plus populaire est l'algorithme d'optimisation par essaim de particules (PSO : Particle Swarm Optimization), développée à l'origine en 1995 par Kennedy et Eberhart **[66].** Les individus de l'algorithme sont appelés particules et la population est appelée essaim. L'algorithme commence avec une initialisation aléatoire de l'essaim de particules dans l'espace de recherche. Les particules de l'essaim représentent des solutions potentielles au problème traité. Chaque particule est caractérisée par sa vitesse de déplacement et par sa position dans l'espace de recherche **[49]**. Au cours du processus de la recherche, chaque particule se déplace pour modifier sa position dans l'espace de recherche en fonction de sa vitesse actuelle, sa position actuelle, sa meilleure position trouvée au cours des itérations passées et la meilleure position trouvée par l'essaim. Son déplacement lui permet de mettre à jour sa position et sa vitesse de déplacement à

chaque itération. Les étapes principales de l'algorithme d'optimisation par essaim de particules sont présentées comme suit :

Etape 1 : Initialisation
Taille de la population (*P*)
Positions, vitesses et paramètres nécessaires
Etape 2 :
Evaluer les positions des particules
Etape 3 :
Enregistrer la meilleure position
Etape 4 :
Calculer la nouvelle vitesse de chaque particule
Calculer la nouvelle position de chaque particule
Etape 5 :
Evaluer les positions des particules
Etape 6 :
Trouver la meilleure position de chaque particule
Trouver la meilleure position par l'essaim
Etape 7 :
Aller à l'étape 4 si le critère d'arrêt n'est pas atteint.

⇒

Pour N itération
Mise à jour de v_{id} et x_{id}
Appliquer l'équation (2.11) et (2.12)
Si $f(x_{id}) < f(p_{bestid})$
$p_{bestid} = v_{id}$
Si $f(p_{bestid}) < f(g_{bestd})$
$g_{bestd} = p_{bestid}$
Fin Si
Fin Si
Fin Pour

Chaque particule *i* de l'essaim est définie par sa position $x_{id} = (x_{i1}, x_{i2}, ..., x_{iD})$, et sa vitesse de déplacement $v_{id} = (v_{i1}, v_{i2}, ..., v_{iD})$ dans un espace de recherche de dimension *D*. Cette particule garde en mémoire la meilleure position par laquelle elle est déjà passée et la meilleure position atteinte par toutes les particules de l'essaim, notées respectivement: $p_{bestid} = (p_{besti1}, p_{besti2}, ..., p_{bestiD})$ et $g_{bestd} = (g_{best1}, g_{best2}, ..., g_{bestD})$.
La particule *i* va se déplacer entre les itérations *t* et *t*+1, en fonction de sa vitesse et des deux meilleures positions qu'elle connaît (la sienne et celle de l'essaim) suivant les deux équations suivantes :

$$v_{id}(t) = v_i(t-1) + c_1 r_1 \left(p_{bestid}(t-1) - x_{id}(t-1) \right) + c_2 r_2 \left(g_{bestd}(t-1) - x_{id}(t-1) \right) \quad (2.11)$$

$$x_{id}(t) = x_{id}(t-1) + v_{id}(t) \quad (2.12)$$

Avec :

$x_{id}(t), x_{id}(t-1)$: La position de la particule *i* dans la dimension *d* aux temps *t* et *t*-1, respectivement.
$v_i(t), v_i(t-1)$: La vitesse de la particule *i* dans la dimension *d* aux temps *t* et *t*-1, respectivement.

$p_{bestid}(t-1)$: La meilleure position obtenue par la particule *i* dans la dimension *d* au temps *t*-1.

$g_{bestd}(t-1)$: La meilleure position obtenue par l'essaim dans la dimension *d* au temps *t*-1.

c_1, c_2 : Deux constantes qui représentent les coefficients d'accélération.

r_1, r_2 : Nombres aléatoires tirés de l'intervalle [0,1].

Afin d'estimer la qualité de la particule *i*, il est indispensable de calculer sa fonction « objectif » (aussi appelée fitness). La valeur de la fonction « objectif » de la particule x_{id} est notée $f(x_{id})$. Cette dernière est calculée en utilisant une fonction spéciale au problème traité. Afin de mettre à jour les valeurs de x_{id}, p_{bestid} et g_{bestid}, leurs fitness sont calculées à chaque itération de l'algorithme. x_{id} est mise à jour selon l'équation (2.12). p_{bestid} et g_{bestd} sont mises à jour si les conditions présentées ci-dessous sont vérifiées respectivement :

Condition 1 : $f(x_{id})$ est meilleur que $f(p_{bestid})$

Conditions 2 : $f(p_{bestid})$ est meilleur que $f(g_{bestd})$

À chaque itération du processus de la recherche, les particules se déplacent en fonction des équations (2.11) et (2.12). Le processus est répété jusqu'à la satisfaction du critère d'arrêt.

2.4.2.2 Algorithme des sauts de grenouilles (SFLA)

L'algorithme des sauts de grenouille est un algorithme basé sur la méta-heuristique mémétique. Cet algorithme connu sous son appellation anglophone SFLA (Shuffled Frog-Leaping algorithm), il a été proposé et développé par Eusuff et Lansey **[80, 81]** pour résoudre des problèmes d'optimisation combinatoire. L'algorithme SFLA est un algorithme de recherche coopérative basé sur la population. Il combine les avantages de l'algorithme d'évolution mémétique et de l'optimisation par essaims de particules (PSO). Il se compose d'un ensemble de grenouilles (chacune représente une solution au problème) partitionnées en différent groupes appelés mèmeplexes **[50]**, et qui évoluent indépendamment pour parcourir l'espace des solutions dans les différentes directions. Les grenouilles peuvent communiquer entre elles et améliorer leurs solutions par contamination (passant l'information). L'algorithme contient des éléments de recherche locale effectuée dans chaque groupe avec un échange d'information globale. L'information entre les différents mèmeplexes circule par l'intermédiaire d'un processus de saut. Au début, les individus initiaux de la population *P* sont produits de façon aléatoire (voir les étapes décrites ci-dessous). Supposons que la population initiale est constituée de *F* grenouilles définie aléatoirement dans l'espace $(X_i,\ i=1,2,.....,F)$. L'adaptabilité (fitness) $f(i)$ de la $i^{ème}$ grenouille représente la valeur de la fonction objectif (fitness). Toutes les grenouilles sont triées dans un ordre décroissant selon la fonction objectif (fitness), et sont divisées en *m* mèmeplexes contenant chacune *n* grenouilles ($F = m \times n$), de telle manière que la première grenouille appartient au premier mèmeplexe, la deuxième grenouille est affectée au deuxième mèmeplexe etc.

Dans chaque mèmeplexe, les grenouilles fournissant la meilleure solution X_b et la plus mauvaise X_w. La grenouille donnant la meilleure solution dans la population entière est notée par X_g. Les étapes de l'algorithme général SFLA sont présentées comme suit :

Etape 1 : Initialisation
Taille de la population (*P*)
Nombre de mèmeplexes (*N*)
Nombre d'itération

Etape 2 :
Générer P aléatoirement
Evaluer chaque solution

Etape 3 :
Trier *P* par ordre décroissant
Enregistrer la meilleure position X_g

Etape 4 :
Partitionner *P* en *m* mèmeplexes

Etape 5 :
Recherche local

Etape 6 :
Regrouper les *m* mèmeplexes

Etape 7 :
Aller à l'étape 3 si le critère d'arrêt n'est pas atteint.

Pour N itération
Déterminer X_w et X_b
Mise à jour de X_w
Appliquer l'équation (2.14)
Si $f\left(X_w^{'}\right) < f\left(X_w\right)$
$X_w = X_w^{'}$
Sinon
Appliquer l'équation (2.16)
Si $f\left(X_w^{'}\right) < f\left(X_w\right)$
$X_w = X_w^{'}$
Sinon
Appliquer l'équation (2.17)
Fin Si
Fin Si
Fin Pour

Pendant l'évolution d'un mèmeplexe, c.-à-d., pendant l'exploration locale, la plus mauvaise grenouille X_w effectue un saut vers la meilleure X_b selon la règle suivante (figure 2.6) **[47]** :

$$S = rand(.).\left(X_b - X_w\right) \quad (0 < rand < 1) \tag{2.13}$$

$$X_w\left(nouveau\right) = X_w + S \quad \left(-S_{\max} \le S \le S_{\max}\right) \tag{2.14}$$

Où S représente la valeur du saut, S_{max} est le saut maximal autorisé et *rand(.)* est un nombre aléatoire compris entre 0 et 1.

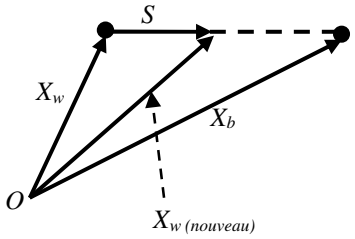


Figure 2.6 *Règle du saut de grenouille (frog leaping).*

Si le saut produit une meilleure solution, alors cette solution X_w *(nouveau)* remplace la plus mauvaise X_w. Sinon, on applique la même règle en remplaçant cette fois X_b par la solution globale X_g :

$$S = rand(.).(X_g - X_w) \quad (0 < rand < 1) \tag{2.15}$$

$$X_w(nouveau) = X_w + S \quad (-S_{\max} \le S \le S_{\max}) \tag{2.16}$$

Si la nouvelle solution reste moins bonne que X_w, on génère aléatoirement une autre solution qui remplace X_w :

$$X_w(nouveau) = X_{\min} + rand(.).(X_{\max} - X_{\min}) \tag{2.17}$$

Où X_{max} et X_{min} représentent le saut maximal et minimal autorisé.

Afin d'assurer l'exploration globale, les mèmeplexes sont mélangés et réorganisés à nouveaux pour former une nouvelle population, ce mécanisme est répété jusqu'à satisfaire un critère d'arrêt.

2.4.2.3 Algorithme d'optimisation de recherche dirigé (DSO)

L'algorithme d'optimisation de recherche dirigé a été proposé en 2011 par Zou et al **[82]**. L'algorithme DSO comprend deux opérations importantes: la mise à jour de position et la mutation génétique. La première opération pour améliorer la convergence du DSO. La deuxième opération pour améliorer la capacité de s'échapper de l'optimum local. De plus, l'algorithme adopte une fonction de pénalité pour équilibrer les violations d'objectifs et de contraintes **[82]**. Pour mettre à jour la position, l'algorithme DSO utilise six paramètres. Elles sont:

- Taille de la population (P_S)
- Nombre maximal d'itérations (K)
- Probabilité de transfert (p_α)
- Coefficient avant (α)
- Coefficient arrière (β)
- Probabilité de mutation génétique (p_m).

Les six paramètres de l'algorithme sont initialisés à l'étape 1 (voir les étapes décrites ci-dessous). Pour générer la population initiale, une distribution uniforme est choisie dans l'intervalle $[x_{iL}, x_{iU}]$ (i = 1,2, ..., N), comme indiqué dans l'équation (2.18):

$$Pop = \begin{pmatrix} x_1^1 & x_2^1 & \dots & x_N^1 \\ x_1^2 & x_2^2 & \dots & x_N^2 \\ \vdots & \vdots & \vdots & \vdots \\ x_1^{P_s} & x_2^{P_S} & \dots & x_N^{P_S} \end{pmatrix} \tag{2.18}$$

Où x_i^j est le $i^{\text{ème}}$ (i = *1, 2,…,N*) composant du $j^{\text{ème}}$ (j = 1, 2, ….,*PS*) vecteur de solution. x_{iL} et x_{iU} sont la limite inférieure et la limite supérieure de la composante de position, respectivement. Les étapes de l'algorithme DSO sont présentées comme suit :

Etape 1 : Initialisation
Taille de la population (*PS*)
Nombre maximal d'itérations (*K*)
Probabilité de transfert (p_α)
Coefficient avant (α)
Coefficient arrière (β)
Probabilité de mutation génétique (p_m)
Etape 2 :
Générer une population (*PS*) par une distribution uniforme donnée par l'équation (2.18)
Etape 3 :
Utiliser les deux opérations : mise à jour de position et mutation génétique, pour améliorer les mauvaises positions.
Etape 4 :
Aller à l'étape 3 si le critère d'arrêt n'est pas atteint.

Pour $j \in [1, PS]$ et $j \neq j_g$ **Faire**
Pour $i \in [1, \mathrm{N}]$ **Faire**
Si $rand(\ ) < p_\alpha$
$x_v = x_i^j(k) + (1+\alpha) \times \left\{ x_i^{j_g}(k) - x_i^j(k) \right\}$
$x_i^j(k+1) = x_i^j(k) + r \times \left(x_v - x_i^j(k) \right)$
Sinon
$x_s = x_i^j(k) - \beta \times \left\{ x_i^{j_g}(k) - x_i^j(k) \right\}$
$x_i^j(k+1) = x_i^j(k) + r \times \left(x_s - x_i^j(k) \right)$
Fin Si
Si $rand(\ ) < p_m$
$x_i^j(k+1) = x_{iL} + r \times \left(x_{iU} - x_{iL} \right)$
Fin Si
Fin Pour
Fin Pour

La position est mise à jour comme indiqué sur la figure 2.7. La région avant est sélectionnée comme principale région de recherche qui est en fait une région proche du $x_i^{j_g}(k)$. Dans cette région $x_i^j(k)$ est située à P, $x_i^{j_g}(k)$ est située à Q et x_v est située à *V*, qui est sur la ligne d'extension avant du segment PQ. La mise à jour de la position $x_i^j(k)$ est déterminée par le paramètre (p_α): s'il est satisfait, la région avant sera le choix approprié, sinon la région arrière est considérée.

Dans la région arrière, x_s est située à S, qui est sur la ligne d'extension arrière du segment PQ. La région arrière est une région supplémentaire, et elle permet de ralentir la vitesse de convergence de l'algorithme DSO, ce qui est bénéfique pour éviter la convergence prématurée du DSO. L'étape d'adaptation donnée dans l'équation 2.19 ajusté dynamiquement pour maintenir un équilibre entre la recherche globale et la recherche locale de l'algorithme DSO.

$$step_i^j(k) = \left| x_i^{j_g}(k) - x_i^j(k) \right| \tag{2.19}$$

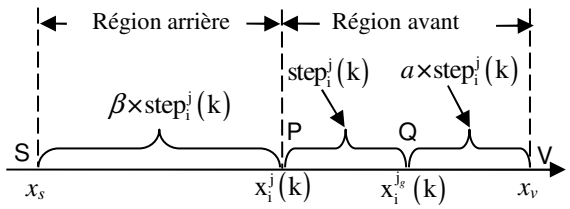


Figure 2.7 *Mise à jour de position.*

La mutation génétique est également une opération efficace et nécessaire de l'étape 3, car elle peut augmenter la diversité des individus, ce qui peut effectivement améliorer la performance du DSO en empêchant la convergence prématurée vers les minima locaux.

2.5 CONCLUSION

Nous avons présenté dans ce chapitre les définitions générales des méthodes d'optimisation qui se divisent en deux grandes classes : les méthodes déterministes et les méthodes non déterministes. Lorsque l'évolution de la méthode de résolution est prévisible et ne laisse aucune place au hasard, celle-ci est qualifiée de déterministe. Les méthodes déterministes s'appuient sur une direction de recherche qui peut être fournie par les dérivées de la fonction objectif. Ces méthodes ont la réputation d'être efficaces lorsque la solution initiale est proche de l'optimum recherché. Les méthodes non-déterministes ont comme caractéristiques communes leurs caractères stochastiques, c.à.d. qu'une partie de la recherche est conduit de façon aléatoire. Une famille de ces méthodes d'optimisation a été présentée : les méta-heuristiques. Les méta-heuristiques sont des méthodes stochastiques qui visent à résoudre un large panel de problèmes, sans pour autant que l'utilisateur ait à modifier leur structure. Ces méthodes sont inspirées d'analogies avec des domaines aussi variés que la physique, la génétique ou encore l'éthologie.

CHAPITRE 3

TECHNIQUES D'EGALISATION

3.1 INTRODUCTION

La transmission d'un signal numérique à travers un canal de transmission conduit à l'ajout de bruit (dégradation du rapport signal à bruit) et à une distorsion due aux limitations de la bande passante du canal (qui provoque le phénomène d'interférences entre symboles). La présence inévitable de bruit et d'interférences entre symboles introduit des erreurs dans le dispositif de décision. Si le canal de transmission avait une atténuation constante et un déphasage linéaire sur la bande du signal, il ne modifierait pas la forme des impulsions émises et le récepteur recevrait tout simplement une version bruitée du signal émis. En pratique, ces deux conditions ne sont que très rarement vérifiées et la réponse du canal a besoin d'être égalisée pour éliminer la distorsion du signal reçu. Dans ce chapitre nous présentons une étude des techniques d'égalisation permettant d'annuler la distorsion provoquée par le canal en annulant ses effets parasites. Dans un premier temps, nous présentons les principales structures d'égaliseur puis, nous comparons leurs performances respectives à partir d'un canal linéaire et non-linéaire à phase non minimale.

3.2 PRINCIPE DE L'EGALISATION

L'égalisation est une procédure qui est utilisée par les récepteurs des systèmes de communications numériques afin de réduire l'effet d'interférence entre symboles (IES) due à la propagation du signal modulé à travers le canal. Pour empêcher une dégradation sévère des performances, il est nécessaire de compenser la distorsion du canal. Cette compensation est effectuée par un filtre adaptatif (figure 3.1).

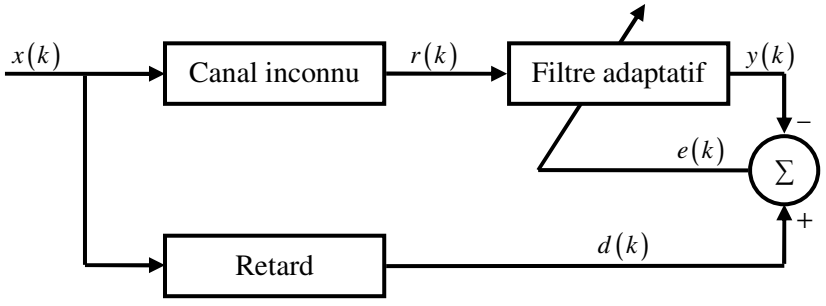


Figure 3.1 : *Compensation de la distorsion par un filtre adaptatif.*

Une chaîne de transmission numérique, en présence d'égalisation, peut être représentée par le schéma de principe de la figure 3.2.

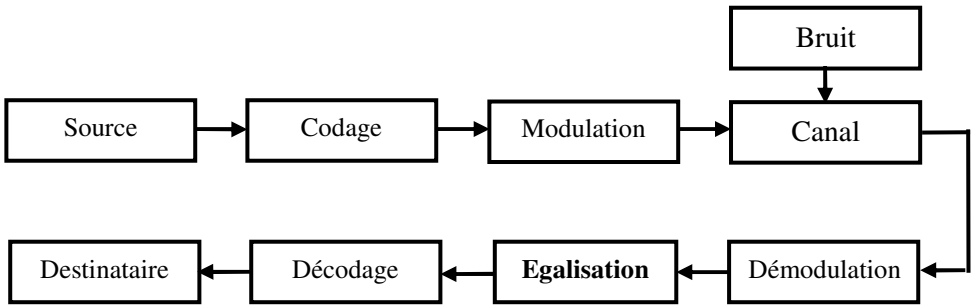


Figure 3.2 : C*haîne de transmission en présence d'égalisation.*

L'ensemble modulateur, milieu de transmission, démodulateur est modélisé par un canal discret équivalent de réponse *H(z)* **[24]**. Cette modélisation correspond au cas où les données *x[k]* sont émises tous les T_s secondes et le signal reçu en sortie du démodulateur est échantillonné à la fréquence $1/T_s$. la quantité T_s est appelée la durée symbole et $1/T_s$ représente la rapidité de modulation.
Le canal discret équivalent est défini par les *L+1* coefficients non nuls *h[i]*, de sa réponse impulsionnelle, ces coefficients seront supposés constants. On définit alors classiquement la transformée en *z* de cette réponse impulsionnelle comme :

$$H(z) = \sum_{i=0}^{L} h[i] z^{-i} \tag{3.1}$$

Le canal est perturbé par un bruit blanc additif, dont échantillons *v[k]* sont centrés, gaussiens et de variance σ_w^2.

$$\sigma_w^2 = E\left\{ |v[k]|^2 \right\} \tag{3.2}$$

On supposera de plus qu'il est alimenté par des symboles *x[k]* centrés, mutuellement indépendants et de variance σ_d^2 normalisée.

$$\sigma_x^2 = E\left\{ |x[k]|^2 \right\} = 1 \tag{3.3}$$

Ainsi, la sortie bruitée *r[k]* du canal discret équivalent peut s'écrire sous la forme :

$$r[k] = \sum_{i=0}^{L} h[i] x[k-i] + v[k] \tag{3.4}$$

Le rapport signal à bruit (SNR : Signal-to-Noise Radio) à l'entrée de l'égaliseur, défini comme le rapport entre la puissance du signal reçu non bruité et la puissance du bruit *v[k]* est donc égal à :

$$SNR = \frac{E\left\{ \left| \sum_{i=0}^{L} h[i] x[k-i] \right|^2 \right\}}{E\left\{ |v[k]|^2 \right\}} \tag{3.5}$$

L'objet de la fonction d'égalisation est de permettre de retrouver à partir de la séquence reçue *r[k]*, présentant de l'interférence entre symboles (IES) introduite par la sélectivité du canal, la séquence émise *x[k]*.
En principe, si le canal est parfaitement connue, il est possible en théorie de minimiser ou même d'annuler l'IES complètement, en utilisant une paire de filtres d'émission et de réception, de telle sorte que la chaîne complète de transmission forme un canal de Nyquist. Si les filtres d'émission et de réception sont fixés, le rôle de l'égaliseur est simplement de compenser la réponse du canal. Cascader le filtre adaptatif avec le canal (système inconnu) fait converger le filtre adaptatif à une solution qui est l'inverse du canal. Si la fonction de transfert du canal est *H(z)* et la fonction de transfert du filtre adaptatif est *E(z)* (figure 3.1), l'erreur à mesurer entre le signal désiré et le signal du système cascadé atteint son minimum quand le produit de *H(z)* et *E(z)* égale 1. Pour que cette relation soit vraie, *E(z)* doit égale à l'inverse de la fonction de transfert du canal.

$$E(z) = \frac{1}{H(z)} \tag{3.6}$$

En pratique, la réponse du canal est en général inconnue. On ne connaît que très rarement les caractéristiques du canal. En outre, le canal peut ne pas être stationnaire, c'est-à-dire que ses caractéristiques varient au cours du temps. L'exemple type est le canal hertzien, qui est fortement non stationnaire. Ces effets conduisent à maintenir une interférence entre symbole résiduelle et variable dans le temps qu'il faut compenser à l'aide d'un filtrage adaptatif capable de s'adapter au canal et de poursuivre ses variations temporelles.
Pour combattre l'effet de l'interférence entre symboles, deux techniques peuvent être employées.

- Une première technique, appelée détection suivant la séquence la plus vraisemblable **[83]** (MLSE : Maximum Likelihood Sequence Estimation) qui donne d'excellents résultats sous réserve que le canal soit connu ou bien estimé **[1]**. La mise en œuvre de cette technique est généralement réalisée en utilisant l'algorithme de Viterbi. Celui-ci permet de sélectionner dans un treillis le chemin de métrique la plus faible, où la métrique est une mesure cumulée, mise à jour de nœud en nœud. Toutefois lorsque la durée de la réponse impulsionnelle du canal est importante et/ou lorsque l'on utilise des modulations à grand nombre d'états, cette technique nécessite un volume de calcul qui devient rapidement prohibitif.
- Une seconde technique, appelée égalisation, consiste à inverser la réponse du canal. Les égaliseurs linéaires (LE : Linear Equalizer) et les égaliseurs à retour de décisions (DFE : Decision-Feedback Equalizer) sont les plus régulièrement employés. L'égalisation linéaire est largement utilisée pour les canaux de type téléphonique. Cependant sur les canaux présentant de sévères distorsions d'amplitude, un égaliseur linéaire donne des résultats assez médiocres du fait qu'il rehausse le bruit aux fréquences présentant de fortes atténuations. Au contraire, le DFE possède des performances proches du récepteur optimal (MLSE) pour une large classe de canaux lorsque le bruit est faible **[40]**.

Les égaliseurs de type LE, DFE possèdent l'avantage d'être assez simples à mettre en œuvre car ils sont composés de filtres numériques à coefficients complexes.

Pour l'optimisation des coefficients d'égaliseur, il existe essentiellement deux critères :

- Le premier critère consiste à forcer la réponse impulsionnelle du couple canal / égaliseur à zéro, cette approche d'égalisation est appelée forçage à zéro (ZF : Zero Forcing). Cette approche tente d'inverser exactement la fonction de transfert du canal, ce qui est *a priori* précisément le but recherché. On a ainsi *E(z) = 1/H(z),* on peut s'apercevoir que cette approche souffre de deux défauts**[23]** : d'abord, *H(z)* peut posséder des zéros de module supérieur à 1, ce qui induit des pôles instable pour *E(z)*, si celui-ci doit être causal, d'autre part, si *h(n)* est une réponse impulsionnelle finie, alors e*(n)* est à réponse impulsionnelle infinie.
  Considérons un seul bloc équivalent au canal discret et à l'égaliseur. Il est représenté par sa réponse impulsionnelle *q(n)* telle que :

$$q(n) = \sum_i e(i)\, h(n-i) \tag{3.7}$$

  A la sortie de l'égaliseur on peut donc écrire :

$$y(k) = q_0 d_k + \sum_{n \neq k} d_n q(n-k) + \sum_n e_n . \tilde{w}(k-n) \tag{3.8}$$

  On retrouve trois termes : le signal utile, le terme d'IES et le terme de bruit en sortie du filtre égaliseur. La distorsion maximale est la valeur maximale du terme d'IES, et cette valeur de distorsion dépond des coefficients de l'égaliseur pour un canal donné. Donc, s'il est possible de choisir les coefficients de telle sorte que ce terme de distorsion soit nul, on aura éliminé l'IES. Cette condition recherchée (c'est le critère ZF) s'écrit **[84]** :

$$q(n) = \sum_i e(i)\, h(n-i) = \begin{cases} 1 & n = 0 \\ 0 & \text{n} \neq 0 \end{cases} \tag{3.9}$$

  On peut remarquer que le bruit est négligé dans le développement du critère ZF. En pratique, le bruit est cependant toujours présent et bien que les termes d'IES soient éliminés, il y a des chances que le filtre égaliseur amplifie l'effet du bruit et donc dégrade les performances. En effet, la fonction de transfert du canal est en général de type passe-bas, et son inverse est de type passe haut. Lorsque le bruit est large bande, il s'en suit une forte augmentation du bruit en haute fréquence et une dégradation du rapport signal-à-bruit **[23]**. L'annulation des interférences entre symboles par le critère ZF se fait généralement au prix d'une augmentation sensible du niveau de bruit.

- Le second critère consiste à adapter les coefficients de l'égaliseur par la minimisation de l'erreur quadratique moyenne entre la séquence égalisée et la séquence désirée. Cette approche est appelée erreur quadratique moyenne (EQM ou MSE : Mean Square Error). Le bruit est ainsi pris en compte dans le critère. L'égalisation obtenue par cette approche est de meilleure qualité que celle fournie

par un Zero Forcing, en raison de la prise en compte effective du bruit. Par ailleurs, pour la mise en œuvre pratique, il est nécessaire de connaitre la séquence désirée. Pour ce faire, on utilise une séquence connue du récepteur (séquence d'apprentissage), pour calculer les coefficients du filtre. La nécessité d'inclure dans l'émission une séquence d'apprentissage, éventuellement répétée périodiquement si le système est non stationnaire, limite en outre le débit en données utiles. Nous nous limiterons dans ce chapitre à l'étude des égaliseurs selon le critère de la minimisation de l'EQM qui semble être le plus performant lorsque le canal est bruité.

3.3 PRINCIPALES STRUCTURES D'EGALISEURS

Le premier rôle de l'égalisation dans les systèmes de communication est d'annuler l'effet du canal sur le signal porteur d'information. Les techniques d'égalisation se divisent en deux grandes catégories: linéaires et non linéaires (selon le type de canal que nous avons). Les techniques linéaires sont généralement les plus simples à mettre en œuvre. Cependant, les performances de ces techniques dégradent remarquablement lorsque les conditions de propagation se dégradent **[1]**, et ne sont donc pas utilisées dans la plupart des applications sans fil. Parmi les techniques d'égalisation non linéaire, l'égaliseur à retour de décision (DFE) est le plus populaire, car il est simple à mettre en œuvre et présente généralement des bons résultats pour les canaux sévèrement dégradés. Toutefois, sur les canaux à faible SNR, le DFE souffre de la propagation d'erreur lorsque les bits sont décodés faussement, conduisant à des performances médiocres. La technique d'égalisation optimale est l'estimation de séquence au sens du maximum de vraisemblance (MLSE, pour Maximum Likelihood Sequence Estimation). Malheureusement, la complexité de cette technique se développe de façon exponentielle avec la longueur du canal, elle est donc inutilisée sur la plupart des canaux d'intérêt. Toutefois, la performance du MLSE est souvent utilisée comme une borne supérieure pour les autres techniques d'égalisation.
Dans cette section, nous présentons les structures d'égaliseurs : l'égaliseur linéaire (LE, pour Linear Equalizer) et l'égaliseur à retour de décision (DFE, pour Decision Feedback Equalizer).

3.3.1 Egaliseur transverse ou linéaire (LE)

L'égaliseur linéaire est le plus simple à mettre en œuvre. Il s'agit de simple filtre numérique linéaire à réponse impulsionnelle finie (FIR) (figure 3.4). Ce filtre non récursif présent l'avantage de ne pas présenter de boucles de contre réaction et donc d'être toujours stable. La figure 3.3, illustre le principe de l'égaliseur linéaire dans un système de réception en communication numérique.

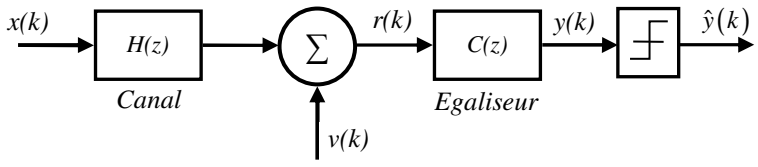


Figure 3.3 : *Egaliseur linéaire (LE).*

La sortie bruitée du canal *r(k)* se présente à l'entrée de l'égaliseur et peut être déterminée par une convolution discrète entre le signal transmit et la réponse impulsionnelle du canal, additionné au bruit comme suit :

$$r(k) = \sum_{i=0}^{N} h_i x(k-i) + v(k) \tag{3.10}$$

Ou d'une façon équivalente par :

$$r[k] = Hx[k] + v[k] \tag{3.11}$$

Où $r[k] = [r_k \quad r_{k-1} \quad \dots \quad r_{k-N}]^T$: est le vecteur des observations bruitées.

