University Sidi Mohammed Ben Abdellah
Faculty of sciences and techniques

Stability and parameters estimation in dynamical systems of type recurrent networks, contributions to new non-linear optimization models : applications to real problems

Joudar Nour-eddine

Faculty of Sciences and Techniques of FEZ

A dissertation submitted to the University of Sidi Mohamed Ben Abdellah in accordance with the requirements of the degree of DOCTOR OF PHILOSOPHY

2018

Contents

Abstract

Thanks to the solidity of its theoretical bases and the robustness of its algorithms, recurrent neural networks have become the alternative the most used for solving combinatorial problems. Among these networks, the Hopfield model (CHN), proposed in 1985, has attracted the interest of many researchers. Indeed, this model allows, thanks to an analogy of neural networks with static physics, to solve optimization problems by associating the variables of the problem with the variables of the energy function of the network. However, the major drawback of this model is the non-feasibility of the obtained solutions, as well as the unsuitable choice of the model parameters. In this thesis, our study focuses on the intrinsic functioning of the dynamical system of Hopfield for the resolution of new nonlinear optimization problems associated with image restorations and the reduction of probabilistic self-organizing maps. In this context, several modelling and improvements are made, thus constituting nonlinear optimization models with mixed variables. Next, the adaptation of the Continuous Hopfield systems with the suggested models is described and analyzed. In addition, a set of mathematical results regarding CHN convergence and the feasibility of the solutions are introduced. For each problem studied in this work, a number of numerical results are presented and compared against several approaches from the literature.

Key-words: non-linear Optimization, dynamical systems, artificial neural networks, Hopfield neural networks, image restoration, probabilistic self-organizing maps, clustering problems.

Acknowledgement

This account of gratitudes is something special to me. For me, it is not just a formal note but more than that. It would make possible for me to express my feelings for all those who had been at my side during this important project of my life and I never found a chance to do that till now.

I was lucky to have Mr. **Ettaouil Mohamed** as my advisor. He is a kind hearted and highly professional person. From the very start till the end, his contribution and guidance were of great value. Whenever, I needed a help, he was there for assistance. His precise and pertinent comments always helped me get out of difficult situations. Without his supervision, it would not have been possible for me to write this dissertation.

With equal fervour, I want to extend a heart-felt thank to my Committee members, Pr **El merouani Mohamed**, Pr. **Qjidaa Hassan** and Pr **Bahaj Mohamed** for their reviews and helpful comments throughout the process as well as Pr. **Sidki Omar** for presiding the doctoral committee. Further, a special thanks to the professors: **LAZAAR Mohamed**, **ENNAHNAHY Nour-eddine** and **EL KHAOULANI EL IDRISSI Rachid**, they have been a source of encouragement, and their remarks were of great help.

I was also fortunate to have many helping friends. It would not be possible for me to mention here the names of all those. But still I would like to mention the names of some of them who had been a great source of encouragement for me. It was my pleasure to have friends and colleagues like **Zakaria En-naimani**, **Benzakour Mohamed** and **Ramchoun Hassan**

At the end, I would like to mention about my parents, wife and son who mean a lot to me. My parents have sacrificed much for my success and I am indebted to them. I wish one day I get a chance to pay back a little for what my parents did for me. My brothers and sister had always been there for me during all kind of circumstances. I am thankful for their encouragement. I pay tribute to my wife **KHADIJA** for her patience and support. She had to take care of many things during my long working hours including our little angel **BARAE**.

Acronyms

PSF : **P**oint **S**pread **F**unction
ZBC : **Z**eros **B**oundary **C**onditions
PBC : **P**eriodic **B**oundary **C**onditions
RBC : **R**eflexive **B**oundary **C**onditions
BTTB : **B**lock **T**oeplitz with **T**oeplitz **B**locks
BTTB : **B**lock **C**irculant with **C**irculant **B**locks
IRP : **I**mage **R**estoration **P**roblem
BV : **B**ounded **V**ariations **P**roblem
PDE : **P**artial **D**ifferential **E**quations
ANN : **A**rtificial **N**eural **N**etwork
MLP : **M**ulti-**L**ayer **P**erceptron
RBF : **R**adial **B**asis **F**unction
DHN : **D**iscrete Hopfield Neural Network
CHN : **C**ontinuous Hopfield Neural Network
GA :**G**enetic **A**lgorithm
PROSM : **P**robabilistic **S**elf **O**rganizing **M**ap
MI-SIR :**M**ixed **I**nteger **S**elective **I**mage **R**estoration

List of Figures

List of Tables

Résumé

STABILITÉ ET ESTIMATION DES PARAMÈTRES DANS LES SYSTÈMES DYNAMIQUES DE TYPE RÉSEAUX RÉCURRENTS, CONTRIBUTIONS À LA RÉSOLUTION DE NOUVEAUX MODÈLES D'OPTIMISATION NON LINÉAIRES ET APPLICATION AUX PROBLÈMES RÉELS

0.1 Introduction

Grâce aux résultats obtenus au cours des dernières décennies, les réseaux de neurones artificiels (ANNs) ont connu un succès croissant et ont prouvé leur efficacité dans plusieurs domaines: analyse et compression d'images, classification, reconnaissance de formes, analyse des signaux...[**35**, **36**, **55**, **56**]. De tous les types des ANNs, les réseaux de neurones récurrents (ou dynamiques) RNNs ont été les plus investis dans l'optimisation combinatoire. En effet, l'utilisation des RNNs pour résoudre les problèmes d'optimisation revient à la définition d'une fonction dont sa minimisation correspond à l'état stable du réseau. Dans la littérature, les réseaux récurrents les plus développés sont: les réseaux cellulaires (CNNs)[**121**], les cartes auto-organisatrices (SOMs)[**66**] et le réseau de Hopfield (HNN) [**47**, **49**, **102**, **104**]. Ce dernier, dès sa création en 1982, a permis de résoudre plusieurs problèmes d'optimisation combinatoire, à savoir le problème du voyageur de commerce, le traitement de signaux, les problèmes de programmation linéaire [**114**]. Il s'agit d'un modèle neuronale qui évolue vers le minimum d'une fonction d'énergie de la même façon qu'un système physique évolue vers l'état d'équilibre[**47**].
Le but de cette thèse est, d'une part, d'aborder les problèmes d'optimisation combinatoire par le réseau de Hopfield continu (CHN), et d'autre part, d'assurer la convergence du réseau et estimer les paramètres de sa fonction d'énergie. Dans ce manuscrit, nous contribuons à deux problèmes majeurs, à savoir le problème mathématique de restauration d'images (IRP) et le choix de l'architecture de la carte auto-organisatrice probabiliste (PRSOM)appliquées au regroupement des données. Dans ce contexte, plusieurs modélisations et améliorations sont apportées aux modèles d'optimisation des deux problèmes. De plus, la résolution via CHN est investie et enrichie par une analyse profonde de la stabilité du CHN. Ce document est structuré de la manière suivante:
Le **chapitre 1** est consacré à la présentation des réseaux de neurones artificiels récurrents (RNNs). Cette présentation montre toute la diversité et la richesse des travaux qui s'inscrivent dans ce domaine. Tout d'abord, nous décrivons le concept d'un réseau de neurones artificiels, ainsi que les composantes principales qui construisent son architecture. Après, nous expliquons les fondements théoriques de l'apprentissage statistique, en définissant la notion de vraisemblance et la relation de sa maximisation avec la minimisation quadratique. Ensuite, nous présentons les différents types des réseaux de neurones artificiels et leurs modes d'apprentissage. Nous terminons ce chapitre par une description du réseau de Hopfield continu (CHN), dans laquelle nous définissons l'architecture associée au modèle CHN, ses paramètres, sa fonction d'énergie et son mode d'évolution.

Le **chapitre 2** s'attache à présenter le problème mathématique de restauration d'images. Dans la première section, nous expliquons les phases de numérisation permettant de construire une image numérique, puis nous interprétons le bruit et le flou en tant que composantes nécessaires pour générer le modèle mathématique associé à IRP. Dans une seconde, nous expliquons la formulation du problème de restauration d'image, dont nous commençons par décrire le modèle de dégradation à partir duquel nous expliquons le problème inverse de la restauration. Comme ce modèle est mal conditionné, nous décrivons par la suite quelques techniques proposées pour surmonter ce problème, en particulier, les moindres carrées, les méthodes itératives et la technique de régularisa-

tion que nous allons adopter dans nos travaux. Ensuite, en se basant sur la régularisation de Tikhonov[**106**], nous décrivons en détail le modèle d'optimisation de restauration d'images, ainsi que les résultats d'existence et d'unicité de la solution qui ont été développés sur les espaces des fonctions à variations bornées $BV(\Omega)$[**10**]. Enfin, nous rappelons l'utilisation des EDPs pour la restauration des images, et en particulier l'équation de la chaleur et le modèle non linéaire de Perona [**64**, **85**].

Le **chapitre 3** concerne la résolution du modèle de restauration d'images via le réseau continu de Hopfield. Dans ce contexte, nous expliquons d'abord le fonctionnement du réseau discret de Hopfield pour la résolution d'IRP, ainsi que le problème de fluctuation qui limite la performance de ce modèle. Ensuite, nous décrivons en détail les étapes d'adaptation du CHN au modèle IRP. En outre, nous enrichissons la phase de résolution par l'analyse de stabilité et l'estimation des paramètres du modèle proposé. Les résultats obtenus par le réseau CHN sont satisfaisants numériquement et visuellement.

Dans le **chapitre 4**, nous proposons une nouvelle vision de restauration d'images appelée restauration d'images sélective (SIR). Dans le modèle classique de IRP, nous introduisons une règle de décision permettant d'indiquer la nature des pixels. Cette règle permet de choisir, parmi l'image dégradée et une image de référence (Médian par exemple), les pixels qui sont reconstruits, c.à.d ceux qui génèrent un coût minimal de la fonction d'énergie associée au modèle SIR. Le modèle ainsi obtenu est un problème d'optimisation non linéaire à variables mixtes, pour lequel la résolution via les méthodes exactes est très difficile, voire impossible. Dans ce contexte, nous utilisons le réseau continu de Hopfield continu et l'algorithme génétique progressivement pour résoudre le modèle proposé, en protégeant à chaque itération les pixels originaux dans l'image dégradée.

Le **chapitre 5** traite le problème de choix de l'architecture de la carte auto-organisatrice probabiliste. Dans ce contexte, le problème envisagé est modélisé en termes d'un problème d'optimisation non linéaire à variables mixtes sous contraintes, et pour des raisons d'augmentation de difficulté de résolution, notre étude s'est ensuite exclusivement consacrée à la résolution du modèle proposé via l'expectation-minimisation. Dans cette phase, le réseau de Hopfield continu est investi pour la phase d'affectation-optimisation, bien que la phase de minimisation soit résolue à partir du gradient de la vraisemblance. Enfin, nous appliquons la carte optimisée au problèmes de clustering (regroupement des données) .
Le manuscrit se termine par une conclusion générale. Des perspectives alors sont exposées pour des travaux futurs.
Ce résumé se présente comme suit: section 0.2 décrit les modèles de Hopfield et la carte auto-organisatrice. Dans la section 0.3, le modèle continu de Hopfield est investi dans le cadre de la résolution du problème de restauration d'images. La construction du modèle de restauration sélective, ainsi que la résolution via CHN-GA sont discutées dans la section 0.4. Dans la section 0.5, un nouveau modèle d'optimisation associé au choix de l'architecture du *PRSOM* est proposé et résolu via CHN.

0.2 Réseaux de neurones récurrents

Contrairement à un réseau de neurones acyclique (Feed-Forward Neural Network - FFNN- paragraphe 1.4.1), un réseau de neurones récurrent (Recurrent Neural Network - RNN) est un réseau de neurones dont le graphe des connexions contient au moins un cycle[**19**, **45**, **76**, **87**, **121**]. Ce type des RNAs est largement exploité dans plusieurs domaines comme l'analyse des données, la reconnaissance de formes, le traitement de la parole, les mémoires associatives et l'optimisation combinatoire[**16**, **22**, **47-50**, **65**, **66**]. Dans ce résumé, nous nous intéressons aux réseaux de Hopfield et la carte auto-organisatrice [**65**, **66**],[**47-50**].

0.2.1 Réseaux de Hopfield continus CHN

Le réseau de Hopfield a été inventé par le physicien J. Hopfield en 1982 [**47**]. Sa découverte a permis de raviver l'intérêt des recherches dans les réseaux de neurones, qui avaient connu une stagnation durant les années 1970 suite à un travail de M. Minsky et S. Papert qui a soulevé les problèmes liés à ces réseaux [**76**]. Par ailleurs, le réseau de Hopfield s'inscrit dans la catégorie des réseaux récurrents. Il est caractérisé par sa capacité de laisser l'information circuler récursivement d'une manière partielle ou bien totale. Il s'agit d'un système à minimisation d'énergie. Ce modèle a été utilisé pour résoudre une grande variété de problèmes d'optimisation combinatoire [**97**, **98**, **101**].

La présente section concerne donc la conception générale du réseau de Hopfield continu (CHN) et ses différents attributs essentiels: son architecture, l'équation d'évolution de ses états, ses points d'équilibre et sa fonction d'énergie. Dans ce contexte, nous présentons les principes d'association des variables du problème de programmation quadratique avec les paramètres de la fonction d'énergie à minimiser.

0.2.1.1 Architecture du réseau de Hopfield continu

Le réseau de Hopfield possède une topologie cyclique. Dans ce modèle, le réseau est constitué d'une seule couche de n neurones entièrement inter-connectés. Ils évoluent à partir d'un état initial.

La figure 1 représente un réseau de Hopfield composé de n neurones. La sortie de chaque neurone est bouclée sur les entrées des autres neurones additionnées à un vecteur du bruit provenant de l'extérieur I_i^b. Un neurone i est relié à un neurone j par des poids symétriques $T_{ij} = T_{ji}$, mais il n'est pas connecté à lui-même $T_{ii} = 0$. De plus, le neurone i est décrit par deux autres variables:

- •Le potentiel intérieur u_i calculé comme une somme des informations pondérées par les poids des connexions des autres neurones.
- • La sortie v_i définie par $v_i = h(u_i)$. Avec h désigne la fonction d'activation du neurone i.

Souvent, on utilise la fonction d'activation de type tangente hyperbolique qui permet d'assurer la convergence des sorties v_i vers les sommets de l'hypercube $[0,1]^n$.

$$v_i = h(u_i) = \frac{1}{2}(1 + \tanh(\frac{u_i}{u_0})) \quad \forall i \in \{1, ..., n\}, \quad u_0 > 0 \tag{1}$$

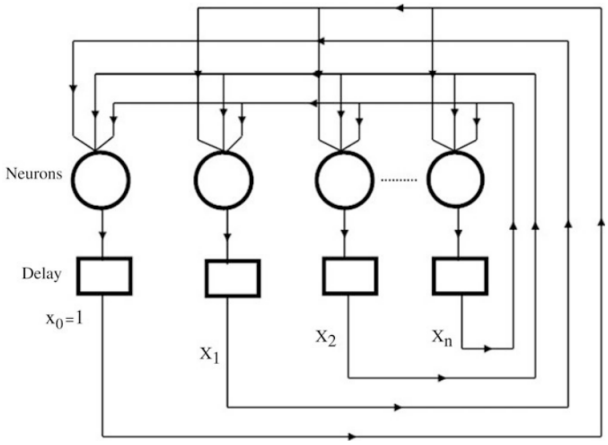


Figure 1: Réseau de Hopfield continu à n neurones

En se référant au cas d'une réalisation physique d'un réseau électronique, basée sur des amplificateurs opérationnels et résistances, l'équation différentielle qui gouverne l'évolution du réseau de Hopfield continu est donnée par[**102**, **104**]:

$$\frac{du}{dt} = -\frac{u}{\tau} + Tv + I^b \tag{2}$$

Ce système est appelé l'équation d'état du réseau de Hopfield.
Les vecteurs u, v et I^b représentent, respectivement, le vecteur des états, le vecteur de sorties et le vecteur du bruit provenant de l'extérieur. T représente la matrice des poids des connexions entre les neurones.

0.2.1.2 Énergie et stabilité des réseaux de Hopfield continus

Nous présentons dans la suite, la fonction d'énergie associée à l'équation d'état du réseau de Hopfield. Puis après, nous étudions également l'existence d'une solution (point d'équilibre) de cette équation, ainsi que les méthodes d'approximations d'un point d'équilibre lorsque celui-ci existe [**103**].
L'évolution du réseau de Hopfield est caractérisée par un système d'équations différentielles, alors l'étude de l'existence d'une solution (point de stabilité) de ce système se base sur l'existence d'une fonction d'énergie dite fonction de Lyapunov [**47**].

Définition 1

Soit $v(0)$ le vecteur qui représente les états initiaux des neurones du réseau de Hopfield. Un vecteur v^e est appelé un point de stabilité du réseau de Hopfield s'il existe un instant t^e tel que :

$$\forall t \geq t^e, \quad v(t) = v^e \tag{3}$$

Dans [**48**], Hopfield a démontré l'existence d'un point d'équilibre pour le système (2). En effet, par un argument de la fonction de Lyapunov, Hopfield a démontré que les systèmes différentiels de la dynamique d'état de son réseau sont globalement asymptotiquement stables, i.e. de n'importe quel état initial, le système converge vers un état stable.

La fonction d'énergie de Hopfield définie sur $H = [0,1]^n$ est donnée par :

$$E(v) = -\frac{1}{2}v^t T v - \left(I^b\right)^t v + \frac{1}{\tau}\sum_{i=1}^{n}\int_0^{v_i} g^{-1}(x)\,dx \tag{4}$$

Il faut noter que cette fonction E est bornée et sa dérivée est négative. Par conséquent, l'évolution de l'état global du système est un déplacement dans l'espace des états à la recherche d'un minimum, qui peut être local de E.

Théorème 0.1: Lyaponuv

Si pour un système différentiel, une fonction d'énergie E décroissante par rapport au temps existe, tel que : $\frac{dE(u(t))}{dt} = 0$ pour $\frac{du(t)}{dt} = 0_{\mathbb{R}^n}$ alors ce système est stable ; c'est-à-dire $u(t)$ tend vers un point stable.

Le théorème suivant donne la forme de la fonction d'énergie associée au système (2).

Théorème 0.2

Si T est une matrice zéro-diagonale et symétrique, alors la fonction définie par (4) est une fonction d'énergie associée au système (2) qui vérifie les conditions du théorème de Lyapunov.

Hopfield a défini une fonction d'énergie E pour le réseau proposé, qui mesure son état à un instant donné et qui est très proche de l'énergie d'un système thermodynamique. De cette façon, l'énergie minimale du réseau de Hopfield correspond à un état d'équilibre de ce réseau. Plus précisément, Hopfield a montré que l'énergie E ne peut que baisser ou rester stable en un minimum local. Aussi, il a prouvé que la symétrie de la matrice des poids est une condition suffisante pour l'existence de la fonction de Lyapunov **[47, 48]**. Un point $v \in H$, peut être un point d'équilibre du réseau de Hopfield si et seulement si, les trois conditions suivantes sont satisfaites **[103]**:

$$E_i(v) = \frac{\partial E(v)}{\partial v_i} \geq 0 \quad \forall i \in \{1,...,n\} \quad tel\ que \quad v_i = 0 \tag{5}$$

$$E_i(v) = \frac{\partial E(v)}{\partial v_i} \leq 0 \quad \forall i \in \{1,...,n\} \quad tel\ que \quad v_i = 1 \tag{6}$$

$$E_i(v) = \frac{\partial E(v)}{\partial v_i} = 0\ \forall i \in \{1,...,n\} \quad tel\ que \quad v_i \in]0,1[\tag{7}$$

0.2.1.3 Programmation quadratique et réseaux de Hopfield continus

Les problèmes d'optimisation combinatoire ont été rencontrés dans différents domaines issues des sciences de l'ingénieur. Ces problèmes sont traités par une grande classe des méthodes d'optimisation issus de la programmation mathématique. Dans ces dernières années, les réseaux de Hopfield continus ont été largement utilisés pour résoudre ces problèmes d'optimisation grâce à leur performance en terme de temps de résolution **[47,**

70, **102**]. Dans ce contexte, une fonction d'énergie généralisée a été proposée pour traiter ce type de problèmes. Pour assurer la faisabilité de cette fonction d'énergie, d'où la stabilité du réseau, une procédure d'estimation des paramètres de la fonction ainsi définie dite la procédure d'hyperplan a été proposée [**104**].
Dans cette sous-section, nous présentons la méthode de résolution d'un programme quadratique à variables bivalentes en se basant sur le réseau de Hopfield continu (CHN). Ce type de problèmes, soumis à des contraintes d'égalité et/ou d'inégalité linéaires à variables mixtes, s'écrit sous la forme suivante :

$$(P) = \begin{cases} \min f(v) = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} T_{i,j}v_i v_j + \sum_{i=1}^{n} c_i v_i \\ s.c. \\ \sum_{i=1}^{n} a_{k,i}v_i = b_k \qquad \forall k \in \{m_1+1, ..., m\} \\ \sum_{i=1}^{n} T_{k,i}v_i + T_{k,n+k}v_{n+k} = b_k \qquad \forall k \in \{1, ..., m_1\} \\ v_i \in \{0,1\} \quad \forall i \in \{1, ..., n\} \\ v_{n+k} \in [0,1] \quad \forall k \in \{1, ..., m_1\} \end{cases} \tag{8}$$

Les contraintes d'inégalité du problème étudié sont transformées en contraintes d'égalité en introduisant des variables d'écart $v_{n+k} \in [0,1]$ avec $k \in \{1, ..., m_1\}$.
Étant donnée la matrice des coûts T et la matrice des contraintes A du problème de programmation quadratique (P), et sans perte de généralité, on peut considérer le programme quadratique à variables mixtes soumis aux contraintes linéaires suivant :

$$P = \begin{cases} \min f(v) = \frac{1}{2}v^t T v + c^t v \\ s.c. \\ Av = b \\ v_i \in \{0,1\} \quad \forall i \in \{1, ..., n\} \\ v_{n+k} \in [0,1] \, \forall k \in \{1, ..., m_1\} \end{cases} \tag{9}$$

Les réseaux de Hopfield continus sont utilisés en programmation quadratique dans laquelle la recherche de l'extremum se ramène à la minimisation d'une fonction d'énergie qui prend en considération le coût du problème et les contraintes auxquelles il doit obéir:

$$E(v) = E^0(v) + E^R(v) \qquad \forall v \in [0,1]^n \tag{10}$$

Où:

- La fonction E^0 est proportionnelle à la fonction objective.

- La fonction E^R est une fonction quadratique qui pénalise non seulement les contraintes violées du problème, mais garantit aussi la faisabilité de la solution obtenue par le réseau de Hopfield continu.

0.2.2 Cartes auto-organisatrices SOM

La carte auto-organisatrice (SOM), également nommée réseau de Kohonen a été introduite par Teuvo Kohonen en 1982 **[60, 66, 67]**. Ce réseau est vu comme une généralisation du modèle K-moyennes (K-means)**[32, 33]**. Dans ce cadre, Kohonen a remarqué que les zones du cerveau qui gèrent le fonctionnement du corps humain respectent la topologie physique (auto organisation). Ce fonctionnement introduit la notion de voisinage qui caractérise la topologie de la carte auto-organisatrice et qui fait la différence avec le modèle des K-moyennes **[6]**.
Plus précisément, ce principe consiste à projeter un ensemble d'observations de dimen-

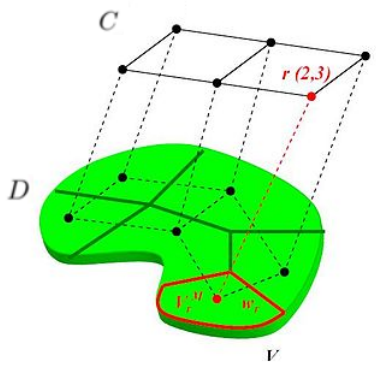


Figure 2: Principe de fonctionnement de la Carte Auto-Organisatrice

sion p d'un espace topologique B dans un espace de dimension réduite (2 ou 3), en faisant apparaître les corrélations qui existent entre ces observations (Figure 2), ce qui constitue aussi le principe de la quantification vectorielle de l'espace d'entrée **[60, 66]**
La carte auto-organisatrice est un outil largement utilisé dans l'analyse des données, la visualisation des données à grande dimension, ainsi que dans l'analyse des relations entre les variables.

0.2.2.1 Architecture et modèle de la Carte Auto-Organisatrice

L'architecture de la carte auto-organisatrice est vraisemblablement l'architecture de réseaux de neurones artificiels la plus proche du réseau de neurones biologiques. Cette carte possède deux couches qui sont décomposées comme suit (Figure 3):

- Une couche d'entrée qui distribue canoniquement les observations de l'espace d'entrées. Généralement, cette couche est constituée de p neurones présentant la dimension de l'espace d'entrées.

- Une couche de sortie constituée de K neurones organisés habituellement dans un plan bidimensionnel, où chaque neurone est relié à tous les autres neurones (neurones totalement connectés). De plus, chaque neurone correspond à un vecteur poids (pondération) dit le prototype du neurone qui est connecté aux observations de l'espace d'entrées.

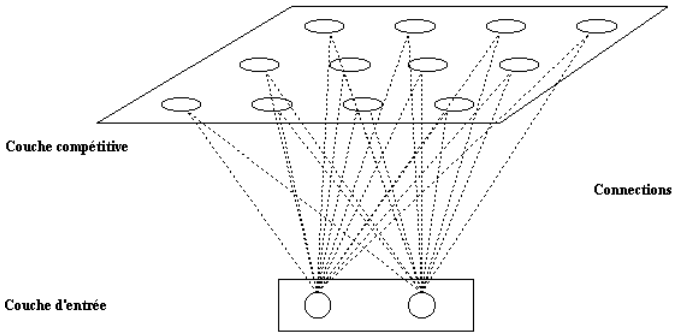


Figure 3: Architecture de la carte auto-organisatrice

La liaison entre les neurones (topologie de la carte) décrit la notion de voisinage entre neurones. Ce voisinage peut être défini de différentes manières. La première façon, consiste à définir le voisinage d'un neurone j d'ordre α dans le modèle classique de Kohonen par :

$$V_j^{\alpha} = \{i \in C, d(i,j) \leq \alpha\}$$

avec $d(i,j)$ représente la distance discrète définie comme étant la longueur du plus court chemin entre i et j sur la carte C.
Par ailleurs, on peut introduire la notion de voisinage dans le modèle récent de la carte auto-organisatrice, à l'aide d'une fonction noyau β qui prend la forme d'une fonction numérique, positive, symétrique telle que $\lim_{|x|\to\infty} \beta(x) = 0$. Cette fonction permet de quantifier automatiquement l'influence relative entre deux neurones i et j sur la carte par la valeur numérique $\beta(d(i,j))$. Plus précisément, on utilise la famille des fonctions paramétrées β^{α} associées à la fonction β définie par:

$$\beta^{\alpha}(d) = \beta\left(\frac{d}{\sigma}\right)$$

où σ est un paramètre de température qui influence sur la taille du voisinage i.e. on commence généralement avec un voisinage couvrant la plupart de la carte, après on diminue ce voisinage avec la diminution de σ selon le temps d'apprentissage.
Le paramètre σ s'exprime comme suit:

$$\sigma(t) = \sigma_{max}\left(\frac{\sigma_{min}}{\sigma_{max}}\right)^{\frac{t}{t_{max}-1}}$$

avec σ_{max} la température initiale, σ_{min} la température finale et t_{max} le nombre d'itérations maximal. , la fonction de voisinage de type Gaussien est adopté dans la plus part des

travaux de recherche[**67**].
Après avoir introduit les différents types de voisinages, nous présentons dans la suite de cette section la correspondance entre l'espace des observations et les pondérations associées aux neurones (vecteurs référents). Ces vecteurs référents fournissent alors une représentation discrète de l'espace des observations. Ils sont positionnés de telle façon qu'ils conservent la forme topologique de cet espace. On peut exprimer alors cette correspondance comme suit :

Définition 2

Soient $I = \{1, ..., K\}$ un ensemble d'indices et B un sous espace convexe borné de $\mathbb{R}^p$.
On définit un état de la carte auto-organisatrice à un instant t par :

$$W(t) = \left[w^1(t), ..., w^K(t)\right] \in \mathbb{R}^{p \times K}$$

Où $w^i(t) \in B$ est le vecteur référent du neurone i dans la carte.
Le modèle de la carte auto-organisatrice est défini par :

$$\begin{array}{rcccc} \phi_W & : & B & \rightarrow & I \\ & & x & \rightarrow & g_{x,W} = \underset{i \in I}{argmin} \left\| x - w^i \right\| \end{array} \tag{11}$$

Avec $g_{x,W}$ l'indice du neurone gagnant de la carte, dont le poids $w^{g_{x,W}}$ est le plus proche de l'entrée x.

Dans ce sens, l'objectif est de trouver un état optimal W^* qui stabilise le modèle ϕ_W (paragraphe 1.4.2.1).

0.3 Stabilité et estimation des paramètres du réseau de Hopfield continu: application à la résolution du modèle d'optimisation de restauration d'images

La restauration d'images (IRP) est un problème qui apparaît dans de nombreux domaines. En effet, quelle que soit l'origine que puissent avoir les images numériques, que ce soient des images d'origine photographique, scanner, satellite..., il est important d'en améliorer la qualité. Les résultats obtenus par un traitement spécifique effectué sur une image numérique, comme la détection de contours ou encore la reconnaissance de textures, sont conditionnés par la qualité de l'image initiale.
Dans cette section, nous commençons par la description du problème mathématique de restauration d'images, puis nous expliquons le fonctionnement des réseaux de Hopfield discret et continu dans le cadre de la résolution du modèle IRP. Nous terminons cette section par l'analyse de la convergence du CHN, ainsi que l'estimation des paramètres adéquats pour la résolution[**56**].

0.3.1 Modèle d'optimisation de restauration d'images

La restauration d'images consiste à retrouver une image représentant au mieux la scène observée x, à partir d'une donnée mesurée $y \in \mathbb{R}^N \times \mathbb{R}^{N'}$. L'entier N est à priori différent de l'entier N'. Cependant, lorsque cela n'est pas ambigu, nous considérons pour simplifier que $N = N'$.
De nombreux problèmes inverses consistant à estimer le vecteur inconnu x à partir d'un vecteur d'observation y peuvent être modélisés sous la forme linéaire suivante:

$$y = h * x + \eta \tag{12}$$

Où h est un opérateur décrivant le flou qui affecte les images et η représente à la fois le bruit de mesure et les erreurs de modélisation. L'opérateur $*$ désigne le produit de convolution. Si on dispose d'une information a priori sur le flou, l'équation précédente peut s'écrire sous la forme matricielle suivante:

$$Y = H.X + \eta \tag{13}$$

Où H est la matrice du flou, X et Y sont deux vecteurs de $\{0, 255\}^{N^2}$. Lorsque la matrice d'observation H est carrée ($N = M$) et inversible ($kerH = 0$), une inversion directe, au sens de la solution calculée par $x^* = H^{-1}y$, peut être envisagée. Dans tous les autres cas, cette inversion directe est inapplicable. La nécessité de définir des solutions bien posées utilisables dans des situations générales est à l'origine des développements de théories alternatives à l'inversion directe, comme l'inversion généralisée, et surtout celle de la régularisation.

0.3.1.1 Régularisation au sens de TIKHONOV

Plusieurs méthodes de régularisation ont apparu dans la littérature [**44**, **106**]. Dans notre travail, nous optons pour la régularisation de Tikhonov comme l'une des techniques

les plus répandues dans la théorie de restauration d'images. Les travaux précurseurs de Tikhonov initialement dans un cadre continu [**106**], consistent dans un cadre discret à définir la solution du problème inverse comme étant le minimisant d'un critère des moindres carrés pénalisé par un terme quadratique. D'où le problème inverse est régularisé par un nouveau modèle défini par:

$$J(X) = \frac{1}{2}\|Y - HX\|^2 + \frac{1}{2}\lambda\|DX\|^2 \tag{14}$$

Où D est une matrice de différenciation d'ordre d. Ce modèle permet de fournir un compromis entre la proximité avec la solution des moindres carrés, et la solution d'une différentielle minimale. Ce compromis est contrôlé par l'intermédiaire du paramètre λ, dont le réglage adéquat est en soi un problème délicat. Le choix le plus répandu de D consiste à utiliser la dérivée première $(d = 1)$, ce qui a pour effet de favoriser l'apparition de zones uniformes au sein de l'image. $\|.\|$ est une norme euclidienne.

0.3.2 Résolution du modèle proposé via les réseaux discrets de Hopfield

Dans cette partie, on s'intéresse à la résolution du problème d'optimisation suivant:

$$\underset{X\in\{0,..,255\}^{N^2}}{Min}\ J(X) = \frac{1}{2}\|Y - HX\|^2 + \frac{1}{2}\lambda\|DX\|^2 \tag{15}$$

Le fonctionnement du réseau de Hopfield discret pour la résolution du problème (15) est décrit par les étapes suivantes:

0.3.2.1 Codage par somme simple

Dans cette phase, chaque pixel est représenté par 256 neurones, où chaque neurone peut prendre les valeurs 0 ou 1. Par conséquent, le niveau de gris d'un pixel est égal à la somme des états des neurones qui le représentent, c.a.d chaque pixel X_i peut s'exprimer par:

$$X_i = \sum_{k=1}^{G} v_{ik} \tag{16}$$

Où G désigne le niveau de gris maximal.
En substituant X par la représentation somme simple, le modèle (15) est reformulé par:

$$\underset{v\in\{0,1\}^{N^2\times G}}{Min} J(v) = \sum_{p=1}^{N^2}(Y_p - \sum_{i=1}^{N^2}(H_{p,i}\sum_{k=1}^{G} v_{ik}))^2 + \frac{1}{2}\lambda\sum_{p=1}^{N^2}(\sum_{i=1}^{N^2}(D_{pi}\sum_{k=1}^{G} v_{ik}))^2 \tag{17}$$

0.3.2.2 Paramètres du réseau de Hopfield discret

Dans cette phase, nous estimons les paramètres du réseau DHN par identification de l'équation (17) avec la fonction de Lyaponuv (4).

Premièrement, nous définissons un réseau de Hopfield approprié sur lequel nous allons implémenter le modèle (17). Ce réseau est caractérisé par:

- États du réseau:
  $v = \{ v_{i,k} \ / \ i = 1,..,N^2 \quad et \quad k = 1,..,G\}$.
- Matrice des poids:
  $T = \{T_{ik,jl} \ / \ i,j = 1,..,N^2 \quad et \quad k,l = 1,..,G\}$

Ensuite, nous extrayons les paramètres du réseau comme suit:

$$DHN = \begin{cases} T_{ik,jl}^{DHN} = -\sum_{p=1}^{N^2} H_{p,i}H_{p,j} - \lambda \sum_{p=1}^{N^2} D_{p,i}D_{p,j} \\ I_{ik}^{DHN} = \sum_{p=1}^{N^2} y_p H_{p,i} \end{cases} \tag{18}$$

Dans le système (18), deux remarques intéressantes sont à mentionner:

- Les poids des interconnexions sont indépendants de la variable k et l.
- Les poids $T_{ik,ik}$ sont différents de 0, ce qui exige des auto-connexions pour chaque neurone et ce qui contredit par la suite les critères de convergence de Hopfield mentionnés dans la proposition 1.1.

Pour résoudre le problème mentionné dans la remarque 2, Zhou et al **[82, 124]** ont proposé une règle de décision qui dépend de la variation d'énergie ΔJ. Cette règle est résumée dans l'équation suivante:

$$v_{ik} = \begin{cases} v_{ik}^{new} \text{ si } \Delta J \text{ associée au changement } \Delta v_{ik} \text{ est inferieur à } 0 \\ v_{ik}^{old} \text{ sinon} \end{cases} \tag{19}$$

Où $\Delta J = J_{new} - J_{old}$ et $\Delta v_{ik} = v_{ik}^{new} - v_{ik}^{old}$.
Le fonctionnement du réseau de Hopfield discret est donc résumé dans l'algorithme suivant:

Data: Image dégradée: Y,
matrices: H et D
Result: Image désirée: X
1 initialiser arbitrairement les états des neurones;
2 **while** *Diminution de la fonction d'énergie* **do**
3 | mettre à jour les états de neurones aléatoirement et successivement selon la régle de décision (19);
4 **end**
5 Construire l'image désirée par le schéma somme simple:

Algorithm 1: Restauration d'images via le réseau discret de Hopfield

Malgré la performance satisfaisante de cet algorithme, la résolution via DHN est toujours incapable de dépasser le problème de fluctuation qui influence sur sa dynamique. En d'autres termes, les états du réseau errent dans l'espace des états proche du minimum de la fonction d'énergie, ce qui génère un comportement oscillatoire. Pour pallier cet inconvénient, nous proposons d'utiliser le modèle continu du réseau de Hopfield CHN.

0.3.3 Résolution du modèle proposé via les réseaux de Hopfield continus

L'objectif principal de ce paragraphe est d'investir la dynamique du réseau de Hopfield continu pour améliorer et supporter le modèle discret (17). Pour ce faire, nous définissons une nouvelle fonction d'énergie qui prend en considération les particularités du problème de restauration d'images. Pour être précis, le choix des paramètres de cette fonction doit assurer la faisabilité des points d'équilibre du CHN.

0.3.3.1 Fonction d'énergie et paramètres du réseau CHN

En se basant sur la fonction d'énergie (15), nous introduisons une nouvelle contrainte décrivant la faisabilité des états de neurones. Le modèle ainsi construit est défini par:

$$(P) = \begin{cases} \text{s.c} & min\ J(v) \\ & \sum_{i=1}^{N^2}\sum_{k=1}^{G}(v_{ik}(1-v_{ik})) = 0 \\ & v \in [0,1]^{N^2\times G} \end{cases} \tag{20}$$

La contrainte $\sum_{i=1}^{N^2}\sum_{k=1}^{G}(v_{ik}(1-v_{ik})) = 0$ est proposée pour forcer les états de neurones de s'installer au coins de l'hypercube $[0,1]^{N^2\times G}$. Ensuite, la notion de pénalité est utilisée pour construire une nouvelle fonction d'énergie associée au modèle (20) **[114]**. Par conséquent, le problème (P) est transformé en:

$$\begin{aligned} \underset{v\in\{0,1\}^{N^2\times G}}{Min} E^H(v) = \frac{1}{2}\alpha \sum_{p=1}^{N^2}\sum_{i=1}^{N^2}\sum_{j=1}^{N^2}\sum_{k=1}^{G}\sum_{l=1}^{G} H_{p,i}H_{p,j}v_{ik}v_{jl} \\ + \frac{1}{2}\lambda \sum_{p=1}^{N^2}\sum_{i=1}^{N^2}\sum_{j=1}^{N^2}\sum_{k=1}^{G}\sum_{l=1}^{G} D_{p,i}D_{p,j}v_{ik}v_{jl} - \alpha \sum_{p=1}^{N^2}\sum_{i=1}^{N^2}\sum_{k=1}^{G} y_p H_{p,i}v_{ik} + \\ \frac{1}{2}\alpha \sum_{p=1}^{N^2} y_p^2 + \frac{1}{2}\beta \sum_{i=1}^{N^2}\sum_{k=1}^{G}(v_{ik}(1-v_{ik})) \end{aligned} \tag{21}$$

Où les paramètres α, β, et λ sont des valeurs réelles à estimer de l'analyse de convergence du réseau CHN.
En comparant l'équation (21) avec celle de Lyapunov (4), les paramètres du réseau CHN sont donc extraits comme suit:

$$\begin{cases} T_{ik,jl}^{CHN} = -\alpha \sum_{p=1}^{N^2} H_{p,i}H_{p,j} - \lambda \sum_{p=1}^{N^2} D_{p,i}D_{p,j} + \beta \\ I_{ik}^{CHN} = \alpha \sum_{p=1}^{N^2} y_p H_{p,i} - 2\beta \end{cases} \tag{22}$$

0.3.3.2 Convergence et estimation des paramètres

La convergence du CHN est une tâche importante dans la résolution des problèmes combinatoires. En effet, dans de nombreux cas, le réseau converge vers une solution irréalisable. Généralement, la stabilité d'un réseau CHN est analysée sur: ensemble des solutions réalisables H_F, ensemble des coins de $[0,1]^{N^2\times G}$ qui ne sont pas des solutions réalisables $H_C - H_F$, et l'ensemble des points intérieures $H - H_C$ (Figure 4).
Dans notre modèle (21), aucune contrainte n'a été imposée sur les variables v_i, alors les

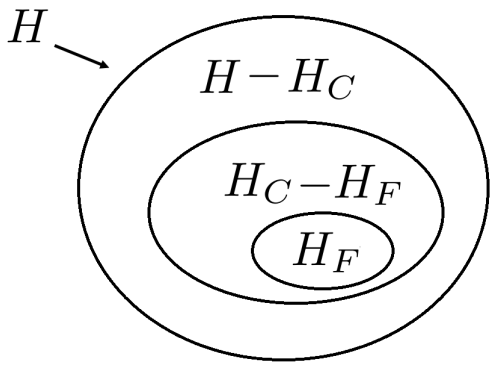


Figure 4: Ensemble des solutions réalisables et irréalisables de l'hypercube H

deux ensembles H_F et H_C coïncident, et par conséquent on ne cherche qu'atteindre les objectifs suivants:

- Convergence des solutions réalisables H_F.
- Divergence des points intérieures.

Convergence des solutions réalisables $v \in H_F$:

La stabilité de l'ensemble des coins de l'hypercube $[0,1]^{N^2\times G}$ est assurée à partir des critères du point d'équilibre donnés par les équations (5), (6) et (7).
D'abord, pour minimiser le premier terme de l'équation (21), nous imposons la condition suivante:

$$\alpha \geq 0 \tag{23}$$

Ensuite, les conditions d'un point d'équilibre sont utilisées pour construire le théorème suivant:

Théorème 0.3

Si α, β et λ sont des paramètres réels vérifiant les conditions suivantes:

$$(S1) = \begin{cases} \alpha, \lambda \geq 0 \\ \beta \geq G \times N^4(\alpha M_1 + \lambda M_2) \end{cases} \tag{24}$$

avec $M_1 = \underset{\forall(p,i)\in\{1,..,N\}^2}{max}\ H_{p,i}$ et $M_2 = \underset{\forall(p,i)\in\{1,..,N\}^2}{max}\ D_{p,i}$.
Alors, chaque point v de H_F représente un point d'équilibre.

Proof. Comme mentionné ci-dessus, le point d'équilibre est atteint à partir de la dérivée première de la fonction d'énergie, dans laquelle nous allons discuter les différents états des neurones. A cet égard, la dérivée de E au point v_{ik} est calculée comme suit:

$$E_{ik}(v) = \frac{\partial E^H(v)}{\partial v_{ik}} = \frac{1}{2}\alpha \sum_{p=1}^{N^2}\sum_{j=1}^{N^2}\sum_{l=1}^{G} H_{p,i}H_{p,j}v_{jl} + \frac{1}{2}\lambda \sum_{p=1}^{N^2}\sum_{j=1}^{N^2}\sum_{l=1}^{G} D_{p,i}D_{p,j}v_{jl} - \alpha \sum_{p=1}^{N^2} y_p H_{p,i} + \frac{1}{2}\beta((1-2v_{ik})) \quad (25)$$

Les coefficients des noyaux qui construisent les opérateurs H et D sont généralement positifs, par conséquent nous construisons l'assertion suivante:

$$\min_{\{i/v_i=0\}} E_i(v) \geq 0 \Longrightarrow \beta - 2\alpha \sum_{p=1}^{N^2} y_p H_{p,i} \geq 0$$

Ce qui signifie que:

$$\min_{\{i/v_i=0\}} E_i(v) \geq 0 \Longrightarrow \beta \geq 2\alpha N^2 G M_1 \quad (26)$$

D'autre part, pour assurer la négativité de $\max_{\{i/v_i=1\}} E_i(v)$, nous obtenons à partir de l'équation (3.20):

$$\max_{\{i/v_i=1\}} E_i(v) \leq 0 \Longrightarrow \alpha \sum_{p=1}^{N^2}\sum_{j=1}^{N^2}\sum_{l=1}^{G} H_{p,i}H_{p,j} + \lambda \sum_{p=1}^{N^2}\sum_{j=1}^{N^2}\sum_{l=1}^{G} D_{p,i}D_{p,j} - \beta \leq 0$$

Par maximisation, l'équation précédente est remplacée par:

$$\min_{\{i/v_i=0\}} E_i(v) \leq 0 \Longrightarrow -\beta + GN^4(\alpha M_1^2 + \lambda M_2^2) \leq 0 \quad (27)$$

Enfin, comme $2\alpha GN^2 \leq \alpha GN^4$, alors la condition (26) peut être remplacée par l'inéquation (27). □

Divergence des solutions invalides $v \in H - H_F$:
Tout d'abord, nous définissons les concepts suivants:

Définition 3

Un point $v \in H$ est dit un point intérieur si et seulement si:

$$\exists\ v_{i,k} \in v\ /\ v_{i,k} \in\ ]0,1\ [$$

Définition 4

Soit v un point intérieur, v est dit un point de bord si et seulement si:

$$card\{(i,k) \in \{1,..,n+1\} \times \{1,..,N_{max}\}, \ \ / v_{i,k} \in]0,1[\} = 1$$

Définition 5

Soit $v \in H$ un point de bord tel que $v_{a,b} \in \ ]0,1\ [$, on dit que v' est le coin adjacent de v si:

$$v'_{i,j} = \begin{cases} v_{i,j} \text{ if } (i,j) \neq (a,b) \\ 1 \text{ or } 0 \text{ if } (i,j) = (a,b) \end{cases}$$

Dans **[83]**, Park a mentionné que tout point intérieur, n'étant pas un point de bord, a une probabilité nulle d'être un point d'équilibre du CHN.
Chaque point intérieur qui représente un point d'équilibre ne peut pas être un minimum local. Par conséquent il est nécessaire d'analyser la dérivée seconde de la fonction généralisée (4):

$$\frac{\partial^2 E^H(v)}{\partial {v_{ik}}^2} = -T_{ik,ik} = \alpha \sum_{p=1}^{N^2} H_{p,i}^2 + \lambda \sum_{p=1}^{N^2} D_{p,i}^2 - \beta \tag{28}$$

A cet effet, la divergence de chaque point de bord est assurée selon le théorème suivant **[102]**:

Théorème 0.4

Soit le point de bord $\{v \in H - H_C \setminus v_{i,k} \in \ ]0,1\ [\}$, si $E_{i,k}(v')$ vérifie:

$$E_{i,k}(v') \begin{cases} \notin \ [0, -T_{ik,ik}\ ] \ \ \forall \ \ (i,k) \setminus v'_{i,k} = 1 \\ \notin \ [T_{ik,ik}, 0\ ] \ \ \forall \ \ (i,k) \setminus v'_{i,k} = 0 \end{cases} \tag{29}$$

alors, v ne peut pas être un point d'équilibre.

Proof. Voir l'appendice A. □

Les conditions (29) peuvent être remplacées par les conditions suivantes:

$$\begin{cases} \underline{E}^0(v) \leq T_{ik,ik} \\ \underline{E}^1(v) \geq -T_{ik,ik} \end{cases} \tag{30}$$

Dans notre cas, ces conditions sont vérifiées car:

$$\underline{E}^0(v) = -2\alpha N^2 G M_1 + \beta \leq T_{ik,ik} = -\alpha \sum_{p=1}^{N^2} H_{p,i}^2 - \lambda \sum_{p=1}^{N^2} D_{p,i}^2 + \beta \tag{31}$$

et,

$$E^1(v) = \alpha \sum_{p=1}^{N^2}\sum_{j=1}^{N^2}\sum_{l=1}^{G} H_{p,i}H_{p,j} + \lambda \sum_{p=1}^{N^2}\sum_{j=1}^{N^2}\sum_{l=1}^{G} D_{p,i}D_{p,j} - \beta$$
$$\geq T_{ik,ik} = \alpha \sum_{p=1}^{N^2} H_{p,i}^2 + \lambda \sum_{p=1}^{N^2} D_{p,i}^2 - \beta \quad (32)$$

Le fonctionnement du CHN pour la résolution du problème de restauration d'images est décrit par l'algorithme suivant:

Data:
- Y: Image dégradée de taille N^2 ;
- H et D: Matrices . ;
- Initialisation de Itermax ;

Result: X: Image reconstruite

- Coder la variable X via la somme simple;
- Implémenter IRP dans un CHN avec l'Eq.(21);
- Extraire les poids T et le bias I de l'Eq.(22);
- Déterminer α, β et λ via le théorème 0.3 et l'Eq.(27);
- Initialiser aléatoirement v^0;
- Initialiser le compteur $iter \leftarrow 1$.;
- $E^{old1} \leftarrow 0$, $E^{new} = E^{old1} + 1$;

while $|E^{new} - E^{old1}| > \epsilon$ *ou* $iter \leq itermax$ **do**
  $E^{old1} \leftarrow E^{new}$;
  for *Chaque neurone* $v_{i,k}$ **do**
    $E^{old2} \leftarrow 0$;
    while $|E^{old2} - E^{new}| \geq \epsilon$ **do**
      $E^{old2} \leftarrow E^{new}$;
      Pour chaque neurone, mettre à jour $v_{i,k}$;
      $E^{new} \leftarrow E(v)$
    end
    $iter \leftarrow iter + 1$;
  end
end

Algorithm 2: Restauration d'images via les réseaux continus de Hopfield

0.4 Stabilité et estimation des paramètres du réseau de Hopfield continu: application à la résolution d'un nouveau modèle d'optimisation de restauration sélective

Dans les techniques de restauration d'images, les processus de reconstruction sont toujours accompagnés par des effets secondaires. En effet, le fonctionnement uniforme de ces méthodes sur toutes les composantes de l'image peut enlever les dégradation aux dépens de la qualité de certaines zones. Pour résoudre ce problème, nous proposons un nouveau modèle d'optimisation non linéaire à variables mixtes noté par (**MI-SIR**); Dans le modèle classique de la restauration d'images (15), un ensemble de variables de décision sont techniquement introduites selon une règle de décision, créant ainsi un modèle d'optimisation permettant d'appliquer un processus de restauration sélective[**57**, **58**]. Le modèle proposé est un problème mathématique non linéaire à variables mixtes dont la résolution par les méthodes exactes est assez compliquée. Pour cette raison, le réseau de Hopfield continu et l'algorithme génétique sont utilisés progressivement pour résoudre le modèle proposé[**59**].

