



Department: Electrical Engineering

Order N° : 03 / 2021

Defense authorization N° 54/2021

## DOCTORAL THESIS

3rd Cycle Doctoral (D-LMD)

Presented by

**Lakhdar KADDOURI**

With a view to obtaining the doctoral diploma in 3rd Cycle Doctoral (D-LMD)

Branch: Electronics

Specialty: Signal Image and Systems

### Topic

**SYNTHESE DE FILTRES RECURSIFS BIDIMENSIONNELS BASEE SUR L'ALGORITHME PSO**

Supported, on 20/02/ 2021, before the jury composed of:

Last and first name	Grade	Institution of affiliation	Designation
Mr. Omar MANSOUR	Professeur	Université de Djelfa	President
Mrs. Amel Baha Houda ADAMOU-MITICHE	Professeur	Université de Djelfa	Supervisor
Mr. Lahcène MITICHE	Professeur	Université de Djelfa	Examiner
Mr. Abdesselam HOCINI	Professeur	Université de M'sila	Examiner
Mr. Farouk CHELALI	MCA	Université de Djelfa	Examiner

**Djelfa University, FST -2021**



Département : Génie Electrique

N° d'Ordre : 03 / 2021

Autorisation de Soutenance N° 54/2021

## THESE DE DOCTORAT

Doctorat 3<sup>ème</sup> Cycle (D-LMD)

Présentée par

**Lakhdar KADDOURI**

En vue de l'obtention du diplôme de Docteur en 3<sup>ème</sup> Cycle D-LMD

Filière : Electronique

Spécialité : Signal Image et Systèmes

### Thème

### **SYNTHESE DE FILTRES RECURSIFS BIDIMENSIONNELS BASEE SUR L'ALGORITHME PSO**

Soutenue publiquement, le 20/02/2021, devant le jury composé de :

Nom et Prénom	Grade	Etablissement de rattachement	Désignation
Mr. Omar MANSOUR	Professeur	Université de Djelfa	Président
Mrs. Amel Baha Houda ADAMOU-MITICHE	Professeur	Université de Djelfa	Directeur de Thèse
Mr. Lahcène MITICHE	Professeur	Université de Djelfa	Examineur
Mr. Abdesselam HOCINI	Professeur	Université de M'sila	Examineur
Mr. Farouk CHELALI	MC A	Université de Djelfa	Examineur

**Université de Djelfa, FST, 2021**

## Résumé

L'optimisation par essaim de particules (OEP), ou (Particle Swarm Optimization (PSO)) en anglais, est un algorithme évolutionnaire largement utilisé dans les problèmes d'optimisation. Il se caractérise par sa convergence rapide, ce qui peut le conduire à stagner dans des optima locaux. Dans la présente étude, un nouvel algorithme Multi-PSO pour la conception de filtres à réponse impulsionnelle infinie bidimensionnelle (RII) est construit. Il est basé sur le PSO standard et utilise une nouvelle stratégie d'initialisation. Cette stratégie est relayée à deux types d'essaims : un principal et des auxiliaires. Pour améliorer les performances de l'algorithme, l'espace de recherche est divisé en plusieurs zones, ce qui permet une meilleure couverture et conduit à une meilleure exploration dans chaque zone séparément. Cela résout le problème de la convergence rapide dans le PSO standard. Les résultats obtenus démontrent l'efficacité du Multi-PSO dans l'optimisation des coefficients de filtre.

**Mots clés :** Filtres 2D récurrents, Optimisation, Essaim, Algorithme PSO

## ABSTRACT

Particle Swarm Optimization (PSO) is an evolutionary algorithm widely used in optimization problems. It is characterized by a fast convergence, which can lead the algorithm to stagnate in local optima. In the present study, a new Multi-PSO algorithm for the design of two-dimensional infinite impulse response (IIR) filters is built. It is based on the standard PSO and uses a new initialization strategy. This strategy is relayed to two types of swarms: a principal and auxiliaries. To improve the performance of the algorithm, the search space is divided into several areas, which allows a best covering and leading to a better exploration in each zone separately. This solved the problem of fast convergence in standard PSO. The results obtained demonstrate the effectiveness of the Multi-PSO algorithm in the filter coefficients optimization.

**Keywords:** two-dimensional recursive Filters, Optimization, Swarm, PSO algorithm

## المخلص

يعد التقريب في حل المسائل عن طريق سرب الجسيمات خوارزمية تطويرية تستخدم على نطاق واسع في الحل التقريبي للمسائل وتتميز بتقاربها السريع الذي يمكن أن يؤدي بها إلى الركود في أوضاع محلية للحلول. وفي هذه الدراسة، تم إنشاء خوارزمية جديدة تستعمل مجموعة اسراب لتصميم مرشحات ذات الاستجابة النبضية اللانهائية الثنائية الأبعاد. نعتمد فيها خوارزمية سرب الجسيمات القياسية مع استخدام استراتيجية جديدة حيث نقسم فضاء البحث الي مقاطعات وترفق كل مقاطعة بسرب حتى يكون البحث عن الحل فعال بالإضافة للاستعمال نوعين من الاسراب واحد رئيسي واسراب أخرى ثانوية وبالفعل كانت نتائج هذه الاستراتيجية جيدة جدا وأثبتت فعاليتها في حل هذه المسألة

**الكلمات المفتاحية:** المرشحات ثنائية الأبعاد التكرارية، الحساب التقريبي، السرب، الخوارزمية

بسو

## *Remerciements*

Je tiens à remercier mes promoteurs, **Professeur Mme Amel B. H. ADAMOUMITICHE** Directeur de la thèse et le **Professeur LAHCENE MITICHE** POUR leur confiance, leurs précieux conseils, leur disponibilité au long de la préparation de cette thèse. Que le président de jury : **Professeur Omar MANSOUR** et les membres de jury : **Professeur Abdesselam HOCINI** et **monsieur Farouk CHELALI, Maître de Conférences** trouvent ici le témoignage de ma reconnaissance pour avoir bien voulu juger mon travail. À ceux qui m'ont soutenu de près ou de loin pour la réalisation de ce travail, un grand merci. Finalement je remercie chaleureusement ma famille pour la confiance qu'ils m'accordent, leurs encouragements et leurs soutiens.

## SOMMAIRE

<b>Introduction générale.....</b>	<b>01</b>
<b>1. Éléments de base sur les signaux et systèmes bidimensionnels (2D).....</b>	<b>05</b>
<b>1.1. Introduction.....</b>	<b>05</b>
<b>1.2. Les signaux déterministes 2D.....</b>	<b>05</b>
<b>1.2.1. Signaux bidimensionnels particuliers .....</b>	<b>06</b>
<b>1.3. Signaux bidimensionnels aléatoires.....</b>	<b>08</b>
<b>1.3.1. Caractéristique des signaux aléatoires bidimensionnels.....</b>	<b>09</b>
<b>1.4. Systèmes 2D .....</b>	<b>11</b>
<b>1.4.1. Propriétés principales.....</b>	<b>11</b>
<b>1.4.2. Les systèmes Linéaire invariants dans le temps.....</b>	<b>11</b>
<b>1.4.3. Système séparable .....</b>	<b>12</b>
<b>1.4.4. Stabilité des systèmes 2D.....</b>	<b>13</b>
<b>1.4.5. Réponse fréquentielle d'un système LTI .....</b>	<b>13</b>
<b>1.5. Transformée en z (TZ) 2D.....</b>	<b>14</b>
<b>1.5.1. Définition.....</b>	<b>14</b>
<b>1.5.2. Région de convergence.....</b>	<b>15</b>
<b>1.5.3 Propriétés de la transformée en z 2D.....</b>	<b>15</b>
<b>1.6. Transformée en z inverse 2D.....</b>	<b>16</b>
<b>1.7. Les Filtres 2D.....</b>	<b>17</b>
<b>1.7.1. Filtres à phase linéaire 2D.....</b>	<b>17</b>
<b>1.7.2. Caractéristiques des filtres 2D.....</b>	<b>18</b>
<b>1.7.3. Classification des filtres numériques 2D.....</b>	<b>21</b>
<b>1.7.4. Comparaison des filtres.....</b>	<b>21</b>
<b>1.8. Conclusion .....</b>	<b>22</b>
<b>2. Synthèse des filtres 2D à Réponse Impulsionnelle Infinie (RII).....</b>	<b>24</b>
<b>2.1. Introduction.....</b>	<b>24</b>
<b>2.2. Définition .....</b>	<b>24</b>
<b>2.3. Méthodes de synthèse des filtres RII 2D.....</b>	<b>25</b>
<b>2.4. Problèmes de stabilité .....</b>	<b>30</b>
<b>2.5. Conclusion.....</b>	<b>35</b>

<b>3. Algorithmes d'optimisation</b> .....	<b>37</b>
<b>3.1. Introduction</b> .....	<b>37</b>
<b>3.2. Algorithmes Heuristiques</b> .....	<b>37</b>
<b>3.3. Algorithmes Métaheuristiques</b> .....	<b>37</b>
<b>3.4. Différentes Algorithme d'optimisation</b> .....	<b>39</b>
<b>3.4.1. Algorithme du recuit simulé</b> .....	<b>39</b>
<b>3.4.2. Algorithme de recherche tabou</b> .....	<b>40</b>
<b>3.4.3. Les algorithmes de colonies de fourmis</b> .....	<b>40</b>
<b>3.4.4. Algorithme à évolution différentielle</b> .....	<b>41</b>
<b>3.4.5. Les algorithmes évolutionnaires</b> .....	<b>41</b>
<b>3.5. Optimisation par essaim de particules</b> .....	<b>43</b>
<b>3.5.1. Principe général</b> .....	<b>43</b>
<b>3.5.2. Formalisation</b> .....	<b>44</b>
<b>3.5.3 Améliorations de PSO</b> .....	<b>45</b>
<b>3.5.3.1. Coefficient de constriction</b> .....	<b>46</b>
<b>3.5.3.2. Topologie de voisinage</b> .....	<b>47</b>
<b>3.5.3.3. Coefficient d'inertie</b> .....	<b>48</b>
<b>3.5.4. L'algorithme Multi-PSO</b> .....	<b>48</b>
<b>3.5.5. Algorithme proposé</b> .....	<b>49</b>
<b>3.5.5.1. Partitionnement de l'espace de recherche</b> .....	<b>50</b>
<b>3.6. Conclusion</b> .....	<b>53</b>
<b>4. Résultats de simulation et discussion</b> .....	<b>55</b>
<b>4.1. Introduction</b> .....	<b>55</b>
<b>4.2. Algorithme</b> .....	<b>55</b>
<b>4.2.1. Les étapes d'algorithme proposé</b> .....	<b>56</b>
<b>4.2.2. Organigramme de l'algorithme proposé</b> .....	<b>57</b>
<b>4.3.1. Formulation du problème de synthèse de filtres récurrents bidimensionnels</b> .....	<b>58</b>
<b>4.3.2 Fonction de transfert</b> .....	<b>63</b>
<b>4.4. Résultats et discussion</b> .....	<b>64</b>
<b>4.5 Conclusion</b> .....	<b>70</b>
<b>Conclusion générale</b> .....	<b>72</b>

## LISTE DES FIGURES

Figure 1.1 Impulsion unitaire 2D	06
Figure 1.2 Le pas unitaire 2D	07
Figure 1.3: Exemple de signal périodique 2D	08
Figure 1.4 : les réponses fréquentielles des filtres idéaux circulairement symétriques.	19
Figure 1.5 schéma de tolérance des caractéristiques de filtre passe-bas 2-D	20
Figure 3.1 Déplacement d'une particule.	44
Figure 3.2 Partitionnement d'un espace de recherche uniforme.	52
Figure 3.3 Partitionnement d'un espace de recherche non uniforme.	52
Figure 4.1 Réponse en amplitude désirée $M_d(\omega_1, \omega_2)$ du filtre 2-D.	59
Figure 4.2 Réseau sans perte et indépendant de la fréquence.	59
Figure 4.3. Réseau de trois bornes	61
Figure 4.4 Comparaison des réponses en amplitude $M(\omega_1, \omega_2)$ en utilisant différents algorithmes	68

### Introduction générale

Le champ d'application du signal bidimensionnel (2D) a grandi rapidement au cours des dernières décennies. Des images satellites, radiographies médicales, micrographies électroniques, enregistrements magnétiques et signaux de radar sont les exemples typiques des signaux bidimensionnels qui nécessitent de traitement par des filtres numériques [1]. Les principales fonctions du filtre sont d'éliminer les parties indésirables d'un signal, comme le bruit aléatoire, ou extraire des parties. Par exemple, une image dégradée par un bruit de bande peut être améliorée sans brouiller ses bords. Une bande magnétique peut être rendue plus lisible en supprimant certains signaux de grande amplitude et basse fréquence [2].

Un signal bidimensionnel continu est une quantité physique ou électrique qui est une fonction continue de deux variables indépendantes réelles. Un exemple d'un tel signal pourrait être l'intensité lumineuse dans le cas d'une photographie ou d'une image en fonction des distances dans les directions  $x$  et  $y$ . Un signal 2D discret est une version échantillonnée d'un signal 2D continu et se présente sous la forme d'un tableau 2D de nombres [3]. Comme un signal discret unidimensionnel (1D), un signal discret bidimensionnel peut être représenté par un spectre fréquentiel qui peut être modifié, remodelé ou manipulé par filtrage. Ce type de traitement peut être réalisé en utilisant des filtres numériques 2D. Les filtres numériques bidimensionnels, comme leurs homologues 1D, sont des systèmes discrets qui peuvent être linéaires ou non linéaires, invariants ou dépendants du décalage, causaux ou non, et stables ou instables. Ils peuvent être caractérisés en termes d'équations de différence ou d'équations d'espace d'états en deux variables.

Les filtres numériques sont généralement classés en deux groupes : filtres à réponse impulsionnelle infinie (RII) ou filtres récursifs et filtres à réponse impulsionnelle finie (RIF) ou filtres non récursifs [4]. Les filtres RIF sont généralement plus faciles à mettre en œuvre, car ils sont non récursifs et toujours stables (par définition). D'autre part, il est beaucoup plus difficile d'obtenir des réponses en phase linéaire et de contrôler les réponses en fréquence

globales avec des filtres RII. Cependant, des réponses en fréquence très pointues (bande de transition étroite) peuvent être facilement réalisées avec des filtres RII et cette caractéristique les rend appropriées pour une large gamme d'applications. Globalement, lorsqu'ils sont correctement conçus, les filtres RII sont capables d'avoir des réponses plus précises que celles obtenues par les filtres RIF. Mais le problème des filtres RII est l'instabilité et pour cela, les chercheurs ont adopté des solutions [5, 6, 7, 8].

Dans le développement initial de la conception du filtre RII, certaines techniques conventionnelles, basées sur des approximations de filtres analogiques, ont été présentées par J. M. Costa et al [9]. Mais ces techniques souffrent de l'effet de quantification et il n'y a aucun contrôle sur la largeur de transition. Après cela, une technique assistée par ordinateur utilisant l'optimisation basée sur le gradient a été explorée par T. Hinamoto et al [10]. Plus tard, pour donner la solution optimale pour le problème de surface ondulée sur la réponse lors de la conception des filtres RII, des techniques évolutionnaires ont été utilisées [1,11]. De nombreux auteurs ont proposé diverses méthodes pour la conception des filtres numériques bidimensionnels RII. Cependant, la plupart de leurs algorithmes ont donné des filtres instables [1,11, 12]. En ce qui concerne la stabilité, les méthodes précédentes, sont basées sur des techniques d'optimisation qui sont plus ou moins des approches d'essais-erreurs et ne peuvent pas toujours garantir la stabilité. Dans ce travail, la méthode proposée pour fixer la stabilité pendant le processus d'optimisation n'est pas une approche d'optimisation, mais basée sur les travaux de Ramamoorthy et al [5]. Ils exploitent le déterminant d'admittance d'entrée d'un réseau bi-variable (passif, sans perte et indépendant de la fréquence), qui est strictement un polynôme de Hurwitz. Ce polynôme est affecté au dénominateur d'une fonction de transfert analogique. Les éléments de réseau sont ensuite utilisés comme des variables d'optimisation garantissant la stabilité. La fonction de transfert du filtre discret bidimensionnel (numérique) correspondante est obtenue par la transformation bilinéaire.

Dans les méthodes basées sur l'optimisation, la conception des filtres numériques IIR bidimensionnels peut être formulée comme un problème de minimisation et résolue par diverses techniques d'optimisation ou par des algorithmes évolutionnaires. Les algorithmes génétiques [11], les réseaux de neurones [12] et l'optimisation par essais de

particules (PSO) [1] sont les plus populaires. Le PSO a été proposé pour la première fois par Russel Eberhart (ingénieur électricien) et James Kennedy (socio-psychologue) en 1995 [13]. Il s'inspire du comportement social des animaux évoluant en essaims tels que les poissons et les oiseaux. Comparé à d'autres algorithmes évolutionnaires, comme l'algorithme génétique (GA), le PSO est facile à mettre en œuvre et il y a peu de paramètres à ajuster. Il a été appliqué avec succès dans de nombreux domaines d'application, y compris l'optimisation des fonctions [14, 15]. Cependant, les études par S. Das et al [16] ont montré que le PSO initial avait des difficultés à contrôler l'équilibre entre l'exploration (étude globale du lieu de recherche) et l'exploitation (la recherche fine autour d'un optimum local). Il converge rapidement vers un optimum local dans les premières périodes d'itérations, il a des problèmes lorsqu'il s'agit d'atteindre une solution quasi-optimale. Pour remédier à ces problèmes, et améliorer les performances du PSO standard, plusieurs tentatives ont été faites, telles que l'hybridation de PSO avec d'autres algorithmes [17]. S. Dhabal et al [1] ont utilisé l'hybridation de deux variantes d'algorithmes d'optimisation, le PSO et le recuit stimulé (SA), pour la conception d'un filtre RII.