$x[k] = [x_k \quad x_{k-1} \quad \dots \quad x_{k-M-N}]^T$: est le vecteur à l'entrée du canal.

$v[k] = [v_k \quad v_{k-1} \quad \dots \quad v_{k-N}]^T$: est le vecteur d'échantillons du bruit.

H : la matrice de convolution du canal de dimension $(N+1)\times(M+N+2)$

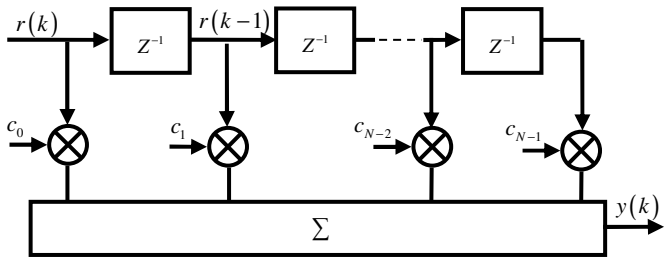


Figure 3.4 : *Structure de l'égaliseur linéaire (LE).*

L'équation (3.11) peut être représentée sous forme matricielle comme suite :

$$\begin{bmatrix} r_k \\ r_{k-1} \\ \vdots \\ r_{k-N} \end{bmatrix} = \begin{bmatrix} h_0 & h_1 & h_2 & \cdots & h_M & 0 & \cdots & 0 \\ 0 & h_0 & h_1 & \cdots & h_{M-1} & h_M & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & h_0 & h_1 & \cdots & h_M \end{bmatrix} \cdot \begin{bmatrix} x_k \\ x_{k-1} \\ \vdots \\ x_{k-M-N} \end{bmatrix} + \begin{bmatrix} v_k \\ v_{k-1} \\ \vdots \\ v_{k-N} \end{bmatrix} \tag{3.12}$$

Le rôle de l'égaliseur est d'utiliser le vecteur d'observations bruitées *r[k]* qui se présente à son entrée, pour estimer la séquence de symboles émise *x[k]*.

L'égaliseur est décrit par ses *N* coefficients représentés par le vecteur: $C = [c_0 \quad c_1 \quad \dots \quad c_{N-1}]^T$

La sortie de l'égaliseur au temps *k* peut être exprimée en fonction de ces coefficients et du signal reçu comme suit :

$$y(k) = \sum_{i=0}^{N-1} c_i r(k-i) \tag{3.13}$$

Où c_i est la réponse impulsionnelle de l'égaliseur, de longueur *N*, *r(k)* est la séquence d'observations, et *y(k)* la sortie de l'égaliseur.

Pour l'optimisation des coefficients de l'égaliseur, Le critère d'optimisation le plus utilisé est celui de l'erreur quadratique moyenne entre la séquence d'entrée (symboles) et la séquence estimée à la sortie de l'égaliseur.

Cette erreur est donnée par :

$$e(k) = y(k) - x(k) \tag{3.14}$$

L'erreur quadratique moyenne (EQM) s'exprime par [23] :

$$J_{MSE} = E\left[\left|e(k)\right|^2\right] \tag{3.15}$$

L'égaliseur transverse reste souvent de qualité médiocre, en particulier en présence d'évanouissements sélectifs (non stationnarités). Ceci est également lié à la structure transverse (pas de pôles) qui limite la capacité de représentation d'une réponse quelconque.

3.3.2 Egalisation à retour de décision (DFE)

L'égaliseur à retour de décision (DFE), est un égaliseur très utilisé pour les canaux présentant de sévères distorsions. L'égaliseur DFE, représenté sur la figure 3.5, est constitué de deux filtres, un filtre direct et un filtre de retour. L'entrée du filtre direct est la séquence des symboles reçus *r(k)*. Le filtre de retour a comme entrée la séquence des décisions sur les symboles préalablement détectés. Il s'agit de prolonger l'idée d'avoir un égaliseur piloté par les décisions, ce qui permet d'éviter une répétition de séquences d'apprentissage, tout en utilisant une structure récursive [23].

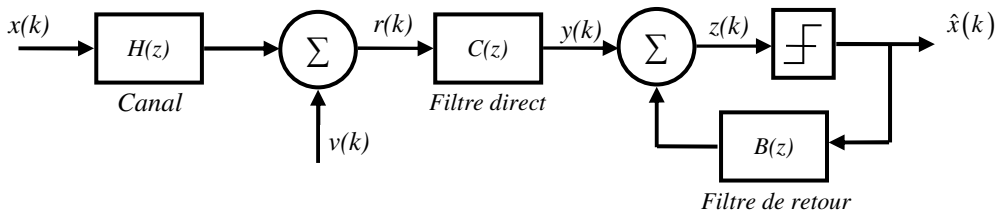


Figure 3.5 *Egaliseur à retour de décision (DFE).*

Le filtre direct est tout simplement un filtre linéaire tel que mentionné précédemment et le filtre de retour est utilisé pour éliminer l'interférence entre symboles de l'estimation courante causée par l'estimation précédente [1]. Le DFE repose sur l'idée que si la valeur du symbole déjà détecté est considérée comme correcte, alors l'interférence entre symboles due à la contribution de ce symbole peut être éliminée. La valeur de ce symbole est soustraite avec une pondération appropriée de la sortie de l'égaliseur. Lorsqu'une décision est incorrecte, une erreur est produite, celle-ci se propage aux autres symboles jusqu'à ce que le futur échantillon compense cette erreur. Ce phénomène de propagation de l'erreur n'est pas catastrophique pour le DFE [2].

A partir de la définition de l'égaliseur DFE, sa sortie est la somme des sorties de la partie directe et de celle de retour (figure 3.6). La sortie de l'égaliseur DFE est donnée par :

$$z(k) = \sum_{i=0}^{N-1} c_i r(k-i) + \sum_{j=1}^{M} b_j \hat{x}(k-j) \tag{3.16}$$

L'équation (3.16) s'écrit en notation vectorielle comme suit :

$$z[k] = c^T r[k] + b^T \hat{x}[k] \tag{3.17}$$

Où $c = [c_0 \quad c_1 \quad \dots \quad c_{N-1}]^T$: est le vecteur des coefficients du filtre directe.

$b = [b_1 \quad b_2 \quad \dots \quad b_M]^T$: est le vecteur des coefficients du filtre retour.

$r[k] = [r_k \quad r_{k-1} \quad \dots \quad r_{k-N+1}]^T$: est le vecteur des observations bruitées.

$\hat{x}[k] = [\hat{x}_{k-1} \quad \hat{x}_{k-2} \quad \dots \quad \hat{x}_{k-M}]^T$: est le vecteur des symboles estimés.

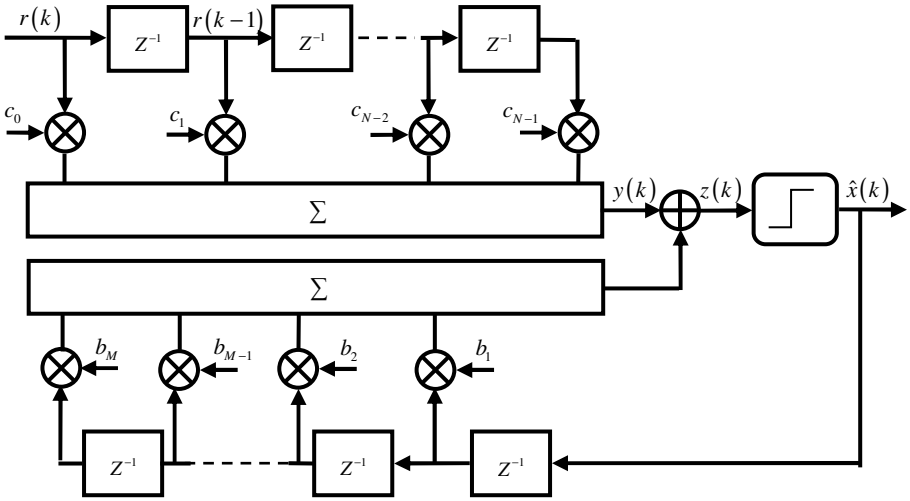


Figure 3.6 : *Structure de l'égaliseur à retour de décision (DFE).*

Les coefficients du filtre direct et ceux du filtre de retour peuvent être ajustés simultanément pour minimiser le critère de l'erreur quadratique moyenne (EQM).

3.4 ESTIMATION DU CANAL

En pratique, les caractéristiques du canal ne sont jamais parfaitement connues, ce qui peut limiter l'efficacité de certaines méthodes d'égalisation, qui supposent un canal stationnaire et requièrent une estimation de la fonction de transfert ou la réponse impulsionnelle du canal. La fonction de transfert de l'égaliseur est créée à partir d'une estimation des paramètres du canal. Dans ce cas, il est possible d'avoir des paramètres d'égalisation variables dans le temps. Il est alors nécessaire de réactualiser régulièrement l'estimation de la fonction de transfert du canal et remettre à jour les paramètres de l'égaliseur. Le récepteur est donc doté d'un algorithme qui adapte les coefficients de l'égaliseur. Ces paramètres peuvent être calculés à partir d'une séquence de données connues du transmetteur et du récepteur appelée séquence d'apprentissage (figure 3.7) ou de façon autodidacte par un retour statistique sur la sortie de l'égaliseur. De ce fait, il est nécessaire de définir des algorithmes adaptatifs qui peuvent suivre l'évolution du canal et qui convergent vers la solution optimale recherchée, dans notre cas celle de la minimisation de l'EQM, il existe deux grandes familles d'algorithmes d'adaptation : l'algorithme des moindres carrés moyens (LMS: Least Mean Square) et l'algorithme des moindres carrés récursifs (RLS: Recursive Least Square).

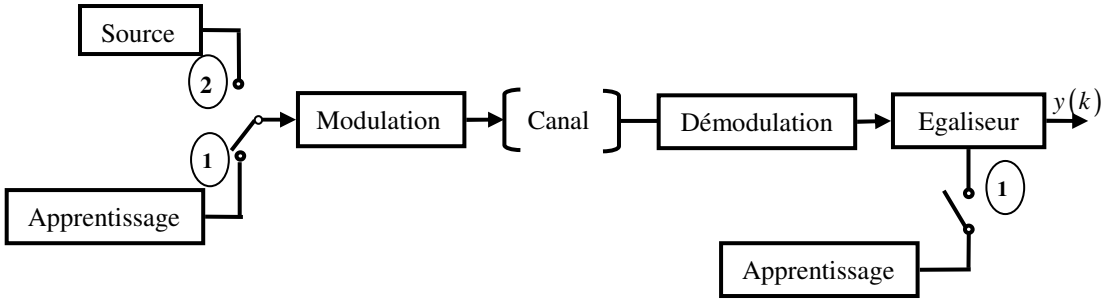


Figure 3.7 : *Egaliseur adaptatif avec période d'apprentissage.*

L'algorithme LMS est largement utilisé pour sa simplicité de mise en œuvre et pour sa stabilité numérique, et l'algorithme RLS est connu par sa bonne rapidité de convergence initiale. Ces deux algorithmes d'adaptation sont commandés par une prédiction d'erreur. Cette erreur est déterminée par la comparaison de la sortie de l'égaliseur avec la valeur désirée à chaque itération de l'algorithme. Pour la mise en œuvre pratique, il est nécessaire de connaitre le symbole désiré **[23]**. Pour cette raison, ces algorithmes d'adaptation utilisent une séquence d'apprentissage (séquence de données de durée limitée connue au récepteur) pour favoriser la convergence.
La mise en œuvre de l'algorithme se fait suivant deux modes opératoires (figure 3.8) :

- Un mode supervisé, où la séquence transmis (symboles) est connue (apprentissage). Le calcul de *y(k)* ne sert alors qu'à adapter le filtre, jusqu'à convergence. La période durant laquelle l'algorithme utilise les symboles de la séquence d'apprentissage s'appelle période d'apprentissage. Après cette période d'apprentissage l'égaliseur passe au mode opérationnel.
- Un mode opérationnel : la sortie de l'égaliseur *y(k)* sert alors à estimer : $\hat{x}(k) = \mathrm{dec}(y(k)) = \hat{y}(k)$, où '*dec*' indique que l'on prend la décision sur *y(k)*. L'erreur est alors maintenant calculée à partir des décisions. Durant cette période, les données décidées présentent un taux d'erreur faible si la période d'apprentissage était suffisante pour la convergence des coefficients vers la solution recherchée.

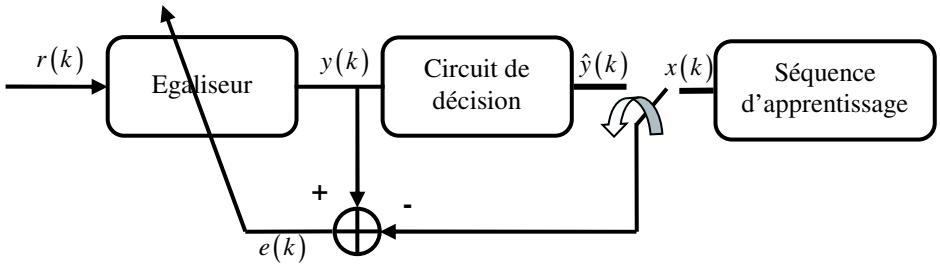


Figure 3.8 : *Egaliseur adaptatif piloté par les décisions.*

3.4.1 Algorithme des moindres carrés moyens (LMS)

Notre but c'est de déterminer les coefficients optimums de l'égaliseur qui minimisent un certain critère d'erreur. Quand un symbole est transmis le mieux qu'on puisse espérer c'est que la sortie égalisée soit une version retardée du symbole considéré, autrement dit : *y(k)=x(k-d)*, *d* étant le retard. Un critère standard pour déterminer les coefficients optimums de l'égaliseur est la minimisation de l'erreur quadratique moyenne (MSE).
L'algorithme LMS est un choix populaire dans beaucoup d'applications exigeant le filtrage adaptatif. En outre, il y a plusieurs variantes de l'algorithme qui peuvent être employées spécifiquement afin de résoudre différents types de problèmes qui sont inhérents à certaines applications (DLMS : Delayed LMS, FXLMS : Filtered-X LMS, ADJLMS : Adjoint LMS, NLMS : Normalized LMS, BLMS : Block LMS, BLMS FFT : Block LMS (FFT)…). La version de base du LMS est un cas spécial du filtre adaptatif du gradient descendant (steepest descent) bien connu **[17]**. Le but de cette technique est de réduire au minimum une fonction de cout quadratique en mettant à jour itérativement des poids de sorte qu'ils convergent à la solution optimale.

3.4.1.1 Adaptation de l'égaliseur linéaire (LE)

Soit une fonction de coût J_{MSE} qu'on cherche à minimiser et qui dépond des coefficients de l'égaliseur. Cette fonction est minimale lorsque son gradient par rapport au vecteur des coefficients est nul.

$$J_{MSE} = E\left[\left|e(k)\right|^2\right] = E\left[\left|y(k) - x(k-d)\right|^2\right] \tag{3.18}$$

Avec : $$y(k) = \sum_{i=0}^{N-1} c_i r(k-i) = c^T r(k) \tag{3.19}$$

Où : $c^T = \begin{bmatrix} c_0 & c_1 & \dots & c_{N-1} \end{bmatrix}$: est le vecteur des coefficients de l'égaliseur.

$r(k) = \begin{bmatrix} r_k & r_{k-1} & \dots & r_{k-N+1} \end{bmatrix}^T$: est le vecteur des observations bruitées.

En considérant l'équation (3.19), l'expression de l'équation (3.18) devient :

$$J_{MSE} = E\left[\left|c^T r(k) - x(k-d)\right|^2\right] \tag{3.20}$$

Le gradient de J_{MSE} par rapport aux coefficients de l'égaliseur est donné par :

$$\frac{\partial J_{MSE}}{\partial c} = 2E\left[r(k)\left(c^T r(k) - x(k-d)\right)\right] = 0 \tag{3.21}$$

Soit : $$E\left[r(k)r(k)^T\right]c = E\left[r(k)x(k-d)\right] \tag{3.22}$$

Sachant que $R_{yy} = E\left[r(k)r(k)^T\right]$ est la matrice de corrélation de *r(k)* et $\rho_{rx} = E\left[r(k)x(k-d)\right]$ est le vecteur d'intercorrélation entre *r(k)* et *x(k-d)*.

En remplaçant chaque expression par sa valeur, on obtient :

$$R_{yy}c = \rho_{rx} \tag{3.23}$$

La solution optimale est obtenue en annulant l'équation (3.23). Les coefficients de l'égaliseur sont déterminés comme suit :

$$c = R_{yy}^{-1}\rho_{rx} \tag{3.24}$$

Cette solution nécessite l'inversion de la matrice corrélation de R_{yy}, comme alternative une procédure itérative qui évite l'inversion directe de la matrice peut être utilisée. La plus simple procédure et la plus populaire est l'utilisation de l'algorithme des moindres carrées LMS qui est très utilisé pour l'adaptation des égaliseurs **[21, 83, 2]**. L'algorithme LMS permet une adaptation itérative des coefficients de l'égaliseur en minimisant le critère de l'erreur quadratique moyenne.

Les coefficients de l'égaliseur sont donnés par :

$$c(k) = c(k-1) - \mu\frac{\partial J_{MSE}}{\partial c} = c(k-1) - \mu\left(cR_{yy} - \rho_{rx}\right) \tag{3.25}$$

Où μ est une constante positive (pas d'adaptation).

La matrice de corrélation R_{yy} et le vecteur d'intercorrélation ρ_{rx} sont des valeurs d'espérance qui peuvent être rapprochés par leurs évaluations instantanées soient : $R_{yy} = \hat{R}_{yy} = r(k)\,r(k)^T$ et $\rho_{rx} = \hat{\rho}_{rx} = r(k)\,x(k-d)$. En tenant compte de ces expressions et des équations (3.25) et (3.19) on obtient :

$$c(k) = c(k-1) - \mu r(k)\left[y(k) - x(k-d)\right] \tag{3.26}$$

L'algorithme devient alors simplement :

$$\begin{cases} c(k) = c(k-1) + \mu r(k)\,\varepsilon(k) \\ \varepsilon(k) = x(k-d) - c(k-1)^T\, r(k) \end{cases} \tag{3.27}$$

On notera que $c(k-1)^T\, r(k)$ est simplement la sortie du filtre adaptatif à l'instant *k*. Cet algorithme permet donc, à chaque instant, de remettre à jour les coefficients du filtre, proportionnellement à l'erreur d'estimation *ε(k)*. En cas de variations des caractéristiques du canal, l'égaliseur sera donc capable de s'adapter à celles-ci, et ce d'autant plus rapidement que μ est grand. En contrepartie, plus μ est grand, plus les variations liées au bruit d'observation induiront une variabilité sur les estimées *c(k)***[23]**. En contre partie, le choix d'un pas d'adaptation μ petit a pour effet de ralentir la vitesse de convergence, et de ce fait le choix de la valeur du pas doit être un compromis entre la vitesse de convergence et la performance de l'égaliseur.

Quand l'égaliseur est en mode opérationnel, le symbole décidé $\hat{x}(k-1) = \mathrm{dec}\left[y(k)\right]$ est utilisé à la place de *x(k-d)*.

3.4.1.2 Adaptation de l'égaliseur DFE

L'algorithme des moindre carrée LMS est aussi utilisé pour adapter l'égaliseur DFE.

Posons ainsi :

$$\begin{cases} a(k) = \left[c_0(k)\ \ c_1(k) .... c_{N-1}(k)\ \ b_1(k)\ \ b_2(k) .... b_M(k) \right]^T = \left[c(k)^T\ b(k)^T \right]^T \\ s(k) = \left[r(k)\ \ r(k-1) .... r(k-N+1)\ \ \hat{x}(k-1) .... \hat{x}(k-M) \right] = \left[r(k)^T\ \hat{x}(k)^T \right]^T \end{cases} \quad (3.28)$$

Où : *a(k)* est le vecteur de la combinaison des coefficients de la partie directe et celle de retour.

s(k) est le vecteur de la combinaison des échantillons à l'entrée des deux filtres.

A l'aide de ces notations, l'erreur $\varepsilon(k)$ s'écrit $\varepsilon(k) = x(k-d) - z(k)$, en mode apprentissage, et $\varepsilon(k) = \hat{x}(k) - z(k)$, en mode opérationnel (voir figure 3.5).

L'algorithme LMS s'écrit alors :

$$a(k) = a(k-1) + \mu s(k) \varepsilon(k) \quad (3.29)$$

Avec $z(k) = c(k)^T\ r(k) + b(k)^T\ \hat{x}(k)$ (3.30)

Ou plus explicitement, les coefficients des filtres direct et de retour sont donnés par :

$$\begin{cases} c(k) = c(k-1) + \mu r(k) \varepsilon(k) \\ b(k) = b(k-1) + \mu \hat{x}(k) \varepsilon(k) \end{cases} \quad (3.31)$$

Lors de la mise en œuvre, on utilisera une période d'apprentissage, au cours de laquelle on prendra les symboles connus pour $\hat{x}(k)$, $\hat{x}(k) = x(k-d)$, puis on basculera en mode opérationnel, en remplaçant la séquence d'apprentissage par les décision, $\hat{x}(k) = \mathrm{dec}(z(k))$.

3.4.2 Algorithme des moindres carrés récursifs (RLS)

L'algorithme RLS est connu pour sa rapidité de convergence. Dans cet algorithme, le critère statistique de l'erreur quadratique moyenne est remplacé par un critère de moindres carrés pondérés, qui est un critère déterministe minimisant la somme pondérée des erreurs quadratique **[21]**.
Ce critère est donnée par :

$$J(k) = \sum_{i=0}^{k} \lambda^{k-i} \left| e(i,k) \right|^2 \quad (3.32)$$

Où : $e(i,k) = x(i) - r(i)^T\ c(k)$ (3.33)

e(i,k) est l'erreur a posteriori qui consiste en la différence entre le signal désiré et la sortie de l'égaliseur, en utilisant les coefficients les plus récents. Le paramètre λ est un facteur de pondération qui prend toujours une valeur positive ($0 < \lambda < 1$). Ce facteur est aussi appelé facteur d'oubli car il sert à oublier les données qui correspondent à un passé distant. Le paramètre λ^{k-i} met en évidence les échantillons d'erreurs les plus

récents (où $i \approx k$). Lorsque $\lambda = 1$, la fonction de coût donnée par l'équation (3.32) est simplement la somme des erreurs quadratiques. Si $\lambda < 1$, les erreurs passées sont pondérées avec un poids (facteur d'oubli) qui décroît exponentiellement.

Le gradient de *J(k)* par rapport aux coefficients de l'égaliseur est donné par :

$$\frac{\partial J(k)}{\partial c(k)} = -2\sum_{i=0}^{k} \lambda^{k-i} r(i)\left[x(i) - r(i)^T c(k) \right] \tag{3.34}$$

L'algorithme RLS changent les coefficients du filtre pour que l'erreur soit minimisée, par les relations suivantes :

$$\left[\sum_{i=0}^{k} \lambda^{k-i} r(i) r(i)^T \right] c(k) = \sum_{i=0}^{k} \lambda^{k-i} r(i) x(i) \tag{3.35}$$

On tire de cela :

$$R_D(k) c(k) = P_D(k) \tag{2.36}$$

et :

$$c(k) = R_D^{-1}(k) P_D(k) \tag{3.37}$$

Où $R_D(k) = \left[\sum_{i=0}^{k} \lambda^{k-i} r(i) r(i)^T \right]$ et $P_D(k) = \sum_{i=0}^{k} \lambda^{k-i} r(i) x(i)$ Sont respectivement, la matrice d'autocorrélation du signal d'entrée et le vecteur d'inter-corrélation entre le signal d'entrée et le signal désiré.

L'équation (3.35) est réécrite sous la forme suivante :

$$\left[\sum_{i=0}^{k} \lambda^{k-i} r(i) r(i)^T \right] c(k) = \lambda \left[\sum_{i=0}^{k-1} \lambda^{k-i-1} r(i) x(i) \right] + r(k) x(k) \tag{3.38}$$

En considérant que $R_D(k-1) c(k-1) = p_D(k-1)$, l'équation (3.38) se réécrite :

$$\begin{aligned} \left[\sum_{i=0}^{k} \lambda^{k-i} r(i) r(i)^T \right] c(k) &= \lambda P_D(k-1) + r(k) x(k) \\ &= \lambda R_D(k-1) c(k-1) + r(k) x(k) \\ &= \left[\sum_{i=0}^{k} \lambda^{k-i} r(i) r(i)^T - r(i) r(i)^T \right] c(k-1) + r(k) x(k) \end{aligned} \tag{3.39}$$

Soit : $R_D(k) c(k) = R_D(k) c(k-1) + \left[x(k) - r(k)^T c(k-1) \right] r(k)$ (3.40)

D'après l'équation (3.40), on définit l'erreur a priori comme suit :

$$e(k) = \left[x(k) - r(k)^T c(k-1) \right] \tag{3.41}$$

En remplaçant l'erreur dans l'équation (3.40), on obtient l'équation de mise à jour des coefficients de l'égaliseur :

$$c(k) = c(k-1) + e(k) R_D^{-1}(k) r(k) \tag{3.42}$$

Où $R_D^{-1}(k)$ est l'inverse de la matrice $R_D(k)$ calculée en utilisant le lemme d'inversion d'une matrice **[85]**. Lemme d'inversion d'une matrice est calculé comme suit :

Si : $A = B^{-1} + C^{-1} D C^T$, (3.43)

Alors : $A^{-1} = B - BC\left(D + C^T BC\right)^{-1} C^T B$, (3.44)

On peut utiliser le lemme d'inversion pour calculer l'inversion de $R_D(k)$ en posant :

$$\begin{aligned} &A = R_D(k), \\ &B^{-1} = \lambda R_D(k-1), \\ &C = r(k), \\ &D = 1. \end{aligned} \tag{3.45}$$

On obtient donc l'équation récursive suivante pour l'inversion de la matrice de corrélation :

$$R_D^{-1}(k) = \frac{1}{\lambda}\left[R_D^{-1}(k-1) - \frac{R_D^{-1}(k-1) r(k) r(k)^T R_D^{-1}(k-1)}{\lambda + r(k)^T R_D^{-1}(k-1) r(k)} \right] \tag{3.46}$$

En posant : $\Phi(k) = R_D^{-1}(k-1) r(k)$, l'équation (3.46) s'écrit :

$$R_D^{-1}(k) = \frac{1}{\lambda}\left[R_D^{-1}(k-1) - \frac{\Phi(k)\Phi(k)^T}{\lambda + \Phi(k)^T r(k)} \right] \tag{3.47}$$

Les conditions initiales de cet algorithme sont : $c(0), R_D^{-1}(0) = \delta^{-1} I$, I matrice de corrélation et δ une faible constante positive.

L'algorithme de type RLS possède une vitesse de convergence élevée qui est indépendante de la dispersion des valeurs propres de la matrice de corrélation d'entrée. Cet algorithme est également très utile dans les applications où l'environnement varie lentement. Le prix de tous ces avantages est une augmentation considérable de la complexité des calculs et certains problèmes de stabilité, qui ne sont pas aussi critique dans l'algorithme LMS **[86, 87]**.

3.5 COMPARAISON DES PRINCIPALES STRUCTURES D'EGALISEURS

Afin d'illustrer les performances et la mise en œuvre des algorithmes décrits dans ce chapitre, nous allons examiner leurs performances avec l'égaliseur linéaire (LE) et l'égaliseur à retour de décision (DFE), pour une transmission numérique en modulation de phase à deux états (BPSK), et avec deux types de canaux : un canal linéaire à phase non minimale et un canal non linéaire.

3.5.1 Canal linéaire à phase non minimale

Le modèle du canal discret équivalent utilisé pour tester les performances des égaliseurs est un canal à phase non minimale, il est représenté par la fonction de transfert en Z définie par :

$$H(z) = 0.3482 + 0.8704z^{-1} + 0.3482z^{-2} \tag{3.48}$$

Les coefficients du canal sont réels au nombre de trois. C'est un canal représentatif de la majorité des canaux rencontrés en pratique, il est très utilisé pour la simulation des égaliseurs **[9, 21, 88-90, 92-94]**. Les caractéristiques de ce canal sont représentées à la figure 3.9. La partie (a) représente la réponse en amplitude du canal qui comporte des zones où le signal sera amplifié (amplitude > 0) et des zones où le signal sera affaibli (amplitude < 0). Cette réponse présente un évanouissement de profondeur moyenne. La partie (b) représente la réponse en phase qui est quasi-linéaire. La partie (c) donne la représentation dans le plan Z et montre que le canal possède deux zéros, dont l'un est à l'intérieur au cercle unité et l'autre est à l'extérieur.

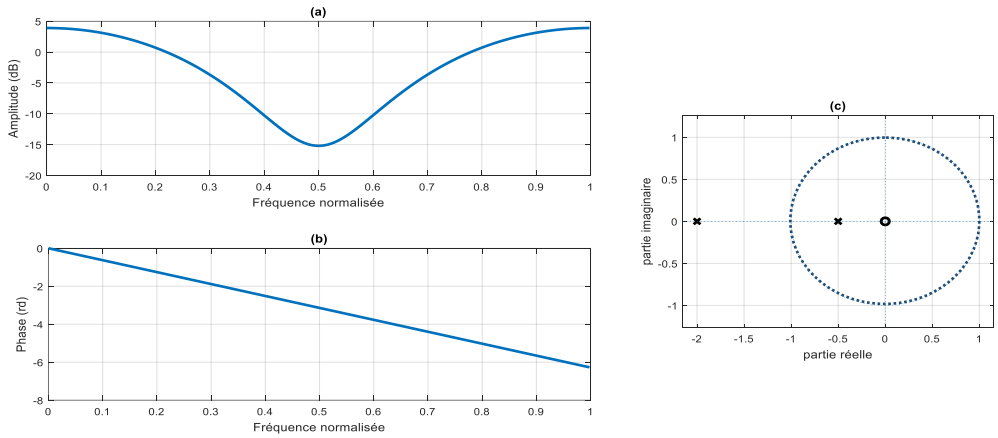


Figure 3.9 : *Caractéristiques du canal H(z).* (a) *Réponse en amplitude*, (b) *Réponse en phase*, (c) *Les zéros de canal*.

3.5.2 Canal non linéaire

Les canaux non linéaires sont communément rencontrés dans la pratique, les distorsions non linéaires peuvent survenir en raison des phénomènes de saturation d'amplificateurs, ou provenir des processus de modulation **[21]**. L'effet non linéaire du canal introduit des distorsions non linéaires d'amplitude et de phase. Nous considérons un canal non linéaire composé d'un canal linéaire à phase non minimale qui est décrit au paragraphe précédent, suivi d'un non linéarité statique de Volterra, ce modèle est représenté par la figure 3.10.