0.4.1 Modèle proposé MI-SIR

Rappelons le modèle fondamental de restauration d'images:

$$\underset{X\in\{0,..,255\}^{N^2}}{Min} J(X) = \sum_{p=1}^{N^2}(Y_p - \sum_{i=1}^{N^2} H_{p,i}X_i)^2 + \frac{1}{2}\lambda \sum_{p=1}^{N^2}(\sum_{i=1}^{N^2} D_{pi}X_i)^2 \tag{33}$$

Malgré que certaines techniques de filtrage, notamment Médian et Gaussien, sont des outils assez puissants dans l'amélioration de qualité d'images, ces méthodes fonctionnent plus que le nécessaire surtout dans les zones originales non affectées par les dégradations. A cet effet, une nouvelle vision de restauration d'images est élaborée en utilisant l'optimisation non linéaire quadratique. Il s'agit de reconstruire les images originales, pendant que les zones non dégradées sont protégées. Pour ce faire, dans le modèle (33), nous introduisons des variables de décision $v \in \{0,1\}^{N2}$ définies comme suit:

$$v_i = \begin{cases} 1, \text{si le pixel } X_i \text{ est original} \\ 0, \text{ sinon} \end{cases} \tag{34}$$

Dans le but de protéger les zones non dégradées, au cours de la procédure de restauration, la solution courante X et une image de référence (Médian par exemple) sont collaborées dans une règle de décision notée R_i. Cette règle fournit deux sorties possibles selon les valeurs de v_i, elle s'exprime par:

$$R_i = v_i X_i + (1 - v_i)M_i \tag{35}$$

L'Eq.(35) permet de décider entre X_i et M_i celle qui génère un coût minimal de $J(X)$. Pour le choix de l'image de référence, il est important de prendre en considération une

image qui garde le maximum possible le caractère original dans cette image. Ensuite, nous remplaçons cette règle dans L'Eq.(33), le modèle est reformulé par:

$$\begin{aligned}\underset{(v,X)\in\{0,1\}^{N^2}\times\{0,..,255\}^{N^2}}{Min} E(v,X) = \frac{1}{2}\sum_{p=1}^{N^2}\sum_{i=1}^{N^2}\sum_{j=1}^{N^2} H_{p,i}H_{p,j}(v_iX_i+(1-v_i)M_i)(v_jX_j+ \\ (1-v_j)M_j) - \sum_{p=1}^{N^2}\sum_{i=1}^{N^2} H_{p,i}(v_iX_i+(1-v_i)M_i)Y_p \\ +\frac{1}{2}\lambda\sum_{p=1}^{N^2}\sum_{i=1}^{N^2}\sum_{j=1}^{N^2} D_{p,i}D_{p,j}(v_iX_i+(1-v_i)M_i)(v_jX_j+(1-v_j)M_j) \\ +\alpha\frac{1}{2}\sum_{p=1}^{N^2} Y_p^2 \quad (36)\end{aligned}$$

La règle stochastique (35) joue un rôle intéressant dans le processus de minimisation. En effet, la proposition (0.1) montre qu'une petite différence entre la valeur X_i et la valeur de référence M_i peut réduire considérablement le coût de la fonction d'énergie:

Proposition 0.1

Soient X, Y, et M trois vecteurs représentant respectivement les images originale, dégradée et Médiane de tailles $N \times N$. Si λ est un paramètre réel vérifiant $\lambda = 1$ et X' est une image de taille N^2 donnée par:

$$X_i' = \begin{cases} X_i, \ \forall i \neq k \\ M_k, \ else \end{cases} \quad (37)$$

alors la variation de J entre X et X' vérifie l'inégalité suivante:

$$|\Delta J(X)| = |J(X) - J(X')| \leq |X_k - M_k| N^2 \|X\|_1 \quad (38)$$

Proof. Paragraphe 4.3. □

0.4.2 Résolution du modèle MI-SIR via CHN-GA

Dans ce paragraphe nous nous intéressons à la résolution du modèle (36). Comme il est difficile d'appliquer une des méthodes exactes à ce genre de problèmes, nous suggérons d'hybrider l'algorithme génétique et le réseau continu de Hopfield pour la résolution du modèle proposé. Dans ce contexte, l'algorithme génétique est utilisé pour résoudre la partie discrète du modèle, alors que la partie binaire est traitée par CHN.

0.4.2.1 Résolution du modèle binaire via CHN

L'algorithme génétique et le réseau de Hopfield continu vont fonctionner progressivement à différentes itérations jusqu'à l'apparition de l'image désirée. A cet égard, on fixe X à l'itération $(t-1)$ et on résout le problème suivant:

$$\min_{v\in\{0,1\}^{N^2}} E(v,X) \tag{39}$$

Fonction de pénalité associée au modèle MI-SIR

En suivant les mêmes étapes de résolution par CHN, la fonction de pénalité associée à (39) est donnée par:

$$\begin{aligned} E^H(v,X) = \frac{1}{2}\alpha \sum_{p=1}^{N^2}\sum_{i=1}^{N^2}\sum_{j=1}^{N^2} H_{p,i}H_{p,j}(v_iX_i+(1-v_i)M_i)(v_jX_j+(1-v_j)M_j) \\ - \alpha \sum_{p=1}^{N^2}\sum_{i=1}^{N^2} H_{p,i}(v_iX_i+(1-v_i)M_i)Y_p \\ + \frac{1}{2}\lambda \sum_{p=1}^{N^2}\sum_{i=1}^{N^2}\sum_{j=1}^{N^2} D_{p,i}D_{p,j}(v_iX_i+(1-v_i)M_i)(v_jX_j+(1-v_j)M_j) \\ + \alpha\frac{1}{2}\sum_{p=1}^{N^2} Y_p^2 + \frac{\gamma}{2}\sum_{i=1}^{N^2} v_i(1-v_i) \end{aligned} \tag{40}$$

Où α, λ et γ sont des paramètres réels qu'on va estimer à partir de l'analyse de convergence du réseau. Dans le but de simplifier l'expression (40), on note par $\Psi^{\alpha,\lambda}_{p,i,j}$ le terme $\alpha H_{p,i}H_{p,j} + \lambda D_{p,i}D_{p,j}$. La nouvelle expression de $E(v,X)$ est donc écrite sous la forme suivante:

$$\begin{aligned} E^H(v,X) = \frac{1}{2}\sum_{p=1}^{N^2}\sum_{i=1}^{N^2}\sum_{j=1}^{N^2} \Psi^{\alpha,\lambda}_{p,i,j} \prod_{k\in\{i,j\}} (v_kX_k+(1-v_k)M_k) \\ - \alpha \sum_{p=1}^{N^2}\sum_{i=1}^{N^2} H_{p,i}(v_iX_i+(1-v_i)M_i)Y_p \\ + \alpha\frac{1}{2}\sum_{p=1}^{N^2} Y_p^2 + \frac{\gamma}{2}\sum_{i=1}^{N^2} v_i(1-v_i) \end{aligned} \tag{41}$$

Où $\prod$ désigne l'opérateur du produit.

Paramètres du CHN

Par implémentation du modèle (41) sur un CHN, et par comparaison avec l'équation de Lyapounov (4), les paramètres du réseau sont extraits comme suit :

$$T_{i,j} = -(X_i - M_i)(X_j - M_j) \sum_{p=1}^{N^2} \Psi_{p,i,j}^{\alpha,\lambda} + \frac{\gamma}{2}\delta_{ij} \tag{42}$$

et

$$I_i = -\sum_{p=1}^{N^2}(\sum_{j=1}^{N^2} \Psi_{p,i,j}^{\alpha,\lambda}(v_i(X_i M_j - M_i M_j)) + \alpha H_{p,i} Y_p(X_i - M_i)) - \frac{\gamma}{2} \tag{43}$$

Estimation des paramètres du modèle MI-SIR

Dans cette partie, nous cherchons à atteindre les deux objectifs suivants:

- **Convergence de l'ensemble des solution réalisables H_F:**
  Comme mentionné ci-dessus, la recherche du point d'équilibre se base sur la dérivée première de $E^H(v, X)$ donnée par:

$$E_i(v, X) = \frac{\partial E(v, X)}{\partial v_i} = \alpha(X_i - M_i) \sum_{p=1}^{N^2}(\sum_{j=1}^{N^2} \Psi_{p,i,j}^{\alpha,\lambda}(v_j X_j + (1 - v_j)M_j) - H_{p,i} Y_p) + \frac{\gamma}{2}(1 - 2v_i) \tag{44}$$

  Pour assurer la minimisation du terme des moindres carrées, nous imposons la condition suivante:

$$\alpha \geq 0 \tag{45}$$

  Supposons que H et D sont deux matrices positives. Si $\alpha, \lambda \geq 0$ alors, à partir d'un certain rang s, nous obtenons l'inégalité suivante:

$$\sum_{j=1}^{s} \Psi_{p,i,j}^{\alpha,\lambda}(v_j X_j + (1 - v_j)M_j) \geq H_{p,i} Y_p \quad \forall i, p = 1, .., N^2 \tag{46}$$

  On note par S le terme suivant:

$$S = \max_{(i,p)\in\{1,2,3,..,N^2\}^2} \{\sum_{j=1}^{s} \Psi_{p,i,j}^{\alpha,\lambda}(v_j X_j + (1 - v_j)M_j) - H_{p,i} Y_p\}$$

  S est une valeur positive que nous allons exploiter pour extraire les conditions d'équilibre. Les conditions (47), (48) sont obtenues à partir des conditions du point d'équilibre appliquées à $v \in H_C$:

$$\underline{E}^0(v) \geq 0 \ \forall v_i = 0 \Longrightarrow -255\alpha N^2 S + \frac{\gamma}{2} \geq 0 \tag{47}$$

$$\underline{E}^1(v) \leq 0 \ \forall v_i = 1 \Longrightarrow 255\alpha N^2 S - \frac{\gamma}{2} \leq 0 \tag{48}$$

La quantité $X_i - M_i$ est bornée par -255 et 255, ce qui correspond au cas du bruit impulsive (Figure 4.2).

- **Divergence des points intérieurs $H - H_C$:**
En utilisant les mêmes techniques de stabilité utilisées dans la section précédente, la stabilité des points intérieurs est évitée avec les conditions du théorème suivant:

Théorème 0.5

Soit le point de bord $\{v \in H - H_C \ / \ v_i \in]0,1[\}$, si γ est suffisamment petite et v' est le coin adjacent de v vérifiant:

$$E_i(v') \begin{cases} \notin [0, (X_i - M_i)^2 \sum_{p=1}^{N^2} \Psi_{p,i,i}^{\alpha,\lambda} - \frac{\gamma}{2}] & \forall i / v_i' = 1 \\ \notin [-(X_i - M_i)^2 \sum_{p=1}^{N^2} \Psi_{p,i,i}^{\alpha,\lambda} + \frac{\gamma}{2}, 0] & \forall i / v_i' = 0 \end{cases} \tag{49}$$

alors, le point de bord v ne peut pas être un point d'équilibre.

Proof. Premièrement, il faut assurer la négativité des auto- connections $T_{i,i}$ données par:

$$T_{i,i} = -(X_i - M_i)^2 \sum_{p=1}^{N^2} \Psi_{p,i,i}^{\alpha,\lambda} + \frac{\gamma}{2}$$

Ce qui est assuré avec γ suffisamment petite. Deuxièmement, le résultat (49) est retrouvé grâce au théorème 0.4. □

0.4.2.2 Résolution du modèle discret via GA

Pour résoudre des problèmes complexes de grande dimension, il est approprié de choisir une Méta-heuristique au lieu d'une méthode exacte. L'une de ces méthodes est l'algorithme génétique, il s'agit d'un processus inspiré de l'évolution biologique des individus [**46**]. Dès sa création, l'algorithme génétique a fonctionné parfaitement dans la résolution des problèmes combinatoires [**57**, **58**].
En effet, ce type de méthodes converge en moins de temps et peut s'adapter à n'importe quel problème. De plus, la convergence peut être plus rapide grâce à la population initiale qui commence par plusieurs solutions possibles. La résolution de la partie discrète du modèle MI-SIR est résumée dans les points suivants:

- **Représentation et initialisation:** La population prend la forme d'une matrice ou chaque ligne représente un chromosome, ce dernier est codé par un vecteur des valeurs dans l'ensemble $\{0, .., 255\}$ (Figure 4.3). La population initiale peut être générée aléatoirement, mais un choix particulier peut garantir une convergence rapide. Par exemple, si l'image à restaurer est peu ou partiellement dégradée, alors une initialisation par les valeurs de l'image dégradée assure une convergence rapide puisque le processus de résolution ne nécessite que quelques mutations pour atteindre la solution.

- **Fitness:** Dans cette étape, une valeur appelée fitness est affectée à chaque chromosome, cette valeur mesure la performance du chromosome autant que solution. Dans notre travail, nous adoptons la fitness suivante:

$$f(v, X) = \frac{1}{1 + E(v, X)} \tag{50}$$

  Où la minimisation de la fonction objective correspond à la maximisation de la fitness.

- **Sélection:** Au cours de l'algorithme, l'étape de sélection consiste à choisir les chromosomes de grande fitnesse. Plusieurs méthodes de sélection existent dans la littérature. Dans ce travail, nous optons pour la sélection par la roulette. C'est une technique inspirée des roues de loterie où chacun des individus de la population est associé à un secteur d'une roue. L'angle du secteur étant proportionnel à la qualité de l'individu qu'il représente. On tourne la roue et on obtient un individu. Les tirages des individus sont ainsi pondérés par leur qualité. Logiquement, les meilleurs individus ont plus de chance d'être croisés et de participer à l'amélioration de notre population.
- **Reproduction: Mutation et croisement**: Les opérateurs de reproduction permettent de garantir la diversification et l'intensification de l'espace de recherche grâce aux opérateurs de croisement et mutation. Les croisements permettent de simuler des reproductions d'individus dans le but d'en créer des nouveaux. Il est tout à fait possible de faire des croisements aléatoires. Toutefois, une solution largement utilisée est d'effectuer des croisements multi-points. Une autre solution que le croisement pour créer de nouveaux individus est de modifier ceux déjà existants. Une fois de plus, le hasard va nous être d'une grande utilité. Il peut s'avérer efficace de modifier aléatoirement quelques individus de notre population en modifiant un gène ou un autre. Dans notre cas, il s'agit de changer un gène, choisi avec une probabilité de mutation généralement petite, par une valeur dans l'ensemble$\{0, .., 255\}$.

Le fonctionnement du modèle proposé pour la restauration sélective est décrit dans l'algorithme suivant:

Algorithme de CHN-MI-SIR

Entrées:

•Y: Image dégradée de taille N^2.
• Matrices H et D.
• Itermax.
Image de référence M.

Sorties:

•Image restaurée X^{**}

Initialisation:

- •Initialiser le vecteur $X^{**} \leftarrow Y$.
- • Initialiser le compteur $iter \leftarrow 1$.
- • $E^{old} \leftarrow 0$, $E^{new} = E^{old} + 1$.

while($|E^{new} - E^{old1}| > \epsilon$ **ou** $iter \leq itermax$**)**

- •$E^{old} \leftarrow E^{new}$;
- • Utiliser le réseau CHN pour trouver la solution binaire:

$$v^* = \underset{v}{argmin} E(v, X^{**});$$

- • Utiliser l'algorithme génétique pour trouver la solution discrète : $X^* = \underset{X}{argmin} E(v^*, X)$;
- • Construire la solution X^{**} en choisissant ses pixels parmi X^* et X^{**} selon la solution v^*;
- • $E^{new} \leftarrow E(v^*, X^{**})$;
- • $iter \leftarrow iter + 1$;

end while
Return

- X^{**}: Image désirée;

0.5 Stabilité et estimation des paramètres du réseau de Hopfield continu: résolution d'un nouveau modèle d'optimisation du choix de l'architecture du PRSOM: application au regroupement des données

Les cartes auto-organisatrices de Kohonen font partie des réseaux de neurones à apprentissage non supervisé[**60**, **65**, **66**]. Ces cartes sont utilisées pour cartographier un espace réel, c'est-à-dire pour étudier la répartition de données dans un espace à grande dimension. Ce réseau comporte deux couches de neurones :

- La 1ère couche : Elle représente une couche d'entrée qui comprend les neurones d'entrée auxquels on attribue des poids choisis au hasard.
- La 2ème couche : Elle comprend les neurones de sortie qui seront représentés par les nouveaux poids ajustés.

Dans cette section, nous proposons un nouveau modèle d'optimisation de l'architecture de la carte auto-organisatrice probabiliste (PRSOM). Nous suggérons une fonction objective adéquate sous des contraintes linéaires et non linéaires. Ainsi, le modèle obtenu est un problème mathématique à variables mixtes sous contraintes dont la résolution via les méthodes analytiques est très difficile, voire impossible. A cette fin, le réseau CHN et les gradients de la vraisemblance sont techniquement et progressivement utilisés pour résoudre le modèle proposé [**55**].

0.5.1 Nouveau modèle d'optimisation d'architecture du PRSOM

Le modèle probabiliste de la carte auto-organisatrice a été découvert par Badran et ses collègues en 1998. Dans ce modèle, une fonction de densité Gaussienne f_k est associée à chaque neurone k de la carte [**5**, **67**]. De même, la notion de voisinage permet d'introduire un ensemble de mélanges de Gaussiennes. Dans ce sens, ces deux notions permettent de construire une fonction de densité définie comme un mélange des mélanges probabilistes où chaque neurone de la carte est associé à une composante du mélange. Avant de définir cette fonction de densité, nous commençons par décrire l'architecture de la carte auto-organisatrice probabiliste, qui est illustrée dans la Figure 5. Le modèle probabiliste duplique la carte classique C en deux cartes similaires C^1 et C^2, munies de la même topologie que C.
De manière plus claire, le réseau possède une architecture à trois couches:

- Une couche d'entrée qui distribue canoniquement les observations de l'espace d'entrée. Elle est constituée de p neurones qui ont la dimension de l'espace d'entrée;
- La couche C^1 qui est constituée de K neurones, où chaque neurone n_k^1 est associé à une fonction Gaussienne f_k;
- La couche C^2 est la couche de décisions (sortie du réseau).

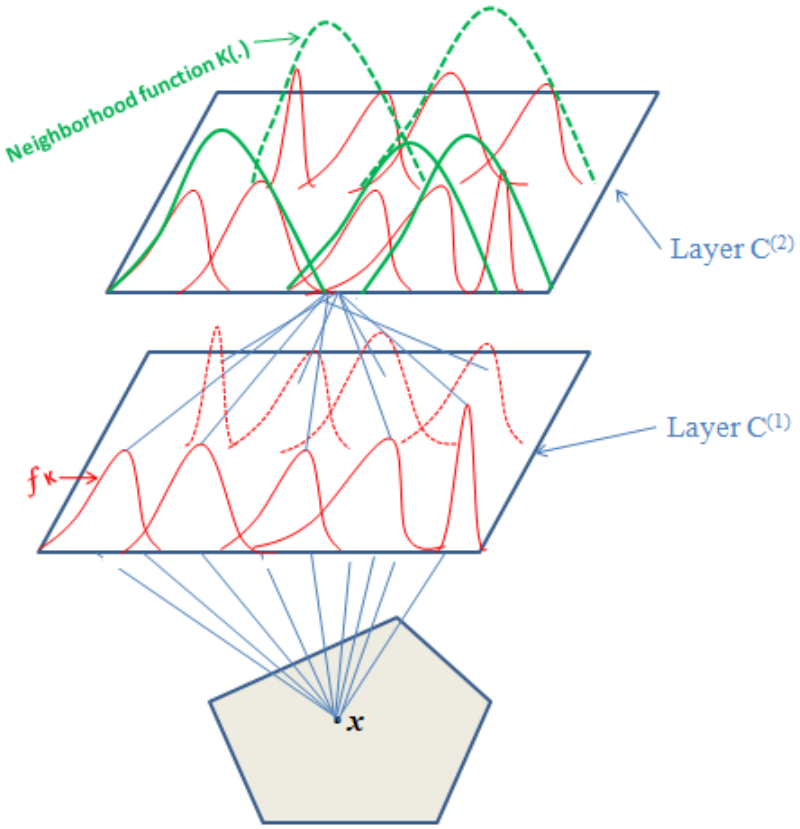


Figure 5: Modèle probabiliste de la carte auto-organisatrice

0.5.1.1 Nouveau modèle d'optimisation d'architecture du PRSOM

Dans cette partie, nous proposons un nouveau modèle d'optimisation pour la réduction la carte auto-organisatrice probabiliste. Dans ce contexte, nous suggérons un modèle d'optimisation non linéaire à variables-mixtes sous contraintes linéaires et non linéaires. Pour formaliser le modèle, nous avons besoin de définir les paramètres ci-dessous:

Paramètres

- n : Nombre d'ensembles d'observation;
- N : Nombre optimal de neurones dans la carte topologique du PRSOM ;
- N_{max} : Nombre maximal de neurones dans la carte topologique du PRSOM.

Variables

- X = $x_{i,j}$: Matrice des éléments de la base d'apprentissage ;
- V = $v_{i,j}$: Matrice des variables binaires ;
- W = $w_{i,j}$: Matrice des vecteurs référents ;
- $\sigma = \sigma_j$: Matrice de covariance.

Fonction objective: En se basant sur le travail de Bishop sur les modèles de mélanges Gaussiennes **[33]**, nous définissons la fonction objective associée à la carte auto-organisatrice

probabiliste par:

$$Max \;\; p(W,\Sigma,v) = \prod_{i=2}^{n+1}\prod_{j=1}^{N_{max}}(\pi_j \sum_{k=1}^{N_{max}} \delta(n_k^1, n_j^2) f_k(x_{i-1}, w_k, \sigma_k))^{v_{i,j}v_{1,j}} \tag{51}$$

Où:

- f_k: densité Gaussienne.
- $\delta(n_k^1, n_j^2)$: fonction de voisinage entre le i^{me} neurone de la couche C^1 et le j^{me} neurone de la couche C^2.

Pour simplifier l'expression de (51), on approxime $p(W,\Sigma,V)$ par log-vraisemblance. Donc le modèle (51) est approché par la fonction suivante:

$$Max(\;\; log[p(W,\Sigma,v)] = \sum_{i=2}^{n+1}\sum_{j=1}^{N_{max}} v_{i,j}v_{1,j} log[(\pi_j \sum_{k=1}^{N_{max}} \delta(n_k^1, n_j^2) f_k(x_{i-1}, w_k, \sigma_k))]) \tag{52}$$

Ce qui est équivalent à:

$$Min \;\; E(W,\Sigma,v) = -\sum_{i=2}^{n+1}\sum_{j=1}^{N_{max}} v_{i,j}v_{1,j} log[(\pi_j \sum_{k=1}^{N_{max}} \delta(n_k^1, n_j^2) f_k(x_{i-1}, w_k, \sigma_k))] \tag{53}$$

Contraintes d'affectation: Un exemple ne peut être représenté qu'avec un seul neurone parmi tous les neurones de la carte PRSOM, la contrainte qui assure cette affectation est décrite par la formule ci-dessous:

$$\sum_{j=1}^{N_{max}} v_{i,j} = 1 \quad \forall i \in \{2,...,n+1\} \tag{54}$$

Contraintes de transmission: Cette contrainte communique entre les variables $v_{ij/i=2,..,n+1}$ et v_{1j}, c'est à dire que si au moins un exemple i est affecté au neurone j, alors ce dernier ne sera pas exclu de l'architecture:

$$\sum_{j=1}^{N_{max}} (1 - v_{1,j}) \sum_{i=2}^{n+1} (v_{i,j}) = 0 \tag{55}$$

En résumé, le modèle proposé permet d'assurer l'affectation des exemples aux neurones utiles dans la carte, et restaurer les neurones représentatifs des données. Ce modèle prend la forme d'un problème d'optimisation non linéaire à variables mixtes.

$$P = \begin{cases} MinE(W,\Sigma,v) = \sum_{i=2}^{n+1}\sum_{j=1}^{N_{max}} v_{i,j}v_{1,j}C_{i-1,j} \\ \sum_{j=1}^{N_{max}} v_{i,j} = 1 \quad \forall i \in \{2,...,n+1\} \\ \sum_{j=1}^{N_{max}} (1 - v_{1,j}) \sum_{i=2}^{n+1} (v_{i,j}) = 0 \\ w_j \in IR^d \quad \forall j \in \{1,...,N_{max}\} \\ \sigma_j \in IR^+ \quad \forall j \in \{1,...,N_{max}\} \\ v \in \{0,1\}^{(n+1)N_{max}} \end{cases}$$

(56)

Avec $C_{i-1,j} = -log[(\pi_j \sum_{k=1}^{N_{max}} \delta(n_k^1, n_j^2) f_k(x_{i-1}, w_k, \sigma_k))]$ et $\Sigma_j = \sigma_j * I_d$.

0.5.2 Résolution du modèle proposé

Comme illustré ci-dessus, le problème (P) appartient aux classes des problèmes d'optimisation à variables mixtes. Comme le problème (P) est analytiquement difficile à résoudre, alors nous proposons une alternative composée de deux phases: affectation-optimisation et minimisation.

0.5.2.1 Fonctionnement du CHN pour la phase affectation-optimisation

Dans cette phase, nous fixons w et σ obtenues dans la phase de minimisation précédente. Ainsi, nous obtenons un problème d'optimisation quadratique à variables binaires. Il s'agit de résoudre le problème suivant:

$$(P_1) = \begin{cases} \underset{v \in \{0,1\}^{(n+1)Nmax}}{min} E(.,.,v) = \sum_{i=2}^{n+1} \sum_{j=1}^{N_{max}} v_{1j} v_{i,j} C_{i-1,j} \\ e_i(v) = \sum_{j=1}^{N_{max}} v_{i,j} = 1 \qquad\qquad i = 2, \ldots, n+1. \\ H(v) = \sum_{j=1}^{N_{max}} ((1 - v_{1j}) \sum_{i=2}^{n+1} v_{i,j}) = 0 \end{cases} \tag{57}$$

Ensuite, nous construisons une fonction d'énergie qui pénalise la fonction objective, ainsi que les contraintes violées. Cette fonction est définie comme suit:

$$E^H(v) = \frac{\alpha}{2} \sum_{i=2}^{n+1} \sum_{j=1}^{N_{max}} v_{1,j} v_{i,j} C_{i-1,j} + \frac{\rho}{2} H(v) + \frac{\beta}{2} \sum_{i=2}^{n+1} (e_i(v) - 1)^2 + \frac{\gamma_1}{2} \sum_{j=1}^{N_{max}} \sum_{i=1}^{n+1} v_{i,j}(1 - v_{i,j}) \tag{58}$$

Où α, β, ρ et γ_1 sont des paramètres réels à estimer dans l'analyse de stabilité du CHN. Le terme $v_{i,j}(1 - v_{i,j})$ garantit la stabilisation des neurones sur les coins de l'hypercube $[0, 1]$.

Paramètres du CHN

Dans ce paragraphe, nous proposons un réseau CHN adéquat au problème (57). Ce réseau est caractérisé par les paramètres suivants:

- États des neurones:
  $v = \{ v_{i,j} \ / \ i = 1, .., n+1 \quad et \quad j = 1, .., N_{max} \}$.

$$v = \begin{bmatrix} v_{1,1} & .. & v_{1,j} & .. & v_{1,N_{max}} \\ - - - & - - - - & - - - - & - - - - & - - - - - \\ v_{2,1} & .. & v_{2,j} & .. & v_{2,N_{max}} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ v_{i,1} & .. & v_{i,j} & .. & v_{i,N_{max}} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ v_{n+1,1} & .. & v_{n+1,j} & .. & v_{n+1,N_{max}} \end{bmatrix}$$

- Matrices des poids:
  $T = \{T_{ij,kl} \ / \ i,k = 1,..,(n+1) \quad et \quad j,l = 1,..,N_{max}\}$

En comparant les équations (58) et (4), nous formons ainsi un système donnant la matrice des poids et le vecteur des bruits:

$$S = \begin{cases} T_{1b,1d} = \gamma_1 \delta_{bd} \\ T_{ab,cd} = -\beta + \gamma_1 \delta_{ac} \delta_{bd} \\ T_{1b,cd} = -(\alpha C_{c-1,b} - \rho)\delta_{bd} \\ I_{1j} = -\frac{\gamma_1}{2} \\ I_{ab} = -\frac{\rho}{2} + \beta - \frac{\gamma_1}{2} \end{cases} \tag{59}$$

où $a,c = 2,..,n+1$, $b,d = 1,..,N_{max}$ et δ_{ij} est égale à 1 si (i=j) et 0 sinon.

Estimation des paramètres de la fonction d'énergie

Dans cette partie, nous discutons la stabilité du CHN associé au problème (P_1) sur les trois ensembles H_F, $H_C - H_F$ et $H_- H_F$:

- $H_C = \{0,1\}^{N_{max}(n+1)}$: Ensemble des coins de l'Hypercube $H = [0,1]^{N_{max}(n+1)}$.
- H_F: Ensemble des solutions réalisables donné par:

  $$H_F = (H_{F1} \cup H_{F2}) \cap H_{F3}$$

  où

  $$H_{F1} = \{V \in \{0,1\}^{(n+1)N_{max}} \ / \ v_{1,j} = 1 \ et \ \sum_{i=2}^{n+1} v_{i,j} \geq 1\} \tag{60}$$

  $$H_{F2} = \{V \in \{0,1\}^{(n+1)N_{max}} \ / \ v_{1,j} = 0 \ et \ \sum_{i=2}^{n+1} v_{i,j} = 0\} \tag{61}$$

  et

  $$H_{F3} = \{V \in \{0,1\}^{(n+1)N_{max}} \ / \ \sum_{j=1}^{N_{max}} v_{i,j} = 1 \ \forall i = 2,..,n+1\} \tag{62}$$

De façon similaire que les sections précédentes, nous commençons par l'estimation des paramètres de la fonction d'énergie à travers les solutions réalisables. A cette fin, nous construisons les dérivées partielles de la fonction d'énergie comme suit:

$$E_{1,b}(v) = \frac{\partial E^H(v)}{\partial v_{1,b}} = \frac{\alpha}{2} \sum_{i=2}^{n+1} v_{i,b} C_{i-1,b} + \frac{\gamma_1}{2}(1 - 2v_{1,b}) - \frac{\rho}{2} \sum_{i=2}^{n+1} v_{i,b} \tag{63}$$

$$E_{a,b}(v) = \frac{\partial E^H(v)}{\partial v_{a,b}} = \frac{\alpha}{2} v_{1,b} C_{a-1,b} + \frac{\rho}{2}(1 - v_{1,b}) + \beta(e_a(v) - 1) + \frac{\gamma_1}{2}(1 - 2v_{a,b}) \quad (64)$$

$\forall\ (a,b) \in \{2,..,n+1\} \times \{1,..,N_{max}\}$.

Pour minimiser la fonction objective, nous imposons la contrainte suivante:

$$\alpha \geq 0 \quad (65)$$

Dans la suite, les Eqs.(66), (67), (68) et (69) sont extraites à partir des conditions d'équilibre appliquées à l'ensemble H_F. D'où, nous obtenons les inégalités suivantes:

$$v_{1,b} = 0 \Longrightarrow \frac{\gamma_1}{2} \geq 0 \quad (66)$$

Puis,

$$v_{a,b} = 0 \Longrightarrow \begin{cases} \alpha C_{a-1,b} + \gamma_1 \geq 0 & \text{if } v_{1,b} = 1 \\ \rho + \gamma_1 \geq 0 & \text{if } v_{1,b} = 0 \end{cases}$$

Les deux cas du dernier système peuvent être remplacés par la condition suivante:

$$v_{a,b} = 0 \Longrightarrow \gamma_1 \geq max\{-\alpha m, \rho\} \quad (67)$$

Avec $m = \min\limits_{\forall (i,b) \in \{2,..,n+1\} \times \{1,..,N_{max}\}} C_{i-1,b}$.

Dans le cas où $v_{1,b} = 1$, il existe $k \in \{1,..,n\}$ tel que $\sum_{i=1}^{n} v_{i+1,b} = k$, par conséquent on obtient:

$$v_{1,b} = 1 \Longrightarrow \alpha k C_{i,b} - k\rho - \gamma_1 \leq 0,$$

Par maximisation des coefficients de C, on retrouve:

$$v_{1,b} = 1 \Longrightarrow \alpha n M - \rho - \gamma_1 \leq 0, \qquad \rho > 0 \quad (68)$$

où $M = \max\limits_{\forall (i,b) \in \{2,..,n+1\} \times \{1,..,N_{max}\}} C_{i-1,b}$.

La dernière condition est donnée par:

$$v_{a,b} = 1 \Longrightarrow \alpha M - \gamma_1 \leq 0 \quad (69)$$

Enfin, nous regroupons les conditions (65)-(69) dans le système suivant:

$$S_1 = \begin{cases} \alpha, \rho, \gamma_1 \geq 0 \\ \gamma_1 \geq max\{-\alpha m, \rho\} \\ \alpha n M - \rho - \gamma_1 \leq 0 \\ \alpha M - \gamma_1 \leq 0 \end{cases} \quad (70)$$

En plus de la convergence des solutions réalisables, les paramètres du CHN doivent assurer l'instabilité du CHN sur les l'ensemble $H_C - H_F$. Pour ce faire, nous définissons d'abord le concept d'adjacence.

Définition 6

Un vecteur $A_{i,j}(v) \in H_C$ est dit un adjacent de v si et seulement si:

$$A_{i,j}(v_{k,l}) = \begin{cases} v_{k,l}, & si \quad (k,l) \neq (i,j) \\ 1 - v_{k,l}, & si \quad (k,l) = (i,j) \end{cases}$$

En général, les sommets adjacents de V sont définis par $A = A^+ \bigcup A^-$, où:

$$A^+ = \{A_{i,j}(v) / v_{i,j} = 0\}$$

$$A^- = \{A_{i,j}(v) / v_{i,j} = 1\}$$

Par utilisation du concept d'adjacence, nous identifions les solutions invalides via le lemme suivant (Paragraphe 5.5.1.2):

Lemme 0.1

Soit $v \in H_C - H_F$, v^+ est une solution invalide si :

- $\exists i \in \{2, .., n+1\}$ tel que $\sum_{j=1}^{N_{max}} v_{i,j} \geq 2$
- $\{e_i(v) = 1, \forall i = 2, .., n+1\} \cap \{\exists! j = 1, ..N_{max} \ / \ v_{1,j} = 1 \text{ et } \sum_{i=2}^{n+1} v_{i,j} = 0\}$

Le théorème 0.6 présente les conditions suffisantes permettant d'éviter l'équilibre des solutions invalides de $H_C - H_F$.

Théorème 0.6

Soit v une solution invalide binaire. Si α, β, ρ et γ_1 sont des paramètres réels vérifiant:

$$S_2 = \begin{cases} \alpha m - \rho \geq 0 & \alpha, \rho \geq 0 \\ \beta - \frac{\gamma_1}{2} \geq 0 \\ -\gamma_1 \geq 0 \\ \beta \geq 0 \end{cases} \tag{71}$$

tel que:

$$\max_{V \in H_C - H_F / \sum_{i=1}^{n+1} \sum_{j=1}^{N_{max}} v_{i,j} = (n+1)N_{max} - 1} \bar{E^0}(v) < 0 < \min_{V \in H_C - H_F / \sum_{i=1}^{n+1} \sum_{j=1}^{N_{max}} v_{i,j} = (n+1)N_{max}} \bar{E^1}(v) . \tag{72}$$

Alors v ne peut pas être un point d'équilibre.

Proof. Tout d'abord, nous définissons le lemme suivant (Paragraphe 5.5.1.2):

Lemme 0.2

Soient $\{\zeta_k\}_{k\in\{1,2,.....,(n+1)N_{max}-1\}}$ et $\{\Psi_k\}_{k\in\{1,2,.....,(n+1)N_{max}\}}$ deux suites numériques définies par:

$$\zeta_k = \max_{V\in H_C-H_F/\sum_{i=1}^{n+1}\sum_{j=1}^{N_{max}} v_{i,j}=k} \underset{-}{E^0}(V)$$

et

$$\Psi_k = \min_{V\in H_C-H_F/\sum_{i=1}^{n+1}\sum_{j=1}^{N_{max}} v_{i,j}=k} \underset{-}{E^{-1}}(v)$$

Si α, β, ρ et γ_1 sont des paramètres réels vérifiant les assertions suivantes:

$$S_2 = \begin{cases} \alpha m - \rho \geq 0 \\ \beta - \gamma_1 \geq 0 \\ -\gamma_1 \geq 0 \qquad \alpha, \rho, \beta \geq 0 \end{cases} \tag{73}$$

Alors, les suites ζ_k et Ψ_k sont respectivement croissante et décroissante.

D'après le Lemme 0.2, ζ_k (Ψ_k) est une suite croissante (décroissante) sous les conditions du système (73). Ce qui signifie que $\zeta_k \leq \zeta_{(n+1)N_{max}-1}\ \forall k \in \{1,2,..,(n+1)N_{max}-1\}$ ($\Psi_k \geq \Psi_{(n+1)N_{max}}\ \forall k \in \{1,2,..,(n+1)N_{max}\}$) et par application des critères du point d'équilibre, nous obtenons le résultat demandé. □

Dans la phase suivante, nous cherchons à garantir l'instabilité des points $v \in H - H_F$. A cet effet, nous nous concentrons sur le cas des points intérieurs qui sont de bord. Pour réaliser cette tâche, nous définissons le concept du coin adjacent.

Définition 7

Soit $v \in H$ un point de bord tel que $v_{a,b} \in [0,1\,]$, v' est dit un coin adjacent de v si:

$$v'_{i,j}=\begin{cases} v_{i,j} \text{ si } (i,j) \neq (a,b) \\ 1 \text{ or } 0 \text{ si } (i,j) = (a,b) \end{cases}$$

En se basant sur ce concept, la non convergence des points de bords est donc donnée par le théorème suivant:

Théorème 0.7

Soit le point de bord $\{v \in H - H_C \ / \ v_{i,j} \in]0,1\,[\ \}$, si $\gamma_1 < 0$, $\beta \geq 0$ et v' est le coin adjacent de v vérifiant:

$$E_{i,j}(v') \begin{cases} \notin (0, -min(-\beta+\gamma_1, \gamma_1))\ \ (i,j)/v'_{i,j} = 1 \\ \notin (min(-\beta+\gamma_1, \gamma_1), 0)\ \ (i,j)/v'_{i,j} = 0 \end{cases} \tag{74}$$

Alors, l'instabilité de chaque point de bord est évitée.

Proof. La démonstration est complètement basée sur le théorème 0.4 (paragraphe 5.5.1.2) □

Comme $H_C - H_F$ est un sous ensemble de $H - H_F$, alors il est très important de maintenir la consistance entre les théorèmes 0.6 et 0.7. C'est l'objet du théorème suivant:

Théorème 0.8

Soit $v \in h_C - H_F$, si les assertions suivantes sont vérifiées:

$$S_3 = \begin{cases} \alpha m - \rho \geq -min(-\beta + \gamma_1, \gamma_1) & \alpha, \rho \geq 0 \\ \beta - \gamma_1 \geq -min(-\beta + \gamma_1, \gamma_1) \\ -\gamma_1 \geq -min(-\beta + \gamma_1, \gamma_1) \\ \beta \geq -min(-\beta + \gamma_1, \gamma_1) \end{cases} \tag{75}$$

Alors, v ne représente pas un point d'équilibre.

Proof. En se basant sur les théorème (0.6) et (0.7), les nouveaux paramètres du système S_3 sont aisément obtenus.

□

0.5.2.2 Phase de minimisation

Dans cette étape, la variable v est fixée à la valeur calculée dans la phase d'affectation-optimisation. Ainsi, le modèle obtenu est un problème à variables continues donné par:

$$P2 = \left\{ \min_{(w,\sigma) \in IR^d \times IR^+} E(w, \sigma, \bullet) = \sum_{i=2}^{n+1} \sum_{j=1}^{N_{max}} v_{1,j} v_{i,j} C_{i-1,j} \right. \tag{76}$$

A partir du gradient de la fonction d'énergie, la solution assure le système suivant:

$$\begin{cases} \frac{\partial E}{\partial w_k} = 0 & k = 1, \ldots, N_{max}. \\ \frac{\partial E}{\partial \sigma_k} = 0 & k = 1, \ldots, N_{max}. \end{cases} \tag{77}$$

Ce qui signifie que:

$$w_k = \frac{\sum_{i=2}^{n+1} \sum_{j=1}^{N_{max}} v_{1,j} v_{i,j} x_{i-1} \frac{\delta(n_k^1, n_j^2) f_k(x_{i-1})}{\sum_{r=1}^{N_{max}} \delta(n_k^1, n_j^2) f_r(x_{i-1})}}{\sum_{i=2}^{n+1} \sum_{j=1}^{N_{max}} v_{1,j} v_{i,j} \frac{\delta(n_k^1, n_j^2) f_k(x_{i-1})}{\sum_{r=1}^{N_{max}} \delta(n_k^1, n_j^2) f_r(x_{i-1})}} \tag{78}$$

et

$$\sigma_k = \frac{\sum_{i=2}^{n+1} \sum_{j=1}^{N_{max}} v_{1,j} v_{i,j} x_{i-1} \frac{||w_k - x_{i-1}||^2 \delta(n_k^1, n_j^2) f_k(x_{i-1})}{\sum_{r=1}^{N_{max}} \delta(n_k^1, n_j^2) f_r(x_{i-1})}}{d \sum_{i=2}^{n+1} \sum_{j=1}^{N_{max}} v_{1,j} v_{i,j} \frac{\delta(n_k^1, n_j^2) f_k(x_{i-1})}{\sum_{r=1}^{N_{max}} \delta(n_k^1, n_j^2) f_r(x_{i-1})}} \tag{79}$$

Enfin, la résolution du modèle d'optimisation d'architecture par CHN est décrite dans l'algorithme suivant:

L'algorithme du **CHN-PRSOM**

Input:

- Dataset $D = \{x_i\}_{i=1}^{n}$;
- N_{iter}, N_{max}.

Output:

- Paramètres optimaux du PRSOM (W^*, Σ^*)

Initialisation:

- $t \leftarrow 0; \quad T \leftarrow T_{max}$;
- $W = \{w_1, ..., w_{N_{max}}\}$, aléatoirement initialisés;
- $\Sigma = \{\Sigma_1, ..., \Sigma_{N_{max}}\}$, aléatoirement initialisés avec des grandes valeures.

Etape 1:
Répéter
Affectation :

$$CHN = \begin{cases} \text{Implémmenter PRSOM sur CHN avec les Eqs.(58) et (59)} \\ \text{Estimer les paramètres avec les theorèmes 1.6, 1.7 et l'Eq.(73)} \end{cases}$$

Minimisation :
Tanque $j \prec N_{max}$ **faire**
Si $u_{1,j} == 1$ **alors**
mise à jour des w_j et Σ_j par les Eqs.(78) et (79)
Finsi

- $t \leftarrow t + 1; \quad j \leftarrow j + 1$
- $T \leftarrow T_{max} \times (\frac{T_{min}}{T_{max}})^{\frac{t}{N_{iter}-1}}$.

Fin tanque
Jusqu'à $(\Delta E(v, W, \Sigma) < \epsilon || iter > N_{iter})$

- $N_{op} = \sum_{j=1}^{N_{max}} u_{1,j}$;
- $k \leftarrow 1$;
  Pour j **allant de 1 à** N_{op}
  si $u_{1,j} == 1$ **alors**
- $w_k^{init} \leftarrow w_j$;
- $\Sigma_k^{init} \leftarrow \Sigma_j$;

- $k \leftarrow k + 1$.
  fin si
  fin pour

Etape 2 :

- $(W^*, \Sigma^*) \leftarrow \underbrace{Argmin}_{(W,\Sigma)} PRSOM(W^{init}, \Sigma^{init})$.

Retourner

- Paramètres optimaux du CHN-PRSOM (W^*, Σ^*).

0.6 Conclusion

Grâce à la solidité de ses bases théoriques et à la robustesse de ses algorithmes, les réseaux de neurones récurrents sont devenus l'alternative la plus privilégiée pour la résolution des problèmes combinatoires. Parmi ces réseaux, le modèle de Hopfield, proposé en 1982, a suscité l'intérêt de nombreux chercheurs. En effet, ce système permet, grâce à une analogie des réseaux de neurones avec la physique statique, de résoudre des problèmes d'optimisation en associant les variables du problème avec les variables de la fonction d'énergie du réseau. Cependant, l'inconvénient majeur de ce modèle est la non-faisabilité de la solution obtenue, ainsi que le choix non adéquat des paramètres du modèle. Pour surmonter ces difficultés, nous avons introduit un ensemble des conditions analytiques garantissant que chaque point d'équilibre caractérise une solution de problème. En outre, les paramètres du modèle ont été étudiés et discutés à travers l'analyse de stabilité du réseau de Hopfield.
Dans ce travail, nous avons traité deux grands problèmes dans lesquels nous avons apporté divers modélisations et améliorations. Le premier problème concerne la restauration des images, pendant que le deuxième est consacré aux choix de l'architecture de la carte auto-organisatrice probabiliste.

Dans le **chapitre 1**, nous avons présenté une synthèse des travaux réalisés dans les réseaux de neurone récurrents. Cette synthèse révèle toute la richesse et la diversité des travaux et des modèles réalisés dans cette discipline. Ayant commencé par une introduction sur les réseaux de neurones artificiels et leurs composantes, nous avons expliqué la théorie d'apprentissage statistique en définissant la vraisemblance et sa relation avec la programmation quadratique. Après, nous avons décrit les différentes architectures des RNAs, ainsi que leurs modes d'apprentissages. Enfin, nous avons terminé ce chapitre par la description du modèle de Hopfield.

Dans le **chapitre 2**, nous avons présenté le problème mathématique de restauration d'images. Premièrement, nous avons décrit les étapes de construction d'une image numérique. Ensuite, nous avons interprété les composantes nécessaires permettant de construire le modèle mathématique de restauration d'images, y compris les phénomènes du flou et du

bruit. Après, nous avons décrit la phase de modélisation dans laquelle nous avons commencé par décrire le modèle de dégradation qui a été exploité dans le cadre de former le problème inverse associé à la restauration d'images. Comme ce modèle est mal conditionné, nous avons par la suite expliqué quelques techniques proposées pour surmonter ce problème, en particulier, les moindres carrées, les méthodes itératives et la technique de régularisation qui a été adoptée tout au long de ce rapport. Deuxièmement, en se basant sur la régularisation de Thikonov, nous avons expliqué le modèle d'optimisation de restauration des images, ainsi que les résultats d'existence et d'unicité de la solution qui ont été développés sur les espaces des fonction à variations bornées $BV(\Omega)$. Troisièmement, la restauration des images par équations aux dérivées partielles a été décrite via l'équation de la chaleur et le modèle non linéaire de l'équation de Perona .

Dans le **chapitre 3**, nous avons employé le modèle continu de Hopfield pour résoudre le problème de restauration des images. Ce modèle a été choisi dans le cadre de la résolution du problème de fluctuation du réseau discret. Dans ce contexte, nous avons rappelé le fonctionnement du réseau discret de Hopfield pour la résolution du modèle de restauration. Puis, nous avons expliqué en détail les étapes d'adaptation du CHN au modèle IRP. En outre, nous avons enrichi la résolution par l'étude de la convergence et l'estimation des paramètres. Les résultats obtenus par le réseau CHN ont été satisfaisants numériquement et visuellement.

Dans le **chapitre 4**, nous avons proposé une nouvelle vision de restauration d'images appelée restauration sélective. Dans le modèle classique de restauration d'images, nous avons techniquement introduit une règle de décision. Cette dernière permet de choisir, à travers une variable de décision, entre le pixel original et un pixel de référence (par exemple, filtre médian), celui qui va générer un côut minimal. Le modèle ainsi obtenu est un problème d'optimisation non linéaire à variables mixtes dont la résolution via les méthodes exactes n'est pas réalisable, voire impossible. Pour cette raison, nous avons utilisé le réseau de Hopfield continu et l'algorithme génétique de façon progressive pour résoudre le modèle proposé. De plus, nous avons identifié les paramètres adéquats pour la résolution en se basant sur l'analyse de stabilité du CHN. Les simulations ont montré l'efficacité de la technique, ainsi que la faisabilité des paramètres estimés.

Dans le **chapitre 5**, le problème de choix de l'architecture de la carte auto-organisatrice probabiliste a été défini et étudié. Dans ce contexte, nous avons modélisé ce dernier en termes d'un problème d'optimisation à variables mixtes avec contraintes linéaires et non linéaires. Comme le modèle obtenu est complexe à variables mixtes, nous avons investi le formalisme Expectation-minimisation pour la phase de résolution. Dans ce cadre, le réseau de Hopfield continu a été employé pour la phase d'affectation optimisation, alors que la phase de minimisation a été résolue par le gradient de la vraisemblance. Enfin, nous avons validé la performance de l'architecture optimisée pour le fameux problème de clustering.