Dans cette étude, une nouvelle dérivée du PSO pour la conception d'un filtre RII bidimensionnel est proposée, l'algorithme appelé multi-PSO [18]. Il est basé sur le PSO Standard et caractérisé par la diversité des particules, sans diminuer la vitesse de convergence. Il utilise une nouvelle stratégie d'initialisation, cette stratégie est relayée à deux types d'essaims un essaim principal noté  $S_1$  et  $s$  essaims auxiliaires désignés par  $S_{2i}$ , où  $1 \leq i \leq s$ . Pour améliorer les performances de l'algorithme, l'espace de recherche est partitionné en plusieurs zones. Ensuite, chaque initialisation de  $S_{2i}$  est effectuée dans l'une de ces zones. Ce partitionnement permet une meilleure couverture de l'espace de recherche, conduisant ainsi à une meilleure exploration dans chaque zone séparément. Le tableau final avec toutes les meilleures positions regroupées d'essaims auxiliaires composera la population initiale de l'essaim principal. Une fois l'essaim principal initialisé, les essaims auxiliaires ne sont plus utilisés. L'essaim principal prend ensuite le relais et poursuit la recherche de la même manière que l'algorithme PSO standard jusqu'à ce qu'un critère d'arrêt soit atteint.

# Chapitre 1

<b>1. Éléments de base sur les signaux et systèmes bidimensionnels (2D)</b> .....	<b>05</b>
<b>1.1. Introduction</b> .....	<b>05</b>
<b>1.2. Les signaux déterministes 2D</b> .....	<b>05</b>
<b>1.2.1. Signaux bidimensionnels particuliers</b> .....	<b>06</b>
<b>1.3. Signaux bidimensionnels aléatoires</b> .....	<b>08</b>
<b>1.3.1. Caractéristique des signaux aléatoires bidimensionnels</b> .....	<b>09</b>
<b>1.4. Systèmes 2D</b> .....	<b>11</b>
<b>1.4.1. Propriétés principales</b> .....	<b>11</b>
<b>1.4.2. Les systèmes Linéaire invariants dans le temps</b> .....	<b>11</b>
<b>1.4.3. Système séparable</b> .....	<b>12</b>
<b>1.4.4. Stabilité des systèmes 2D</b> .....	<b>13</b>
<b>1.4.5. Réponse fréquentielle d'un système LTI</b> .....	<b>13</b>
<b>1.5. Transformée en z (TZ) 2D</b> .....	<b>14</b>
<b>1.5.1. Définition</b> .....	<b>14</b>
<b>1.5.2. Région de convergence</b> .....	<b>15</b>
<b>1.5.3 Propriétés de la transformée en z 2D</b> .....	<b>15</b>
<b>1.6. Transformée en z inverse 2D</b> .....	<b>16</b>
<b>1.7. Les Filtres 2D</b> .....	<b>17</b>
<b>1.7.1. Filtres à phase linéaire 2D</b> .....	<b>17</b>
<b>1.7.2. Caractéristiques des filtres 2D</b> .....	<b>18</b>
<b>1.7.3. Classification des filtres numériques 2D</b> .....	<b>21</b>
<b>1.7.4. Comparaison des filtres</b> .....	<b>21</b>
<b>1.8. Conclusion</b> .....	<b>22</b>

## 1. Éléments de base sur les signaux et systèmes bidimensionnels (2D)

### 1.1. Introduction

Le traitement et l'analyse des signaux 2D soulèvent de nombreuses questions théoriques et pratiques qui ne s'appliquent qu'à eux et que l'on ne retrouve pas en 1D. En effet, si certaines notions ou techniques (comme la transformée de Fourier) trouvent une extension naturelle aux signaux 2D, il n'en est pas de même pour la notion de causalité ou la fonction rationnelle de transfert 2D.

Dans ce chapitre, nous nous concentrerons sur les signaux 2D discrets. Cette restriction est une conséquence de l'importance actuelle des systèmes d'acquisition et de traitement numérique des signaux. De plus, l'accent sera mis sur les spécificités des signaux et des systèmes 2D par rapport à 1D. Nous aborderons d'abord le cas des signaux déterministes et leur représentation. Après cela, nous aborderons les signaux aléatoires 2D. Nous revenons à l'hypothèse de stationnarité et d'ergodicité, déjà bien connue en 1D et qui nous permettra d'étendre les caractéristiques statistiques les plus traditionnelles au 2D (statistiques d'ordre 2 et d'ordre supérieur à 2). La troisième partie sera consacrée aux systèmes, pour lesquels des questions majeures relatives au 2D sont soulevées (séparation, stabilité, causalité). Nous concluons enfin par les caractéristiques fréquentielles de ces signaux et systèmes en examinant les transformées en  $z$  2D [19].

### 1.2. Les signaux déterministes 2D

Un signal bidimensionnel déterministe discret  $x$  est une fonction définie sur l'ensemble  $\mathbb{Z}^2$  tel que :

$$x\{(k_1, k_2)\}_{(k_1, k_2) \in \mathbb{Z}^2} \quad (1.1)$$

On souligne que les signaux bidimensionnels 2D discrets, peuvent être en pratique manipulés dans une matrice formée de coefficients réels ou complexes, c'est-à-dire :

$$\mathbf{X} = \{x_{k_1, k_2}; 0 \leq k_1 \leq K_1 - 1, 0 \leq k_2 \leq K_2 - 1\} \quad (1.2)$$

Où  $K_1$  et  $K_2$  sont les paramètres de la taille de la matrice  $\mathbf{X}$  pour chacune des deux dimensions.

### 1.2.1. Signaux bidimensionnels particuliers :

Dans cette section, nous décrivons quelques signaux 2D qui sont traditionnellement utilisés.

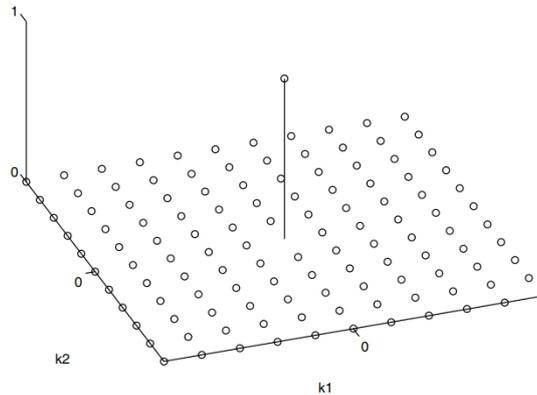


Figure 1.1 L'impulsion unitaire 2D

L'impulsion unitaire 2D ou séquence de Kronecker (voir Figure 1.1) est un signal important dans le traitement des signaux discrets. Il est défini dans le cas 2D par :

$$\delta(k_1, k_2) = \begin{cases} 1 & \text{pour } k_1 = k_2 = 0 \\ 0 & \text{pour autre} \end{cases} \quad (1.3)$$

On note que l'impulsion unitaire 2D correspond au produit de deux séquences de Kronecker 1D :

$$\delta(k_1, k_2) = \delta(k_1)\delta(k_2) \quad (1.4)$$

Un autre signal qui est beaucoup utilisé dans le traitement du signal 1D est le pas unitaire. Son extension en bidimensionnels est définie par :

$$U(k_1, k_2) = \begin{cases} 1 & \text{pour } k_1 \geq 0 \text{ et } k_2 \geq 0 \\ 0 & \text{pour } \text{autre} \end{cases} \quad (1.5)$$

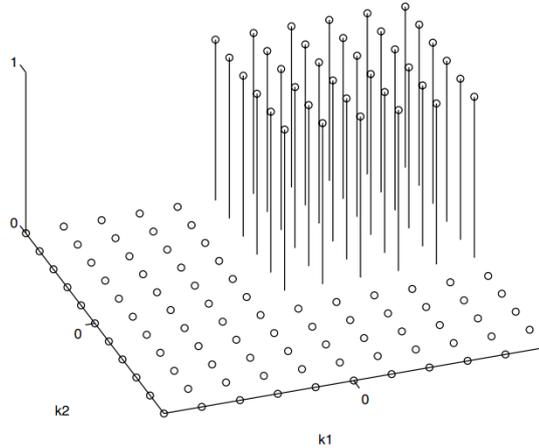


Figure 1.2 Le pas unitaire 2D

Dans ce cas aussi, on voit immédiatement que le signal 2D est le produit des deux pas d'unité 1D. Cependant, nous notons que le pas unitaire 2D illustré à la figure 1.2 est une fonction non nulle sur un seul quadrant.

Les deux signaux précédents sont des séquences particulières et appelés signaux séparables. Cette classe de signal 2D est définie par la relation suivante :

$$x(k_1, k_2) = x_1(k_1)x_2(k_2) \quad (1.6)$$

On peut montrer [20] que tout signal 2D de durée finie peut être écrit sous la forme d'une somme finie de signaux séparables. Les signaux séparables, bien que rarement rencontrés dans la pratique, peuvent cependant être utilisés lors du développement de systèmes 2D.

Les séquences périodiques constituent également une classe importante de signaux 2D (voir Figure 1.3). Cependant, la généralisation du cas 1D n'est pas immédiate, car la périodicité d'un signal doit être vérifiée dans les deux dimensions de la représentation.

Un signal 2D  $x(k_1, k_2)$  est périodique si :

$$x(k_1 + K_{11}, k_2 + K_{21}) = x(k_1, k_2) \quad (1.7)$$

$$x(k_1 + K_{12}, k_2 + K_{22}) = x(k_1, k_2) \quad (1.8)$$

Avec

$$K_{11}K_{22} - K_{12}K_{21} \neq 0$$

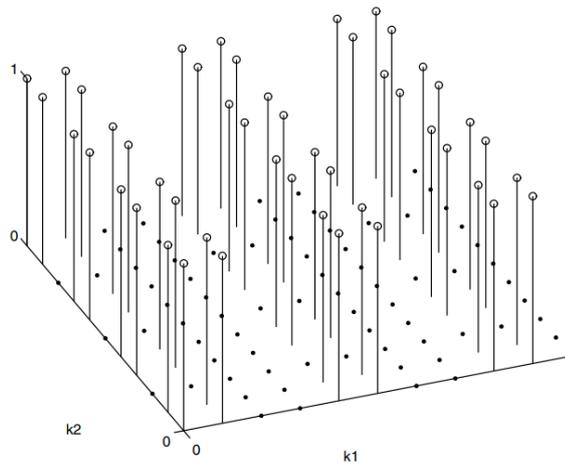


Figure 1.3: Exemple de signal périodique 2D (les périodes  $K_{11} = 4$ ,  $K_{21} = 0$  et  $K_{12} = 0$ ,  $K_{22} = 3$ )

### 1.3. Signaux bidimensionnels aléatoires

L'analyse des signaux aléatoires bidimensionnels est une extension de l'analyse des signaux monodimensionnels. En effet l'étude des monodimensionnels ne nécessite que l'étude des propriétés du deuxième ordre comme la fonction d'autocorrélation et la densité spectrale où les notions de phase ont moins d'importance que la répartition énergétique du signal. Mais la phase est un élément essentiel pour la reconstruction de la forme du signal 2D. Donc pour les études portées sur la recherche de la forme d'un objet dans une image ou sa reconstitution, on fait intervenir les moments d'ordre supérieur qui prennent en compte les notions de phase.

### 1.3.1. Caractéristique des signaux aléatoires bidimensionnels

Certains nombres des caractéristiques du signal aléatoire sont nécessaires pour être capables de définir d'autres concepts et entités manipulables [21].

L'étude d'un processus aléatoire 2D permet de définir quatre grandeurs fondamentales :

-La moyenne quadratique du processus (ou moment statistique du premier ordre) :

$$\mu_x(k_1, k_2) = E[X(k_1, k_2)] \quad (1.9)$$

-La variance

$$\sigma_x^2(k_1, k_2) = E[|X(k_1, k_2) - \mu_x(k_1, k_2)|^2] \quad (1.10)$$

-L'autocovariance de second ordre

$$\sigma_x^2(k_1, k_2) = E[|X(k_1, k_2) - \mu_x(k_1, k_2)|^2] \quad (1.11)$$

-L'autocovariance de premier ordre

$$C_x(k_1, k_2; l_1, l_2) = E[X_c(k_1, k_2)X_c^*(l_1, l_2)] \quad (1.12)$$

Avec

$$X_c(k_1, k_2) = X(k_1, k_2) - \mu_x(k_1, k_2) \quad (1.13)$$

Stationnarité :

Un processus stochastique est dit stationnaire au sens strict si toutes ses propriétés statistiques sont invariables selon une translation des coordonnées spatiales et elle est dite stationnaire jusqu'à l'ordre  $K$  si toutes ses propriétés statistiques jusqu'à l'ordre  $K$  sont invariables. Cependant, on considère souvent un autre type de stationnarité, qui est étroitement liée à des considérations énergétiques : on dit qu'un processus aléatoire est stationnaire au sens large (WSS) si sa moyenne est une constante :

$$\mu_x(k_1, k_2) = \mu \quad (1.14)$$

Et si la deuxième fonction d'autocovariance est invariante pour tout changement d'origine :

$$R_x(k_1, k_2; l_1, l_2) = R_x(k_1 + m_1, k_2 + m_2; l_1 + m_1, l_2 + m_2) = r_x(k_1 - l_1, k_2 - l_2) \quad (1.15)$$

Ergodicité :

Comme pour le traitement du signal 1D, il y a un problème avec l'estimation de caractéristiques d'un processus 2D. Rappelons simplement que l'ergodicité doit d'une part, se rattacher à un ordre donné de caractérisation d'un processus et d'autre part (même si on y fait moins souvent référence), au type de convergence de l'estimateur envisagé pour l'instant correspondant. La question réside dans la capacité d'estimer correctement, dans les critères de convergence donnés, un moment qui se définit à partir d'une valeur attendue d'un ensemble de résultats d'un processus aléatoire uniquement à l'aide d'un résultat (plus souvent un horizon fini) de ce processus. Par exemple, si le résultat d'un processus aléatoire 2D nous permet, à l'aide d'un calcul de moyenne spatiale (temporel dans le cas de 1D), pour estimer en bonne mesure la moyenne  $\mu$  du processus 2D, alors ce processus est dit ergodique moyen. De la même manière, à l'ordre 2, on parle d'autocorrélation du processus ergodique. En général, la convergence est envisagée dans une moyenne quadratique.

### 1.4. Systèmes 2D

Le traitement du signal, en 2D comme en 1D, consiste souvent en une ou plusieurs transformations d'un signal à l'aide d'appareils appelés systèmes. Un système réagit sur  $N$  séquences d'entrée  $x_1, x_2, \dots, x_N$  pour avoir une séquence de sortie  $y$  telle que [22] :

$$y = T[x_1, x_2, \dots, x_N] \quad (1.16)$$

Où  $T$  désigne l'opérateur associé au système. Formellement, ce dernier peut être équipé de toute règle permettant de déterminer la sortie en connaissant les entrées.

#### 1.4.1. Propriétés principales

Un système représenté par un opérateur  $T$  est dit linéaire si :

$$\forall x_1, x_2, \forall (c_1, c_2) \in \mathbb{C}^2: T[c_1x_1 + c_2x_2] = c_1T[x_1] + c_2T[x_2] \quad (1.17)$$

Toute séquence  $x$  peut être réécrite sous la forme suivante :

$$x(k_1, k_2) = \sum_{m_1=-\infty}^{+\infty} \sum_{m_2=-\infty}^{\infty} x(m_1, m_2) \delta(k_1 - m_1, k_2 - m_2) \quad (1.18)$$

#### 1.4.2. Les systèmes Linéaire invariants dans le temps :

Les systèmes les plus étudiés et utilisés dans le traitement du signal, même s'ils souffrent de la généralité, sont les systèmes dits invariants dans le temps (LTI). Pour ces systèmes, la réponse impulsionnelle ne dépend plus de la localisation en  $(m_1, m_2)$  de l'impulsion unitaire, mais uniquement du vecteur de translation entre cette impulsion et la position  $(k_1, k_2)$ . Ainsi, la relation, qui exprime la sortie du système, devient :

$$y(k_1, k_2) = \sum_{m_1=-\infty}^{+\infty} \sum_{m_2=-\infty}^{+\infty} x(m_1, m_2) h(k_1 - m_1, k_2 - m_2) \quad (1.19)$$

Cette relation, connue sous le nom de produit de convolution 2D, souligne le fait qu'un système LTI est complètement caractérisé par sa réponse impulsionnelle  $h(m_1, m_2)$  qui est invariante par translation

Nous noterons l'opérateur de convolution 2D par le symbole  $\otimes \otimes$  (le symbole  $\otimes$  représente traditionnellement l'opérateur de convolution 1D). La sortie d'un système LTI est écrite comme :

$$y(k_1, k_2) = [x \otimes \otimes h](k_1, k_2) \quad (1.20)$$

### 1.4.3. Système séparable

Un système LTI, dont la réponse impulsionnelle est une séquence 2D séparable, sa réponse impulsionnelle prend alors la forme suivante :

$$h(m_1, m_2) = h_1(m_1)h_2(m_2) \quad (1.21)$$

La sortie  $y$  d'un système séparable en réponse à une entrée  $x$  s'écrit :

$$\begin{aligned} y(k_1, k_2) &= \sum_{m_1=-\infty}^{+\infty} \sum_{m_2=-\infty}^{+\infty} x(m_1 - k_1, m_2 - k_2) h_1(m_1)h_2(m_2) \\ &= \sum_{m_1=-\infty}^{+\infty} h_1(m_1) \sum_{m_2=-\infty}^{+\infty} x(m_1 - k_1, m_2 - k_2)h_2(m_2) \end{aligned} \quad (1.22)$$

#### 1.4.4. Stabilité des systèmes 2D

La stabilité des systèmes LTI en 2D, comme en 1D, est une condition nécessaire à leur utilisation pratique. Il existe deux types de stabilité :

**Forte stabilité** : un système bidimensionnel est stable si pour toute entrée bornée  $x$  corresponde à une sortie bornée  $y$ . Par conséquent, nous parlons de stabilité BIBO (entrée bornée, sortie bornée). Pour un système BIBO, si  $|x(k_1, k_2)| \leq B \quad \forall (k_1, k_2)$  alors il existe  $B'$  tel que :  $|y(k_1, k_2)| \leq B' \quad \forall (k_1, k_2)$ . Comme dans le cas de 1D, nous pouvons montrer qu'une condition nécessaire et suffisante de stabilité est que la réponse impulsionnelle  $h$  du système puisse être absolument sommable [19] :

$$\sum_{m_1=-\infty}^{+\infty} \sum_{m_2=-\infty}^{+\infty} |h(m_1, m_2)| < +\infty \quad (1.23)$$

**Faible stabilité** : la stabilité en moyenne quadratique est une stabilité plus « faible » que la précédente. On dit qu'un système LTI est stable en moyenne quadratique si :

$$\sum_{m_1=-\infty}^{+\infty} \sum_{m_2=-\infty}^{+\infty} |h(m_1, m_2)|^2 < +\infty \quad (1.24)$$

#### 1.4.5. Réponse fréquentielle d'un système LTI

La réponse fréquentielle d'un système 2D représente le gain généralement complexe de la sortie du système pour une entrée aussi complexe composée d'harmonique [22].