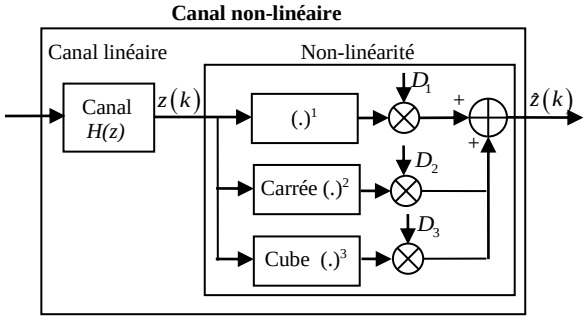


Figure 3.10 : *Canal non linéaire.*

La sortie du canal linéaire à l'instant k est exprimée par l'équation suivante:

$$z(k)=\sum_{i=0}^{N-1} h_i x(k-i) \tag{3.49}$$

Où $h_i\,(i=0,1,.....N-1)$ est la réponse impulsionnelle du canal.

La sortie de ce canal est ensuite passée dans une fonction non linéaire dont la non linéarité est décrite par :

$$\hat{z}(k)=D_1 z(k)+D_2 z^2(k)+D_3 z^3(k) \tag{3.50}$$

Où D_1, D_2, D_3 sont les paramètres du canal non linéaire.

3.5.3 Comparaisons des égaliseurs LE-LMS et DFE-LMS

La figure 3.11 présente une comparaison des courbes de convergence de l'erreur quadratique moyenne (MSE) des égaliseurs DFE et LE dont l'adaptation est faite à l'aide de l'algorithme LMS, et pour un rapport signal sur bruit de 20 dB. Les courbes de l'erreur quadratique sont calculées en moyennant 600 simulations indépendantes, chacune avec des symboles aléatoires différents et des paramètres initiaux aléatoires.
L'égaliseur DFE contient 5 coefficients dans sa section directe et 1 coefficient dans sa section récursive (5,1). L'égaliseur LE comporte 6 coefficients. Le pas d'adaptation μ = 0.035.

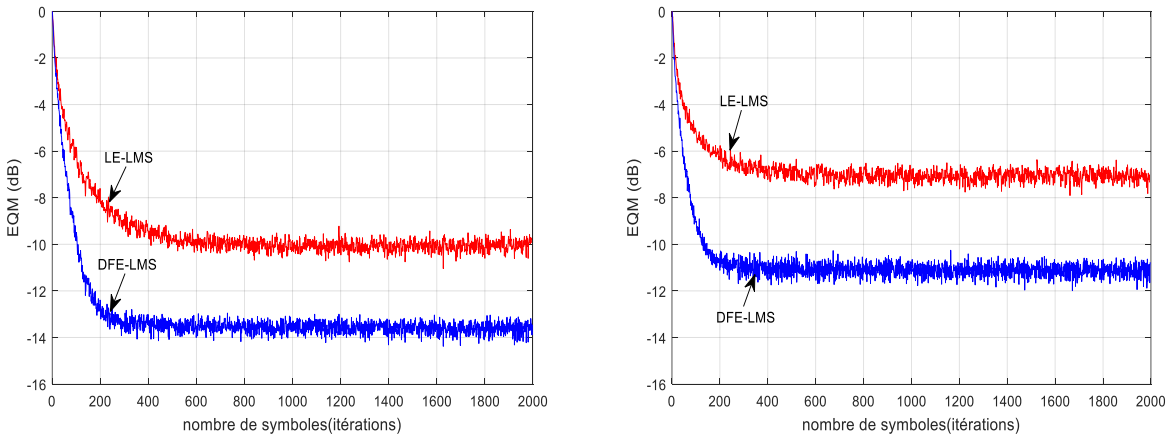
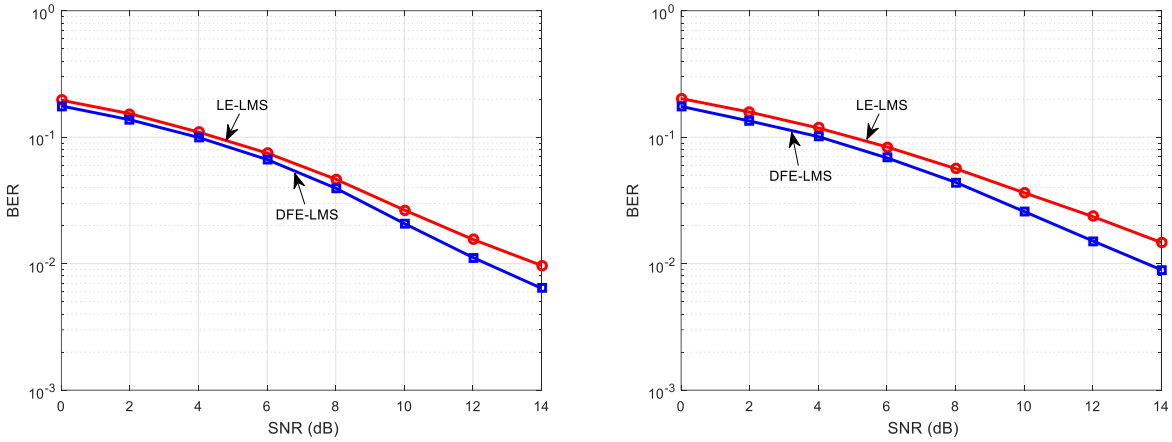


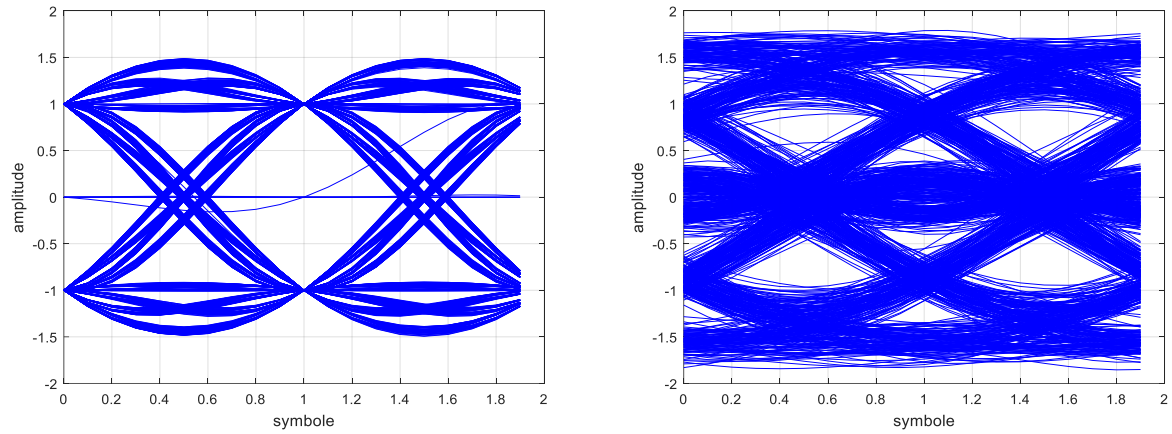
Figure 3.11 *Courbes de convergence de l'EQM des égaliseurs LE-LMS et DFE-LMS*
(a) : *Canal linéaire,* **(b) :** *Canal non-linéaire*



(a) **(b)**

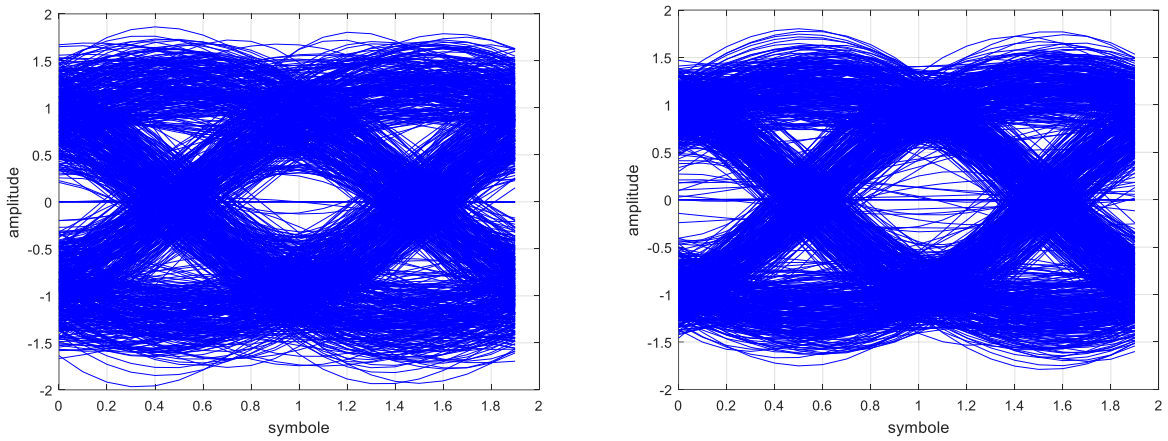
Figure 3.12 *Courbes BER des égaliseurs LE-LMS et DFE-LMS*
(a) : *Canal linéaire,* **(b) :** *Canal non-linéaire*

La figure 3.11 (a), représente les courbes de convergences des deux égaliseurs pour un canal linéaire. L'égaliseur LE converge vers un état stable de -10 dB après 500 itérations, alors que l'égaliseur DFE atteint le niveau de -10 dB après environ 100 itérations seulement et passe à son état stable de -14 dB. L'égaliseur DFE apporte un gain en temps de convergence par rapport à l'égaliseur LE. La figure 3.11(b) montre les performances des égaliseurs à travers le canal non linéaire avec : D_1 = 1, D_2=0.1, D_3=0.1 **[30]**. Les performances des deux égaliseurs se dégradent sous l'effet de la non-linéarité du canal. La figure 3.12 illustre les courbes du BER (Bit Error Rate) des deux types d'égaliseurs. Ces courbes sont obtenues en prenant en moyenne 100 simulations indépendantes de longueur 2000 symboles. Les 1000 premiers symboles sont utilisés pour l'apprentissage et les autres sont utilisées pour le test. Ces courbes montrent que l'égaliseur DFE opère encore mieux et présente un BER plus faible que l'égaliseur LE.



(a) **(b)**

Figure 2.13 *Diagramme de l'œil du signal BPSK*
(a) : *Signal transmis,* **(b) :** *Signal reçu (canal linéaire)*



(a) **(b)**

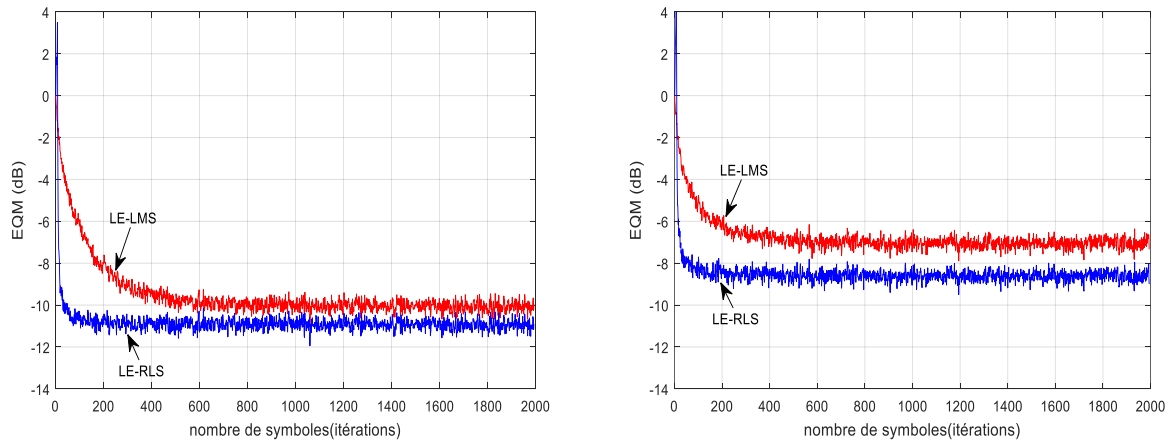
Figure 3.14 *Diagramme de l'œil du signal en sortie de l'égaliseur*
(a) : *avec l'égaliseur LE-LMS,* **(b) :** *avec l'égaliseur DFE-LMS*

La figure 3.13 représente le diagramme de l'œil du signal BPSK à l'entrée et à la sortie du canal, la figure 3.13(b) montre que l'œil est fermé et les erreurs de décisions sont fortement probables.

La figure 3.14 représente le diagramme de l'œil du signal à la sortie des égaliseurs LE et DFE. Le signal à la sortie de l'égaliseur DFE montre une amélioration du diagramme de l'œil par rapport à celui de l'égaliseur LE, et l'œil devient plus ouvert.

3.5.4 Comparaisons des égaliseurs LE-LMS et LE-RLS

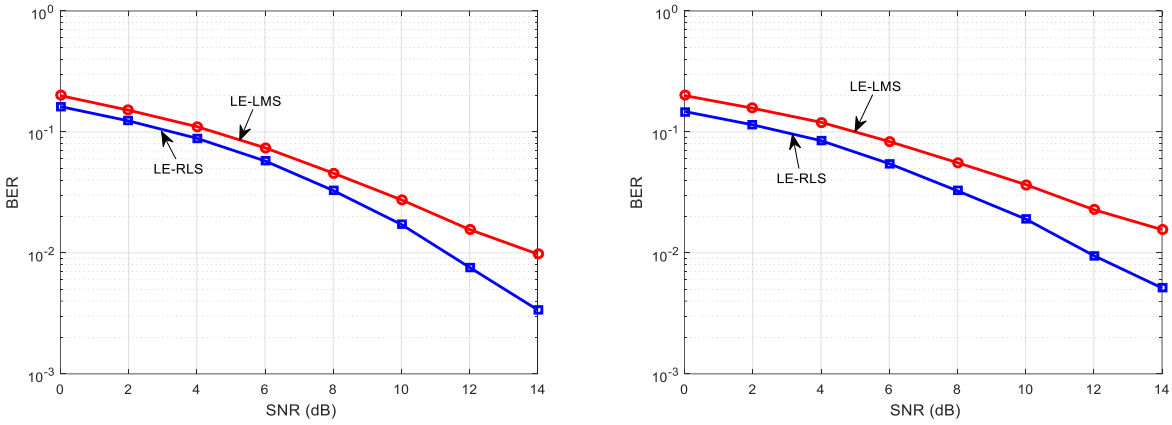
La figure 3.15 présente une comparaison des courbes de convergence de l'égaliseur linéaire (LT) pour l'algorithme LMS (Least Mean Square) avec un pas d'adaptation μ = 0.035, et l'algorithme RLS (Recursive Least Square) avec un facteur d'oubli λ = 1. L'égaliseur linéaire (LE) est de l'ordre de six coefficients. L'EQM atteint sa valeur optimale après environ 500 itérations pour l'égaliseur LE-LMS, et 100 itérations pour l'égaliseur LE-RLS. La valeur optimale de l'égaliseur LE pour les deux canaux : linéaire et non-linéaire est de -10 et -7dB respectivement avec l'algorithme LMS, est de -11 et -8dB respectivement avec l'algorithme RLS. L'égaliseur LE-RLS converge plus rapidement que l'égaliseur LE-LMS.



(a) **(b)**

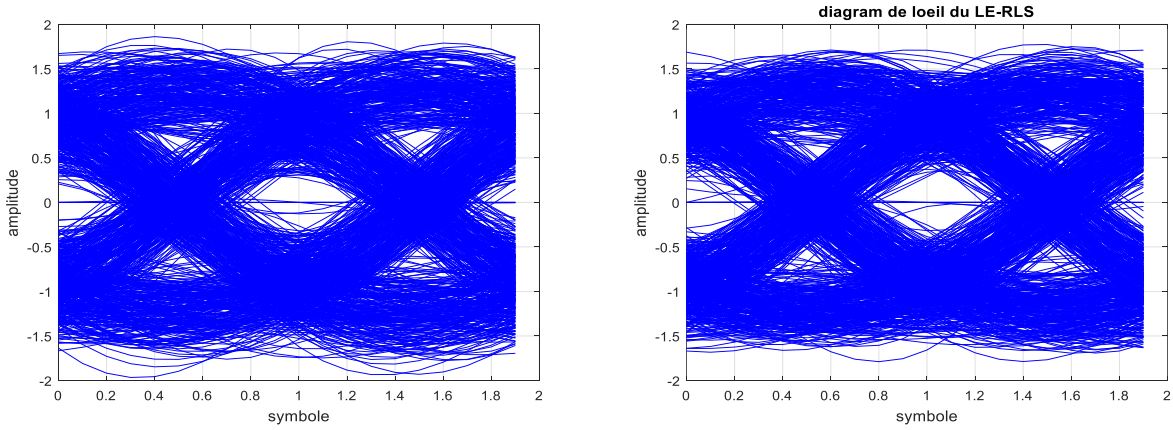
Figure 3.15 *Courbes de convergence de l'EQM des égaliseurs LE-LMS et LE-RLS*
(a) : *Canal linéaire,* **(b) :** *Canal non-linéaire*

La figure 3.16 illustre les courbes du BER (Bit Error Rate) des deux types d'égaliseurs, ces courbes montrent que l'égaliseur LE avec l'algorithme RLS donne un BER plus faible que l'égaliseur LE avec l'algorithme LMS. Ces courbes montrent aussi que l'égaliseur LE-RLS donne un BER plus faible que l'égaliseur DFE-LMS (figure 3.12).



(a) **(b)**

Figure 3.16 *Courbes BER des égaliseurs LE-LMS et LE-RLS*
(a) : *Canal linéaire,* **(b) :** *Canal non-linéaire*



(a) **(b)**

Figure 3.17 *Diagramme de l'œil du signal en sortie de l'égaliseur*
(a) : *avec l'égaliseur LE-LMS,* **(b) :** *avec l'égaliseur LE-RLS*

La figure 3.17 montre le diagramme de l'œil du signal à la sortie des égaliseurs LE-LMS et LE-RLS, ces diagrammes indique que des erreurs de décision peuvent survenir, et que ces deux égaliseurs sont sous optimales et donnent des résultats qui ne sont satisfaisants.

3.6 CONCLUSION

Dans ce chapitre nous avons discuté le contexte d'égalisation des canaux de communication numériques et nous avons également mis en évidence certaines structures d'égaliseurs les plus communes : l'égaliseur LE et l'égaliseur DFE adaptés à l'aide des algorithmes LMS et RLS. Les méthodes classiques ont été présentées dans le but de mettre en évidence leurs performances médiocres pour un canal à phase non minimale, les niveaux MSE atteints sont sous optimales. Les égaliseurs conventionnels présentent une incapacité à égaliser les canaux non linéaires à faible SNR, d'où la nécessité des architectures présentant un traitement non linéaire pour améliorer les performances. Ces architectures feront l'objet du chapitre suivant.

CHAPITRE 4

EGALISATION A BASE DES RESEAUX DE NEURONES ARTIFICIELS

4.1 INTRODUCTION

L'essor des réseaux de neurones dans divers domaines actuels est certainement dû à leurs grandes capacités de calcul et à leurs hautes habilités d'apprentissage. De plus, l'estimation de leurs paramètres est indépendante de la complexité du problème traité ce qui leur permet d'être bien adaptés aux problèmes actuels qui ne cessent d'être de plus en plus complexes **[95]**. L'égalisation non linéaire à base des réseaux de neurones offre une solution plus avancée au problème d'égalisation avec des améliorations dans les performances et dans les types des canaux qui peuvent être égalisés par rapport aux techniques conventionnelles qui aboutissent à des performances sous optimales. Les architectures des égaliseurs LE et DFE peuvent profiter de la mise en œuvre de ces structures en montrant une amélioration des performances **[30, 39]**. Trois types particuliers des réseaux de neurones sont investis dans l'égalisation des canaux de communication : le perceptron multicouches (MLP: Multi Layer Perceptron), la fonction à base radiale (RBF: Radial Basis Fonction) et le réseau récurent (RNN : Recurrent Neural Network). Nous focaliserons notre étude principalement sur le perceptron multicouches MLP et la fonction à base radiale. Nous commençons d'abord par donner un aperçu général sur les réseaux de neurones.

4.1.1 Le neurone biologique

Les neurones biologiques sont des cellules vivantes constituant le système nerveux. Ce sont des cellules spécialisées dans le traitement des signaux électriques et la transmission du message nerveux. Les neurones sont reliés entre eux par des liaisons appelées axones. Un neurone est doté de ramifications que l'on nomme les dendrites par lesquelles transite l'information (sous la forme de courants électriques) venue de l'extérieur vers le corps cellulaire. Le neurone traite cette information et renvoie le résultat au travers de son axone. Ce signal émis par le neurone peut ensuite être transmis, au travers d'une synapse, à un autre neurone (figure 4.1) **[96]**.

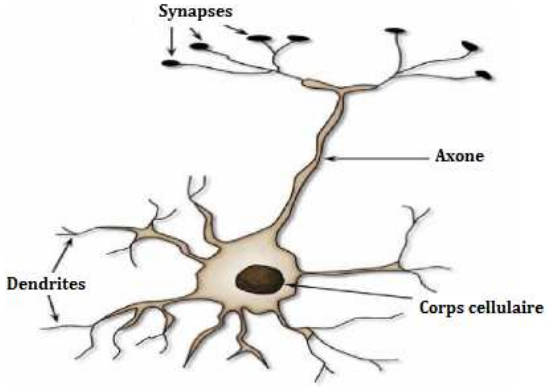


Figure 4.1 *Le neurone biologique et ses principaux composants.*

La figure 4.1 montre les éléments essentiels constituant le neurone biologique. Un neurone biologique est composé de quatre parties essentielles :

- **Le corps cellulaire** effectue les transformations biochimiques nécessaires à la vie du neurone. Il contient le noyau du neurone, sa taille est de quelques microns de diamètre. C'est le centre de l'influx nerveux qui représente l'état d'activité du neurone.

- **Les dendrites** forment une arborescence autour du corps cellulaire et permettent au neurone de capter les signaux qui parviennent de l'extérieur. Ils constituent les entrées principales du neurone. Leur longueur est de quelques dizaines de microns.

- **L'axone** est une fibre nerveuse qui transporte les signaux émis par le neurone, il se ramifie à son extrémité ou il communique avec les autres neurones à travers des synapses. Sa taille peut varier de quelques millimètres à plusieurs mètres.

- **Les synapses**, ce sont des jonctions entre deux neurones et qui sont essentielles dans le fonctionnement du système nerveux.

L'influx nerveux se propage toujours de la dendrite vers le corps cellulaire et de celui-ci vers l'axone. Chaque neurone reçoit des signaux excitateurs ou inhibiteurs par ses dendrites. Ces signaux pondérés sont combinés dans le corps cellulaire. Le potentiel résultant est comparé au seuil interne, s'il est supérieur à celui-ci, le neurone provoque l'émission dans l'axone d'un train d'impulsions vers les synapses, dans le cas contraire il reste inactif.

4.1.2 Le neurone formel

Les réseaux de neurones formels sont à l'origine d'une tentative de modélisation mathématique dont la structure s'inspire de celle du système nerveux. Le premier modèle mathématique du neurone biologique est proposé par Warren McCulloch et Walter Pitts en 1943 **[8]**. En s'appuyant sur les propriétés des neurones biologiques connues à cette époque, issues d'observations neurophysiologiques et anatomiques. Ils présentent un modèle assez simple pour les réseaux de neurones formels qui peuvent réaliser des fonctions logiques, arithmétique et symbolique complexes (tout au moins au niveau théorique) **[97]**. Il s'agit d'un neurone binaire, c'est-à-dire dont la sortie vaut 0 ou 1. Pour calculer cette sortie, le neurone effectue une somme pondérée de ses entrées (qui, en tant que sorties d'autres neurones formels, valent aussi 0 ou 1) puis applique une fonction d'activation à seuil : si la somme pondérée dépasse une certaine valeur, la sortie du neurone est 1, sinon elle vaut 0. Ce modèle n'a pas possédé une règle d'apprentissage jusqu'à 1949 où Donald. Hebb a proposé une simple règle d'apprentissage dans son ouvrage : The Organization of Behaviour **[98]**. Le premier processus artificiel capable d'apprendre par expérience a été proposé par Franck Rosenblatt **[99]** en 1957, c'était le perceptron. Mais, Marvin Lee Minsky et Seymour Papert **[100]** ont publié un travail énonçant les limitations de ce dernier, et principalement l'impossibilité de traiter des

problèmes non linéaires. Cette conclusion pessimiste a étendu tous les modèles des réseaux de neurones et a abaissé leurs intérêts pendant deux décennies. Il a fallu attendre l'apparition du perceptron multicouche, introduit par Rumelhart et al **[101]** en 1986, pour donner renaissance aux réseaux de neurones. Cette découverte a été une vraie révolution qui a permis aux réseaux de neurones de connaitre un essor considérable **[95]**.
La figure 4.2 montre le modèle du neurone formel.

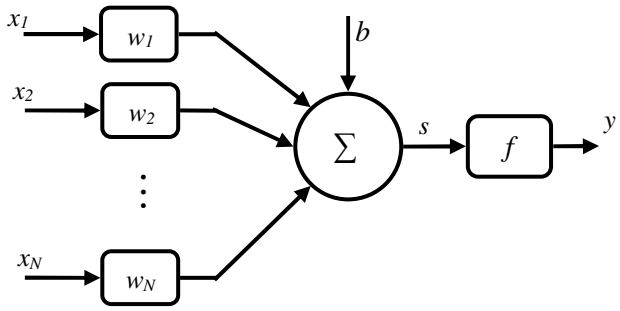


Figure 4.2 *Modèle du neurone formel.*

La valeur de la sortie (y) résulte de la somme pondérée des entrées (x_i) pondérées par des coefficients (w_i), plus un biais (b) et du calcul d'une fonction d'activation (f) de cette somme. La formalisation mathématique de son comportement est donnée par :

$$y = f(s) = f\left(b + \sum_{i=1}^{N} w_i x_i\right) \tag{4.1}$$

La modélisation consiste donc à mettre en œuvre un système de réseaux neuronaux sous un aspect non pas biologique mais artificiel, cela suppose que d'après le principe biologique on aura une correspondance pour chaque élément composant le neurone biologique, donc une modélisation pour chacun d'entre eux. On pourra résumer cette modélisation par le tableau 4.1, qui nous permettra de voir clairement la transition entre le neurone biologique et le neurone formel.

Tableau 4.1 *Analogie entre le neurone biologique et le neurone formel*

Neurone biologique	Neurone formel
Synapses	Poids de connexions
Axones	Signal de sortie
Dendrite	Signal d'entrée
Noyau	Fonction d'activation

Il est à noter que cette modélisation simplifiée est loin de fournir une explication exacte concernant la complexité de fonctionnement des neurones biologique. Malgré cela, cette formalisation permet d'étudier les connexions entre ces neurones dans d'autres processus plus complexes comportant plusieurs neurones interconnectés **[95]**.

4.2 ÉGALSATION PAR LES RÉSEAUX DE NEURONES

Un réseau de neurones artificiels est un ensemble de neurones associés en couches et fonctionnant en parallèle. Ce réseau neuronal possède la capacité de stocker de la connaissance empirique et de la rendre disponible à l'usage **[21]**. Les habiletés de traitement et la connaissance du réseau vont être stockées dans les poids synaptiques, obtenus par des processus d'adaptation ou d'apprentissage. La structure des connexions entre les différents neurones détermine la topologie du réseau.

Les réseaux de neurones sont capables d'apporter des solutions à des problèmes complexes en communications numériques grâce à leur traitement non linéaire, leur architecture parallèlement distribuée, leur auto-organisation, leur capacité d'apprentissage et de généralisation et leur implantation efficace. Parmi les applications des réseaux de neurones aux communications numériques on trouve, l'identification, l'égalisation, le codage et le décodage, la quantification vectorielle, le traitement d'images, le filtrage non linéaire, les techniques d'étalement de spectre, etc. Le point clé pour une utilisation efficace des réseaux de neurones est de trouver une architecture adaptée au problème et qui donne les meilleurs résultats.

L'utilisation des réseaux de neurones pour l'égalisation des canaux a pour principal avantage de pouvoir traiter des canaux non linéaires. Les égaliseurs à base des réseaux de neurones ont été proposés comme une alternative des égaliseurs classiques, et ont fourni des améliorations significatives en performance dans une variété des canaux de transmissions. Le contexte d'utilisation d'un réseau de neurone à titre d'égalisation est donné à la figure 4.3 **[102]**.

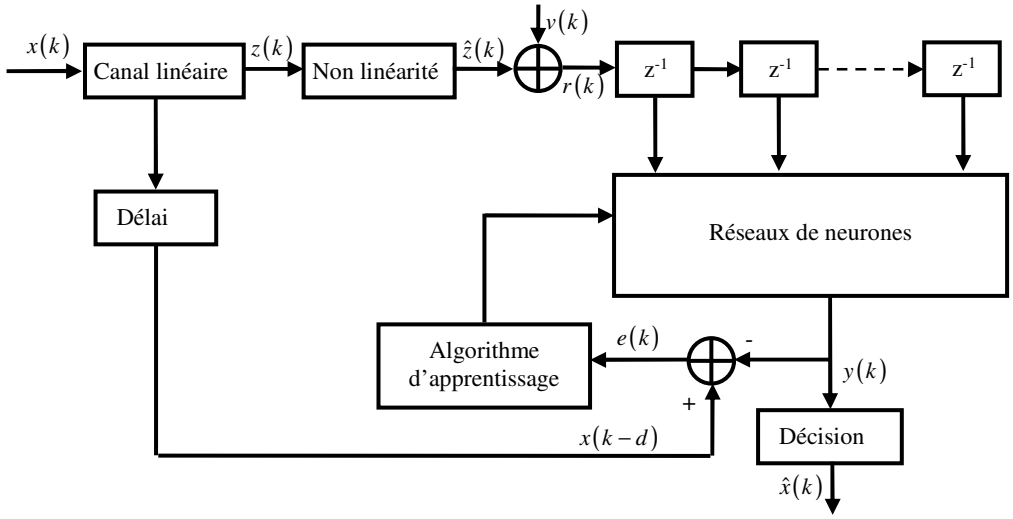


Figure 4.3 *Égaliseur à base de réseaux de neurones*

L'égaliseur à base des réseaux de neurones est réalisé en reliant les échantillons du signal reçu à la couche d'entrée du réseau. Le signal d'erreur calculé entre la sortie de l'égaliseur et la sortie désirée est utilisé pour ajuster les poids du réseau vers leurs valeurs optimales en utilisant un algorithme d'apprentissage.

4.2.1 Egaliseur DFE a base de réseaux de neurones

La structure conventionnelle de l'égaliseur à retour de décision utilise des algorithmes linéaires tels que LMS ou RLS et se compose d'un filtre à action directe et d'un filtre à rétroaction, comme le montre la figure 3.5 dans le chapitre 3, où le filtre à action directe est un égaliseur linéaire [9]. La linéarité de l'égaliseur limite les performances du système. Pour bénéficier des capacités de l'égaliseur à retour de décision DFE et du traitement non linéaire offert par les réseaux de neurone, nous présentons l'architecture d'un égaliseur DFE à base de réseaux de neurones noté DFE-ANN. L'architecture de l'égaliseur DFE-ANN est représentée par la figure 4.4. Les entrées de l'égaliseur DFE-ANN sont les échantillons de la séquence des symboles reçus ainsi que les échantillons en sortie du circuit de décision qui représentent les estimées de la séquence transmise. Le signal égalisé constituant la sortie du DFE-ANN se présente à l'entrée du circuit de décision pour décider de la valeur du symbole transmis.