Introduction

Recurrent (Dynamical) Neural Networks (RNNs) are artificial neural networks models that are well-suited for pattern classification tasks where the inputs and the outputs are sequences. The importance of investing in techniques for mapping sequences to sequences is exemplified by tasks such as signal recognition, language modelling, image processing and machine translation. RNNs represent a sequence with a high-dimensional vector (called the hidden state) of a fixed dimensionality that integrates new observations using an intricate non-linear function. RNNs are highly expressive and can implement arbitrary memory-bounded computation, and as a result, they can likely be adapted to achieve non-trivial performance on difficult sequence tasks.
Until now, Hopfield neural networks (HNNs) are the most important RNNs models [**49**, **47**, **102**, **104**]. From its original definition as a model of associative memory[**119**], in which the information retrieval can be realized through the states evolution[**47**], Hopfield networks were soon applied to optimization field[**101**], which has become its main application. Despite plenty of theoretical studies and applications have been developed in HNNs, yet the study of this subject deserves a deeper attention. Firstly, because combinatorial optimization problems have a wide range of important applications in almost every scientific discipline. Besides, other problems in engineering and sciences can be reformulated as optimization problems. Secondly, Hopfield networks are very interesting from the point of view of dynamical systems, and their study can lead to advance our knowledge on questions of convergence and stability of more general systems. Finally, a detailed bibliographic analysis reveals a frequent erroneous usage of these networks, consisting in establishing spurious relations among the activation function, the dynamics and the Lyaponuv function. HNNs have been investigated for different kinds of optimization problems[**70**]. In fact, Since the first use of HNN in solving travelling salesman problem (TSP) [**49**, **88**], HNNs have been adopted for several kinds of optimization problems such as LP [**38**, **90**], NLP [**62**], MILP [**113**], and multi-objective linear programmings [**97**].
Another interesting RNN, which was proposed by Teuvo Kohonen, is self organizing maps (SOM)[**66**]. SOM models provide a data visualization technique that helps to understand high dimensional data by reducing its dimensions to a map. Moreover, self organizing maps represent a clustering concept by grouping similar data together. Therefore, it can be said that SOM reduces data dimensions and displays similarities among

data. A strong extension of SOM called Probabilistic SOM (PRSOM) has been proposed by Fouad and al. [**5**]. The suggested model gives a maximum approximation (likelihood) of the density distribution and a clustering of the data space. In PRSOM models, the density distribution is a Gaussian mixture [**5**] depending on parameters (weights and covariance matrix) estimated by a learning algorithm that uses the likelihood maximum. [**32**, **33**].
Hopfield neural networks have been properly used is image processing (IP), notably the image restoration. Indeed, this problem has been reformulated as the optimization of a penalized criterion. Before going any further, let us first give a general overview of image restoration.
Image Restoration Problem (IRP) is started since the 50s after many researches carried out to improve the quality of the images received in the space program [**61**]. Restoration techniques are applied to remove:

- System degradations such as the blur owing to optical system aberrations, atmospheric turbulence, motion and diffraction.
- Electronic noise which affects the measurement.

Several methods have been proposed to solve IRP and they have proved to a great extent their effectiveness in image restoration while preserving as possible the image details. Filtering method is one of the efficient approaches that have been used widely to enhance the image quality. In fact, in addition to the known Gaussian filter, these techniques include also the Median filter [**7**, **58**], Wiener filter [**81**], inverse and kalaman filters [**117**]. However, Median filter allows a good noise suppression at the expense of features deterioration. Wiener filter is implemented only if the wide sens stationary assumption has been made for images. Inverse filter generates a good restoration when the image is not affected by noise. In noisy case, it produces bad effects. Kalaman filter can deal with non stationary images, but it is computationally very intensive.
Recently, a novel image processing technology based on variational methods is motivating many mathematicians and image scholars to research on this challenging topic. This technique protects the image details by solving the mathematical IRP along the boundary variation space, in which the discontinuities are preserved (2). Among the variational image restoration models, the Total Variational (TV) model proposed by Rudin, Osher and Fatemi [**92**] has been the most important. It has been properly applied in several applications[**9**]. Other extensions of TV technique have been suggested to restore degraded images as proposed in [**118**]. Image restoration problem has been also solved by Partial Differential Equations (PDE)[**9**]. Motivated by the isotropic and anisotropic properties described by the Heat and Perona equations, several researchers have investigated these PDEs for image restorations tasks [**64**, **85**]. However, these equations of second order are very weak since they tend to cause the processed image to exhibit staircase effects. To avoid this drawback, recent studies have been proposed based on Four order PDE (FOPDE) and they have proved their performance thanks to the high smoothing ability [**123**]. Optimization filed has also contributed to image restoration using the modelling techniques and resolution. For example, we may cite some techniques that are inspired by the biological behaviours like genetic algorithms [**57**, **94**] and artificial neural networks [**124**], [**82**, **108**].

In this thesis, the continuous Hopfield neural network is proposed to deal with new optimization models of image restoration and architecture choice of PRSOM. Indeed, since the first appear of the optimization model of image restoration, several approaches have been arisen to solve it. Discrete Hopfield neural network has been also exploited to solve this issue, and it has succeeded in that task. However, the dynamic of the DHN is still a limited model because of the fluctuation behaviour which comes from the use of the hard limit activation function. In this context, we relax the HNN by using the continuous Hopfield network. The use of CHN allows, from one hand to broaden the space solutions thanks to the function *tanh*. From the other hand, the possibility to settle the parameters of the energy function based on the stability analysis.
While there is nothing intrinsically wrong with image restoration model, some features are deteriorated because of the unwanted process which may create phenomenons like isotropic phenomena. To avoid this drawback, we propose to develop the classical model of IRP. In this regard, we introduce decision variables that permit to select carefully areas concerned by the restoration. This modification creates a mixed-integer non linear optimization model (MI-SIR), which means that resolution by exact methods is not an easy task. For this purpose, genetic algorithm and CHN are used iteratively to solve the proposed model, trying every iteration to preserve original parts. Resolution phase is supported by the stability analysis, in which the model parameters are settled so that the equilibrium point is achieved.
The convergence and parameters estimation of the energy function associated with CHN have been established for the two models of image restoration. Now, the stability analysis of CHN is studied in a deep level. In fact, the CHN is adopted next to deal with a new optimization model of probabilistic self organizing map. The proposed model allows, thanks to the quadratic programming, to solve the architecture choice problem of PRSOM. The built model is a mixed-variables problem with linear and non linear constraints, which means that resolution task by CHN needs further studies, especially regarding the constraints.
Contributions are organized in the following layout:

Chapter 1- Recurrent artificial neural networks: This chapter presents the fundamental concepts of artificial neural networks, notably, the recurrent neural networks. Firstly, a brief history of ANNs is given as well as the principal components that form an artificial neural network. Next, the theory of learning is explained from a mathematical vision, especially, the strong principle of likelihood function and its relationship with the quadratic minimisation. Thereafter, architectures of ANNs and theirs learnings methods are described and illustrated by figures. At the end of this chapter, Hopfield recurrent neural networks are carefully explained through theirs architectures, their dynamics, as well as their modes of convergence.

Chapter 2- Mathematical vision of image restoration problem: In this chapter, we present the image restoration problem from the mathematical vision[**9**]. In the first section, we describe the concept of a digital image (DI), which is the fundamental unit on which we work, by explaining the principal components and the mathematical operations needed for the process of digitization [**89**, **96**]. Next, we explain in details the necessary steps for

building the image restoration model, as well its theoretical study including the existence and the uniqueness. We end this chapter by reviewing the use of partial differential equation to solve image restoration, in particular through Heat and Perona equations.

Chapter 3- Convergence and parameters estimation of continuous Hopfield neural network applied to optimization model of image restoration: In this chapter, the continuous Hopfield neural network is proposed to replace the Discrete HNN in solving the mathematical image restoration problem. The resolution phase is not limited to resolution, but also supported by the convergence and parameter estimation extracted from the stability analysis of the CHN.

Chapter 4- New mixed-integer model for selective image restoration: modelling and resolution by CHN and genetic algorithm: In this chapter, a quadratic programming based technique is proposed to deal with the issue of details preservation during the restoration process. Based on the classical model of image restoration, a modified model is constructed by introducing a set of binary variables that indicate the pixels categories. Each pixel is combined with the median of its neighbours in a decision rule so that one of them generates the optimal solution. Since obtained model is a non-linear mixed-integer problem, so the resolution by exact methods is very difficult, if not impossible. In this regard, the continuous Hopfield neural network (CHN) and genetic algorithm (GA) are progressively used to solve the suggested model. During resolution phase, suitable parameters are extracted from the stability criteria of CHN.

Chapter 5- Using Continuous Hopfield neural network for solving a new architecture optimization model of the recurrent probabilistic self organizing map: In the present chapter, the architecture choice of probabilistic SOM is discussed. Based on the classical model of PRSOM, a new improved model is suggested and explained. The built model is mixed-variables non-linear optimization model under linear and quadratic constraints. While resolution by exact methods is not easy task, both of continuous Hopfield network and gradient method are used iteratively to achieve the solution. Resolution phase is supported by analytical studies of stability and parameters estimation.

Chapter 7- Conclusions and Future work : This chapter provides the concluding remarks with more emphasis on achievements and limitations of the proposed schemes. The scopes for further research are outlined at the end. The contributions made in each chapter are discussed in sequel, which include proposed models, their simulation results, and the comparative analysis.

Chapter 1

Reccurent artificial neural networks

1.1 Introduction

Artificial neural networks (ANNs) is a field of Artificial Intelligence (AI) where we, by inspiration from the human brain, find data structures and algorithms for learning and classification of data [**34**, **35**] . Many tasks that humans perform naturally fast, such as the recognition of a familiar face, proves to be a very complicated task for a computer when conventional programming methods are used[**42**, **112**].
By applying neural networks techniques, a program can learn by examples, and create an internal structure of rules to classify different inputs, such as images recognition. ANNs consist of two categories. Feed-forward neural networks (FFNNs) have been the first and the simplest type. In these networks, the information moves only from the input layer directly through a set of hidden layers, thus generating an observed information stored on the output layer without cycles. Feedforward networks can be constructed with various types of units, such as binary or sigmoidal McCulloch-Pitts neurons [**74**]. The models the most known of feedforward ANNs are the multilayer perceptrons and the Radial Basis Function (RBF)[**45**, **121**]. The other kind of ANNs, named by recurrent neural networks (RNNs), are dynamic models that contain at least one feed-back connection, so the activations can flow round in a loop. In the literature, there are several examples of recurrent networks including Self Organizing Maps (SOMs) [**66**], Cellular neural networks (CNNs) [**22**] and notably the Hopfield neural networks (HNNs) [**48**, **49**, **122**].
The rest of this chapter is organized as follows: Section 1.2 presents the fundamental concepts of artificial neural networks. In that section, a brief history of ANNs is given as well as the principle components that form an artificial neural network. In Section 1.3, the theory of learning is explained from a basic vision. Architectures of ANNs and theirs learnings methods are given in Section 1.4. Section 1.5 describes the Hopfield model as one of the most known recurrent neural network. Finally, a brief conclusion is given in Section 1.6.

1.2 Artificial neural networks

1.2.1 History of artificial neural networks

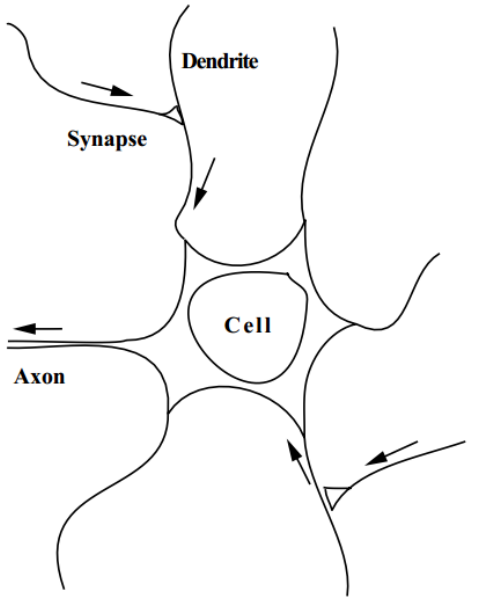
Artificial neural networks have been studied extensively over three periods. The first appear in the 1940 was due to McCullotch and Pitts who proposed a simple parametric non linear computational model of real neuron [**74**]. The second development occurred in the 1960s with Rosenblatts who proposed a layered neural network consisting of perceptrons and a suitable algorithm for adjusting the parameters of a single layer for the purpose of implementing a desired task [**91**]. In [**76**], Minsky and Paperts have shown the limitations of a simple perceptron. As consequence, the resulting lull in neural networks research lasted almost 20 years. Since the earlier 1980s, ANNs have received a considerable renewed interest. The major developments behind this resurgence include Hopfield neural networks [**47**] and the back-propagation algorithm for multilayer perceptrons which is proposed first by Webros [**115**], reinvented several times, and then popularized by Remulhart and al in 1986 [**93**].

1.2.2 From biological neural networks to artificial neural networks

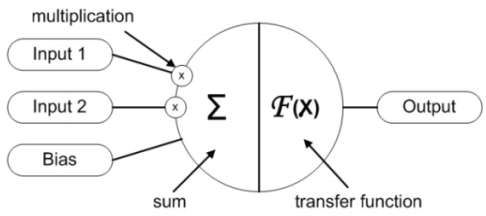
Artificial neural network is an attempt at modelling the information processing capabilities of nervous systems. In this context, we need first to consider the essential properties of biological neural networks from the viewpoint of information processing. A neuron (or nerve cell) is a special biological cell that processes information (Figure 1.1a). It is composed of a cell body or soma and two types of out-reaching tree like branches: the axon and the dendrites:

- **Cell body (Soma):** the body of neuron cell contains the nucleus and carries out biochemical transformation necessary to the life of neurons.
- **Dendrite:** are fibres that emanate from the cell body and provide the receptive zones that receive activation from other neurons. They branch out into tree around the cell body. They accept incoming signals.
- **Axon:** it is a long, thin and tubular structure which works like a transmission line.
- **Synapse:** are junctions that allow signal transmission between the axons and dendrites.

Artificial neuron is a basic building block of every artificial neural network. Its design and functionalities are derived from observation of a biological neuron that is a basic building block of biological neural networks (systems) which include the brain, spinal cord and peripheral ganglia. The similarities in design and functionalities can be seen in Figure 1.1 so that the sub-Figure 1.1a represents a biological neuron with its soma, dendrites and axon while the Figure 1.1b shows an artificial neuron with its inputs, weights, transfer function, bias and outputs.
In Figure 1.1b, various inputs to the network are represented by the mathematical symbol x_n. Each of these inputs is multiplied by a connection weight represented by w_n. In the



(a)



(b)

Figure 1.1: Description of a simple neuron: (a) Biological (b) Artificial

simplest case, these products are summed, fed to a transfer function (activation function) to generate a result, and this result is sent as output. This is also possible with other network structures, which utilize different summing functions as well as different transfer functions. Seven major components make up an artificial neuron:

- **Weighting Factors:** A neuron usually receives many simultaneous inputs. Each input has its own relative weight, which gives the input the impact that it needs on the processing element's summation function. Some inputs are made more important than others to have a greater effect on the processing element as they combine to produce a neural response. Weights are adaptive coefficients that determine the intensity of the input signal as registered by the artificial neuron. They are a measure of an input's connection strength. These strengths can be modified in response to various training sets and according to the network topology or its learning rules.
- **Summation Function**: If the inputs and the corresponding weights are given by the two vectors $(x_1, x_2.., x_n)$ and $(w_1, w_2.., w_n)$, then the total input signal is the dot product of these two vectors. The obtained result; $(x_1 * w_1) + (x_2 * w_2) + .. + (x_n * w_n)$; is a scalar. The summation function can be more complex than just weight sum of products. The input and weighting coefficients can be combined in many

different ways before passing on to the transfer function. In addition to summing, the summation function can select the minimum, maximum, majority, product or several normalizing algorithms. The specific algorithm for combining neural inputs is determined by the chosen network architecture and paradigm. Some summation functions have an additional activation function applied to the result before it is passed on to the transfer function for the purpose of allowing the summation output to vary with respect to time.

- **Transfer Function:** The result of the summation function is transformed to a working output through an algorithmic process known as the transfer function. In the transfer function, the summation can be compared with some threshold to determine the neural output. If the sum is greater than the threshold value, the processing element generates a signal and if it is less than the threshold, no signal (or some inhibitory signal) is generated. Both types of response are significant. The threshold, or transfer function, is generally non-linear. Linear functions are limited because the output is simply proportional to the input.
  The step type of transfer function would output zero and one, one and minus one, or other numeric combinations. Another type, the *threshold* function, can mirror the input within a given range and still act as a step function outside that range. It is a linear function that is clipped to minimum and maximum values, making it non-linear. Another option is a *S* curve, which approaches a minimum and maximum values at the asymptotes. It is called a sigmoid when it ranges between 0 and 1, and a hyperbolic tangent when it ranges between -1 and 1.
- **Scaling and Limiting:** After the transfer function, the result can pass through additional processes, which scale and limit. This scaling simply multiplies a scale factor times the transfer value and then adds an offset. Limiting is the mechanism which insures that the scaled result does not exceed an upper, or lower bound. This limiting is in addition to the hard limits that original transfer functions may have performed.
- **Output Function (Competition):** Each processing element allows the generation of one output signal, subsequently, these outputs are given to hundreds of other neurons. Normally, the output is directly equivalent to the transfer function's result. Some network topologies modify the transfer result to incorporate competition among the neighbouring processing elements. Neurons are allowed to compete with each other inhibiting processing elements unless they have great strength. Competition can occur at one or both levels. First, the competition determines which artificial neuron will be active or provides an output. Second, competitive inputs help to determine which processing element will participate in the learning or adaptation process.

- **Error Function and Back-Propagated Value:** In most learning techniques, the difference between the current output and the desired output is calculated as an error which is then transformed by the error function to match a particular network architecture. Most basic architectures use this error directly but some square the error while retaining its sign, some cube the error, other paradigms modify the error to

fit their specific purposes. The error is propagated backwards to a previous layer. This back-propagated value can be either the error, the error scaled in some manner (often by the derivative of the transfer function) or some other desired output depending on the network type. Normally, this back-propagated value, after being scaled by the learning function, is multiplied against each of the incoming connection weights to modify them before the next learning cycle.

- **Learning Function:** Its purpose is to modify the weights on the inputs of each processing element according to some neural based algorithm. The theory of learning is the subject of the next section. Through this document, the learning function is denoted by $\boldsymbol{l}$.

1.3 Learning theory

The learning phase is an important task in ANNs. This step is performed after the choice of models. Generally, the learning phase is divided into two categories:

- Supervised learning.
- Unsupervised learning.

1.3.1 Supervised learning.

Statistically, the supervised learning consists in estimating a prior and conditional probabilities denoted consecutively by $P(X \in C_i)$ and $P(X|C_i)$. Thus, the choice of the probability density reduces considerably the problem difficulty. In this context, if this probability is represented by a Gaussian density, then the learning problem is transformed from estimating the probability $P(X \in C_i)$ to estimate the Gaussian parameters μ_i and Σ_k.
The classical method used in the parametric approach is the maximum likelihood. In this technique, the parameters are supposed fixed but unknown. The optimal parameters are those that maximize the observation probability over the sample that we dispose.

1.3.1.1 Likelihood maximum

Consider the learning set $D = \{D^1, ..., D^m\}$ which consists of observations from m classes denoted by $\{C_1, .., C_m\}$. The Observations $\{D^k \in C_k\}_{k=1,..,m}$ are assumed to be independent according to the law $P(X \in C_i)$, which is supposed to be known and determined by the parameters vector w^k. We consider also that the parameter vectors are independent for each $i, j \in \{1, .., m\}^2$ such that ($w^i \neq$w^j).
If $D_N = \{x^1, .., x^N\}$ are the observations of a class C_i, we say that the plausibility of these observations is captured by the likelihood function $L(w) = L(w|x^1, .., x^N)$ which, through the independence of observations, can be expressed by:

$$L(w) = L(w|x^1, .., x^N) = P(D_N|w) = \prod_{n=1}^{N} P(x^n|w) \tag{1.1}$$

Practically, we use $log(L(w))$ instead of $L(w)$ which represent the same thing since the logarithm is continuous and strictly monotone. Moreover, this choice is very special when the density is exponential. Thus, we have:

$$log(L(w)) = log(P(D_N|w)) = \sum_{n=1}^{N} log(P(x^n|w)) \tag{1.2}$$

If $w = (w^1, .., w^m)^t$ and $\nabla_w = \{\frac{\partial}{\partial w_1}, \frac{\partial}{\partial w_2}, .., \frac{\partial}{\partial w_m}\}^t$, then the estimation of the maximum of the likelihood is obtained by solving the following system:

$$\nabla_w log(L(w)) = \sum_{n=1}^{N} \nabla_w log(P(x^n|w)) = 0 \tag{1.3}$$

1.3.1.2 Case of Gaussian distribution

Now, let's suppose that the observations that constitute the set $D^k_{N_k} = \{x^k_1, x^k_2, .., x^k_{N_k}\}$ of N_k examples are distributed according to the Gaussian law with the mean μ_k and the covariance matrix Σ_k. In **[4]**, μ_k and Σ_k are estimated by:

$$\widehat{\mu_k} = \bar{x}^k = \frac{1}{N_k} \sum_{n=1}^{N_k} x^k_n \tag{1.4}$$

and

$$\widehat{\Sigma_k} = S_k = \frac{1}{N_k} \sum_{n=1}^{N_k} (x^k_n - \bar{x}^k)(x^k_n - \bar{x}^k)^t \tag{1.5}$$

1.3.1.3 Rigid estimation of Σ

If the number of the examples in the sample is less than the dimension of the data space, then the estimation of Σ_k using Eq.(1.5) provides a matrix S_k which is singular ($rank(S_k) < N_k$). Therefore, several numerical problems arises.
To overcome this drawback, Friedman, Aivazian and others have proposed to estimate Σ by $S_k + \lambda I$ where I is the identity matrix and λ is a small positive constant **[37]**. The objective of such technique is augmenting the diagonal values of the matrix S_k, thus allowing S_k to be inverse. This kind of estimation is called "rigid estimation".

1.3.1.4 Formulation of learning problem

The formulation of the Learning theory is a statistical formulation which deals perfectly with the classification problem. First of all, let's suppose the following notations:

- A set of random vectors $x \in IR^p$ with an unknown fixed distribution $P(x)$.
- A set of responses where each response d is assigned to x according to an unknown conditional density denoted by $P(d|x)$.

- A learning machine $\mathcal{F}_{\mathbb{A},\mathbb{W}}$ of architecture $\mathbb{A}$ and a parameter space $\mathbb{W}$, this machine is responsible for generating the function $g(x,w)$ where $w \in \mathbb{W}$.
- The dissimilarity between the response d and the value $g(x,w)$ is measured by the local cost function $l(d,g(x,w))$.

The couples (x,d) build a joint space "input-output" with a joint density defined by $p(x,d) = p(x)p(d|x)$. Our goal is to estimate the unknown dependence between x and d thanks to the learning machine $\mathcal{F}_{\mathbb{A},\mathbb{W}}$.
The main purpose of the learning is to optimize the following risk:

$$Q(w) = E_x\left[l(d,g(x,w))\right] = \int l(d,g(x,w))P(x,d)d\mathrm{d}dx \tag{1.6}$$

where $E_x\left[.\right]$ is the mathematical expectation that expresses the theoretical risk associated with the problem.
For a fixed architecture $\mathbb{A}$, the learning problem consists in minimizing the risk (1.6) over the parametric set $\mathbb{W}$ with a limited number of examples coming from the unknown density $p(x,d)$. For this reason, we aim to approximate the conditional density $P(d|x)$ using the learning machine $\mathcal{F}_{\mathbb{A},\mathbb{W}}$. We denote by $g(x,w^*)$ each function verifying:

$$w^* = \underset{w\in\mathbb{W}}{argmin} Q(w) \tag{1.7}$$

1.3.1.5 Learning from examples

Generally, the main problem of the learning is the absence of an explicit expression of the joint density $P(x,d)$, which means that all information that we have is situated in a sample of size N:

$$D_N = \{(x^1,d^1),(x^2,d^2),...,(x^N,d^N)\} \tag{1.8}$$

where each couple (x^n,d^n) is supposed to be identically and independently drawn from population according to the joint density $P(x,d) = P(x)P(d|x)$. The set D_N is called the *learning set*.
To deal with the theoretical risk (1.6), the most common approach named *Empiric risk minimization* is based on the inductive principle given by:

- First, we use the examples D_N to replace the theoretical risk (1.6) by the empiric error defined by:

$$C(w) = \frac{1}{N}\sum_{n=1}^{N} l(d^n,g(x^n,w)) = \frac{1}{N}\sum_{n=1}^{N} C^n(w) \tag{1.9}$$

- Secondly, we replace w^* which minimizes the theoretical risk by $\widehat{w}$:

$$\widehat{w} = \underset{w\in\mathbb{W}}{argmin} C(w) \tag{1.10}$$

So, if the learning machine approximates the parameters space using these steps, then we say that the inductive principle defines a learning process.
The value of $Q(w)$ at the point $\widehat{w}$ is called the error in generalization and it is expressed by:

$$Q(\widehat{w}) = \int l(d^n, g(x^n, \widehat{w}))P(x, d)d\mathrm{d}dx \tag{1.11}$$

$Q(\widehat{w})$ depends not only on the model $\mathcal{F}_{\mathbb{A},\mathbb{W}}$, but also on the learning set D_N. The model quality is measured by the average error in generalization, which is the expectation of the error in generalization through the sample D_N:

$$\bar{Q} = E_D\left[Q(\widehat{w})\right] \tag{1.12}$$

1.3.1.6 Cost function

Obviously, looking for parameters based on the examples involves the optimization of a specific cost function generally defined by the user. Usually, the cost function takes a quadratic form that measures the distance between the output d^k and the output generated by the classifier. If the outputs of the network are interpreted by a probabilistic distribution, then we may use the likelihood or another measure like the relative entropy. In the next paragraph, we describe an interesting result that concerns the relationship between the likelihood maximum and the quadratic error minimization.

1.3.1.7 From likelihood maximum to quadratic minimum

As shown from Eq.(1.1), the Likelihood maximum associated with the set D_N is given by:

$$L(W) = P(D_N|W) = \prod_{n=1}^{N} P(x^n, d^n|W) = \prod_{n=1}^{N} P(d^n|x^n, W)P(x^n) \tag{1.13}$$

One of the important results in the learning theory is the equivalence of maximizing the likelihood and minimizing $-\ln(L(W))$. Indeed:

$$-\ln(L(W)) = -\sum_{n=1}^{N} \ln(P(d^n|x^n, W)) - \sum_{n=1}^{N} \ln(P(x^n)) \tag{1.14}$$

Since the term $P(x^n)$ does not depend on the parameters W, then, we can ignore it, thus constituting the new cost function:

$$G(W) = -\sum_{n=1}^{N} \ln(P(d^n|x^n, W)) \tag{1.15}$$

hypothetically, we suppose that the desired outputs are distributed according to the Gaussian law $N(0, \sigma^2 I)$, where I is the identity matrix. The probability distribution of the desired output variable is expressed by:

$$P(d^n|x^n, W) = \frac{1}{\sqrt{2\pi}\sigma} exp\{-\frac{1}{\sqrt{2\sigma^2}}(g(x^n, W) - d^n)^2\} \tag{1.16}$$

By substituting in Eq.(1.15), we have the following cost function :

$$G(W) = \frac{1}{\sqrt{2\sigma^2}} \sum_{n=1}^{N} \sum_{k=1}^{m} (g_k(x^n, W) - d_k^n)^2 + Nm\sqrt{2\pi}\ln(\sigma) \tag{1.17}$$

We neglect the second term in Eq.(1.17) and the term $\frac{1}{\sigma^2}$ which is a constant, we obtain as criteria the following quadratic error:

$$G(W) = C(W) = \frac{1}{2} \sum_{n=1}^{N} \sum_{k=1}^{m} (g_k(x^n, W) - d^n)^2 = \frac{1}{2} \sum_{n=1}^{N} ||g(x^n, W) - d^n||^2 \tag{1.18}$$

In summary, we deduce that under the conditions that the desired values are independent and distributed to the Gaussian law, then the minimization of squared error is equivalent to the maximization of the Likelihood function.

1.3.2 Unsupervised learning

The other type is the unsupervised training (learning). In this type, the network is provided with inputs but not with desired outputs. In this case, this system itself is responsible for choosing suitable features that it will use to group the input data. This is often referred to "self-organization or adaptation". These networks use no external influences to adjust their weights. Instead, they improve their performances based on theirs own behaviours. These networks look for regularities or trends in the input signals, and makes adaptations according to the function of the network. Even without being told whether it is right or wrong, the network still must have some informations about how to organize itself. These informations are created into the network topology and learning rules. An unsupervised learning algorithm might emphasize cooperation among clusters of processing elements. In such a scheme, the clusters would work together. If some external input activated any node in the cluster, the cluster's activity as a whole could be increased. Likewise, if external input to nodes in the cluster was decreased, that could have an inhibitory effect on the entire cluster. Competition between processing elements could also form a basis for learning. Training of competitive clusters could amplify the responses of specific groups to specific stimuli. As such, it would associate those groups with each other and with a specific appropriate response. Normally, when competition for learning is in effect, only the weights belonging to the winning processing element will be updated. Presently, the unsupervised learning is not well understood and there continues to be a lot of research in this aspect

Unsupervised learning problems can be further grouped into clustering and association problems.

- **Clustering**: A clustering problem is where you want to discover the inherent groupings in the data, such as grouping customers by purchasing behavior.
- **Association**: An association rule learning problem is where you want to discover rules that describe large portions of your data, such as people that buy X also tend to buy Y.

1.4 Artificial neural networks architectures

A single node is insufficient for the practical problems, and networks with a large number of nodes are frequently used. The way in which nodes are connected determines how computations proceed and constitutes an important early design decision by neural network developer. An artificial neural network is essentially a data processing system comprised of a large number of simple highly interconnected units. Various neural networks architectures are found in the literature. Here we define different networks which are commonly used in current literature [**84**].

1.4.1 Feed-Forward neural networks

Feed forward neural networks are the simplest form of artificial neural networks. The feed forward neural network was the first and arguably simplest type of artificial network devised. In this network, the information moves in only one direction, forward, from the input nodes, through the hidden nodes (if any) and to the output nodes as shown in Figure 1.2. There are no cycles or loops in the network. In a feedforward system, the processing elements (PE) are arranged into distinct layers with each layer receiving input from the previous layer and outputting to the next layer. Weights of direct feedback paths, from a neuron to itself are zero. Weights from a neuron to a neuron in a previous layer are also zero. Weights for the forward paths may also be zero depending on the specific network architecture, but they do not need to be in every case. Mathematically, If L_i and L_j denote respectively the i^{th} and j^{th} layers, then the weights are expressed by:

$$\begin{cases} w_{ij} = 0 \text{ if i>j} \\ w_{ij} \geq 0 \text{ otherwise} \end{cases} \tag{1.19}$$

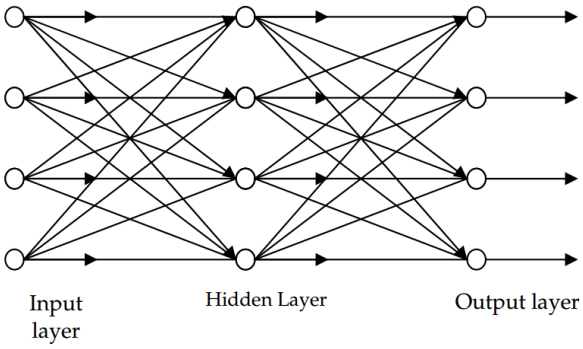


Figure 1.2: Example of a feed forward neural network

1.4.1.1 Radial Basis Function neural networks

- **Architecture:**
  Radial basis function neural networks are FFNNs structures comprising the input, hidden and output layers (Figure 1.3) [**121**]. Neurons of the hidden layer represent

Gaussian basis functions while the output neurons are described by a linear activation function. Let W^k_{RBF} be the weights between the input nodes and the k^{th} RBF node ($W^k_{RBF} = X - W^k$); so the output of the k^{th} RBF node is

$$h^k_{RBF} = exp(-\frac{||W^k_{RBF}||^2}{\sigma^2_k}) \tag{1.20}$$

where σ_k is the spread of k^{th} RBF function, $X = (x_1, x_2, .., x_{m_I})$ is the input vector,

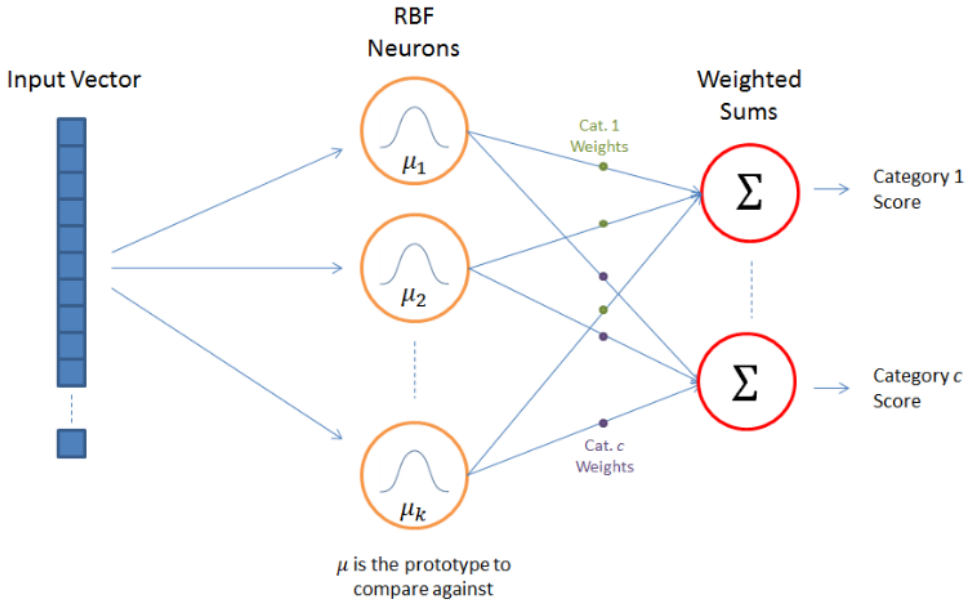


Figure 1.3: Example of Radial basis function neural network

$W = (w_{1k}, w_{2k}, .., w_{mk})$ and $||W^k_{RBF}|| = \sum_{i=1}^{m}(x_i - w_{ik})^2$. Each node of the output layer generates the following quantity:

$$\Phi^j_{RBF} = \sum_{k=1}^{m_H} w_{kj} h^k_{RBF} \tag{1.21}$$

- **Training algorithm**
  Training algorithm for RBF starts with one RBF node using one of the data points as the centre of the Gaussian functions, then it finds the data points with the highest error, which is used as the centre of a new RBF node. Squared errors are minimized by adjusting the connection weights between the hidden layer and the output layer. The process is continued till the error goal in terms of square of error is achieved as the number of RBF nodes attains a given maximum value.

1.4.1.2 Multilayer Perceptron neural networks

Multilayer Perceptron neural network (MLP) is the most common and popular type of neural networks in use today [**45**]. It belongs to a class of neural networks called feed forward neural networks. MLPs represent a generic function approximator and classifier. It

is an efficient tool to approximate generic classes of functions, including continuous and integrable functions [**95**]. Moreover, it can distinguish data that are not linearly separable. Thanks to these properties, the MLP neural network has been widely used in modelling and optimization problems. This famous system was first introduced by M. Minsky and S. Papert in 1969 [**76**].

- **Architecture of multilayer Perceptron**
  MLPs are feed forward networks with one or more layers (hidden layers) of units (hidden neurons) between the input and output layers. The neurons of the previous layer are always connected with those of the next one while the neurons in the same layer are not interconnecte (Figure 1.4). The number of hidden layers as well

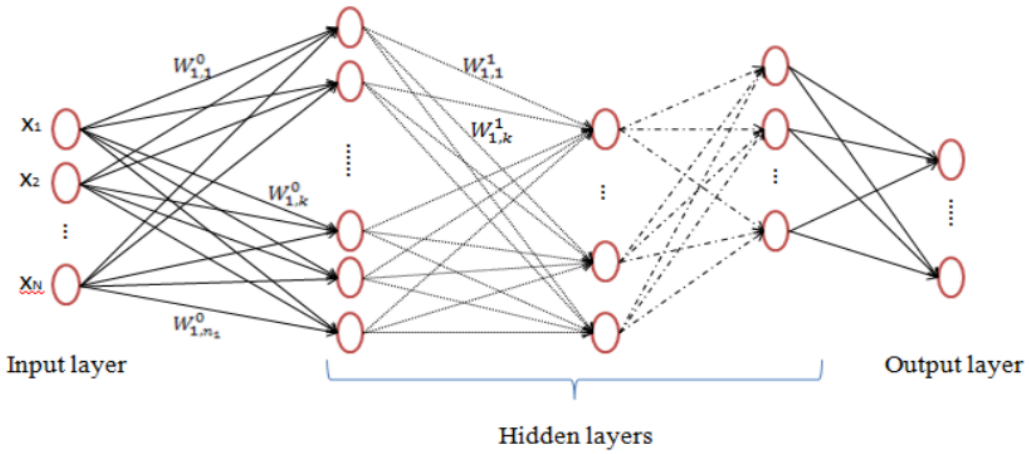


Figure 1.4: Multilayer Perceptron neural network with three hidden layers

as theirs neurons are generated randomly. Many efforts [**6**, **19**, **87**] are employed to look for the optimal parameters of MLPs layouts. The number of neurons in the input layer equals the dimensions of observations while the number of neurons in the output layer is fixed a priori according to the purpose of the problem at hand. The input layer is reserved to present the features to the fist hidden layer of the perceptron model. Hidden layers propagate the information from the inputs to the outputs (outputs of each hidden layer) along the network. The neurons of the output layer represent a hyper plane in the space of the input patterns [**45**].

- **Learning in MLP**
  There are various methods that are proposed for training the MLP, but the most common and successful one is the Back-propagation [**68**, **120**]. This method was first proposed by Werbos [**115**] in 1974. Later, Rumelhart, Hinton and Williams exploited the back-propagation in their work in simulating cognitive process. Since then back-propagation has been employed in a number of fields for solving problems that would be quite difficult using conventional computer science techniques.
  The main steps of this algorithm are summarized in the following algorithm:

Back propagation Learning algorithm

1. Initialize the weights to small random values.
2. Randomly choose an input x^i.
3. Propagate the signal forwardly through the network.
4. Compute δ_i^L in the output layer as follows:

$$\delta_i^L = f'(S_i^L)(d_i - y_{(}i))$$

   where S_i^L is the network input to the i^{th} unit in the L^{th} layer, and f' is the derivative of the activation function f.
5. Compute the deltas of the preceding layers by propagating the errors backwards;

$$\delta_i^k = f'(S_i^k) \sum_j w_{ij}^{k+1} \delta_j^{k+1} \quad \forall k = L-1, .., 1$$

6. Update weights using

$$\Delta w_{ji}^k = \eta \delta_i^k y_j^{k-1}$$

7. Go to step to and repeat for the next pattern until the error in the output layer is below a pre-specified threshold or a maximum number of iterations is reached.

So far we have looked at networks with supervised learning techniques, in which there is a desirable output for each input pattern, and the network needs a parameter settings to produce the required outputs. In the next paragraph, we describe the other kind of ANNs that takes account of the feed-back behaviour.

1.4.2 Recurrent neural networks

A feed forward neural network having one or more hidden layers with at least one feed-back loop is known as recurrent network (Figure 1.5). The feedback may be a self feed-back, i.e., where the output of each neuron is fed back to its own input. Sometimes, feed-back loops involve the use of unit delay elements, which results in non-linear dynamic behaviour, assuming that neural network contains non linear units. There are various types of recurrent networks. The most known are described below:

1.4.2.1 Kohonen's self-organizing maps

The Self-Organizing Map (SOM) **[60, 66]** has the desirable property of topology preservation, which captures an important aspect of the feature maps in the cortex of highly developed animal brains. *SOM* helps thanks to its topological carte to better understanding of the high dimensional data by reducing the dimensions of data to a map. In addition, *SOM* deals properly with the clustering issue owing to its ability to display similarities among data **[65]**. Basically, SOM consists of two dimensional array of units, each connected to all n input nodes. w_{ij} denotes the n-dimensional vector associated with the unit

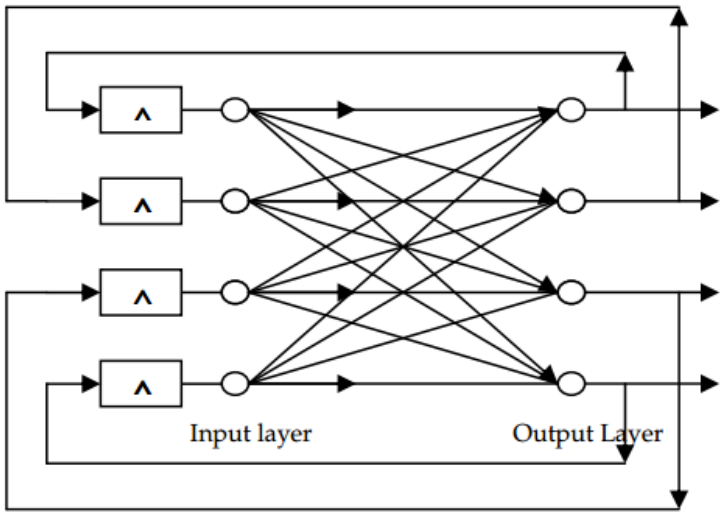


Figure 1.5: Scheme of recurrent neural network

at location (i, j) of the 2-D array. Each neuron computes the euclidean distance between the input vector x and the stored weight vector w_{ij} (see Figure 1.6). The learning phase associated with SOM is based on the competitive learning. It is given as follows:

SOM Learning algorithm

1. Initialize weights to small random numbers; initial learning rate and neighbourhood.
2. Present a pattern x and evaluate the network outputs.
3. Select the unit (C_i, C_j) with the minimum output:

$$x - w_{C_i,C_j} = \min_{ij} ||x - w_{ij}||$$

4. Update all weights according to the following learning rule:

$$w_{ij}(t+1) = \begin{cases} w_{ij}(t) + \alpha(t)\left[x(t) - w_{ij}(t)\right] & \text{if}(i,j) \in N_{C_i,C_j}(t) \\ w_{ij}^t \text{ otherwise} \end{cases} \quad (1.22)$$

   where $N_{C_i,C_j}(t)$ is the neighborhood of the unit (C_i, C_j) at time t, and $\alpha(t)$ is the learning rate.
5. Decrease the value of $\alpha(t)$ and shrink the neighbourhood $N_{C_i,C_j}(t)$.
6. Repeat steps 2 through 5 until the change in weight values is less than pre specified threshold or a maximum number of iterations is reaches.

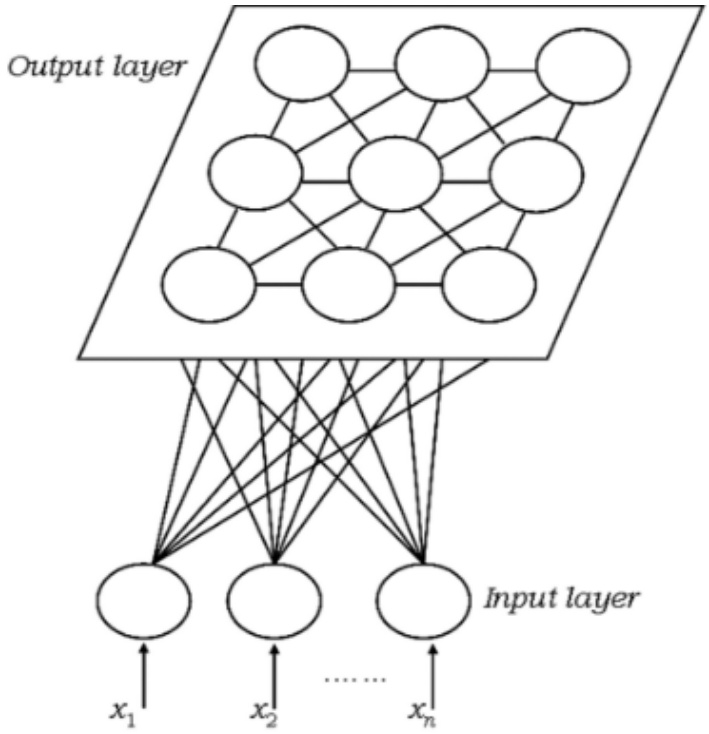


Figure 1.6: Self Organizing Map

1.4.2.2 Cellular Neural Networks

A cellular neural network is an artificial neural network which features a multidimensional array of neurons and local interconnections among cells. The important feature of CNN paradigm is the use of the analogue cells with continuous signal values and local interaction within finite radius. Its architecture consists of regular spaced cloned circuits called cells. Each cell is connected to its neighbour cells and can interact directly with each other. CNN consists of a finite number of cells. Each cell has an input, a state, and an output, and it interacts directly only with the cells within its radius of neighborhood r, when r = 1, which is a common assumption, the neighborhood includes the cell itself and its eight nearest neighboring cells (Figure 1.7).

Each cell is composed of linear and non linear circuit elements, which typically are linear capacitors, linear resistors, linear and non linear controlled sources and independent sources as shown in Figure (1.8), the typical circuit of a single cell. In the Figure, Eu_{ij} is the independent voltage source, I is the independent current source, I_n^{yij}, I_n^{uij} is voltage controlled current sources and Ey_{ij} is the output voltage controlled source.

The cell has direct connections to its neighbors through two kinds of weights: the feedback weight and the control weight and the index pair represents the direction of signal from one cell to another cell. The global behaviour of CNN is characterized by a template set containing A-Template, B-Template, and the bias I. Cellular neural network has important potential applications in areas such as image processing and pattern recognition. It has the best features of both the words, its continuous time feature allows real time

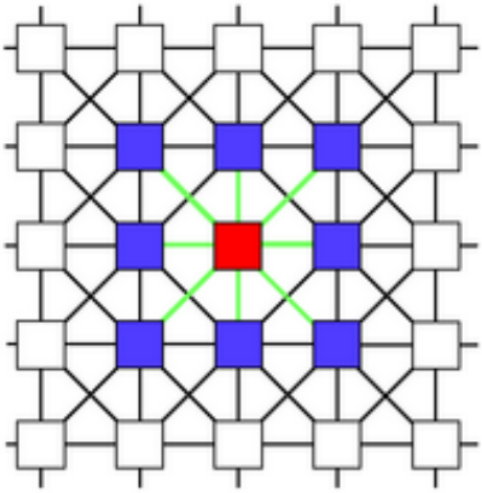


Figure 1.7: Two-dimensional CNN with neighbourhood radius $r = 1$: the red cell has nine neighbours (the eight blue cells and itself)

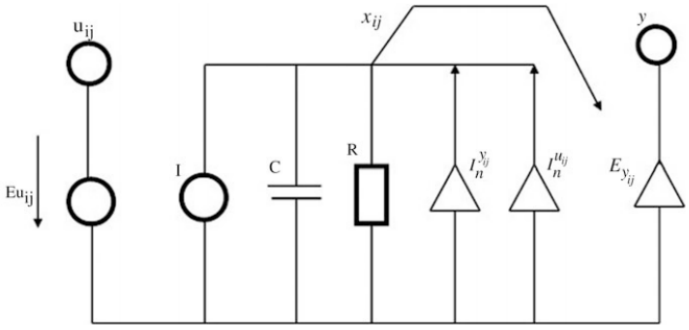


Figure 1.8: Components of one cell

signal processing important in the digital domain and its local interconnection feature makes it tailor made for VLSI implementation.

1.5 Hopfield recurrent neural networks

Hopfield neural network (HNN) was invented by Dr. John Hopfield in 1982 [**47**]. It consists of a single layer which contains one or more fully connected recurrent neurons. The Hopfield networks are commonly used for auto-association and optimization tasks [**16**, **48**, **49**, **50**, **98**]. Hopfield has proposed two basic models: discrete and continuous HNNs, the difference between the two models refers to the nature of the state variables and time in these networks. Next, we provide further details for the two models.

1.5.1 Discrete Hopfield networks

In the discrete HNN, the output of each neuron is calculated through the following formula:

$$X_i(t+1) = sign(\sum_{j=1}^{n} T_{ij} x_j(t)) \tag{1.23}$$

Thus, the vector X is given as:

$$X = sign(TX - I) \tag{1.24}$$

where X, W, T and the sign function are :

X is the activation vector of the n neurons: $X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$

T is the weight matrix :$T = \begin{bmatrix} T_{11} & T_{12} & T_{13} & \dots & T_{1n} \\ T_{21} & T_{22} & T_{23} & \dots & T_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ T_{d1} & w_{d2} & T_{d3} & \dots & T_{dn} \end{bmatrix}$ where T_{ij} is the connection between the i^{th} and j^{th} neurons.

I is the threshold vector: $I = \begin{bmatrix} I_1 \\ I_2 \\ \vdots \\ I_n \end{bmatrix}$

The sign function is defined as:

$$sign(x) = \{ \begin{array}{ll} +1 & \text{if} x > 0 \\ -1 & otherwise \end{array}$$

As consequence, the Hopfield neural network can be represented by fully interconnected network with undirected weights (Figure 1.9)

1.5.1.1 Update mode

There are at least two ways in which we might carry out the updating specified by the above equation:

(a) Synchronously: update all the units simultaneously at each time step.

(b) Asynchronously: update them one at a time. In this case we have two options:

- At each time step, select at random an unit i to be updated, and apply the formula (1.23);
- Let each unit independently choose to update itself according to the above formula, with some constant probability per unit time.