Sois  $h(m_1, m_2)$  la réponse impulsionnelle d'un système LTI et  $x(m_1, m_2)$  une composante harmonique d'entrée de type exponentiel complexe :

$$x(m_1, m_2) = e^{j2\pi(\omega_1 m_1 + \omega_2 m_2)} \quad (1.25)$$

La sortie du système peut être déterminée par :

$$\begin{aligned}
 y(k_1, k_2) &= \sum_{m_1=-\infty}^{+\infty} \sum_{m_2=-\infty}^{+\infty} h(m_1, m_2) e^{j2\pi(\omega_1[k_1-m_1]+\omega_2[k_2-m_2])} = e^{j2\pi(\omega_1 k_1 + \omega_2 k_2)} \\
 &\cdot \left[ \sum_{m_1=-\infty}^{+\infty} \sum_{m_2=-\infty}^{+\infty} h(m_1, m_2) e^{-j2\pi(\omega_1 m_1 + \omega_2 m_2)} \right] = e^{j2\pi(\omega_1 k_1 + \omega_2 k_2)} \cdot \hat{h}(\omega_1, \omega_2)
 \end{aligned}
 \tag{1.26}$$

## 1.5. Transformée en z (TZ) 2D

### 1.5.1. Définition

La transformée en z pour un signal 2D nous permet de représenter une plus grande variété des signaux et des systèmes. De plus, elle nous permet également d'étudier les équations différentielles et surtout d'analyser la stabilité des systèmes 2D. La transformée en z est connue pour posséder plus de caractéristiques contrairement au FT [19].

La transformée en z d'une séquence  $x(k_1, k_2)$  est notée par  $X(z_1, z_2)$  et définie par

$$X(z_1, z_2) = \sum_{k_1=-\infty}^{+\infty} \sum_{k_2=-\infty}^{+\infty} x(k_1, k_2) z_1^{-k_1} z_2^{-k_2}
 \tag{1.27}$$

Où  $(z_1, z_2) \in \mathbb{C}^2$  La première difficulté dans les cas 2D. C'est l'espace pour évaluer  $(z_1, z_2)$  (quatre dimensions 4-D). Il est ainsi extrêmement difficile de visualiser des points ou des segments de  $X(z_1, z_2)$ .

La transformée en z 2D est une généralisation de la transformée de Fourier 2D, comme :

$$X(z_1, z_2) \Big|_{(z_1=e^{j2\pi\omega_1}, z_2=e^{j2\pi\omega_2})} = \sum_{n_1=-\infty}^{+\infty} \sum_{n_2=-\infty}^{+\infty} x(k_1, k_2) e^{-j2k_1\pi\omega_1} e^{-j2k_2\pi\omega_2}
 \tag{1.28}$$

### 1.5.2. Région de convergence

En représentant des variables complexes sous la forme  $z_1 = r_1 e^{-j2\pi\omega_1}$ ,  $z_2 = r_2 e^{-j2\pi\omega_2}$

Avec  $r_1 = |z_1|$ ,  $r_2 = |z_2|$  et le TZ 2D s'écrit donc :

$$X(z_1, z_2) = \sum_{k_1=-\infty}^{+\infty} \sum_{k_2=-\infty}^{+\infty} x(k_1, k_2) r_1^{-k_1} r_2^{-k_2} e^{-j2\pi\omega_1 k_1} e^{-j2\pi\omega_2 k_2} = \mathcal{FT}[x(k_1, k_2) r_1^{-k_1} r_2^{-k_2}] \quad (1.29)$$

La convergence FT 2D nécessite que le signal soit absolument sommable tel que :

$$\sum_{k_1=-\infty}^{+\infty} \sum_{k_2=-\infty}^{+\infty} |x(k_1, k_2)| r_1^{-k_1} r_2^{-k_2} < +\infty \quad (1.30)$$

La convergence de TZ 2D dépend donc des valeurs de  $r_1$  et  $r_2$ . Ainsi, la région  $R \subset \mathbb{C}^2$  pour laquelle  $X(z_1, z_2)$  converge est appelée région de convergence ou domaine de Reinhardt.

### 1.5.3 Propriétés de la transformée en z 2D

$$x(k_1, k_2) \leftrightarrow X(z_1, z_2), ROC: R_x$$

$$y(k_1, k_2) \leftrightarrow Y(z_1, z_2), ROC: R_y$$

Propriété 1 Linéarité :

$$ax(k_1, k_2) + by(k_1, k_2) \leftrightarrow aX(z_1, z_2) + bY(z_1, z_2), ROC: R_x \cap R_y$$

Propriété 2 Convolution :

$$x(k_1, k_2) \otimes \otimes y(k_1, k_2) \leftrightarrow X(z_1, z_2)Y(z_1, z_2), ROC: R_x \cap R_y$$

Propriété 3 Séquence séparable :

$$x(k_1)y(k_2) \leftrightarrow X(z_1)Y(z_2) \text{ Région convergence } |z_1| \in ROC \text{ de } X(z_1) \text{ et } |z_2| \in ROC \text{ de } X(z_2)$$

Propriété 4 Décalage :

$$x(k_1 - m_1, k_2 - m_2) \leftrightarrow X(z_1, z_2) z_1^{-m_1} z_2^{-m_2} \text{ Région convergence } R_x \text{ à l'exception } |z_1| = 0, +\infty, |z_2| = 0, +\infty$$

Propriété 5 Différentiation :

$$a) -k_1 x(k_1, k_2) \leftrightarrow \frac{\partial X(z_1, z_2)}{\partial z_1} z_1 \text{ Région convergence } R_x$$

$$b) -k_2 x(k_1, k_2) \leftrightarrow \frac{\partial X(z_1, z_2)}{\partial z_2} z_2 \text{ Région convergence } R_x$$

Propriété 6 Symétrie :

$$a) x^*(k_1, k_2) \leftrightarrow X^*(z_1^*, z_2^*) \text{ Région convergence } R_x$$

$$b) x(-k_1, -k_2) \leftrightarrow X(z_1^{-1}, z_2^{-1}) \text{ Région convergence } |z_1^{-1}|, |z_2^{-1}| \text{ Dans } R_x$$

Propriété 7 Transformation linéaire des variables :

$$x(k_1, k_2) = y(m_1, m_2) |_{m_1=k_1 l_2 + k_2 l_2, m_2=k_1 l_3 + k_2 l_4} \leftrightarrow Y(z_1, z_2) = X(z_1^{l_1} z_2^{l_3}, z_1^{l_2} z_2^{l_4}) \text{ ROC } |z_1^{l_1} z_2^{l_3}|, |z_1^{l_2} z_2^{l_4}| \text{ dans } R_x$$

## 1.6. Transformée en z inverse 2D

Comme dans le cas unidimensionnel, nous pouvons déterminer la relation de transformée en z inverse qui exprime  $x(k_1, k_2)$  en fonction de  $X(z_1, z_2)$  et de son ROC.

$$x(k_1, k_2) = \frac{1}{(2\pi j)^2} \oint_{C_1} \oint_{C_2} X(z_1, z_2) z_1^{-k_1-1} z_2^{-k_2-1} dz_1 dz_2 \quad (1.31)$$

Où  $C_1$  et  $C_2$  sont deux contours dans le ROC de  $X(z_1, z_2)$

Pour calculer la transformée inverse en z sans évaluation de l'intégration de contour. Des méthodes comme fraction partielle et les propriétés de transformée en z sont utilisées. Malheureusement pour les signaux 2D, la méthode de fraction partielle n'est pas une procédure générale. La factorisation de tout polynôme 1D  $D(z)$  comme produit des polynômes d'ordre inférieur est garantie par les théorèmes fondamentaux de l'algèbre. Cependant, un polynôme 2D ne peut pas en général être factorisé comme produit des polynômes d'ordre inférieur. Par conséquent, il n'est pas possible d'utiliser une procédure analogue.

### 1.7. Les Filtres 2D

Les principales fonctions du filtre sont d'éliminer les parties indésirables d'un signal, comme le bruit aléatoire, ou extraire celles qui sont utiles. Par exemple, une image dégradée par un bruit de bande peut être améliorée sans brouiller ses bords. Une bande magnétique peut être rendue plus lisible en supprimant certains signaux de grande amplitude et basse fréquence [2]. Les filtres numériques sont classés en deux groupes : filtres à réponse impulsionnelle infinie (RII) ou filtres récursifs et filtres à réponse impulsionnelle finie (RIF) ou filtres non récursifs [4]. Les filtres RIF sont généralement plus faciles à implémenter, car ils sont non récursifs et toujours stables (par définition). D'autre part, il est beaucoup plus difficile d'obtenir des réponses de phase linéaire et contrôler les réponses en fréquence globales avec des filtres RII. Cependant, des bandes de transition très nettes et étroites peuvent être facilement réalisées avec des filtres RII. Cette fonctionnalité les rend adaptés à une large gamme d'applications. Généralement, les filtres RII sont capables d'avoir des meilleures réponses que celles fournies par les filtres RIF. Mais les filtres RII souffrent de leur problème d'instabilité.

#### 1.7.1. Filtres à phase linéaire 2D

Un filtre numérique  $h(k_1, k_2)$  est dit à zéro phase (à phase linéaire) lorsque sa réponse en fréquence est une fonction réelle de sorte que :

$$H(\omega_1, \omega_2) = H^*(\omega_1, \omega_2) \quad (1.32)$$

D'après les propriétés de symétrie de la transformée de Fourier, l'équation (1.32) est équivalente dans le domaine spatial à ce qui suit :

$$h(k_1, k_2) = h^*(-k_1, -k_2) \quad (1.33)$$

Si on considère que  $h(k_1, k_2)$  réelle, on peut écrire :

$$h(k_1, k_2) = h(-k_1, -k_2) \quad (1.34)$$

L'équation (1.34) montre que la réponse impulsionnelle d'un filtre à phase linéaire est symétrique par rapport à l'origine. Dans les applications telles que le traitement de la parole, la

phase linéaire d'un filtre n'est pas très critique. Le système auditif humain répond aux caractéristiques spectrales de grandeur de brève durée, ainsi la forme d'onde de la parole peut parfois changer rigoureusement sans que l'auditeur humain la distingue de l'original. Dans le traitement d'images, la caractéristique de phase linéaire semble être plus importante. Notre monde visuel se compose des lignes, des gribouillages, etc. Une phase non-linéaire déformée les différents composants de fréquences qui composent les lignes et les gribouillages.

### 1.7.2. Caractéristiques des filtres 2D

Comme les filtres numériques 1D, les filtres numériques 2D sont généralement spécifiés dans le domaine fréquentiel. On peut écrire :

$$H(\omega_1, \omega_2) = H(\omega_1 + 2\pi, \omega_2) = H(\omega_1, \omega_2 + 2\pi): \forall (\omega_1, \omega_2) \in [-\pi, \pi]^2 \quad (1.35)$$

On dit qu'un filtre a une réponse fréquentielle circulairement symétrique, si  $H(\omega_1, \omega_2)$  est une fonction de  $\omega_1^2 + \omega_2^2$  avec  $\sqrt{\omega_1^2 + \omega_2^2} \leq \pi$  et reste constant en dehors de l'intervalle  $[-\pi, \pi]$ . Un filtre a une réponse impulsionnelle circulairement symétrique, si  $h(k_1, k_2)$  est une fonction de  $k_1^2 + k_2^2$ . Alors la symétrie circulaire de  $H(\omega_1, \omega_2)$  implique la symétrie circulaire de  $h(k_1, k_2)$ . Cependant la symétrie circulaire de  $h(k_1, k_2)$ , n'implique pas la symétrie circulaire de  $H(\omega_1, \omega_2)$ .

Les réponses fréquentielles des filtres passe-bas, passe-haut, passe-bande, et coupe-bande idéaux circulairement symétriques sont affichées dans les Figures 1.4 (a), (b), (c), et (d), respectivement. Les régions grises dans les figures ont l'amplitude 1, et les régions en blanc ont l'amplitude [22].

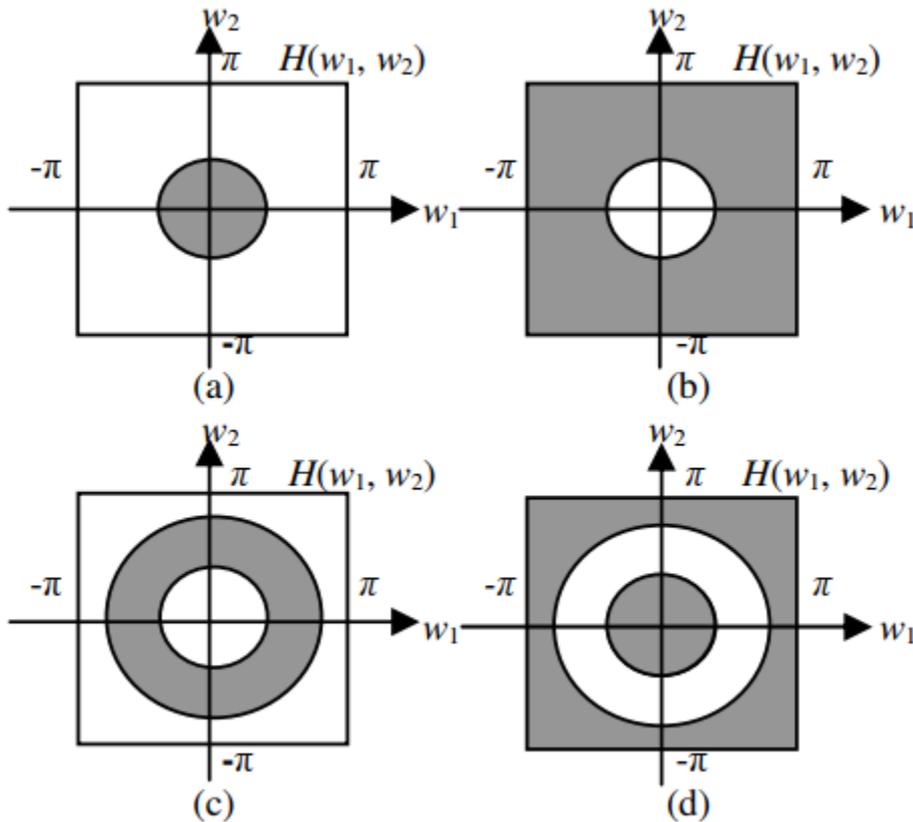


Figure 1.4 : les réponses fréquentielles des filtres idéaux circulairement symétriques. (a) Filtre passe-bas ;(b) filtre passe-haut ;(c) filtre passe-bande ;(d) filtre coupe-bande

-la réponse impulsionnelle pour le filtre passe bas est :

$$h_{lp}(k_1, k_2) = \frac{R}{2\pi\sqrt{k_1^2 + k_2^2}} J_1(R\sqrt{k_1^2 + k_2^2}) \quad (1.36)$$

-la réponse impulsionnelle pour le filtre passe haut est :

$$h_{hp}(k_1, k_2) = \delta(k_1, k_2) - h_{lp}(k_1, k_2) \quad (1.37)$$

-la réponse impulsionnelle pour le filtre passe bande est :

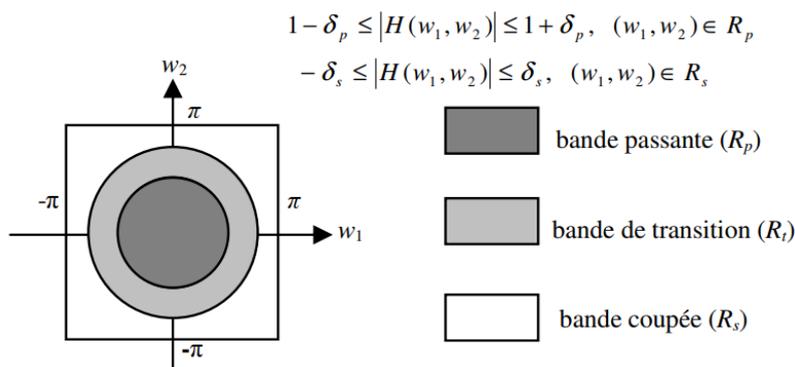
$$h_{bp}(k_1, k_2) = \frac{R_2}{2\pi\sqrt{k_1^2 + k_2^2}} J_1(R_2\sqrt{k_1^2 + k_2^2}) \quad (1.38)$$

-la réponse impulsionnelle pour le filtre coupe bande est :

$$h_{bs}(k_1, k_2) = \delta(k_1, k_2) - h_{bp}(k_1, k_2) \quad (1.39)$$

où  $h_{lp}(k_1, k_2)$ ,  $h_{hp}(k_1, k_2)$ ,  $h_{bp}(k_1, k_2)$  et  $h_{bs}(k_1, k_2)$  représentent les filtres passe-bas, passe-haut, passe-bande, et coupe-bande, respectivement, et  $J_1(x)$  est la fonction Bessel de la première espèce et premier ordre [22]

La fonction de transfert  $H(\omega_1, \omega_2)$  est en général une fonction complexe de  $(\omega_1, \omega_2)$ , on doit spécifier l'amplitude et la phase de  $H(\omega_1, \omega_2)$ . Cependant, la phase est linéaire généralement, ainsi nous devons spécifier seulement la grandeur de l'amplitude. La méthode la plus utilisée généralement avec les caractéristiques de l'amplitude s'appelle le schéma de tolérance. Pour illustrons ce schéma, considérons les caractéristiques d'un filtre passe-bas. Dans le cas idéal, un filtre passe-bas a seulement une région de bande passante et une région de bande coupée. Dans la pratique, une bande transition entre les deux régions existe ne peut pas être négligée, la région de bande passante correspond  $(\omega_1, \omega_2) \in R_p$  et la région de bande coupée correspond  $(\omega_1, \omega_2) \in R_s$ , suivant les indications de la figure 1.5. La région de fréquence entre la bande passante et la bande coupée s'appelle la bande de transition [22].