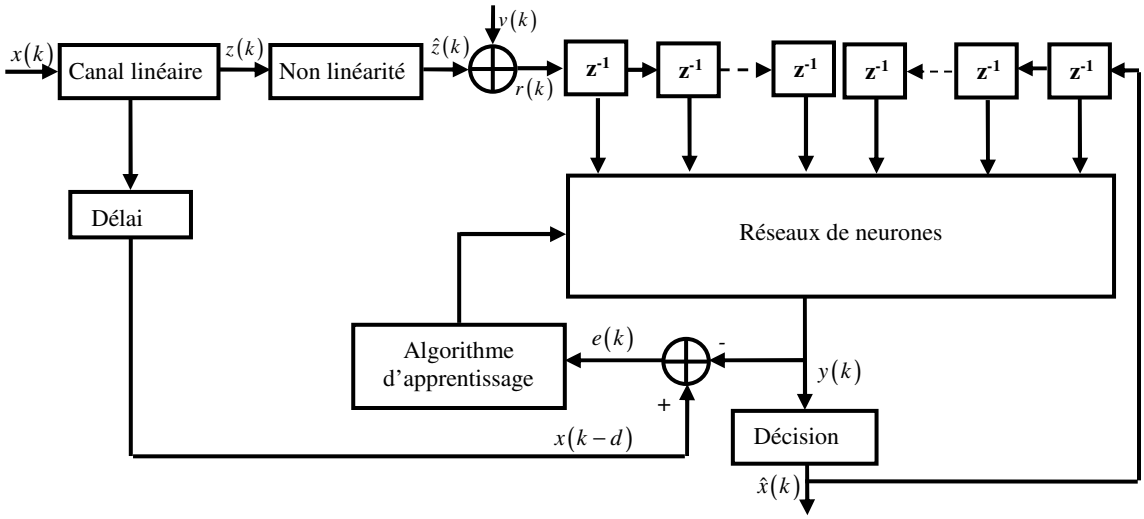


Figure 4.4 *Égaliseur DFE à base de réseaux de neurones*

Les égaliseurs neuronaux à retour de décision améliorent considérablement la vitesse de convergence et l'erreur quadratique moyenne (EQM) par rapport au DFE conventionnel ou l'égaliseur neuronal sans retour de décision [9].
Les réseaux de neurones fournissent une bonne cartographie non linéaire du modèle inverse du canal et peuvent gérer l'incertitude incluse dans les données reçues. Les réseaux de neurones de type Feed-Forward (FNN pour: Feed-Forward Neural Network) tels que les perceptrons multicouches (MLP) ou les réseaux de fonctions à base radiales (RBF) sont principalement concernés par la conception de l'égaliseur en raison de leur simplicité structurelle [9, 103]. S. Siu et al. [9] ont montré que l'égaliseur DFE basé sur le perceptron multicouche offre une performance supérieure par rapport à celle du DFE conventionnel, en raison de sa capacité à former des régions de décision complexes avec des limites non linéaires. Il convient de noter, cependant, que les structures invoquées par l'égaliseur DFE-ANN sont considérablement plus complexes que le DFE conventionnel.

4.2.2 Apprentissage des réseaux de neurones

L'apprentissage est une propriété fondamentale pour les réseaux de neurones, c'est le processus permettant au réseau de se spécialiser sur un problème spécifique à partir de son expérience. Il consiste généralement à modifier les poids, jusqu'à ce que le réseau puisse effectuer la tâche désirée. Donc l'apprentissage se traduit par une modification dans la valeur des poids reliant les neurones du réseau. Chaque poids $w_{i,j}$ reliant un neurone *i* au un neurone *j* à l'itération *n* est modifié selon l'équation suivante :

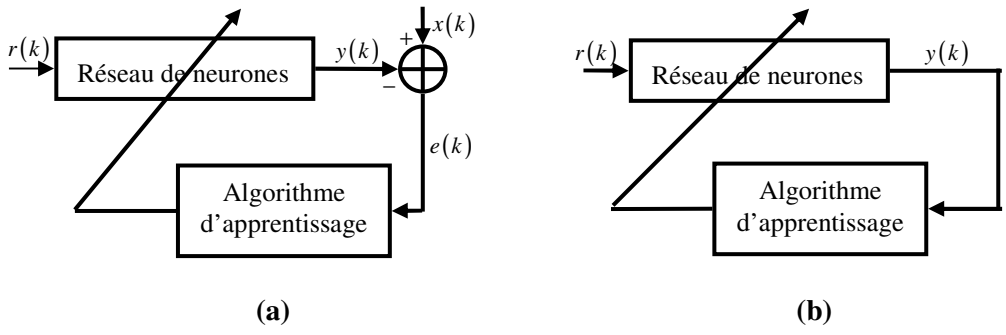
$$w_{i,j}(n) = w_{i,j}(n-1) + \Delta w_{i,j}(n-1) \tag{4.2}$$

Où $w_{i,j}(n)$ et $w_{i,j}(n-1)$ sont respectivement les valeurs de ce poids à la $n^{ème}$ et à la $(n-1)^{ème}$ itération et $\Delta w_{i,j}$ est le changement correspondant.

Selon la présence ou l'absence de la réponse désirée, l'apprentissage des réseaux de neurones peut être catégorisé en deux grandes familles, apprentissage supervisé et non supervisé (figure 4.5) :

- Dans l'apprentissage supervisé, le réseau est forcé à converger vers un état final précis, ce qui nécessite la connaissance à priori de la réponse désirée. chaque exemple de la base d'apprentissage est associé à une solution désirée. Cette dernière permet au réseau de connaître ces erreurs et de s'adapter à la présentation de chaque exemple d'apprentissage afin de se rapprocher du résultat souhaité. La rétro-propagation du gradient est l'une des méthodes les plus utilisées pour l'apprentissage supervisé des réseaux de neurones. Elle consiste à présenter des exemples au réseau, calculer sa sortie, ajuster les poids de façon à réduire l'écart entre cette sortie et la réponse désirée pour satisfaire un certain critère de performance.

- Dans l'apprentissage non supervisé, seules les valeurs d'entrée sont disponibles et le réseau est laissé libre de converger vers n'importe quel état final. La connaissance à priori de la sortie désirée n'est pas nécessaire, la procédure d'apprentissage est basée uniquement sur les valeurs d'entrées. Donc le réseau ne dispose pas d'une solution désirée sur les exemples d'apprentissage pour l'aider à ajuster ses paramètres, il doit chercher à représenter au mieux l'espace des exemples qui lui sont présentés.



(a) **(b)**

Figure 4.5 (a) *Apprentissage supervisé,* (b) *Apprentissage non supervisé*

4.3 DIFFERENTS TYPES DES RÉSEAUX DE NEURONES

Les réseaux de neurones artificiels peuvent être classés en deux grandes catégories :

- Réseaux neuronaux non bouclé de type Feed-Forward.
- Réseau neuronaux récurrent de type Feed-back.

Les réseaux neuronaux de type Feed-Forward sont formés de couches constituées de neurones de mêmes types. Plusieurs couches peuvent composer un réseau ayant plus de flexibilité afin de répondre à un plus grand nombre d'application. Dans ce type de réseau, l'information se propage couche par couche sans que le retour en arrière soit possible. La connectivité des neurones dans les réseaux à couche est locale, où ils ne sont reliés qu'à leur plus proche voisin de la couche précédente et la couche suivante. La couche de sortie est celle qui produit la sortie du réseau, il y a un neurone par sortie. Ce réseau est considéré comme un système neuronal non linéaire statique. Nous présentons dans cette section deux types de réseau neuronaux multicouches de type Feed-Forward qui sont : le réseau RBF (Radial Basis Function) et le perceptron multicouche MLP (Multi Layer Perceptron).

4.3.1 Le réseau de fonction à base radiale (RBF)

Le réseau de fonction à base radiale RBF (Radial Basis Functions) est un réseau à deux couches (figure 4.6), utilisé pour la première fois par Broomhead et Lowe **[104]**. Les neurones de la première couche sont reliés aux entrées et ont chacun deux paramètres : un vecteur prototype (ou centre) c et un coefficient d'étalement σ strictement positif. La fonction RBF la plus courante est la fonction gaussienne (figure 4.7) donnée par **[105]** :

$$f(x) = \exp\left(-\frac{x^2}{2\sigma^2}\right) \tag{4.3}$$

La sortie de la couche cachée notée x_{1i} est donnée par :

$$x_{1i} = f_i\left(\|x - c_i\|\right) \tag{4.4}$$

Avec x le vecteur d'entrée et $\|\,.\,\|$ dénote la norme euclidienne.

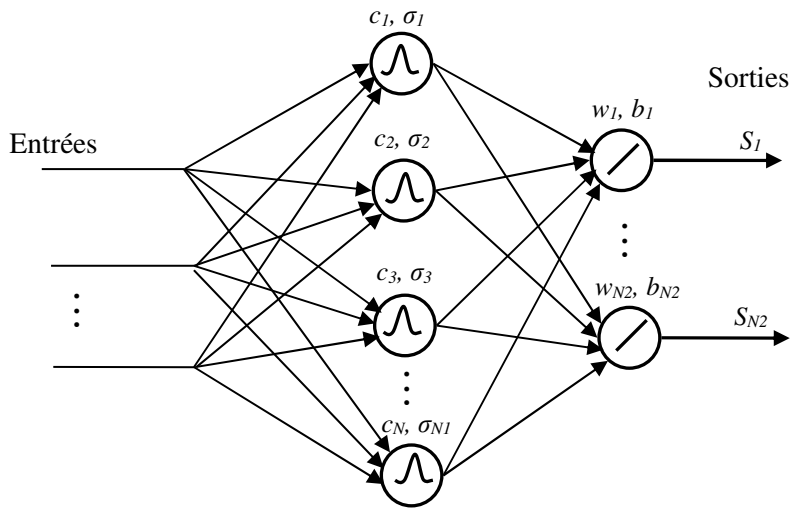


Figure 4.6 *Réseau RBF*

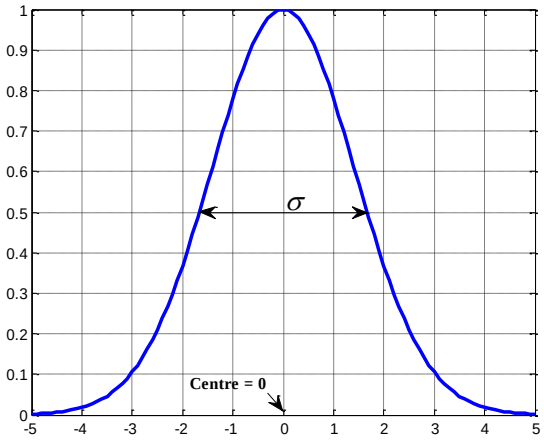


Figure 4.7 *fonction à base radiale (RBF)*

Le vecteur prototype *c* définit un point dans l'espace d'entrée. Un neurone RBF calcule sa sortie en se basant sur la distance entre chaque entrée et son vecteur prototype. Sa sortie est d'autant plus grande que cette distance est plus petite, c'est-à-dire d'autant que le vecteur d'entrée est très semblable à son vecteur prototype. La sortie du neurone est égale à 1 pour une entrée *x* égale à *c*, puis décroît vers 0 lorsque l'entrée s'éloigne de *c*. La vitesse de décroissance est réglée par σ : plus le coefficient est petit et plus la fonction sera concentrée autour du point *c* et proche de 0 ailleurs.
Les neurones de la seconde couche calculent la sortie du réseau en effectuant une combinaison linéaire des sorties de ceux de la première couche, et un biais est ajouté au total.
Chaque sortie du réseau RBF est donnée par la formule :

$$y_j = \sum_{i=1}^{N_1} w_{i,j} f_i \left(\| x - c_i \| \right) + b_i \qquad (4.5)$$

Où *j* est le numéro de la sortie (c'est à dire le numéro du neurone de la seconde couche dont on calcule la sortie), N_1 le nombre de neurones de la première couche, c_i le vecteur prototype du neurone numéro *i* de la première couche, $w_{i,j}$, $i = 1.....N_1$ les poids du neurone de sortie *j*, et b_i son biais. L'utilisation habituelle des RBF conserve une fonction d'activation linéaire en sortie mais l'utilisation d'une fonction non linéaire reste possible. Il a été montré que le réseau RBF capable d'approximer n'importe quelle fonction douce avec une précision donnée, pourvu que l'on fournisse un nombre suffisant de neurones, et que l'on utilise un algorithme d'apprentissage adéquat **[106]**.
Lors de l'apprentissage d'un réseau RBF deux problèmes se posent : la constitution de la première couche (choix du nombre de neurones, choix des prototypes et des coefficients d'étalement), et la détermination des poids et biais de la seconde couche **[105]**. L'apprentissage d'un réseau RBF consiste donc à déterminer les paramètres inconnus de ce réseau. D'une manière générale l'apprentissage est composé de deux étapes :

Une étape de para-métrisation des fonctions de base qui consiste en une règle d'apprentissage non supervisée dans la couche cachée. Elle a pour rôle de trouver la forme exacte des fonctions de base qui consiste à déterminer le nombre et la position

des centres. Le choix de la position des centres et le nombre de neurones reste généralement arbitraires. Il existe des méthodes de positionnement automatique des centres, ces méthodes consistent à déplacer les centres vers les points où l'erreur est grande **[21]**. L'algorithme des k-means et la méthode d'apprentissage non supervisée souvent utilisée pour la sélection des centres. La version la plus couramment utilisée est la classification par k-means adaptatif basé sur la distance euclidienne. Les étapes principales de l'algorithme k-means sont présentées comme suit :

Etape 1 : Initialisation
Le nombre k (nombre de fonction RBF, ou neurones cachés)
Les centres des fonctions de base radiale.

Etape 2 :
Calculer la distance euclidienne entre les centre de chaque fonction RBF et le vecteur d'entée x :

$$D_i(n) = \|x(n) - c_i(n)\|$$

Etape 3 :
Déterminer le centre I le plus proche du vecteur d'entrée, tel que :

$$D_I(n) = \min_i D_i(n)$$

Etape 4 :
Ajuster le vecteur des centres c_I en utilisant la loi d'adaptation suivante :

$$c_I(n+1) = c_I(n) + \mu\left(x - c_I(n)\right)$$

Où μ est une petite constante positive.

Une autre étape utilisée pour l'apprentissage des poids de la couche de sortie qui consiste en une règle d'apprentissage supervisé pour l'adaptation des poids tels que les algorithmes RLS ou LMS. L'algorithme LMS est utilisé pour ajuster les poids de la couche de sortie suivant les étapes décrites ci-dessous.

Etape 1 :
Présenter le vecteur de sortie du réseau et le vecteur de sortie désiré.

Etape 2 :
Calculer l'erreur entre la sortie du réseau et la sortie désirée :

$$e_j(n) = d_j(n) - y_j$$

Etape 3 :
Ajuster les poids de la couche de sortie en utilisant la loi d'adaptation suivante :

$$w_{kj}(n+1) = c_{kj}(n) + \mu e_j(n)\phi_k\left(\|x - c_k\|\right)$$

Où : $\phi(t) = \exp\left(-\frac{t^2}{2\sigma^2}\right)$

μ est une petite constante positive.

4.3.2 Le perceptron multicouche (MLP)

Le perceptron multicouche MLP (Multi Layer Perceptron), est la deuxième grande famille de réseaux de neurones. C'est le réseau de neurones le plus utilisé et le plus étudié. Un perceptron multicouche se compose d'une couche d'entrée, une couche de sortie et une ou plusieurs couches cachées. La figure (4.8) illustre un MLP avec deux couches cachées. La première couche est reliée aux entrées (appelée couche d'entrée), puis ensuite chaque couche est reliée à la couche suivante. C'est la dernière couche qui produit les sorties du MLP, et elle est appelée couche de sortie. Les sorties des autres couches ne sont pas visibles à l'extérieur du réseau, et elles sont appelées pour cette raison couches cachées. Chaque neurone d'une couche étant totalement connecté aux neurones de la couche suivante et les neurones d'une même couche ne sont pas interconnectés. Chaque neurone d'une couche reçoit des signaux de la couche précédente et transmet le résultat à la couche suivante. L'orientation du réseau est fixée par le sens unique de la propagation de l'information, de la couche d'entrée vers la couche de sortie.

La sortie du $j^{ème}$ neurone de la $k^{ème}$ couche $y_j^{(k)}$ est donnée par :

$$y_j^{(k)} = f_k\left(b_j^{(k)} + \sum_{i=1}^{N_{k-1}} w_{i,j}^{(k)} s_j^{(k)}\right) \tag{4.6}$$

Où : f_k est la fonction d'activation de ce neurone, N_{k-1} est le nombre de neurones de la $(k-1)^{ème}$ couche, $w_{i,j}^{(k)}$ est le poids reliant le $i^{ème}$ neurone de la couche précédente à ce neurone et $b_j^{(k)}$ son biais.

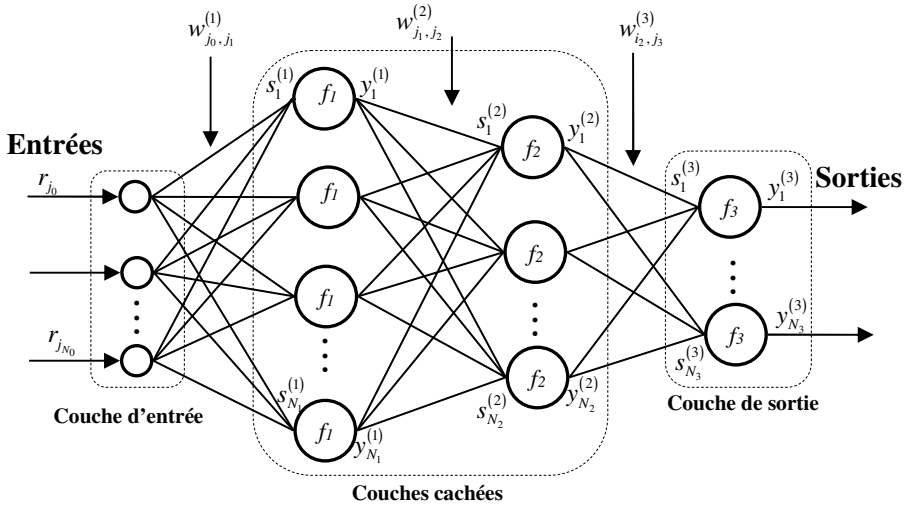


Figure 4.8 *Perceptron multicouche avec deux couches cachées.*

L'importance du MLP réside essentiellement dans ces capacités de former des frontières de décision complexe. Ces dernières sont déterminées par le nombre de couches cachées et le nombre de neurones de chaque couche. Cependant, trop de couche cachées compliquera l'apprentissage et augmentera le coût de calcule et peut de couches sera

inadéquat pour créer la non linéarité suffisante. Comme pour le RBF, il a été montré qu'un perceptron à deux couches avec des fonctions d'activation non-linéaire sur la première couche et une fonction d'activation linéaire sur la seconde est capable d'approximer n'importe quelle fonction douce avec une précision donnée **[107, 108]**, pourvu que l'on fournisse un nombre suffisant de neurones dans la couche cachée.
La fonction d'activation utilisée dans le modèle de W. McCulloch et W. Pitts **[8]** est la fonction échelon. Elle fait passer l'activation du neurone d'une valeur à une autre dès que l'entrée résultante dépasse un certain seuil. L'inconvénient de cette fonction est qu'elle n'est pas différentiable, ce qui pose un problème pour les algorithmes d'apprentissage basés sur le gradient. Pour remédier à cet inconvénient, on cherche à approximer cette fonction d'activation par une fonction non-linéaire différentiable. Deux fonctions de ce type sont particulièrement intéressantes et sont souvent utilisées : la tangente hyperbolique (f_1) et la fonction sigmoïde (f_2). La figure 4.9 montre les courbes de la tangente hyperbolique et sigmoïde standard. Elles sont définies par :

$$f_1(x) = \tanh(x) = \frac{1-e^{-ax}}{1+e^{-ax}} \tag{4.7}$$

$$f_2(x) = \text{logsig}(x) = \frac{1}{1+e^{-ax}} \tag{4.8}$$

Où : a est le paramètre (dit aussi gain), qui contrôle la pente non linéaire de la fonction.
La différence entre ces deux dernières fonctions est le domaine des valeurs prises, qui est de {-1,1} pour la tangente hyperbolique et de {0,1} pour la fonction sigmoïde standard.

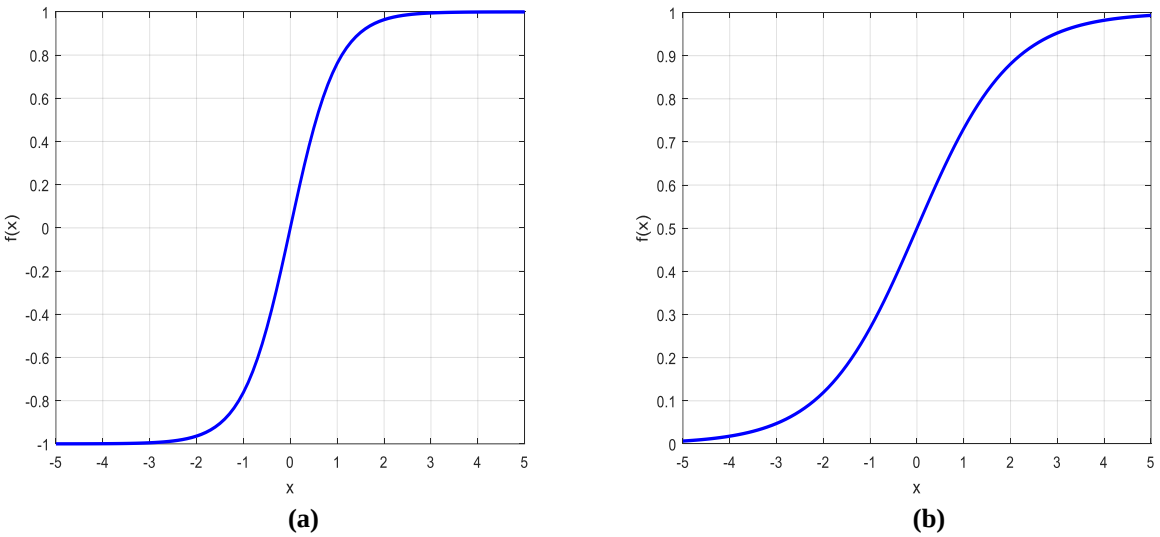


Figure 4.9 *Fonctions d'activation :* (a) *tangente hyperbolique* (a=*2),*
(b) *sigmoïde standard* (a=*1).*

4.3.2.1 Apprentissage du MLP par la Rétro-Propagation (RP)

La rétro-propagation du gradient BP (Back Propagation) est l'une des méthodes supervisée les plus utilisées pour l'apprentissage des réseaux de neurones. C'est une généralisation de l'algorithme LMS conçu spécialement pour le MLP. Son développement est lié aux travaux de Rumelhart et al en 1986 **[15]**. L'apprentissage consiste à ajuster les coefficients synaptiques pour que les sorties du réseau soient les plus proches possible des sorties de l'ensemble d'appretissage **[17]**. Donc il faut spécifier une règle d'apprentissage afin de l'utiliser pour l'adaptation de ces paramètres. Pour cela la rétro-propagation utilise une méthode d'apprentissage qui se divise en deux étapes **[17]** :

- Une étape de propagation qui consiste à présenter une configuration d'entrée au réseau, puis à propager cette entrée de proche en proche de la couche d'entée à la couche de sortie en passant par les couches cachées.
- Une étape de rétro-propagation qui consiste, après le processus de propagation, à minimiser l'erreur commise sur l'ensemble des exemples présentés. Cette erreur représente la somme des différences au carré entre les réponses calculées et celle désirées pour tous les exemples contenus dans l'ensemble d'apprentissage.

La rétro-propagation du gradient commence l'apprentissage par un choix aléatoire des valeurs des poids en présentant l'entrée et la sortie désirée correspondante. A la sortie du réseau, l'erreur commise et le gradient de l'erreur par rapport à tous les poids sont calculés, ensuite les poids sont ajustés. Cette procédure est répétée jusqu'à ce que les sorties du réseau soient suffisamment proches des sorties désirées.
Le gradient de l'erreur est calculé par la rétro-propagation pour chaque neurone d'un réseau de neurones, de la dernière couche vers la première. L'erreur quadratique est souvent employée comme étant la fonction de coût. Il consiste donc à minimiser la distance entre la sortie calculée $Y^{(q)}$ et la sortie désirée $T^{(q)}$ correspondantes à chaque exemple d'apprentissage $X^{(q)}$. Pour un ensemble de Q exemples d'apprentissage, l'erreur quadratique totale est donnée par :

$$E = \sum_{q=1}^{Q} \sum_{j=1}^{J} \left(t_j^{(q)} - y_j^{(q)} \right)^2 \tag{4.9}$$

L'algorithme BP modifie les poids du réseau de façon à effectuer une descente de gradient sur le critère d'erreur pour atteindre le minimum. Au début de l'apprentissage, les poids sont initialisés avec des valeurs aléatoires et modifiés ensuite dans une direction qui réduira l'erreur. La modification Δw d'un poids w est donnés par :

$$\Delta w = -\eta \frac{\partial E}{\partial w} \tag{4.10}$$

Où : η est une constante positive appelée pas d'apprentissage permettant de définir la taille des modifications des poids. Il s'agit donc de prendre un pas dans l'espace des poids permettant de réduire la fonction coût. La phase d'initialisation des poids s'effectue après avoir fixé la topologie du réseau. Dans cette phase, nous avons intérêt à choisir des

valeurs de sorte que l'apprentissage soit rapide et uniforme, en d'autre termes les valeurs finales d'équilibre des poids seront atteintes simultanément **[109]**. Pour ce faire, les poids doivent être initialisés aléatoirement selon une distribution uniforme autour d'une certaine valeur moyenne $\left(w_{\min} < w < w_{\max}\right)$ **[110]**. Ces deux paramètres sont choisis avec des valeurs petites afin que l'activation des neurones cachés soit petite. Le choix du pas d'apprentissage influe beaucoup sur la rapidité de convergence, un pas trop petit ralenti l'apprentissage, un pas trop important provoque un risque d'instabilité. Théoriquement, le pas d'apprentissage optimal est celui qui mène au minimum d'erreur dans une seule époque d'apprentissage (figure 4.10). La détermination d'un gain optimale est l'un des problèmes de la RP **[95]**.

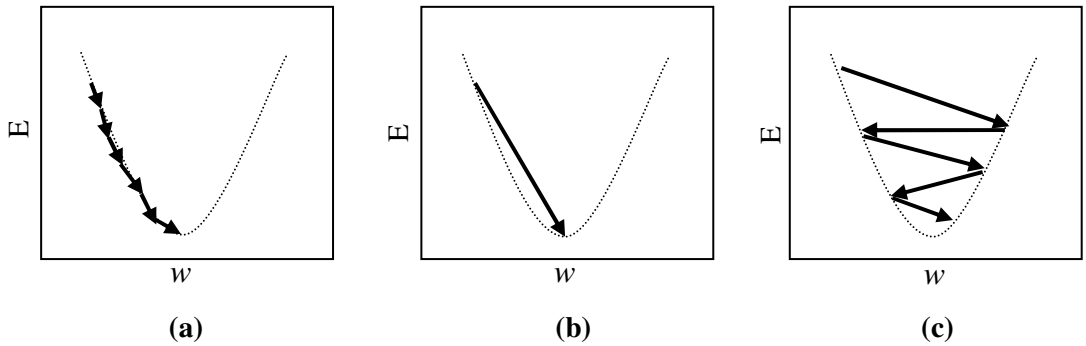


Figure 4.10 *Descente de gradient dans un problème unidimensionnel avec différents valeur du pas d'apprentissage :* (a) $\eta < \eta_{optimal}$, (b) $\eta = \eta_{optimal}$, (c) $\eta > \eta_{optimal}$

Du fait que la RP est basée sur une descente de gradient, nous pourrons avoir une idée sur cet algorithme en étudiant les surfaces d'erreur de la fonction coût employée en fonction des poids. Le problème le plus important concerne les minima locaux, si la surface d'erreur admet plusieurs minima locaux, il sera peu probable que le réseau atteint le minimum global, et par conséquent cela pourra conduire à une mauvaise performance.

Pour un MLP avec deux couches cachée (Fig. 4.8), la mise à jour des poids des couches cachées et ceux de la couche de sortie est donnée par :

$$w_{j_{k-1},j_k}^{(k)}(n) = w_{j_{k-1},j_k}^{(k)}(n-1) - \eta \frac{\partial E(n)}{\partial w_{j_{k-1},j_k}^{(k)}} \quad k = 1,2,3 \tag{4.11}$$

Le développement de l'équation (4.11) donne les équations d'adaptation suivantes :

$$w_{j_2,j_3}^{(3)}(n) = w_{j_2,j_3}^{(3)}(n-1) + \eta_3 \left(t_{j_3} - y_{j_3}^{(3)}\right) f_3^{'}\left(s_{j_3}^{(3)}\right) y_{j_2}^{(2)} \tag{4.12}$$

$$w_{j_1,j_2}^{(2)}(n) = w_{j_1,j_2}^{(2)}(n-1) + \eta_2 \left(\sum_{j_3=1}^{N_3} \left(t_{j_3} - y_{j_3}^{(3)}\right) f_3^{'}\left(s_{j_3}^{(3)}\right) w_{j_2,j_3}^{(3)} \right) f_2^{'}\left(s_{j_2}^{(2)}\right) y_{j_1}^{(1)} \tag{4.13}$$

$$w_{j_0,j_1}^{(1)}(n) = w_{j_0,j_1}^{(1)}(n-1) + \eta_1 \left(\sum_{j_3=1}^{N_3} \sum_{j_2=1}^{N_2} \left(t_{j_3} - y_{j_3}^{(3)}\right) f_3^{'}\left(s_{j_3}^{(3)}\right) w_{j_2,j_3}^{(3)} f_2^{'}\left(s_{j_2}^{(2)}\right) w_{j_1,j_2}^{(2)} \right) f_1^{'}\left(s_{j_1}^{(1)}\right) r_{j_0} \tag{3.14}$$

Où : η_1, η_2, η_3, sont les pas d'apprentissage, et $f_1^{'}, f_2^{'}, f_3^{'}$, sont respectivement les dérivées des fonctions d'activation des neurones de la première et la deuxième couches cachées,

et de la couche de sortie. Dans le cas des fonctions sigmoïde et tangente hyperbolique, les dérivées sont respectivement données par **[39]** :

$$f_k^{'}\left(s_{j_k}^{(k)}\right)=y_{j_k}^{(k)}\left(1-y_{j_k}^{(k)}\right) \quad k=1,2,3 \tag{4.15}$$

$$f_k^{'}\left(s_{j_k}^{(k)}\right)=\left(1+y_{j_k}^{(k)}\right)\left(1-y_{j_k}^{(k)}\right) \quad k=1,2,3 \tag{4.16}$$

Les fonctions d'activations jouissent d'une place importante dans les réseaux neuronaux multicouche. Cela a conduit à l'utilisation d'une expansion de fonctions d'activation. Etant donné qu'elles gouvernent la non-linéarité du réseau et influencent directement la vitesse de convergence des algorithmes d'adaptation.