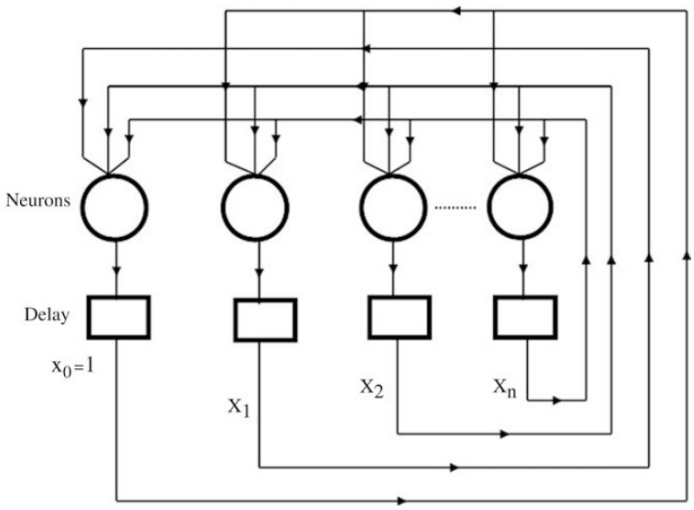


Figure 1.9: Hopfield neural network

1.5.1.2 Energie and convergence

Hopfield networks converge to a local state[**47**, **48**]. The energy function of a Hopfield network in a certain state is :

$$E = -\frac{1}{2}X^t TX - IX^t \tag{1.26}$$

Proposition 1.1

A Hopfield network with n units, symmetric and zero-diagonal weights matrix and asynchronous dynamics, which starts from any given network state, eventually reaches a stable state at local minimum of the energy function

Proof. As the dynamic is asynchronous, only one unit is evaluated at each time. If the k^{th} unit is selected, there is two options:

- k does not change state.
- k changes of state (-1 $\rightarrow$ 1 or 1 $\rightarrow$ -1).

In the first case, the energy function remains the same, $E(t+1) = E(t)$. Else, the energy function changes, according to the new excitation value (x'_k)of the k^{th} unit. The difference of energy of the system is given by:

$$E(x) - E(x') = -(\sum_{j=1}^{n} x_j w_{kj} x_k - \sum_{k=1}^{n} I_k x_k) - (-(\sum_{j=1}^{n} x'_j w_{kj} x'_k - \sum_{k=1}^{n} I_k x'_k))$$

Since $w_{kk} = 0$, then:

$$E(x) - E(x') = -(x_k - x'_k) * (\sum_{j=1}^{n} w_{kj} x_j - I_k) \tag{1.27}$$

The second term of this equation ($\sum_{j=1}^{n} w_{kj}x_j - I_k$) is the excitation e_k of the k^{th} unit. The unit changed its state, so the excitation has a different sign than x_k and x_k according to (1.23). Therefore, the Eq.(1.27) is positive, which implies that $E(x) > E(x')$ So the energy will decrease for every change, and since there is only a finite number of possible states, the network should reach a state where the energy cannot decrease more, it will be a stable state. Of course the stable state is not unique, but it is certain that the system will converge. The final state will depend on the input and the initial state of the system. □

Remark *In the discrete HNN, there is not only one choice for the activation function. Indeed, several works have used the Heavside function as a neurons activation and this choice still agrees with the previous proposition.*

1.5.2 Continuous Hopfield neural networks

In, 1982, Hopfield and Tank have proposed a neural network called *Continuous Hopfield neural network* for the purpose of solving a wide variety of combinatorial problems and realizing a self-associative memories [**49**, **50**]

1.5.2.1 Dynamic and activation function in continuous Hopfield neural networks

The Continuous Hopfield neural network represents a strong extension of the DHNN that is governed by the differential equation[**41**]:

$$\frac{du}{dt} = -\frac{u}{\tau} + Tv + I \tag{1.28}$$

where u, v and I are, respectively, vectors of neuron states, outputs and biases. τ is a time constant and the output function $v_i = g(u_i)$ is an hyperbolic tangent defined by:

$$g(u_i) = \frac{1}{2}(1 + tanh(\frac{u_i}{u_0})) \quad u^0 > 0, \ \forall i = 1, .., n \tag{1.29}$$

where u^0 denotes an input vector and the matrix T represents the weights of the synaptic connections between neurons.

1.5.2.2 Equilibrium point

The evolution of the Hopfield neural network is characterized by a differential equation of first order. In this part, we describe this equation, notably, the existence of solution and convergence approximation[**103**].

1.5.2.3 Lyaponuv function

Generally, the stability of a differential system is profoundly based on Lyaponuv function [**24**]. Moreover, the existence of an equilibrium point is closely liked to the existence of an energy function

Definition 1

If a vector v^e satisfies $v(t) = v^e,\ \forall t \geq t^e$ for some $t^e \geq 0$, then, v^e is called an equilibrium point of the system (1.28).

Definition 2

Let S be a states space of the Hopfield neural network. A function E is said an energy function over S, if it is defined, bounded and decreased over each a continuous non-increased path in S.

Theorem 1.1: Lyapunov

If for a differential system, an energy function decreased with times t such that: $\frac{d(E(u(t)))}{dt} = 0$ for $\frac{d(u(t))}{dt} = 0_{IR^n}$,then this system is stable; which means that $u(t)$ converge toward a stable point.

Solving combinatorial problem via CHN requires the building of an energy function associated with the problem. Indeed, the Theorem 1.1 provides only a sufficient condition for the existence of the stability point.
To deal with this issue, Hopfield has introduced the energy function E on $[0,1]^n$, which is defined by:

$$E(v) = -\frac{1}{2}v^t T v - (I)^t v + \frac{1}{\tau}\sum_{i=1}^{n}\int_0^{v_i} g^{-1}(v)dv \tag{1.30}$$

when $\tau \to \infty$, the Lyaponuv function is associated with the cost function to be minimized in the combinatorial problem. Thus, CHN will solve problems with the following form:

$$E(v) = -\frac{1}{2}v^t T v - (I)^t v \tag{1.31}$$

Theorem 1.2

If T is a symmetric matrix, then the function (1.30) defines an energy function associated with the system (1.28) under conditions of Theorem (1.1).

If $H = \{v \in [0,1]^n\}$ denotes an Hypercube of IR^n, then, a point $v \in H$ is an equilibrium point for the CHN if and only if, the three following conditions are satisfied **[103]**:

$$E_i(v) = \frac{\partial E(v)}{\partial v_i} \geq 0 \quad \forall i = 1,...,n \quad such \quad that \quad v_i = 0 \tag{1.32}$$

$$E_i(v) = \frac{\partial E(v)}{\partial v_i} \leq 0 \quad \forall i = 1,...,n \quad such \quad that \quad v_i = 1 \tag{1.33}$$

$$E_i(v) = \frac{\partial E(v)}{\partial v_i} = 0 \quad \forall i = 1,...,n \quad such \quad that \quad v_i \in]0,1[\tag{1.34}$$

Following the notation used in **[2]**, the conditions (1.32) and (1.33) are dominated by:

$$\underline{E}^{0}(v) \geq 0 \tag{1.35}$$

$$\overline{E}^{1}(v) \leq 0 \tag{1.36}$$

where $\underline{E}^{0}(v) = \min_{\{i/v_i=0\}} E_i(v)$ and $\overline{E}^{1}(v) = \max_{\{i/v_i=1\}} E_i(v)$.
It should be noted that if the energy function (or Lyaponuv function) exists, the equilibrium point exists as well. Hopfield **[47]** proved that the symmetry of the weights matrix with zero-diagonal is a sufficient condition for the existence of Lyaponuv function.
The extrema of the Eq.(1.31) are the corners of the hypercube $[0,1]^n$. Thus, the outputs of the CHN represent a solution of (1.31). It must be mention too that it is very difficult to solve the differential equation that characterises the dynamic of the CHN. In fact, from one hand, this equation has proven to be very sensitive according to the initial conditions, and it suffers from the high time complexity on the other hand.
CHN solves the combinatorial problems as follow:
Let's consider the following combinatorial problem:

$$(P) = \begin{cases} \min\limits_{v\in\{0,1\}^N} -\frac{1}{2}v^tTv - (I)^tv \\ \text{subject to} \\ Av = b \\ v_i \in \{0,1\} \qquad\qquad i = 1,..,n \end{cases} \tag{1.37}$$

where:

- $H = [0,1]^n$ is the Hamming hypercube.
- $H_C = \{0,1\}^n$ denotes the Humming hypercube corners.
- $H_F = \{v \in H_C / Av = b\}$ is the Feasible solutions set.

First, we associate with the problem (1.37) a suitable energy function. Next, the problem is transformed from the system (1.37) to the minimization of the following function:

$$E(v) = E^c(v) + E^p(v) \qquad \forall v \in H \tag{1.38}$$

where $E^c(v)$ is proportional to the objective function and $E^p(v)$ is quadratic function that ensures having feasible solutions and penalizing the violated constraints.

1.6 Conclusion

Artificial neural networks play an important role in applications of sciences and engineering. In this chapter, the first section has described the ANNs through three important points. In the first point, a brief history of ANNs has been presented. The second point has shown the biological interpretation of ANNs, notably the description of biological and artificial neurons, as well as theirs comparisons. The principle components that make

an artificial neural network have been defined and explained in the last point.
In the second section, the learning theory has been explored and developed. In that section, we began by the likelihood function as one of the most interesting stochastic concept that have been used in the literature of machine learning. Next, we have described the learning from examples, in particular when the samples are distributed according the Gaussian law. This section has been ended by a fundamental result concerning the coherence between the maximum likelihood and the quadratic cost minimization.
The different architectures of ANNs have been briefly shown in Section 4. In the first part, the feed forward neural network has been defined and supported by its two common models represented by Radial basis function and Multilayer perceptrons. The second type called recurrent neural networks is described and illustrated by some graphical representations. This class of ANNs is composed of three major examples: self organizing maps and cellular neural networks which have been introduced in this section, and Hopfield neural network that has been the subject of the last section.
Hopfield recurrent neural networks, which is among the main contributions of this thesis, have been provided in this section. First, we have explained the discrete model of HNNs through its architecture, its update mode and the associated energy. Next, we have described the continuous HNNs, as well as theirs principles, their equilibrium points and the associated Lyaponuv function.

Chapter 2

Mathematical vision of image restoration problem

2.1 Introduction

Because of the imperfections in the image formation process and the imaging devices, the observed image often represents the degraded version of the original image. The corrections of these imperfections are mandatory in many of the subsequent image processing and vision tasks.
To retrieve the meaningful information from the degraded images, image restoration techniques have been used. It is not a surprise to see that digital image restoration is used in astronomical imaging even today. Ground based imaging systems were also subject to blurring due to change in refractive index of the atmosphere [**12**]. Image restoration also plays an important role in medical imaging. It has been used to remove film-grain noise in X-ray images, angiography images and additive noise in magnetic resonance images [**23**, **100**]. It has applications to quantitative auto radiography (QAR), in which the image is obtained by exposing X-ray film to a radio active specimen. Though image restoration has been successfully applied, but still has a scope for improvement in quality and resolution.
In this chapter, we present the image restoration problem from the mathematical vision[**9**]. In the first section, we describe the concept of a digital image (DI), which is the fundamental unit on which we work, by explaining the principal components and the mathematical operations needed for the process of digitization [**89**, **96**]. In fact, such operations as sampling and quantization allow to convert the signal from its analogue representation to a digital one using the mathematical convolution[**52**]. With regard to the components of DI, two disruptive behaviours that may appear, the first one is the noise, which is often treated as side effects of digitization process, it is usually modelled by a probability density distributed according to the Gaussian law [**89**]. The second effect, which is the blur, is generally defined as a convolution between the original image and the point spread function [**1**], the blur matrix can take several forms according to the boundary conditions used in the problem [**43**].
In the next part, we explain in details the necessary steps for building the image restoration model. First, we provide the common scheme of degradation on which we generate

the mathematical inverse problem that needs to be minimized [**63**, **73**]. Unfortunately, such model is typically an ill-posed problem [**54**, **86**], so we recall the different approaches that have been used to deal with this issue. For example, the least squares and the iterative methods [**79**]. Thereafter, we use the regularization technique, which interests us in such work, to build the regularized mathematical problem, which is reinforced by some analytical results of existence and uniqueness of the solution [**9**]. Next, the use of partial differential equation to solve the mathematical image restoration is described through the Heat and Perona models [**64**, **85**]. This chapter is ended by a brief conclusion.

2.2 Digital image

In our world, any visual scene is represented by a continuous function (bi-dimensional) of an analogue quantity. Such representation named "image" models the reflectance function of the scene, i.e, the light reflected at each visible point in the scene. A continuous analogue image cannot be interpreted by machine, in this regard, a special process called "digitization" must be used to build an alternative form (digital image). Specifically, the digitization includes the quantization of the intensity function value and the sampling of the two spatial dimensions [**39**, **89**, **96**].
The sampling rate determines the spatial resolution of the digitized image, while the quantization level determines the number of grey levels in the digitized image. A magnitude of the sampled image is expressed as a digital value in image processing. The transition between continuous values of the image function and its digital equivalent is called quantization.
The number of quantization levels should be high enough for human perception of fine shading details in the image. The occurrence of false contours is the main problem in image which has been quantized with insufficient brightness levels. In summary, digital image acquisition is essentially concerned with the generation of two dimensional array of integer values representing the reflectance function of the actual scene at discrete spatial intervals, and this is accomplished by the processes of sampling and quantization.

2.2.1 Convolution

In image processing, all measures devices begin by the process of "averaging" over each pixel neighbours before assigning this value to the pixel. This "averaging" does not depend on the pixel. Mathematically, this operation is known by *convolution* [**51**, **52**].

2.2.1.1 Continuous convolution

Let h and f are two functions in $L^1(IR^2)$, we denote by $h * f$ the convolution product of h by f is expressed as follows:

$$\forall (x,y) \in IR^2,\ \ h * f(x,y) = \int_{IR}\int_{IR} h(x-x', y-y') f(x',y') dx' dy'. \tag{2.1}$$

which is obviously a function in $L^1(IR^2)$.
The convolution product can be seen as the "averaging" of f over the neighbours of (x, y). For example, if $h(x, y) = \mathbb{1}_{[-a,a]}(x, y)$, then we have:

$$\forall (x,y) \in IR^2,\ \ h * f(x,y) = \int_{IR}\int_{IR} h(x-x', y-y') f(x',y') dx'dy' \tag{2.2}$$

$$= \int_{y-a}^{y+a}\int_{x-a}^{x+a} f(x',y')dx'dy' \tag{2.3}$$

The values of h can be changed according to the desired process. In fact, modifying the values of h is equivalent to give weights to the neighbours, and by consequence taking account of some neighbours more than the others.
It should be noted that the "averaging" process does not depend on h and it is the same $\forall (x, y) \in IR^2$, so we said that h is shift invariant **[14]**. Generally, h is an "averaging" operator if it satisfies the following condition:

$$\int\int_{IR} h(x,y)dxdy = 1 \tag{2.4}$$

Consequently, the convolution is the first degradation that attacks the analogical image.

2.2.1.2 Discrete convolution

Based on the continuous convolution (2.1), we provide the discrete extension of the continuous convolution for the case of finite sequences.
Let $(h_{m,n})_{1\leq m\leq N, 1\leq n\leq N}$ and $(f_{m,n})_{1\leq m\leq N, 1\leq n\leq N}$ two finite sequences. We denote by $h * f$ the following sequence:

$$(h*f)_{m,n} = \sum_{i=1}^{N}\sum_{j=1}^{N} h_{m-i,n-j} f_{i,j} \tag{2.5}$$

2.2.2 Sampling

A two-dimensional scene can be represented by a 2D function $f(x, y)$ of light intensity at the spatial location (x, y). However, in order for the continuous scene to be represented and processed digitally in a computer, it needs to be digitized. Correspondingly, the digital processing of the image can be classified into intensity (gray level) operations applied to the pixel values and geometric operations in the two spatial dimensions.
The continuous image function f(x,y) can be sampled using a discrete grid of sampling points in the plane. The image is sampled at points $x = j\Delta_x$, $y = k\Delta_y$ by the following expression:

$$F(x,y) = \sum_j \sum_k f(j\Delta_x, k\Delta_y)\delta(x - j\Delta_x, y - k\Delta_y) \tag{2.6}$$

The weights $f(j\Delta_x, k\Delta_y)$ associated with the δ functions are the pixels of the image. Dirac distributions in Eq.(2.6) can be regarded as periodic with period Δ_x, Δ_y and it can be expanded into a Fourier series **[53]**.

Shannon theorem for 2 dimensions gives the necessary condition to build the analogical image based on the sampled image [**7**].

Theorem 2.1: Shanon

An analogue image $f(x,y)$ limited to the spatial frequencies x_{max} and y_{max} can be reconstructed using the sampled data $F(x,y)$ if the sampling periods (Δ_x, Δ_y) verify:

$$\begin{cases} \Delta_x \leq \frac{1}{x_{max}} \\ \Delta_y \leq \frac{1}{y_{max}} \end{cases} \tag{2.7}$$

2.2.3 Quantization

In mage degitization, the magnitudes of the sampled image are expressed as digital values. For this reason, a special transition is needed to transform the continuous values of the image function (brightness) to digital equivalents. This process is called quantization [**30**]. The number of quantization levels should be high enough for human perception of fine shading details in the image. Most digital image processing devices use quantization into k equal intervals. As example, if b bits are used, then the number of brightness levels is $k = 2^b$. Eight bits per pixel are commonly used, specialized measuring devices use twelve and more bits per pixel [**30**].

2.2.4 Noise

In digitization phase, the measure devices generate always a noise. There can be many causes for noise (probabilistic character of the photons number, electronic imperfections of devices, sensors imperfections,...). In our work, we suppose only the additive noise denoted by η, which is a random sequence defined over the set $\{0,1,..,N\}^2$. Although rare, it is often reasonable to assume that the noise is white (values of η are independents). In addition, it can be represented by the *Gaussian* distribution where each value η_{ij} is a realisation of the continuous law of the following density:

$$p(t) = \frac{1}{\sqrt{2\pi}} exp(-\frac{t^2}{2\sigma^2}) \tag{2.8}$$

where $\sigma > 0$ is the parameter that expresses the distribution level of the noise. Thus, we obtain the following function:

$$h * f(m,n)\mathbb{1}_{[0,N]^2}(m,n) + \eta_{m,n} \tag{2.9}$$

The expected value of the noise η is given by:

$$\mathbb{E}(f(\eta)) = \int_{IR} f(t)p(t)dt \tag{2.10}$$

where $p(t)$ is the density of the law of η. The expected value represents the average of $f(\eta)$ for a finite number of independent realizations of the random variable η. In the case of the normal density (2.8), the expected value of $\eta$is given by:

$$\mathbb{E}(\eta) = \frac{1}{\sqrt{2\pi}} \int_{IR} t\, exp(-\frac{t^2}{2\sigma^2})$$

In addition, the variance of η is given by:

$$\begin{aligned} \mathbb{E}((\eta - \mathbb{E}(\eta))^2) &= \frac{1}{\sqrt{2\Pi}} \int_{IR} t^2\, exp(-\frac{t^2}{2\sigma^2}) - \mathbb{E}^2(\eta) && (2.11) \\ &= \sigma^2 && (2.12) \end{aligned}$$

2.2.5 Blur

It is well known that the image blur can be modelled as a convolution between the image and a specific function that describes the blurring of a point source. This function is called *point spread function* (PSF) [**1**]. Due to the linearity of convolution, we can express the observed image as:

$$y = h * x \tag{2.13}$$

where h is the blur operator, x is a variable that represents the object (usually as an unknown variable), y is the observed image,$*$ is the convolution product (2.1) (Figure 4.32).
A blurring is called spatially invariant (paragraph 2.2.1.1) if the PSF looks the same no

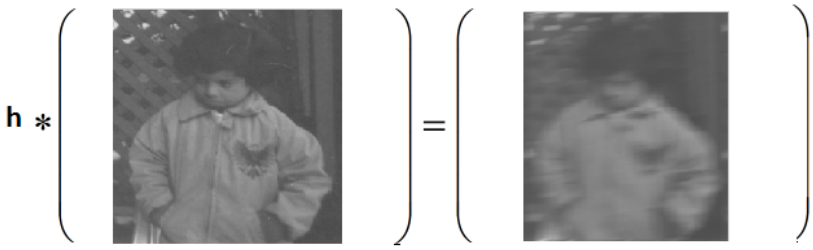


Figure 2.1: Example of a blurred image

matter where the point source is. Spatial invariance is assumed throughout this document.
Practically, the convolution product is transformed to the matrix product according to the nature of the boundary conditions that we assume near to the image edges [**43**]. If x is the gray level of a digital image of size $N \times N$, then we distinguish tree types of boundary conditions (see Figure 2.2):

- **Zero boundary conditions:** also known as the Dirichlet boundary conditions, in which we have:

$$x_{ij} = 0 \qquad \forall i \geq N \text{ and } \forall j \geq N \tag{2.14}$$

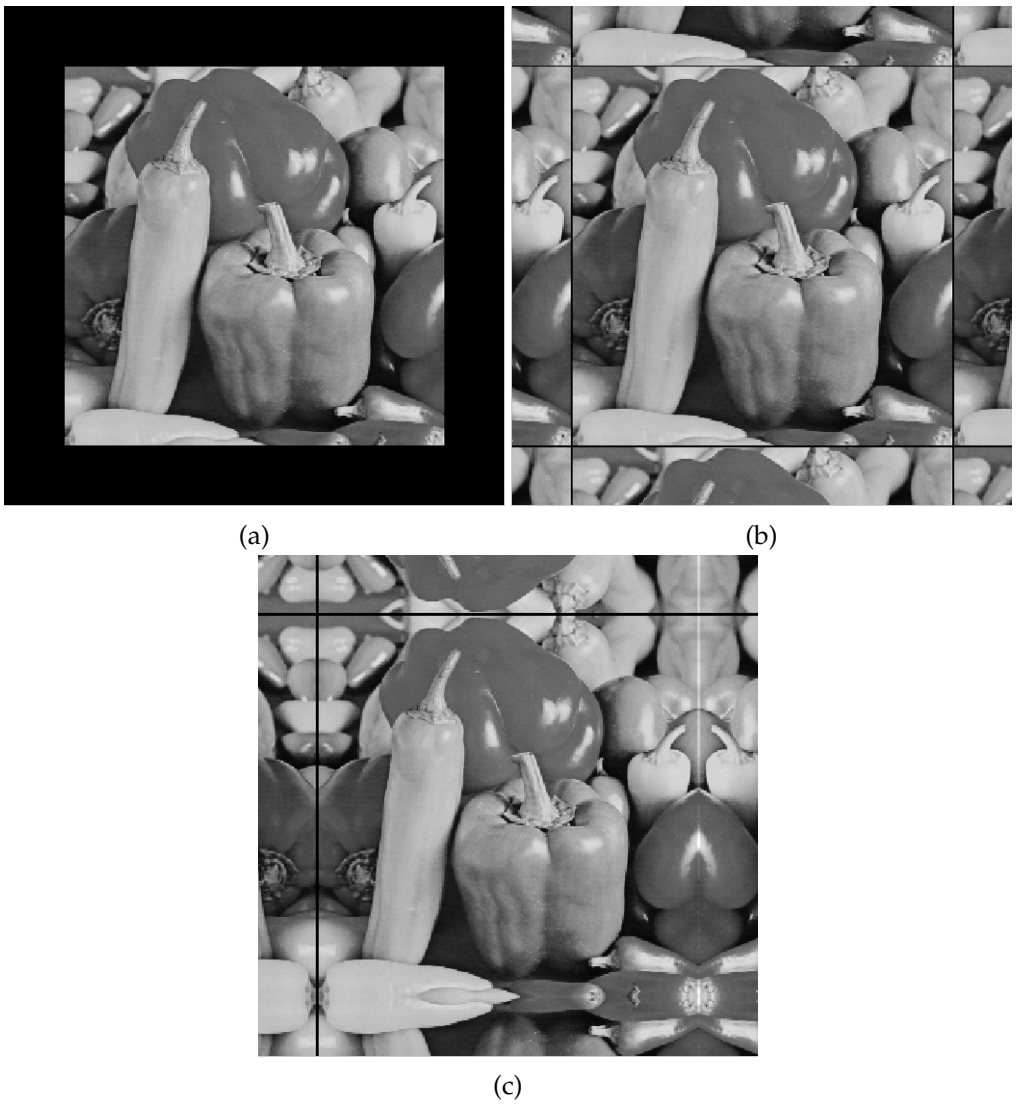
- **Periodic boundary conditions:**, in this kind of conditions, we suppose that:

$$x_{ij} = x_{i \bmod N, j \bmod N} \tag{2.15}$$

  Periodic end conditions rarely (if ever) hold exactly in practical imaging systems, but are a very useful approximation for analysing algorithms in the frequency-domain.

- **Reflexive boundary conditions:** also known as mirror-end-conditions or Neumann boundary conditions , in such case, we assume that

$$\begin{cases} x_{N-i,\bullet} = x_{N+i,\bullet} \\ x_{\bullet,N-i} = x_{\bullet,N+i} \end{cases} \tag{2.16}$$



(a) (b)

(c)

Figure 2.2: Boundary conditions: (a) dirichlet (b) Neumann (c) reflexive

2.2.5.1 Matrix representation of convolution

Numerically, the product defined by the Eq.(2.13) is represented by a matrix product, and this is performed through the following steps:

1. Rotate the PSF h by 180, i.e write its elements from bottom to top.
2. Place the center of the rotated PSF over the pixel x_{ij}.
3. Perform a component-wise multiplication and sum up to get the y_{ij}.

The transformation from convolution to the matrix product necessitates the following definitions:

Definition 3
A matrix with entries that are constant on each diagonal is called a Toeplitz matrix.

Definition 4
A Hankel matrix is a matrix with entries that are constant on each anti-diagonal.

Definition 5
A circulant matrix is a special Toeplitz matrix where each row is a periodic shift of its previous row.

Example: Consider the following example where a true scene x and a PSF h are given and we want to construct the blurred image y:

$$x=\begin{bmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \end{bmatrix}, h=\begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}, y=\begin{bmatrix} y_{11} & y_{12} & y_{13} \\ y_{21} & y_{22} & y_{23} \\ y_{31} & y_{32} & y_{33} \end{bmatrix}$$

For example, to compute y_{11}, we rotate the PSF and we place its center over x_{11}, next we perform a element-wise multiplication and sum up. Thus, we obtain:

$$\begin{bmatrix} \xi.h_{33} & \xi.h_{32} & \xi.h_{31} & \\ \xi.h_{23} & x_{11}.h_{22} & x_{12}.h_{21} & x_{13} \\ \xi.h_{13} & x_{21}.h_{12} & x_{22}.h_{11} & x_{23} \\ & x_{31} & x_{32} & x_{33} \end{bmatrix}$$

where ξ designates the boundary condition elements. As consequence, y_{11} is calculated as follow:

$$y_{11} = x_{11}.h_{22} + x_{12}.h_{21} + x_{21}.h_{12} + x_{22}.h_{11} + *.h_{33} + *.h_{32} + *.h_{31} + *.h_{23} + *.h_{13}$$

The outcome y is given as follows:

- If ξ are given by the Zero boundary condition (2.14), then the Eq.(2.13) is transformed to:

$$\begin{bmatrix} y_{11} \\ y_{21} \\ y_{31} \\ y_{12} \\ y_{22} \\ y_{32} \\ y_{13} \\ y_{23} \\ y_{33} \end{bmatrix} = \left(\begin{array}{ccc|ccc|ccc} h_{22} & h_{12} & & h_{21} & h_{11} & & & & \\ h_{32} & h_{22} & h_{12} & h_{31} & h_{21} & h_{11} & & & \\ & h_{32} & h_{22} & & h_{31} & h_{21} & & & \\ \hline h_{23} & h_{13} & & h_{22} & h_{12} & & h_{21} & h_{11} & \\ h_{33} & h_{23} & h_{13} & h_{32} & h_{22} & h_{12} & h_{31} & h_{21} & h_{11} \\ & h_{33} & h_{23} & & h_{32} & h_{22} & & h_{31} & h_{21} \\ \hline & & & h_{23} & h_{13} & & h_{22} & h_{12} & \\ & & & h_{33} & h_{23} & h_{13} & h_{32} & h_{22} & h_{12} \\ & & & & h_{33} & h_{23} & & h_{32} & h_{22} \end{array}\right) \begin{bmatrix} x_{11} \\ x_{21} \\ x_{31} \\ x_{12} \\ x_{22} \\ x_{32} \\ x_{13} \\ x_{23} \\ x_{33} \end{bmatrix}$$

The shown matrix is called "block Toeplitz with Toeplitz blocks" and it is denoted by ***BTTB***.

- In the case where ξ are represented by the periodic condition (2.15), the corresponding matrix is "block circulant with circulant blocks" (***BCCB***) where each circulant block is expressed as:

$$B_c = \begin{bmatrix} h_{22} & h_{12} & h_{32} \\ h_{32} & h_{22} & h_{12} \\ h_{12} & h_{32} & h_{22} \end{bmatrix}$$

- Regarding the reflexive boundary conditions, the matrix A is a sum of ***BTTB***, ***BTHB***, ***BHTB*** and ***BHHB*** matrices (e.g. ***BHTB*** = "block Hankel with Toeplitz blocks"-matrix).

2.3 Mathematical problem of image restoration

Since the main components of image restoration problem IRP (Noise and blur) were described, we aim in this section to explain the mathematical formulation of image restoration problem, as well as the different approaches proposed in the literature to solve it.

2.3.1 Image degradation

Restoration of high-quality image from degradation is an important pre-processing task in image processing. Restoration techniques are applied to remove:

- System degradations such as the blur created by an incorrect lens adjustment or by motion, atmospheric turbulence, and diffraction...
- Statistical degradations due to noise.

A commonly used model is the following:
Let $x : \Omega \subset \mathbb{R}^2 \to \mathbb{R}$ be an original image describing a real scene and let y be the observed image of the same scene i.e., a degraded version of x which , due to the linearity of convolution, can be express by(Figure 2.3):

$$y = h * x + \eta \tag{2.17}$$

where η stands for a white additive Gaussian noise and where h is a linear operator representing the blur (usually a convolution ()4.32)). Given y, the problem is then to reconstruct

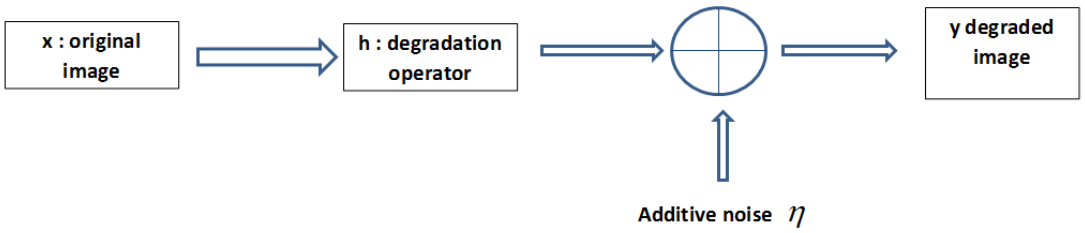


Figure 2.3: Process of degradation

x knowing (2.17). As we will see, the problem is ill-posed (Figure 2.4) , and we are able to carry out only an approximation of x.

2.3.2 Energy method for solving image restoration problem

As shown from the degradation model (2.17), the restoration process is a typical example of an inverse problem [**44**, **51**, **63**, **111**]. Let assume that the noise is a random vector that is distributed according to the law $\mathbb{N}(0, \sigma^2 I)$. Thus, the restoration problem can be seen as the maximization of the following likelihood function (paragraph 1.3.1.1) [**15**, **21**, **25**]:

$$\underset{x\in\Omega}{Max} P(y/x) \tag{2.18}$$

where

$$P(y/x) = P(y - hx) = \frac{1}{\sqrt{2\pi}\sigma} exp(-\frac{||y - hx||^2}{2\sigma^2}) \tag{2.19}$$

by simplifying, we obtain the minimization of the following term:

$$\inf_x \int_\Omega |y - hx|^2 dt \tag{2.20}$$

2.3.2.1 Resolution by least squares

Let suppose for a moment that: y and x are discrete variables in $\mathbb{R}^L$ such that $L = N^2$, h is an $L \times L$ matrix, and $||.t||$ stands for the Euclidean norm. The problem (2.20) is transformed to the following error function:

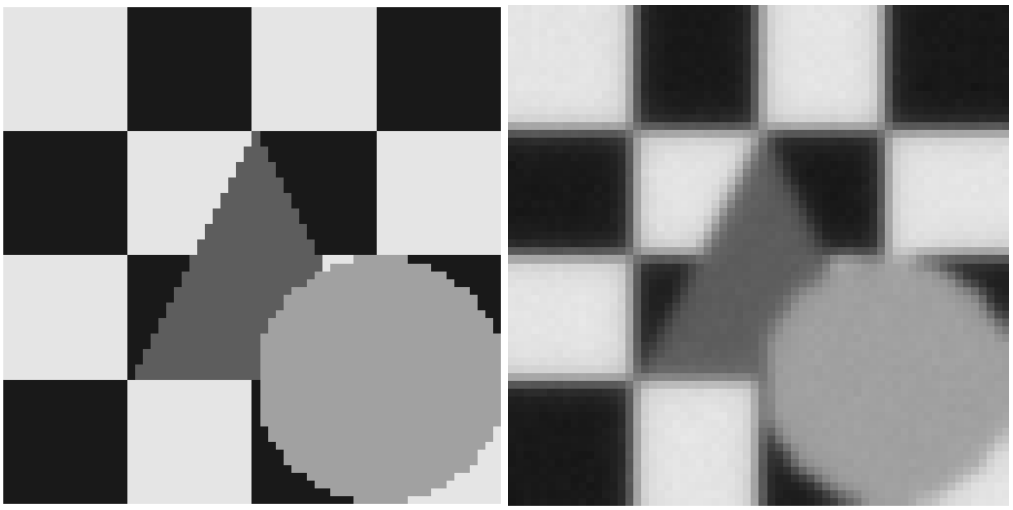
$$\underset{x\in\mathbb{R}^L}{Min} \|y - hx\|^2 \tag{2.21}$$

A necessary and sufficient condition to achieve the minimum of Eq.(2.21) is given by the following equation:

$$h^t hx - h^t y = 0 \tag{2.22}$$

where the solution is obviously given by:

$$x = (h^t h)^{-1} h^t y$$



(a) (b)

(c)

Figure 2.4: Restoration of a synthetic image degraded by only the blur: (A) Original synthetic image (B) degradation by blur (C) restoration using the inverse process

Solving (2.22) is in general an ill posed problem. Indeed, $h^t h$ is not always one-to-one, and even if $h^t h$ were one-to-one, its eigenvalues may be small, causing numerical instability.

2.3.2.2 Resolution by iterative methods

The problem (2.21) has been also solved by iterative methods where several studies have suggested [**14**, **79**]. All these works were based on the following function:

$$J(x) = \|y - hx\|^2$$

The gradient of J is defined by:

$$\nabla J(x) = h^t(hx - y)$$

Using the gradient descent, we have the following scheme:

$$x^{k+1} = x^k + \nabla J(x^k) \tag{2.23}$$

A special case of Eq.(2.23), which is also a one of the alternatives to regularize (2.20), is the **Landweber** (or Richardson iteration) algorithm where the corresponding scheme is formalized as [**17**, **14**]:

$$x^{k+1} = x^k - \alpha h^t(hx^k - y) \tag{2.24}$$

where the relaxation factor α satisfies $0 < \alpha < \frac{2}{\sigma_l^2}$ with σ is the largest singular value of h. Although it might be difficult to compute $\sigma_l^2 = ||h^t h||_2$, a specific choice for α can be found using the well-known bound [**109**]:

$$\sigma_l^2 = ||h^t h||_2 \leq ||h||_1 ||h||_\infty \tag{2.25}$$

In general, the matrix norms $||h||_1$ and $|h||_\infty$ are easy to compute. In fact, if h has no negative entries, as is the case in image restoration problems, and if we define O to be a vector with every entry equal to 1, then

$$||h||_\infty = \underset{j=1,..,N^2}{Max}\, hO(j)$$

and

$$||h||_1 = \underset{j=1,..,N^2}{Max}\, h^t O(j)$$

Thus, using these bounds , we get:

$$\alpha = \frac{1}{||h||_1 ||h||_\infty} \leq \frac{1}{\sigma_l^2} < \frac{2}{\sigma_l^2} \tag{2.26}$$

The analysis of such scheme, which has been described in [**14**], proves that the convergence is very slowly due to the large number of iterations needed to reconstruct the components that corresponds to the small singular values of h.
However, the method can be accelerated with preconditioning. It also provides a good introduction to discuss other iterative methods, including ones that enforce the non negativity constraint.
An algorithm for The steps of Richardson iterative method are stated in algorithm 2. As known, for any iterative regularization method, it is important to choose an appropriate criteria for terminating the while loop, which is a non trivial task. About this topic, we mention only an approach known as the Morozov's discrepancy principle [**77**], which terminates the iteration when

$$||y - hx||_2 \leq s\epsilon$$

where ϵ is the expected value of the noise η and $s > 1$. The discrepancy principle is easy to implement, but it does require a good estimate of ϵ. Other approaches are described in [**111**].

Data: h,y
Output: x
$x \leftarrow x^0$
$\alpha \leftarrow \alpha^0$
Compute
$r = y - hx$
$d = h^t r$
while *Stop critera not achieved* **do**
    $x = x + \alpha^0 d$
    $r = y - hx$
    $d = h^t r$
end

Algorithm 3: Algorithm of Landweber

2.3.2.3 Resolution by regularisation

In the problem (2.21), the matrix h is severely ill-conditioned, with singular values decaying to, and clustering at 0. This means that regularization is needed to avoid computing solutions that are corrupted by noise **[31, 44]**. Regularization can be enforced through well-known techniques such as *Wiener* filtering **[44]** and Tikhonov regularization **[40]**, and/or by incorporating constraints such as non-negativity **[13]**.
In this work, we opt for the known regularization of Tikhonov and Arsenin, which have suggested the following linear regularization:

$$J(x) = \int_\Omega |y - hx|^2 dt + \lambda \int_\Omega |\nabla x|^2 dt \tag{2.27}$$

The term $\int_\Omega |y - hx|^2 dt$ measures the fidelity to the data while the second is responsible for the smoothing phenomena.
In the study of this problem, the functional space for which both terms are well-defined is

$$W^{1,2}(\Omega) = \{x \in L^2(\Omega), \nabla x \in L^2(\Omega)^2\}$$

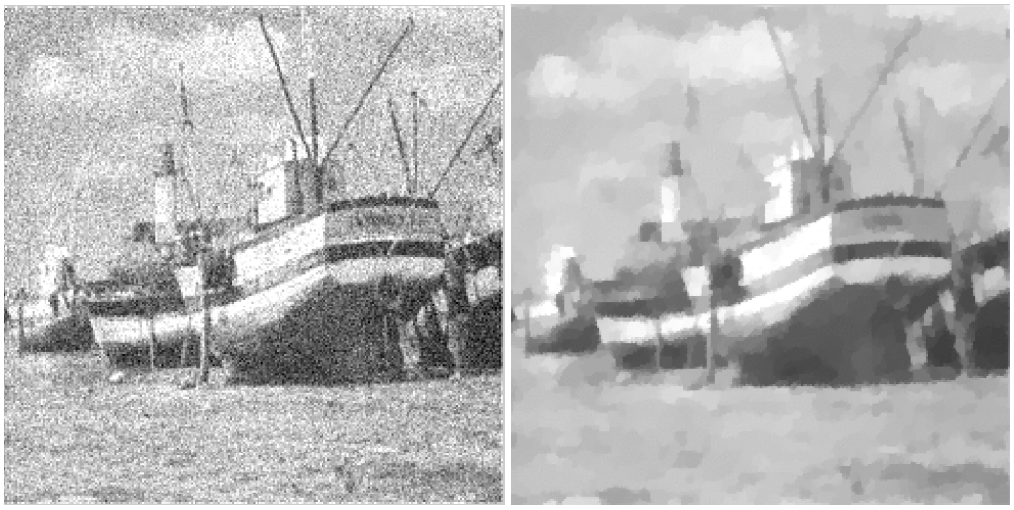
The problem $\inf_{x \in W^{1,2}} J(x)$ admits an unique solution characterized by the Euler Lagrange equation

$$h^t hx - h^t y - \lambda \Delta x = 0$$

with the neumann boundary condition

$$\frac{\partial x}{\partial N} = 0 \qquad \text{on} \quad \partial\Omega$$

In Eq.(2.27), the Laplacian operator has a very strong isotropic smoothing properties, which do not preserve the edge (Figure 2.5).
The suppression of the details comes from the fact that the L^2 norm of the gradient which



(a) (b)

Figure 2.5: Restoration of the "Boat" image degraded by an additive Gaussian noise of $\sigma = 0.3$ and $h = I_d$: (A) Original noisy image (B) Restored image by minimization of (2.27)

reduces perfectly the noise but unfortunately penalizes too much the gradients corresponding to edges.
To deal with this issue, some of the first works that have been widely investigated is proposed by Rudin, Osher and Fatemi **[92]**. In this work, the authors have used the L^1 norm of the gradient of x which is also called **Total variation**. To explain in detail the effect of the smoothness in the energy model defined by (2.27), we assume the following non linear regularization model **[10]**:

$$J(x) = \int_\Omega |y - hx|^2 dt + \lambda \int_\Omega g(|\nabla x|) dt \tag{2.28}$$

where g is a specific function that verifies certain assumptions (see the next paragraph). The Euler Lagrange equation associated with (2.28) is:

$$h^t hx - h^t y - \lambda div(\frac{g(|\nabla x|)}{|\nabla x|}\nabla x) = 0 \tag{2.29}$$

For each $|\nabla x| \neq 0$, there exists $N(x) = \frac{\nabla x}{|\nabla x|}$ and $|T(x)| = 1$ such that $< T(x), N(x) >= 0$ where $< ., . >$ denotes the scalar product. By expanding the div term and using T and N, Eq.(2.29) can be rewritten as:

$$h^t hx - h^t y - \lambda(\frac{g'(|\nabla x|)}{|\nabla x|} x_{\vec{T}\vec{T}} + g''(|\nabla x|) x_{\vec{N}\vec{N}}) = 0 \tag{2.30}$$

where we denote by $x_{\vec{T}\vec{T}}$ and $x_{\vec{N}\vec{N}}$ the second derivatives of x in the T-direction and N-direction:

$$x_{\vec{T}\vec{T}} = T^t \nabla^2 x T = \frac{1}{|\nabla x|^2}((x^{\vec{i}})^2 x^{\vec{j}\vec{j}} + (x^{\vec{j}})^2 x^{\vec{i}\vec{i}} - 2x^{\vec{i}} x^{\vec{j}} x^{\vec{i}\vec{j}})$$

and

$$x_{\overrightarrow{NN}} = N^t \nabla^2 x N = \frac{1}{|\nabla x|^2}((x^{\vec{i}})^2 x^{\vec{i}\,\vec{i}} + (x^{\vec{j}})^2 x^{\vec{j}\,\vec{j}} + 2x^{\vec{i}} x^{\vec{j}} x^{\vec{i}\,\vec{j}})$$

where $x^{\vec{i}}, x^{\vec{j}}, x^{\vec{i}\,\vec{j}}, ..$ denote the first and the second partial derivative of x.
In fact, the use of the two dimensional T and N for the diffusion operator allows a good understanding of behaviour of these operators over the domain Ω. In our case, this is also useful for determining how the function g should be chosen:

- At locations where the variations of the intensity are weak (low gradients), we would like to encourage smoothing, the same in all directions. Assuming that the function g is regular, this isotropic smoothing condition may be achieved by imposing:

$$g'(0) = 0, \qquad \lim_{t \to 0^+} \frac{g'(t)}{t} = g''(0) > 0 \tag{2.31}$$

- Therefore, at the points where $|\nabla x|$ is small, (2.30) becomes:

$$h^t h x - h^t y - \lambda g''(0)(x_{\overrightarrow{TT}} + x_{\overrightarrow{NN}}) = 0 \tag{2.32}$$

  which means that:

$$h^t h x - h^t y - \lambda g''(0) \Delta x = 0 \tag{2.33}$$

In a neighborhood of an edge ξ, if we want to preserve the edge, which is also characterized by a strong gradient, it is preferable to diffuse along ξ (in the T-direction) and not across it. To do this, we assume that:

$$\lim_{t \to +\infty} g''(t) = 0 \qquad \lim_{t \to +\infty} \frac{g'(t)}{t} = k > 0$$

Unfortunately, these two conditions are incompatible. One must find a compromise. For example, $g''(t)$ and $\frac{g'(t)}{t}$ both converge to zero when $t \to +\infty$, but at different rates:

$$\lim_{t \to +\infty} g''(t) = \lim_{t \to +\infty} \frac{g'(t)}{t} = 0 \text{ and } \lim_{t \to +\infty} \frac{t g''(t)}{g'(t)} = 0. \tag{2.34}$$

Notice that many functions g satisfying the conditions (2.31) and (2.34) can be found, for example, the function:

$$g(t) = \sqrt{1 + t^2} \tag{2.35}$$

which is usually called the *hypersurface minimal function*.
Remark: *The conditions (2.31) and (2.34) on g are qualitative. They have been imposed in order to describe the regularization conditions. Naturally, they are not sufficient to ensure that the model is mathematically well posed. Other hypotheses such as convexity, and linear growth are necessary. This is developed in the coming paragraph*

2.3.3 Existence and uniqueness of image restoration problem

In this section, we aim to show the existence and the uniqueness of the following problem:

$$inf\{J(x) = \int_\Omega |y - hx|^2 dt + \lambda \int_\Omega g(|\nabla x|)dt\} \tag{2.36}$$

The following assumptions are necessary for the calculus of variations [**9**, **10**]:

- **Assumption 1:**

$$g \text{ is a strictly convex, non-decreasing function from } \mathbb{R}^+ \text{ to } \mathbb{R}^+. \tag{2.37}$$

- **Assumption 2:**

$$\lim_{t\to+\infty} g(t) = +\infty \tag{2.38}$$

- **Assumption 3:** g grows at most linearly: There exist two constants $\alpha > 0$ and $\beta > 0$ such that

$$\alpha t - \beta < g(t) < \alpha t + \beta, \qquad \forall t > 0 \tag{2.39}$$

- **Assumption 4:**

$$h \text{ is linear continuous operator from } L^2(\Omega) \to L^2(\Omega) \tag{2.40}$$

.

The second remark is assumed with the aim of not penalizing too much the gradient which consequently leads to the deterioration of the edges. We can also make this condition more strong by proposing a growth condition of the type $\lim_{t\to+\infty} g(t) = cte$.
The mathematical study of the problem (2.28) reveals some major difficulties, the space $V = \{x \in L^2(\Omega), \nabla x \in L^1(\Omega)^2\}$ is not reflexive, and we cannot deduce any information from bounded minimizing sequences. So we relax the problem by studying it in the space functions of bounded variations $BV(\Omega)$ which is defined as follow:

Definition 6

Let Ω be a bounded open subset of $\mathbb{R}^L$ and let x be a function in $L^1(\Omega)$. We define $BV(\Omega)$, the space of functions of bounded variation, as:

$$BV(\Omega) = \{x \in L^1(\Omega); \int_\Omega |Dx| = \sup_{\phi\in C_0^1(\Omega)^L, ||\phi||_{L^\infty}\leq 1} \int_\Omega x div(\phi)dt < +\infty\}$$

where $C_0^1(\Omega)^L$ is the space of continuously differentiable functions with compact support in Ω, and $||\phi||_{L^\infty} = \sup_t \sqrt{\sum_{i=1}^L \phi_i^2(t)}$.

Let us return to $BV(\Omega)$. If x belongs to $BV(\Omega)$ and if dx is the N-dimensional Lebesgue measure, then, we get, according to Lebesgue decomposition, the following decomposition:

$$Dx = \nabla x dt + D_{sing}x \tag{2.41}$$

where ∇x and $D_{sing}x$ are respectively the continuous and singular parts of Dx. As stated in [**9**], the term (2.41) is given as:

$$Dx = \nabla x dt + (x^+ - x^-)N_x d\mathbb{H}^1_{|S_x} + C_x$$

where $x^+(t)$ and $x^-(t)$ are respectively the approximate upper and lower limit of x, $\mathbb{H}$ is the **Hausdroff** measure, the term $(x^+ - x^-)N_x d\mathbb{H}^1$ is the jump part and C_x is the Cantor part of $D_{sing}x$. N_x is the normal vector on the jump set $S_x = \{t \in \Omega, x^-(t) < x^+(t)\}$. Replacing in the energy function, the relaxed function associated with (2.28) for the BV-weak topology is defined by [**29**]:

$$\bar{J}(x) = \int_\Omega |y - hx|^2 dt + \lambda \int_\Omega g(|\nabla x|)dt + \lambda\nu \int_{\partial S_x} (x^+ - x^-)d\mathbb{H}^1 + \lambda\nu \int_{\Omega\setminus\partial S_x} |C_x| \tag{2.42}$$

where $\nu = \lim\limits_{t\to+\infty} \frac{g(t)}{t}$. Using all these concepts, the existence of the minimum is ensured thought the following theorem [**110**]:

Theorem 2.2

Under assumptions (2.37-2.40), the minimization problem:

$$\inf_{x\in BV(\Omega)} \bar{J}(x) \tag{2.43}$$

where $\bar{J}$ is defined by (2.42), admits an unique solution $x \in BV(\Omega)$.