**Figure 1.5** schéma de tolérance des caractéristiques de filtre passe-bas 2D.

Idéalement, l'amplitude est unité dans la région de bande passante et zéro dans la région de bande coupée. Dans la pratique, on requiert de  $1 - \delta_p \leq |H(\omega_1, \omega_2)| \leq 1 + \delta_p$  pour  $(\omega_1, \omega_2) \in R_p$  et  $|H(\omega_1, \omega_2)| \leq \delta_s$  pour  $(\omega_1, \omega_2) \in R_s$ . Les constants  $\delta_p$  et  $\delta_{ps}$  s'appellent la tolérance de bande passante et la tolérance de bande coupée, respectivement. Un filtre

passé-bas est spécifié par  $\delta_p$ ,  $\delta_{ps}$ ,  $R_p$  et  $R_s$ . D'autres filtres peuvent également être spécifiés d'une façon analogue.

### 1.7.3. Classification des filtres numériques 2D

Les filtres numériques 2D sont classés en deux groupes. Dans le premier groupe,  $h(k_1, k_2)$  est une séquence finie, ainsi les filtres dans ce groupe s'appellent les filtres à réponse impulsionnelle finie (RIF). Dans le deuxième groupe,  $h(k_1, k_2)$  est une séquence infinie, et dans ce groupe ils sont appelés les filtres à réponse impulsionnelle infinie (RII), comme dans 1D, les filtres de RIF diffèrent considérablement dans la synthèse et l'implémentation des filtres RII.

Un filtre à réponse impulsionnelle infinie a une séquence  $h(k_1, k_2)$  qui est infinie. En conséquence, un filtre RII diffère à quelques majeurs aspects d'un filtre RIF. Par exemple, la stabilité n'est jamais un problème pour des filtres RIF dans la synthèse et l'implémentation, alors que c'est un problème très important pour des filtres RII. D'une part, un filtre RII exige typiquement d'un nombre sensiblement plus petit des coefficients pour répondre à une caractéristique particulière de l'amplitude que fait un filtre RIF. Comme des filtres RIF, l'utilisation des filtres RII comporte des caractéristiques, la synthèse et l'implémentation de filtre. Dans le cahier des charges de filtre RII, nous devons spécifier l'amplitude et la phase de la réponse fréquentielle  $H(\omega_1, \omega_2)$ . Pour un filtre RIF, il est très facile de réaliser un filtre avec phase linéaire, cependant, le contrôle de phase est très difficile, en conséquence, seulement la réponse de l'amplitude est spécifiée quand un filtre RII est synthétisé. La caractéristique de phase du filtre en résultant est alors considérée comme acceptable [22].

### 1.7.4. Comparaison des filtres

Les filtres RIF sont généralement plus faciles à implémenter, car ils sont non récursifs et toujours stables (par définition). D'autre part, il est beaucoup plus difficile d'obtenir des réponses en fréquence à phase linéaire et contrôlable avec les filtres RII. Cependant, des bandes de transition très nettes et étroites peuvent être facilement réalisées avec les filtres RII. Cette fonctionnalité les rend adaptés à une large gamme d'applications. Généralement, ils sont capables d'avoir des réponses meilleures que celles fournies par les filtres FIR [2]. Mais ils souffrent de leur problème d'instabilité. Pour y remédier à ce problème, certains chercheurs ont adopté des solutions qui sont révélées dans la littérature [5-8].

La synthèse et l'implémentation des filtres 2D RIF et RII sont des domaines de recherche actifs. Les différences importantes entre les filtres 1D et 2D existent également dans la synthèse et l'implémentation des filtres numériques. Dans 1D, il y a des méthodes pratiques pour la synthèse des filtres optimaux. Dans le 2D, des méthodes pratiques qui peuvent être sûrement employées pour synthétiser les filtres optimaux de RIF ou RII ne sont pas encore développées. Dans 1D, l'implémentation en cascade et en parallèles est des méthodes générales de réalisation pour des fonctions transferts rationaux. Dans le 2D, l'implémentation en cascade et en parallèles ne peuvent pas généralement être utilisée pour la réalisation des fonctions transferts rationaux. La synthèse et l'implémentation sont plus complexes pour les filtres numériques 2D que pour les filtres 1D [22].

### 1.8. Conclusion

Ce chapitre était consacré à l'étude des signaux 2D, des filtres 2D à phase linéaire, et les caractéristiques des filtres 2D dans le domaine fréquentiel. L'analyse d'un filtre 2D RII qui peut être plus difficile que d'un filtre RIF à cause des propriétés de stabilité et la phase linéaire, qui ne sont pas un problème dans la synthèse des filtres RIF.

# Chapitre 2

<b>2. Synthèse des filtres 2D à Réponse Impulsionnelle Infinie (RII).....</b>	<b>24</b>
<b>2.1. Introduction.....</b>	<b>24</b>
<b>2.2. Définition .....</b>	<b>24</b>
<b>2.3. Méthodes de synthèse des filtres RII 2D.....</b>	<b>25</b>
<b>2.4. Problèmes de stabilité .....</b>	<b>30</b>
<b>2.5. Conclusion.....</b>	<b>35</b>

## 2. Synthèse des filtres 2D à Réponse Impulsionnelle Infinie (RII)

### 2.1. Introduction

Dans ce chapitre, nous examinons les méthodes de conception qui impliquent l'application des techniques d'optimisation. Ces méthodes sont à la fois flexibles et polyvalentes et peuvent être utilisées pour obtenir une variété de caractéristiques de filtre. Cependant, il convient de mentionner, que la quantité de calcul peut parfois être considérable.

### 2.2. Définition

Le problème de synthèse d'un filtre RII est de déterminer une fonction de transfert rationnelle et stable avec une région de support qui répond aux caractéristiques de filtre. En d'autres termes, on doit déterminer une procédure de calcul de filtre stable qui est récursivement calculable. Dans la discussion du problème de synthèse de filtre, il est utile d'utiliser une fonction de transfert et une procédure de calcul l'un pour l'autre. Cependant, une fonction de transfert rationnelle donnée  $H(z_1, z_2)$  peut mener à beaucoup de procédures de calcul différentes [4]. Pour rendre la relation unique, nous adopterons une convention dans l'expression de  $H(z_1, z_2)$ . Spécifiquement, nous supposerons que  $a(0,0)$  de (2.1) est toujours 1, ainsi  $H(z_1, z_2)$  sera alors de la forme :

$$H(z_1, z_2) = \frac{\sum_{(k_1, k_2) \in R_b} b(k_1, k_2) z_1^{-k_1} z_2^{-k_2}}{1 + \sum_{(k_1, k_2) \in R_a - (0,0)} a(k_1, k_2) z_1^{-k_1} z_2^{-k_2}} \quad (2.1)$$

Avec  $a(k_1, k_2)$  et  $b(k_1, k_2)$  sont les coefficients de filtre RII. La première étape dans la synthèse d'un filtre RII est de déterminer les régions de  $R_a$  et  $R_b$ . Les régions du support de  $a(k_1, k_2)$  et  $b(k_1, k_2)$  sont déterminées par plusieurs considérations, comme le système en résultant doit être récursivement calculable.

### 2.3. Méthodes de synthèse des filtres RII 2D

- **Algorithme de moindre puissance** : Dans cette approche, une fonction objective sous la forme d'une somme de fonctions d'erreur élémentaires, chacune élevée à la puissance  $p$ , est d'abord formulée puis minimisée en utilisant l'une des nombreuses méthodes d'optimisation sans contrainte disponibles [23,24].

- Formulation du Problème :

Considérez la fonction de transfert :

$$H(z_1, z_2) = H_0 \prod_{k=1}^K \frac{N_k(z_1, z_2)}{D_k(z_1, z_2)} = H_0 \prod_{k=1}^K \frac{\sum_{l=0}^{L_{1k}} \sum_{m=0}^{M_{1k}} a_{lm}^{(k)} z_1^{-l} z_2^{-m}}{\sum_{l=0}^{L_{2k}} \sum_{m=0}^{M_{2k}} b_{lm}^{(k)} z_1^{-l} z_2^{-m}} \quad (2.2)$$

Où  $N_k(z_1, z_2)$  et  $D_k(z_1, z_2)$  sont des polynômes d'ordre égal ou inférieur à 2 et  $H_0$ , est une constante.

$$x = [a^T \quad b^T \quad H_0]^T \quad (2.3)$$

Où

$$a^T = [a_{10}^{(1)} a_{20}^{(1)} \dots a_{L_{11}M_{11}}^{(1)} a_{10}^{(2)} a_{20}^{(2)} \dots a_{L_{12}M_{12}}^{(1)} \dots a_{L_{1K}M_{1K}}^{(K)}]$$

$$b^T = [b_{10}^{(1)} b_{20}^{(1)} \dots b_{L_{21}M_{21}}^{(1)} b_{10}^{(2)} b_{20}^{(2)} \dots b_{L_{22}M_{22}}^{(2)} \dots b_{L_{2K}M_{2K}}^{(K)}]$$

$a^T$  et  $b^T$  : sont deux vecteurs dont ces éléments sont les coefficients de  $N_k(z_1, z_2)$  et  $D_k(z_1, z_2)$ , respectivement. La tâche de conception à accomplir consiste à trouver un vecteur de paramètres  $x$  qui minimise la plus petite  $p^{ieme}$  fonction objectif  $J(x)$  définie par :

$$J(x) = \sum_{k_1=1}^{K_1} \sum_{k_2=1}^{K_2} [M(k_1, k_2) - M_I(n_1, n_2)]^p \quad (2.4)$$

Avec :

$$M(k_1, k_2) = |H(e^{j\omega_1 k_1 T_1}, e^{j\omega_2 k_2 T_2})|, k_1 = 1, \dots, K_1 \text{ et } k_2 = 1, \dots, K_2$$

$K_1$  et  $K_2$  sont des échantillons de la réponse en amplitude du filtre sur les deux axes.

$M_I(k_1, k_2)$  : Représente la réponse en amplitude désirée et  $p$  est un entier pair positif.

**- Algorithmes d'optimisation Quasi-Newton** : le problème de conception décrit ci-dessus peut être résolu en utilisant l'un quelconque des algorithmes d'optimisation standard sans contrainte. Une classe d'algorithmes de ce type qui s'est avérée très polyvalente, efficace et robuste est la classe des algorithmes quasi-Newton [23, 24]. Celles-ci sont basées sur le principe que le point minimum  $x^*$  d'une fonction quadratique  $F(x)$  de  $N$  variables peut être obtenue en appliquant la correction.

$$\delta = -H^{-1}g \quad (2.5)$$

Pour un point arbitraire  $x$  on a :

$$x^* = x + \delta \quad (2.6)$$

Ou le vecteur :

$$g = \nabla f(x) \quad (2.7)$$

Et la matrice  $N \times N$ .

$$H = \begin{bmatrix} \frac{\partial^2 f(x)}{\partial x_1^2} & \frac{\partial^2 f(x)}{\partial x_1 \partial x_2} & \dots & \frac{\partial^2 f(x)}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f(x)}{\partial x_1 \partial x_2} & \frac{\partial^2 f(x)}{\partial x_2^2} & \dots & \frac{\partial^2 f(x)}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f(x)}{\partial x_n \partial x_1} & \frac{\partial^2 f(x)}{\partial x_n \partial x_2} & \dots & \frac{\partial^2 f(x)}{\partial x_n^2} \end{bmatrix}$$

(2.8)

Où  $g$  et  $H$  sont le vecteur de gradient et la matrice Hessien de  $f(x)$  au point  $x$ , respectivement. L'algorithme de base quasi-Newton appliqué à notre problème de conception est le suivant :

Etape 1 : entrez  $x_0$  et mètre  $S_0 = I_N$  ou  $I_N$  est la matrice unité de  $N \times N$  et  $N$  présente la dimension de  $x$ , on met  $k = 0$  et on calcule  $g_0 = \nabla J(x_0)$ .

Etape 2 : mètre  $d_k = S_k g_k$  et cherche  $\alpha_k$ , la valeur de  $\alpha$  minimise  $J(x_k + \alpha d_k)$ .

Etape 3 : mètre  $\delta_k = \alpha_k d_k$  et  $x_{k+1} = x_k + \delta_k$ .

Etape 4 : si  $\|\delta_k\| < \epsilon$  alors  $x^* = x_{k+1}$ ,  $J(x^*) = J(x_{k+1})$  et stop.

Etape 5 : calculer  $g_{k+1} = \nabla J(x_{k+1})$  et mètre  $\gamma_k = g_{k+1} - g_k$ .

Etape 6 : calculer  $S_{k+1} = S_k + C_k$ ,  $C_k$  est matrice de correction.

Etape 7 : mètre  $k = k + 1$  et allez à Etape 2.

$x_0$  est le vecteur du point initial et  $\epsilon$  une tolérance qui dépend de l'application. Aux étapes 1 et 5, le vecteur gradient  $\nabla J$  est requis. De l'Eq. (2.4), nous avons :

$$\nabla J(x) = \left[ \frac{\partial J}{\partial x_1} \quad \dots \quad \frac{\partial J}{\partial x_N} \right]$$

(2.9)

Où.

$$\frac{\partial J}{\partial x_1} = p \sum_{n_1=1}^{N_1} \sum_{n_2=1}^{N_2} [M(n_1, n_2) - M_I(n_1, n_2)]^{p-1} \frac{\partial M(n_1, n_2)}{\partial x_k}$$

La matrice de correction  $C_k$  requise à l'étape 6 peut être calculée à l'aide de la formule de Davidon-Fletcher-Powell (DFP) comme ceci :

$$C_k = \frac{\delta_k \delta_k^T}{\delta_k^T \gamma_k} - \frac{S_k \gamma_k \gamma_k^T S_k}{\gamma_k^T S_k \gamma_k} \quad (2.10)$$

-Minimax méthode : dans la conception de filtres récurrents et non récurrents, les méthodes d'optimisation minimax sont souvent préférées car leur application minimise le maximum de l'erreur d'approximation. Dans cette section, la conception des filtres récurrents est formulée comme un problème minimax.