Bien que la RP fonctionne pratiquement avec n'importe quelle fonction d'activation, étant données quelques conditions simples telles que la continuité de cette fonction ainsi que sa dérivée, il y a un certain nombre de propriétés souhaitables **[109]**: D'abord, la fonction d'activation doit être non linéaire. Deuxièmement, la fonction d'activation doit être saturable, sa sortie doit être limitée par une valeur maximale et une valeur minimale. Troisièmement, la fonction d'activation doit être continue et lisse, en d'autre termes la fonction et sa dérivée doivent être définies sur tout l'intervalle d'entrée. La fonction tangente hyperbolique possède toutes les propriétés précédentes (Fig 4.11), elle est lisse, différentiable, non-linéaire, et saturable.

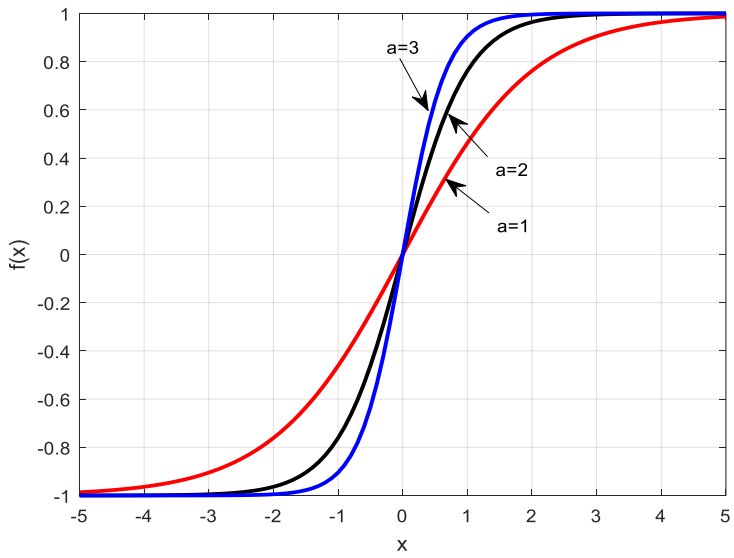


Figure 4.11 *La fonction tangente hyperbolique avec différentes valeurs de* a.

Certains travaux ont été effectués pour montrer la relation entre les réseaux RBF et le réseau MLP. Particulièrement, si l'on considère qu'un MLP est un approximateur universel, il peut approximer un réseau RBF et vice versa **[111]**.

La différence entre les deux réseaux porte sur l'architecture qui est figée en deux couches pour les RBF et peut comporter plusieurs couches pour le MLP. En plus le réseau MLP jouit de la flexibilité de la structure, de bonnes capacités de représentation et une large utilisation pour les applications industrielles et un grand nombre d'algorithmes d'apprentissage **[21, 112, 113]**.

4.4 ÉGALISATION PAR LES RÉSEAUX MLP et RBF

Pour apercevoir la performance de l'égaliseur à base de réseaux de neurones, nous présentons dans cette section les résultats de simulation obtenus en utilisant le logiciel MTLAB pour les égaliseurs à base de réseau (RBF), à base du perceptron multicouche (MLP), l'égaliseur à retour de décision à base du perceptron multicouche (DFE-MLP) et l'égaliseur à retour de décision à base de réseau RBF (DFE-RBF). La performance se fait sur les courbes de convergence qui décrit l'évolution de l'erreur quadratique moyenne (EQM ou MSE: Mean Square Error) et les courbes de taux d'erreurs binaires (TEB ou BER: Bit Error Ratio).
La séquence de données est constituée des symboles (-1,1) uniformément distribués, le rapport signal sur bruit SNR est de 25 dB.

4.4.1 Modèle de canal

Deux modèles standard de canaux sont utilisés dans la simulation, le premier canal $H_1(z)$ est le même que celui utilisé au chapitre précédent. Le deuxième est un canal de Proakis B, il est substantiellement plus dur à égaliser par rapport au premier canal. Ce canal est très utilisé pour la simulation des égaliseurs. Il est représenté par la fonction de transfert en Z définie par :

$$H_2(z) = 0.407 + 0.815z^{-1} + 0.407z^{-2} \tag{4.17}$$

Les caractéristiques de ce canal sont illustrées à la figure 4.12. Il possède deux zéros, dont l'un est à l'intérieur du cercle unité et l'autre est à l'extérieur. Les zéros de ce canal sont donnés par : z_1= -1.0508 et z_2= -0.9516. La réponse en amplitude présente un évanouissement très profond d'environ -30 dB, sélectif en fréquence et assez difficile à identifier. Ce canal résulte des interférences entre symboles sévères **[1, 115]**. La réponse en phase de ce canal est quasi linéaire.

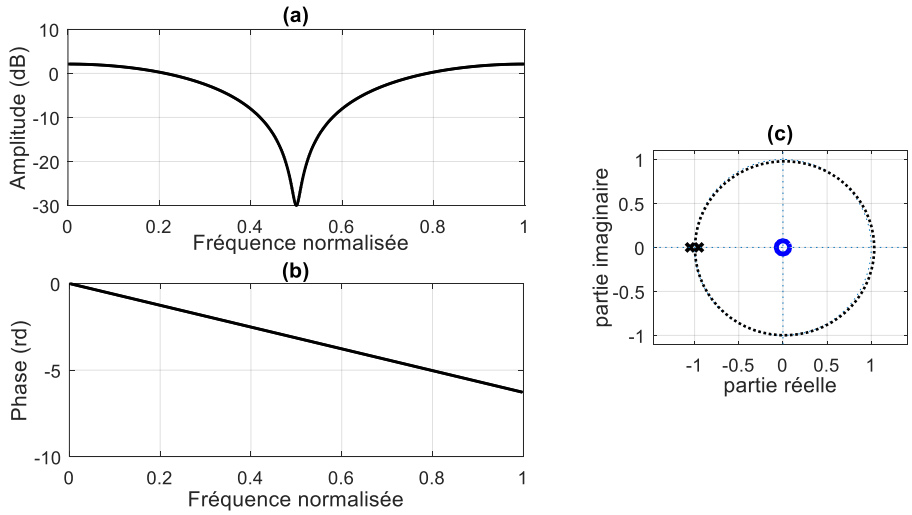


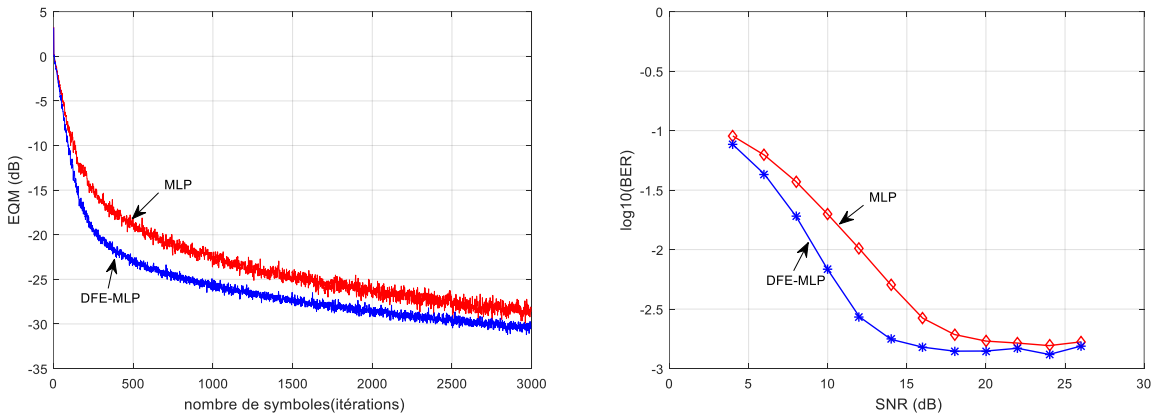
Figure 4.12 *Caractéristiques du canal $H_2(z)$:* (a) *Réponse en amplitude*, (b) *Réponse en phase*, (c) *Les zéros de canal*.

L'égaliseur à base des réseaux de neurones étant un égaliseur non linéaire possédant des capacités attirantes, il sera intéressant de tester ses performances pour des canaux non linéaires. Nous considérons un modèle de canal non linéaire composé d'un canal linéaire suivi d'une non-linéarité définie au chapitre précédent. Les paramètres contrôleurs de la

non-linéarité D_1, D_2 *et* D_3 de l'équation (3.50) sont définis respectivement à 1, 0.1 et 0.05, comme indiqué dans **[91, 114]**.

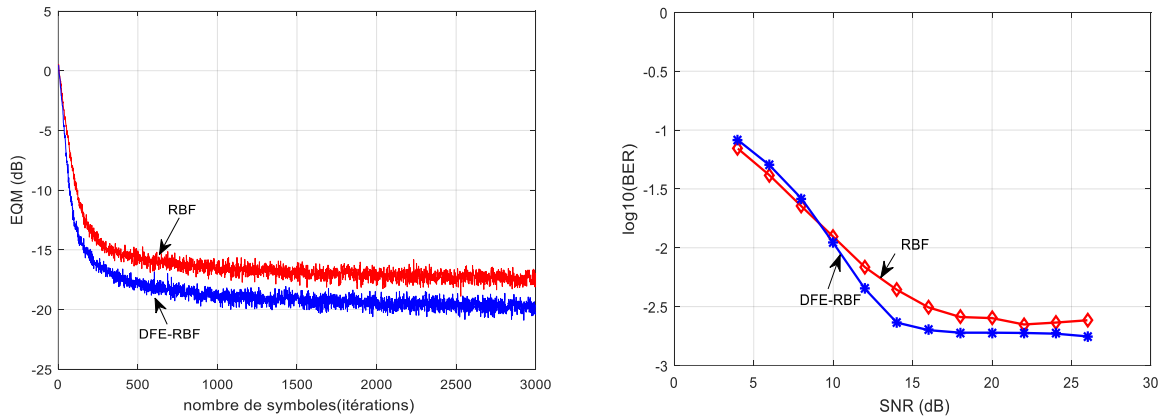
4.4.2 Performances des égaliseurs MLP, DFE-MLP, RBF, DFE-RBF

La structure de l'égaliseur MLP et l'égaliseur DFE-MLP utilisé dans la simulation contient cinq échantillons dans la couche d'entrée pour le MLP, quatre échantillons dans la section direct et un échantillon dans la section de retour pour le DFE-MLP, le nombre de neurones dans la première et la deuxième couche cachée et dans la couche de sortie est 9, 3 et 1, respectivement {(4,1) DFE avec (9,3,1) MLP Structure}**[9,88, 93, 94, 116]**. Pour le réseau RBF, le traitement du problème d'égalisation est le même que dans le MLP, sauf que la structure de l'égaliseur RBF et les paramètres internes sont différents. La structure de l'égaliseur RBF et l'égaliseur DFE-RBF utilisé dans la simulation contient trois échantillons dans la couche d'entrée pour le RBF, deux échantillons dans la section direct et un échantillon dans la section de retour pour le DFE-RBF, le nombre de neurones dans la couche cachée et la couche de sortie est 30, 1, respectivement **[14]**.



(a) **(b)**

Figure 4.13 *Courbe de l'EQM (a) et BER (b) de l'égaliseur MLP et DFE-MLP (canal $H_1(z)$).*



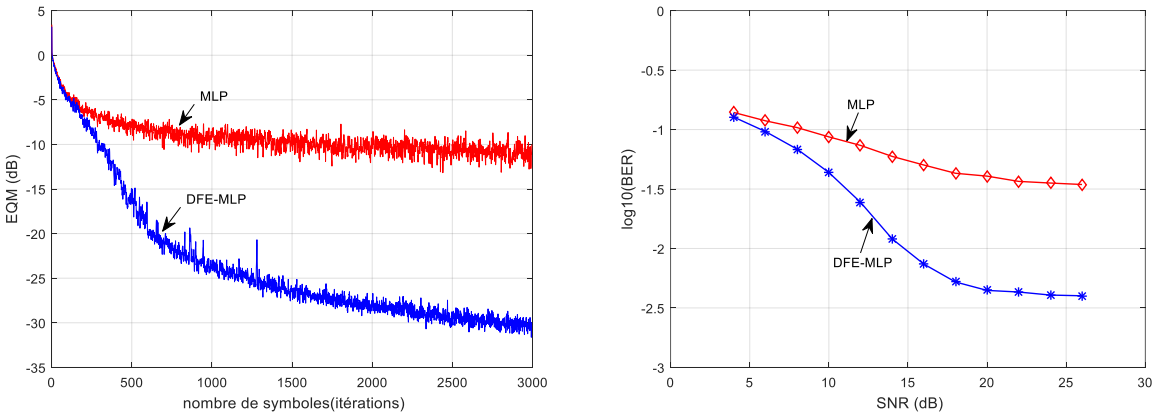
(a) **(b)**

Figure 4.14 *Courbe de l'EQM (a) et BER (b) de l'égaliseur RBF et DFE-RBF (canal $H_1(z)$).*

L'erreur quadratique moyenne est l'une des mesures les plus utiles pour la performance d'un égaliseur. La performance de l'EQM est obtenue en moyennant 600 simulations indépendantes, chacune consistant en 3000 itérations et caractérisée par des symboles aléatoires différents et des paramètres initiaux aléatoires. Les figures 4.13(a) et 4.14(a) présentent les courbes de convergence de l'erreur quadratique moyenne (MSE) des

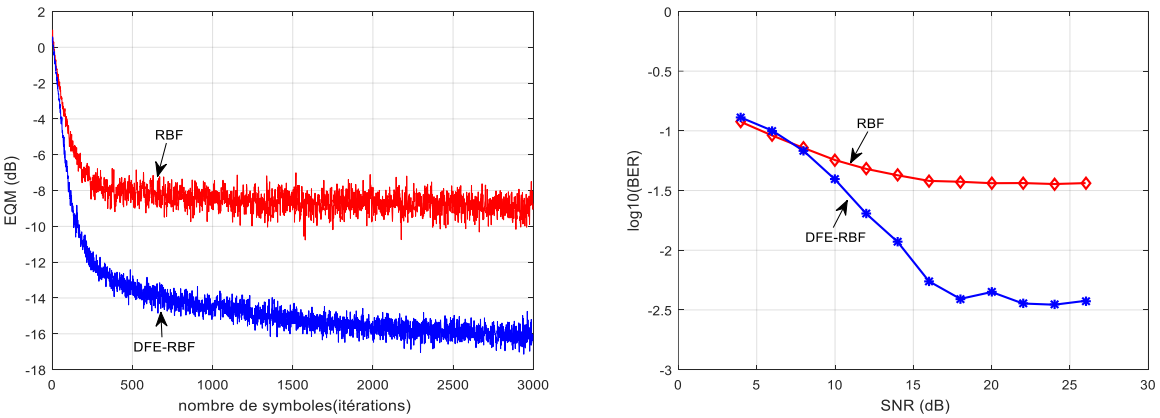
égaliseurs considérés pour le canal $H_1(z)$. Ces courbes montrent une amélioration dans le temps de convergence et aussi dans le niveau MSE lors de l'utilisation de DFE. Il est également clairement montré que l'égaliseur MLP fonctionne mieux que l'égaliseur RBF (avec une architecture de neurone différente). Comme le montre les figures 4.15(a) et 4.16(a), une amélioration similaire est également obtenue par les deux égaliseurs DFE-MLP et DFE-RBF pour le canal $H_2(z)$. L'égaliseur DFE-MLP engendre un niveau d'EQM supérieur à l'égaliseur DFE-RBF. Cependant, le temps de convergence de l'égaliseur DFE-MLP montre une dégradation dans la performance par rapport à l'égaliseur DFE-RBF, en prouvant la supériorité de l'égaliseur DFE-RBF pour les 250 premières itérations. Pour l'égaliseur MLP et RBF (sans retour de décision), les courbes montres une dégradation dans la performance qui se manifeste par une augmentation de la valeur de l'état stable de l'EQM à cause de la distorsion apporter par ce canal.

La partie (b) des figures 4.13, 4.14, 4.15, 4.16 montrent les courbes du taux d'erreur binaire (BER) des égaliseurs considérés. Ces courbes sont obtenues en prenant en moyenne 100 simulations indépendantes de longueur 10000 symboles. Les 2000 premiers symboles sont utilisés pour l'apprentissage et les autres sont utilisées pour le test. Les courbes BER montrent avec clarté la même classification des égaliseurs en termes de performances, en prouvant la supériorité de l'égaliseur DFE. Cependant, d'après les courbes de la figure 4.14(b), une légère amélioration au niveau des faibles SNR est observée dans l'égaliseur RBF par rapport à l'égaliseur DFE-RBF.



(a) **(b)**

Figure 4.15 *Courbe de l'EQM (a) et BER (b) de l'égaliseur MLP et DFE-MLP (canal $H_2(z)$).*



(a) **(b)**

Figure 4.16 *Courbe de l'EQM (a) et BER (b) de l'égaliseur RBF et DFE-RBF (canal $H_2(z)$).*

4.5 AMÉLIORATION DES PERFORMANCES DE LA RÉTRO-PROPAGATION (RP)

Nous consacrons cette section à l'égalisation par le perceptron multicouche qui est le réseau de neurones le plus utilisé, et qui constitue également l'application principale de la méthode proposée.

4.5.1 Méthode proposée pour l'apprentissage du perceptron multicouche

Nous abordons l'égalisation par le perceptron multicouche en décrivant son architecture et son apprentissage par la rétro-propagation qui est l'algorithme de base d'apprentissage du MLP, et dont l'amélioration de ses performances constitue l'objectif principal de notre travail. La RP est un algorithme puissant, utile et relativement facile à comprendre. C'est un algorithme simple même pour les modèles complexes ayant des centaines ou des milliers de paramètres [109, 95]. Malgré son succès accentué dans nombreuses application, la RP possède cependant un taux de convergence lent. Afin d'accroître les performances de la RP, plusieurs travaux ont été proposés dans la littérature [117-130]. Parmi les méthodes proposées, citons : Méthodes basées sur les poids initiaux [117-123], Méthodes basées sur le gain d'apprentissage [110, 124], Méthodes basées sur les fonctions d'activation [125-127].

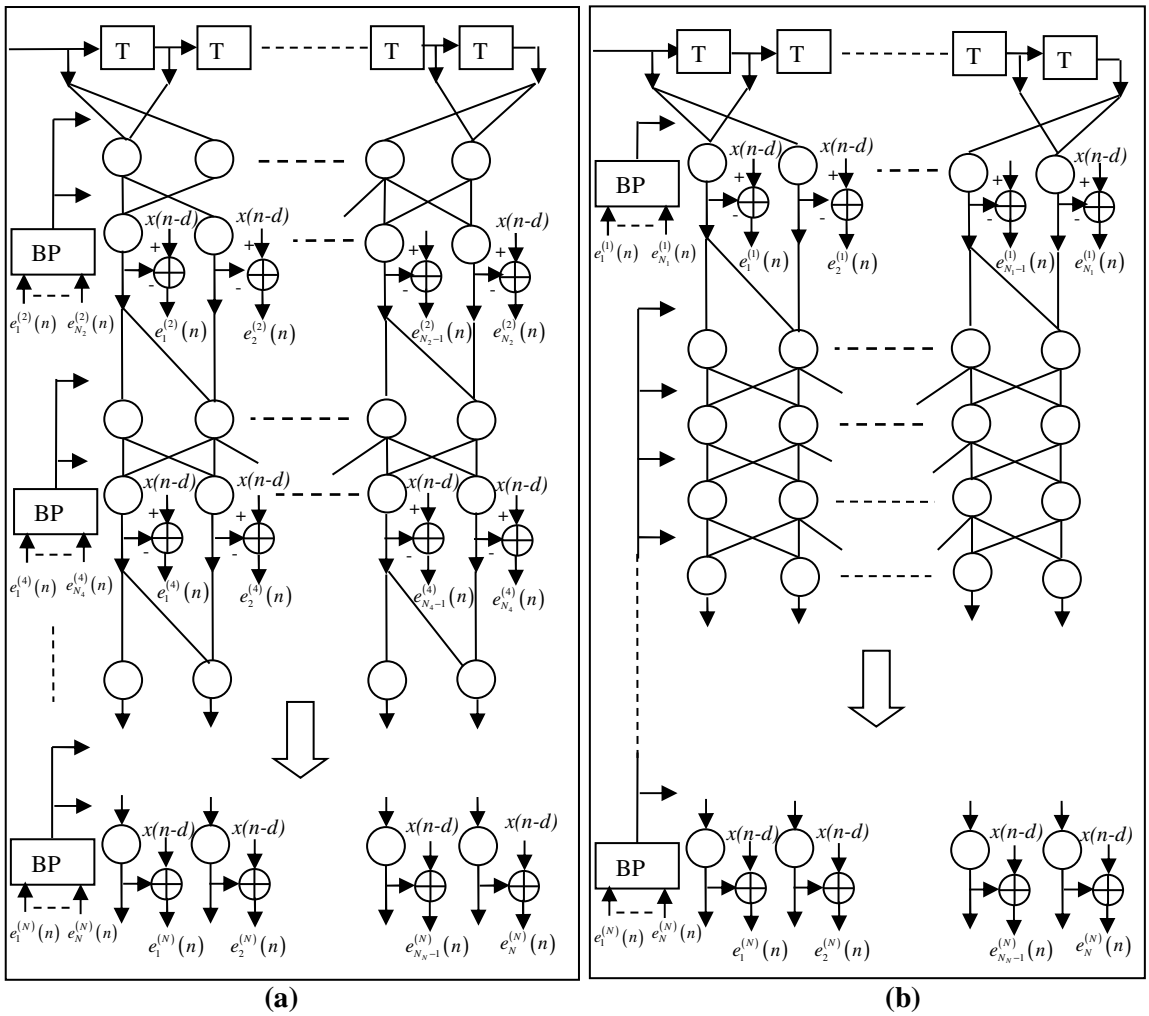


Figure 4.17 *Structure de l'égaliseur MLP avec deux types d'apprentissage, l'apprentissage HBP* (a) *et l'apprentissage NHBP* (b).

Notre méthode consiste à améliorer l'algorithme conventionnel de la rétro-propagation du gradient, en créant un nouvel hiérarchique RP pour optimiser les coefficients de l'égaliseur à base du perceptron multicouche (MLP). La méthode proposée est basé sur une méthode précédemment introduite dans **[94, 114]**, pour améliorer les performances de l'algorithme RP. Cette méthode est basée sur un nouveau schéma d'apprentissage nommé HBP. Nous avons dénoté notre méthode par NHBP (pour : A new scheme hierarchical BP). La figure 4.17 montre la différence entre la nouvelle stratégie d'apprentissage (NHBP) **[131]**, et la stratégie proposée dans **[94, 114]**.
Dans la figure 4.17 (a), Chaque algorithme optimise un sous-MLP à deux couches. Ces sous-MLP sont disposés de la couche d'entrée à la couche de sortie du MLP. Dans la figure 4.17 (b), le MLP peut être divisé en deux sous-MLP, et chaque sous-MLP est optimisé par son propre algorithme BP dans lequel le premier BP optimise la première couche et l'autre optimise toutes les autres couches (deuxième couche vers la couche de sortie).

4.5.2 Performance de la méthode proposée

La performance du nouvel hiérarchique BP (NHBP) est évaluée et comparée à la stratégie d'apprentissage conventionnelle (BP) et à la stratégie d'apprentissage (HBP). Les figures 4.18 et 4.19 montrent la différence entre la stratégie d'apprentissage conventionnel (BP) et la nouvelle stratégie d'apprentissage (NHBP).
La structure du MLP-DFE utilisé dans la simulation contient cinq échantillons dans la couche d'entrée, quatre échantillons dans la section direct et un échantillon dans la section de retour, le nombre de neurones dans la première et la deuxième couche cachée et dans la couche de sortie est 9, 3 et 1, respectivement {(4,1) DFE avec (9,3,1) MLP Structure}. Les trois algorithmes sont évalués dans la même structure DFE-MLP.
Le message numérique appliqué au canal est constituée des symboles {-1, 1} uniformément distribués. Le bruit du canal est considéré comme un bruit blanc Gaussien additif avec un rapport signal sur bruit (SNR) de 25dB.

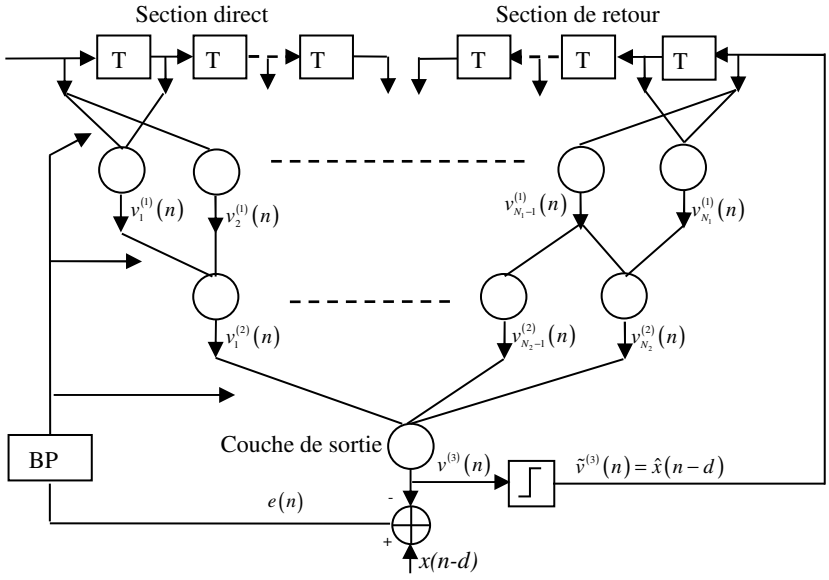


Figure 4.18 *Apprentissage de trois couches DFE-MLP par la stratégie conventionnelle (BP).*

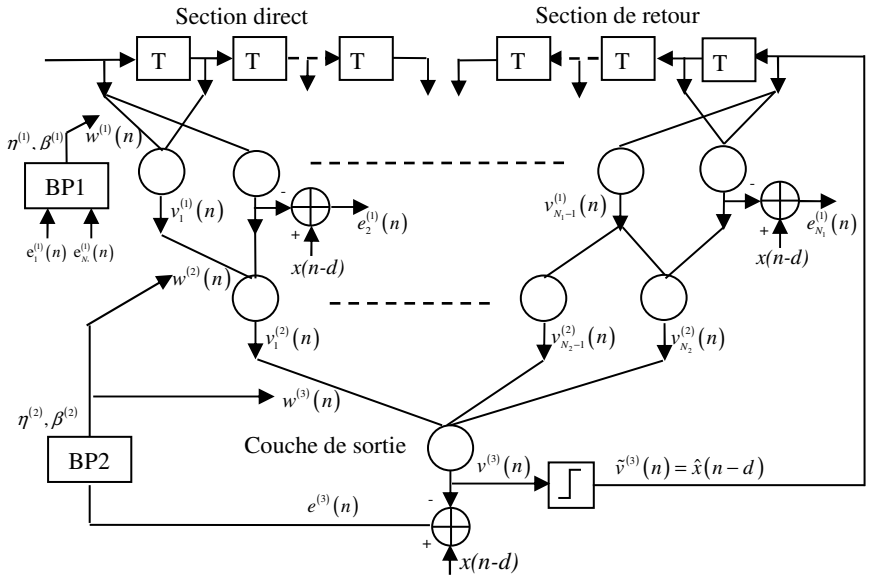


Figure 4.19 *Apprentissage de trois couches DFE-MLP par la nouvelle stratégie (NHBP).*

Les paramètres d'apprentissage η = 0,07, β = 0,05 sont utilisés pour l'algorithme BP et les paramètres d'apprentissage $\eta^{(1)}$ = 0,07, $\beta^{(1)}$ = 0,05, $\eta^{(2)}$ = 0,07, $\beta^{(2)}$ = 0,05 et $\eta^{(3)}$ = 0,07, $\beta^{(3)}$ = 0,05 sont utilisés pour les algorithmes NHBP et HBP **[9,94, 114]**.

Où : η est le pas d'apprentissage des poids, β est le pas d'apprentissage des biais (seuil propre au neurone qui est un nombre réel représentant la limite à partir de laquelle le neurone s'activera).

D'après la figure 4.18, on peut voir que le $i^{ème}$ neurone de la $p^{ème}$ couche accepte les entrées réelles $v_j^{(p-1)}(n)$, $\left(j = 1,2,.....N_{p-1}\right)$ et génère une sortie exprimée par :

$$s_i^{(p)}(n) = \sum_{j=1}^{N_{p-1}} w_{ij}^{(p)}(n) v_j^{(p-1)}(n) + I_i^{(p)}(n) \tag{4.18}$$

$$v_i^{(p)}(n) = f\left(s_i^{(p)}(n)\right) \tag{4.19}$$

Les ajustements des poids $w_{ij}^{(p)}(n)$ et les biais $I_i^{(p)}(n)$ dans l'algorithme BP s'écrivent comme suit :

$$w_{ij}^{(p)}(n) = -\eta^{(p)} . \nabla_{\Delta w_{ij}^{(p)}(n)}\left(\zeta_i^{(p)}(n)\right) + w_{ij}^{(p)}(n-1)$$

$$= \eta^{(p)} . \delta_i^{(p)}(n) . v_j^{(p-1)}(n) + w_{ij}^{(p)}(n-1) \tag{4.20}$$

$$I_i^{(p)}(n) = -\beta^{(p)} . \nabla_{\Delta I_i^{(p)}(n)}\left(\zeta_i^{(p)}(n)\right) = \beta^{(p)} . \delta_i^{(p)}(n) \tag{4.21}$$

Où $\zeta_i^{(p)}(n)$ la fonction de coût à la sortie du *i*ème neurone de la *p*ème couche, cette fonction est donnée par l'équation (4.22), $\nabla_x(\ )$ est le gradient de $\zeta_i^{(p)}(n)$.

$$\zeta_i^{(p)}(n) = \left|e_i^{(p)}(n)\right|^2 \tag{4.22}$$

$$e_i^{(p)}(n) = x(n-d) - v_i^{(p)}(n) \tag{4.23}$$

Où $v_i^{(p)}(n)$ est la sortie du *i*ème neurone de la *p*ème couche, et $x(n-d)$ les données transmises retardées.