Proof. • **Existence**

Let x_n be a minimizing sequence for (2.43). According to (2.38), we have:

$$\begin{cases} |Dx_n|(\Omega) = \int_\Omega g(|\nabla x_n|)dt + \lambda\nu \int_{\partial S_{x_n}} (x_n^+ - x_n^-)d\mathbb{H}^1 + \lambda\nu \int_{\Omega\setminus\partial S_{x_n}} |C_{x_n}| \le M \\ \int_\Omega |y - hx_n|^2 dt \le M \end{cases} \tag{2.44}$$

where M denotes an universal strictly positive constant that may differ from line to line. The first inequality above says that the total variation of Dx_n is uniformly bounded. It remains to prove that $|x_n|_{L^1(\Omega)}$ is bounded. Let $w_n = \frac{1}{\Omega}\int_\Omega x_n dt$ and $v_n = x_n - w_n$. Then, $\int_\Omega v_n dt = 0$ and $Dx_n = Dv_n$. Hence, $|Dv_n| < +\infty$, using the generalized **poincaré-wirtinger** inequality we obtain

$$|v_n|_{L^2(\Omega)} \le k|Dv_n|(\Omega) \le M \tag{2.45}$$

where k is a constant. From the second inequality of (2.44), we deduce

$$|hx_n - y|^2_{L^2(\Omega)} \leq M$$

which is equivalent to:

$$|hv_n + hw_n - y|^2_{L^2(\Omega)} \leq M$$

Rewriting hw_n by

$$hw_n = (hv_n + hw_n - y) - (hv_n - y)$$

we get

$$|hw_n|_{L^2(\Omega)} \leq M + |hv_n|_{L^2(\Omega)} + |y|_{L^2(\Omega)} \leq M'$$

So,

$$|hw_n|_{L^2(\Omega)} = |\frac{1}{|\Omega|}\int_\omega x_n dt||R|_{L^2(\Omega)} \leq M'$$

and thanks to (2.40), we obtain that $|\int_\Omega x_n dt|$ is uniformly bounded. From (2.45), we get $|x_n|_{L^2(\Omega)} = |v_n + \frac{1}{|\Omega|}\int_\omega x_n dt|_{L^2(\Omega)} \leq |v_n|_{L^2(\Omega)} + |\int_\omega x_n dt| \leq M$. Hence, x_n is bounded in $L^2(\Omega)$ and in $L^1(\Omega)$ (Ω is bounded). Since $|Dx_n|(\Omega)$ is also bounded, we deduce that x_n is bounded in $BV(\Omega)$. Thus, up to a subsequences there exists x in $BV(\Omega)$ such that $x_n \underset{BV-topology}{\rightharpoonup} x$ and $hx_n \underset{L^2(\Omega)}{\rightharpoonup} hx$. At the end, from the weak semi-continuity property of the convex function of measures and the weak semi-continuity of the $L^2 - norm$, we get

$$\int_\Omega |hx - y|^2 dt \leq \underline{lim}_{n\to+\infty} \int_\Omega |hx_n - y|^2 dt$$

,

$$\int_\Omega g(Dx) \leq \underline{lim}_{n\to+\infty} \int_\Omega g(Dx_n)$$

which means that,

$$\overline{J}(x) \leq \underline{lim}_{n\to+\infty}\, \overline{J}(x_n) = \inf_{u\in BV(\Omega)} \overline{J}(u)$$

i.e., x is minimum point of $\overline{J}(x)$.

- **Uniqueness**
  Let x^1 and x^2 be two minima of $\overline{J}(x)$. From the strict convexity of g we easily get that $Dx^1 = Dx^2$, which implies that $x^1 = x^2 + c$. But since the function $x \to \int_\Omega |hx_n - y|^2 dt$ is also strictly convex, we deduce that $hx^1 = hx^2$, and therefore $hc = 0$, and from (2.40) we conclude that $c = 0$ and $x^1 = x^2$

□

2.3.4 Image restoration using partial differential equations

2.3.4.1 Linear diffusion (Heat equation)

Among the fundamental operations in image processing, notably image restoration, image filtering is is one of the strong tools for image enhancement. If x^0 is the observed

image to be filtered, then the Gaussian filter is defined as follow:

$$(G_\sigma * x^0)(s) = \int_{IR^2} G_\sigma(s-z)x^0(z)dz \tag{2.46}$$

where G_σ is the Gaussian density of 0-mean and variance σ^2 (Eq.2.8). Two important reasons behind the strength of such filter, the first reason is the fact that $G_\sigma \in C^\infty(IR^2)$, which allows having $G_\sigma * x^0 \in C^\infty(IR^2)$ even if x^0 is not regular. The second advantage is related to the frequency domain. In fact, the Fourier transformation of convolution product $G_\sigma * x^0$ is defined as the product of the Fourier transformation of G_σ and x^0. Since it is known that the Fourier transformation of G_σ is also a Gaussian density of variance $\frac{1}{\sigma^2}$. Thus, for each σ increase, the frequencies in $G_\sigma * x^0$ are mitigated.
Based on Eq.(2.46), it appeared the first and most investigated equation in image processing, it is the known parabolic linear heat equation [**3**, **64**, **69**]:

$$\begin{cases} \frac{\partial x}{\partial t}(t,s) = \Delta x(t,s) \quad t \geq 0, x \in IR^2 \\ x(0,s) = x^0(s) \end{cases} \tag{2.47}$$

where

$$x(s,t) = G_{\sqrt{2t}} * x^0(s)$$

From this formula, we observe that the scale factor t depends on the standard deviation σ by $\sigma = \sqrt{2t}$. That means also that filtering by Gaussian filter with the parameter σ necessitates stopping the diffusion process of (2.47) at the scale $t = \frac{\sigma^2}{2}$.
A point s is said an **edge point** at the scale σ if Δx changes sign in the neighbourhood of s and $|\nabla x(s,t)|$ is very large. When σ is too large, only the high trends are perceivable, which means that the noise is removed but with the expense of blurred features. Whereas, if σ is small, then $x(s,t)$ is close to the degraded image and the noise is still perturbing the image. Several approaches have been proposed to solve this problem [**9**]. The most known is the linear filtering theory elaborated by Marr Hildreth [**72**] and formalized by Witkin [**116**] and Koedernik [**64**]. It is defined as:

$$x(s,t) = \int_{IR^2} \frac{1}{4\pi\sqrt{det(D_t)}} exp(-\frac{(s-z)^T D_t^{-1}(s-z)}{4})x^0(z)dz \tag{2.48}$$

where $D_t = tD$ and D is 2×2-positive defined matrix. Once D is fixed, the convolution product (2.48) is equivalent to the solution of the following equation:

$$\begin{cases} \frac{\partial x}{\partial t}(t,s) = div(D\nabla x(t,s) \quad t \geq 0, x \in IR^2 \\ x(.,s) = x^0(.) \end{cases} \tag{2.49}$$

2.3.4.2 Non linear diffusion (Perona and Malik model)

The objective of the linear models is smoothing the images in order to eliminate the noise without taking account of the images features. The models that we will discuss in this section provide the coherence between the restoration process and the edge protection.
The original idea of the non linear models is orienting the diffusion process toward the

restoration. For now, the domain image will be a bounded open set Ω of IR^2. Let us consider the following equation, initially proposed by Perona and Malik **[85]**:

$$\begin{cases} \frac{\partial x}{\partial t}(t,s) = div(|g(|\nabla x|^2)\nabla x) & (t,s) \in \Omega \times (0,T) \\ \frac{\partial x}{\partial N} = 0 & \text{on } \partial\Omega \times (0,T) \\ x(0,s) = x^0(s) & \text{in } \Omega \end{cases} \tag{2.50}$$

where $g : [0,+\infty[\rightarrow]0,+\infty[$. Before going further, we can remark that if we choose $g \equiv 1$, then we recover the heat Eq.(2.47). Now, imagine that $g(s)$ is a decreasing function satisfying $g(0) = 1$ and $\lim\limits_{t\rightarrow+\infty} g(s) = 0$. With this choice:

- Inside the regions where the magnitude of the gradient of x is weak, Eq.(2.50) acts like the heat equation, resulting in isotropic smoothing.
- Near the region's boundaries where the magnitude of the gradient is large, the regularization is "stopped" and the edges are preserved.

Indeed, we can be more precise if we interpret this divergence operator using the directions $\vec{T}$ and $\vec{N}$ associated with the image. By developing formally the divergence operator, we get:

$$\begin{aligned} div(|g(|\nabla x|^2)\nabla x) = 2((x^{\vec{i}})^2 x^{\vec{i}\,\vec{i}} + (x^{\vec{j}})^2 x^{\vec{j}\,\vec{j}} + 2x^{\vec{i}} x^{\vec{j}} x^{\vec{i}\,\vec{j}}) g'(|\nabla x|^2) \\ + g(|\nabla x|^2)(x^{\vec{i}\,\vec{i}} + x^{\vec{j}\,\vec{j}}) \end{aligned} \tag{2.51}$$

and if we define $\tilde{g}(s) = g(s) + 2sg'(s)$, then (2.50) is rewritten as:

$$\frac{\partial x}{\partial t}(t,s) = g(|\nabla x|^2) x^{\vec{T}\,\vec{T}} + \tilde{g}(|\nabla x|^2) x^{\vec{N}\vec{N}} \tag{2.52}$$

Therefore, (2.52) may be interpreted as a sum of the diffusions in the T and N-Directions. The functions $g(|\nabla x|^2)$ and $\tilde{g}(|\nabla x|^2)$ acting as weighting coefficients. Of course, since N is normal to the edges, it would be preferable to smooth more in the tangential direction T than in the normal direction N. Thus, we impose $\lim\limits_{t\rightarrow+\infty} \frac{\tilde{g}(s)}{g(s)} = 0$, or equivalently, according to the definition of $\tilde{g}$, $\lim\limits_{t\rightarrow+\infty} \frac{sg'(s)}{g(s)} = -\frac{1}{2}$ If we restrict ourselves to functions $g(s) > 0$ with a power growth, then the above limit implies that $g(s) \simeq \frac{1}{\sqrt{(s)}}$ as $s \rightarrow \infty$. The question now is to know whether (2.50) is well posed or not. First, the Eq.(2.50) can be written as:

$$\frac{\partial x}{\partial t}(t,s) = a_{11}(|\nabla x|^2) x^{\vec{i}\,\vec{i}} + a_{22}(|\nabla x|^2) x^{\vec{j}\,\vec{j}} + 2a_{12}(|\nabla x|^2) x^{\vec{i}\,\vec{j}} \tag{2.53}$$

where

$$\begin{cases} a_{11}(|\nabla x|^2) = 2(x^{\vec{i}})^2 g'(|\nabla x|^2) + g(|\nabla x|^2) \\ a_{12}(|\nabla x|^2) = 2x^{\vec{i}} x^{\vec{j}} g'(|\nabla x|^2) \\ a_{22}(|\nabla x|^2) = 2(x^{\vec{j}})^2 g'(|\nabla x|^2) + g(|\nabla x|^2) \end{cases} \tag{2.54}$$

To tackle perfectly the Eq.(2.53), we may use the parabolic PDE theory, which means that the coefficients a_{ij} must verifies:

$$\sum_{i,j=1,2} a_{ij} v_i v_j \geq 0, \quad \forall v \in IR^2 \tag{2.55}$$

which is equivalent to

$$g(s) + 2g'(s) > 0, \ \ \forall s > 0$$

To summarize, the assumptions imposed on $g(s)$ are

$$\begin{cases} g : [0, +\infty[\rightarrow]0, +\infty[, & \text{g is decreasing.} \\ g(0) = 1, & g(s) \simeq \frac{1}{\sqrt{(s)}} \text{ as } s \rightarrow \infty \\ g(s) > 0 & \end{cases} \tag{2.56}$$

A suitable choice for these conditions is $g(s) = \frac{1}{\sqrt{s+1}}$.

Remark:*It must be noted that the assumptions (2.56) are not sufficient to apply directly the result of the parabolic equations to show the existence. Indeed, the difficulty comes from the highly degenerate behaviour of (2.50) due to the vanishing condition $g(s) \simeq \frac{1}{\sqrt{s}}$ as s tends to $+\infty$ [**9**].*

As a matter of fact, one can find some classical results for equations of the form:

$$\frac{\partial x}{\partial t} = div(a(t, s, x, \nabla x)) \tag{2.57}$$

where the function a satisfies the structural conditions:

$$a(t, s, x, \nabla x) \geq \alpha_0 |\nabla x|^p - \beta_0(s, t),$$

and

$$|a(t, s, x, \nabla x)| \leq \alpha_1 |\nabla x|^{p-1} + \beta_1(s, t),$$

almost every (s, t) with $p > 1$, $\alpha_{i_{i=1,2}}$ are given constants and β_i are non negative functions satisfying some integrability conditions [**27**].

More details about the Results of existence and uniqueness for the solution of Eq.(2.50) are described in [**9**].

2.4 Conclusion

Image degradation is a common phenomenon that exists in various applications like photography, remote sensing, medical imaging, etc. The degradation may occur due to camera mis-focusing, atmospheric turbulence, relative object-camera motion and various other reasons. For the last few decades, researchers are working in the field of image restoration. The problem is still open due to its ill-posed nature and requires significant research.

This chapter has discussed the mathematical vision of image restoration. In first section, we have described the major components that make image restoration model. In fact, we have started by image digitization where we have explained the elementary processes,

(sampling and quantization), used for building a digital image. Next, the side effects of the digitization as the noise and the blur are illustrated and supported by examples. The mathematical image restoration problem has been explored and developed in the second section, in which we have began by the degradation model that generates distorted images. This model has been used afterwards to reformulate restoration as minimization of a suitable energy function. Thereafter, some techniques as the least squares and the iterative methods have been utilized for solving the build model. Since the proposed model is typically an example of ill-posed problems, the regularization technique has been introduced, thus making a well posed energy function with a regularization term that verifies several analytical properties that we have use to explore existence and uniqueness results. In the last part, we have recalled the use of the Heat and Perona equations for image restoration, as two fundamental partial differential equations that have been investigated properly in image restoration.

Chapter 3

Stability and parameters estimation of CHN applied to the non linear problem of image restoration

3.1 Abstract

In image processing, restoration of grey level images from blur and noise has drawn much attention by researchers. Mathematically, this problem has been solved by artificial neural networks, notably the discrete Hopfield neural network (DHN), which has been widely investigated as a solver of combinatorial problems. However, DHN is still a limited model because of the fluctuation problem that comes from the use of the hard limit function as neurons activators. To overcome this shortcoming, we propose to use the continuous Hopfield network (CHN). In fact, this model allows, throughout the relaxation, to extend the research area of solutions. In this regard, we suggest a new energy function with appropriate parameters settled from the stability analysis of the network. Performance of our method is demonstrated numerically and visually by several computational tests.
Keywords: Image restoration. Optimization. Artificial neural network. Continuous Hopfield neural network. Penalty function.

3.2 Introduction

Image Restoration Problem (IRP) has started since the 50s after many studies carried out to improve the quality of the images received in the space program [**61**]. Restoration techniques are applied to remove:

- System degradations, such as blur owing to optical system aberrations, atmospheric turbulence, motion and diffraction.
- The electronic noise, which affects the measurement.

Several methods have been proposed in the literature to solve IRP, and they have proved to a great extent their effectiveness in image restoration while preserving as possible the

image details.
The filtering methods are one of the most effective approaches that have been used widely to enhance the image quality. Indeed, in addition to the Gaussian filter, which was described in chapter 2, these techniques also include the Median filter [**7**, **58**], Wiener filter [**81**], inverse and kalaman filters [**117**]. However, the Median filter allows a good noise suppression, but it deteriorates the image features. Wiener filter is implemented only after the wide sens stationary assumption has been made for images. Inverse filter generates a good restoration when the image is not affected by noise. However, in noisy case, it produces bad effects. Kalaman filter can deal with non-stationary images, but it is computationally very intensive
Recently, a novel image processing technology based on variational methods is motivating many mathematicians and image scholars to research on this challenging topic. This technique as it was explained in chapter 2 protects the image details by the resolution of the mathematical IRP along the boundary space function where the discontinuities are preserved.
Among the variational image restoration models, the Total Variational (TV) model, proposed by Rudin and Fatemi, has been the most important and has been properly applied in several applications [**92**]. Other extensions of TV technique have been suggested to restore degraded images as proposed in [**11**, **125**].
Image restoration problem has been also solved by Partial Differential Equations (PDE) [**9**]. Motivated by the isotropic and anisotropic properties described by the Heat (2.47) and Perona equations (2.50),(2.53), different studies have deeply investigated these PDEs to rebuild the degraded images [**64**, **85**]. However, these equations of second order are very weak since they tend to cause the processed image to exhibit staircase effects. To avoid this drawback, recent studies have been proposed based on Four order PDE (FOPDE), and they have proved their performance because of the high smoothing ability [**123**]. The optimization filed has also contributed to image restoration using the modelling techniques and resolution. For example, we may cite those that are inspired by the biological behaviours like genetic algorithms [**94**, **57**], and artificial neural networks [**124**], [**82**, **108**]. The first use of ANNs in image restoration was due to Zhou and al [**124**]. Indeed, they have been the first who proposed to solve the combinatorial problem of IRP by Hopfield *ANN* [**48**, **49**], in which they have mapped IRP into the discrete HNN by comparing the energy function associated to IRP with the Lyapunov function. In spite of the efficiency of this model, it still suffering from the instability during the neurons updates. To overcome this problem, Paik and Katssagelos have proposed a Modified HNN model, in which they suggested several improved algorithms using sequential, decoupled n-simultaneous and partially asynchronous modes of updates [**82**]. In [**108**], authors have reduced the number of iterations by feeding the neurons outputs only to higher-order neurons. However, these methods are based on the discrete activation function, which often limits the search space because of the fluctuation problem. To avoid this drawback, we suggest in this chapter to use the continuous Hopfield neural network for image restoration problem. In the new process of resolution named (CHN-IRP), we start by defining a new energy function associated with IRP utilizing the penalty method. Next, we map IRP into the CHN, thus extracting the weights and the bias of the network. Thereafter, we discuss the parameters of the energy function ensuring, on one hand, the convergence of the feasible

solutions, and the instability of interior points on the other hand.
The rest of this chapter is organized as follows. In the next Section, we briefly review the different modes of number representations. Image restoration problem is described in section 3.4. Resolution of IRP by the discrete Hopfield neural network is explained in Section 3.5. The use of the continuous Hopfield neural network for image restoration, as well as the convergence discussions are given in Section 3.6. Section 3.7 presents experimental results of CHN-IRP. Finally, conclusions are given in Section 3.8.

3.3 Number representation schemes

Practically, for each problem, the solutions are described by a set of numbers. Therefore, the first task to do is encoding these numbers using the neuron states space V [**78**]. While allowing neurons to take continuous state values during the process of energy function minimization, we demand that they take binary values of 1 or 0 at the final stage, so that we can obtain digital solutions like those given by digital computers. For simplicity, we first assume the numbers are positive integers including 0, although we can also represent general bipolar and complex numbers by using additional neurons. Generally, there are three different schemes of mapping the positive integer numbers onto the neuron states.

3.3.1 Binary scheme

A common scheme of representing numbers in machines is to use binary digits. For example, 5 is expressed by 0101. This scheme utilize $log_2(N+1)$ bits to represent a number N. If we let one neuron represent 1 bit, we have a one-to-one correspondence between elements in the number space $\mathbb{Z}^+$ and those in the neuron state space V. Despite the economy in the number of bits or neurons used, the the binary scheme is not fault-tolerant. In other words, even a single failure in a highly significant bit gives rise to a large error in the number represented.

3.3.2 Simple Sum Scheme

In this scheme, each number is expressed by a simple sum of the neuron state variables v_i, i.e., the total number of firing $(v_i = 1)$ neurons. For example, the number Five can be represented by 0011111, 0101111, 1101011, all of which have five 1-bits. This is a one-to-many mapping from $\mathbb{Z}^+$ to V, and the numbers have degenerate representations. This scheme requires N bits to express a number N and is not economical in the number of bits or neurons. However, it is highly fault-tolerant because an error in a single bit does not cause a large error in the number represented. So far, we have compared the binary scheme and the simple-sum scheme from the viewpoint of their fault tolerance. More important is their difference in problem-solving capability. As will be seen, problems are solved through a spontaneous energy minimization process in a neural network, and the solution is given by a point in the neuron state-variable space that is reached after this minimization process. Unlike binary scheme where only one neuron is used represents the correct solution, the simple-sum scheme uses multiple points to express the correct

solution. Because of this degeneracy and the clustering of quasi-minimum energy points in the neuron state-variable space, the simple-sum scheme gives more chances to reach the correct solution. Suppose, for example, 3 is the correct solution. In the simple sum scheme, we can get a correct solution when the final state is either 00111, 10110, 11100, or 10101 etc., whereas we can get the correct solution in the binary scheme only when the final state is 00011 **[56]**.

3.3.3 Group-and-Weight Scheme

Despite its merit in fault-tolerance and computational capability, the simple-sum scheme requires too many neurons when solutions include large numbers. We propose the group-and-weight scheme which lies between the binary and the simple-sum schemes. In this scheme, we divide the total q bits into K groups, each of which has M bits $(q = KM)$ and interpret the groups as digits whose numbers are given by simple sums of the bits in the corresponding groups. For example, with $q = 6, K = 2, M = 3$, then 5 is expressed either by $100100([4^1(1+0+0)+4^0(1+0+0)=5), 010001, 001010$, or 100001 etc.
A number expression for the simple-sum scheme is given by

$$\sum_{i=1}^{K}\left[(M+1)^{i-1}(\sum_{j=1}^{M} v_{(i-1)M+j})\right] \tag{3.1}$$

The expression includes the binary and the simple sum schemes as special cases. When we put $M = 1$and $K = q$, we obtain a number expression for the binary scheme

$$\sum_{i=1}^{q} 2^{i-1} v_i \tag{3.2}$$

and when we put $M = q$ and $K = 1$,we obtain a number expression for the simple-sum scheme

$$\sum_{i=1}^{q} v_i \tag{3.3}$$

The number of bits needed to express a number N is given in the Table 3.1.

Table 3.1: Complexities of bits required for the numbers representations

Simple Sum Scheme	Simple Sum Scheme	Group-and-Weight Scheme
$log_2(N+1)$	$Nlog_{N+1}(N+1)$	$Mlog_{M+1}(N+1)$

3.4 Image restoration problem

Practically, the image degradation can be, adequately, modelled by a linear blur (motion, defocusing, atmospheric turbulence...) and an additive white Gaussian noise (2.46). In

this sense, the degradation model is given by:

$$y = h * x + \eta \tag{3.4}$$

where x and y denote respectively the original and the degraded images, h is the blur operator where its form is extracted from the point spread function (PSF). In the case of the shift invariant blur (paragraph 2.2.1.1), H is represented by a Toeplitz matrix where the corresponding form depends on the choice of the boundary condition (BCs) used in the problem (section 2.2). η is an additive noise and $*$ is the convolution product.
The restoration model as it was mentioned in (2.27) is given as:

$$\min_{x\in\Omega} J(x) = \int_\Omega |y - h * x|^2 dt + \lambda \int_\Omega |\nabla x|^2 dt \tag{3.5}$$

In the discrete variable, the problem (3.5) is expressed by:

$$\min_{x\in[|0,..,255|]^{N^2}} J(X) = \frac{1}{2}\|Y - HX\|^2 + \frac{1}{2}\lambda\|DX\|^2 \tag{3.6}$$

where X, Y are the vectors of the original and degraded images. H is the Blur matrix (2.2). The added term $\|DX\|$ is a high pass filter (paragraph 2.3.2.3). λ is the regularization parameter that measures the trade off between the closeness to data and and the prior knowledge of the solution. $\|.\|$ is the euclidean norm.
Obviously Eq.(3.6) is simplified by:

$$\underset{X\in[|0,..,255|]^{N^2}}{Min} J(X) = \sum_{p=1}^{N^2}(Y_p - \sum_{i=1}^{N^2} H_{p,i}X_i)^2 + \frac{1}{2}\lambda \sum_{p=1}^{N^2}(\sum_{i=1}^{N^2} D_{pi}X_i)^2 \tag{3.7}$$

In the next section, we aim to explain the details of the resolution of *IRP* by the discrete Hopfield neural network.

3.5 Discrete Hopfield neural Network for solving IRP

In this paragraph, we explain the principle steps that have been used to solve image restoration problem based on discrete Hopfield network [**124**, **82**].
Let X be the Gray level function of an image of size $N \times N$, and G its maximal value. First, we transform X into a vector using the following formula:

$$X_m = X_{(i-1)N+j} \tag{3.8}$$

In the second step, we represent each element of the image X by a simple sum scheme (3.9) as it is described in [**78**] :

$$X_i = \sum_{k=1}^{G} v_{ik} \tag{3.9}$$

where :

$$v_{ik} = f(u_{ik}) = f(\sum_{j=1}^{N^2}\sum_{l=1}^{G} T_{ik,jl}v_{jl} + I_{ik}) = \begin{cases} 1 \ if \ \ u_{ik} \geq 0 \\ 0 \ if \ \ u_{ik} < 0 \end{cases} \tag{3.10}$$

The function $f(t)$ denotes the Heavside activation function.
Next, we replace the data X in Eq.(3.7). Thus, we obtain:

$$\underset{v\in\{0,1\}^{N^2\times G}}{Min} E(v) = \sum_{p=1}^{N^2}(Y_p - \sum_{i=1}^{N^2}(H_{p,i}\sum_{k=1}^{G} v_{ik}))^2 + \frac{1}{2}\lambda \sum_{p=1}^{N^2}(\sum_{i=1}^{N^2}(D_{pi}\sum_{k=1}^{G} v_{ik}))^2 \tag{3.11}$$

which equals to:

$$\begin{aligned}\underset{v\in\{0,1\}^{N^2\times G}}{Min} E(v) = \frac{1}{2}\sum_{p=1}^{N^2}\sum_{i=1}^{N^2}\sum_{j=1}^{N^2}\sum_{k=1}^{G}\sum_{l=1}^{G} H_{p,i}H_{p,j}v_{ik}v_{jl} \\ + \frac{1}{2}\lambda\sum_{p=1}^{N^2}\sum_{i=1}^{N^2}\sum_{j=1}^{N^2}\sum_{k=1}^{G}\sum_{l=1}^{G} D_{p,i}D_{p,j}v_{ik}v_{jl} \\ - \sum_{p=1}^{N^2}\sum_{i=1}^{N^2}\sum_{k=1}^{G} y_p H_{p,i}v_{ik} + \frac{1}{2}\sum_{p=1}^{N^2} y_p^2\end{aligned} \tag{3.12}$$

Afterwards, we map the model (3.12) in a *DHN* by proposing a network with $N^2 \times G$ neurons mutually interconnected. The suggested network is characterized by:

- The set of the network states:
  $v = \{\ v_{i,k}\ /\ i = 1,..,N^2 \quad and \quad k = 1,..,G\}$.
- The weight matrix:
  $T = \{T_{ik,jl}\ /\ i,j = 1,..,N^2 \quad and \quad k,l = 1,..,G\}$

Using the identification between Eq.(3.12) and Lyapunov function (1.31), we obtain

$$Map = \begin{cases} T_{ik,jl}^{DHN} = -\sum_{p=1}^{N^2} H_{p,i}H_{p,j} - \lambda\sum_{p=1}^{N^2} D_{p,i}D_{p,j} \\ I_{ik}^{DHN} = \sum_{p=1}^{N^2} y_p H_{p,i} \end{cases} \tag{3.13}$$

Two remarks that must be mentioned, the first is that the weights are independent of the subscripts (k,l), which reduces the complexity by neglecting the repeated terms. The second remark concerns the self-connections $T_{ik,ik}$, which are not equal to zero, and then, it contradicts the convergence criterion mentioned in the proposition 1.1. To solve this problem, Zhou and al have proposed a decision rule, which depends on the variations of the energy function ΔE and the neurons states v_{ik}. It is described as follow **[124]**:

$$v_{ik} = \begin{cases} v_{ik}^{new} & \text{if } \Delta E \text{ due to state change } \Delta v_{ik} \text{ is less than zero} \\ v_{ik}^{old} & \text{otherwise} \end{cases} \tag{3.14}$$

where $\Delta E = E_{new} - E_{old}$ and $\Delta v_{ik} = v_{ik}^{new} - v_{ik}^{old}$.
Image restoration using *DHN* is summarized in the following steps:

1. Set the initial state of the neurons

2. Update the state of all neurons randomly and asynchronously according to the decision rule (3.14).
3. Check the energy function, if energy does not change continue, otherwise go back to step 2.
4. Construct the image using Eq.(3.8).

Although the DHN deals successively with image restoration problem, it suffers from the fluctuation problem. In other words, the network state keeps wandering around the state space near the minima of the energy function, thus generating an oscillatory behaviour. To overcome this drawback, we propose to use the continuous model of Hopfield neural networks. This is the subject of the next section.

3.6 Using continuous Hopfield neural network for solving IRP

3.6.1 Mapping IRP into CHN

The main purpose of this section is to investigate the CHN for solving image restoration problem. To this end, we define an appropriate energy function which takes into a count the IRP particularities. To be precise, the choice of the parameters of this function must ensure the feasibility of CHN equilibrium points. Based on the energy function defined in Eq.(3.12), we add a new constraint that describes the feasibility of the neurons states. The constructed model is:

$$(P) = \begin{cases} \text{subject to} & min\ E(v) \\ & \sum_{i=1}^{N^2}\sum_{k=1}^{G}(v_{ik}(1-v_{ik})) = 0 \\ & v \in [0,1]^{N^2\times G} \end{cases} \tag{3.15}$$

The role of such constraint is to force neurons states to take 0 or 1.
Next, we use the penalty function approach to build a new energy function associated with (P) **[114]**. Thus, the energy function E is transformed to:

$$\begin{aligned} \underset{v\in[0,1]^{N^2\times G}}{Min} E(v) = {} & \frac{1}{2}\alpha \sum_{p=1}^{N^2}\sum_{i=1}^{N^2}\sum_{j=1}^{N^2}\sum_{k=1}^{G}\sum_{l=1}^{G} H_{p,i}H_{p,j}v_{ik}v_{jl} \\ & + \frac{1}{2}\lambda \sum_{p=1}^{N^2}\sum_{i=1}^{N^2}\sum_{j=1}^{N^2}\sum_{k=1}^{G}\sum_{l=1}^{G} D_{p,i}D_{p,j}v_{ik}v_{jl} - \alpha \sum_{p=1}^{N^2}\sum_{i=1}^{N^2}\sum_{k=1}^{G} y_p H_{p,i}v_{ik} + \\ & \frac{1}{2}\alpha \sum_{p=1}^{N^2} y_p^2 + \frac{1}{2}\beta \sum_{i=1}^{N^2}\sum_{k=1}^{G}(v_{ik}(1-v_{ik})) \end{aligned} \tag{3.16}$$

where the scaling parameters α, β, and λ are real parameters that are settled by the network convergence analysis.

Comparing this equation with Lyaponuv function described by Eq.(1.31), we obtain the following parameters:

$$\begin{cases} T_{ik,jl}^{CHN} = -\alpha \sum_{p=1}^{N^2} H_{p,i} H_{p,j} - \lambda \sum_{p=1}^{N^2} D_{p,i} D_{p,j} + \beta \\ I_{ik}^{CHN} = \alpha \sum_{p=1}^{N^2} y_p H_{p,i} - 2\beta \end{cases} \tag{3.17}$$

3.6.2 Convergence analysis and parameters estimation

Convergence of CHN is an important task in the resolution of combinatorial problems. Indeed, in many cases, the adopted network converges toward an equilibrium point which is not a feasible solution. Generally, the stability of CHN is analysed over the three sets: feasible set H_F, Hypercube infeasible corner set $H_C - H_F$ and interior points set $H - H_C$ (Figure 3.1). In our case, any point of H_C makes a feasible solution which means that H_F and H_C are identical and thereafter the stability analysis will be made only for H_C and $H - H_C$. Thus, we aim to ensure the following objectives:

- Convergence of feasible solutions to an equilibrium point.
- guarantee the instability of interior points.

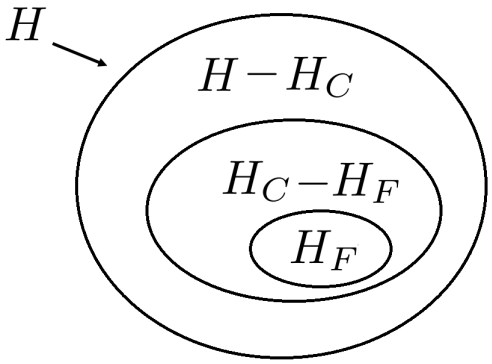


Figure 3.1: Feasible and infeasible sets

3.6.2.1 Convergence of feasible solutions $v \in H_F$

For each feasible solution, the stability of corner set is ensured from the equilibrium point criteria (1.35), (1.36).
First, to minimize the first term of Eq.(3.16), we impose the following condition:

$$\alpha \geq 0 \tag{3.18}$$

In the next conditions, we will focus on equilibrium points to achieve the stability conditions. Thus, we have:

Theorem 3.1

If α, β and λ are three parameters verifying the following assertions:

$$(S1) = \begin{cases} \alpha, \lambda \geq 0 \\ \beta \geq G \times N^4(\alpha M_1 + \lambda M_2) \end{cases} \tag{3.19}$$

with $M_1 = \underset{\forall(p,i)\in\{1,..,N\}^2}{max} H_{p,i}$ and $M_2 = \underset{\forall(p,i)\in\{1,..,N\}^2}{max} D_{p,i}$.
Then, any point v is an equilibrium point.

Proof. As stated above, the equilibrium point is ensured from the partial derivative of the energy function. In this regard, we calculate the derivative on v_{ik} as follow:

$$E_{ik}(v) = \frac{\partial E}{\partial v_{ik}} = \frac{1}{2}\alpha \sum_{p=1}^{N^2}\sum_{j=1}^{N^2}\sum_{l=1}^{G} H_{p,i}H_{p,j}v_{jl} + \frac{1}{2}\lambda \sum_{p=1}^{N^2}\sum_{j=1}^{N^2}\sum_{l=1}^{G} D_{p,i}D_{p,j}v_{jl} - \alpha \sum_{p=1}^{N^2} y_p H_{p,i} + \frac{1}{2}\beta((1-2v_{ik})) \tag{3.20}$$

taking into account that the terms of H and D are positive, we obtain the following assertion:

$$\underset{\{i/v_i=0\}}{min} E_i(v) \geq 0 \Longrightarrow \beta - 2\alpha \sum_{p=1}^{N^2} y_p H_{p,i} \geq 0$$

Which means that:

$$\underset{\{i/v_i=0\}}{min} E_i(v) \geq 0 \Longrightarrow \beta \geq 2\alpha N^2 G M_1 \tag{3.21}$$

On the other side, to ensure the negativity of $\underset{\{i/v_i=1\}}{max} E_i(v)$, we obtain from (3.20):

$$\underset{\{i/v_i=1\}}{max} E_i(v) \leq 0 \Longrightarrow \alpha \sum_{p=1}^{N^2}\sum_{j=1}^{N^2}\sum_{l=1}^{G} H_{p,i}H_{p,j} + \lambda \sum_{p=1}^{N^2}\sum_{j=1}^{N^2}\sum_{l=1}^{G} D_{p,i}D_{p,j} - \beta \leq 0$$

By maximizing, we obtain the following result:

$$\underset{\{i/v_i=0\}}{min} E_i(v) \leq 0 \Longrightarrow -\beta + GN^4(\alpha M_1^2 + \lambda M_2^2) \leq 0 \tag{3.22}$$

Finally, it is obvious to see that $2\alpha GN^2 \leq \alpha GN^4$. Consequently, the condition (3.21) is covered by the inequation (3.22). □

3.6.2.2 Non convergence of infeasible solutions $v \in H - H_F$:

Before going any further, we begin by giving the following definitions:

Definition 7

a point $v \in H$ is said an interior point if and only if :
$\exists\, v_{i,k} \in v \ / \ v_{i,k} \in \,]0,1\,[$

Definition 8

Let $v \in H$ be an interior point, v defines an edge point if and only if:
$card\{(i,j) \in \{1,..,n+1\} \times \{1,..,N_{max}\} / \ \ v_{i,k} \in \,]0,1[\} = 1$

Definition 9

Let $v \in H$ be an edge point such that $v_{a,b} \in \,]0,1\,[$, v' is called the adjacent corner of v if:

$$v'_{i,j} = \begin{cases} v_{i,j} \ \text{ if } \ (i,j) \neq (a,b) \\ 1 \ \text{ or } \ 0 \ \text{ if } \ (i,j) = (a,b) \end{cases} \tag{3.23}$$

As mentioned in Appendix (B), interiors points not being edge points have a null probability of being the convergence point of the CHN.
Any equilibrium interior point $v \notin H_C$ cannot be a local minimum and it is necessary to analyze the second derivatives of the generalized energy function (3.16):

$$\frac{\partial^2 E}{\partial {v_{ik}}^2} = -T_{ik,ik} = \alpha \sum_{p=1}^{N^2} H_{p,i}^2 + \lambda \sum_{p=1}^{N^2} D_{p,i}^2 - \beta \tag{3.24}$$

In this regard, non-convergence of any edge point is ensured through the following Theorem **[102]**:

Theorem 3.2

Let $v \in H - H_C$ be an edge point such that $v_{i,k} \in \,]0,1\,[$, if $E_{i,j}(v')$ verifies:

$$E_{i,j}(v') \begin{cases} \notin \,[0, -T_{ij,ij}\,] \ \ \forall \ \ (i,j)/v'_{i,j} = 1 \\ \notin \,[T_{ij,ij}, 0\,] \ \ \forall \ \ (i,j)/v'_{i,j} = 0 \end{cases} \tag{3.25}$$

Then, the stability of v is avoided.

Practically, these conditions are replaced by:

$$\begin{cases} \underline{E}^0(v) \leq T_{ik,ik} \\ \underline{E}^1(v) \geq -T_{ik,ik} \end{cases} \tag{3.26}$$

Fortunately, these conditions are verified as shown below:

$$\underline{E}^0(v) = -2\alpha N^2 G M_1 + \beta \leq T_{ik,ik} = -\alpha \sum_{p=1}^{N^2} H_{p,i}^2 - \lambda \sum_{p=1}^{N^2} D_{p,i}^2 + \beta \tag{3.27}$$

and,

$$E^1(v) = \alpha \sum_{p=1}^{N^2} \sum_{j=1}^{N^2} \sum_{l=1}^{G} H_{p,i} H_{p,j} + \lambda \sum_{p=1}^{N^2} \sum_{j=1}^{N^2} \sum_{l=1}^{G} D_{p,i} D_{p,j} - \beta$$
$$\geq T_{ik,ik} = +\alpha \sum_{p=1}^{N^2} H_{p,i}^2 + \lambda \sum_{p=1}^{N^2} D_{p,i}^2 - \beta \quad (3.28)$$

In summary, solving the problem of image restoration by CHN is described in the following algorithm:

Data:
- Degraded image Y of size N^2 ;
- The matrix H and D ;
- Initialization of itermax ;

Result: Restored image X

- Encode the image X using Eq.(3.9);
- Map IRP into CHN using Eq.(3.16);
- Extract weights matrix T and bias I from Eq.(3.17);
- Determine α, β and λ using Theorem 3.1;
- Initialize a random v^0;
- Initialize the counter $iter \leftarrow 1$.;
- $E^{old1} \leftarrow 0$, $E^{new} = E^{old1} + 1$;

while $|E^{new} - E^{old1}| > \epsilon$ *or* $iter \leq itermax$ **do**
  $E^{old1} \leftarrow E^{new}$;
  for *Each neuron* $v_{i,k}$ **do**
    $E^{old2} \leftarrow 0$;
    while $|E^{old2} - E^{new}| \geq \epsilon$ **do**
      $E^{old2} \leftarrow E^{new}$;
      for each neuron, update the state using (3.10) ;
      $E^{new} \leftarrow E(v)$
    end
    $iter \leftarrow iter + 1$;
  end
end

Algorithm 4: Image restoration using the continuous Hopfield network (CHN-IRP)

3.7 Experiments

In this section, we present some numerical results to illustrate the performance of the proposed method. The test images are shown in Figure 3.2 with sizes of 256 x256. All experiments are carried out on Windows 7 32-bit and Matlab 9 running on a desktop

equipped with an Intel Core i2 CPU 2.20 GHz and 3 GB of RAM.
In most cases, restoration methods provide results which are almost similar and this

(a) (b) (c)

Figure 3.2: Test images: (A) IRM: spine (B) House (C) Boat.

makes some difficulties to distinguish visually between the restored images, therefore, the judgment becomes subjective. For this reason, we propose the known measure PSNR (Peak to Signal Noise Ratio) to identify and to compare the effects of image enhancement algorithms on image quality. The PSNR is defined as follows:

$$PSNR = 10log_{10}\frac{255^2}{MSE}, \tag{3.29}$$

MSE: is the Mean Square Error, which is defined as:

$$MSE = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N}(X_1(i,j) - X_2(i,j))^2}{M \times N}, \tag{3.30}$$

where X_1 and X_2 denote, respectively, the restored and the original images.

The performance of the proposed model is compared against some of the most popular image filters, such as Median, wiener, FOPDE and Total variational method TV. In order to fairly compare our method with these methods, the next Tables show the PSNR of the restored images using these filters. Methods were applied on different noise densities =0.01 to =0.29 with increments of 0.04. The parameters of the CHN are calculated based on Eqs.(3.19).

As shown in these tables, denoising filters such as wiener and median filters have provided a better PSNR in comparison with the other analytical methods that have generated a lower PSNR because of the original features changes owing to the isotropy phenomena. In addition, theses results reports that our algorithm produces for all noise densities a good PSNR than those obtained by Median, wiener, FOPDE and TV. Our results are supported by graphic result of the CHN-IRP applied to "House" image. In fact, in all the noise densities, the curve of our methods (black colour) outperforms the others curves with a considerable exceed.

Table 3.2: PSNR values of CHN-IRP and comparison filters for "IRM spine" test image at different Gaussian noise density

Methods	0.01	0.05	0.09	0.13	0.17	0.21	0.25	0.29
Noisy	20.09	19.12	17.43	15.72	14.13	12.72	11.47	10.37
Median	29.36	24.80	20.46	17.52	15.29	13.55	12.05	10.78
FOPDE	28.83	24.29	20.27	17.42	15.22	13.48	12.01	10.78
Wiener	26.97	23.64	20.00	17.29	15.15	13.44	11.99	10.76
TV	7.40	7.40	7.41	7.41	7.41	7.42	7.42	7.42
CHN-IRP	32.47	27.13	24.33	22.13	19.23	18.81	15.33	13.73

Table 3.3: PSNR values of CHN-IRP and comparison filters for "Boat" test image at different Gaussian noise density

Methods	0.01	0.05	0.09	0.13	0.17	0.21	0.25	0.29
Noisy	20.02	19.09	17.48	15.77	14.21	12.81	11.65	10.61
Median	24.89	22.59	19.58	17.03	15.02	13.34	11.95	10.74
FOPDE	23.55	21.69	19.07	16.74	14.84	13.23	11.93	10.80
Wiener	26.16	23.21	19.85	17.20	15.15	13.45	12.09	10.92
TV	5.44	5.44	5.44	5.44	5.45	5.45	5.45	5.45
CHN-IRP	31.51	26.73	24.05	21.87	19.08	18.31	15.52	14.23

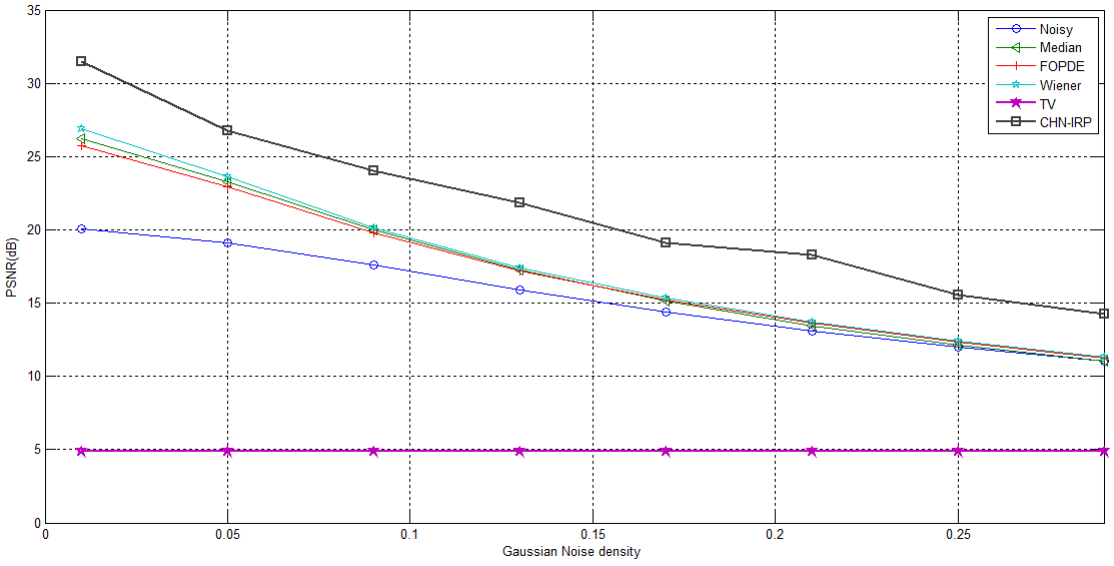


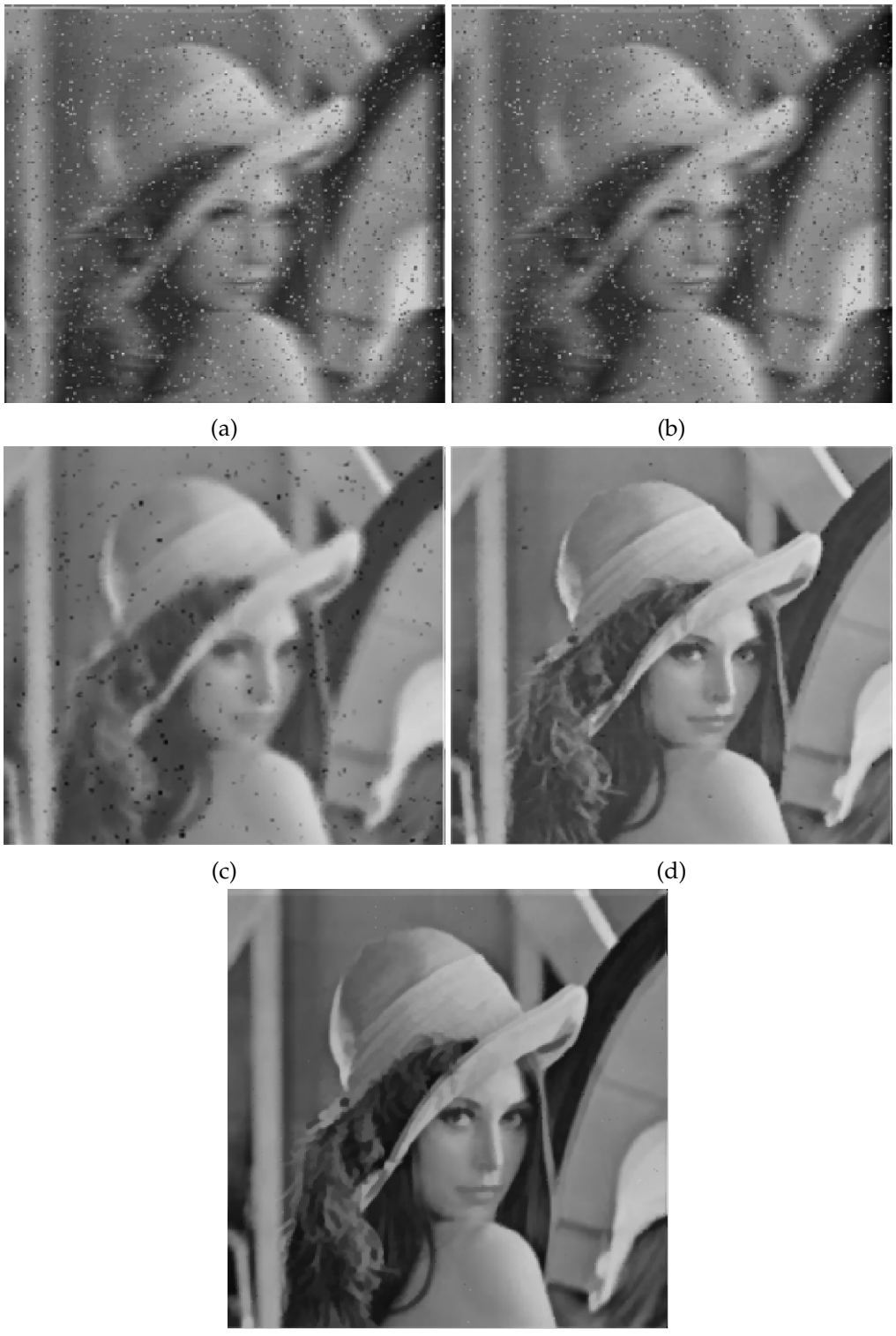
Figure 3.3: PSNR curves of CHN-IRP and comparison filters for "House" test image at different Gaussian noise density

In the next experiment, we compare the efficiency of our model with the others HNN-based filters including MHNN [**82**] and MRHNN [**108**]. For this purpose, we use "Lena" (Figure 3.4a) test image that is degraded by a shift invariant blur and 30% of impulse noise

(Figure 3.4b). In Figure 3.4, MHNN has restored the image with a small improvement at the deblurring scale contrary to the denoising process where the restoration does not give a much (Figure 3.4c). Restoration by MRHNN outperforms MHNN as shown in [108], it has generated an enhanced image in term of denoising and deblurring while preserving as much as possible the image quality (Figure 3.4d). In Figure 3.4e, Our method has generated a good restored image which is close to the original image. Features are also protected in term of discontinuities and contrast.
In the Figure 3.5, we show the performance of our model applied to "Spine" image, as well as its comparison with other methods from the literature. As shown, the proposed CHN-IRP provides an encouraging outcome against the others methods. Indeed, CHN-IRP allows a good restoration thanks to its ability to protect the image information especially with this kind pictures. For more illustration, in Figure 3.5h, the position of the heart is clear and can be easily marked on the right of the picture which is not possible for the other restoration techniques.