- Formulation du problème : considérons la fonction de transfert  $H(z_1, z_2)$  donnée par l'équation. (2.1). Le filtre doit avoir une réponse en amplitude qui est approximativement symétrique de manière circulaire,  $H_1(z_1, z_2)$  avec un dénominateur séparable [25]. Par conséquent, la fonction de transfert prend la forme :

$$H_1(z_1, z_2) = H_0 \prod_{k=1}^K \frac{z_1 z_2 + (z_1 z_2)^{-1} + \alpha_k (z_1 + z_1^{-1} + z_2 + z_2^{-1}) + z_1^{-1} z_2 + z_1 z_2^{-1} + \beta_k}{(1 + c_k z_1^{-1} + d_k z_1^{-2})(1 + c_k z_2^{-1} + d_k z_2^{-2})} \quad (2.11)$$

et si le numérateur de  $H_1(z_1, z_2)$  est également séparable, alors la fonction de transfert prend la forme :

$$H_2(z_1, z_2) = H_0 \prod_{k=1}^K \frac{(1 + \alpha_k z_1^{-1} + \beta_k z_1^{-2})(1 + \alpha_k z_2^{-1} + \beta_k z_2^{-2})}{(1 + c_k z_1^{-1} + d_k z_1^{-2})(1 + c_k z_2^{-1} + d_k z_2^{-2})} \quad (2.12)$$

La réponse en amplitude du filtre représenté par  $H_1(z_1, z_2)$  est donnée par :

$$M_1(x, \omega_1, \omega_2) = \prod_{k=1}^K \frac{N_k}{D_k} \quad (2.13)$$

Avec.

$$N_k = |4 \cos \omega_1 T_1 \cos \omega_2 T_2 + 2\alpha_k (\cos \omega_1 T_1 + \cos \omega_2 T_2) + \beta_k|$$

$$D_k = D_{k1} D_{k2}$$

$$D_{ki} = [1 + c_k^2 + d_k^2 + 2d_k (\cos^2 \omega_i T_i - 1) + 2c_k (1 + d_k) \cos \omega_i T_i]^{\frac{1}{2}}$$

$$i = 1, 2$$

$$x = [\alpha_1 \beta_1 c_1 d_1 \cdots \alpha_k \beta_k c_k d_k H_0]$$

Algorithmes d'optimisation Minimax :

Le problème décrit dans la section précédente peut être résolu en formulant une fonction objective  $\Phi(x)$  en termes de  $L_p \text{ norm}$  de la fonction d'erreur, puis en minimisant  $\Phi(x)$  pour des valeurs croissantes de  $p$ . Une telle fonction objective peut être obtenue tel que :

$$\Phi(x) = F(x) \left\{ \sum_{i=1}^m \left[ \frac{|E(x, \omega_{1i}, \omega_{2i})|}{F(x)} \right]^p \right\}^{1/p} \quad (2.14)$$

où  $F(x)$  est donné par l'équation. (2.12). Un algorithme minimax basé sur  $\Phi(x)$  mentionné en [26] :

Etape 1 : introduire la valeur de  $x_0$  et mettre  $k = 1, p = 2, \mu = 2, F_0 = 10^{99}$ .

Etape 2 : initialiser la pulsation  $\omega_{1i}, \omega_{2i}$  pour  $i = 1, 2, \dots, m$ .

Etape 3 : utilisation de  $\bar{x}_{k-1}$  pour initialiser le point, minimiser  $\Phi(x)$  et mettre  $F_x = F(\bar{x}_k)$ .

Etape 4 : si  $|F_{k-1} - F_k| < \varepsilon$ , alors afficher  $x^* = \bar{x}_k$ ,  $F_k$  et stop sinon mettre  $p = \mu p$ ,  $k = k + 1$  allez à l'étape 3.

La minimisation à l'étape 3 peut être effectuée à l'aide de l'algorithme décrit ci-dessus. Le gradient de  $F(x)$  est donné par [26].

$$\nabla\Phi(x) = \left\{ \sum_{i=1}^m \left[ \frac{|E(x, \omega_{1i}, \omega_{2i})|}{F(x)} \right]^p \right\}^{(1/p)-1} \times \left\{ \sum_{i=1}^m \left[ \frac{|E(x, \omega_{1i}, \omega_{2i})|}{F(x)} \right]^{p-1} \right\} \nabla|E(x, \omega_{1i}, \omega_{2i})| \quad (2.15)$$

L'algorithme précédent donne d'excellents résultats sauf qu'il nécessite une quantité considérable de calculs.

#### 2.4. Problèmes de stabilité

Un filtre numérique doit être stable afin d'assurer que toute entrée bornée produira une sortie bornée. Sinon, le filtre ne sera pas utilisable pour une application pratique. Lorsqu'un filtre est en cours de conception, il est possible d'obtenir une fonction de transfert qui satisfait aux spécifications de réponse en amplitude et en phase requises alors que le filtre est instable. D'autre part, un filtre qui a été conçu pour satisfaire aux conditions nécessaires à la stabilité et à cause de la non-précision ou calculs mathématique peut devenir instable.

La stabilité peut être étudiée dans le domaine de fréquence par l'examen de la fonction de transfert et on peut faire cette étude dans l'espace d'état par l'examen de l'équation d'état.

$$H(z_1, z_2) = \frac{N(z_1, z_2)}{D(z_1, z_2)} \quad (2.16)$$

Où

$$N(z_1, z_2) = \sum_{i=0}^{K_1} \sum_{j=0}^{K_2} a_{ij} z_1^{-i} z_2^{-j}$$

Et

$$D(z_1, z_2) = 1 + \sum_{i=0}^{K_1} \sum_{j=0}^{K_2} b_{ij} z_1^{-i} z_2^{-j}, \quad b_{00} = 1$$

Si les deux polynômes  $N(z_1, z_2)$  et  $D(z_1, z_2)$  n'ont aucun facteur commun ou d'une autre manière s'il n'existe aucun point  $(z_1, z_2)$  où les deux polynômes assument une valeur de zéro, on dit que les polynômes  $N(z_1, z_2)$  et  $D(z_1, z_2)$  sont indépendants.

- Les propriétés de la stabilité :

Un filtre bidimensionnel caractérisé par sa fonction de transfert est stable si et seulement si la réponse impulsionnelle  $h(k_1, k_2)$  du filtre est absolument sommable [4]

$$\text{Un filtre 2D numérique stable (BIBO)} \quad \longrightarrow \quad \sum_{i=0}^{\infty} \sum_{j=0}^{\infty} |h(i, j)| < \infty \quad (2.17)$$

Alors que l'équation (2.17) est une condition nécessaire et suffisante pour une stabilisation BIBO, mais l'évaluation de la double sommation est très peu pratique puisque la réponse impulsionnelle n'est pas en général facile à obtenir à partir de la fonction de transfert. Une autre approche pour déterminer si le filtre 2D numérique est stable ou non est d'utiliser le théorème de stabilité de Shanks.

**Théorème 1(Shanks) :** le filtre numérique 2D représenté par la fonction de transfert (2.16) est stable BIBO si :

$$D(z_1, z_2) \neq 0 \text{ for } (z_1, z_2) \in \bar{U}^2 \quad (2.18)$$

D'où  $\bar{U}^2$  est l'unité disque définie par :

$$\bar{U}^2 = \{(z_1, z_2) : |z_1| \leq 1, |z_2| \leq 1\} \quad (2.19)$$

La condition de Shanks, bien que suffisante pour la stabilité du BIBO, n'est pas toujours nécessaire,

- Analyse de la stabilité par l'utilisation de l'espace d'état :

Les propriétés de stabilité des filtres numériques 2D peuvent également être examinées, par utilisation de la représentation de l'espace d'état. Certains avantages sont obtenus grâce à l'utilisation de cette représentation : d'abord, toutes les conditions de stabilité qui sont énoncées en termes des propriétés spectrales de certaines matrices et par conséquent, la matrice standard des méthodes d'analyse peut être utilisée pour vérifier la stabilité des filtres. Deuxièmement, le théorème de Lyapunov, qui se révèle être très utile dans l'analyse de la stabilité du 1D, peut être généralisé sur le cas 2D.

Un filtre numérique représenté par la fonction de transfert de l'équation (2.16) peut être également représenté par l'espace d'état et on peut écrire :

$$H(z_1, z_2) = \frac{N(z_1, z_2)}{D(z_1, z_2)} = \mathbf{C} \begin{bmatrix} z_1 \mathbf{I}_N - \mathbf{A}_1 & -\mathbf{A}_2 \\ -\mathbf{A}_3 & z_2 \mathbf{I}_M - \mathbf{A}_4 \end{bmatrix}^{-1} \mathbf{B} + \mathbf{D} \quad (2.20)$$

D'où

$$D(z_1, z_2) = \det \begin{bmatrix} \mathbf{I}_N - z_1^{-1} \mathbf{A}_1 & -z_1^{-1} \mathbf{A}_2 \\ -z_2^{-1} \mathbf{A}_3 & \mathbf{I}_M - z_2^{-1} \mathbf{A}_4 \end{bmatrix}$$

Comme dans les sections précédentes, on suppose que le numérateur et le dénominateur dans l'équation (3.16) sont indépendants (aucun zéro commun) et que la fonction de transfert n'a pas de singularités non essentielles du second type sur le bi-cercle unité.

$$\begin{aligned} D(z_1, z_2) &= \det(\mathbf{I}_N - z_1^{-1} \mathbf{A}_1) \times \det\{\mathbf{I}_M - z_2^{-1} [\mathbf{A}_4 + \mathbf{A}_3 (z_1 \mathbf{I}_N - \mathbf{A}_1)^{-1} \mathbf{A}_2]\} \\ &= \det(\mathbf{I}_M - z_2^{-1} \mathbf{A}_4) \times \det\{\mathbf{I}_N - z_1^{-1} [\mathbf{A}_1 + \mathbf{A}_2 (z_2 \mathbf{I}_M - \mathbf{A}_4)^{-1} \mathbf{A}_3]\} \end{aligned} \quad (2.21)$$

Sur la base des équations (2.20) et (2.21), les conditions de la stabilité peuvent être facilement obtenues. Ces conditions sont énoncées dans le théorème suivant [3].

**Théorème 2** : filtre numérique représenté par la fonction de transfert de l'équation (2.16) un BIBO stable si et seulement si l'une des conditions suivantes est réalisée

- 1- (i) Toutes les valeurs propres de la matrice  $A_1$  sont situées sur le disque unitaire ouverte  
(ii) Toutes les valeurs propres de la matrice  $A_4 + A_3(Z_1 I_N - A_1)^{-1} A_2$  sont situées sur le disque unitaire ouvert tel que  $Z_1 \in U_1 = \{Z_1: |Z_1| = 1\}$
- 2- (i) Toutes les valeurs propres de la matrice  $A_4$  sont situées sur le disque unitaire ouvert  
(ii) Toutes les valeurs propres de la matrice  $A_1 + A_2(Z_2 I_M - A_4)^{-1} A_3$  sont situées sur le disque unitaire ouvert tel que  $Z_2 \in U_2 = \{Z_2: |Z_2| = 1\}$
- 3- (i) Toutes les valeurs propres de la matrice  $A$  sont situées sur le disque unitaire ouverte  
(ii) la matrice  $A_1$  n'a pas de valeurs propres situées sur le disque unitaire  
(iii) la matrice  $A_4 + A_3(Z_1 I_N - A_1)^{-1} A_2$  n'a pas de valeurs propres situées sur le disque unitaire  
pour  $Z_1 \in U_1$
- 4- (i) Toutes les valeurs propres de la matrice  $A$  sont situées sur le disque unitaire ouverte  
(ii) la matrice  $A_4$  n'a pas des valeurs propres dans le disque unitaire  
(iii) la matrice  $A_1 + A_2(Z_2 I_M - A_4)^{-1} A_3$  n'a pas de valeurs propres situées sur le disque unitaire ouverte.

#### La stabilité du filtre numérique 2D de faible ordre :

Considérons le filtre numérique 2D général d'ordre (2, 2) qui peut être représenté par la fonction de transfert

$$H(z_1, z_2) = \frac{N(z_1, z_2)}{D(z_1, z_2)} = \frac{N(z_1, z_2)}{\sum_{i=0}^{N_1} \sum_{j=0}^{N_2} b_{ij} z_1^{-i} z_2^{-j}} \quad b_{00} = 1$$

(2.22)

On suppose que le numérateur et le dénominateur sont Independent et il n'y a aucun terme commun,

La stabilité il y a plusieurs cas peuvent être identifiés selon l'ordre du polynôme dénominateur.

**Cas 1 :**

$$D(z_1, z_2) = 1 + b_{10}Z_1^{-1} \quad (2.23)$$

Alors le filtre est d'ordre (1, 0). De toute évidence, on a un filtre l-D qui est stable si et seulement si

$$|b_{10}| < 1$$

**Cas 2 :**

$$D(z_1, z_2) = 1 + b_{01}Z_2^{-1} \quad (2.24)$$

Le filtre est d'ordre (0, 1) et on obtient un filtre l-D qui est stable si et seulement si

$$|b_{01}| < 1$$

**Cas 3 :**

$$D(z_1, z_2) = 1 + b_{10}Z_1^{-1} + b_{01}Z_2^{-1} \quad (2.25)$$

Le filtre est d'ordre (1,1) et il est stable (BIBO) si :

$$|b_{01}| + |b_{10}| < 1$$

Case 4:

$$D(Z_1^{-1}, Z_2^{-1}) = 1 + b_{10}Z_1^{-1} + b_{01}Z_2^{-1} + b_{01}Z_1^{-1}Z_2^{-1} \quad (2.26)$$

Le filtre est d'ordre (1,1) et il est stable (BIBO) si :

$$|b_{10} + b_{01}| - 1 < b_{11} < 1 - |b_{10} - b_{01}|$$

#### Cas 5 :

La stabilité d'un filtre d'ordre (2, 2) devient très compliquée. Mai, un algorithme O'Connor et Huang [7] est disponible qui peut être utilisé pour la vérifier

#### 2.5. Conclusion

Dans ce chapitre, on fait l'étude du problème de synthèse d'un filtre RII par l'optimisation et l'utilisation des trois méthodes qui sont : moindre puissance, Minimax méthode et Quasi-Newton, ces méthodes sont connues de leur efficacité en ce qui concerne la résolution des problèmes d'optimisation. Les algorithmes précédents donnent d'excellents résultats sauf qu'il nécessite une quantité considérable de calculs.

# Chapitre 3

<b>3. Algorithmes d'optimisation</b> .....	37
<b>3.1. Introduction</b> .....	37
<b>3.2. Algorithmes Heuristiques</b> .....	37
<b>3.3. Algorithmes Métaheuristiques</b> .....	37
<b>3.4. Différentes Algorithme d'optimisation</b> .....	39
<b>3.4.1. Algorithme du recuit simulé</b> .....	39
<b>3.4.2. Algorithme de recherche tabou</b> .....	40
<b>3.4.3. Les algorithmes de colonies de fourmis</b> .....	40
<b>3.4.4. Algorithme à évolution différentielle</b> .....	41
<b>3.4.5. Les algorithmes évolutionnaires</b> .....	41
<b>3.5. Optimisation par essaim de particules</b> .....	43
<b>3.5.1. Principe général</b> .....	43
<b>3.5.2. Formalisation</b> .....	44
<b>3.5.3 Améliorations de PSO</b> .....	45
<b>3.5.3.1. Coefficient de constriction</b> .....	46
<b>3.5.3.2. Topologie de voisinage</b> .....	47
<b>3.5.3.3. Coefficient d'inertie</b> .....	48
<b>3.5.4. L'algorithme Multi-PSO</b> .....	48
<b>3.5.5. Algorithme proposé</b> .....	49
<b>3.5.5.1. Partitionnement de l'espace de recherche</b> .....	50
<b>3.6. Conclusion</b> .....	53

### 3. Algorithmes d'optimisation

#### 3.1. Introduction

Un algorithme d'optimisation est caractérisé par un ensemble de variables, une fonction objective et un ensemble de contraintes. L'espace de recherche est l'ensemble des solutions possibles du problème. Résoudre un problème à travers un algorithme d'optimisation consiste à trouver les meilleures solutions, tout en satisfaisant un ensemble de contraintes bien définies et par l'utilisation d'une fonction objective. Certains problèmes d'optimisation sont qualifiés de difficiles, et leur résolution dans un temps raisonnable nécessite l'utilisation d'algorithmes sophistiqués, comme les méthodes approchées (Les heuristiques et les métaheuristiques). Parmi les métaheuristiques destinées à résoudre ces problèmes, plus précisément les problèmes à variables continues, l'optimisation par essaim particulaire (OEP, ou PSO en anglais) apparue en 1995.

#### 3.2. Algorithmes Heuristiques

La résolution des problèmes par des méthodes exactes n'est pas toujours possible. Certains nombres de contraintes (comme le temps de calcul souvent important) posent des difficultés. Pour faire face à ces contraintes, nous utilisons des méthodes approchées, appelées heuristiques, ce terme dérive du grec ancien heuriskêin et signifie « trouver ». Il sert à la découverte et à l'exploitation. Une méthode heuristique peut être déterministe ou stochastique. Ce type d'algorithmes fournit des solutions rapides, approchées et réalisables, pour des problèmes d'optimisation difficiles. Les algorithmes heuristiques sont des méthodes approximatives ou contrairement des algorithmes exacts, ils donnent des solutions optimales pour des problèmes donnés. Il y a une multitude d'heuristiques qui ont déjà été proposées dans la littérature. Nous pouvons citer des heuristiques très simples telles que les algorithmes gloutons [27,28] ou les approches par amélioration itérative.

#### 3.3. Algorithmes Métaheuristiques

Le terme métaheuristiques a été inventé par Fred Glover en 1986, lors de la conception de la recherche Tabou [29]. Les métaheuristiques forment une famille d'algorithmes

d'optimisation visant à résoudre des problèmes d'optimisation difficiles, pour lesquels nous ne connaissons pas de méthodes classiques plus efficaces. Elles sont généralement utilisées comme des méthodes génériques pouvant optimiser une large gamme de problèmes différents. Leur capacité, à optimiser un problème à partir d'un nombre minimal d'informations est contrebalancée par le fait qu'elles n'offrent aucune garantie quant à l'optimalité de la meilleure solution trouvée. Cependant, du point de vue de la recherche opérationnelle, ce n'est pas forcément un désavantage, puisque l'on préfère toujours une approximation de l'optimum global trouvée rapidement à une valeur exacte trouvée dans un temps rédhibitoire.