La fonction delta $\delta_i^{(p)}(n)$ dans l'équation (4.20) et l'équation (4.21) peut être définie comme :

$$\delta_i^{(p)}(n) = -\nabla_{s_i^{(p)}(n)}\left(\zeta_i^{(p)}(n)\right) = e_i^{(p)}(n).\left(1-\left(v_i^{(p)}(n)\right)^2\right) \tag{4.24}$$

En outre, la formule pour la $(p\text{-}1)^{\text{ème}}$ couche peut être obtenue comme :

$$\delta_i^{(p-1)}(n) = \left(\sum_{j=1}^{N_p}\delta_j^{(p)}(n)\,w_{ji}^{(p)}(n)\right).v_i^{'(p-1)}(n) \tag{4.25}$$

Où $v_i^{'(p-1)}(n)$ est la dérivée de $v_i^{(p-1)}(n)$.

$$w_{ij}^{(p-1)}(n) = \eta^{(p)}.\delta_i^{(p-1)}(n).v_j^{(p-2)}(n) + w_{ij}^{(p-1)}(n-1) \tag{4.26}$$

$$I_i^{(p-1)}(n) = \beta^{(p)}.\delta_i^{(p-1)}(n) \tag{4.27}$$

Les formules pour la nouvelle stratégie NHBP (voir figure 4.19) s'écrivent comme suit :

Pour le premier algorithme (BP1): les ajustements des poids et les biais peuvent être écrits comme :

$$\begin{aligned} w_{ij}^{(1)}(n) &= -\eta^{(1)}.\nabla_{w_{ij}^{(1)}(n)}\left(\xi_i^{(1)}(n)\right) + w_{ij}^{(1)}(n-1) \\ &= -\eta^{(1)}.\nabla_{s_i^{(1)}(n)}\left(\xi_i^{(1)}(n)\right).\frac{\partial s_i^{(1)}(n)}{\partial w_{ij}^{(1)}(n)} + w_{ij}^{(1)}(n-1) \\ &= \eta^{(1)}.\delta_i^{(1)}(n).v_j^{(0)}(n) + w_{ij}^{(1)}(n-1) \end{aligned} \tag{4.28}$$

$$\begin{aligned} I_i^{(1)}(n) &= -\beta^{(1)}.\nabla_{I_i^{(1)}(n)}\left(\xi_i^{(1)}(n)\right) \\ &= -\beta^{(1)}.\nabla_{s_i^{(1)}(n)}\left(\xi^{(1)}(n)\right).\frac{\partial s_i^{(1)}(n)}{\partial I_i^{(1)}(n)} \\ &= \beta^{(1)}.\delta_i^{(1)}(n) \end{aligned} \tag{4.29}$$

La fonction delta peut être évaluée comme :

$$\delta_i^{(1)}(n) = -\nabla_{s_i^{(1)}(n)}\left(\xi_i^{(1)}(n)\right) = e_i^{(1)}(n).\left(1-\left(v_i^{(1)}(n)\right)^2\right) \tag{4.30}$$

Pour le deuxième algorithme (BP2): les ajustements des poids $w_j^{(3)}(n)$ et les biais $I^{(3)}(n)$ peuvent être écrits comme suit :

$$\begin{aligned} w_j^{(3)}(n) &= -\eta^{(3)}.\nabla_{w_j^{(3)}(n)}\left(\xi^{(3)}(n)\right) + w_j^{(3)}(n-1) \\ &= -\eta^{(3)}.\nabla_{s^{(3)}(n)}\left(\xi^{(3)}(n)\right).\frac{\partial s^{(3)}(n)}{\partial w_j^{(3)}(n)} + w_j^{(3)}(n-1) \\ &= \eta^{(3)}.\delta^{(3)}(n).v_j^{(2)}(n) + w_j^{(3)}(n-1) \end{aligned} \tag{4.31}$$

$$I^{(3)}(n) = -B^{(3)}.\nabla_{I^{(3)}(n)}\left(\xi^{(3)}(n)\right)$$

$$= -B^{(3)}.\nabla_{s^{(3)}(n)}\left(\xi^{(3)}(n)\right).\frac{\partial s^{(3)}(n)}{\partial I^{(3)}(n)}$$

$$= B^{(3)}.\delta^{(3)}(n) \qquad (4.32)$$

La fonction delta $\delta^{(3)}(n)$ peut être évaluée comme :

$$\delta^{(3)}(n) = -\nabla_{s^{(3)}(n)}\left(\xi^{(3)}(n)\right) = e^{(3)}(n).\left(1-\left(v^{(3)}(n)\right)^2\right) \qquad (4.\ 33)$$

Les ajustements des poids $w_{ij}^{(2)}(n)$ et les biais $I_i^{(2)}(n)$ peuvent être écrits comme :

$$w_{ij}^{(2)}(n) = \eta^{(3)}.\delta_i^{(2)}(n).v_j^{(1)}(n) + w_{ij}^{(2)}(n-1) \qquad (4.34)$$

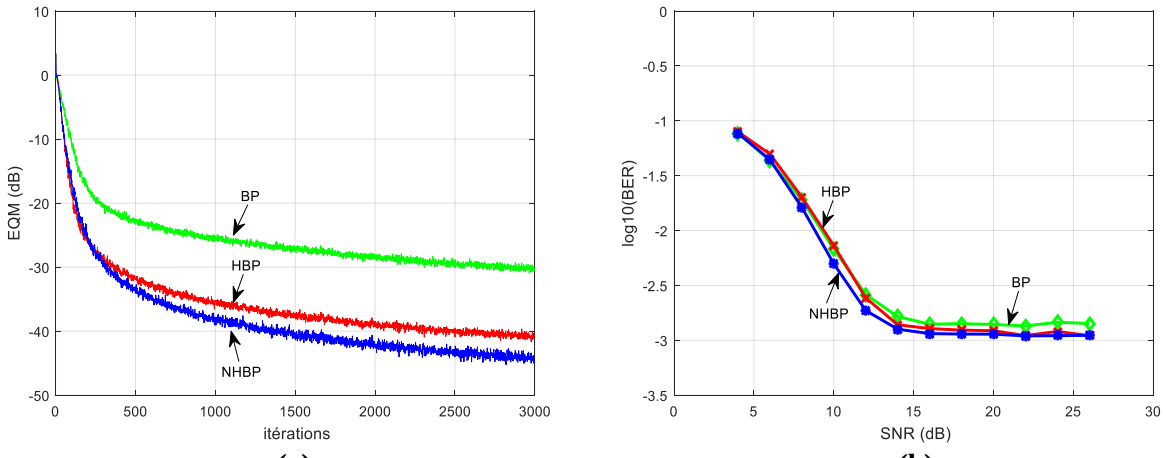
$$I_i^{(2)} = B^{(3)}.\delta_i^{(2)}(n) \qquad (4.35)$$

La fonction delta $\delta_i^{(2)}(n)$ peut être évaluée comme :

$$\delta_i^{(2)}(n) = \delta^{(3)}(n)\,w_i^{(3)}.v_i^{'(2)}(n) \qquad (4.36)$$

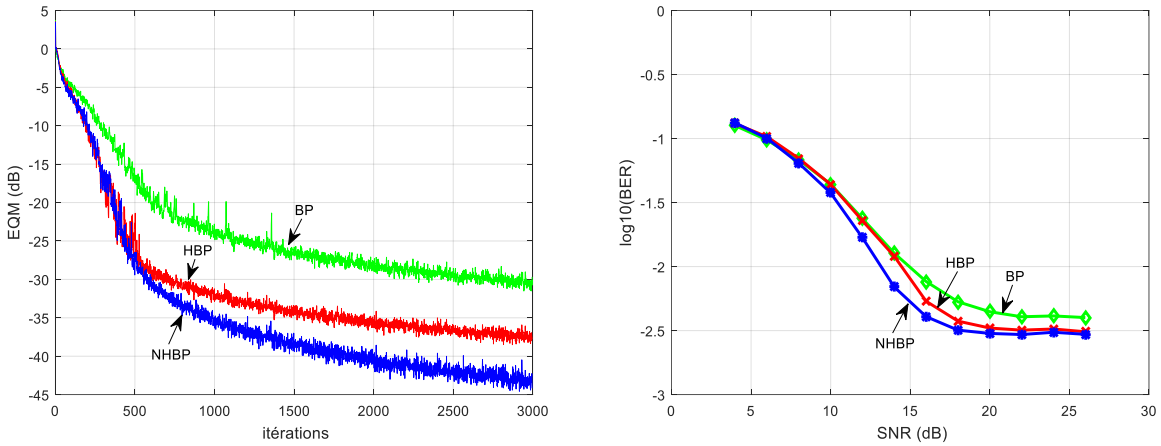
Où $v_i^{'(2)}(n)$ est la dérivée de $v_i^{(2)}(n)$.

La figures 4.20(a), montre les courbes de l'erreur quadratique moyenne des trois algorithmes dans le canal $H_1(z)$, ces courbes montrent une amélioration dans le temps de convergence et aussi dans le niveau MSE lors de l'utilisation de la méthode proposée. Il est également clairement montré que la nouvelle stratégie NHBP présente un niveau MSE plus faible que la stratégie d'apprentissage HBP. Comme le montre la figure 4.21(a), une amélioration similaire est également obtenue pour le canal $H_2(z)$ malgré sa distorsion sévère par rapport au canal $H_1(z)$.



(a) **(b)**

Figure. 4.20 (a) *courbes de l'erreur quadratique moyenne,* (b) *courbes BER du canal $H_1(z)$.*



(a) **(b)**

Figure. 4.21 (a) *courbes de l'erreur quadratique moyenne,* (b) *courbes BER du canal $H_2(z)$.*

Les figures 4.20(b) et 4.21(b) illustrent les courbes du BER (Bit Error Rate) des trois algorithmes pour les deux canaux $H_1(z)$ et $H_2(z)$, respectivement, ces courbes montrent que la méthode proposée présente de bons résultats dans un environnement fortement bruité par rapport aux autres méthodes. Ces courbes sont obtenues en prenant en moyenne 100 simulations indépendantes de longueur 10000 symboles. Les 2000 premiers symboles sont utilisés pour l'apprentissage et les autres sont utilisées pour le test. Les résultats indiquent que la méthode proposée donne un BER plus faible que les autres méthodes.

4.6 CONCLUSION

Nous avons utilisé, dans ce chapitre, les réseaux de neurones dans la fonction d'égalisation. Nous avons examiné la fonctionnalité et les performances de ces égaliseurs en tenant compte du type du réseau de neurones et l'algorithme d'apprentissage utilisés.

Les réseaux de neurone constituent un outil puissant dans le domaine du filtrage adaptatif non linéaire, pour les communications numériques. Beaucoup de recherches se sont consacrés à développés des techniques permettant d'accélérer le taux d'apprentissage de ces réseaux. Dans ce contexte, nous avons proposé des améliorations des capacités d'apprentissage des filtres égaliseurs adaptatifs à base du perceptron multicouche. Notre contribution est focalisée plus particulièrement au niveau de la structure du filtre à base du MLP et à l'algorithme de rétro-propagation du gradient. La performance de la nouvelle méthode est évaluée avec deux couches cachées et il est démontré qu'une amélioration de la performance est obtenue par rapport à d'autres méthodes.

CHAPITRE 5

ALGORITHMES META-HEURISTIQUE POUR L'EGALISATION : HYBRIDATION ET METHODE PROPOSEE

5.1 INTRODUCTION

Les algorithmes d'optimisation méta-heuristiques sont des méthodes stochastiques qui visent à résoudre un large panel de problèmes, sans pour autant que l'utilisateur ait à modifier leur structure. Ces méthodes sont inspirées d'analogies avec des domaines aussi variés que la physique, la génétique ou encore l'éthologie. Les méta-heuristiques ont vite rencontré un vif succès grâce à leur simplicité d'emploi mais aussi à leur forte modularité. Ces méthodes ont une grande capacité à trouver l'optimum global du problème. Contrairement à la plupart des méthodes déterministes, elles ne nécessitent ni point de départ, ni la connaissance du gradient de la fonction objectif pour atteindre la solution optimale. On sépare tout d'abord les méthodes d'optimisation en deux grandes familles de méthodes : les méthodes exactes et les méthodes approchées. Les méthodes exactes sont connues par le fait qu'elles garantissent l'optimalité de la solution mais elles sont très gourmandes en termes de temps de calcul et de l'espace mémoire nécessaire. La nécessité de disposer d'une solution de bonne qualité (semi optimale) avec un coût de recherche (temps de calcul et espace mémoire) raisonnable a conduit les chercheurs à proposer un autre type de méthodes d'optimisation, communément connues par les méthodes approchées (voir figure 5.1). De nombreuses méthodes approchées ont été proposées. Ces méthodes sont souvent classées en deux catégories: des méthodes heuristiques et des méthodes méta-heuristiques **[49]**.

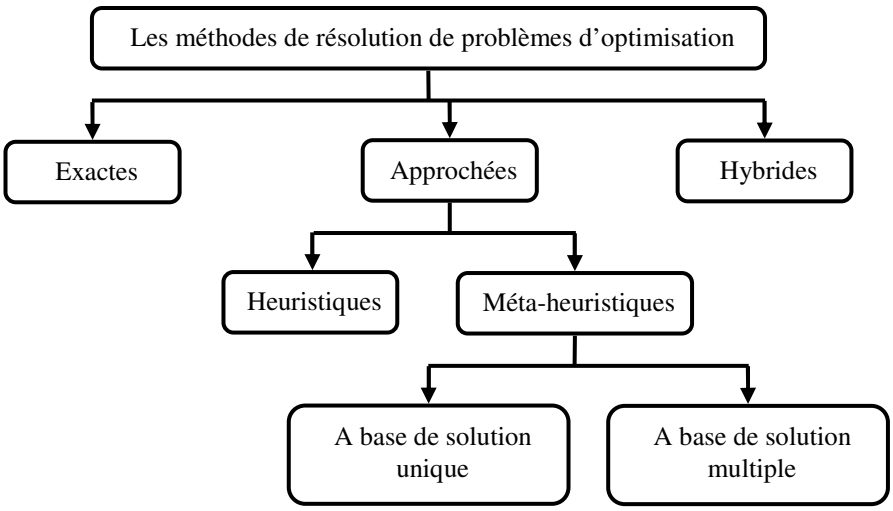


Figure 5.1 *Classification de méthodes de résolution de problèmes d'optimisation.*

Les heuristiques ont rencontré des difficultés pour avoir une solution réalisable et de bonne qualité aux problèmes d'optimisation difficiles, pour cela les méta-heuristiques ont fait leur apparition. Ces algorithmes sont plus complets et complexes qu'une simple heuristique, et permettent d'explorer l'espace de recherche efficacement afin de déterminer des solutions (presque) optimales **[50]**.
Les méta-heuristiques peuvent être classées en deux catégories: les méthodes à base de solution unique et les méthodes à base de population de solutions (ou à base de solution multiple). Les méthodes à base de solution unique se basent généralement sur une recherche par voisinage, malheureusement la plupart de ces méthodes souffrent du problème de la convergence vers l'optimum local. Cependant, les méthodes à base de population de solutions se basent sur une recherche globale sur tout l'espace de recherche. En fait, elles permettent d'échapper au problème de l'optimum local et de déterminer l'optimum global.
L'idée de combiner des méthodes exactes et/ou des méthodes approchées pour créer de nouvelles méthodes a donné naissance à un pseudo classe de méthodes. C'est la classe des méthodes hybrides. Ces dernières ont construit une tendance qui a suscité l'intérêt de plusieurs communautés de chercheurs. Aucune méthode n'est efficace pour tous les types de problèmes. En fait, chaque méthode propose des avantages dont on cherche à maximiser et des lacunes dont on cherche à combler. Partant de ce principe, beaucoup de chercheurs ont envisagé la combinaison des méthodes de résolution des problèmes afin de tirer profit des points forts de chacune et de proposer des alternatives plus efficaces et plus performantes. On distingue trois types d'hybridations: des hybridations entre des méthodes exactes, des hybridations entre des méthodes approchées, des hybridations entre des méthodes exactes et des méthodes approchées **[49]**.
Ce chapitre est consacré pour la description de la méthode hybride proposée pour améliorer les performances de l'égaliseur à base des réseaux de neurones et aussi à donner la notion de l'hybridation.

5.2 HYBRIDATION

L'hybridation est une tendance observée dans de nombreux travaux réalisés sur les algorithmes d'optimisation ces dernières années. L'hybridation consiste à combiner les caractéristiques de deux méthodes différentes pour tirer les avantages des deux méthodes, elles peuvent être vues comme la solution parfaite vis-à-vis des désavantages et des méthodes locales et des méthodes globales. Au fait, les méthodes locales mènent à une solution locale. Tandis que les méthodes globales consomment énormément de temps, il parait claire qu'un compromis entre exploitation et exploration doit être trouvé d'où l'utilité des méthodes hybrides. Quand l'hybridation est basée sur une vraie maîtrise de l'idée derrière chacune des méthodes candidates, l'augmentation de la précision ainsi que la diminution du temps de calcul est assuré **[42]**.

Actuellement, les méta-heuristiques hybrides sont devenues plus populaires car les meilleurs résultats trouvés pour plusieurs problèmes d'optimisation ont étés obtenus avec des algorithmes hybrides.

L'hybridation des méta-heuristiques peut être divisée en deux grandes parties : hybridation des méta-heuristiques avec des méta-heuristiques et hybridation des méta-heuristiques avec des méthodes exactes.

5.2.1 Hybridation méta-heuristiques/ méta-heuristiques

Selon les taxinomies existantes, dans divers domaines, l'hybridation des méta-heuristiques entre elles se fait en deux classifications principales. Une classification hiérarchique et une classification à plat. La figure 5.2 représente la structure de cette taxinomie.

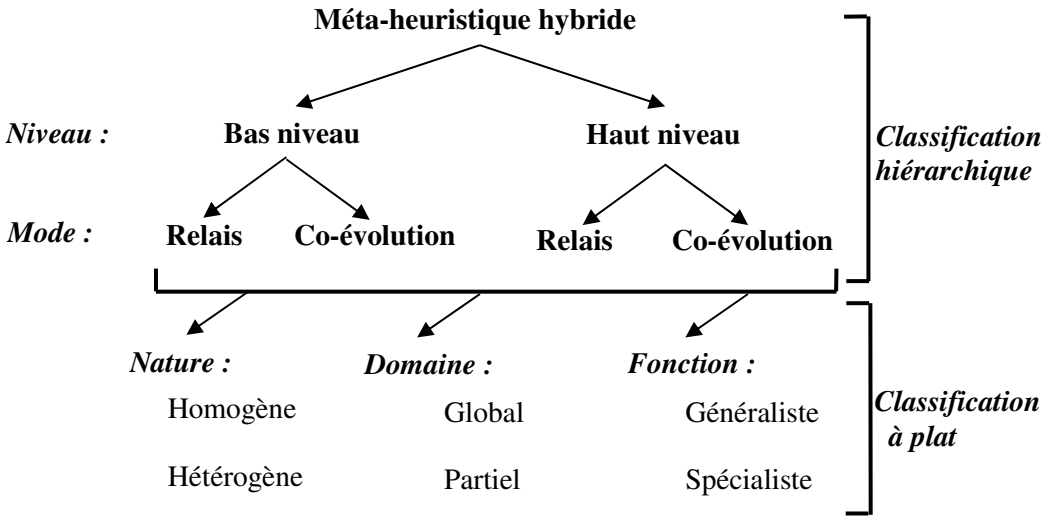


Figure 5.2 *Taxinomie de l'hybridation des méta-heuristiques.*

5.2.1.1 La classification hiérarchique

La classification hiérarchique est caractérisée par le niveau et le mode de l'hybridation. Le premier pas dans la classification sépare différents niveaux d'hybridation, on distingue les hybridations de bas niveau (Low-Level) et les hybridations de haut niveau (High-Level). Dans l'hybridation de bas niveau, une fonction interne d'une méta-heuristique est remplacée par une autre méta-heuristique. Par contre, dans l'hybridation de haut niveau, chaque méta-heuristique garde sa propriété au cours de l'hybridation. Il n'y a pas de relation directe entre les mécanismes internes des méta-heuristiques hybridées. La figure 5.3 illustre la notion de niveau pour l'hybridation de deux méta-heuristiques **[132]**.

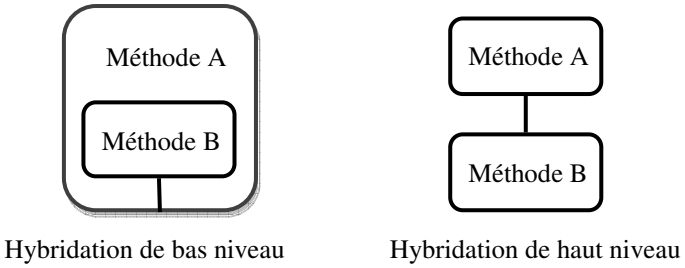


Figure 5.3 *Le niveau d'hybridation. Dans l'hybridation de bas niveau, la méthode est modifiée, ici la méthode B devient un élément fonctionnel de A. Pour l'hybridation de haut niveau, les méthodes A et B ne sont pas modifiées.*

Chaque niveau d'hybridation engendre deux modes de coopération à savoir, le mode relais et le mode co-évolutionnaire. Dans le mode relais, les méthodes sont exécutées les unes après les autre dans un ordre prédéterminé, c'est-à-dire le résultat de la première méthode est le début de la méthode suivante. Quand les différentes méthodes fonctionnent en parallèle pour explorer l'espace de recherche, on parle de mode co-évolutionnaire. Dans ce mode, chaque méthode hybridée conduit une recherche dans un espace de solution donné.
Selon la classification hiérarchique, nous obtenons, à partir de la combinaison des modes et des niveaux, quatre classes d'hybridation qui sont : l'hybridation relais de bas niveau, l'hybridation co-évolutionnaire de bas niveau, l'hybridation relais de haut niveau et l'hybridation co-évolutionnaire de haut niveau.

- **L'hybridation relais de bas niveau** (LRH : *Low-level Relay Hybrid*), elle englobe les méta-heuristiques à base de solution unique dans lesquelles une autre méthode est incorporée pour former un nouvel algorithme. Un exemple de cette classe est réalisé dans **[150]**, les auteurs combine un recuit simulé (SA : *Simulated Annealing*) et une méthode de descente déterministe (SDW : *Steepest Descend Walk*) pour résoudre le problème du voyageur de commerce (TSP : *Traveling Salesman Problem*). Le principe général de leur algorithme hybride est de limiter la recherche du recuit simulé à l'espace des optima locaux par l'intégration de la méthode de descente déterministe au recuit simulé.
- **L'hybridation Co-évolutionnaire de bas niveau** (LCH : *Low-level Co-evolutionary Hybrid*), consiste à incorporer une ou plusieurs méta-heuristiques à base de solution unique dans une méta-heuristique à population de solutions. L'avantage de ce type d'hybridation est de compenser la puissance d'exploitation d'une recherche locale et celle d'exploration d'une recherche globale **[42]**. Les méta-heuristiques basées sur une population de solutions sont plutôt performantes pour l'exploration de l'espace de recherche alors qu'elles sont beaucoup moins pour l'exploitation des solutions qu'elles ont trouvées. C'est pourquoi, la plupart de ces algorithmes ont été hybridées avec des méthodes à solution unique qui elles sont des méthodes de recherche performantes en terme d'exploitation. Ainsi, l'hybride LCH réalise l'équilibre exploration/exploitation en hybridant deux algorithmes complémentaires, l'un optimise localement et l'autre optimise d'une manière globale.
- **L'hybridation relais de haut niveau** (HRH : *High-level Relay Hybrid*), elle a lieu lorsque les méta-heuristiques sont utilisées de manière séquentielle c'est-à-dire la solution finale de la première méta-heuristique est la solution initiale de la méta-heuristique suivante. Dans cette procédure, toutes les méthodes gardent leur intégrité. Par exemple, on sait que les algorithmes à population de solutions ne parviennent pas à ajuster finement les solutions proches des bons optima. Mais, à l'inverse, leur force réside dans la capacité de trouver rapidement des régions de bonne qualité, même pour des espaces de recherche très complexes. Une fois ces régions repérées, il peut être intéressant de poursuivre la recherche

en affinant les solutions performantes qui s'y trouvent, pour cela, on applique une recherche itérative à solution unique après la recherche itérative à solution multiple.

- **L'hybridation Co-évolutionnaire de haut niveau** (HCH : *High-level Co-evolutionary Hybrid*), la structure interne des méta-heuristique hybridées n'est pas modifiée. Les méta-heuristiques utilisées travaillent simultanément (en parallèle) en échangeant des informations entre elles afin de trouver la solution optimale du problème posé. Un exemple d'hybridation HCH est le modèle des algorithmes génétiques en îles **[132]**. Dans ce modèle, la population est partagée en petites sous-populations géographiquement éloignées. Un GA fait évoluer chaque sous-population et les individus peuvent migrer d'une sous-population à l'autre. Plusieurs paramètres configurent cet hybride : la topologie des connexions entre les sous-populations, le nombre d'individus qui migrent, la fréquence de la migration et la stratégie du remplacement.

5.2.1.2 La classification à plat

La classification à plat des méta-heuristiques est caractérisée par le type des méthodes hybridées, leur domaine d'application et la nature de leurs fonctions. Selon le type d'hybridation, on trouve des méthodes hybridées homogènes et des méthodes hybridées hétérogènes. Un hybride est dit homogène lorsque les méthodes hybridées se basent sur la même méta-heuristique. Par exemple, le modèle d'algorithme génétique en iles est un hybride homogène. En général, on utilise des paramétrages différents pour chaque méta-heuristique hybridée **[132]**. Un hybride est dit hétérogène lorsque les méta-heuristiques utilisées sont différentes (voir figure 5.4).

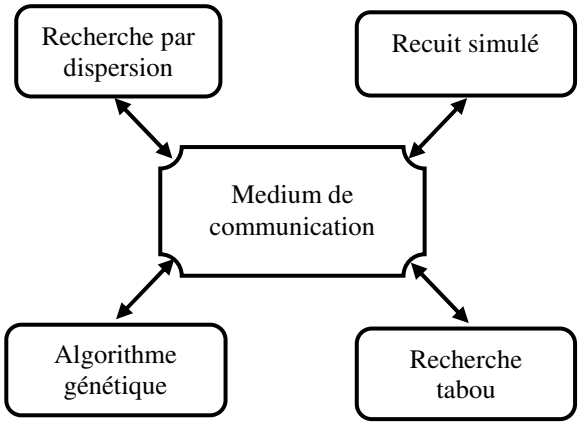


Figure 5.4 *Hybridation HCH hétérogène. Plusieurs algorithmes de recherche différents coopèrent pour résoudre le problème.*

Le domaine d'application des méta-heuristiques hybridées permet de distinguer deux grandes classes d'hybridation, les hybridations globales et les hybridations partielles. L'hybridation globale a lieu lorsque toutes les méthodes hybridées sont appliquées à la totalité de l'espace de recherche. L'objectif de ce type d'hybridation est d'explorer l'espace de recherche plus intensément. Pour l'hybridation partielle, le problème à

résoudre est décomposé en sous problèmes, ayant chacun un espace de recherche propre. Ainsi, chaque méta-heuristique de l'hybride résout un sous-problème dans son propre espace de recherche. Selon le problème traité, on distingue deux types d'hybridation, une hybridation généraliste et une hybridation spécialiste. On parle d'hybridation généraliste quand toutes les méta-heuristiques hybridées traitent le même problème d'optimisation. A l'inverse, les hybridations spécialistes ont lieu lorsque chaque méta-heuristique traite un problème différent. Un exemple de ce type est l'utilisation d'une méta-heuristique pour initialiser les paramètres d'une autre méta-heuristique.

5.2.2 Hybridation méta-heuristiques/méthodes exactes

L'hybridation des méta-heuristiques avec les méthodes exactes a été moins usuelle que l'hybridation méta-heuristique/méta-heuristique car la plupart des chercheurs la considérait assez inutile. En ces derniers temps, cette hybridation commence à s'étendre et un grand nombre d'articles a été publié concernant cette étude. Dans **[133]**, Talbi a généralisé sa taxinomie aux méthodes exactes de sorte que la classification hiérarchique est applicable à ce genre d'hybridation. La classe d'hybridation relais de bas niveau (LRH) est plus efficace en ayant une méta-heuristique hybridée avec une méthode exacte. La classe d'hybridation Co-évolutionnaire de bas niveau (LCH) regroupe une méta-heuristique à base de population dont laquelle un opérateur est remplacé par une méthode exacte. La classe LCH concerne aussi toutes les méta-heuristiques basées sur l'exploration du voisinage où une méthode exacte intervient pour trouver la meilleure solution du voisinage. Dans la classe d'hybridation relais de haut niveau (HRH), les méta-heuristiques et les méthodes exactes hybridées sont réalisées séquentiellement en gardant leur propriété. L'hybridation Co-évolutionnaire de haut niveau (HCH) est difficile à réaliser entre une méthode exacte et une méta-heuristique car chaque approche résout un problème différent puis, un échange d'information est indispensable entre elles.

Pour la classification à plat, Talbi **[133]** considère que les mêmes étapes qu'en hybridation méta-heuristique/méta-heuristique sont applicables à l'hybridation méta heuristique/méthodes exactes. Dans **[134]**, les auteurs proposent une autre classification pour l'hybridation des méta-heuristiques avec les méthodes exactes. Ils divisent les méthodes hybridées en deux grandes classes, les hybridations collaboratives et celle intégratives. Dans l'hybridation collaborative, les deux approches communiquent entre elles en échangeant des informations séquentiellement ou parallèlement. L'hybridation séquentielle a lieu quand une méthode exacte initialise une méta-heuristique ou vice-versa. L'hybridation collaborative parallèle a lieu quand la communication entre les méta-heuristiques et les méthodes exactes s'effectue parallèlement.

Dans l'hybridation intégrative, l'un des algorithmes (méta-heuristique ou exact) est intégré dans l'autre, c'est-à-dire qu'une méta-heuristique est intégrée dans une méthode exacte ou inversement. Pour résoudre un problème combinatoire de façon exacte, une méthode exacte est incorporée dans une méta-heuristique.

5.3 MÉTHODE PROPOSÉE

A la fin de l'état d'art sur l'hybridation, celle-ci est résumée en trois grands types : hybridation en série, l'hybridation parallèle ou hybridation d'insertion. La technique d'hybridation adoptée dans notre travail est celle de l'hybridation en insertion décrite par la suite. En effet, l'idée d'hybridation en insertion est venue pour améliorer les résultats de la fonction objectif. Cependant nous avons choisi d'hybrider les deux algorithmes méta-heuristiques décrits au deuxième chapitre : SFLA et DSO en insérant une partie du DSO dans le SFLA. En d'autre terme nous utilisons la stratégie de recherche de DSO pour l'exploration locale de l'algorithme SFLA. L'algorithme SFLA est considéré comme l'un des algorithmes les plus importants. Leurs applications ont montré leur efficacité pour résoudre divers problèmes d'optimisation dans différents domaines. Récemment, Panda et al. **[135]** ont utilisé cet algorithme pour optimiser la topologie de l'ANN dans l'égalisation de canal. Les résultats ont montré que cet algorithme était capable de trouver une meilleure solution comparativement à d'autres algorithmes. En outre, de nombreuses modifications sont proposées pour améliorer les performances de l'algorithme standard **[136-144]**. Les auteurs dans **[136]** ont présenté une version modifiée de l'algorithme SFLA (MSFLA) dans laquelle un facteur d'accélération C est introduit. Ce facteur doit être compris entre 1,3 et 2,1.