3.8 Conclusion

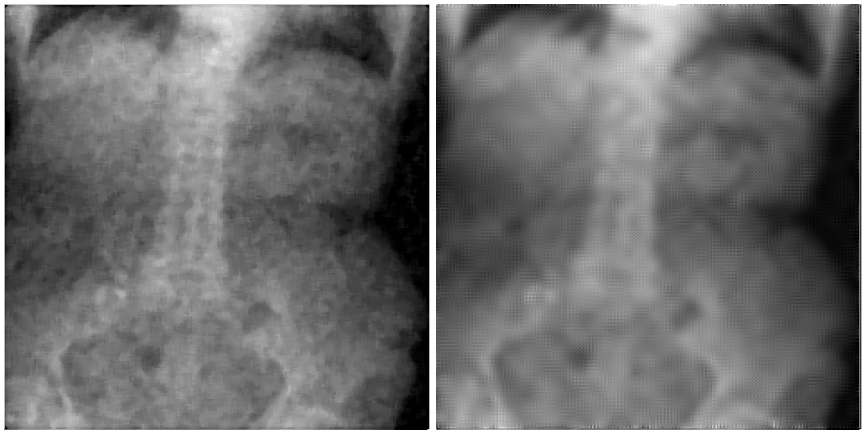
In this chapter, the continuous Hopfield neural network has been employed to deal with image restoration problem. The choice of the use of such model comes from the fluctuation drawback that limits the search space of solutions. Firstly, IRP has been mapped into the CHN using the penalty approach. Parameters of the network are extracted from Lyaponuv function while the the parameters of the IRP energy are estimated from the stability analysis of CHN in which the equilibrium point in attained. Experimental results have proved the efficiency of CHN-IRP through several numerical and visual tests.



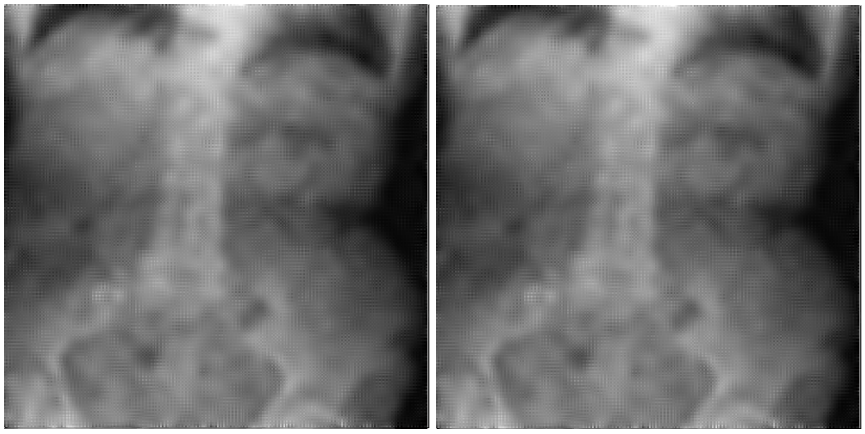
(a) (b) (c) (d) (e)

Figure 3.4: Comparison of "Lena" outcomes using the different methods : (a) original (b) degraded by blur and impulse noise $\rho = 0.3$ (c) MHNN (d) RMHNN (e) CHN.

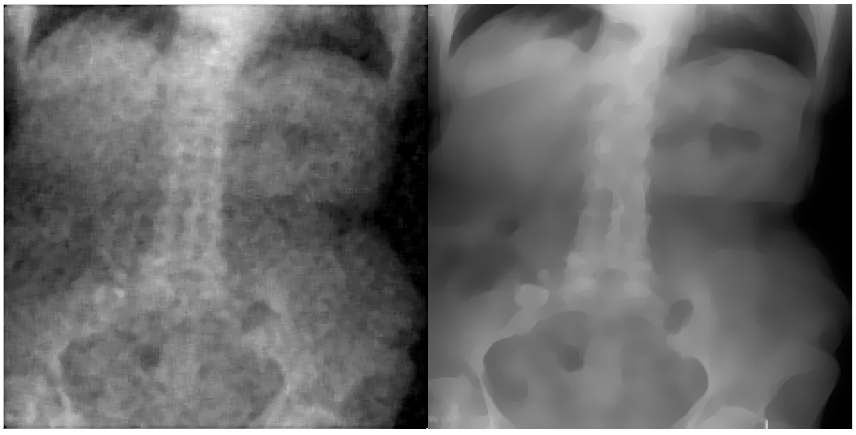
(a) (b)



(c) (d)



(e) (f)



(g) (h)

Chapter 4

New non linear optimization model for selective image restoration: modelling and resolution by CHN and genetic algorithm

4.1 Abstract

In grey level image restoration, a prior knowledge of degraded areas allows, thanks to the selective filtering, to achieve a good protection of the image features. In this chapter, we propose a quadratic programming based technique that deals with the issue of details preservation during the restoration process. Based on the classical model of image restoration, we build a modified model by introducing a set of binary variables that indicate the pixels categories. We combine each pixel with the median of its neighbours in a decision rule so that one of them generates the optimal solution. The obtained model is a non-linear mixed-integer problem where resolution by exact methods is not feasible. In this regard, we use the both of continuous Hopfield neural network (CHN) and genetic algorithm (GA) to solve the suggested model. Performance of our method is demonstrated numerically and visually by several computational tests. .

Keywords: Image restoration. Optimization. Selective image restoration. Median. Continuous Hopfield neural network. Penalty function. Quadratic programming. Genetic algorithm.

4.2 Introduction

Despite the effectiveness of the restoration techniques, their uses are always subject to some imperfections. One of these deficiencies is the deterioration of the original features during the restoration. In fact, the current restoration methods perform uniformly across the images, and thus tend to change both of degraded and free degraded areas. Consequently, the features of the image are blurred and distorted. For this reason, we propose a mixed-integer model for selective image restoration (MI-SIR). In the classical model of

restoration (3.7), a set of binary variables are technically introduced through a decision rule, thus building a mixed-integer function that takes account of the degraded and free degraded areas [**36**]. The constructed model is a complex non-linear problem where the resolution by exact methods is very difficult, if not impossible. In this context, we suggest, on one side, to use the continuous Hopfield network (CHN) to solve the binary part of the model, in which we start by defining a new energy function associated with MI-SIR utilizing the penalty method [**114**]. Next, we map the MI-SIR model into a suitable CHN by extracting the weights and the bias of the network. Thereafter, we discuss the parameters of the energy function ensuring, on one hand, the convergence of the feasible solutions and the instability of interior points on the other hand. On the other side, we use the genetic algorithm (GA) to deal with the discrete part of the proposed model.
The rest of the chapter is organized as follows: the proposed selective image restoration model is explained in Section 4.3. In Section 4.4, we show in details the resolution using CHN and genetic algorithm. Section 4.5 presents the experimental results of the new technique. Finally, conclusions are given in Section 4.6.

4.3 New optimization model for selective image restoration

Recalling the fundamental model of image restoration (3.7):

$$\underset{X\in\{0,..,255\}^{N^2}}{Min} J(X) = \sum_{p=1}^{N^2}(Y_p - \sum_{i=1}^{N^2} H_{p,i}X_i)^2 + \frac{1}{2}\lambda \sum_{p=1}^{N^2}(\sum_{i=1}^{N^2} D_{pi}X_i)^2 \tag{4.1}$$

Generally, the most of studies that have contributed to image restoration are performed globally and uniformly across the images. Consequently, these methods apply to each area in the image without considering whether it is degraded or not. Thus, the image is restored at the expense of blurred and distorted original features [**58**]. While the filtering techniques, notably Median and Gaussian filters, are efficient for image restoration, they still performing more than the necessary, especially with the areas that are not degraded. For this purpose, we propose to introduce a novel concept of image filtering using the non linear quadratic optimization. It consists in restoring while preserving the image features using the model (4.1), in which we insert a new variable $v \in \{0,1\}^{N^2}$ defined as follows:

$$v_i = \begin{cases} 1, \text{if the current pixel } X_i \text{ is original} \\ 0, \text{ else} \end{cases} \tag{4.2}$$

In the next step, we collaborate the value of each pixel X_i with the median of its neighbours M_i in a decision rule named R_i. The proposed rule provides two outputs according to v_i values. It is expressed as follows:

$$R_i = v_i X_i + (1 - v_i)M_i \tag{4.3}$$

The idea behind Eq.(4.3) is to decide on which one of X_i and M_i provides a minimal value of R_i. Afterwards, the proposed rule R_i replaces X_i in Eq.(4.1). Thus, the problem is transformed to a mixed-integer model denoted by $E(v, X)$:

$$\underset{(v,X)\in\{0,1\}^{N^2}\times\{0,..,255\}^{N^2}}{Min} E(v,X) = \frac{1}{2}\sum_{p=1}^{N^2}\sum_{i=1}^{N^2}\sum_{j=1}^{N^2} H_{p,i}H_{p,j}(v_iX_i+(1-v_i)M_i)(v_jX_j+ (1-v_j)M_j) - \sum_{p=1}^{N^2}\sum_{i=1}^{N^2} H_{p,i}(v_iX_i+(1-v_i)M_i)Y_p + \frac{1}{2}\lambda\sum_{p=1}^{N^2}\sum_{i=1}^{N^2}\sum_{j=1}^{N^2} D_{p,i}D_{p,j}(v_iX_i+(1-v_i)M_i)(v_jX_j+(1-v_j)M_j) + \frac{1}{2}\sum_{p=1}^{N^2} Y_p^2 \quad (4.4)$$

The rule proposed by (4.3) plays an important rule in the optimization of image restoration problem. Indeed, the value of R_i may considerably reduce the cost of the energy $J(X)$ following the proposition 4.1

Proposition 4.1

Let X, Y, and M denote respectively the original, degraded and Median images of size $N \times N$. If λ is a real parameter that verifies $\lambda = 1$ and X' is an image of size N^2 that verifies

$$X'_i = \begin{cases} X_i, \ \forall i \neq k \\ M_k, \ else \end{cases} \quad (4.5)$$

Then, the absolute difference of the energy function between X and X' is bounded as follow:

$$|\Delta J(X)| \leq |X_k - M_k| N^2 \|X\|_1 \quad (4.6)$$

Proof. First, let develop the energy function defined in (4.1):

$$J(X) = \frac{1}{2}\sum_{p=1}^{N^2}\sum_{i=1}^{N^2}\sum_{j=1}^{N^2} H_{p,i}H_{p,j}X_iX_j - \sum_{p=1}^{N^2}\sum_{i=1}^{N^2} H_{p,i}X_iY_p + \frac{1}{2}\lambda\sum_{p=1}^{N^2}\sum_{i=1}^{N^2}\sum_{j=1}^{N^2} D_{p,i}D_{p,j}X_iX_j + \frac{1}{2}\sum_{p=1}^{N^2} Y_p^2 \quad (4.7)$$

By identifying X_k in the expression of $J(x)$, (4.7) becomes:

$$J(X) = \frac{1}{2}\sum_{p=1}^{N^2}\sum_{j=1}^{N^2}\Big(\sum_{\substack{i=1\\ i\neq k}}^{N^2} H_{p,i}H_{p,j}X_iX_j + H_{p,k}H_{p,j}X_kX_j\Big) - \sum_{p=1}^{N^2}\Big(\sum_{\substack{i=1\\ i\neq k}}^{N^2} H_{p,i}X_iY_p + H_{p,k}X_kY_p\Big) + \frac{1}{2}\lambda\sum_{p=1}^{N^2}\sum_{j=1}^{N^2}\Big(\sum_{\substack{i=1\\ i\neq k}}^{N^2} D_{p,i}D_{p,j}X_iX_j + D_{p,k}D_{p,j}X_kX_j\Big) + \frac{1}{2}\sum_{p=1}^{N^2} Y_p^2 \quad (4.8)$$

Now, let replace the pixel X_k by the median value M_k:

$$\underset{x_k=M_k}{J(X)} = \frac{1}{2}\sum_{p=1}^{N^2}\sum_{j=1}^{N^2}(\sum_{i=1}^{N^2} H_{p,i}H_{p,j}X_iX_j + H_{p,k}H_{p,j}M_kX_j) - \sum_{p=1}^{N^2}(\sum_{i=1}^{N^2} H_{p,i}X_iY_p + H_{p,k}M_kY_p) + \frac{1}{2}\lambda\sum_{p=1}^{N^2}\sum_{j=1}^{N^2}(\sum_{i=1}^{N^2} D_{p,i}D_{p,j}X_iX_j + D_{p,k}D_{p,j}M_kX_j) + \frac{1}{2}\sum_{p=1}^{N^2} Y_p^2 \quad (4.9)$$

Thus, the absolute variation of J, given by $|\Delta J(X)| = |J(X) - J(X')|$, is expressed as

$$|\Delta J(X)| = |\frac{1}{2}(X_k - M_k)\sum_{p=1}^{N^2}\sum_{j=1}^{N^2} H_{p,k}H_{p,j}X_j + \frac{1}{2}\lambda(X_k - M_k)(\sum_{p=1}^{N^2}\sum_{j=1}^{N^2} D_{p,k}D_{p,j}X_j) - (X_k - M_k)\sum_{p=1}^{N^2} H_{p,k}Y_p| \quad (4.10)$$

which is also equals to

$$|\Delta J(X)| = \frac{1}{2}|X_k - M_k||\sum_{p=1}^{N^2}\sum_{j=1}^{N^2} H_{p,k}H_{p,j}X_j + \frac{1}{2}\lambda\sum_{p=1}^{N^2}\sum_{j=1}^{N^2} D_{p,k}D_{p,j}X_j - 2\sum_{p=1}^{N^2} H_{p,k}Y_p| \quad (4.11)$$

while in most cases of image processing, the coefficients of the kernels D and H belong to the range $[0, 1]$. Thus, under the assumption $\lambda = 1$, we obtain:

$$|\Delta J(X)| \leq |X_k - M_k|N^2\|X\|_1 \quad (4.12)$$

□

From proposition 4.1, we deduce that for only one pixel X_k, the choice of the original or the median values can make a large difference in the minimization of the energy function.

4.4 Using CHN-GA for solving MI-SIR

4.4.1 CHN for solving the binary part of MI-SIR

In the resolution phase, we aim to solve a mixed-integer model with binary and discrete variables. Regarding the binary variables, we suggest to use the CHN instead of the Discrete HHN for two reasons:

- To avoid the fluctuation problem, which appears from the discrete HNN.
- To use the parameter settings to ensure the stability of the dynamic system associated with CHN.

4.4.1.1 Mapping the proposed MI-SIR model

CHN and GA will perform progressively at several iterations until the desired image is obtained. In this regard, we fix the X at the iteration $(t-1)$ and we solve the following model:

$$\min_{v\in\{0,1\}^{N^2}} E(v,X) \tag{4.13}$$

Next, we use the penalty function approach to build a new energy function associated with (4.28) [**114**]. Thus, the Hopfield energy function E^H is defined by:

$$\begin{aligned} E^H(v,X) = \frac{1}{2}\alpha \sum_{p=1}^{N^2}\sum_{i=1}^{N^2}\sum_{j=1}^{N^2} H_{p,i}H_{p,j}(v_iX_i+(1-v_i)M_i)(v_jX_j+(1-v_j)M_j) \\ - \alpha \sum_{p=1}^{N^2}\sum_{i=1}^{N^2} H_{p,i}(v_iX_i+(1-v_i)M_i)Y_p \\ + \frac{1}{2}\lambda \sum_{p=1}^{N^2}\sum_{i=1}^{N^2}\sum_{j=1}^{N^2} D_{p,i}D_{p,j}(v_iX_i+(1-v_i)M_i)(v_jX_j+(1-v_j)M_j) \\ + \alpha\frac{1}{2}\sum_{p=1}^{N^2} Y_p^2 + \frac{\gamma}{2}\sum_{i=1}^{N^2} v_i(1-v_i) \end{aligned} \tag{4.14}$$

where α, λ and γ are real parameters that we will settle from the network analysis. The role of the last term is to keep the (0-1) feasible solution [**114**]. In order to reduce the expression of Eq.(4.14), we denote by $\Psi^{\alpha,\lambda}_{p,i,j}$ the term $\alpha H_{p,i}H_{p,j}+\lambda D_{p,i}D_{p,j}$. So, the Eq.(4.14) is rewritten as

$$\begin{aligned} E^H(v,X) = \frac{1}{2}\sum_{p=1}^{N^2}\sum_{i=1}^{N^2}\sum_{j=1}^{N^2} \Psi^{\alpha,\lambda}_{p,i,j} \prod_{k\in\{i,j\}} (v_kX_k+(1-v_k)M_k) \\ - \alpha \sum_{p=1}^{N^2}\sum_{i=1}^{N^2} H_{p,i}(v_iX_i+(1-v_i)M_i)Y_p \\ + \alpha\frac{1}{2}\sum_{p=1}^{N^2} Y_p^2 + \frac{\gamma}{2}\sum_{i=1}^{N^2} v_i(1-v_i) \end{aligned} \tag{4.15}$$

where $\prod$ denotes the product operator.
In the following step, we map the Eq.(4.15) into a CHN by proposing a network with N^2 neurons mutually interconnected. The suggested network is characterized by:

- The set of the network states:

$$v = \{\ v_i\ /\ i = 1,..,N^2\}$$

- The weight matrix:

$$T = \{T_{i,j} \ / \ i,j = 1,..,N^2\}$$

Comparing Eqs.(1.31) and (4.15), and by neglecting the constant term, the weights are then given by

$$T_{i,j} = -(X_i - M_i)(X_j - M_j) \sum_{p=1}^{N^2} \Psi_{p,i,j}^{\alpha,\lambda} + \frac{\gamma}{2}\delta_{ij} \tag{4.16}$$

where $\delta_{ij} = 1$ if (i=1) and 0 otherwise.
Thereafter, we extract the bias I from the following term:

$$Iv = \frac{1}{2}\alpha \sum_{p=1}^{N^2}\sum_{i=1}^{N^2}\sum_{j=1}^{N^2} \Psi_{p,i,j}^{\alpha,\lambda}(v_i(X_iM_j - M_iM_j) + v_j(X_jM_i - M_jM_i)) \\ - \alpha \sum_{p=1}^{N^2}\sum_{i=1}^{N^2} H_{p,i}(v_iX_i + (1-v_i)M_i)Y_p + \frac{\gamma}{2}\sum_{i=1}^{N^2} v_i. \tag{4.17}$$

The symmetry of the term $v_i(X_iM_j - M_iM_j)$ alows deducing the i^{th} component of I. Thus, we get:

$$I_i = -\sum_{p=1}^{N^2}(\sum_{j=1}^{N^2} \Psi_{p,i,j}^{\alpha,\lambda}(v_i(X_iM_j - M_iM_j)) + \alpha H_{p,i}Y_p(X_i - M_i)) - \frac{\gamma}{2} \tag{4.18}$$

In the next paragraph, we use the convergence and the stability analysis of CHN to discuss the parameters of Eq.(4.14) over the Hypercube $[0,1]^{N^2}$.

4.4.1.2 Stability analysis

Convergence of CHN is an important task in the resolution of combinatorial problems . In fact, in many cases, the adopted network converges toward an equilibrium point which is not feasible solution. Generally, the stability of CHN is analysed over the three sets: feasible set $H_F = \{v \in H \ / \$ v stistsfy the constraints$\}$, Hypercube infeasible corner set $H_C - H_F = \{v \in H_C / v \notin H_F\}$ and interior points $H - H_C$ (Figure 4.1a). In our case, any point of H_C makes a feasible solution, which means that H_F and H_C are identical and thereafter the stability analysis will be made only for H_C and $H - H_C$ (Figure 4.1b).
Based on Eqs.(1.35) and (1.36), the stability analysis is ensured from the partial derivative of the energy function which is given by:

$$E_i^H(v,X) = \frac{\partial E^H(v,X)}{\partial v_i} = \alpha(X_i - M_i)\sum_{p=1}^{N^2}(\sum_{j=1}^{N^2} \Psi_{p,i,j}^{\alpha,\lambda}(v_jX_j + (1-v_j)M_j) \\ - H_{p,i}Y_p) + \frac{\gamma}{2}(1-2v_i) \tag{4.19}$$

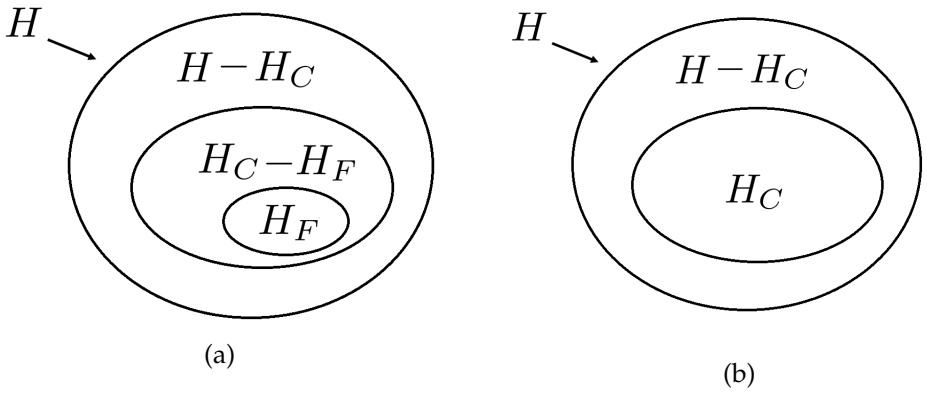


Figure 4.1: Feasible and infeasible sets to be analysed: (a) constrained problems (b) unconstrained problems

4.4.1.3 Convergence of corners set $H_c = \{0,1\}^{N^2}$

Stability of the corners set is ensured from the equilibrium point criteria [**102**].
To minimize the first term of equation 4.15, we impose the following condition:

$$\alpha \geq 0 \tag{4.20}$$

Let suppose that H and D are positive matrix. If $\alpha, \lambda \geq 0$, then, it is obvious to see that above a certain rank s, we have the following inequality:

$$\sum_{j=1}^{s} \Psi_{p,i,j}^{\alpha,\lambda}(v_j X_j + (1 - v_j)M_j) \geq H_{p,i}Y_p \quad \forall i, p = 1, .., N^2 \tag{4.21}$$

Let S denotes the following term:

$$S = \max_{(i,p)\in\{1,2,3,..,N^2\}^2} \{\sum_{j=1}^{s} \Psi_{p,i,j}^{\alpha,\lambda}(v_j X_j + (1 - v_j)M_j) - H_{p,i}Y_p\}$$

S is positive value that we subsequently use to extract the equilibrium conditions.
The quantity $X_i - M_i$ is bounded by -255 and 255 which refers to the two cases of Salt and pepper noise (Figure 4.2).
Equations (4.22), (4.23), are obtained from the equilibrium point criteria applied to $v \in H_C$:

$$\underline{E}^0(v) \geq 0 \ \forall v_i = 0 \Longrightarrow -255\alpha N^2 S + \frac{\gamma}{2} \geq 0 \tag{4.22}$$

For the second condition, we have:

$$\underline{E}^1(v) \leq 0 \ \forall v_i = 1 \Longrightarrow 255\alpha N^2 S - \frac{\gamma}{2} \leq 0 \tag{4.23}$$

Obviously, this condition is grossly ensured since $255\alpha N^2 S$ is too large.

$$\begin{pmatrix} 255 & 255 & 255 \\ 255 & 0 & 255 \\ 255 & 255 & 255 \end{pmatrix} \quad \begin{pmatrix} 255 & 255 & 255 \\ 255 & 255 & 255 \\ 255 & 255 & 255 \end{pmatrix}$$

(a) (b)

$$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 255 & 0 \\ 0 & 0 & 0 \end{pmatrix} \quad \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

(c) (d)

Figure 4.2: (a) White block with black center (b) Median version of a (c) black block with white center (d) Median version of c

4.4.1.4 Divergence of interior points

In the previous part, some conditions were imposed to ensure the stability of any corner point $v \in H_C$. However, there could exist an invalid solution $v \in H - H_C$ which is an equilibrium point. To deal with this issue, we give some conditions to avoid that the CHN will not be trapped in any invalid solution $v \in H - H_C$. In this regard, we recall the following concepts:

Definition 10

a point $v \in H$ is said an interior point if, and only if :

$$\exists v_i \in v / v_i \in \,]0,1\,[$$

Definition 11

Let $v \in H$ be an interior point, v defines an edge point if, and only if:

$$card\{i \in \{1,..,N^2\} /\ \ v_i \in \,]0,1\,[\} = 1$$

Definition 12

Let $v \in H$ be an edge point such that $v_a \in]0,1[$, v' is called the adjacent corner of v if:

$$v'_i = \begin{cases} v_i \text{ if } i \neq a \\ 1 \text{ or } 0 \text{ if } i = a \end{cases}$$

As mentioned in **[83]**, interiors points not being edge points have a null probability of being the convergence point of the CHN. In this context, the stability of any edge point is avoided by the use of the following theorem:

Theorem 4.1

Let the edge point $\{v \in H - H_C \ / \ v_i \in]0,1[$, if γ is sufficiently small and v' is the adjacent corner of v verifying:

$$E_i^H(v') \begin{cases} \notin [0, (X_i - M_i)^2 \sum_{p=1}^{N^2} \Psi_{p,i,i}^{\alpha,\lambda} - \frac{\gamma}{2}] & \forall i / v'_i = 1 \\ \notin [-(X_i - M_i)^2 \sum_{p=1}^{N^2} \Psi_{p,i,i}^{\alpha,\lambda} + \frac{\gamma}{2}, 0] & \forall i / v'_i = 0 \end{cases} \tag{4.24}$$

Then, the stability of any edge point is avoided.

Proof. As stated in **[102]**, the stability of any edge's adjacent with $T_{i,i} < 0$ is guaranteed if and only if:

$$E_i^H(v') \begin{cases} \notin [0, -T_{i,i}] \ \ \forall \ \ i / v'_i = 1 \\ \notin [T_{i,i}, 0] \ \ \forall \ \ i / v'_i = 0 \end{cases} \tag{4.25}$$

From the Eq.(4.16), the auto-connections are given by:

$$T_{i,i} = -(X_i - M_i)^2 \sum_{p=1}^{N^2} \Psi_{p,i,i}^{\alpha,\lambda} + \frac{\gamma}{2}$$

Thus, with a γ small enough, the condition $T_{i,i} < 0$ is easily satisfied.
Therefore, we obtain:

$$E_i^H(v') \notin [T_{i,i}, 0] \Longleftrightarrow E_i(v') \notin [-(X_i - M_i)^2 \sum_{p=1}^{N^2} \Psi_{p,i,i}^{\alpha,\lambda} + \frac{\gamma}{2}, 0] \quad / v'_i = 0 \tag{4.26}$$

$$E_i^H(v') \notin [0, -T_{i,i}] \Longleftrightarrow E_i(v') \notin [0, (X_i - M_i)^2 \sum_{p=1}^{N^2} \Psi_{p,i,i}^{\alpha,\lambda} - \frac{\gamma}{2}] \quad / v'_i = 1 \tag{4.27}$$

□

4.4.2 GA for solving the discrete part

In larger problems such as image restoration, it is appropriate to use a meta-heuristic method instead of exact one. One such method is the genetic algorithm (GA) which is widely used in many domains due to its advantages [**46**]. Indeed, this method converges in less time and can be adapted to any type of problem without any condition (continuity or derivability of the objective function). In addition, the possibility of convergence to a local solution is reduced thanks to the initial population which starts from many possible solutions.

4.4.2.1 Representation and initialisation

In this part, we aim to solve the following problem:

$$\min_{X \in \{0,255\}^{N^2}} E(v, X) \tag{4.28}$$

where v is the current solution obtained from the CHN.
The population is a matrix of integer values chosen from the set $\{0, 255\}$, where each row named a chromosome (individual) represents a possible solution of the image X. Figure 4.3 shows an example of one individual.
Initial population can be randomly generated, but there exist other ways to generate the

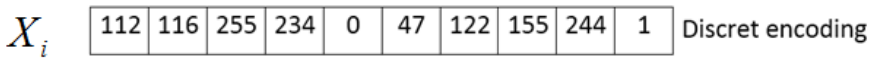


Figure 4.3: Example of one chromosome

initial population like applying other heuristics. Moreover, a prior information on the nature of degradation may help to set a good initialization of the population. For example, in the case of impulse noise reduction [**58**], if the noise density is very low, then, few pixels are affected by noise. In this regard, a custom initialization of population may leads to better and fast searching process. In fact, the initialization with the degraded image values allows a fast convergence toward the solution with a small number of iterations.

- **Fitness:**
  In this step, a numerical value called Fitness is assigned to each individual. It measures the performance of chromosomes based on the objective function .i.e an individual who has a great fitness is the most adapted to the problem. In our work, we suggest the following fitness:

  $$f(v, X) = \frac{1}{1 + E(v, X)} \tag{4.29}$$

  where minimizing the objective cost is equivalent to maximizing the Fitness function.

- **Selection:**
  During the processes of the algorithm, we select the best individuals having the greatest fitness to be reproduced. This selection is conducted by roulette-wheel selection. This type of selection assigns to each individual a spot in a wheel. The spot size depends on the fitness of the individual. Precisely the size of the spot increases with the fitness of the individual. Then a ball is thrown in the wheel. The more the individual is fittest, the more the probability that the ball throws in its spot is great. The probability is defined by $P_i = \frac{f(v_i, X_i)}{\sum_{i=1}^{N} f(v_i, X_i)}$ and the selection point is chosen randomly from the interval $[0, 1]$. Figure 4.4 shows an example of the RWS with 4 individuals.

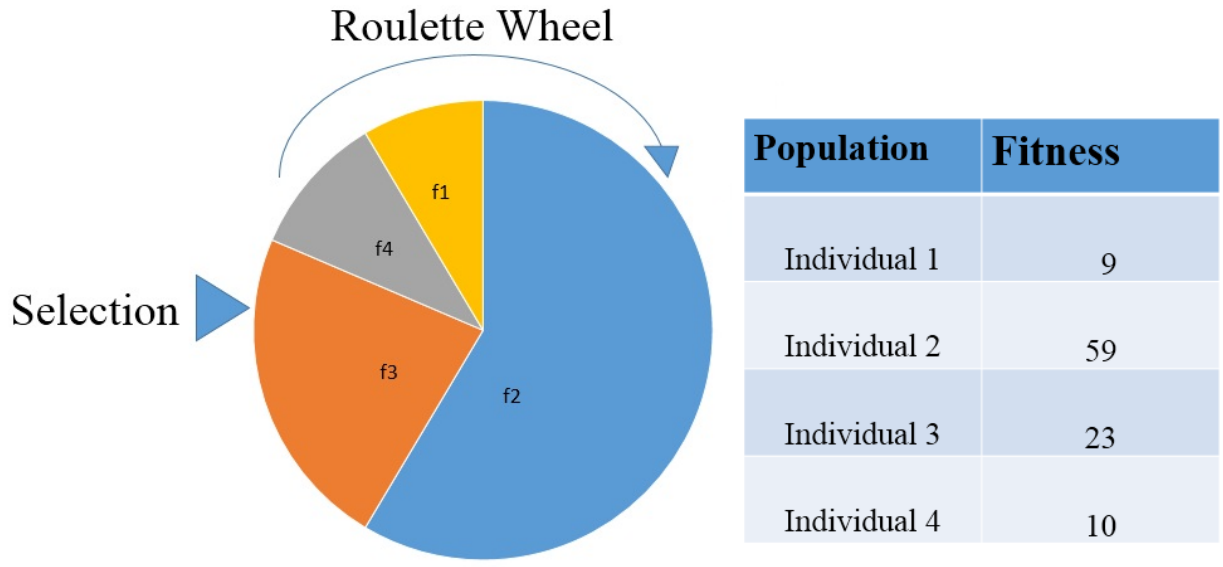


Figure 4.4: Example of RWS applied to four individuals

4.4.2.2 Reproduction: Mutation and Crossover

Reproduction guarantees diversification and intensification of the search space thanks to its two operators: crossover and mutation. Regarding the first operator, we have applied the one-point crossover which consists in cutting at the same point (chosen randomly) two parents and then, swapping the right portions between themselves, thus constituting two new strings. While the crossover can only explore the combinations of the current parents, the mutation operation creates new individuals by changing the value of a predefined number of an arbitrary location in the chromosome. We chose randomly a gene which will be replaced by a random integer from 0 to 255. This operator is generally applied with a low probability to avoid disrupting crossover results.
The CHN and GA will be performed progressively at several iterations. At each iteration of resolution, the process gives two solutions $I^t = (v^t, X^t)$, which minimize Eq.(4.15). The binary solution v^t protects the image features in X^{t-1} while the discrete solution provides

X^t as a restored image. Then, X^{t-1} and X^t are used to construct the current solution using the following rule:

$$X_i^t = \begin{cases} X_i^{t-1}, \text{if } v_i^t = 1 \\ X^t, \text{ else} \end{cases} \tag{4.30}$$

The generated solution X^t will define an initial input in the next iteration. The main steps of the proposed technique are described in the following algorithm:

CHN-GA for solving MI-SIR.

1. Take the degraded image as an initial value denoted X^{**}
2. Use the CHN to achieve the decision solution: $v^* = \underset{v}{argmin} E(v, X^{**})$
3. Use the Genetic algorithm to compute the discrete solution: $X^* = \underset{X}{argmin} E(v^*, X)$
4. Use v^* to perform the selective filtering on $\{X^{**}, X^*\}$, thus forming a new restored solution that overwrites X^{**}
5. Check the energy function; if the energy variation does not change any more, a restored image is obtained; otherwise, go back to step 2 for another iteration.
6. Return the restored image X^{**}.

Remark: in the decision rule R_i, we may replace the median value M_i by a new value generated by an improved version of median filter or another method. Indeed, these methods correct the pixels of degraded images by values that are close to the reel pixel. As a consequence, they provide a minimal cost of the energy function defined by Eq.(4.14).

4.5 Experiments

In this section, we perform some experiments to evaluate the proposed method, as well as the effect of the selective technique for image features protection. The test images are shown in Figure 4.5 with sizes of 256×256. All experiments are carried out on Windows 7 32-bit and Matlab 9 running on a desktop equipped with an Intel Core i2 CPU 2.20 GHz and 3 GB of RAM.

The performances are quantitatively measured by the PSNR that we recall as follows:

$$PSNR = 10 log_{10} \frac{255^2}{MSE}, \tag{4.31}$$

where MSE is the Mean Square Error:

$$MSE = \frac{\sum_{i=1}^{M} \sum_{j=1}^{N} (X_1(i,j) - X_2(i,j))^2}{M \times N},$$

and X_1, X_2 denote, respectively, the restored and the original images.
In the first experiment, the mixture original-median defined by the decision rule (4.3) is

Figure 4.5: Test images: (a) Synthetic block (b) Boat (c) House (d) Zelda (e) Barbara (f) Airplane (g) Man (h) Peppers (i) Mandrill (j) Cameraman (k) Stephen

tested on a deep level. For this purpose, the model is applied to a synthetic block of size 4×4. Figure 4.6 provides the restored outcomes of a noisy block using the Median filter and MI-SIR, as well as their gray-level values.
As shown from Figure 4.6c, the degradation has affected four pixels ($x(2,2)$, $x(3,1)$, $x(3,3)$ and $x(4,4)$), which thereafter are replaced by Median filter values in Figure 4.6e. The Median filter has improved the quality of the noisy block to a reasonable level, but it was unsuccessful in protecting the original pixels. In Figure 4.6g, the restored image using MI-SIR is close to the original block. Indeed, thanks to the decision rule, the restoration is performed only for the targeted pixels, which improves considerably the quality of the restored image and make it close as possible to the original image.
In the second experiment, the performance of the proposed model is compared against some of the most popular image filters, such as Median, Wiener, FOPDE, classical Total Variational method (TV), BM3D and WNNM. In order to fairly compare our method with these methods, Tables 4.1, 4.2, 4.3 and 4.4 show the PSNR of the restored images. These methods have been applied to several test images (Figure 4.5) degraded by a shift invariant blur of the following kernel [**124**]:

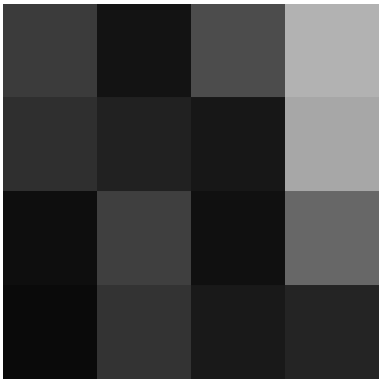
$$h(k,l) = \begin{cases} \frac{1}{2}, \text{if} k = l = 0, \\ \frac{1}{16}, \text{if } |k| < 1, |l| < 1 \text{and } (k,l) \neq (0,0), \end{cases} \tag{4.32}$$

and a Gaussian noise of densities (0.05 to 0.35 with increments of 0.1). For each iteration of the process, the parameters of the CHN are calculated based on Eqs.(4.22), (4.23) and Theorem 4.1.

Table 4.1: PSNR values of MI-SIR and comparison filters applied to test images degraded by a shift invariant blur and 5% of Gaussian noise

	σ=5%							
Methods	Degraded	Median	FOPDE	Wiener	TV	BM3D	WNNM	MI-SIR
Boat	19.58	21.72	20.96	22.30	5.43	21.90	23.56	24.10
House	19.57	22.64	21.09	22.72	7.90	22.09	23.68	24.23
Zelda	19.86	22.94	21.97	22.99	8.19	21.27	22.86	23.97
Barbara	19.72	22.41	20.85	22.93	6.48	22.61	22.88	23.67
Airplane	19.12	22.37	21.57	22.65	5.83	22.43	23.14	23.79
Man	19.66	22.69	21.73	22.48	7.67	22.80	23.17	23.93
Peppers	19.55	22.54	21.08	22.86	7.72	22.89	23.44	24.03
Mandrill	19.59	22.68	21.16	22.50	6.52	22.77	23.52	24.12
Cameraman	19.45	22.78	21.45	22.52	6.61	22.82	23.64	24.23
Stephen	19.60	22.15	21.63	22.36	6.59	22.61	23.31	23.69

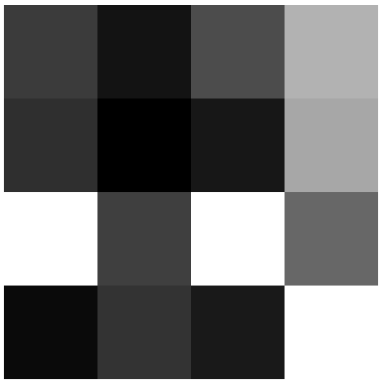
As shown from Tables 4.1, 4.2, 4.3 and 4.4, BM3D and WNNM , which are known by their efficiency in denoising processes [**59**], have provided a significant $PSNR$ in comparison with the other methods except the case of 5%, in which they are almost similar to the Median and wiener filters. In fact, BM3D and WNNM perform successfully in noise suppression, but, as we will see in Figures 4.7f and 4.7g , the output is still blurred. Moreover,



$$\begin{pmatrix} 59 & 19 & 76 & 178 \\ 47 & 33 & 23 & 167 \\ 14 & 63 & 16 & 103 \\ 10 & 51 & 25 & 36 \end{pmatrix}$$

(b)

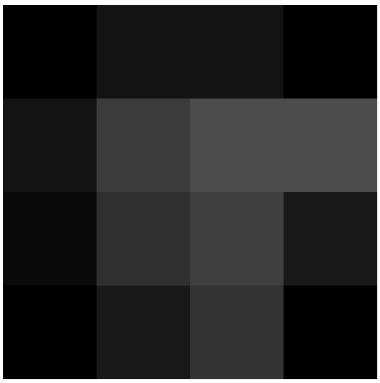
(a)



$$\begin{pmatrix} 59 & 19 & 76 & 178 \\ 47 & 0 & 23 & 167 \\ 255 & 63 & 255 & 103 \\ 10 & 51 & 25 & 255 \end{pmatrix}$$

(d)

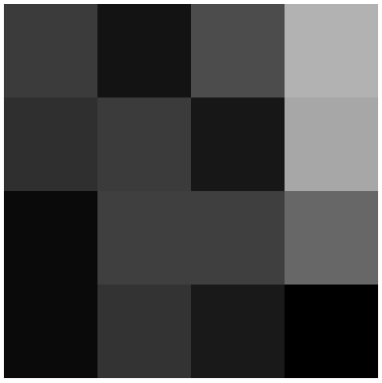
(c)



$$\begin{pmatrix} 0 & 19 & 19 & 0 \\ 19 & 59 & 76 & 76 \\ 10 & 47 & 63 & 25 \\ 0 & 25 & 51 & 0 \end{pmatrix}$$

(f)

(e)



$$\begin{pmatrix} 59 & 19 & 76 & 178 \\ 47 & 59 & 23 & 167 \\ 10 & 63 & 63 & 103 \\ 10 & 51 & 25 & 0 \end{pmatrix}$$

(h)

(g)

Figure 4.6: MI-SIR applied to Synthetic block: (a) original block of size 4x4 (c) original block corrupted by 20% of 'Salt pepper' noise (e) Restored block using the Median filter (g) Restored block using MI-SIR

Table 4.2: PSNR values of MI-SIR and comparison filters applied to test images degraded by the shift invariant blur and 15% of Gaussian noise

	σ=15%							
Methods	Degraded	Median	FOPDE	Wiener	TV	BM3D	WNNM	MI-SIR
Boat	17.32	19.38	18.81	19.52	5.44	20.64	21.35	21.63
House	17.56	20.01	19.73	20.12	5.91	20.90	21.49	21.77
Zelda	16.62	18.71	18.64	18.68	8.20	20.17	20.96	21.33
Barbara	17.01	19.19	18.83	19.20	6.48	20.91	21.45	21.60
Airplane	18.52	20.09	20.26	20.74	6.84	21.12	21.51	21.59
Man	16.43	18.05	17.43	18.13	7.73	19.94	20.31	20.66
Peppers	17.08	19.24	18.73	19.30	5.68	21.03	21.42	21.53
Mandrill	17.35	19.38	18.89	19.50	5.53	21.11	21.46	21.48
Cameraman	16.91	18.60	17.89	18.77	5.62	18.93	20.31	20.57
Stephen	16.73	18.61	18.07	18.66	7.90	20.16	20.63	20.71

Table 4.3: PSNR values of MI-SIR and comparison filters applied to test images degraded by shift invariant blur and 25% of Gaussian noise

	σ=25%							
Methods	Degraded	Median	FOPDE	Wiener	TV	BM3D	WNNM	MI-SIR
Boat	13.41	14.06	13.91	14.14	5.45	20.00	20.09	20.17
House	13.73	14.39	14.40	14.53	4.92	20.21	21.24	21.48
Zelda	13.20	13.53	13.48	13.50	8.21	20.17	21.13	21.31
Barbara	13.13	13.85	13.75	13.87	5.49	19.77	19.89	20.11
Airplane	15.20	15.38	15.53	15.97	8.84	20.83	21.26	21.69
Man	13.68	13.30	13.07	13.32	7.73	20.36	21.22	21.46
Peppers	13.33	14.00	13.89	14.08	5.68	20.21	20.88	21.38
Mandrill	13.43	14.11	13.98	14.17	5.53	20.02	20.13	20.16
Cameraman	13.14	13.72	13.50	13.81	5.62	19.83	19.98	20.08
Stephen	12.90	13.56	13.40	13.60	6.91	19.34	19.52	19.66

Table 4.4: PSNR values of MI-SIR and comparison filters applied to test images degraded by shift invariant blur and 35% of Gaussian noise

	σ=35% of noise							
Methods	Degraded	Median	FOPDE	Wiener	TV	BM3D	WNNM	MI-SIR
Boat	10.51	10.70	10.71	10.82	5.45	15.16	14.50	18.41
House	10.92	11.02	11.18	11.24	4.92	16.28	16.93	19.15
Zelda	9.86	10.21	10.20	10.22	8.22	14.73	15.68	18.61
Barbara	10.23	10.50	10.50	10.57	6.50	16.96	18.35	19.31
Airplane	12.80	12.56	12.78	13.02	2.84	15.19	16.12	18.98
Man	9.84	10.10	10.02	10.14	7.74	14.74	16.14	19.04
Peppers	10.52	10.68	10.74	10.84	5.69	15.28	15.91	18.72
Mandrill	10.52	10.69	10.74	10.84	5.54	14.16	16.78	18.87
Cameraman	10.38	10.49	10.50	10.66	5.63	14.41	15.28	18.26
Stephen	10.05	10.31	10.27	10.37	6.91	15.34	16.80	19.35

the restoration using these techniques are performed at the expense of smoothing effects. These Tables also report that our method produces for all the degraded images a higher PSNR than the other techniques. In fact, the selective property described by the decision rule protects perfectly a large number of original pixels, which in turn leads to a small MSE and higher PSNR.
In the next experiment, we support the previous results by a visual comparison. For this purpose, MI-SIR is applied to ”Barbara” test image degraded by a shift invariant blur Eq.(4.32) and $\sigma = 30\%$ of Gaussian noise (Figure 4.7).

At first glance, it seems that the BM3D outcome (Figure 4.7f) is the cleanest image in comparison with the other methods, but in terms of details protection, this technique along with WNNM and FOPDE generate a lower quality. These results also report that WNNM and FOPDE are too blurred due to the smoothness effects that tend to deteriorate the image features. We can also observe that filters as Weiner and Median filters (Figures 4.7c and 4.7d) have less influence regarding the smoothness effects, but their outcomes are still degraded. In Figure 4.7h, the proposed method provides an encouraging outcome against the other methods. In fact, our technique performs successfully in image reconstruction while preserving as mush the image details as possible .
For more illustration, Figure 4.8 shows the number of original and changed pixels in MI-SIR output, as well as its comparison with those of Median, Wiener, BM3D and WNNM. For this reason, these techniques are applied on "Peppers" test image at 60% of a random impulse noise.
In Figure 4.8, the number of original pixels in MI-SIR output is the higher in comparison with the other techniques. In fact, the selective technique allows preserving as much as possible the original pixels.In addition, the number of original pixels augments thanks to the pixels that represent the similarity between the original image and the solution of the model. Moreover, Figure 4.8 attests that the changed pixels in BM3D, Wiener, Median and

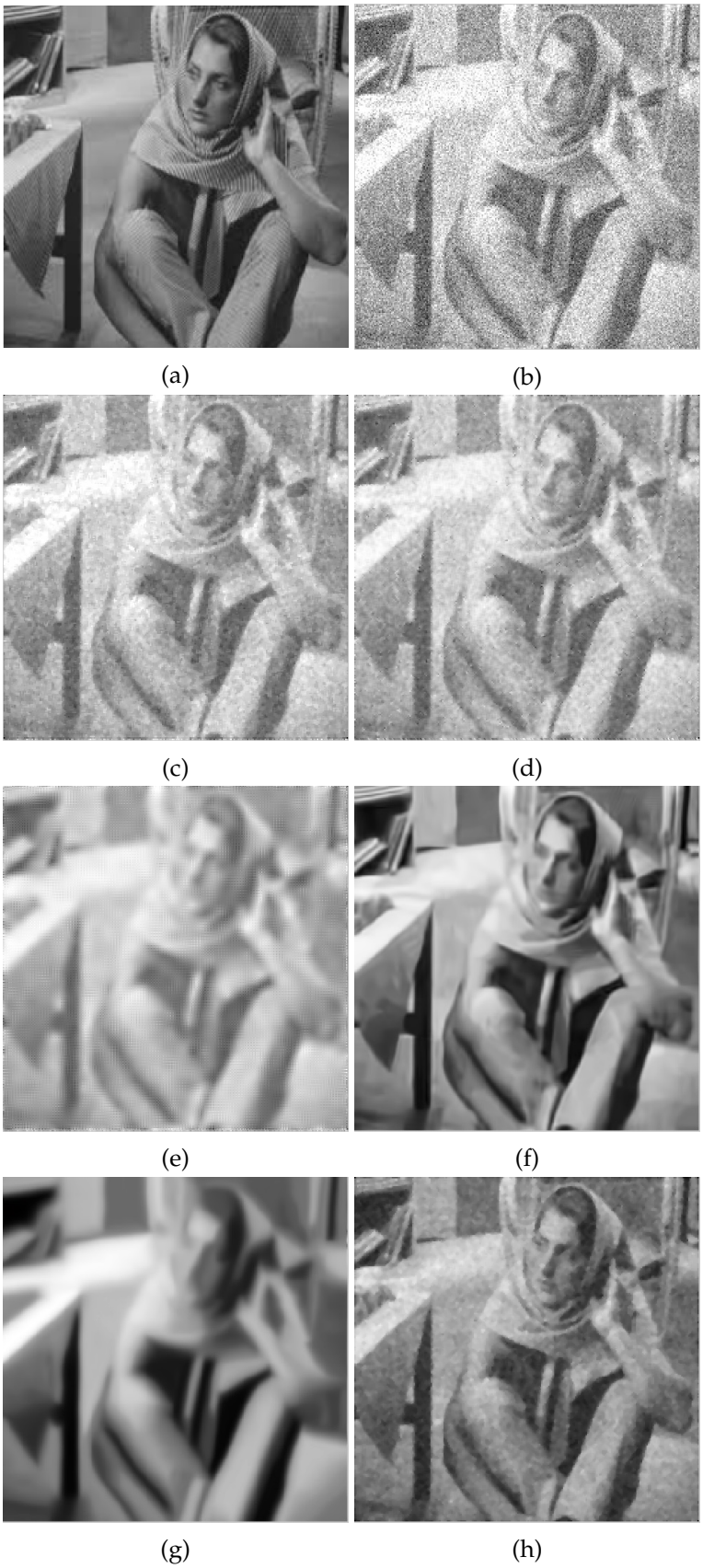


Figure 4.7: Outputs of "Barbara" test image using the different methods : (a) original (b) degraded image (c) Median (d) Wiener (e) FOPDE (f) BM3D (g) WNNM (h) MI-SIR

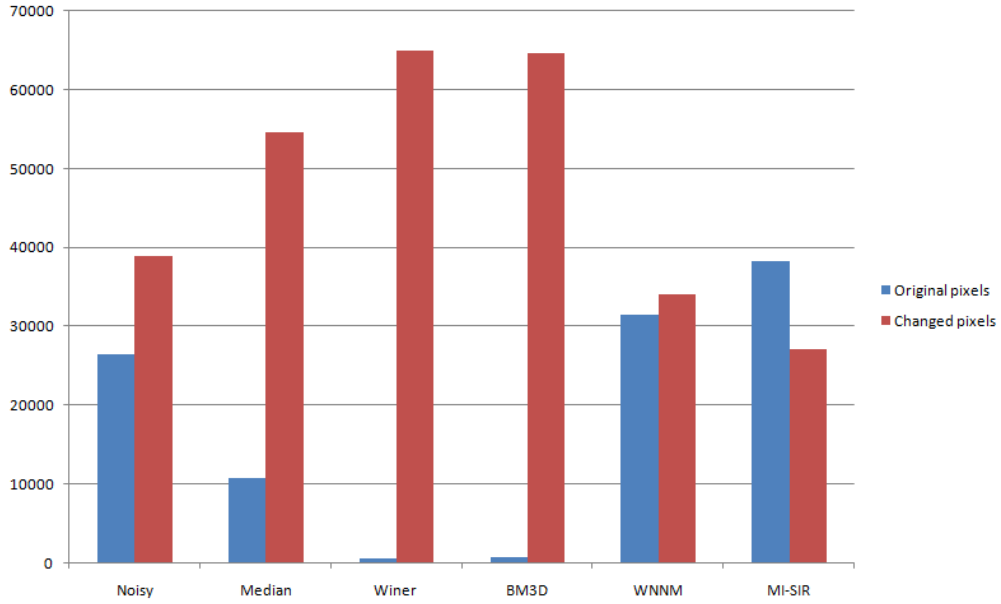


Figure 4.8: Number of original and changed pixels in the outputs of MI-SIR and comparison filters applied to "Peppers" test image at 60% of random impulse noise

relatively WNNM are larger than MI-SIR. Indeed, such techniques perform uniformly across images, which lead to reconstruct the images at the expense of original pixels.
In order to give a better illustration to the selective approach, we use in the next experiment a block of size 20×20. The suggested block assumes partially an impulse noise of type "Salt and pepper" with density $\rho = 70\%$ (Figures 4.9, 4.10 and 4.11).
By comparing the grey-level function of the both of the original and the degraded blocks, we observe, from the decision solution described in Figure 4.11, that the selective approach deals perfectly with details preservation. In fact, In free noisy area, the decision solution provides only eight false detections among 364 pixels. Moreover, in noisy area, our technique generates four wrong pixels among 36.