Il existe un grand nombre de métaheuristiques différentes, allant de la simple recherche locale à des algorithmes complexes de recherche globale. La plupart des métaheuristiques utilisent des processus aléatoires comme moyens de récolter de l'information et de faire face à des problèmes comme l'explosion combinatoire. Les métaheuristiques peuvent être considérées comme des algorithmes stochastiques itératifs, où elles manipulent une ou plusieurs solutions à la recherche de l'optimum. Les itérations successives doivent permettre de passer d'une solution de mauvaise qualité à la solution optimale. L'algorithme s'arrête après avoir atteint un critère d'arrêt, consistant généralement en l'atteinte du temps d'exécution imparti ou en une précision demandée. Ces méthodes tirent leur intérêt de leur capacité à éviter les optima locaux, soit en acceptant des dégradations de la fonction objective au cours du traitement, soit en utilisant une population de points comme méthode de recherche. Les métaheuristiques sont souvent inspirées de processus naturels qui relèvent de la physique (l'algorithme du recuit simulé), de la biologie de l'évolution (les algorithmes génétiques) ou encore de l'éthologie (les algorithmes de colonies de fourmis ou l'optimisation par essaim particulaire). Les métaheuristiques se caractérisant par leur capacité à résoudre des problèmes très divers, elles se prêtent naturellement à des extensions. Pour illustrer celles-ci, nous pouvons citer :

– Les métaheuristiques pour l'*optimisation multiobjectif* [30] : où il faut optimiser plusieurs objectifs contradictoires. Le but ne consiste pas ici à trouver un optimum global, mais à trouver un ensemble d'optima, qui forment une surface de compromis pour les différents objectifs du problème :

- Les métaheuristiques pour l'*optimisation multimodale* [31] : où l'on ne cherche plus l'optimum global, mais l'ensemble des meilleurs optima globaux et/ou locaux.
- Les métaheuristiques pour l'*optimisation de problèmes bruités* : où il existe une incertitude sur le calcul de la fonction objective, dont il faut tenir compte dans la recherche de l'optimum.
- Les métaheuristiques pour l'*optimisation dynamique* [32] : où la fonction objective varie dans le temps, ce qui nécessite d'approcher l'optimum à chaque pas de temps.
- Les métaheuristiques *hybrides* [33] : qui consistent à combiner différentes métaheuristiques, afin de tirer profit des avantages respectifs.
- Les métaheuristiques *parallèles* [34] : où l'on cherche à accélérer le calcul, en répartissant la charge de calcul sur des unités fonctionnant de concert. Le problème revient alors à adapter les métaheuristiques pour qu'elles soient distribuées.

### 3.4. Différentes Algorithmes d'optimisation

#### 3.4.1. Algorithme du recuit simulé

Le recuit simulé est une méthode inspirée d'un processus utilisé en métallurgie (appelé le *recuit*) pour atteindre les états de basse énergie d'un solide, on chauffe celui-ci jusqu'à des températures élevées, avant de le laisser refroidir lentement. L'algorithme du recuit simulé a été proposé par Kirkpatrick et al. [35] (et indépendamment, Cerny [36]). La description classique du recuit simulé le présente comme un algorithme probabiliste, où un point évolue dans l'espace de recherche. Le recuit simulé s'appuie sur l'algorithme de Metropolis [37,38], qui permet de décrire l'évolution d'un système en thermodynamique. Cette procédure permet de sortir des minima locaux avec une probabilité élevée si la température  $T$  est élevée et, quand l'algorithme atteint de basses températures, il conserve les états les plus probables. Ici, la méthode de Metropolis (ou toute autre méthode d'échantillonnage [39,40]) tient lieu de diversification, associée à la décroissance de température, qui contrôle l'intensification.

### 3.4.2. Algorithme de recherche tabou

La recherche Tabou est une métaheuristique itérative qualifiée de « recherche locale » au sens large. L'idée de la recherche Tabou consiste, à partir d'une position donnée, à en explorer le voisinage et à choisir le voisin qui minimise la fonction objective.

L'algorithme de recherche Tabou a été introduit par Fred Glover en 1986 [29]. Le principe de base de cet algorithme est de pouvoir poursuivre la recherche de solutions même lorsqu'un optimum local est rencontré et ce, en permettant des déplacements qui n'améliorent pas la solution, et en utilisant le principe de l'enregistrement pour éviter les retours en arrière (mouvements cycliques). Il est essentiel de noter que cette opération peut conduire à augmenter la valeur de la fonction (dans un problème de minimisation), c'est le cas lorsque tous les points du voisinage ont une valeur plus élevée. C'est à partir de ce mécanisme que l'on sort d'un minimum local. En effet, comme l'algorithme de recuit simulé, la méthode de recherche Tabou fonctionne avec une seule configuration courante, qui est actualisée au cours des itérations successives. La nouveauté ici est que, pour éviter le risque de retour à une configuration déjà visitée, on tient à jour une liste de mouvements interdits (ou de solutions interdites), appelée « liste Tabou ». Le rôle de cette dernière évolue au cours de la résolution pour passer de l'exploration (aussi appelée « diversification ») à l'exploitation (également appelée « intensification »). Cette liste contient  $m$  mouvements ( $t \rightarrow s$ ) qui sont les inverses des  $m$  derniers mouvements ( $s \rightarrow t$ ) effectués. L'algorithme modélise ainsi une forme primaire d'enregistrement à court terme.

### 3.4.3. Les algorithmes de colonies de fourmis

Les algorithmes de colonies de fourmis sont des algorithmes inspirés du comportement des fourmis et constituent une famille de métaheuristicques d'optimisation pour résoudre naturellement des problèmes complexes. Une telle aptitude s'avère possible en raison de la capacité des fourmis à communiquer entre elles indirectement, par le dépôt dans l'environnement de substances chimiques, appelées *phéromones*. Ce type de communication indirecte est appelé *stigmergie*. En anglais, le terme consacré à la principale classe d'algorithmes est *Ant Colony Optimization* (ACO). La principale illustration de ce phénomène

comme ceci : si un obstacle est placé sur le trajet des fourmis qui, après une étape d'exploration, finiront par emprunter le plus court chemin entre le nid et la source de nourriture [41]. Les fourmis qui sont retournées le plus rapidement au nid en passant par la source de nourriture sont celles qui ont emprunté le chemin le plus court. Il en découle que la quantité de phéromones déposées par unité de temps sur ce trajet est plus importante que sur les autres. Par ailleurs, une fourmi est d'autant plus attirée à un certain endroit que le taux de phéromones y est important. De ce fait, le plus court chemin a une probabilité plus importante d'être emprunté par les fourmis que les autres chemins et sera donc, à terme, emprunté par toutes les fourmis.

#### 3.4.4. Algorithme à évolution différentielle

L'évolution différentielle (Differential Evolution DE) est une métaheuristique stochastique d'optimisation qui a été inspirée par les algorithmes génétiques et des stratégies évolutionnaires combinées avec une technique géométrique de recherche. Les algorithmes génétiques changent la structure des individus en utilisant la mutation et le croisement, alors que les stratégies évolutionnaires réalisent l'auto-adaptation par une manipulation géométrique des individus. Ces idées ont été mises en œuvre grâce à une opération simple, mais puissante, de mutation de vecteurs, proposée en 1995 par K. Price et R. Storn [42]. Même si, à l'origine, la méthode de l'évolution différentielle était conçue pour les problèmes d'optimisation continus et sans contraintes, ses extensions actuelles peuvent permettre de traiter les problèmes à variables mixtes et gèrent les contraintes non linéaires.

Dans la méthode DE, la population initiale est générée par tirage aléatoire uniforme sur l'ensemble des valeurs possibles de chaque variable. Les bornes inférieures et supérieures des variables sont spécifiées par l'utilisateur selon la nature du problème. Après l'initialisation, l'algorithme effectue une série de transformations sur les individus, dans un processus appelé évolution.

#### 3.4.5. Les algorithmes évolutionnaires

Les algorithmes évolutionnaires (AEs), élaborés au cours des années 1950 [43], sont des techniques de recherche inspirées par l'évolution biologique des espèces. Ils s'inspirent de l'évolution des êtres vivants pour résoudre des problèmes d'optimisation. L'idée ici est que, les

individus qui ont hérité des caractères bien adaptés à leur milieu ont tendance à vivre assez longtemps pour se reproduire, alors que les plus faibles ont tendance à disparaître. Au cours des années 1970, avec l'avènement des calculateurs de forte puissance, de nombreuses approches de modélisation de l'évolution ont été réalisées. Nous pouvons citer :

- Les stratégies d'évolution : qui ont été conçues pour résoudre des problèmes à variables continues. Elles sont axées sur la modélisation des paramètres stratégiques qui contrôlent la variation dans l'évolution, autrement dit « l'évolution de l'évolution » :
- La programmation évolutionnaire : qui vise à faire évoluer les structures d'automates à états finis par des successions de croisements et de mutations.
- Les algorithmes génétiques : qui ont été conçus pour résoudre des problèmes d'optimisation à variables discrètes, en modélisant l'évolution génétique.
- La programmation génétique : basée sur les algorithmes génétiques, mais où les individus (ou chromosomes) sont des programmes informatiques, représentés en utilisant une structure d'arbre.
- L'évolution différentielle : qui a été conçue pour résoudre des problèmes à variables continues. Sa stratégie consiste à biaiser un opérateur de mutation, appliqué à un individu, en fonction des différences calculées avec d'autres individus sélectionnés aléatoirement. Les approches évolutionnaires s'appuient sur un modèle commun. Les individus soumis à l'évolution sont des solutions possibles du problème posé. L'ensemble de ces individus constitue une population. Cette population évolue durant une succession d'itérations, appelées générations. Au cours de chaque génération, une série d'opérateurs est appliquée aux individus, pour créer la population de la génération suivante. Chaque opérateur utilise un ou plusieurs individus, appelés parents, pour engendrer de nouveaux individus, appelés enfants. A la fin de chaque génération, une sélection d'enfants créés durant la génération remplace un sous-ensemble d'individus de la population.

### 3.5. Optimisation par essaim de particules

#### 3.5.1. Principe général

L'optimisation par essaim de particules (OEP), ou (Particle Swarm Optimization) (PSO) en anglais, est un algorithme évolutionnaire qui utilise une population de solutions candidates pour développer une solution optimale au problème. Cet algorithme a été proposé par Russel Eberhart (ingénieur en électricité) et James Kennedy (socio-psychologue) en 1995 [13]. Il s'inspire du comportement social des animaux évoluant en essaim, tels que les poissons et les oiseaux. En effet, on peut observer chez ces animaux des dynamiques de déplacement relativement complexes, alors qu'individuellement, chaque individu a une « intelligence » limitée, et ne dispose que d'une connaissance locale de sa situation dans l'essaim. L'information locale de chaque individu est utilisée pour décider de son déplacement. Des règles simples, telles que « rester proche des autres individus », « aller dans une même direction » ou « aller à la même vitesse », suffisent pour maintenir la cohésion de l'essaim, et permettent la mise en œuvre de comportements collectifs complexes et adaptatifs. L'essaim de particules correspond à une population d'agents simples, appelés particules. Chaque particule est considérée comme une solution du problème, où elle possède une position (le vecteur solution) et une vitesse. De plus, chaque particule possède un enregistrement lui permettant de se souvenir de sa meilleure performance (en position et en valeur) et de la meilleure performance atteinte par les particules « voisines » (informatrices) : chaque particule dispose en effet d'un groupe d'informatrices, historiquement appelé son voisinage. Un essaim de particules, qui sont des solutions potentielles au problème d'optimisation, « survole » l'espace de recherche, à la recherche de l'optimum global. Le déplacement d'une particule est influencé par les trois composantes suivantes :

1. Une composante d'inertie : la particule tend à suivre sa direction courante de déplacement.
2. Une composante cognitive : la particule tend à se diriger vers le meilleur site par lequel elle est déjà passée.
3. Une composante sociale : la particule se diriger vers le meilleur site déjà atteint par ses voisins. La stratégie de déplacement d'une particule est illustrée dans la figure 3.1.

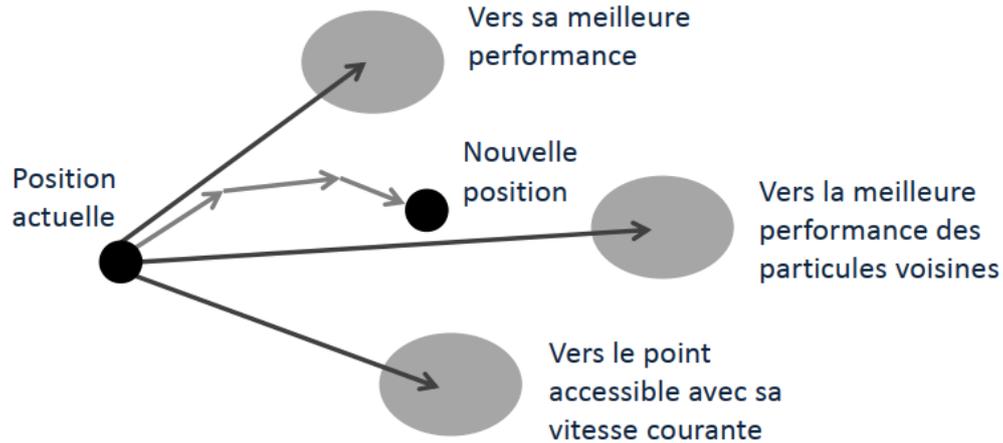


Figure :3.1 Déplacement d'une particule.

### 3.5.2. Formalisation

Dans un espace de recherche de dimension  $D$ , la particule  $i$  de l'essaim est modélisée par son vecteur de position  $\vec{x}_i(x_{i1}, x_{i2}, x_{i3}, x_{i4}, \dots, x_{iD})$  et son vecteur de vitesse  $\vec{v}_i(v_{i1}, v_{i2}, v_{i3}, v_{i4}, \dots, v_{iD})$ . La qualité de sa position est déterminée par la valeur de la fonction objective à ce stade. La particule se souvient de la meilleure position locale, par laquelle elle est déjà passée, est notée  $\vec{P}best(pbest_{i1}, pbest_{i2}, \dots, pbest_{iD})$ .

La meilleure position globale, atteinte par les particules de l'essaim, est notée :  $\vec{G}best(gbest_{i1}, gbest_{i2}, \dots, gbest_{iD})$ .

Nous nous référons à la version Standard de PSO, où toutes les particules de l'essaim sont considérées voisines de la particule  $i$ .

Au départ de l'algorithme, les particules de l'essaim sont initialisées de manière aléatoire dans l'espace de recherche du problème. Ensuite, à chaque itération, chaque particule se déplace, en combinant linéairement les trois composantes citées ci-dessus. En effet, à l'itération  $t + 1$ , le vecteur vitesse et le vecteur position sont calculés à partir de l'équation (3.1) et de l'équation (3.2), respectivement.

$$v_{i,j}^{t+1} = w * v_{i,j}^t + C_1 * r_1(pbest_{i,j}^t - x_{i,j}^t) + C_2 * r_2(gbest_{i,j}^t - x_{i,j}^t)$$

(3.1)

$$x_{i,j}^{t+1} = x_{i,j}^t + v_{i,j}^{t+1} \quad j \in \{1,2,3, \dots, D\} \quad (3.2)$$

Où  $w$  est une constante, appelée coefficient d'inertie ;  $C_1$  et  $C_2$  sont deux constantes, appelées coefficients d'accélération ;  $r_1$  et  $r_2$  sont deux nombres aléatoires tirés uniformément dans  $[0, 1]$ , à chaque itération  $t$  et pour chaque dimension  $j$ . Les trois composantes mentionnées ci-dessus (d'inertie, cognitive et sociale) sont représentées dans l'équation (3.1) par les termes suivants :

1.  $w * v_{i,j}^t$  correspond à la composante d'inertie du déplacement, où le paramètre  $w$  contrôle l'influence de la direction de déplacement sur le déplacement futur.
2.  $C_1 * r_1 (pbest_{i,j}^t - x_{i,j}^t)$  correspond à la composante cognitive du déplacement, où le paramètre  $c_1$  contrôle le comportement cognitif de la particule.
3.  $C_2 * r_2 (gbest_{i,j}^t - x_{i,j}^t)$  correspond à la composante sociale du déplacement, où le paramètre  $C_2$  contrôle l'aptitude sociale de la particule.

Une fois le déplacement des particules effectué, les nouvelles positions sont évaluées et les deux vecteurs  $pbest_{i,j}^t$  et  $gbest_{i,j}^t$  sont mis à jour, à l'itération  $t + 1$ .

### 3.5.3 Améliorations de PSO

Pour éviter que le système ne s'explode en cas d'amplification trop grande d'oscillations (il est possible que le déplacement d'une particule soit trop rapide et la conduise à sortir de l'espace de recherche), nous pouvons introduire un nouveau paramètre  $Vmax$ , qui permet de limiter la vitesse sur chaque dimension et ainsi de contrôler l'explosion du système [44].

Notons que cela ne restreint pas les valeurs de  $x_i$  à l'intervalle  $[Vimin, Vimax]$ , mais limite seulement la distance maximale qu'une particule va parcourir au cours d'une itération. Cette méthode permet de contrôler la divergence de l'algorithme et de réaliser ainsi un compromis efficace entre intensification et diversification.