Dans cette partie nous allons proposer une nouvelle stratégie basée sur l'algorithme DSO pour améliorer la position de la plus mauvaise grenouille dans l'algorithme SFLA. L'algorithme DSO est utilisé par Tripathy et al. **[145]** comme un algorithme d'apprentissage d'un réseau de neurones à trois couches, ainsi utilisée par Panda et al. **[146]** pour le problème de l'égalisation. Pour améliorer les résultats de la fonction objectif, l'algorithme DSO utilise une stratégie de recherche efficace, supérieure à celle utilisée dans l'algorithme SFLA standard. Cependant la convergence de l'algorithme DSO est lente par rapport à l'algorithme SFLA. En effet, l'idée d'hybridation est venue pour donner une meilleure convergence (rapidité et proximité entre la solution optimale et la solution obtenue). Pour cela nous avons utilisé la stratégie de recherche du DSO pour l'exploration locale dans le standard SFLA, et pour la recherche globale, nous avons conservé la stratégie de l'algorithme SFLA. Dans la recherche locale de l'algorithme SFLA, la plus mauvaise grenouille X_w effectue un saut vers la meilleure X_b. Si le saut ne produit pas une meilleure solution, on applique la même règle en remplaçant cette fois X_b par la solution globale X_g. Pour l'algorithme proposé, la plus mauvaise grenouille est également améliorée sa position par un saut vers la meilleur grenouille X_b dans le sous-mèmeplexe et de la meilleure grenouille dans l'ensemble des grenouille. Cependant, l'algorithme hybride proposé, adopte une stratégie différente pour améliorer la position de la plus mauvaise grenouille (utilise la stratégie de recherche de DSO).

Les étapes de l'algorithme proposé sont résumées comme suit **[147]**:

Étape 1: Générez une population P composée d'un ensemble de grenouilles (chaque grenouille représente une solution au problème). La grenouille j est exprimée comme : $X_j = \left(X_j^1, X_j^2, ..., X_j^N\right)$ où N représente le nombre de variables.

Étape 2: Évaluez chaque solution. Ensuite, triez la population en fonction de sa solution (fitness) et déterminez la meilleure solution.

Étape 3: Partitionner la population en *m* mèmeplexes.

Etape 4: Recherche locale, pour chaque mèmeplexe de *n* grenouilles, il existe un sous-mèmeplexe contenant des grenouilles qui sont générées par une distribution de probabilité triangulaire donnée comme suit **[148]** :

$$P_j = \frac{2(n+1-j)}{n(n+1)}, \quad j = 1,.....,n \tag{5.1}$$

Où j est le nombre de la grenouille actuelle dans la population en fonction de la valeur de l'adaptabilité (fitness) dans l'ordre décroissant.

Étape 5: Déterminez la meilleure et la plus mauvaise position de grenouille. La position de la pire grenouille est amélioré en fonction de l'opération « mise à jour de position » utilisée par l'algorithme DSO; pour cela, nous utilisons trois paramètres qui sont: coefficient avant (α), coefficient arrière (β), probabilité de transfert (p_α).

La position est améliorée comme indiqué sur la figure 5.5. X_b^i représente le $i^{ème}$ $(i = 1,2,....,N)$ variable de la meilleure grenouille dans son sous-mèmeplexe. L'amélioration de la position X_w^i est déterminée par la probabilité de transfert (p_α). Notez que, si $rand(\ ) < p_\alpha$, la région avant sera le bon choix, sinon, la région arrière est prise en compte.

L'étape adaptative est donnée par l'équation suivante :

$$step^i = \left| X_b^i - X_w^i \right| \tag{5.2}$$

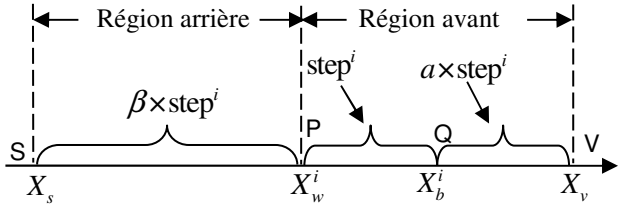


Figure 5.5 *Mise à jour de la position.*

La nouvelle position de la pire grenouille est mise à jour comme suit :

Pour chaque dimension $i \in [1,N]$ **Faire**

Si $rand(\ ) < p_\alpha$

$$X_v = X_w^i + (1+\alpha).(X_b^i - X_w^i) \tag{5.3}$$

$$X_w^i(new) = X_w^i + rand(\ ).(X_v - X_w^i), \ (-D_{\max} \le X_v \le D_{\max}) \tag{5.4}$$

Sinon

$$X_s = X_w^i - \beta.(X_b^i - X_w^i) \tag{5.5}$$

$$X_w^i(new) = X_w^i + rand(\ ).(X_s - X_w^i), \ (-D_{\max} \le X_s \le D_{\max}) \tag{5.6}$$

Fin Si
Fin Pour

Où X_b et X_w sont respectivement la meilleure solution et la mauvaise solution dans le sous-mèmeplexe. Le terme $rand(\ )$ est un nombre aléatoire dans l'intervalle $[0,1]$. $D_{\max}$ est le changement maximum autorisé de la position de la grenouille à chaque saut. Si la nouvelle position de la plus mauvaise grenouille n'est pas meilleure qu'auparavant, les calculs dans les équations (5.3), (5.4), (5.5) et (5.6) sont répétés en remplaçant X_b par X_g. Si cette opération ne peut toujours pas améliorer la position, dans ce cas, la position de la pire grenouille est généré aléatoirement par l'équation suivante :

$$X_w^i(new) = X_{\min}^i + rand(\ ).\left(X_{\max}^i - X_{\min}^i\right) \tag{5.7}$$

Où $X_{\max}$ et $X_{\min}$ représentent l'intervalle de recherche maximale et minimale. Après la mis à jour de la plus mauvaise grenouille dans chaque mèmeplexe. Les mèmeplexes sont mélangées pour améliorer l'échange de l'information globale.

5.4 APPLICATION Á L'ÉGALISATION

Dans cette section, nous présentons les résultats numériques obtenus par l'algorithme proposé, pour optimiser les poids d'un réseau de neurones appliqués à l'égalisation de canaux dans les systèmes de communication numériques. L'intérêt est de montrer que l'algorithme proposé donne des résultats meilleurs pour différents modèles des canaux non linéaires.

5.4.1 Modèle de système

Le modèle de système utilisé pour tester l'algorithme proposé est illustré à la figure 5.6. Une séquence binaire est transmise à travers un canal qui est modélisé comme un filtre RIF (Réponse Impulsionnelle Finie) avec une fonction de transfert donnée par :

$$H(z) = \sum_{i=0}^{N-1} h_i z^{-i} \tag{5.8}$$

La sortie du canal linéaire à l'instant k est exprimée par l'équation suivante:

$$z(k) = \sum_{i=0}^{N-1} h_i x(k-i) \tag{5.9}$$

Où $h_i\,(i = 0,1,.....N-1)$ est la réponse impulsionnelle du canal. Pour tester les performances de l'égaliseur dans un canal non linéaire, nous introduisons la non-linéarité (modélisée par un filtre de Volterra du troisième ordre) comme suit :

$$\hat{z}(k) = D_1 z(k) + D_2 z^2(k) + D_3 z^3(k) \tag{5.10}$$

Où D_i est le $i^{ème}$ coefficient non linéaire.

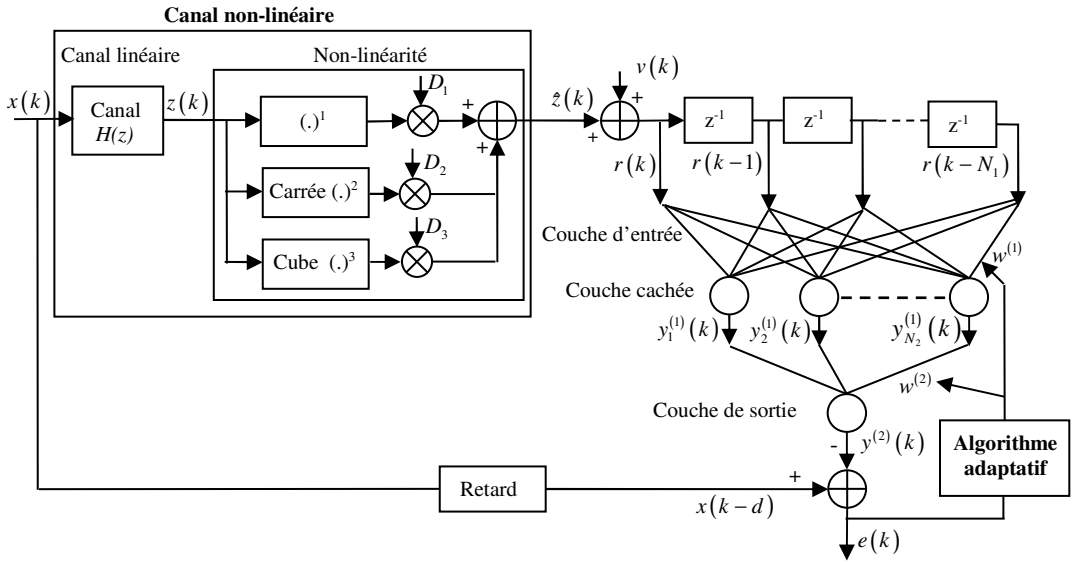


Figure 5.6 *Système de communication numérique avec un égaliseur* à base *de réseau de neurones*

La sortie du canal non linéaire $\hat{z}(k)$ est corrompue par un bruit additif blanc gaussien (BABG ou AWGN : Additive White Gaussian Noise). Les données reçues $r(k)$ sont données par :

$$r(k) = \hat{z}(k) + v(k) \tag{5.11}$$

La séquence $r(k)$ est ensuite envoyée à l'égaliseur à base de réseau de neurones.
La structure de l'égaliseur basé sur le réseau de neurones utilisé dans ce travail est illustrée à la figure 5.6. La structure est conçue pour être simple pour éviter la complexité de calcul et minimiser le facteur temps de décision. Dans ce cas, trois couches sont fixées : La couche d'entrée, une seule couche cachée et la couche de sortie. D'après la figure 5.6, on peut voir que le $j^{ème}$ neurone dans la couche cachée accepte les entrées réelles $r(k-i)$, $(i = 0,1,.....N_1)$ et génère une sortie exprimée par :

$$y_j^{(1)}(k) = f\left(\sum_{i=0}^{N_1} w_{ji}^{(1)}(k)\, r(k-i) + b_j^{(1)}(k)\right) \tag{5.12}$$

A partir de l'équation (5.12), la sortie de réseau de neurones représentée sur la figure 5.6 peut être décrite comme suit :

$$y^{(2)}(k) = h\left(\sum_{j=1}^{N_2} w_j^{(2)}(k) f\left(\sum_{i=0}^{N_1} w_{ji}^{(1)}(k) r(k-i) + b_j^{(1)}(k)\right) + b^{(2)}(k)\right) \tag{5.13}$$

Où $w_{ji}^{(1)}$ est le poids de connexion du $i^{ème}$ nœud de la couche d'entrée au $j^{ème}$ nœud de la couche cachée, $w_j^{(2)}$ est le poids de connexion du $j^{ème}$ nœud de la couche cachée au nœud de sortie de la couche de sortie et b est le niveau de seuil. Dans les expressions ci-dessus, h représente la fonction d'activation de sortie (linéaire dans ce cas) et f est

une fonction d'activation non linéaire de la couche cachée. Pour la fonction d'activation de la couche caché, notre choix s'est porté sur la fonction non linéaire tangente hyperbolique, car la plupart du temps, cette fonction converge rapidement par rapport aux fonctions sigmoïdes et logistiques. La fonction d'activation non linéaire utilisée (figure 5.7), est donnée par :

$$f(x) = \frac{2}{1+\exp(-2.x)} - 1 \tag{5.14}$$

L'algorithme d'apprentissage modifie les poids du réseau afin de minimiser une fonction de coût. La fonction de coût la plus fréquemment utilisée est l'erreur quadratique moyenne (EQM) donnée par :

$$MSE = \frac{1}{l}\sum_{i=1}^{l}\left(e_i(k)\right)^2 \tag{5.15}$$

Où l est le nombre d'échantillons transmis. L'erreur peut être écrite comme suite :

$$e(k) = x(k-d) - y(k) \tag{5.16}$$

Où $y(k)$ est la sortie de l'égaliseur et $x(k-d)$ les données transmises retardées.

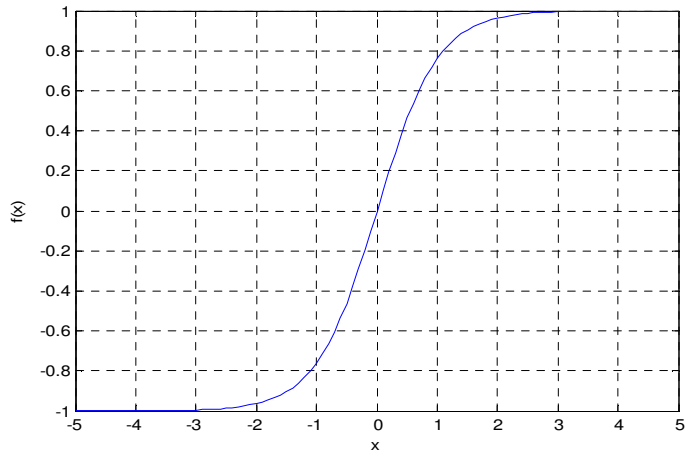


Figure 5.7 *Fonction tangente hyperbolique.*

5.4.2 Validation et comparaison avec d'autres méthodes

Dans cette section, La performance de l'algorithme proposée est maintenant examinée et comparée avec d'autres algorithmes d'apprentissage. Par conséquent, nous avons utilisé l'erreur quadratique moyenne (EQM) et le taux d'erreur binaire (TEB) pour évaluer les performances. Le réseau de neurones utilisé dans cette expérience a la structure de 3-*h*-1, où 3 et 1 sont respectivement le nombre d'entrées et de sorties, et *h* est le nombre de neurones dans la couche cachées. Notez que, l'expérience a été évaluée pour différents nombres de neurone dans la couche cachés à savoir 3, 4, 5 et 6, et nous avons choisi la taille de données préparée pour l'entraînement du réseau égale à 500. Les poids sont initialement générés de manière aléatoire dans l'intervalle [0, 1], tandis que les biais sont fixés à zéro **[149]**. La séquence de données transmise est constituée

des symboles (-1,1) uniformément distribués. Les performances ont été évaluées avec trois canaux différents possèdent les fonctions de transfert suivantes **[146]**:

$$H_1(z) = 0.26 + 0.93z^{-1} + 0.26z^{-2} \quad (5.17)$$

$$H_2(z) = 0.304 + 0.9029z^{-1} + 0.304z^{-2} \quad (5.18)$$

$$H_3(z) = 0.341 + 0.876z^{-1} + 0.341z^{-2} \quad (5.19)$$

La performance est testée en fixant les coefficients donnés dans l'équation (5.10), à 1, 0,1 et 0,05 respectivement **[91, 114]**. Les données reçues sont données par :

$$r(k) = z(k) + 0.1z^2(k) + 0.05z^3(k) + v(k) \quad (5.20)$$

Où $z(k)$ est la sortie de la partie linéaire du canal et $v(k)$ est le bruit blanc gaussien additif. Afin de comparer l'algorithme proposé avec d'autres méthodes, nous avons reproduit les trois algorithmes suivants:

- PSO proposé par Kennedy et Eberhart **[66]**. Utilisé dans cette section pour l'apprentissage de réseau de neurones (ANN-PSO). Les valeurs des coefficients C1 et C2 sont choisis les mêmes valeurs utilisées dans **[146]**, pour le problème d'égalisation de canal.
- DSO proposé par Zou et al **[82]**. Utilisé dans cette section pour l'apprentissage de réseau de neurones (ANN-DSO). Les paramètres de contrôle sont choisis de la même manière que dans **[82, 146]**.
- MSFLA proposé par Elbeltagi et al **[136]**. Utilisé dans cette section pour l'apprentissage de réseau de neurones (ANN-MSFLA). Pour le MSFLA, nous avons choisi C (facteur d'accélération) égal à 1,7 car il donne les meilleurs résultats.

Les performances de l'égaliseur avec les trois canaux différents et les différents algorithmes sont présentées dans le tableau 5.1 et la figure 5.8. Ces performances en termes d'EQM et taux d'erreurs binaires (BER) sont obtenues en prenant en moyenne 20 simulations indépendantes. Le nombre d'itérations utilisées pour chaque simulation est de 500 itérations. Les 100 premières itérations sont utilisées pour l'apprentissage et les autres sont utilisées pour le test. Les paramètres de simulation des algorithmes PSO, DSO et SFLA sont présentés dans le tableau 5.2.

Les résultats de l'EQM présentés sur la figure 5.8 (a, c et e) et le tableau 5.1 sont déterminés en choisissant un rapport signal sur bruit (SNR) de 15 dB.

D'après le tableau 5.1 (EQM ou MSE moyenne de 500 itérations), on peut voir que l'EQM obtenu par l'algorithme proposé est mieux que les trois autres algorithmes. L'EQM des deux égaliseurs ANN-MSFLA et ANN-DSO est mieux que l'EQM de ANN-PSO. Cependant, la performance de ANN-MSFLA est mieux que ANN-DSO pour les canaux $H_1(z)$ et $H_2(z)$ et c'est l'inverse pour le canal $H_3(z)$, c'est-à-dire la performance de ANN-DSO est mieux que ANN-MSFLA.

Tableau 5.1: *Comparaison des résultats de PSO, DSO, MSFLA et de l'algorithme proposée avec SNR = 15 dB.*

h	Stratégie	Canal $H_1(z)$		Canal $H_2(z)$		Canal $H_3(z)$	
		EQM	**BER**	**EQM**	**BER**	**EQM**	**BER**
3	ANN-PSO	0.0077	0.0015	0.0109	0.0018	0.0188	0.0029
3	ANN-DSO	0.0043	7.5843e-04	0.0060	8.1687e-04	0.0093	0.0018
3	ANN-MSFLA	0.0036	1.5502e-04	0.0051	5.6466e-04	0.0111	0.0020
3	**ANN-proposed**	**7.9909e-04**	**4.4177e-05**	**0.0018**	**1.6988e-04**	**0.0067**	**0.0010**
4	ANN-PSO	0.0112	0.0019	0.0149	0.0022	0.0263	0.0035
4	ANN-DSO	0.0071	9.1867e-04	0.0093	9.7229e-04	0.0141	0.0026
4	ANN-MSFLA	0.0045	1.6185e-04	0.0072	5.7289e-04	0.0205	0.0022
4	**ANN-proposed**	**9.9477e-04**	**6.5462e-05**	**0.0020**	**1.5161e-04**	**0.0080**	**0.0011**
5	ANN-PSO	0.0148	0.0024	0.0222	0.0026	0.0327	0.0038
5	ANN-DSO	0.0084	0.0010	0.0095	0.0014	0.0188	0.0026
5	ANN-MSFLA	0.0050	2.8534e-04	0.0077	6.5743e-04	0.0221	0.0021
5	**ANN-proposed**	**0.0017**	**9.7590e-05**	**0.0032**	**2.2490e-04**	**0.0072**	**0.0011**
6	ANN-PSO	0.0182	0.0025	0.0228	0.0027	0.0348	0.0047
6	ANN-DSO	0.0124	0.0011	0.0160	0.0016	0.0216	0.0031
6	ANN-MSFLA	0.0061	2.3253e-04	0.0149	7.8474e-04	0.0226	0.0026
6	**ANN-proposed**	**0.0019**	**1.8454e-04**	**0.0036**	**3.0803e-04**	**0.0087**	**0.0011**

Tableau 5.2: *Paramètres de simulation*

PSO		**DSO**		**SFLA**	
Paramètre	**Valeur**	**Paramètre**	**Valeur**	**Paramètre**	**Valeur**
Nombre d'itérations	500	Nombre d'itérations	500	Nombre d'itérations	500
Nombre de particules	25	Taille de la population	25	Taille de la population	25
Coefficient C1	0.7	Probabilité de transfert	0.8	Mèmeplexes	5
Coefficient C2	0.7	Coefficient avant	1	Mèmes en Mèmeplexe	5
		coefficient arrière	10	Mèmes en sous-Mèmeplexe	4
		Probabilité de mutation génétique	0.01		

Les figures 5.8 (a, c et e) montrent les courbes de convergence de l'égaliseur avec différents algorithmes. Ces courbes montrent une amélioration des performances en termes de niveau de l'EQM lors de l'utilisation de l'algorithme proposé.
Les figures 5.8 (b, d, f) et le tableau 5.1 montrent les performances en termes de BER des égaliseurs considérés. Le BER est défini comme suit **[135]** :

$$\text{BER}=\frac{\text{Nombre de bits erronés après l'équalization}}{\text{Nombre total de bits transmis}} \tag{5.21}$$

D'après le tableau 5.1, les résultats montrent que l'algorithme proposé présente de bons résultats par rapport aux autres algorithmes. Il est également clairement observé à partir des figures 5.8 (b, d et f) que l'algorithme proposé montre sa supériorité sur les autres algorithmes lorsque le SNR dépasse 6, 8 et 10 dB respectivement. En plus, d'après les tableaux 5.3, 5.4 et 5.5, nous remarquons que l'algorithme proposé (ANN-proposé) présente un BER plus faible par rapport aux autres algorithmes malgré le milieu fortement bruité (pour la petite valeur de SNR).

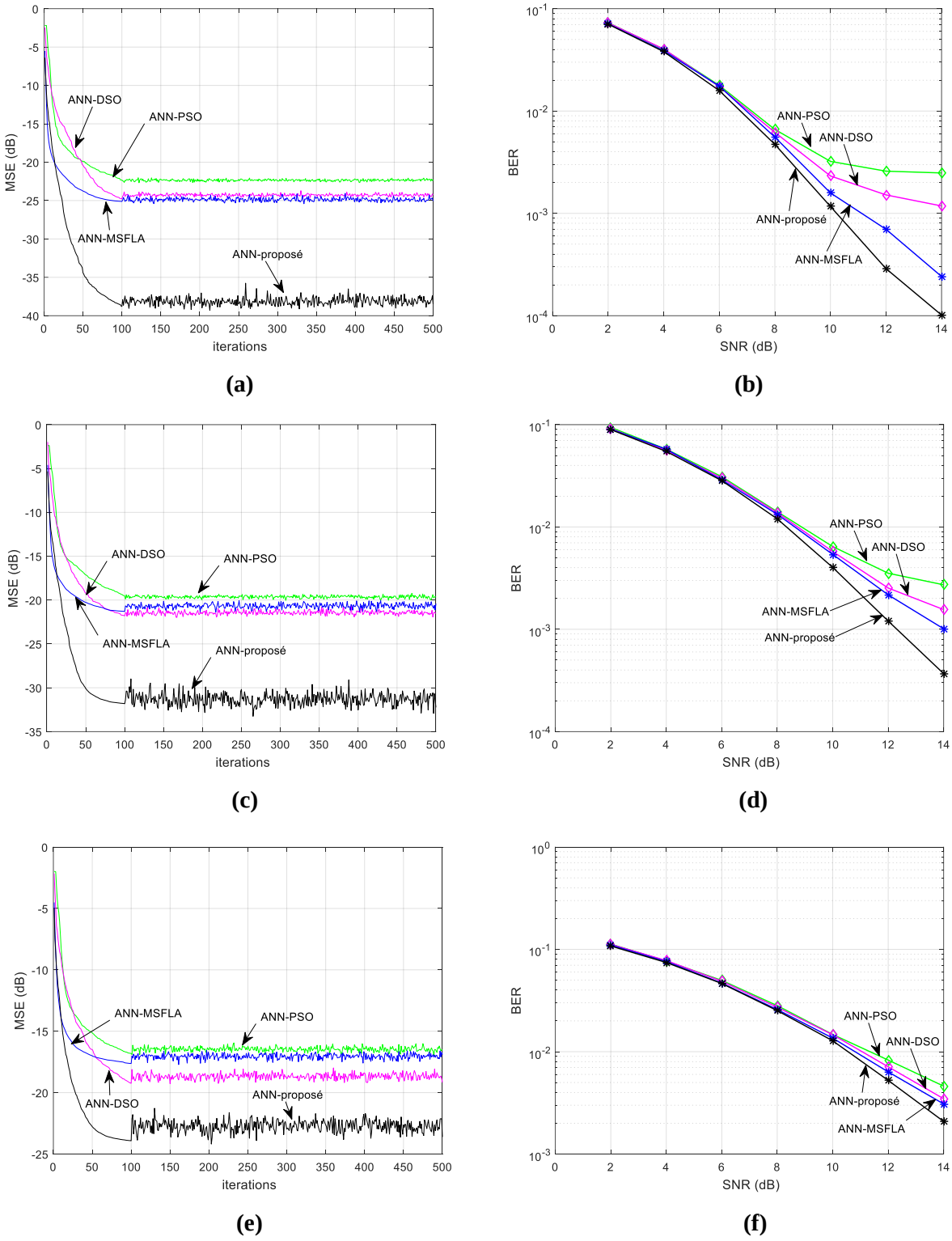


Figure 5.8: *Canal $H_1(z)$: courbes MSE (a) et courbes BER (b). Canal $H_2(z)$: courbes MSE (c) et courbes BER (d). Canal $H_3(z)$: courbes MSE (e) et courbes BER (f)*

On note que les courbes MSE et BER représentées sur la figure 5.8 sont réalisés à l'aide d'une structure de réseau de neurones contenant une seule couche cachée de cinq nœuds.

Tableau 5.3*: Résultats numériques du BER obtenus sur la figure 5.8(b) pour différentes valeurs de SNR*

algorithmes	**SNR(dB)**				
	2	**4**	**6**	**8**	**10**
ANN-PSO	0,07289	0,04008	0,01779	0,00661	0.00322
ANN-DSO	0,07254	0,04039	0,01732	0,00619	0,00232
ANN-MSFLA	0,07110	0,03881	0,01750	0,00556	0,00158
ANN-proposé	**0,07064**	**0,03805**	**0,01583**	**0,00472**	**0,00116**

Tableau 5.4*: Résultats numériques du BER obtenus sur la figure 5.8(d) pour différentes valeurs de SNR*

algorithmes	**SNR(dB)**				
	2	**4**	**6**	**8**	**10**
ANN-PSO	0,09399	0,05757	0,03098	0,01411	0.00643
ANN-DSO	0,09156	0,05615	0,03008	0,01380	0,00575
ANN-MSFLA	0,09040	0,05751	0,02887	0,01331	0,00538
ANN-proposé	**0,08953**	**0,05476**	**0,02837**	**0,01200**	**0,00402**

Tableau 5.5*: Résultats numériques du BER obtenus sur la figure 5.8(f) pour différentes valeurs de SNR*

algorithmes	**SNR(dB)**				
	2	**4**	**6**	**8**	**10**
ANN-PSO	0,11268	0,07689	0,04975	0,02820	0.01473
ANN-DSO	0,11185	0,07773	0,04882	0,02722	0,01465
ANN-MSFLA	0,11002	0,07596	0,04681	0,02583	0,01357
ANN-proposé	**0,10785**	**0,07438**	**0,04641**	**0,02520**	**0,01277**

5.5 CONCLUSION

Dans ce chapitre, nous avons présenté une taxinomie des modèles d'hybridation des algorithmes méta-heuristique. De plus nous avons présenté notre méthode d'optimisation des poids d'un réseau de neurones appliquée à l'égalisation de canaux dans les systèmes de communication numériques. Cette méthode est basée sur deux algorithmes qui sont: Algorithme de saut de grenouille (SFLA) et l'algorithme d'optimisation de recherche dirigée (DSO). La stratégie de la mise à jour dans la recherche locale de l'algorithme SFLA est remplacée par la stratégie de la mise à jour de l'algorithme DSO pour améliorer la convergence. La méthode proposée est comparée à l'optimisation standard d'essaims de particules (PSO), l'algorithme modifié de saut de grenouille (MSFLA) et l'algorithme DSO afin d'évaluer leurs performances pour l'égalisation de canaux non linéaires. Les résultats des simulations révèlent que la méthode proposée présente les meilleurs résultats et surpasse les autres algorithmes.

CONCLUSION GÉNÉRALE

Les travaux présentés dans cette thèse concernent l'égalisation adaptative dans les systèmes de communications numériques. Nous avons focalisé nos recherches sur les algorithmes d'apprentissage destinés à résoudre des problèmes d'égalisation des canaux.

Dans le premier chapitre, nous avons présenté les principales notions relatives aux systèmes de communication numérique. Nous avons présenté quelques généralités sur la chaîne de communication numérique. Les différents modèles mathématiques des canaux de transmission les plus rencontrés en pratique, sont également évoqués, avec les distorsions introduites par ces canaux, notamment l'interférence entre symboles qui a été bien détaillé en considérant qu'il constitue un problème dont son élimination fait l'objectif principal de l'égalisation. On a présenté aussi un aperçu sur les filtres numériques.

Dans le deuxième chapitre, nous avons présenté les définitions générales des méthodes d'optimisation qui se divisent en deux grandes classes : les méthodes déterministes et les méthodes non déterministes. Lorsque l'évolution de la méthode de résolution est prévisible et ne laisse aucune place au hasard, celle-ci est qualifiée de déterministe. Les méthodes déterministes s'appuient sur une direction de recherche qui peut être fournie par les dérivées de la fonction objectif. Ces méthodes ont la réputation d'être efficaces lorsque la solution initiale est proche de l'optimum recherché. Les méthodes non-déterministes ont comme caractéristiques communes leurs caractères stochastiques, c.à.d. qu'une partie de la recherche est conduit de façon aléatoire.

Dans le troisième chapitre, nous avons discuté le contexte d'égalisation des canaux de communication numériques et nous avons également mis en évidence certaines structures d'égaliseurs les plus communes, l'égaliseur LE et l'égaliseur DFE adaptés à l'aide des algorithmes LMS et RLS. Les méthodes classiques ont été présentées dans le but de mettre en évidence leurs performances médiocres pour un canal à phase non minimale, les niveaux de l'erreur quadratique moyenne (EQM) atteints sont sous optimales. Les égaliseurs conventionnels présentent une incapacité à égaliser les canaux non linéaires à faible SNR, d'où la nécessité des architectures présentant un traitement non linéaire pour améliorer les performances.

Dans le quatrième chapitre, nous avons utilisé les réseaux de neurones dans la fonction d'égalisation. Nous avons examiné la fonctionnalité et les performances de ces égaliseurs en tenant compte du type du réseau de neurones et l'algorithme d'apprentissage utilisé. Les réseaux de neurone constituent un outil puissant dans le domaine du filtrage adaptatif non linéaire, pour les communications numériques. Beaucoup de recherches se sont consacrés à développés des techniques permettant d'accélérer le taux d'apprentissage de ces réseaux.