4.6 Conclusion

In this chapter, a new approach of image restoration called selective image restoration is introduced and analysed. The suggested model, which takes the form of a non linear mixed-integer model, may not only produce a solution that is close to the original image, but also supports the original features protection thanks to a suitable decision rule. Continuous Hopfield neural network and Genetic algorithm are performed progressively to solve the model, thus constituting an efficient algorithm that allows a significant preservation of the image details during the restoration process. Experimental results illustrate that the proposed MI-SIR method produces results superior to several approaches from the literature in terms of visual image quality and quantitative measures.

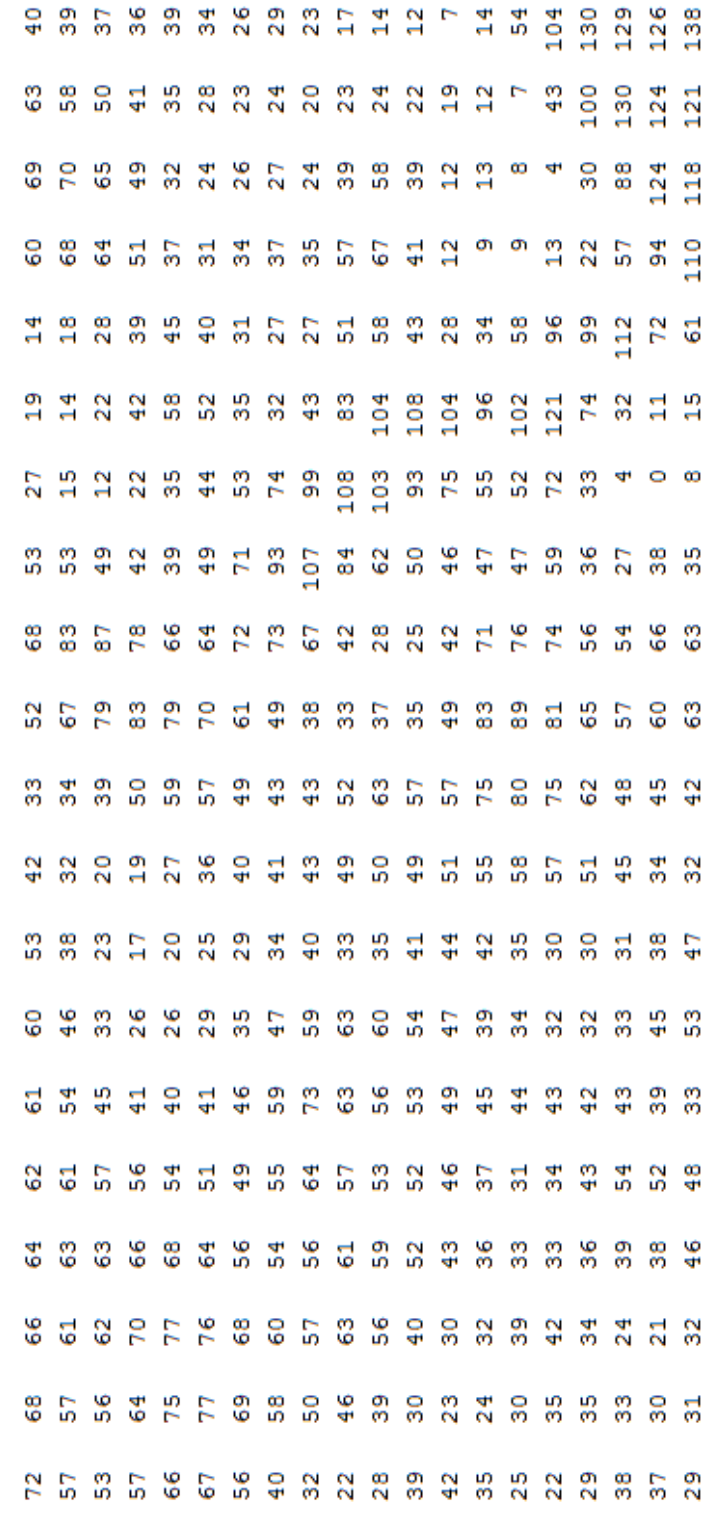


Figure 4.9: Gray level of an original block of size 20×20

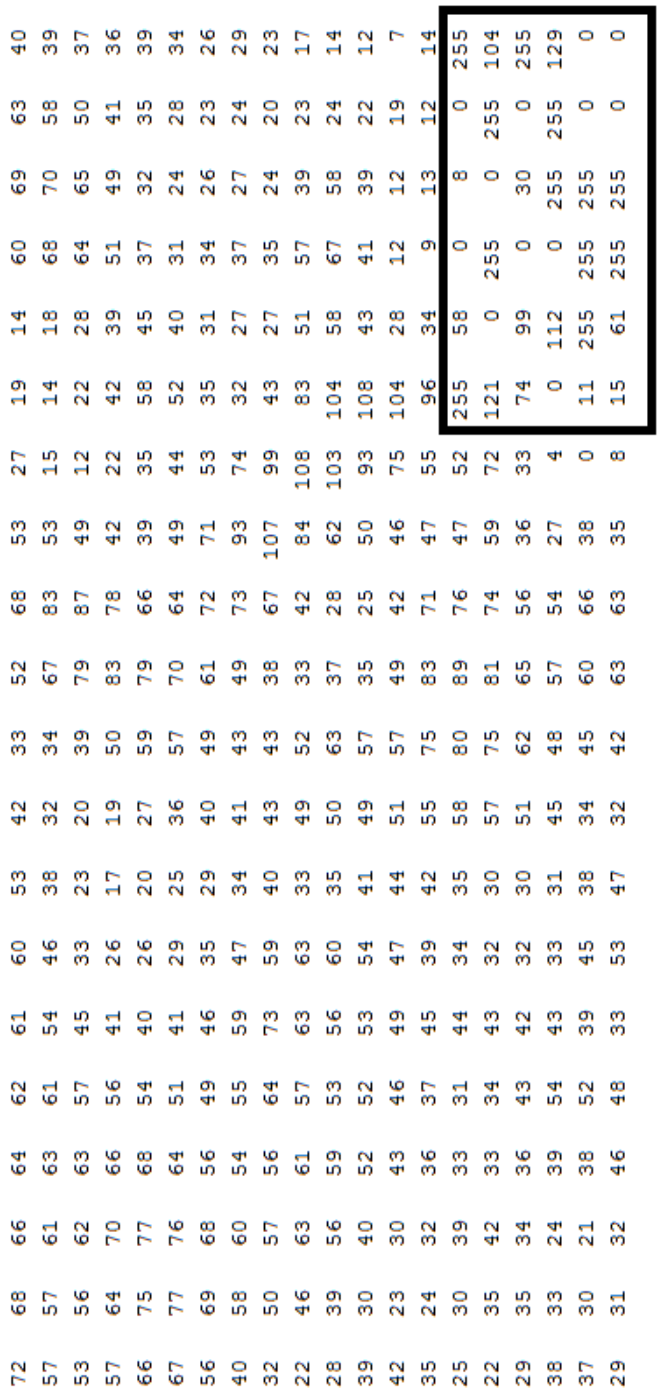


Figure 4.10: Gray level of the block (4.9) with a partial degradation at the last block of size 6×6

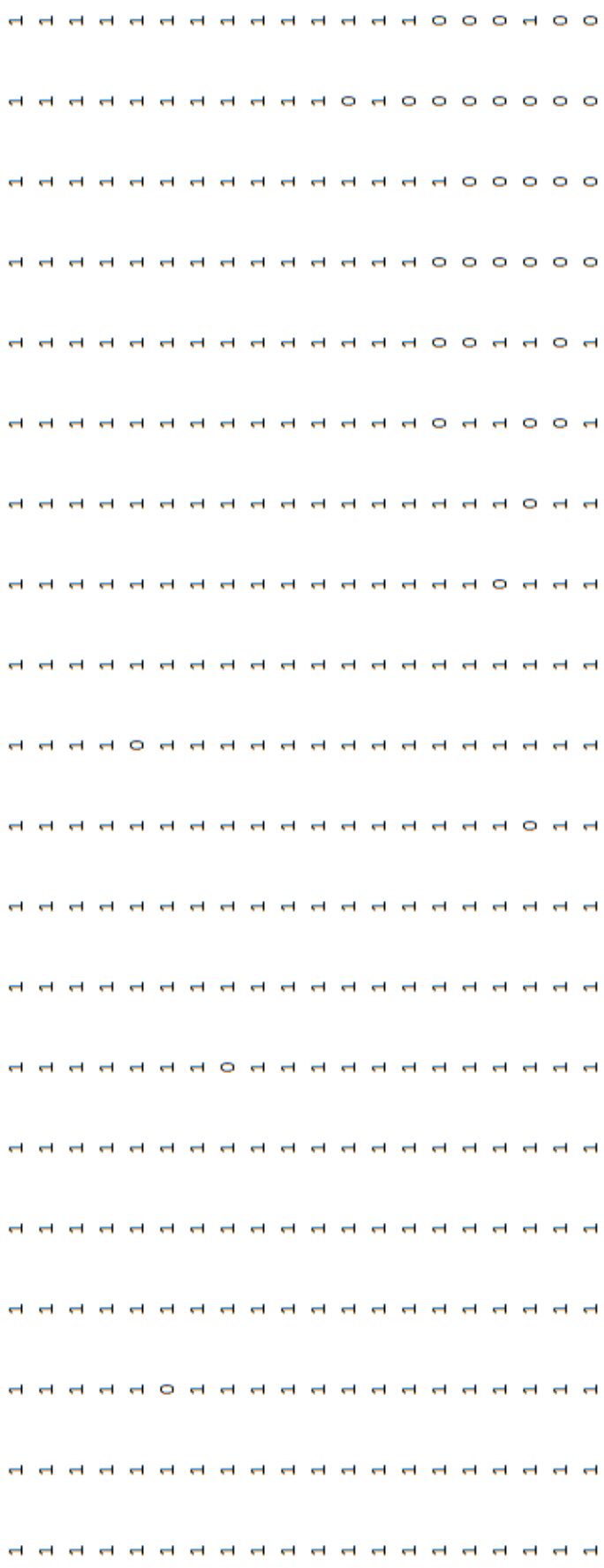


Figure 4.11: Decision solution corresponding to (4.10)

Chapter 5

Convergence and parameters estimation of CHN applied to a new architecture optimization model of probabilistic SOM: application to data clustering

5.1 Abstract

Probabilistic models have proven their strength to model many natural phenomena as close as possible to reality, in particular, the probabilistic self organizing map (PRSOM) that belongs to the unsupervised learning models. It allows to provide an estimation of the probability density function through the likelihood maximization. This function depends on several parameters given by the model architecture. In this context, the aim of this chapter is to deal with the architecture choice problem that consists in determining the optimal number of components needed for a better performance. In this chapter, we propose a new optimization model that describes the problem above. We present the architecture of PRSOM in a mathematical system of a non linear objective function with mixed variables under linear and quadratic constraints. Due to the complexity of the resolution, we suggest the continuous Hopfield neural network (CHN) that we support by a deep stability analysis. Performance of the proposed model is demonstrated through the data clustering.
keywords: Artificial neural networks; self organizing map; Hopfield neural network; probabilistic self organizing map; continuous Hopfield neural network; Mathematical programming; mixed-variables non linear problem.

5.2 Introduction

The mixture models have became an interesting statistical tool for scientists in both theoretical and practical domains, thanks to the raising of many application areas in the real life. These applications have provided a huge number of databases characterized by their

heterogeneity, i.e. they come from sources containing many subpopulations. As a result, we could not model the totality of these observations by only one classical probability distribution. Therefore, we model each subpopulation in a separate way, by using a finite mixture of probability distributions[**75**].
In the present chapter, we study the probabilistic self organizing map (PRSOM) as one of the most useful statistical models for the continuous data mixture [**5**]. Actually, the probabilistic self organizing map is an improved version of the Self Organizing Map (SOM) introduced by the scientist Teuvo Kohonen in the late 1990s [**60**]. This model, that is used for the vector quantization of the input space, gives an approximation of the probability density function of this space, which refers to maximize the likelihood function of a set of samples. This likelihood is a Gaussian mixture depending on parameters (weights and covariance matrix) estimated by a specific learning algorithm. Therefore, this estimation depends on the parameters given by the model's architecture [**32**, **33**].
Despite its efficiency to provide a better estimation of the density distribution for data mixture, PRSOM has disadvantages, notably, the dependence of this density on the choice of components number of the used mixtures. The random choice of many components can lead to week solutions. i.e. choosing a high number of neurons could lead to an overlearning and produces poor interpretations. Whereas choosing a few number of neurons could lead to a bad approximation of the data structure. Talking about the number of neurons constituting the map is equivalent to talking about the architecture choice problem. In fact, this problem has not only a significant impact on the convergence, but it also affects the quality of the obtained results [**32**, **33**].
The main purpose of this work is modelling the problem of the PRSOM architecture using a mixed-integer nonlinear problem with linear and quadratic constraints. Once the model is built, the important coming step is its resolution. For the reason of the model complexity, we opt for the heuristic approach, in which use the continuous Hopfield neural network (CHN). [**47**, **48**].
This work is organised as follows: Section 5.3 describes the probabilistic self organizing map. The proposed optimization model of PRSOM architecture is described in Section 5.4. Resolution of PRSOM by CHN, which was later called PRSOM-CHN, is discussed in Section 5.5. Finally, experimental results are reported and discussed in Section 5.6. Conclusions, based on the present study, are finally drawn in Section 5.7.

5.3 Probabilistic self-organizing map model

In the real research domains, we rarely find observations that could be modeled by only one classical probability distribution. The majority of observed data $\{x_1, ..., x_n\} = D \subset \mathbf{R}^d$ comes from sources that contain many sub-populations modeled each in a different way. Seeing that the entire population is a mixture of these sub-populations, we consequently model it by using a finite mixture of probability distributions [**18**, **75**], such as the probabilistic self-organizing map.
In order to define the PRSOM model, Badran and al. propose to associate each neuron k of the map with a Gaussian density function f_k [**5**, **67**]. Furthermore, the neighborhood notion allows introducing a mixture of Gaussians. These two concepts allow to introduce

the density function defined as a mixture of probabilistic mixtures where each neuron of the map is assigned to a component of the mixture (Figure 5.1).

Probabilistic self-organizing map model is used to approach distribution density of the

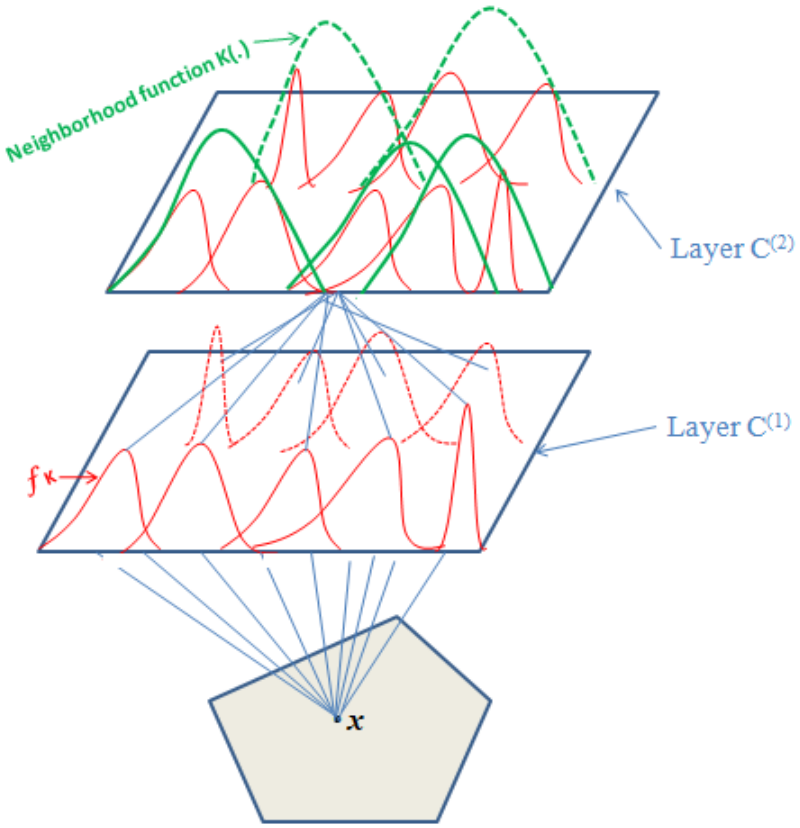


Figure 5.1: Probabilistic self organizing map model

observations of a set D of elements that are supposed to be i.i.d. using a mixture of normal probability densities:
$\forall \theta = (W, \Sigma)$

$$L(\theta) = p(x_1, ..., x_n, W, \Sigma) = \prod_{i=1}^{n} \sum_{j=1}^{K} p(n_j^{(2)}) p(x_i | n_j^{(2)}) \tag{5.1}$$

With:

- $p(n_j^{(2)})$: the initial probability of each neuron in the map $C^{(2)}$.
- $p(x_i | n_j^{(2)})) = \sum_{k=1}^{K} p(n_k^{(1)} | n_j^{(2)}) p(x_i | n_k^{(1)})$: a mixture of probabilities for the conditional generation of the observation x_i.
- $p(n_k^{(1)} | n_j^{(2)}) = \frac{K_T(\delta(n_k^{(1)}, n_j^{(2)}))}{\sum_{k=1}^{K} K_T(\delta(n_k^{(1)}, n_j^{(2)}))}$: the probability that generates the neighborhood function between the neurons of the map (the influence of the map $C^{(1)}$ neurons on the map $C^{(2)}$ neurons).

In sum, this model is characterized by the two sets $W = \{w_1, ..., w_K\}$ and $\Sigma = \{\Sigma_1, ..., \Sigma_K\}$ that have to be estimated during the network's learning phase. We use the maximization estimator of the likelihood function given by the equation(1), which is equivalent to estimate the optimal parameters (W^*, Σ^*) that maximize this likelihood **[26, 71]**. The learning is performed using the two following phases: assignation phase and maximization phase **[5]**.

5.4 New model for probabilistic self organizing map

To overcome the problem of architecture choice, we propose in this chapter a new mathematical model of the probabilistic self organizing map that controls the size of the map. In this section, we will describe the construction steps of our model. The first one consists of integrating the special term which controls the size of the map. The second step gives in a first time the affectation constraints that ensure the allocation of each data to only one neuron (component), and in a second time it gives another constraint that ensures the communication between the affectation and optimization variables.
The principal idea is modelling the PRSOM architecture as an optimization mathematical program represented by a mixed-integer non-linear objective function with linear and non-linear constraints. The components of the model are defined as follows:
Parameters:

- n : number of observation data;
- d : Vector dimension of observation data;
- N : Optimal number of neurons (components) in the topological map of PRSOM;
- N_{max}: Maximal number of neurons in the topological map of PRSOM.

-Variables:

- D : Matrix of Training base elements;
- W : Matrix of referent vectors;
- Σ : Matrix of covariance;
- v : Matrix of binary variables;
  where $v_{i,j}$, is the assignment variable that define the relationship between data and neuron, and $v_{1,j}$ denotes the control variable that allows controlling the size of PRSOM map.

-Objective function:
Basing on the work of Bishop for the Gaussian mixture model **[80]**, we define the objective function of the probabilistic self-organizing as follows:

$$Max \;\; p(W, \Sigma, v) = \prod_{i=2}^{n+1} \prod_{j=1}^{N_{max}} (\pi_j \sum_{k=1}^{N_{max}} \delta(n_k^1, n_j^2) f_k(x_{i-1}, w_k, \sigma_k))^{v_{i,j} v_{1,j}} \tag{5.2}$$

Given the strong non-linearity of the likelihood, we use the log-likelihood to simplify the formula of $p(W, \Sigma, v)$, we obtain an equivalent problem:

$$Max \;\; log[p(W, \Sigma, v)] = \sum_{i=2}^{n+1} \sum_{j=1}^{N_{max}} v_{i,j} v_{1,j} log[(\pi_j \sum_{k=1}^{N_{max}} \delta(n_k^1, n_j^2) f_k(x_{i-1}, w_k, \sigma_k))] \tag{5.3}$$

The research for a maximum can always be transformed to the research of a minimum. The objective function is thus defined as follows:

$$Min \;\; E(W, \Sigma, V) = - \sum_{i=2}^{n+1} \sum_{j=1}^{N_{max}} v_{i,j} v_{1,j} log[(\pi_j \sum_{k=1}^{N_{max}} \delta(n_k^1, n_j^2) f_k(x_{i-1}, w_k, \sigma_k))] \tag{5.4}$$

-Constraints:
Each data element must be allocated to one neuron (component). In consequence we obtain the following n constraints called assignment constraints :

$$\sum_{j=1}^{N_{max}} v_{i,j} = 1 \quad \forall i \in \{2, ..., n+1\} \tag{5.5}$$

Besides assignment constraints, we add another one called transmission constraint. If the neuron j is not used $v_{1,j} = 0$, i.e., $\sum_{i=2}^{n+1} v_{i,j} = 0$.

$$\sum_{j=1}^{N_{max}} (1 - v_{1,j}) \sum_{i=2}^{n+1} (v_{i,j}) = 0 \tag{5.6}$$

Using Eqs.(5.4), (5.5) and (5.6), the new PRSOM model is given by:

$$(P) = \begin{cases} MinE(W, \Sigma, v) = \sum_{i=2}^{n+1} \sum_{j=1}^{N_{max}} v_{i,j} v_{1,j} C_{i-1,j} \\ \sum_{j=1}^{N_{max}} v_{i,j} = 1 \quad \forall i \in \{2, ..., n+1\} \\ \sum_{j=1}^{N_{max}} (1 - v_{1,j}) \sum_{i=2}^{n+1} (v_{i,j}) = 0 \\ w_j \in IR^d \quad \forall j \in \{1, ..., N_{max}\} \\ \sigma_j \in IR^+ \quad \forall j \in \{1, ..., N_{max}\} \\ V \in \{0, 1\}^{(n+1)N_{max}} \end{cases}$$

(5.7)

Where $C_{i-1,j} = -log[(\pi_j \sum_{k=1}^{N_{max}} \delta(n_k^1, n_j^2) f_k(x_{i-1}, w_k, \sigma_k))]$ and $\Sigma_j = \sigma_j * I_d$

5.5 Using CHN for solving PRSOM

As shown from the Eq.(5.7), the proposed model belongs to the class of mixed-integer non linear programming problems (MINLP). The suggested problem has an objective function that is neither continuous nor differentiable, with linear and quadratic constraints. Therefore, it is very difficult if not impossible to solve (P) Exactly. For this reason, we propose one of the technical approaches that have been widely used for solving MINLP, it consists of two principal phases:

— Assignment phase: we fix the continuous variables and we solve the obtained problem (binary quadratic problem with constraints) using the continuous Hopfield neural network. Indeed, CHN allows solving combinatorial problems thanks to its energy minimization ability. In addition, unlike the discrete systems including DHN, the continuous model of Hopfield is more efficient in discussing the parameters of the hopfield energy, due largely to its stability analysis **[56]**.

— Minimization phase: we fix the obtained assignment vectors and we solve the continuous problem (non linear optimization model with continuous variables).

5.5.1 Assignment phase

Since the resolution is performed sequentially using the two phases: expectation and minimization, we fix the obtained W and Σ at the iteration $t-1$ and we solve the binary optimization system (P_1) with decision variables using CHN.

$$(P_1) = \begin{cases} \underset{v\in\{0,1\}^{(n+1)N_{max}}}{min} E(.,.,v) = \sum_{i=2}^{n+1}\sum_{j=1}^{N_{max}} v_{1j}v_{i,j}C_{i-1,j} & \\ e_i(v) = \sum_{j=1}^{N_{max}} v_{i,j} = 1 & i = 2,\dots,n+1. \\ H(v) = \sum_{j=1}^{N_{max}}((1-v_{1j})\sum_{i=2}^{n+1} v_{i,j}) = 0 & \end{cases} \tag{5.8}$$

5.5.1.1 Proposed CHN-map for PRSOM

Firstly, we use the penalty function approach to build a new energy function associated with (P_1) **[114]**. Thus, the energy function E is transformed to:

$$E(v) = \frac{\alpha}{2}\sum_{i=2}^{n+1}\sum_{j=1}^{N_{max}} v_{1,j}v_{i,j}C_{i-1,j} + \frac{\rho}{2}H(v) + \frac{\beta}{2}\sum_{i=2}^{n+1}(e_i(v)-1)^2 + \frac{\gamma_1}{2}\sum_{j=1}^{N_{max}}\sum_{i=1}^{n+1} v_{i,j}(1-v_{i,j}) \tag{5.9}$$

where the scaling parameters α, β, ρ and γ_1 are real parameters that are settled by the network convergence analysis. In Eq.(5.9), the first term represents the objective function to be minimized, constraints are penalized by the second and third terms, the last term is imposed to keep the (0–1) variables feasible.
Afterwards, we map the PROSM model into a CHN by proposing a network with $(n+1)\times N_{max}$ neurons mutually interconnected. The suggested network is characterized by:

— The set of the network state:
$v = \{ v_{i,j} \ / \ i = 1,..,n+1 \quad and \quad j = 1,..,N_{max}\}$.

$$v = \begin{bmatrix} v_{1,1} & .. & v_{1,j} & .. & v_{1,N_{max}} \\ \hline v_{2,1} & .. & v_{2,j} & .. & v_{2,N_{max}} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ v_{i,1} & .. & v_{i,j} & .. & v_{i,N_{max}} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ v_{n+1,1} & .. & v_{n+1,j} & .. & v_{n+1,N_{max}} \end{bmatrix}$$

— The weight matrix:
$T = \{T_{ij,kl} \ / \ i,k = 1,..,(n+1) \quad and \quad j,l = 1,..,N_{max}\}$

The first row of v denotes the variables that are responsible for the PRSOM architecture while the rest represent the assignment variables. By comparing Eqs.(5.9) and (1.31) and by neglecting the constant term, we obtain the system of the network parameters:

$$Map = \begin{cases} T_{1b,1d} = \gamma_1 \delta_{bd} \\ T_{ab,cd} = -\beta + \gamma_1 \delta_{ac} \delta_{bd} \\ T_{1b,cd} = -(\alpha C_{c-1,b} - \rho)\delta_{bd} \\ I_{1j} = -\frac{\gamma_1}{2} \\ I_{ab} = -\frac{\rho}{2} + \beta - \frac{\gamma_1}{2} \end{cases} \tag{5.10}$$

where $a,c = 2,..,n+1$, $b,d = 1,..,N_{max}$ and δ_{ij} is equal to 1 (i=j) and 0 (otherwise).

5.5.1.2 Stability analysis and parameters estimation

As mentioned previously, there are tree important set that we need to study in the stability analysis. In the two last chapter, there was no constraints on the decision variables which thereafter makes the corner set a feasible solution set. Unfortunately, this is not the situation in this case.
In the proposed model, a feasible solution is guaranteed by the CHN from the non linear stability analysis of the hypercube corners set: $H_c = \{0,1\}^{(n+1)N_{max}}$.

The set of feasible solutions is formed by $H_F = (H_{F1} \cup H_{F2}) \cap H_{F3}$ where:

$$H_{F1} = \{v \in \{0,1\}^{(n+1)N_{max}} \ / \ v_{1,j} = 1 \ and \ \sum_{i=2}^{n+1} v_{i,j} \geq 1\} \tag{5.11}$$

$$H_{F2} = \{v \in \{0,1\}^{(n+1)N_{max}} \ / \ v_{1,j} = 0 \ and \ \sum_{i=2}^{n+1} v_{i,j} = 0\} \tag{5.12}$$

and

$$H_{F3} = \{v \in \{0,1\}^{(n+1)N_{max}} \ / \ \sum_{j=1}^{N_{max}} v_{i,j} = 1 \ \forall i = 2,..,n+1\} \tag{5.13}$$

Indeed, we can rewrite H_F as follows:

$$\begin{aligned} H_F &= (H_{F1} \cup H_{F2}) \cap H_{F3} \\ &= (H_{F1} \cap H_{F3}) \cup (H_{F2} \cap H_{F3}) \end{aligned}$$

where the set $(H_{F1} \cap H_{F3})$ means that, form one hand, if the j^{th} neurone of the architecture variables is activated $(v_{1,j} = 1)$, then at least one of its affectation neurons ($v_{i,j}$, $\forall i = 2, ..n+1$) is activated (i.e. $\sum_{i=2}^{n+1} v_{i,j} \geq 1$). From the other hand, the affectation constraint is satisfied thanks to H_{F3}. On the basis of a similar reasoning, we interpret the set $(H_{F2} \cap H_{F3})$.
Convergence of feasible solutions is ensured from the partial derivatives of E which are expressed by:

$$E_{1,b}(v) = \frac{\partial E(v)}{\partial v_{1,b}} = \frac{\alpha}{2} \sum_{i=2}^{n+1} v_{i,b} C_{i-1,b} + \frac{\gamma_1}{2}(1 - 2v_{1,b}) - \frac{\rho}{2} \sum_{i=2}^{n+1} v_{i,b} \tag{5.14}$$

$$E_{a,b}(v) = \frac{\partial E(v)}{\partial v_{a,b}} = \frac{\alpha}{2} v_{1,b} C_{a-1,b} + \frac{\rho}{2}(1 - v_{1,b}) + \beta(e_a(v) - 1) + \frac{\gamma_1}{2}(1 - 2v_{a,b}) \tag{5.15}$$

$\forall\ (a, b) \in \{2, .., n+1\} \times \{1, .., N_{max}\}$.
To minimize the first term of Eq.(5.9), we impose the following condition:

$$\alpha \geq 0 \tag{5.16}$$

Eqs.(5.17), (5.18), (5.19) and (5.20) are obtained from the equilibrium point criteria applied to $v \in H_F$:

$$v_{1,b} = 0 \Longrightarrow \frac{\gamma_1}{2} \geq 0 \tag{5.17}$$

$$v_{a,b} = 0 \Longrightarrow \begin{cases} \alpha C_{a-1,b} + \gamma_1 \geq 0 & \text{if } v_{1,b} = 1 \\ \rho + \gamma_1 \geq 0 & \text{if } v_{1,b} = 0 \end{cases}$$

which can be covered by:

$$v_{a,b} = 0 \Longrightarrow \gamma_1 \geq max\{-\alpha m, \rho\} \tag{5.18}$$

with $m = \min_{\forall (a,b) \in \{2,..,n+1\} \times \{1,..,N_{max}\}} C_{i-1,b}$.
In the case where $v_{1,b} = 1$, there exist $k \in \{1, .., n\}$ such that $\sum_{i=1}^{n} v_{i+1,b} = k$, therefore, we have:

$$v_{1,b} = 1 \Longrightarrow \alpha k C_{i,b} - k\rho - \gamma_1 \leq 0,$$

By maximizing:

$$v_{1,b} = 1 \Longrightarrow \alpha n M - \rho - \gamma_1 \leq 0, \qquad \rho > 0 \tag{5.19}$$

with $M = \max_{\forall(a,b)\in\{2,..,n+1\}\times\{1,..,N_{max}\}} C_{i-1,b}$.
The last condition is given by:

$$v_{a,b} = 1 \Longrightarrow \alpha M - \gamma_1 \leq 0 \tag{5.20}$$

Thus, we summarize all the conditions in the following system:

$$S_1 = \begin{cases} \alpha, \rho, \gamma_1 \geq 0 \\ \gamma_1 \geq max\{-\alpha m, \rho\} \\ \alpha n M - \rho - \gamma_1 \leq 0 \\ \alpha M - \gamma_1 \leq 0 \end{cases} \tag{5.21}$$

Given the size of the description space and the size of the PRSOM model. the feasible parameters are obtained by drawing the system S_1.
To ensure that the network does not converge to invalid solution(spurious steady states[**105**]), all these states need to be suppressed for both cases $H_c - H_F$ and $H - H_F$.
Divergence of non valid corner solutions:
Before going any further, we give the following definition:

Definition 13

A vertex point $A_{i,j}(v) \in H_C$ is said an adjacent of v if and only if:

$$\mathrm{A}_{i,j}(v_{k,l}) = \begin{cases} v_{k,l}, & if \quad (k,l) \neq (i,j) \\ 1 - v_{k,l}, & if \quad (k,l) = (i,j) \end{cases}$$

The adjacent vertices set of v is defined by $A = A^+ \bigcup A^-$, where:

$$A^+ = \{A_{i,j}(v) \in / v_{i,j} = 0\}$$

$$A^- = \{A_{i,j}(v) \in / v_{i,j} = 1\}$$

Using the adjacency concept, an invalid solution v can be identified through the Lemma 5.1.

Lemma 5.1

Let $v \in H_C - H_F$. v^+ is an invalid solution if and only if:

- $\exists i \in \{2, .., n+1\}$ such that $\sum_{j=1}^{N_{max}} v_{i,j} \geq 2$
- $\{e_i(v) = 1, \forall i = 2, .., n+1\} \cap \{\exists! j = 1, ..N_{max} \ / \ v_{1,j} = 1 \text{ and } \sum_{i=2}^{n+1} v_{i,j} = 0\}$

Proof. Let v^+ be an adjacent point defined by changing $v_{a,b}$ from 0 to 1, and let suppose that the i^{th} array is violated such that $\sum_{j=1}^{N_{max}} v_{i,j} \geq 2$.

- regarding the first assertion, two cases that must be discussed: if $(a = i)$, then, $\sum_{j=1}^{N_{max}} v_{i,j} \geq 3$, as consequence v^+ is an invalid solution (see Figure 5.2a). Now, if $(a \neq i)$, v^+ is still an invalid solution since $e_i(v)$ is not changed(Figure 5.2b).

- In the second assertion, we have one column where the first element is equal to 1 and the others are equal to 0. So, any change of zeros which are situated on the column will make an invalid solution by violating $e_i(v) = 1$(Figure 5.2c).

□

The Lemma 5.2 is necessary for the non convergence of invalid corners set.

Lemma 5.2

Let $\{\zeta_k\}_{k\in 1,2,.....,(n+1)N_{max}-1}$ and $\{\Psi_k\}_{k\in 1,2,.....,(n+1)N_{max}}$ the numerical sequences defined by:

$$\zeta_k = \max_{v\in H_C - H_F / \sum_{i=1}^{n+1}\sum_{j=1}^{N_{max}} v_{i,j}=k} \overset{-}{E}^{0}(v)$$

and

$$\Psi_k = \min_{v\in H_C - H_F / \sum_{i=1}^{n+1}\sum_{j=1}^{N_{max}} v_{i,j}=k} \overset{-}{E}^{1}(v)$$

If α, β, ρ and γ_1 are reel parameters verifying that:

$$S_2 = \begin{cases} \alpha m - \rho \geq 0 \\ \beta - \gamma_1 \geq 0 \\ -\gamma_1 \geq 0 \qquad \alpha, \rho, \beta \geq 0 \end{cases} \tag{5.22}$$

Then, the sequences ζ_k and Ψ_k are respectively increasing and decreasing.

Proof. Since v is one of the invalid solutions, there exist a row a and column b such that $v_{a,b} = 0$. using the right adjacency concept at the level of $v_{a,b}$, we obtain $v^{+}_{a,b} = 1$ and we distinguish the following cases:

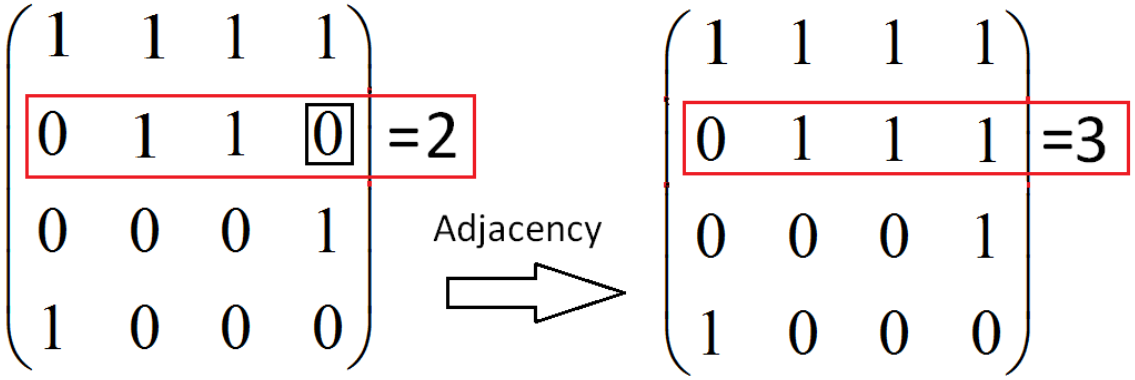
- **Case 1**: { (i $\neq$ 1)}:
  1. if $v_{a,b}$ is on the same array of $v_{i,j}$, then, $E_{i,j}(v)$ is increased subject to $\beta \geq 0$ that ensures the increasing of the third term in (5.15).
  2. if $v_{a,b}$ is situated in any position of v except the first array, then, $E_{i,j}(v)$ remains the same.
  3. in the case where $v_{a,b}$ is one of the first array, we have:

$$\begin{cases} \text{No change in the energy is observed,} & \text{if } b \neq j \\ \text{E}_{i,j}(v) \text{ is increased} \Longleftrightarrow \alpha m - \rho \geq 0, & \textit{otherwise} \end{cases}$$

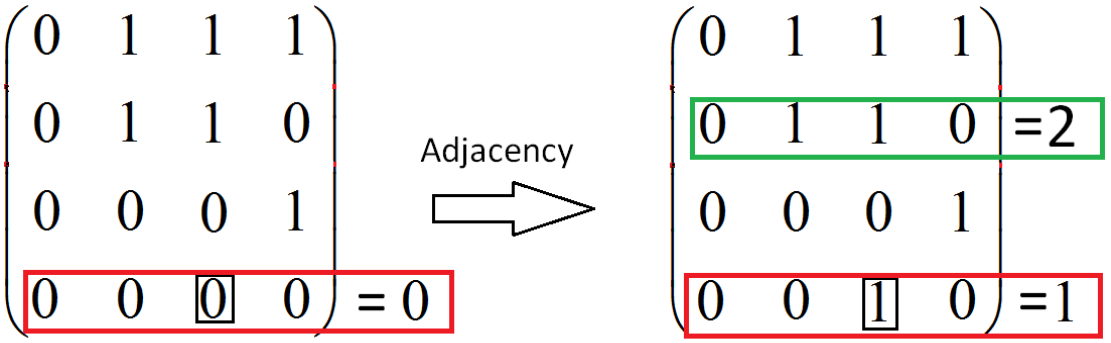
  4. if $v_{a,b} = v_{i,j}$, then $E_{i,j}$ is increased if and only if

$$\beta - \gamma_1 \geq 0 \tag{5.23}$$

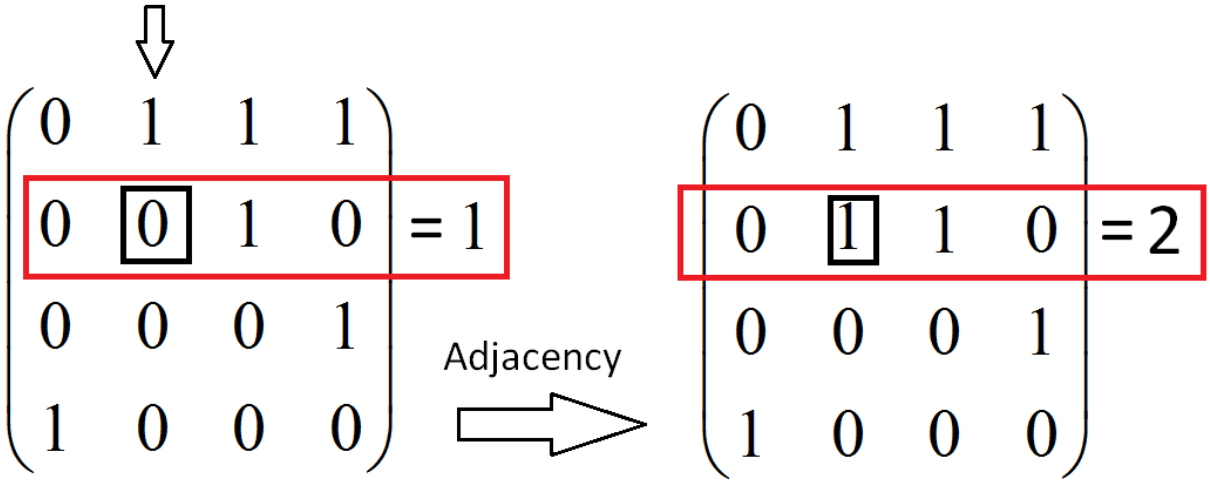
- **Case 2**: $\{i = 1\}$:
  In this case, $E_{1,j}(v)$ does not change, except if $v_{a,b}$ is one of the j^{th} column, in such case, we have:



(a) Adjacency transformation applied to a violated row



(b) Adjacency transformation applied to a row of zeros



(c) Adjacency transformation applied to a violated column

Figure 5.2: Adjacency transformation for obtaining an invalid solution

$$a \neq 1 \Longrightarrow \alpha m - \rho \geq 0) \tag{5.24}$$

$$a = 1 \Longrightarrow -\gamma_1 \geq 0 \tag{5.25}$$

Therefore, the satisfaction of Eqs.(5.23), (5.24) and (5.25) leads to deduce for each $v \in H_C - H_F$:

$$E^0(v^+) \geq E^0(v)$$

$$E^0(v^+) \geq \underset{v \in H_C - H_F / \sum_{i=1}^{n+1} \sum_{j=1}^{N_{max}} v_{i,j} = k}{max} E^0(v)$$

$$\underset{v \in H_C - H_F / \sum_{i=1}^{n+1} \sum_{j=1}^{N_{max}} v_{i,j} = k+1}{max} E^0(v) \geq E^0(v^+) \geq \underset{v \in H_C - H_F / \sum_{i=1}^{n+1} \sum_{j=1}^{N_{max}} v_{i,j} = k}{max} E^0(v)$$

Hence,

$$\zeta_{k+1} \geq \zeta_k \tag{5.26}$$

Finally, ζ_k is increasing sequence.
Analogously, based on Figure 5.3, we prove that Ψ_k is decreasing sequence

□

Using Lemma 5.2, the following theorem gives sufficient conditions to avoid the equilibrium of invalid corner points.

Theorem 5.1

For any binary invalid solution v. If α, β, ρ and γ_1 are reel parameters verifying:

$$S_2 = \begin{cases} \alpha m - \rho \geq 0 & \alpha, \rho \geq 0 \\ \beta - \frac{\gamma_1}{2} \geq 0 \\ -\gamma_1 \geq 0 \\ \beta \geq 0 \end{cases} \tag{5.27}$$

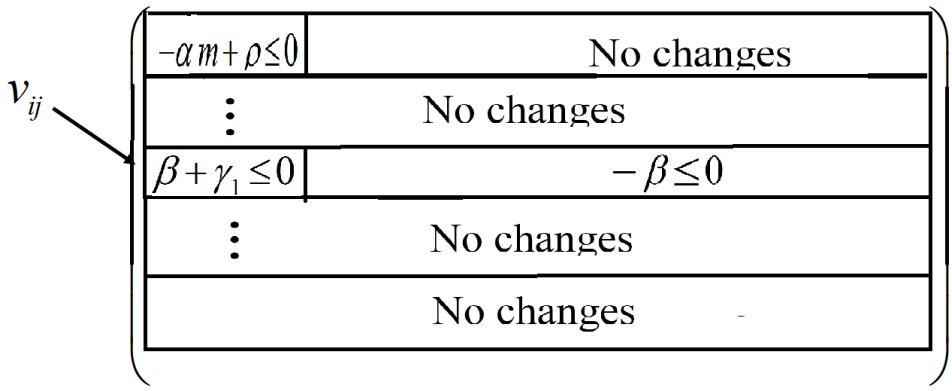
and for each invalid solution $V \in H_C - H_F$ such that:

$$\underset{v \in H_C - H_F / \sum_{i=1}^{n+1} \sum_{j=1}^{N_{max}} v_{i,j} = (n+1)N_{max} - 1}{max} E^0(v) < 0 < \underset{v \in H_C - H_F / \sum_{i=1}^{n+1} \sum_{j=1}^{N_{max}} v_{i,j} = (n+1)N_{max}}{min} \bar{E^1}(v) \quad . \tag{5.28}$$

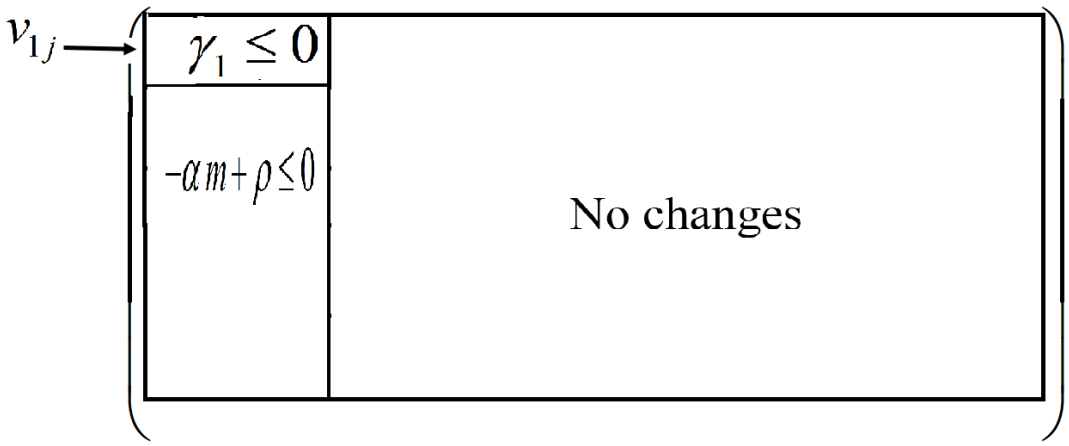
Then, v can not be an equilibrium point.

Proof. As stated from Lemma 5.2, ζ_k (Ψ_k) is increasing (decreasing) sequence under the terms agreed with (5.22). which means obviously that $\zeta_k \leq \zeta_{(n+1)N_{max}-1}$ $\forall k \in \{1, 2, .., (n+1)N_{max} - 1\}$ ($\Psi_k \geq \Psi_{(n+1)N_{max}}$ $\forall k \in \{1, 2, .., (n+1)N_{max}\}$) and by breaching the conditions of an equilibrium point, we obtain the result (5.28).

□



(a) E_{ij} variation according to $v_{a,b}$ positions



(b) E_{1j} variation according to $v_{a,b}$ positions

Figure 5.3: Energy variation according to $v_{a,b}$ positions (Case of $v_{ij} = 1$) .

Donvergence of the non valid interior points $v \in H - H_C$**:**
Firstly, we recall the following definitions:

Definition 14

A point $v \in H$ is said an interior point if and only if :
$\exists\, v_{i,j} \in v \;/\; v_{i,j} \in \,]0,1\,[]$

Definition 15

Let $v \in H$ be an interior point, v defines an edge point if and only if:
$card\{(i,j) \in \{1,..,n+1\} \times \{1,..,N_{max}\} /\;\; v_{i,j} \in \,]0,1[\} = 1$

Definition 16

Let $v \in H$ be an edge point such that $v_{a,b} \in \,]0,1\,[]$, v' is called the adjacent corner of v if:

$$v'_{i,j}=\begin{cases} v_{i,j} \text{ if } (i,j) \neq (a,b) \\ 1 \text{ or } 0 \text{ if } (i,j) = (a,b) \end{cases}$$

As mentioned in **[83]**, interiors points not being edge points have a null probability of being the convergence point of the CHN. In this context, the divergence of an edge point is ensured through the following theorem:

Theorem 5.2

Let the edge point $\{v \in H - H_C \;/\; v_{i,j} \in \,]0,1\,[\}$, if $\gamma_1 < 0$, $\beta \geq 0$ and v' is the adjacent corner of v verifying

$$E_{i,j}(v') \begin{cases} \notin (0, -min(-\beta+\gamma_1, \gamma_1)) \;\; (i,j)/v'_{i,j} = 1 \\ \notin (min(-\beta+\gamma_1, \gamma_1), 0) \;\; (i,j)/v'_{i,j} = 0 \end{cases} \tag{5.29}$$

Then, the stability of any edge point is avoided.