De plus, une stratégie de confinement des particules peut être introduite. Une telle stratégie permet de ramener une particule sortie de l'espace de recherche à l'intérieur de celui-ci. Dans ce cadre, plusieurs méthodes peuvent être employées :

- La particule est laissée à l'extérieur de l'espace de recherche, mais on n'évalue pas sa fonction objective. Ainsi, elle ne pourra pas attirer les autres particules en dehors de l'espace de recherche.
- La particule est arrêtée à la frontière et les composantes associées à sa vitesse sont annulées.
- La particule rebondit sur la frontière. La particule est stoppée sur la frontière, mais les composantes correspondantes de la vitesse sont multipliées par un coefficient tiré aléatoirement dans l'intervalle [1,0].

### 3.5.3.1. Coefficient de constriction

Certaines améliorations ont été apportées à l'algorithme de base, notamment en termes de contrôle de divergence. En particulier, l'inclusion du paramètre  $V_{max}$  qui permet de limiter la divergence des particules. Par ailleurs, d'autres études ont été menées sur la dynamique des particules, tentées d'analyser l'algorithme lorsque les conditions de convergence de l'essaim sont assurées [45].

La combinaison des paramètres  $w$ ,  $c_1$  et  $c_2$  ajuste l'équilibre entre les phases de diversification et l'intensification du processus de recherche. Clerc et Kennedy [46] ont montré qu'une bonne convergence peut être obtenue en rendant ces paramètres dépendants. L'utilisation d'un coefficient de constriction  $\chi$  (ou facteur de constriction) permet de mieux contrôler la divergence de l'essaim et de s'affranchir de la définition de  $V_{max}$ . Cette variante de PSO, largement utilisée dans la littérature, est connue sous le nom de PSO canonique. En utilisant le coefficient de constriction, Eq. (3.1) devient :

$$v_{i,j}(t+1) = \chi * \left( w * v_{i,j}(t) + \phi_1 * r_1 * (pbest_{i,j} - x_{i,j}(t)) + \phi_2 * r_2 * (gbest_{i,j} - x_{i,j}(t)) \right) \quad (3.3)$$

Avec  $\chi = \frac{2}{\phi - 2 + \sqrt{\phi^2 - 4\phi}}$  et  $\phi = \phi_1 + \phi_2$ ,  $\phi > 4$

Les valeurs optimales de  $\phi_1$  et  $\phi_2$  ont été déterminées dans [17], en effectuant de nombreux tests. En général, on utilise  $\phi = 4,1$  et  $\phi_1 = \phi_2$ , ce qui donne un coefficient  $\chi = 0,7298844$ . Dans

[47], les auteurs ont indiqué que l'utilisation d'un coefficient de constriction donne généralement un meilleur taux de convergence, sans avoir à fixer de vitesse maximale  $V_{max}$ . Cependant, dans certains cas, le coefficient de constriction seul ne permet pas la convergence vers la solution optimale pour un nombre d'itérations donné. Pour remédier à ce problème, il pourrait être intéressant de fixer  $V_{max} = (x_{max} - x_{min})/2$  en plus du coefficient de constriction, ce qui, selon les études de Shi et Eberhart, permet d'améliorer les performances globales de l'algorithme. Ainsi, il est à noter que PSO, utilisant un coefficient de constriction, n'est pas la seule version de PSO qui garantisse la convergence vers un état d'équilibre.

### 3.5.3.2. Topologie de voisinage

Comme nous l'avons vu précédemment le PSO est une méthode d'optimisation stochastique inspirée d'un comportement social. Ce comportement a été modélisé par les deux équations (3.1) et (3.2) pour guider les particules durant le processus de déplacement. Le choix d'une topologie (le réseau de communication entre les particules) a donc une influence importante sur les performances de PSO. A l'origine, les auteurs ont défini une topologie entièrement connectée (chaque particule est reliée à toutes les autres). Cette version de PSO est appelée version globale, car la particule est informée par la totalité des autres, et l'information effectivement utilisée est incarnée par le terme  $G_{best}$  de la troisième composante de l'équation (3.1). Cette version a l'inconvénient majeur de ne pas donner lieu à une exploration suffisante, ce qui peut conduire à une stagnation dans un optimum local et donc à une convergence prématurée. De nombreuses variantes de la version originale, dites versions locales ( $L_{best}$ ), ont été proposées dans la littérature de PSO, afin d'améliorer sa convergence. Parmi ces variantes, nous pouvons citer celle proposée dans [Eberhart et al., 1995] et qui utilise un graphe d'information statique sous forme d'anneau (cette version est connue comme étant la version *locale classique*). Dans les versions locales, le terme  $G_{best}$  est remplacé par les termes  $L_{best}$ , où, pour chaque particule  $i$ , on définit un ensemble de voisinage (i.e. l'information qui doit être partagée est la meilleure solution trouvée dans le voisinage de chaque particule).

### 3.5.3.3. Coefficient d'inertie

Le coefficient d'inertie  $w$ , introduit par Shi et Eberhart [47], contrôle l'influence de la direction de la particule sur le déplacement futur. Le but de l'introduction de ce paramètre est de réaliser un équilibre entre la recherche locale (exploitation) et la recherche globale (exploration). L'intensité de l'exploration de l'espace de recherche dépend de la valeur du poids d'inertie, une grande valeur de  $w$  facilitant une exploration globale, alors qu'une petite valeur facilite l'exploration locale. Du fait de son influence sur les performances de l'algorithme PSO, le poids d'inertie a suscité un grand intérêt de la part de la communauté des chercheurs. Dans [48], les auteurs ont proposé un coefficient d'inertie dynamique qui varie au cours du temps. Il commence par une valeur proche de 0,9 et descend linéairement pour arriver à 0,4. Cette stratégie a beaucoup amélioré les performances de PSO pour plusieurs problèmes d'optimisation. Le coefficient d'inertie  $w$  varie linéairement avec le temps selon la formule suivante :

$$w = w_{min} + (w_{max} - w_{min}) \cdot \left( \frac{iter}{max_{iter}} \right) \quad (3.4)$$

où  $iter$  est l'itération courante et  $max_{iter}$  est le nombre maximal d'itérations.  $w_{max}$  et  $w_{min}$  désignent respectivement les valeurs maximum et minimum du coefficient  $w$  (généralement,  $w_{max}, w_{min} \in [0, 1]$ ).

### 3.5.4. L'algorithme Multi-PSO

Multi-PSO est une nouvelle variante de l'algorithme original d'optimisation par essaim des particules, dans lequel deux idées ont été ajoutées ; La première idée est d'utiliser deux types d'essaims, un principal et des auxiliaires. La deuxième idée est la partition de l'espace de recherche en plusieurs zones. Dans un autre sens, l'algorithme est basé sur le PSO standard avec une nouvelle stratégie d'initialisation. Cette stratégie est basée sur l'utilisation de deux types d'essaims : l'essaim principal désigné par S1, et les essaims auxiliaires notés S2i, où  $1 \leq i \leq s$  [18]. Dans l'algorithme Multi-PSO, chaque essaim auxiliaire est destiné à découvrir une zone

de l'espace de recherche en effectuant un certain nombre d'itérations, son rôle se termine lorsque la meilleure position de la zone est atteinte. Plus précisément, les populations des essaims auxiliaires sont initialisées aléatoirement dans les différentes zones, selon une partition déjà réalisée dans l'espace de recherche. Ensuite, les particules de chaque essaim effectuent  $n$  générations prédéfinies dans leur zone et, à la fin, la position la mieux trouvée est enregistrée dans un tableau vectoriel. Les meilleures positions obtenues par tous les essaims auxiliaires sont donc regroupées dans un vecteur de taille égale au nombre d'essaims initialisés et également au nombre de zones. Les toutes meilleures positions composeront alors la population initiale de l'essaim principal. Une fois l'essaim principal initialisé, les essaims auxiliaires ne sont plus utilisés. L'essaim principal continue la recherche de la même manière que l'algorithme PSO standard jusqu'à ce qu'un critère d'arrêt soit atteint.

### 3.5.5. Algorithme proposé

Dans Multi-PSO, les particules de l'essaim auxiliaire sont initialisées aléatoirement chacun dans la zone qui lui correspond. Après l'initialisation, l'essaim exécute un nombre  $n$  itérations de la même façon que PSO standard, et il garde à la fin de ces itérations la meilleure position trouvée pendant la recherche. Le but, dans la génération d'un petit nombre d'itérations est d'explorer uniquement la région où l'essaim a été initialisé, et donc le plus grand nombre d'itérations est laissé pour l'essaim principal. Du fait de la convergence rapide de l'algorithme PSO et de l'utilisation de l'approche multi-essaims dans un espace partitionné, les essaims auxiliaires dans Multi-PSO convergent chacun vers une position intéressante dans les différentes zones de l'espace, et donc le cas où plusieurs essaims convergent vers la même position est évité. Ces positions sont des optima locaux des problèmes obtenus des différentes zones. Donc, à chaque fois que l'essaim auxiliaire termine son travail, la meilleure solution trouvée par cet essaim est ajoutée, comme une particule initiale, à l'essaim principal  $S1$ . La taille de population de l'essaim principal  $S1$  est donc égale au nombre des essaims auxiliaires  $S2i$ . La phase de construction de l'essaim  $S1$  se termine après avoir généré tous les essaims  $S2i$ . Après cette phase,  $S1$  prend le relais et continue seul à faire une recherche globale dans l'espace. A la différence de PSO standard, dans deuxième phase le Multi-PSO (phase de l'essaim

principal) commence la recherche avec des particules bien positionnées dans l'espace de recherche.

### 3.5.5.1. Partitionnement de l'espace de recherche

Dans les travaux quotidiens, le partage des tâches donne toujours des résultats intéressants. Par exemple, la recherche des trésors perdus ou le forage des puits de pétrole, où l'on commence la recherche en l'effectuant sur plusieurs régions prédéfinies, pour ensuite intensifier la recherche dans les zones prometteuses. Nous nous inspirons de ce modèle de recherche pour trouver une solution optimale du problème d'optimisation et, à cet effet, nous proposons de partitionner l'espace de recherche en plusieurs zones.

La recherche de Multi-PSO est faite alors dans un espace de recherche partitionné et donc nous partitionnons l'espace de recherche  $E$  en plusieurs zones séparées, dans lesquelles les essais auxiliaires sont initialisés. Ce partitionnement contribue à augmenter la diversification des particules dans tout l'espace de recherche, et d'assurer une bonne couverture pendant la phase d'initialisation. En pratique, nous fixons un nombre maximal de zones, dénommé *maxzone*, qui est égal au nombre des essais auxiliaires  $S$ . Les zones qui forment l'espace de recherche  $E$  sont définies par :

$$zone_i, 1 \leq i \leq S \text{ tels que } \bigcup_{i=1}^S zone_i = E ; (\forall i, j \in \{1, \dots, S\}, zone_i \cap zone_j = \emptyset) \quad (3.5)$$

Afin de partitionner un espace de recherche  $E$  de dimension  $D$  ( $E = [min_d, max_d]^D$ ) où  $min_d$  et  $max_d$  représentent respectivement les bornes inférieure et supérieure dans chaque dimension  $d$ ,  $1 \leq d \leq D$ , nous définissons le centre de l'espace de recherche comme un point de départ pour le partitionnement. Les centres des intervalles, qui définissent les bornes dans chaque dimension  $d$ , sont définis par l'équation suivante :

$$centre_d = \frac{(max_d + min_d)}{2} \quad (3.6)$$

Une fois que les centres d'intervalles sont définis, nous commençons par partitionner les intervalles de chaque dimension, et donc l'espace de recherche, en se basant sur ces centres. A cet effet, nous devons calculer un pas  $pas_d$  pour chaque intervalle  $[min_d, max_d]$ . Les  $pas_d$  sont utilisés pour construire les nouvelles bornes  $[zone_{imin}, zone_{imax}]$  de chaque zone  $i$ ,  $1 \leq i \leq max_{zone}$ . Le  $pas_d$  de chaque dimension est calculé selon la formule suivante :

$$pas_d = \frac{max_d - min_d}{2 * max_{zone}} \quad (3.7)$$

Chaque zone  $[zone_{imin}, zone_{imax}]^D$  est déterminée en utilisant les deux équations suivantes :

$$zone_{imin} = centre_d - i * pas_d \quad (3.8)$$

$$zone_{imax} = centre_d + i * pas_d \quad (3.9)$$

où  $zone_{imin}$  et  $zone_{imax}$  sont, respectivement, les bornes inférieures et les bornes supérieures de chaque zone. La figure 3.2 et la figure 3.3 illustrent le partitionnement de l'espace de recherche dans les cas uniforme et non uniforme.

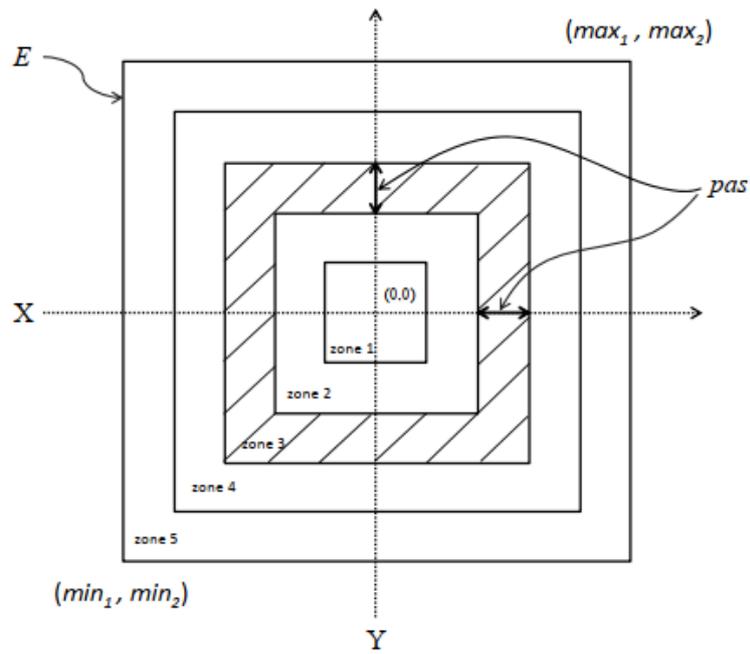


Figure 3.2: Partitionnement d'un espace de recherche uniforme.

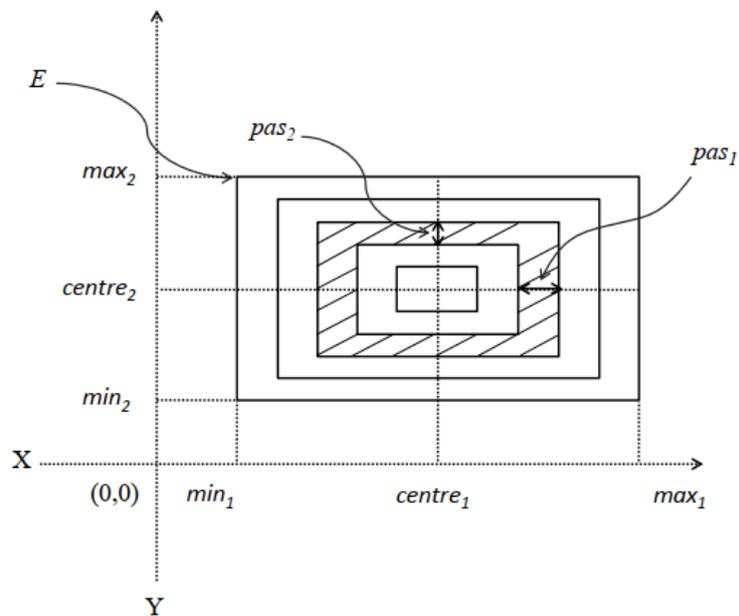


Figure 3.3: Partitionnement d'un espace de recherche non uniforme.

### 3.6. Conclusion

Parmi les métaheuristiques présentées pour résoudre des problèmes, un intérêt particulier a été porté à la méthode d'optimisation par essaim de particules (PSO). Cette méthode, qui est inspirée du comportement social des animaux évoluant en essaim, a rencontré un succès remarquable depuis sa création, grâce à sa simplicité. Elle présente l'avantage d'être efficace sur une vaste gamme de problèmes. Cependant, PSO présente un problème majeur, qui rebute encore certains utilisateurs : le problème de la convergence rapide, qui peut conduire les algorithmes de ce type à stagner dans un optimum local. De nombreux travaux ont été proposés pour améliorer les performances de PSO, tout en essayant de remédier à ce problème.

# Chapitre 4

<b>4. Résultats de simulation et discussion .....</b>	<b>55</b>
<b>4.1. Introduction.....</b>	<b>55</b>
<b>4.2. Algorithme.....</b>	<b>55</b>
<b>4.2.1. Les étapes d'algorithme proposé .....</b>	<b>56</b>
<b>4.2.2. Organigramme de l'algorithme proposé .....</b>	<b>57</b>
<b>4.3.1. Formulation du problème de synthèse de filtres récursifs bidimensionnels.....</b>	<b>58</b>
<b>4.3.2 Fonction de transfert.....</b>	<b>63</b>
<b>4.4. Résultats et discussion.....</b>	<b>64</b>
<b>4.5 Conclusion.....</b>	<b>70</b>

## 4. Résultats de simulation et discussion

### 4.1. Introduction

Pour résoudre le problème de conception de filtre RII, d'importantes méthodes basées sur l'optimisation ont vu le jour en particulier, celles des techniques évolutionnaires, où l'idée principale de ces méthodes d'optimisation est la conception des filtres numériques bidimensionnels RII considérés comme problème de minimisation. Parmi eux, l'optimisation par essaim de particules (PSO), cet algorithme apparaît comme un outil puissant dans les problèmes d'optimisation. Il utilise seulement deux équations mathématiques avec des opérateurs primitifs et conceptuellement est très simple. Inspiré du comportement social des animaux évoluant en essaim comme les poissons et les oiseaux, le PSO montre une facilité de mise en œuvre par rapport à l'algorithme génétique (GA), et il a moins de paramètres à ajuster. Il a déjà été appliqué avec succès dans de nombreux domaines d'application, y compris l'optimisation des fonctions.