Dans ce contexte, nous avons proposé des améliorations des capacités d'apprentissage des filtres égaliseurs adaptatifs à base du perceptron multicouche. Notre contribution est focalisée plus particulièrement au niveau de la structure du filtre à base du MLP et à l'algorithme de rétro-propagation du gradient. La performance de la nouvelle méthode est évaluée avec deux couches cachées et il est démontré qu'une amélioration de la performance est obtenue par rapport à d'autres méthodes.

Dans le cinquième chapitre, nous avons utilisés les algorithmes méta-heuristiques à base de population de solution pour optimisés les poids du réseau MLP. Les méta-heuristiques sont des méthodes stochastiques qui visent à résoudre un large panel de problèmes, sans pour autant que l'utilisateur ait à modifier leur structure. Ces méthodes sont inspirées d'analogies avec des domaines aussi variés que la physique, la génétique ou encore l'éthologie. Les méta-heuristiques ont vite rencontré un vif succès grâce à leur simplicité d'emploi mais aussi à leur forte modularité. On a vu dans ce chapitre que les méta-heuristiques sont facilement adaptables et/ou hybridables en vue d'obtenir les meilleures performances possibles. Nous avons présenté une nouvelle méthode d'optimisation des poids d'un réseau de neurones appliquée à l'égalisation de canaux dans les systèmes de communication numériques. Cette méthode est basée sur deux algorithmes qui sont: l'algorithme des sauts de grenouilles (SFLA) et l'algorithme d'optimisation de recherche dirigée (DSO). La stratégie de mise à jour dans la recherche locale de la standard SFLA est remplacée par la stratégie de mise à jour de l'algorithme DSO pour améliorer la convergence. La méthode proposée est comparée à l'optimisation standard d'essaims de particules (PSO), l'algorithme modifié de saut de grenouille (MSFLA) et l'algorithme DSO afin d'évaluer leurs performances pour l'égalisation de canaux non linéaires. Les résultats des simulations révèlent que la méthode proposée présente les meilleurs résultats et surpasse les autres algorithmes.

Les travaux effectués dans cette thèse ouvrent plusieurs perspectives pour les futurs travaux. Tout d'abord il est possible de travailler à développés des structures de réseaux de neurones permettant d'accélérer le taux d'apprentissage de ces réseaux. Il est aussi envisageable d'utiliser d'autres techniques d'optimisations intelligentes. Il est toujours possible d'étendre ce travail en utilisant les modulations à grand nombre d'états et avec d'autres type de canaux. Nous avons utilisé les méthodes supervisées celle-ci utilisent une partie de la bande passante ce qui entraine une diminution de l'efficacité spectrale et donc du débit. Ce travail peut être prolongé également sous un autre volé en considérant des algorithmes autodidactes dit (Non-data aided) basés sur les statistiques d'ordre supérieur à deux. Ces algorithmes jouissent d'une complexité remarquable, cependant économisent la bande passante et permettent d'augmenter l'efficacité des systèmes de communication.

RÉFÉRENCES

[1] **A. Berdai,** "Egalisation aveugle et turbo égalisation dans les canaux sélectifs en fréquence invariants et variants dans le temps", Mémoire Maître ès science, Université Laval, Québec, 2006.

[2] **S. Qureshi**, "Adaptive equalization", Proceeding of IEEE, vol. 73, no. 9, pp.1349-1387, Sep.1985.

[3] **H. Nyquist**, "Certain topics in telegraph transmission theory", Trans. AIEE (Commun. Electron), vol. 47, pp. 617-644, Apr. 1928.

[4] **R. W. Lucky**, "Automatic equalization for digital communication", Bell System Technical Journal, vol. 44, pp. 547-588, 1965.

[5] **J. G. Proakis, J. H. Miller**, "An adaptive receiver for digital signaling through channels with intersymbol interference", IEEE Trans. Inf. Theory, vol. 15, no. 4, pp. 484-497, 1969.

[6] **C. Macchi, J. P. Jouannaud, O. Macchi**, "Récepteur adaptatifs pour transmission de données à grande vitesse",Annals of Telecommunications, vol. 30, pp. 311-330, 1975.

[7] **M. Austin**, "Decision feedback equalization for digital communication over dispersive channels", M.I.T. Res. Lab Electron.,Tech. Rep. 461, Aug. 1967.

[8] **W. McCulloch, W. Pitts**, "A Logical Calculus of Ideas Immanent in Nervous Activity", Bulletin of Mathematical Biophysics, vol. 5, pp.115-133, 1943.

[9] **S. Chen, G.J. Gibson, C.F.N. Cowan**, "Decision feedback equalization using neural network structures and performance comparison with standard architecture", IEE Proceeding, Vol. 137, No. 4, pp. 221-225, August 1990.

[10] **S. Chen, B. Mulgrew, P. M. Grant**, "A Clustering Technique for Digital Communications Channel Equalization Using Radial Basis Function Networks", IEE Transactions on neural networks, vol. 4, no. 4, pp. 570-579, 1993.

[11] **P. Chandra Kumar, P. Saratchandran, N. Sundararajan**, "Minimal radial basis function neural networks for nonlinear channel equalisation", IEE Proceedings, Vision, Image and Signal processing, vol. 147, no. 5, pp. 428-435, 2000.

[12] **J. C. Patra, P. K. Meher, G. Chakraborty**, "Nonlinear channel equalization for wireless communication systems using Legendre neural networks", Signal Processing, vol. 89, pp. 2251-2262, 2009.

[13] **K. Burse, R. N. Yadav, S. C. Shrivastava**, "Channel Equalization Using Neural Networks: A Review", IEEE Transactions On Systems, Man, and Cybernetics-Part C: Applications and Reviews, vol. 40, no. 3, pp. 352-357, 2010.

[14] **B. Lu, B. L. Evans**, "Channel equalization by feed-forward neural network", IEEE International Symposium on circuit and Systems (ISCAS), 6 August, 2000.

[15] **D. E. Rumelhart, G. E. Hinton, R. J. Williams**, "Learning representations by back-propagation errors", Nature, 323, pp. 533–536, 1986.

[16] **B. Sklar,** "Digital Communications: Fundamentals and Applications", 2ème Edition, Prentice Hall PTR, Upper Saddle River, New Jersey, 2013.

[17] **F. Bouguerra**, "Contribution au développement d'algorithmes d'optimisation dans les réseaux mobiles", Thèse de doctorat, Université de Batna 2- Mostefa Ben Boulaid, 2018.

[18] **P. F. Combes**, "Transmission en espace libre et sur des lignes ", Dunod Université 1988.

[19] **H. SARI**, "Transmission des signaux numériques ", Technique de l'ingénieur, traité Télécoms, 1999.

[20] **M. Joindot, A. Glavieux**, "Introduction aux Communications Numériques", Dunod, 2007.

[21] **Z. Zerdoumi**, "Estimation des Filtres de Restauration des Signaux en Communication Numériques", Thèse de Doctorat, Université de Batna 2, 2018.

[22] **A. Boyer**, "Canaux de transmission bruités", Support de cours, Institut National des Sciences Appliquées de Toulouse. 2011.

[23] **G. Baudoin, Coll**, "Radiocommunications Numérique /1 : Principe, modélisation et simulation", 2e édition, Dunod, 2007.

[24] **J.G. Proakis**, "Digital Communication", Mc Graw Hill, 4th Edition, 2000.

[25] **J.G. Proakis, M. Salehi**, "Digital Communications", 5th Ed, New York: McGraw Hill, 2008.

[26] **S. Hayin**, "Communication systems", 4th Ed, New York: Wiley &sons, 2001.

[27] **A. Glavieux, M. joindot**, "Communications numériques" : Introduction, Paris, Masson, 1996.

[28] **P. Kadionik**, "Bases de transmission numériques : les modulations numériques", Support de cours, École nationale supérieure Électronique, Informatique et Radiocommunication, 2000.

[29] **M. Benyarou**, "Optimisation des systèmes multi-antennes appliqués aux systèmes MC-CDMA", Thèse de Doctorat, Université Abou bakr belkaid-TLEMCEN, 2013.

[30] **Z. Zerdoumi**, "Applications de Réseaux de Neurones Artificiels à la Poursuite des non Linéarités fluctuantes des systèmes satellitaire", Mémoire de Magister, Université Mohamed Boudiaf, M'sila, 2006.

[31] **N. L. Gallou**, "Modélisation par séries de Volterra dynamique des phénomènes de mémoire non linéaire pour la simulation des systèmes d'amplification de puissance", Thèse de doctorat, Université de Limoges, 2001.

[32] **M. Schetzen**, "The Volterra and Wiener Theories of nonlinear systems", Krieger Publishing Co., Inc. Melbourne, FL, USA, 2006.

[33] **DD. Falconer**, "Adaptive Equalization of Channel nonlinearities in QAM Data Transmission, Systems", Bell System Technical Journal, vol.57, no.7, pp.2589-2611, 1978.

[34] **E. Biglieri, A. Gersho, R. Gitlin, T. Lim**, "Adaptive Cancellation of nonlinear Intersymbol Interference for Voiceband Data Transmission", IEEE Journal on Selected Areas in Communications, vol. 2, no.5, pp.765-777, 1984.

[35] **E. Telatar**, "Capacity of Multi-antenna Gaussian Channels", European Transactions on Telecommunications, vol. 10, no. 6, pp. 585–595, 1999.

[36] **T. M Cover, J. A. Thomas**, "Elements of information theory", Second Edition, John Wiley & Sons, Inc, 2006.

[37] **John G. Proakis**, "Digital communications", McGraw-Hill, Third Edition, 1995.

[38] **I. Andriyanova,** "Introduction aux communications numériques", Support de cours, Université de Cergy-Pontoise, 2012.

[39] **M. Ghadjati**, "Applications des Réseaux de Neurones aux système de communication numériques", Mémoire de Magister, Université 8 Mai 1945 Guelma, 2012.

[40] **C. Laot**, "Egalisation autodidacte et turbo-égalisation. Application aux canaux sélectifs en fréquence", Thèse de doctorat, Université de Rennes 1, 1997.

[41] **S. Paulo, R. Diniz**, "Adaptive Filtering Algorithms and Practical Implementation," 4th Edition, Springer Science, 2013.

[42] **H. Hachimi**, "Hybridations d'algorithmes méta heuristiques en optimisation globale et leurs applications", Thèse de Doctorat, Ecole Mohammadia D'ingénieurs, Université Mohammed-V, Rabat, 2013.

[43] **I. Boussaid**, "Perfectionnement de méta heuristiques pour l'optimisation continue", Thèse de Doctorat, Université Houari Boumediene, 2013.

[44] **O. Hajji**, "Contribution au développement de méthodes d'optimisation stochastiques. Application à la conception des dispositifs électrotechniques ", Thèse de Doctorat, Université des sciences et technologies de lille, 2003.

[45] **J. C. Spall**, "Multivariate stochastic approximation using a simultaneous perturbation gradient approximation", IEEE transactions on automatic control, vol. 37, pp : 332-341, 1992.

[46] **J. C. Spall**, "An overview of the simultaneous perturbation method for efficient optimization", Johns Hopkins apl technical digest, vol. 19, 1998.

[47] **N. Ketfi**, "Contribution à la gestion des réseaux de distribution en présence de génération d'énergie dispersée", Thèse de Doctorat, Université El Hadj Lakhdar, Batna, 2014.

[48] **R. Moore,** "Interval analysis", Prentice Hall, NJ, 1966.

[49] **A. Gherboudj,** "Méthodes de résolution de problèmes difficiles académiques", Thèse de Doctorat, Université de Constantine 2, 2013.

[50] **B. Benmammar**, "Optimisation de la QoS dans un réseau de radio cognitive en utilisant la métaheuristique SFLA (Shuffled Frog Leaping Algorithm) ", [Rapport de recherche] UABT, (hal-01491952v2), 2017.

[51] **J. C. Boisson**, "Modélisation et résolution par méta-heuristiques coopératives : de l'atome à la séquence protéique", Thèse de Doctorat, Université des Sciences et Technologies de Lille, 2008.

[52] **F. Glover, M. Laguna**, "Tabu Search", Kluwer Academic Publishers, Boston.1997.

[53] **S. Mirjalili, A. Lewis**, "The Whale Optimization Algorithm, Advances in Engineering Software", vol. 95, pp. 51-67, 2016.

[54] **J. H. Holland**, "Genetic algorithms", Scientific American, vol. 267, no. 1, pp. 66-72, 1992.

[55] **I. Rechenberg**, "'Evolutions strategien". Springer Berlin Heidelberg, pp. 83-114, 1978.

[56] **D. Dasgupta, M. Zbigniew**, "Evolutionary algorithms in engineering applications". Springer Science & Business Media, 2013.

[57] **J. R. Koza**, "Genetic programming", MIT Press, 55 Hayward St, Cambridge, MA, United States, 1992.

[58] **D. Simon**, "Biogeography-based optimization', IEEE Trans Evol Comput, vol. 12, pp.702-713, 2008.

[59] **S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi**, "Optimization bysimmulated annealing", Science, vol. 220, pp. 671–680, 1983.

[60] **B. Webster, P. J. Bernhard**, "A local search optimization algorithm based on natural principles of gravitation", In Proceedings of the international conference on information and knowledge engineering (IKE'03), pp.255-61, 2003.

[61] **E. Rashedi, H. Nezamabadi, S. Saryazdi**, "GSA a gravitational search algorithm", Inf Sci, vol. 179, pp. 2232–2248, 2009.

[62] **A. Kaveh, S. Talatahari**, "A novel heuristic optimization method", charged system search.Acta Mech, vol. 213, pp. 267-89, 2010.

[63] **R.A. Formato**, "Central force optimization: A new metaheuristic with applications in applied electromagnetics", Prog Electromag Res, vol. 77, pp. 425-491, 2007.

[64] **B. A. Alatas**, "Artificial Chemical Reaction Optimization Algorithm for global optimization", Expert Syst Appl, vol. 38, pp. 13170-13180, 2011.

[65] **H. Shah-Hosseini**, "Principal components analysis by the galaxy-based search algorithm: a novel metaheuristic for continuous optimisation", Int J Comput Sci Eng, vol. 6, pp. 132-140, 2011.

[66] **J. Kennedy, R. Eberhart**, "Particle swarm optimization", In Proceedings of the IEEE international conference on neural networks, pp. 1942-1948,1995.

[67] **A. Kaveh, N. Farhoudi**, "A new optimization method: dolphin echolocation", Advances in Engineering Software, vol. 59, pp: 53-70, 2013.

[68] **B. Basturk, D. Karaboga**, "An artificial bee colony (ABC) algorithm for numeric function optimization", In Proceedings of the IEEE swarm intelligence symposium, pp. 12-14, 2006.

[69] **M. Dorigo, M. Birattari, T. Stutzle**, "Ant colony optimization", IEEE Comput Intell, vol. 1,pp. 28-39, 2006.

[70] **X. S. Yang, S. Deb**, "Cuckoo search via Lévy flights", In Proceedings of the world congress on nature & biologically inspired computing, NaBIC, pp.210-214, 2009.

[71] **R. V. Rao, V. J. Savsani, D. P. Vakharia**, "Teaching–learning-based optimization a novel method for constrained mechanical design optimization problems", Computer-Aided Des, vol. 43, pp. 303-15, 2011.

[72] **Z. W. Geem, J. H. Kim, G. Loganathan**, "A new heuristic optimization algorithm: harmony search", Simulation, vol. 76, pp. 60-68, 2001.

[73] **F. Glover**, " Tabu search – Part I ", ORSA Journal of Computing, vol.1, pp.190-206, 1989.

[74] **F. Glover**, " Tabu search–PartII ", ORSA Journal of Computing, vol. 2, pp. 4-32, 1990.

[75] **S. He, Q. Wu, J. Saunders**, "A novel group search optimizer inspired by animal behavioural ecology", In Proceedings of the IEEE congress on evolutionary computation, CEC, pp. 1272-1278, 2006.

[76] **S. He, Q. H. Wu, J. Saunders**, "Group search optimizer: an optimization algorithm inspired by animal searching behavior", IEEE Trans Evol Comput, vol. 13, pp. 973-990, 2009.

[77] **N. Mladenovié, P. Hansen**, "Variable neighborhood search", Computers and Operations Research, vol. 24, no. 11, pp. 1097-1100, 1997.

[78] **H. R. Lourenço, O. Martin, T. Stützle**, "Iterated Local Search", Handbook of Metaheuristics, Kluwer Academic Publishers, International Series in Operations Research & Management Science, vol. 57, pp. 321-353, 2003.

[79] **C. Voudouris, E. Tsang**, "Guided local search and its application to the traveling salesman problem", European Journal of Operational Research", vol. 113, no. 2, pp. 469-499, 1999.

[80] **M. M. Eusuff, K. E. Lansey**, "Shuffled frog leaping algorithm: a memetic meta-heuristic for combinatorial optimization", J. Heuristics, in press, 2000.

[81] **M. M. Eusuff, K. E. Lansey**, "Optimization of Water Distribution Network Design Using the Shuffled Frog Leaping Algorithm", Journal of Water Resources Planning and Management, vol. 129, no. 3, pp. 210-225, 2003.

[82] **D. Zou, H. Liu, L. Gao, S. Li**, "Directed searching optimization algorithm for constrained optimization problems", Expert Systems with Applications, vol. 38, no. 7, pp. 8716-8723, 2011.

[83] **G. David, JR. Forney**, "Maximum-Likelihood Seauence Estimation of Digital Sequences in the Presence of Intersymbol Interference", IEEE Transactions on information theory, Vol.18, No. 3,1972.

[84] **A. J. Vergonjanne**, "Introduction à l'égalisation en communications numériques", Support de cours, ENSIL, France, 2013.

[85] **G. LE. Pemp**, "Capacité de poursuite des algorithmes adaptatifs dans un canal de transmission sous-marin à trajets multiples", Mémoire de maitre ès sciences, Université de Laval, 2013.

[86] **Paulo S.R. Diniz**, "Adaptive Filtering Algorithms and Practical Implementation", 3rd Edition, Springer, Science and Business Media, 2008.

[87] **J. A. Apolinario**, "QRD-RLS Adaptive Filtering", Springer, Science and Business Media, LLC 2009.

[88] **A. Zerguine, A. Shafi, M. Bettayeb**, "Multilayer perceptron based DFE with lattice structure", IEEE Transaction on Neural network, vol.12, no.3, pp.532-545, May 2001.

[89] **I. Santamaria, D. Erdogmus, J.C. Principe**, "Entropy Minimization for Supervised Digital Communications Channel Equalization", IEEE Transactions on Signal Processsing, vol.5, no.5, pp.1184-1192, May.2002.

[90] **J. Choi, C. Lima, S. Haykin**, "Kalman Filter-Trained Recurrent Neural Equalizers for Time-Varying Channels", IEEE Transaction on communications, vol.53, no.3, 2005.

[91] **A. T. Al-Awami, A. Zerguine, L. Cheded, A. Zidouri, W. Saif**, "A new modified particle swarm optimization algorithm for adaptive equalization", Digital Signal Processing, vol. 21, no.2, pp. 195-207. 2011.

[92] **P. Sivakumar, L. Arthi,** "Enhancement of New Channel Equalizer Using Adaptive Neuro Fuzzy Inference System, IOSR Journal of Electronics and Communication Engineering (IOSR-JECE)", vol. 6, no.1, pp. 34-43. 2013.

[93] **K. Mahmood, A. Zidouri, A. Zerguine,** "Performance analysis of a RLS-based MLP-DFE in time-invariant and time-varying channels", Digital Signal Processing 18, 307–320, 2008.

[94] **C. M. Lee, S.S. Yang, C. L. Ho,** "Modified back-propagation algorithm applied to decision-feedback equalization", IEE Proc.-Vis. Image Signal Process, vol. 153, no. 6, 2006.

[95] **Mohamed NEMISSI,** "Classification et reconnaissances des formes par algorithmes hybrides", Thèse de Doctorat, Université 8 mai 45, Guelma, 2009.

[96] **P. Poincot**, "Classification et recherche d'information bibliographique par l'utilisation des cartes auto-organisatrices, applications en astronomie", Thèse de doctorat, Université Louis Pasteur, 1999.

[97] **C. Touzet**, "Les réseaux de neurones artificiels, introduction au connexionnisme : cours, exercices et travaux pratiques", EC2, Collection de l'EERIE, N. Giambiasi. hal-01338010,1992.

[98] **D. Hebb**, "The organization of the Behavior", Wiley, New York, 1949.

[99] **F. Rosenblatt**, "The perceptron: a probabilistic model fir information storage and organization in the brain", Psychological Review, vol. 654, 386-408, 1958.

[100] **M. Minsky, S. Papert**, "Perceptrons", MIT Press, Cambridge, MA, 1969.

[101] **D. E. Rumelhart, G. E. Hinton, R. J. Williams**, "Learning internal representation by error propagation, Parallel distributed processing: exploration in the microstructure of cognition", MIT press Cambridge, pp. 318-362, 1986.

[102] **T. Dufour,** "Egalisation adaptatifs pipelines par la technique de l'anticipation relaxée", Mémoire, Université du Québec à trois-rivières, 2002.

[103] **B. Mulgrew**, "Applying Radial Basis Functions", IEEE Signal Processing Magazine, vol. 13, 50-65, 1996.

[104] **D. S. Broomhead, D. Lowe**, "Multivariable Functional Interpolation and Adaptive Networks", Complex System vol. 2, pp. 321-355, 1988.

[105] **S. Tertois**, "Réduction des effets des non-linéarités dans une modulation multi porteuse à l'aide de réseau de neurones", Thèse de Doctorat, Université de Rennes 1, 2003.

[106] **J. Parks, I.W. Sandberg**, "Universal Approximation using Radial-Basis Function Network", Neural Computation, vol. 3. no. 2. pp. 246-257. 1991.

[107] **K. Hornik**, "Multilayer feed forward network are universal approximator", Neural Networks, vol. 2, pp. 359-366. 1989.

[108] **K. Hornik**, "Some New Results on Neural Network Approximation". Neural Networks. vol. 6. no. 8. pp. 1069-1072. 1993.

[109] **R. O. Duda, P. E. Hart, D. G. Stork**, "Pattern Classification", 2ed., John Wiley & Sons,New York, 2001.

[110] **C. Looney**, "Pattern Recognition Using Neural Networks", Oxford University Press, New York, 1997.

[111] **Y. H. Hu, J. N. Hwang**, "Handbook of neural network signal processing", by CRC PRESS LLC Florida, USA, 2002.

[112] **S. K. Sharma, P. Chandra**, "An Adaptive Sigmoidal Activation Function Cascading Neural Networks", Advances in Intelligent and Soft Computing, vol. 87, 2011.

[113] **A, Fiordaliso,** "Une application des réseaux de neurones artificiels MLP à la prévision du prix d'une option négociable", Économie prévision, vol.127, pp.47-62, 1997.

[114] **S. S. Yang, C. Lu Ho, C. Min Lee,** "HBP: Improvement in BP Algorithm for an Adaptive MLP Decision Feedback Equalizer", IEEE Transactions on circuits and systems, vol. 53, no. 3, 2006.

[115] **M. Zidane**, "Contribution à l'identification et égalisation aveugles des canaux de transmission pour des systèmes de 4ème génération MC-CDMA", Thèse de doctorat, Université Sultan Moulay Slimane, Béni Mellal, 1999.

[116] **S. Saidani, I. Tifouti, M. Ghadjati, A. Moussaoui,** "Performance of the Multilayer Perceptron based DFE for Linear and Nonlinear Channels", International Conference on Automatic control, Telecommunications and Signals (ICATS15), University BADJI Mokhtar, Annaba, Algeria, November 16-18, 2015.

[117] **A. P. Russo**, "Neural Networks for sonar signal processing", tutorial no. 8, IEEE Conf. on Neural Networks for Ocean engineering, Wachington, D.C, 1991.

[118] **D. Nguyen, B. Widrow,** "Improving the learning speed two-layer neural networks by choosing initial values of the adaptive weights", in Proc. 1990 IEEE Int. Joint Conf. Neural Networks, San Diego, vol. 3, 21-26, 1990.

[119] **J. F. Shepanski,** "Fast learning in artificial neural systems multilayer perceptron training using optimal estimation", IEEE International Conference on Neural Networks 1, IEEE Press, New York, pp. 465-472, 1988.

[120] **T. Masters,** "Practical Neural Network Recipes in C++", Academic Press, Boston, 1993.

[121] **Y.F. Yam, T.W.S. Chow,** "Determining initial weights of feedforward neural networks based on least squares method", Neural Processing letter, vol. 2, pp. 13-17, 1995.

[122] **Y.F. Yam, T.W.S. Chow,** "A new method in determining the initial weights of feedforward neural networks", Neurocomputing, vol.16, pp. 23-32, 1997.

[123] **J. Y. F. Yam, T. W. S. Chow,** "A weight initialization method for improving training speed in feedforward neural network", Neurocomputing, vol. 30, pp. 219-232. 2000.

[124] **R. A. Jacobs,** "Increased rates of convergence through learning rate adaptation", Neural Networks, vol. 1, pp. 295-307, 1988.

[125] **M. Zurada,** "Lambda learning rule for feedforward neural networks", in Proc, IEEE int. Conf. Neural Networks, vol. 3, pp. 1808-1811. 1993.

[126] **P. Chandra, Y. Singh,** "An activation function adapting training algorithm for sigmoidal feedforward networks", Neurocomputing, vol. 61, 2004, pp. 429– 437.

[127] **K. Eom, K. Jung H. Sirisena,** "Performance improvement of backpropagation algorithm by automatic activation function gain tuning using fuzzy logic", Neurocomputing, vol. 50, pp. 439 – 460, 2003.

[128] **Y. H. Zweiri, J. F. Whidborne L. D. Seneviratne,**"Three-term backpropagation algorithm", Neurocomputing, Vol. 50, pp. 305-318, 2003.

[129] **Y. H. Zweiri,** "Optimization of a Three-Term Backpropagation Algorithm Used for Neural Network Learning", International Journal of Computational Intelligence. vol. 3, pp. 322–327, 2006.

[130] **X.G. Wang, Z. Tang, H. Tamura M. Ishii,** "A modified error function for the backpropagation algorithm", Neurocomputing, vol. 57, pp. 477- 484, 2004

[131] **S. Saidani, A. Moussaoui, M. Ghadjati,** "A New Training Strategy for DFE-MLP Using Modified BP Algorithm and Application to Channel Equalization", WSEAS Transactions on Signal Processing, vol.13, pp. 115-120, 2017.

[132] **V. Bachelet,** "Métaheuristiques parallèles hybrides : Application au problème d'affectation quadratique", Thèse de Doctorat, Université des sciences et technologies de Lille, 1999.

[133] **E. D. Talbi, L. Geneste, B. Grabot, R. Prévitali, P. Hostachy**, "Application of optimization techniques to parameter set-up in scheduling", Computers in Industry, vol, 55, no. 2, pp. 105-124, 2004.

[134] **C. Blum, J. Puchinger, G. R. Raidl, A. Roli**, "Hybrid metaheuristics in combinatorial optimization : A survey, Applied Soft Computing", vol. 11, no. 6, pp. 4135-4151, 2011.

[135] **S. Panda, A. Sarangi, S. P. Panigrahi**, "A new training strategy for neural network using shuffled frog-leaping algorithm and application to channel equalization", International Journal of Electronics and Communications (AEÜ), vol. 68, no. 11, pp. 1031–1036, 2014.

[136] **E. Elbeltagi, T. Hegazy, D. Grierson**, "A modified shuffled frog-leaping optimization algorithm: applications to project management", Structure and Infrastructure Engineering, vol. 3, no. 1, pp. 53-60, 2007.

[137] **X. Zhang, Y. Zhang, Y, Shi, L. Zhao, C. Zou**, "Power control algorithm in cognitive radio system based on modified Shuffled Frog Leaping Algorithm", International Journal of Electronics and Communications (AEÜ), vol. 66, no. 6, pp. 448-454, 2012.

[138] **P. Roy, P. Roy, A. Chakrabarti**, "Modified shuffled frog leaping algorithm with genetic algorithm crossover for solving economic load dispatch problem with valve-point effect", Applied Soft Computing, vol. 13, no. 11, pp. 4244-4252, 2013.

[139] **J. Zhou, E. Dutkiewicz, R. P. Liu, X. Huang, G. Fang, Y. Liu**, "A Modified Shuffled Frog Leaping Algorithm for PAPR Reduction in OFDM Systems", IEEE Transaction on Broadcasting, vol. 61, no. 4, pp. 698-709, 2015.

[140] **M. H. Oboudi, R. Hooshmand, A. Karamad**, "Feasible method for making controlled intentional islanding of microgrids based on the modified shuffled frog leap algorithm", Electrical Power and Energy Systems, vol.78, pp. 745-754, 2016.

[141] **P. Kaur, S. Mehta**, "Resource provisioning and work flow scheduling in clouds using augmented Shuffled Frog Leaping Algorithm", Journal of Parallel and Distributed Computing, vol. 101, pp. 41-50, 2017.

[142] **R. Dash**, "Performance analysis of a higher order neural network with an improved shuffled frog leaping algorithm for currency exchange rate prediction", Applied Soft Computing, vol. 67, pp. 215-231, 2018.

[143] **C. Liu, P. Niu, G. Li, Y. Ma, W. Zhang, K. Chen**, "Enhanced shuffled frog-leaping algorithm for solving numerical function optimization problems", Journal of Intelligent Manufacturing, vol. 29, no. 5, pp. 1133-1153, 2018.

[144] **R. Li, Z. Jiang, A. Li, S. Yu, C. Ji**, "An improved shuffled frog leaping algorithm and its application in the optimization of cascade reservoir operation", Hydrological Sciences Journal, vol. 63, no. 15-16, pp. 2020-2034, 2019.

[145] **B. Tripathy, S. Dash, S. K. Padhy**, "Dynamic task scheduling using a directed neural network", Journal of Parallel and Distributed Computing, vol. 75, pp. 101-106, 2015.

[146] **S. Panda, P. K. Mohapatra, S. P. Panigrahi**, "A new training scheme for neural network and application in non-linear channel equalization", Applied Soft Computing, vol. 27, pp. 47–52, 2015.

[147] **S. Saidani, A. Moussaoui, M. Ghadjati**, "Channel Equalization Based On SFLA and DSO Trained Artificial Neural Network", Telecommunications and Radio Engineering, vol. 78, no. 17, pp. 1589-1600, 2019.

[148] **M. Eusuff, K. Lansey, F. Pasha**, "Shuffled frog-leaping algorithm: a memetic meta-heuristic for discrete optimization", Engineering Optimization, vol. 38, no. 2, pp. 129-154, 2006.

[149] **J. R. Zhang, J. Zhang, T. M. Lok, M. R. Lyu,** "A hybrid particle swarm optimization-back-propagation algorithm for feed-forward neural network training", Applied Mathematics and Computation, vol. 185, pp. 1026-1037, 2007.

[150] **O. Martin, S. Otto, E. Felten**,"Large-step markov chains for the tsp: Incorporating local search heuristics", Operation Research Letters, vol. 11, pp. 219-224, 1992.