Proof. As demonstrated in **[102]**, the stability of any edge's adjacent with $T_{ij,ij} < 0$ is guaranteed if, and only if:

$$E_{i,j}(v') \begin{cases} \notin [0, -T_{ij,ij}\,] \;\; \forall \;\; (i,j)/v'_{i,j} = 1 \\ \notin [T_{ij,ij}, 0\,] \;\; \forall \;\; (i,j)/v'_{i,j} = 0 \end{cases} \tag{5.30}$$

From Eq.(5.10), auto-connections are given by:

$$T_{ij,ij} = \begin{cases} \gamma_1, & if \;\; i = 1 \\ -\beta + \gamma_1, & otherwise \end{cases} \tag{5.31}$$

So, based on the given conditions in the theorem, the conditions ($T_{ij,ij} < 0$) is satisfied. Therefore, we obtain:

$$E_{i,j}(v') \notin [T_{ij,ij}, 0\,] \Longleftrightarrow E_{i,j}(v') \notin [min(-\beta+\gamma_1, \gamma_0), 0\,] \quad /v'_{i,j} = 0 \tag{5.32}$$

$$E_{i,j}(v') \notin [0, -T_{ij,ij}] \Longleftrightarrow E_{i,j}(v') \notin [0, -min(-\beta+\gamma_1, \gamma_0)] \quad / v'_{i,j} = 1 \tag{5.33}$$

□

Since the set $H_C - H_F$ is a subset of $H - H_F$, it is important to maintain consistency between the systems (5.27) and (5.29). This is the next theorem.

Theorem 5.3

A point $v \in h_C - H_F$ is not an equilibrium point if the following assertions are verified:

$$S_3 = \begin{cases} \alpha m - \rho \geq -min(-\beta+\gamma_1, \gamma_0) & \alpha, \rho \geq 0 \\ \beta - \gamma_1 \geq -min(-\beta+\gamma_1, \gamma_0) \\ -\gamma_1 \geq -min(-\beta+\gamma_1, \gamma_0) \\ \beta \geq -min(-\beta+\gamma_1, \gamma_0) \end{cases} \tag{5.34}$$

Proof. By hybridizing the theorems 5.1 and 5.2, the parameters system S_3 can be obtained easily.

□

5.5.2 Minimization phase

In this step, we fix the variables vector v, and we solve the following optimization problem with continuous variables :

$$(P_2) : \begin{cases} \min\limits_{(w,\sigma)\in IR^d \times IR^+} E(w, \sigma, .) = \sum_{i=2}^{n+1} \sum_{j=1}^{N_{max}} v_{1,j} v_{i,j} C_{i-1,j} \end{cases} \tag{5.35}$$

The solution of problem (P_2) is given by the following system:

$$\begin{cases} \frac{\partial E}{\partial w_k} = 0 \quad k = 1, \ldots, N_{max}. \\ \frac{\partial E}{\partial \sigma_k} = 0 \quad k = 1, \ldots, N_{max}. \end{cases} \tag{5.36}$$

By using only a simple gradient method. We obtain the following solution:

$$w_k = \frac{\sum_{i=2}^{n+1} \sum_{j=1}^{N_{max}} v_{1,j} v_{i,j} x_{i-1} \frac{\delta(n_k^1, n_j^2) f_k(x_{i-1})}{\sum_{r=1}^{N_{max}} \delta(n_k^1, n_j^2) f_r(x_{i-1})}}{\sum_{i=2}^{n+1} \sum_{j=1}^{N_{max}} v_{1,j} v_{i,j} \frac{\delta(n_k^1, n_j^2) f_k(x_{i-1})}{\sum_{r=1}^{N_{max}} \delta(n_k^1, n_j^2) f_r(x_{i-1})}} \tag{5.37}$$

and

$$\sigma_k = \frac{\sum_{i=2}^{n+1} \sum_{j=1}^{N_{max}} v_{1,j} v_{i,j} x_{i-1} \frac{||w_k - x_{i-1}||^2 \delta(n_k^1, n_j^2) f_k(x_{i-1})}{\sum_{r=1}^{N_{max}} \delta(n_k^1, n_j^2) f_r(x_{i-1})}}{d \sum_{i=1}^{n} \sum_{j=1}^{N_{max}} v_{1,j} v_{i,j} \frac{\delta(n_k^1, n_j^2) f_k(x_{i-1})}{\sum_{r=1}^{N_{max}} \delta(n_k^1, n_j^2) f_r(x_{i-1})}} \tag{5.38}$$

The CHN-PRSOM learning algorithm is given by the pseudocode below :

CHN-PRSOM Algorithm

•**Input:**

-Dataset $D = \{x_i\}_{i=1}^{n}$, N_{iter}, N_{max}.

•**Output:**

- Optimal parameters of CHN-PRSOM (W^*, Σ^*)

— **Initialization:**

— $t \leftarrow 0$; $\quad T \leftarrow T_{max}$;

— $W = \{w_1, ..., w_{N_{max}}\}$, Randomly initialized;

— $\Sigma = \{\Sigma_1, ..., \Sigma_{N_{max}}\}$, Randomly initialized with a great values.

— **Step 1: Repeat**

— **Assignement-decision :**

$$CHN = \begin{cases} \text{Mapping PRSOM into CHN using Eqs.(5.9) and (5.10)} \\ \text{Parameter settings using Theorems 5.1, 5.3 and Eq.(5.21)} \end{cases}$$

— **Minimization :**

— **While** $j \prec N_{max}$ **do**

— **if** $u_{1,j} == 1$ **then**

— Update w_j and Σ_j via Eqs.(5.37) and (5.38)

— **endif**

— $t \leftarrow t + 1, \quad j \leftarrow j + 1, \quad T \leftarrow T_{max} \times (\frac{T_{min}}{T_{max}})^{\frac{t}{N_{iter}-1}}$.

— **done**

— **Until**$(\Delta E(v, W, \Sigma) < \epsilon || iter > N_{iter})$

— $N_{op} = \sum_{j=1}^{N_{max}} u_{1,j}, \quad k \leftarrow 1;$

For j **= 1 to** N_{op}

— **if** $u_{1,j} == 1$ **then**

— $w_k^{init} \leftarrow w_j$;

— $\Sigma_k^{init} \leftarrow \Sigma_j$;

— $k \leftarrow k + 1$.

— **Endif**

— **Endfor**

— **Step 2 :**

— $(W^*, \Sigma^*) \leftarrow \underbrace{Argmin}_{(W,\Sigma)} PRSOM(W^{init}, \Sigma^{init})$.

— **Return**

— Optimal parameters of CHN-PRSOM (W^*, Σ^*).

5.6 Computer simulation

In the present section, we aim to validate the performance of our method in reducing the PRSOM architecture and show the efficiency of the suggested PRSOM-CHN in dealing with data clustering problem[**107**]. In this context, experiences are realized using Four benchmark data sets token from UCI machine repository [**8**].

Table 5.1: Datasets characteristics

Data set	Size	Nr.Tr.D	Nr.Ts.D	At.Nr.	Cl.Nr.
Iris	150	75	75	4	3
Wine	178	90	88	13	3
Glass	214	109	105	9	6
Pima Indians Diabetes	768	384	384	8	2

In Table 5.1, we give brief informations on the numbers of patterns, attributes (At.Nr) and classes (CL.Nr) for each data set. This later is almost divided into two parts: 50% for learning set (Nr.Tr.D) and 50% for tests (Nr.Ts.D).
The first experience relates to the optimal number of neurons in the PRSOM architecture. For this purpose, The proposed model was applied on the data (Table 5.1) with different initializations (N_{max}) that are chosen not large enough to avoid the degenerate solution, and not small enough to cover the optimal solution. Results are illustrated numerically in Table 5.2 and graphically in Figure 5.4.

Table 5.2: Optimal topological maps sizes of Dataset

Data set \ N_{max}	10	15	20	25	30	40	50
Iris	8	7	7	9	7	7	8
Wine	9	10	10	13	10	11	10
Glass	10	13	15	15	16	15	15
Pima Indians Diabetes	10	13	10	11	10	10	11

As shown in Table 5.2, for each data set, all the initializations lead to optimal numbers which are almost similar. For instance, regarding "IRIS" data, our model provides 7 neurons for $N_{max} \in \{15, 20, 30, 40\}$ while it generates 8 neurons for $N_{max} \in \{10, 50\}$ and 9 neurons in the case of $N_{max} = 25$. This result is supported by Figure 5.4 where neurons numbers decrease with the increasing of iterations number. The "Wine" results, as shown in Table 5.2, attest the performance of our model. In fact, the architecture size is obviously decreased at several initializations. Moreover, the two other results concerning "Glass" and "Pima Indians Diabetes" proved the efficiency of our algorithm since the neurons number is significantly reduced.

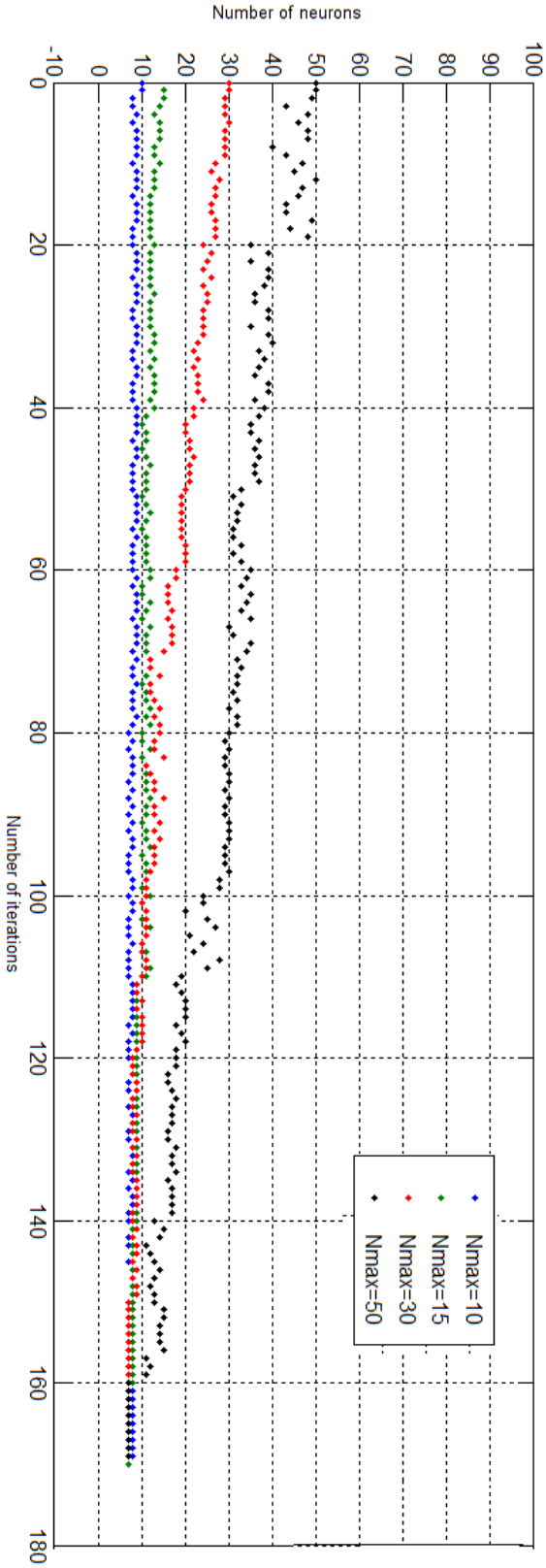


Figure 5.4: Optimal topological size for Iris dataset with different initial number of neurons

Table 5.3: Numerical results of the training data classification

Dataset	Nr.Tr.D.	C.C.Tr	MC.Tr	Overall
Iris	75	73	2	97.3
Wine	90	68	22	75.55
Glass	109	86	23	78.9
Pima Indians Diabetes	384	308	76	80.2

Table 5.3 presents the obtained clustering results of the four training datasets.
The Table 5.3 shows that, for Iris dataset, our method gives good results, because all the training data were correctly classified except two. For the other datasets, we obtained classification rates of 75.55%, 78.9%, 80.2% for respectively Wine, Glass and Pima Indians Diabetes datasets.
In the Table 5.4, we present the obtained clustering results of the four testing datasets.

Table 5.4: Numerical results of the testing data classification

Dataset	Nr.Ts.D.	C.C.Ts	MC.Ts	Overall
Iris	75	74	1	98.6
Wine	88	64	24	72.72
Glass	105	80	25	76.19
Pima Indians Diabetes	384	294	90	76.56

As shown from Table 5.4, for Iris dataset, our method provides important results. In fact, our optimal PRSOM generated only one data misclassified among 75 testing data. For the other datasets, we obtained classification rates of 72.72%, 76.19%, 76.56% for respectively Wine, Glass and Pima Indians Diabetes datasets.
The Figure 5.5 illustrates a comparison between the classification rates of the datasets for both training and testing data. We observe that, for iris dataset, the classification rate of testing data is slightly higher than classification rate of training data. For the three other datasets, the two rates are almost equal. Thus, Our optimal PRSOM is a satisfactory model for the resolution of generalization problems.
PRSOM is a probabilistic model that has proven its performance against several methods of classification including SOM, K-means, ACP and others [**5**, **107**]. Therefore, we aim to reinforce the comparison in term of the optimal topological architecture. To this end, the results of PRSOM-CHN model (Table 5.2) were compared against that of the classical PRSOM model in term of the classification accuracy applied to Iris dataset. In this context, we applied the PRSOM at different sizes (8, 12, 15, 20, 25, 30,). The obtained results are then presented numerically and graphically in the table (5.6) and figure (5.6).

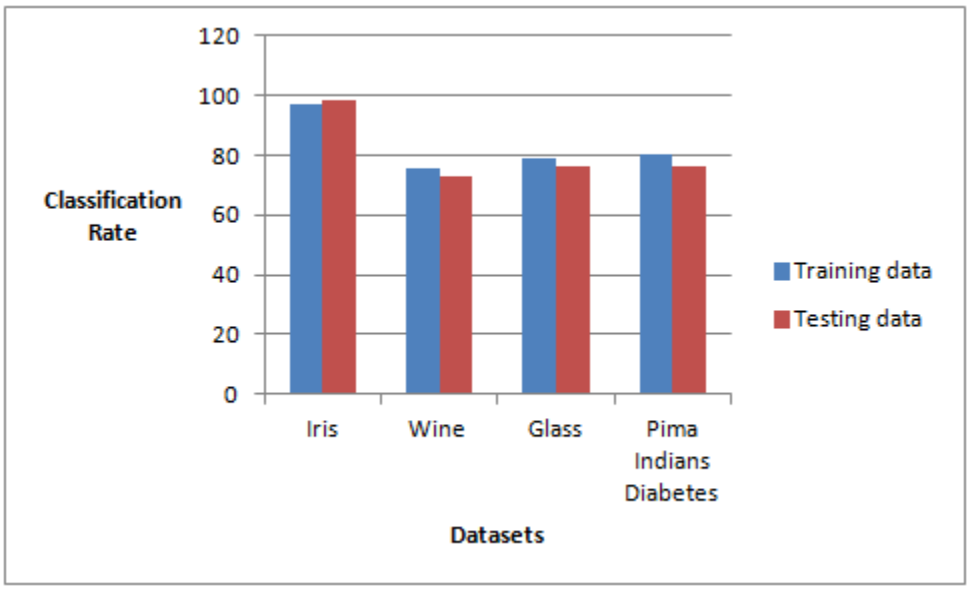


Figure 5.5: Comparison between classification rates of training and testing data

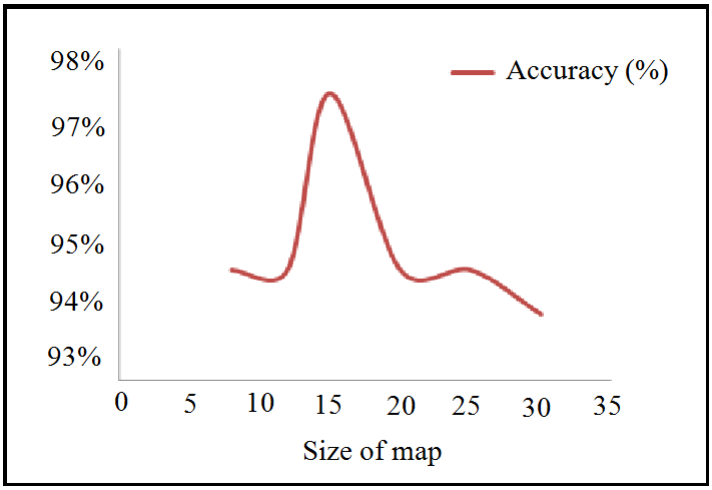


Figure 5.6: Classification rate of PRSOM at different sizes applied to Iris dataset

Table 5.6 as well as Figure 5.6 attested that the map of 15 neurons has provided a higher average (between testing and training data) than the other maps of the classical PRSOM model. Consequently, both models have generated the same classification rate with an advantage of 7 neurons used for PRSOM against 15 for the classical PRSOM.

5.7 Conclusion

In this chapter, a new optimization model of PRSOM architecture is proposed. It consists of a main objective function that describes the PRSOM functioning and constraints on variables. the obtained model is a non linear mathematical program with mixed variables. Resolution phase has been conducted in two phases: assignment and minimiza-

Table 5.5: Classification accuracy of Iris dataset using PRSOM at different maps

K	C.C.Tr.	C.C.Ts.	Accuracy
8	70	72	94.67
12	72	70	94.67
15	72	74	97.33
20	71	71	94.67
25	73	71	96.00
30	70	71	94.00

Table 5.6: Classification accuracy of Iris dataset using PRSOM at different maps

tion phases. The first phase has been performed by continuous Hopfield neural network which has been adapted technically to the problem according to specific parameters extracted from stability conditions. Assignment phase is carried out by gradient method. Suggested Model has been efficiently accomplished to data clustering. In fact, obtained results are satisfactory in term of the optimal architecture and data classification.

Conclusions and Future work

Recurrent neural networks (RNNs) are class of artificial neural networks where connections between units form a directed graph along a sequence. This allows it to exhibit a dynamic temporal behavior for a time sequence. Unlike feed-forward neural networks, RNNs can use their internal state (memory) to process sequences of inputs. In addition, RNNs can be used for optimisations tasks. There are several examples of reccurent neural networks, the most known are the Hopfield neural networks and self organizing maps. In this thesis, the use of the continuous Hopfield neural network to deal with optimization models has been studied in a deep level. Indeed, CHN has been utilized for solving the mathematical problem of image restoration which, nowadays, is classified as one of the major problems in signal processing. The idea behind this choice comes from the fact that the discrete HNN, which has already used to solve IRP, is a limited model owing to the fluctuation drawback that reduces the search space. In this context, CHN has been successively adapted to the model of image restoration, as well as the stability and parameters estimation have been established. Although CHN and other approaches, like Total variation and PDEs, have provided satisfactory results for the image restoration model, they are still limited since the reconstruction process has been performed uniformly across images, which therefore correct some areas at the expense of others. For this reason, a new approach of image restoration called selective image restoration has been introduced and analysed. The suggested model is non linear Mixed-integer optimization problem with two kinds of variables, the first ones are discrete variables, which represent the searched image while the others are decision variables that are responsible for protecting original features using an image of reference. Resolution phase has been performed using CHN and genetic algorithm, in which each one has been charged to solve a part of the model. In this thesis too, the architecture choice problem associated to the recurrent probabilistic self organizing maps is discussed and supported by a new optimisation problem. Indeed, based on the classical PRSOM model, new set of variables have been introduced, thus building a non-linear optimization model under linear and quadratic constraints. In the resolution task, the CHN and the gradient method have been used progressively under suitable parameters chosen from the stability analysis of CHN. The performance of all contributions have been demonstrated by several experiments, which have been divided into visual, graphical and numerical experiments.

Future work

The research findings made out of this thesis has opened several research directions, which have a scope for further investigations. The proposed schemes mostly deal with grayscale images, which can be extended to colour images using the geometric concepts. The computational complexity of each algorithms can be studied and schemes need to be devised to implement them in parallel for better response time. It is also observed that most of the image restoration problems can be modelled as multi-objective optimisation problem for better approximation of solution. Proposed models can also be studied with the spatial variant blur. CHN needs to be supported by new algorithms of architecture reduction. Even though the scheme generates a comparable result, the scheme takes awesome computation. Hence, direction for reduction of computation will be an interesting direction of research.

Appendix A

Instability of interiors points

Theorem A.1

Let $v \in H - H_C$ be an edge point such that $v_{i^0,k^0} \in]0,1[$, this point v will be an equilibrium point for the CHN with $T_{ik,ik} < 0 \; \forall (i,k) \in \{1,..,N^2\} \times \{1,..,G\}$ if, and only if the corner point $v' \in H_C$ defined by:

$$v'_{ik} = \begin{cases} v_{ik} \text{ if } (i,k) \neq (i^0,k^0) \\ 1 \text{ if } (i,k) = (i^0,k^0) \end{cases} \tag{A.1}$$

verifies all these conditions:

$$0 < E_{i^0,k^0}(v') < -T_{i^0k^0,i^0k^0} \tag{A.2}$$

$$E_{i,k}(v') \geq \frac{E_{i,k}(v')}{T_{i^0k^0,i^0k^0}} T_{ik,i^0k^0} \;\; \forall \;\; (i,k) \setminus v'_{i,k} = 0 \tag{A.3}$$

$$E_{i,k}(v') \leq \frac{E_{i,k}(v')}{T_{i^0k^0,i^0k^0}} T_{ik,i^0k^0} \;\; \forall \;\; (i,k) \setminus v'_{i,k} = 1 \tag{A.4}$$

Proof. We notice that the point $v \in H$ will be an equilibrium point for the CHN if, and only if

1.
$$E_{i,k}(v) = 0 \;\; \forall (i,k) \text{ such that } v_{i,k} \in]0,1[$$

As demonstrated in **[102]**, we have the following result:

$$E_{i^0k^0}(v) = E_{i^0k^0}(v') + \sum_{jl} (v'_{jl} - v_{jl}) T_{i^0k^0,jl}$$

$$E_{i^0k^0}(v) = E_{i^0k^0}(v') + (1 - v_{i^0k^0}) T_{i^0k^0,i^0k^0}$$

In this way, taking into account that $T_{i^0k^0,i^0k^0} < 0$

$$E_{i^0k^0}(v) = 0 \Leftrightarrow v_{i^0k^0} = \frac{E_{i^0,k^0}(v')}{T_{i^0k^0,i^0k^0}} + 1$$

By definition of the edge point,

$$v_{i^0k^0} \in]0,1[\Leftrightarrow 0 < E_{i^0,k^0}(v') < -T_{i^0k^0,i^0k^0}$$

2.

$$E_{i,k}(v) \geq 0 \ \ \forall (i,k) \text{ such that } v_{i,k} = 0$$

Let (i,k) be any pair such that $v'_{i,k} = 0$, then we have:

$$E_{ik}(v) = E_{ik}(v') + \sum_{jl}(v'_{jl} - v_{jl})T_{ik,jl}$$

$$E_{ik}(v) = E_{ik}(v') + (1 - v_{i^0k^0})T_{ik,i^0k^0}$$

Replacing the value $v_{i^0k^0} \in]0,1[$, the following equation is obtained:

$$E_{ik}(v) = E_{ik}(v') - \frac{E_{i^0,k^0}(v')}{T_{i^0k^0,i^0k^0}}T_{ik,i^0k^0}$$

which is greater of equal to 0 if, and anly if:

$$E_{ik}(v') \geq \frac{E_{i^0,k^0}(v')}{T_{i^0k^0,i^0k^0}}T_{ik,i^0k^0} \ \ \forall (i,k)/v'_{ik} = 0$$

3.

$$E_{i,k}(v) \leq 0 \ \ \forall (i,k) \text{ such that } v_{i,k} = 1$$

Let (i,k) be any pair such that $v'_{i,k} = 1$, the following equation can be easily proven

$$E_{ik}(v') \leq \frac{E_{i^0,k^0}(v')}{T_{i^0k^0,i^0k^0}}T_{ik,i^0k^0} \ \ \forall (i,k)/v'_{ik} = 1$$

□

The necessary and sufficient conditions given in Theorem (A.1) are based on the corner point $v' \in H_C$ obtained when the value $v_{i^0k^0}$ is increased. Analogously if this value is decreased, another corner point can be obtained and some other conditions are deduced. This is the next theorem:

Theorem A.2

Let $v \in H - H_C$ be an edge point such that $v_{i^0,k^0} \in \,]0,1\,[\}$, this point v will be an equilibrium point for the CHN with $T_{ik,ik} < 0 \; \forall (i,k) \in \{1,..,N^2\} \times \{1,..,G\}$ if, and only if the corner point $v' \in H_C$ defined by:

$$v'_{ik} = \begin{cases} v_{ik} \text{ if } (i,k) \neq (i^0,k^0) \\ 0 \text{ if } (i,k) = (i^0,k^0) \end{cases} \tag{A.5}$$

verifies all these conditions:

$$T_{i^0k^0,i^0k^0} < E_{i^0,k^0}(v') < 0 \tag{A.6}$$

$$E_{i,k}(v') \geq \frac{E_{i,k}(v')}{T_{i^0k^0,i^0k^0}} T_{ik,i^0k^0} \;\; \forall \;\; (i,k) \setminus v'_{i,k} = 0 \tag{A.7}$$

$$E_{i,k}(v') \leq \frac{E_{i,k}(v')}{T_{i^0k^0,i^0k^0}} T_{ik,i^0k^0} \;\; \forall \;\; (i,k) \setminus v'_{i,k} = 1 \tag{A.8}$$

Proof. The demonstration is similar to the proof of theorem A □

Avoiding the conditions of Theorems (A.1) and (A.2), the stability of any edge point is avoided if, and only if:

$$E_{ik}(v') \begin{cases} \notin \; [0, -T_{ik,ik}\,] \;\; \forall \;\; (ik) \setminus v'_{ik} = 1 \\ \notin \; [T_{ik,ik}, 0\,] \;\; \forall \;\; (ik) \setminus v'_{ik} = 0 \end{cases} \tag{A.9}$$

Appendix B

Divergence of multi-interior states

Observation 1
To visualise the landscape of N dimensional Lyapunov energy function, the Hessian of E is obtained as:

$$H = [\frac{\partial^2 E}{\partial v_i \partial v_j}] = -T \tag{B.1}$$

where T=T$_{i,j}$. Since the diagonal of H are all zeros because of $T_{ik,ik} = 0$ for all $i = 1, .., N$. H is always indefinite. This can be shown using the trace of H.

$$T_r(H) = -\sum_{i=1}^{N} T_{i,i} = \sum_{i=1}^{N} \lambda_i \tag{B.2}$$

where λ_i, is the eigenvalues of H. Since $\sum_{i=1}^{N} T_{i,i}$,it becomes obvious that if some $\lambda_i's$ are positive, there must exist negative eigenvalues so that the summation becomes zero unless all $\lambda_i's$ are zeros. This can occur only when H or T matrix is a null matrix. If T is a null matrix, the landscape of E is a monotonic function of $v_i's$. Also,

$$\frac{\partial E}{\partial v_i}] = -\sum_{j=1}^{N} T_{i,j} v_j - I_i \tag{B.3}$$

The critical point of E can be obtained by solving the following matrix equation

$$TV = -I \tag{B.4}$$

Unless T^{-1} does not exist, E has a single critical point V_c where

$$V_c = -T^{-1} I \tag{B.5}$$

Since the Hessian of E is always indefinite, the critical point must be a saddle point. However, if T^{-1} does not exist, E has a continiuum of a saddle point. If T is a null matrix, E is a monotonic function of $v_i's$, $i = l, 2, ..., N$. In this case, the saddle point may be said to be located at the infinity. The hypercube has 2^N vertices where N is the number of neurons.

The length of all its edges is one.
Now it became obvious that the Lyapunov energy function E can have many local minimum points. Also the local minima are formed by the boundary of the hypercube. This is because the Hessian is always indefinite.
Observation 2
It is shown in the previous section that local minima can be located only on the boundary of the hypercube. The boundary includes faces, edges, and vertices of the hypercube. Notice that only one of the local minimum points can be the global minimum. On a face, only two states v_i and v_j are variables while the rest states are kept constant such as zeros and ones. Hence the Lyapunov energy function on the face can be written as

$$E_{face} = -\frac{1}{2}(T_{ij}v_iv_j + T_{ji}v_iv_j) - K_1v_i - k_2v_j \tag{B.6}$$

where K_1 and K_2 are constants.
However, the Hessian of E on the face is obtained as
-$\begin{bmatrix} 0 & T_{ji} \\ T_{ij} & 0 \end{bmatrix}$ As we have shown, since the diagonals are zeros again, there can be no local minimum on the face except on the boundary of the face, which includes vertices and edges of the hypercube. On an edge of the hypercube, E can be written as

$$E_{edge} = K_3v_i$$

where K_3 is a constant and v_i is the state forming the edge. Thus E is a monotonic function on the edge. Therefore a local minimum can't occur on an edge except for a vertex.

Bibliography

[1] Ahi, K. (2017). Mathematical modeling of THz point spread function and simulation of THz imaging systems. IEEE Transactions on Terahertz Science and Technology, 7(6), 747-754.

[2] Aiyer, S. V., Niranjan, M., Fallside, F. (1990). A theoretical investigation into the performance of the Hopfield model. IEEE transactions on neural networks, 1(2), 204-215.

[3] Alvarez, L., Guichard, F., Lions, P. L., Morel, J. M. (1992). Axiomatisation et nouveaux opérateurs de la morphologie mathématique. Comptes rendus de l'Académie des sciences. Série 1, Mathématique, 315(3), 265-268.

[4] Anderson, T. W., Anderson, T. W., Anderson, T. W., Anderson, T. W., Mathématicien, E. U. (1958). An introduction to multivariate statistical analysis (Vol. 2, pp. 5-3). New York: Wiley.

[5] Anouar, F., Badran, F., Thiria, S. (1998). Probabilistic self-organizing map and radial basis function networks. Neurocomputing, 20(1-3), 83-96.

[6] Arthur, D., Vassilvitskii, S. (2007, January). k-means++: The advantages of careful seeding. In Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms (pp. 1027-1035). Society for Industrial and Applied Mathematics.

[7] Astola, J., and Kuosmanen. P, (1997). Fundamentals of nonlinear digital filtering (Vol. 8). CRC press.Rafael

[8] Asuncion, A., Newman, D. (2007). UCI machine learning repository.

[9] Aubert, G., Kornprobst, P. (2006). Mathematical problems in image processing: partial differential equations and the calculus of variations (Vol. 147). Springer Science Business Media.

[10] Aubert, G., Vese, L. (1997). A variational method in image recovery. SIAM Journal on Numerical Analysis, 34(5), 1948-1979.

[11] A. Chambolle, An algorithm for total variation minimization and applications, Journal of Mathematical Imaging and Vision, Vol.20, No.1, March , pp.89-97, (2004)

[12] Banham, M. R., Katsaggelos, A. K. (1997). Digital image restoration. IEEE signal processing magazine, 14(2), 24-41.

[13] Bardsley, J. M., Nagy, J. G. (2006). Covariance-preconditioned iterative methods for nonnegatively constrained astronomical imaging. SIAM journal on matrix analysis and applications, 27(4), 1184-1197.

[14] Berisha, S., Nagy, J. G. (2014). Iterative methods for image restoration. In Academic Press Library in Signal Processing (Vol. 4, pp. 193-247). Elsevier.

[15] Bouman, C., Sauer, K. (1993). A generalized Gaussian image model for edge-preserving MAP estimation. IEEE Transactions on image processing, 2(3), 296-310.

[16] Bouzerdoum, A., Pattison, T. R. (1993). Neural network for quadratic optimization with bound constraints. IEEE transactions on neural networks, 4(2), 293-304

[17] Brianzi, P., Di Benedetto, F., Estatico, C. (2008). Improvement of space-invariant image deblurring by preconditioned Landweber iterations. SIAM Journal on Scientific Computing, 30(3), 1430-1458.

[18] Bruneau, P., Gelgon, M., Picarougne, F. (2010). Parsimonious reduction of Gaussian mixture models with a variational-Bayes approach. Pattern Recognition, 43(3), 850-858.

[19] Castillo, P. A., Merelo, J. J., Prieto, A., Rivas, V., Romero, G. (2000). G-Prop: Global optimization of multilayer perceptrons using GAs. Neurocomputing, 35(1-4), 149-163.

[20] Chambolle, A., Caselles, V., Cremers, D., Novaga, M., Pock, T. An introduction to total variation for image analysis. Theoretical foundations and numerical methods for sparse recovery, 9(263-340), 227.(2010)

[21] Chellappa, R., Jain, A. (1993). Markov random fields. Theory and application. Boston: Academic Press, 1993, edited by Chellappa, Rama; Jain, Anil.

[22] Chua, L. O., Yang, L. (1988). Cellular neural networks: Applications. IEEE Transactions on circuits and systems, 35(10), 1273-1290.

[23] Chan, C. L., Katsaggelos, A. K., Sahakian, A. V. (1993). Image sequence filtering in quantum-limited noise with applications to low-dose fluoroscopy. IEEE Transactions on Medical Imaging, 12(3), 610-621.

[24] Cochocki, A., Unbehauen, R. (1993). Neural networks for optimization and signal processing. John Wiley Sons, Inc.

[25] Demoment, G. (1989). Image reconstruction and restoration: Overview of common estimation structures and problems. IEEE Transactions on Acoustics, Speech, and Signal Processing, 37(12), 2024-2036.

[26] Dempster, A. P., Laird, N. M., Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. Journal of the royal statistical society. Series B (methodological), 1-38.

[27] DiBenedetto, E. (1993). Degenerate and singular parabolic equations, Universitext.

[28] Dirac, P. A. M. (1953). The Lorentz transformation and absolute time. Physica, 19(1-12), 888-896.

[29] D. Geman and G. Reynolds. 'Constrained restoration and the recovery of discontinuities.' IEEE Transactions on Pattern Analysis and Machine Intelligence, 14(3), 1993, pp.367-383.

[30] Digital Image Processing Chapter 2, Part II Digitized image and its properties: Image digitization, http://user.engineering.uiowa.edu/ dip/lecture/ImageProperties2.html

[31] Engl, H. W., Hanke, M., Neubauer, A. (1996). Regularization of inverse problems (Vol. 375). Springer Science Business Media.

[32] En-Naimani, Z., Lazaar, M., Ettaouil, M. (2016). Architecture Optimization Model for the Probabilistic Self-Organizing Maps and Speech Compression. International Journal of Computational Intelligence and Applications, 15(02), 1650007.

[33] EN-NAIMANI, ZAKARIAE, LAZAAR, MOHAMED, et ETTAOUIL, MOHAMED. Hybrid system of optimal self organizing maps and hidden Markov model for Arabic digits recognition. WSEAS Transactions on Systems, 2014, vol. 13, no 60, p. 606-616.

[34] ETTAOUIL, M., ABDELATIF, E., HARCHLI, F. (2013). IMPROVING THE PERFORMANCE OF K-MEANS ALGORITHM USING AN AUTOMATIC CHOICE OF SUITABLE CODE VECTORS AND OPTIMAL NUMBER OF CLUSTERS. Journal of Theoretical Applied Information Technology, 56(3).

[35] Ettaouil, M., Ghanou, Y. (2009). Neural architectures optimization and Genetic algorithms. Wseas Transactions On Computer, 8(3), 526-537.

[36] Ettaouil, M., Nour-eddine, J., Harchli, F., Abdelatif, E. (2015, October). A genetic algorithm for solving a new image restoration model based on selective filtering. In Intelligent Systems: Theories and Applications (SITA), 2015 10th International Conference on (pp. 1-5). IEEE.

[37] Friedman, J. H. (1989). Regularized discriminant analysis. Journal of the American statistical association, 84(405), 165-175.

[38] García, L., Talaván, P. M., Yáñez, J. (2017). Improving the Hopfield model performance when applied to the traveling salesman problem. Soft Computing, 21(14), 3891-3905.

[39] Gonzalez, R. C., Woods, R. E. (2002). Digital Image Processing. PrenticeMHall.

[40] Groetsch, C. W. (1984). The theory of tikhonov regularization for fredholm equations. 104p, Boston Pitman Publication.

[41] Gopal, M. (1993). Modern control system theory. New Age International.

[42] Ham, F. M., Kostanic, I. (2000). Principles of neurocomputing for science and engineering. McGraw-Hill Higher Education.

[43] Hansen, P. C., Nagy, J. G., O'leary, D. P. (2006). Deblurring images: matrices, spectra, and filtering (Vol. 3). Siam.

[44] Hansen, P. C. (2010). Discrete inverse problems: insight and algorithms (Vol. 7). Siam.

[45] HARCHLI, F., JOUDAR .N, Es-SAFI .A, ETTAOUIL .M. Adaptation of Multilayer Perceptron Neural Network to unsupervised Clustering using a developed version of k-means algorithm. WSEAS TRANSACTIONS on COMPUTERS. vol.15, 103-116.

[46] Holland J H (1975). 'Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence'. University of Michigan Press,.

[47] Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. Proceedings of the national academy of sciences, 79(8), 2554-2558.

[48] Hopfield, J. J. (1984). Neurons with graded response have collective computational properties like those of two-state neurons. Proceedings of the national academy of sciences, 81(10), 3088-3092.

[49] Hopfield, J. J., Tank, D. W. (1985). "Neural" computation of decisions in optimization problems. Biological cybernetics, 52(3), 141-152.

[50] Hopfield, J. J., Tank, D. W. (1986). Computing with neural circuits: A model. Science, 233(4764), 625-633.

[51] Jain, A. K. (1989). Fundamentals of digital image processing. Prentice-Hall, Inc.

[52] Jähne, B. (1993). Spatio-temporal image processing: theory and scientific applications (Vol. 751). Springer Science Business Media.

[53] Jensen, J. R., Lulla, K. (1987). Introductory digital image processing: a remote sensing perspective.

[54] J Hadamard, 'Sur les problèmes aux dérivées partielles et leur signification physique.', Princeton Univ.Bull, Vol.13, ,1902, pp.49-52.

[55] Joudar, N-E., En-naimani, Z., Ettaouil, M., Using Continuous Hop
eld neural network for solving a new optimization architecture model of probabilistic self organizing map, Neurocomputing, Elsevier, (Revised version)

[56] Joudar, N., and El Moutouakil, K. and Ettaouil, M. (2015). 'An original Continuous Hopfield Network for optimal images restoration'. WSEAS TRANSACTIONS on COMPUTERS, Vol.14, pp.668-678.

[57] Joudar, N., and Ettaouil, M. (2017, March). 'An improved model for restoring and protecting the digital images: mathematical modelling and optimization'. In Proceedings of the 2nd international Conference on Big Data, Cloud and Applications (p. 72). ACM.

[58] Joudar, N-E., Harchli, F., Abdelatif, E-S. and Ettaouil, M. (2017) 'New adaptive switching scheme for impulse noise removal: modelling and resolution by genetic optimization', Int. J. Signal and Imaging Systems Engineering, Vol. 10, No. 6, pp.316-–323.

[59] Joudar, N-E., Ettaouil, M., Mixed-integer model for selective image restoration: Modelling and resolution by CHN and GA. Journal of circuits systems and signal processing CSSP, Springer (Revised version)

[60] Kangas, J. A., Kohonen, T. K., Laaksonen, J. T. (1990). Variants of self-organizing maps. IEEE transactions on neural networks, 1(1), 93-99.

[61] Katsaggelos, A. K, (2012). 'Digital image restoration', Springer Publishing Company, Incorporated.

[62] Kennedy, M. P., Chua, L. O. (1988). Neural networks for nonlinear programming. IEEE Transactions on Circuits and Systems, 35(5), 554-562.

[63] Kirsch, A. (2011). An introduction to the mathematical theory of inverse problems (Vol. 120). Springer Science Business Media.

[64] J.J. Koenderink. The structure of images. Biological Cybernetics, 50:363– 370, 1984.

[65] Kohonen, T., Kaski, S., Lagus, K., Salojarvi, J., Honkela, J., Paatero, V., Saarela, A. (2000). Self organization of a massive document collection. IEEE transactions on neural networks, 11(3), 574-585.

[66] Kohonen, T. (2012). Self-organization and associative memory (Vol. 8). Springer Science Business Media.

[67] Lebbah, M., Jaziri, R., Bennani, Y., Chenot, J. H. (2015). Probabilistic self-organizing map for clustering and visualizing non-iid data. International Journal of Computational Intelligence and Applications, 14(02), 1550007.

[68] Liang, X. (2007). Removal of hidden neurons in multilayer perceptrons by orthogonal projection and weight crosswise propagation. Neural Computing and Applications, 16(1), 57-68.

[69] Lindeberg, T., ter Haar Romeny, B. M. (1994). Linear scale-space I: Basic theory. In Geometry-Driven Diffusion in Computer Vision (pp. 1-38). Springer, Dordrecht.

[70] Looi, C. K. (1992). Neural network methods in combinatorial optimization. Computers Operations Research, 19(3-4), 191-208.

[71] López-Rubio, E. (2010). Probabilistic self-organizing maps for qualitative data. Neural Networks, 23(10), 1208-1225.

[72] Marr, D., Hildreth, E. (1980, February). Theory of edge detection. In Proc. R. Soc. Lond. B (Vol. 207, No. 1167, pp. 187-217). The Royal Society.

[73] M. Bertero and P. Boccacci. Introduction to Inverse Problems in Imaging. Bristol: IoP, Institute of Physics Publishing, 1998.

[74] McCulloch, W. S., Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. The bulletin of mathematical biophysics, 5(4), 115-133.

[75] McLachlan, G., Peel, D. (2004). Finite mixture models. John Wiley Sons.

[76] Minsky, M., Papert, S. A., Bottou, L. (2017). Perceptrons: An introduction to computational geometry. MIT press.

[77] Morozov, V. A., Stessin, M. (1993). Regularization methods for ill-posed problems (p. 257). Boca Raton, FL:: CRC press.

[78] M.Takeda and J.W.Goodman, 'Neural networks for computation: number representation and programming complexity', Applied optics, Vol.25, No.18, September 1986, pp.3033-3046.

[79] Nagy, J. G., Palmer, K., Perrone, L. (2004). Iterative methods for image deblurring: a Matlab object-oriented approach. Numerical Algorithms, 36(1), 73-93.

[80] Nasrabadi, N. M. (2007). Pattern recognition and machine learning. Journal of electronic imaging, 16(4), 049901.

[81] Ozkam M K, Erdem A T, Sezan M I. Efficient multiframe Wiener restoration of blurred and noisy image sequences. IEEE Transactions on Image Processing,1992,1(4):453-478.

[82] Paik, J.K and Katsaggelos. A. (1992) 'Image restoration using a modified hopfield network', IEEE Trans. on Image Processing, Vol. 1, No. 1, pp.546–555.

[83] Park, S. (1989, May). 'Signal space interpretations of Hopfield neural network for optimization'. In Circuits and Systems, 1989., IEEE International Symposium on (pp. 2181-2184). IEEE.

[84] Patterson, D. W. (1998). Artificial neural networks: theory and applications. Prentice Hall PTR.

[85] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. IEEE Transactions on Pattern Analysis and Machine Intelligence, 12(7):629–639, July 1990.

[86] P.Hansen, analysis of discrete ill-posed problems by means of the L-curve, SIAM, Review.43, pp.561-580, (1992)

[87] Prieto, A., Atencia, M., Sandoval, F. (2013). Advances in artificial neural networks and machine learning.

[88] Qu, H., Yi, Z., Tang, H. (2006). Improving local minima of columnar competitive model for TSPs. IEEE Transactions on Circuits and Systems I: Regular Papers, 53(6), 1353-1362.

[89] Rafael, C. G., Woods, R. E., Masters, B. R. (2009). Digital Image Processing Third Edition. Journal of Biomedical Optics, 14(2), 331-333.

[90] Rodriguez-Vazquez, A., Dominguez-Castro, R., Rueda, A., Huertas, J. L., Sanchez-Sinencio, E. (1990). Nonlinear switched capacitor'neural'networks for optimization problems. IEEE Transactions on Circuits and Systems, 37(3), 384-398.

[91] Rosenblatt, F. (1962). Principles of neurodynamics.

[92] Rudin, L. I., Osher, S., Fatemi, E. (1992). Nonlinear total variation based noise removal algorithms. Physica D: nonlinear phenomena, 60(1-4), 259-268.

[93] Rumelhart, D. E., McClelland, J. L., PDP Research Group. (1987). Parallel distributed processing (Vol. 1, p. 184). Cambridge, MA: MIT press.

[94] Sakthidasan K, Nagappan N, Velmurugan N (2016). 'Noise free image restoration using hybrid filter with adaptive genetic algorithm'. journal of computer and electrical engineering,. pp54: 382-392.

[95] Scarselli, F., Tsoi, A. C. (1998). Universal approximation using feedforward neural networks: A survey of some existing methods, and some new results. Neural networks, 11(1), 15-37.

[96] Schalkoff, R. J. (1989). Digital image processing and computer vision (Vol. 286). New York: Wiley.

[97] Shih, H. S., Wen, U. P., Lee, S., Lan, K. M., Hsiao, H. C. (2004). A neural network approach to multiobjective and multilevel programming problems. Computers Mathematics with Applications, 48(1-2), 95-108.

[98] Smith, K., Palaniswami, M., Krishnamoorthy, M. (1998). Neural techniques for combinatorial optimization with applications. IEEE Transactions on Neural Networks, 9(6), 1301-1318.

[99] Specht, D. F. (1990). Probabilistic neural networks. Neural networks, 3(1), 109-118.

[100] Soltanian-Zadeh, H., Windham, J. P., Yagle, A. E. (1995). A multidimensional nonlinear edge-preserving filter for magnetic resonance image restoration. IEEE Transactions on Image Processing, 4(2), 147-161.

[101] Syed, M. N., Pardalos, P. M. (2013). Neural network models in combinatorial optimization. In Handbook of Combinatorial Optimization (pp. 2027-2093). Springer New York.

[102] Talavan .P.M and J.Yanaz, 'Parameter setting of the Hopfield network applied to TSP', Neural networks, Vol.15, 2002, pp 363-373.

[103] Talaván, P. M., Yáñez, J. A continuous Hopfield network equilibrium points algorithm. Computers operations research, 32(8), 2179-2196. (2005)

[104] Talavan .P.M, J. Yanez 'The generalized quadratic knapsack problem. A neuronal network approach', Neural Networks, Vol. 19, 2006, pp. 416-428.

[105] Tan, K. C., Tang, H., Ge, S. S. (2005). On parameter settings of Hopfield networks applied to traveling salesman problems. IEEE Transactions on Circuits and Systems I: Regular Papers, 52(5), 994-1002.

[106] Tikhonov and Arsenine, 'Methodes de resolution de problemes mal poses'. Editions MIR, Moscou, Russie, 1976.

[107] Tzortzis, G. F., Likas, A. C. (2009). The Global Kernel *k*-Means Algorithm for Clustering in Feature Space. IEEE Transactions on Neural Networks, 20(7), 1181-1194.

[108] Uma, S., Annadurai, S. (2008). 'Image restoration using Modified Recurrent Hopfield Neural Network'. International Journal of Signal and Imaging Systems Engineering, Vol.1 No.3-4, pp.264-272.

[109] Van Loan, C. F. (1996). Matrix computations (Johns Hopkins studies in mathematical sciences).

[110] Vese, L. (2001). A Study in the BV Space of a Denoising—Deblurring Variational Problem. Applied Mathematics and optimization, 44(2), 131-161.

[111] Vogel, C. R. (2002). Computational methods for inverse problems (Vol. 23). Siam.

[112] Wang, J. (1993). Analysis and design of a recurrent neural network for linear programming. IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, 40(9), 613-618.

[113] Watta, P. B., Hassoun, M. H. (1996). A coupled gradient network approach for static and temporal mixed-integer optimization. IEEE Transactions on Neural Networks, 7(3), 578-593.

[114] Wen, K.-M. Lan, H.-S. Shih, 'A review of hopfield neural networks for solving mathematical programming problems', European Journal of Operational Research 198 (3) (2009) pp.675–687.

[115] Werbos, P. (1974). Beyond regression: new fools for prediction and analysis in the behavioral sciences. PhD thesis, Harvard University.

[116] Witkin, A. P. (1987). Scale-space filtering. In Readings in Computer Vision (pp. 329-332).

[117] Wu W, Kundu A. Image estimation using fast modified reduced update Kalman filter . IEEE Transaction on Signal Processing.1992,40(4):915-926.

[118] Wu, Y. D. and al, (2006). 'Image restoration using variational PDE-based neural network'. Neurocomputing, Vol.69 No.16, pp.2364-2368.

[119] Xavier, G. M. T., Castañeda, F. G., Nava, L. M. F., Cadenas, J. A. M. (2018). Memristive recurrent neural network. Neurocomputing, 273, 281-295.

[120] X. Zeng and D. S. Yeung, "Hidden neuron pruning of multilayer perceptrons using a quantified sensitivity Measure", Neurocomputing, Vol. 69, 2006, pp. 825-837

[121] Yadav, N., Yadav, A., Kumar, M. (2015). An introduction to neural network methods for differential equations. Berlin, Germany: Springer.

[122] Yang, J., Wang, L., Wang, Y., Guo, T. (2017). A novel memristive Hopfield neural network with application in associative memory. Neurocomputing, 227, 142-148.

[123] Yu-Li You, M. Kaveh, 2000, 'Fourth Order Partial Differential Equations for Noise Removal', IEEE Trans. Image Processing, vol. 9, no. 10, pp 1723-1730,

[124] Y.ZHOU and R.Chellapa, (1988) "Image restoration using a neural network", IEEE transactions on acoustics, speech, and signal processing. Vol. 36, No. 7, pp.1141-1151.

[125] Zuo, Z., Zhang, T., Lan, X., and Yan, L. An adaptive non-local total variation blind deconvolution employing split Bregman iteration. Circuits, Systems, and Signal Processing, 32(5), 2407-2421.(2013)