### 4.2. Algorithme

Le Multi-PSO est une nouvelle variante de l'algorithme original d'optimisation par essaim de particules, dans lequel deux idées ont été ajoutées : la première idée est l'utilisation de deux types d'essaims, un principal et des auxiliaires. La deuxième idée est la partition de l'espace de recherche en plusieurs zones. L'algorithme utilise le PSO standard avec une nouvelle stratégie d'initialisation. Cette stratégie est basée sur deux types d'essaims : l'essaim principal désigné par  $S_1$ , et les essaims auxiliaires notés  $S_{2i}$ , où  $1 \leq i \leq s$ . Dans l'algorithme Multi-PSO, chaque essaim auxiliaire est destiné à découvrir une zone de l'espace de recherche en effectuant un certain nombre d'itérations, son rôle se termine lorsque la meilleure position de la zone est atteinte. Plus précisément, les populations des essaims auxiliaires sont initialisées aléatoirement dans les différentes zones, selon une partition déjà réalisée dans l'espace de recherche. Ensuite, les particules de chaque essaim effectuent  $n$  générations prédéfinies dans leur zone et, à la fin, la meilleure position trouvée est enregistrée dans un tableau. Ces positions obtenues par tous les essaims auxiliaires composeront alors la population initiale de l'essaim principal. Une fois

l'essaim principal initialisé, les essais auxiliaires ne sont plus utilisés. L'essaim principal continue la recherche de la même manière que l'algorithme PSO standard jusqu'à ce qu'un critère d'arrêt soit atteint.

#### 4.2.1. Les étapes d'algorithme proposé :

Etape 1 : Introduire le filtre-modèle.

Etape 2 : Démarrer le programme et charger les paramètres initiaux.

Etape 3 : Génération de la population initiale pour l'essaim(i).

Etape 4 : Exécution du PSO auxiliaire (i) avec K itérative période (update la vitesse  $\vec{v}_i$  et le vecteur de position  $\vec{x}_i$  ).

Etape 5 : Sauvegarder la meilleure position globale essaim(i) in gbest (i).

Etape 6 : Si l'essaim(i) n'est pas le dernier incrémenter la valeur de i et aller à l'étape 3.

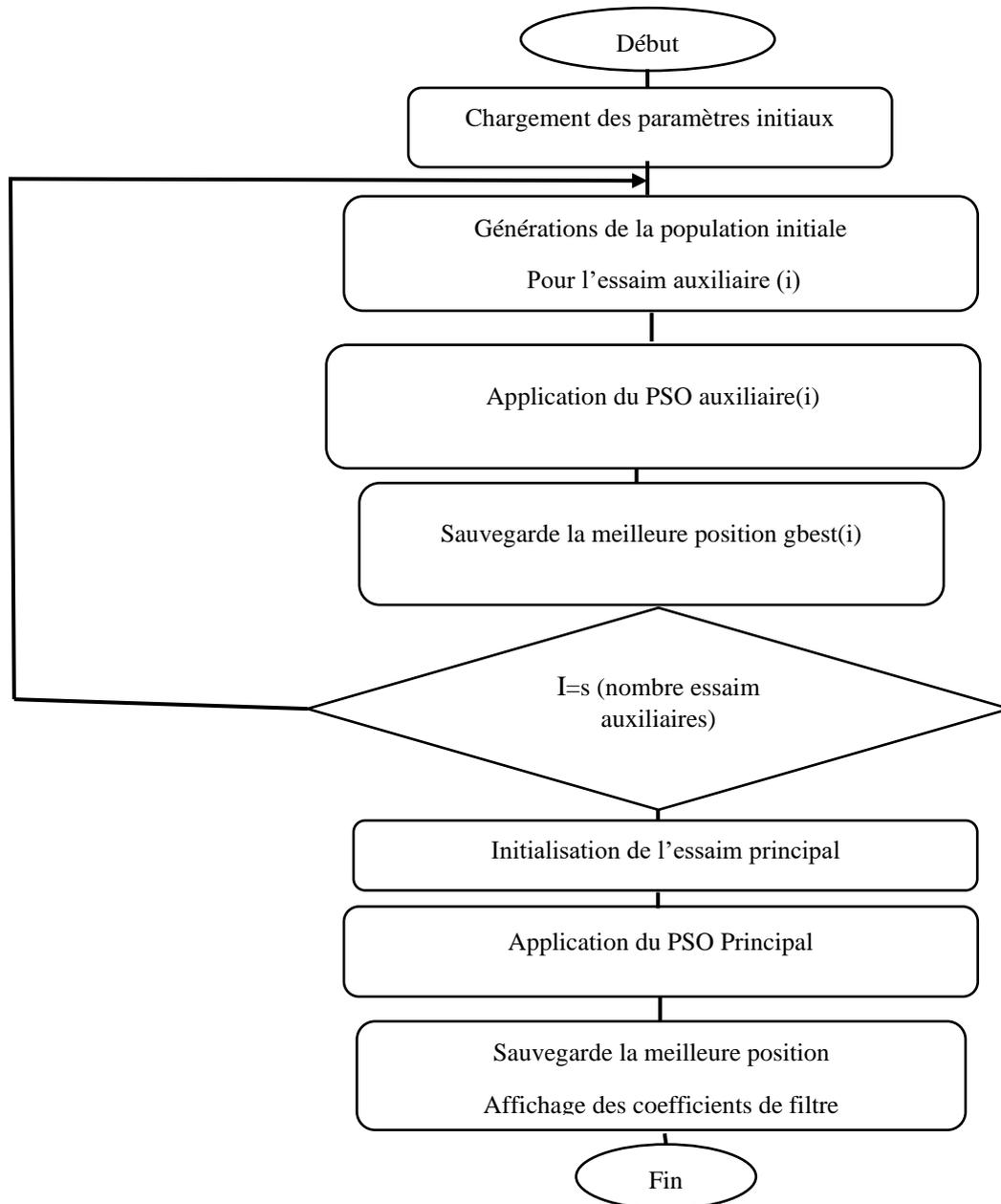
Etape 7 : Initialiser le principal essaim avec les meilleures positions des essaim auxiliaires.

Etape 8 : Exécution du PSO principal avec K' itérative période (update la vitesse  $\vec{v}_i$  et le vecteur de position  $\vec{x}_i$  et prendre la meilleure position globale).

Etape 9 : Sauvegarder la meilleure position globale.

Etape 10 : Afficher les coefficients et la réponse en amplitude de fréquence du filtre optimisé.

## 4.2.2. Organigramme d'algorithme proposé :



### 4.3.1. Formulation du problème de synthèse de filtres récursifs bidimensionnels

Considérons un filtre bidimensionnel RII avec sa fonction de transfert,  $H(z_1, z_2)$  de la forme [5] :

$$H(z_1, z_2) = A \prod_{k=1}^K \frac{\sum_{j=0}^{J_2^{(k)}} \sum_{i=0}^{J_1^{(k)}} b_{ij}^{(k)} z_1^i z_2^j}{\sum_{j=0}^{I_2^{(k)}} \sum_{i=0}^{I_1^{(k)}} d_{ij}^{(k)} z_1^i z_2^j} \quad (4.1)$$

$H(z_1, z_2)$ : se compose d'une cascade de  $K$  sections de filtre,

$b_{ij}^{(k)}$  et  $d_{ij}^{(k)}$ ,  $k = \overline{1, K}$ : sont les coefficients inconnus de chaque section,

$A$  est un gain constant qui peut être laissé arbitraire pendant la procédure de conception,

$J_1^{(k)}, J_2^{(k)}, I_1^{(k)}$  et  $I_2^{(k)}$  sont des entiers, ils doivent être choisis entre 0 et 2 afin d'assurer la stabilité.

La tâche de conception des filtres 2D est de trouver les coefficients de la fonction de transfert  $H(z_1, z_2)$  comme dans l'Eq. (1), telle que la fonction d'amplitude  $M(\omega_1, \omega_2) = |H(e^{-j\omega_1}, e^{-j\omega_2})|$  se rapproche de la réponse d'amplitude désirée  $M_d(\omega_1, \omega_2)$  dans un sens optimal.

Ainsi, la conception d'un filtre récursif 2D est équivalente au problème de minimisation suivant [1] :

$$\text{minim error} = \sum_{n_1=0}^{N_1-1} \sum_{n_2=0}^{N_2-1} (M(\omega_1, \omega_2) - M_d(\omega_1, \omega_2))^2 \quad (4.2)$$

Où  $\omega_1 = (\pi x n_1)/N_1$  et  $\omega_2 = (\pi x n_2)/N_2$  avec  $N_1$  ET  $N_2$ : nombres des échantillons du signal sur les deux axes.

Dans cette étude, le filtre-modèle est défini comme [1]:

$$M_d(\omega_1, \omega_2) = \begin{cases} 1 & \text{si } \sqrt{\omega_1^2 + \omega_2^2} \leq 0.08\pi \\ 0.5 & \text{si } 0.08\pi \leq \sqrt{\omega_1^2 + \omega_2^2} \leq 0.12\pi \\ 0 & \text{autre part} \end{cases} \quad (4.3)$$

Avec  $N_1 = N_2 = 50$  [1].

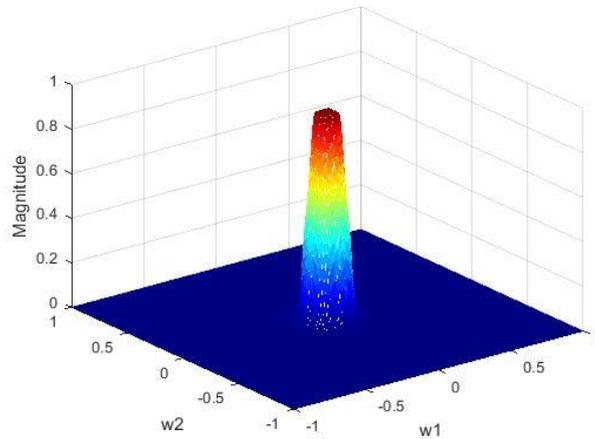


Figure 4.1 Réponse en amplitude désirée  $M_d(\omega_1, \omega_2)$  du filtre 2-D.

Pour résoudre le problème de stabilité, le travail dans [5] est exploité et un polynôme de la forme suivante est utilisé :

$$D(s_1, s_2) = \sum_{j=0}^{I_2} \sum_{i=0}^{I_1} d_{ij} s_1^i s_2^j \tag{4.4}$$

Où  $(s_1, s_2)$  sont deux variables de la transformée de Laplace et  $(I_2, I_1)$  sont deux entiers égaux à 1 ou 2. Les coefficients  $d_{ij}$  sont exprimés par des variables réelles non nulles  $[q_l]$ . La relation entre les termes  $d_{ij}$  et les paramètres  $[q_l]$  est exprimée par le réseau analogique (contient  $n_1$  condensateurs  $c_1 = s_1$  et  $n_2$  condensateurs  $c_2 = s_2$ ) comme le montre la figure 4.2 [5].

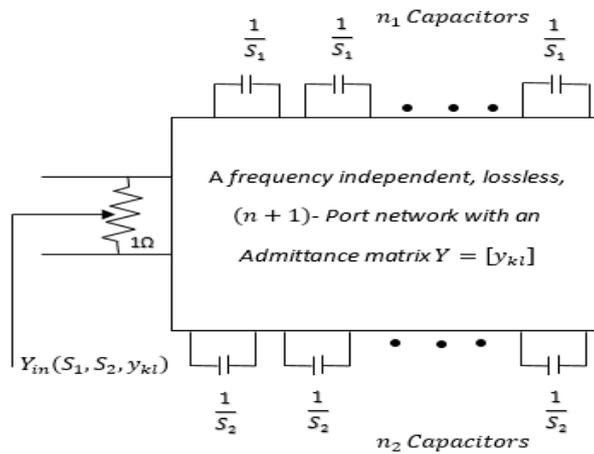


Figure 4.2 Réseau sans perte et indépendant de la fréquence.

Si le réseau est sans perte et indépendant de la fréquence, son admittance  $Y$  est réelle et peut être exprimée par [5] :

$$Y = \left[ \begin{array}{c|cccc} 0 & y_{12} & y_{13} & \cdots & y_{1n} \\ \hline -y_{12} & 0 & y_{23} & \cdots & y_{2n} \\ -y_{13} & -y_{23} & 0 & \cdots & y_{3n} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ -y_{1n} & -y_{2n} & -y_{3n} & \cdots & 0 \end{array} \right] = \left[ \begin{array}{c|c} Y_{11} & Y_{12} \\ \hline -Y_{12}^T & Y_{22} \end{array} \right] \quad (4.5)$$

L'admittance du réseau, qui se termine par une résistance à une borne et des condensateurs  $n_1 + n_2$  des autres bornes, peut être exprimée comme suit [49] :

$$\hat{Y}(s_1, s_2, y_{kl}) = Y + \text{diag} \left[ 1 \begin{array}{c} \overleftarrow{s_1} \quad \overrightarrow{s_1} \\ \cdots \quad \cdots \end{array} \quad \begin{array}{c} \overleftarrow{s_2} \quad \overrightarrow{s_2} \\ \cdots \quad \cdots \end{array} \right] \quad (4.6)$$

Selon la théorie du réseau analogique, l'admittance d'entrée peut être déduite comme suite :

$$Y_{in}(s_1, s_2, y_{kl}) = 1 + \frac{Y_{12} \text{adj}[\hat{Y}_{22}(s_1, s_2, y_{kl})] Y_{12}^T}{\det[\hat{Y}_{22}(s_1, s_2, y_{kl})]} \quad (4.7)$$

Le déterminant de la matrice d'admittance  $[Y_{in}(s_1, s_2, y_{kl})]$  est le polynôme  $D(s_1, s_2, y_{kl})$  et tant que le réseau sans perte et termine par une résistance  $D(s_1, s_2, y_{kl})$  est un polynôme de Hurwitz de deux variables  $s_1$  et  $s_2$ .

Ce qui veut dire que le polynôme  $D(s_1, s_2, y_{kl})$  obtenu est le déterminant de la matrice d'admittance d'un réseau sans perte et indépendant de la fréquence se termine par des condensateurs  $s_1, s_2$  et une résistance à l'entrée ce qui satisfait la condition.

$$D(s_1, s_2, y_{kl}) \neq 0 \text{ pour } \text{Re}(s_1) \geq 0 \text{ et } \text{Re}(s_2) \geq 0$$

Le résultat obtenu ci-dessus est illustré dans le réseau de trois bornes, qui se termine par un condensateur  $s_1$ , un condensateur  $s_2$  et une résistance  $R = 1$  Ohm, comme le montre la figure 4.3.

La matrice d'admittance est [5].

$$\hat{Y}(s_1, s_2, y_{kl}) = \begin{bmatrix} 1 & y_{12} & y_{13} \\ -y_{12} & s_1 & y_{23} \\ -y_{13} & -y_{23} & s_2 \end{bmatrix} \quad (4.8)$$

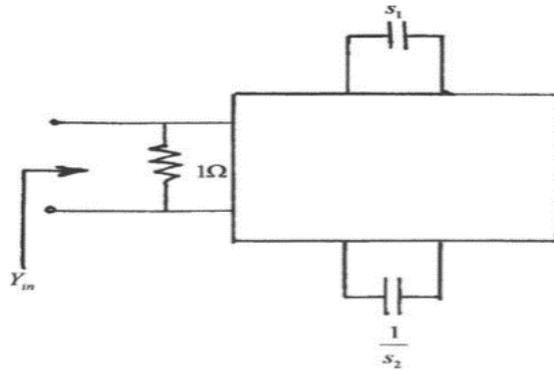


Figure 4.3. Réseau de trois bornes.

Le déterminant de  $\hat{Y}(s_1, s_2, y_{kl})$  est [5] :

$$\det(\hat{Y}(s_1, s_2, y_{kl})) = y_{23}^2 + y_{13}^2 s_1 + y_{12}^2 s_2 + s_1 s_2 \quad (4.9)$$

et aussi

$$D(s_1, s_2) = \sum_{j=0}^{l_2} \sum_{i=0}^{l_1} d_{ij} s_1^i s_2^j = d_{00} + d_{10} s_1 + d_{01} s_2 + d_{11} s_1 s_2 \quad (4.10)$$

En comparant (9) et (10), le résultat suivant est obtenu :

$$d_{00} = y_{23}^2, \quad d_{10} = y_{13}^2, \quad d_{01} = y_{12}^2 \text{ et } d_{11} = 1.$$

On pose :  $y_{23}=q_1$ ;  $y_{13}=q_2$ ; et  $y_{12}=q_3$ .

Les coefficients  $d_{ij}$  sont exprimés par les paramètres  $q_i$ .

Cas I:  $I_1=I_2=1$

$$d_{00} = q_1^2,$$

$$d_{10} = q_2^2,$$

$$d_{01} = q_3^2,$$

$$d_{11} = 1.$$

Prenons un réseau sans perte terminé par deux condensateurs  $s_1$ , un condensateur  $s_2$  et une résistance d'entrée ( $n_1=2, n_2=1$ ) [5].

$$\hat{Y}(s_1, s_2, y_{kl}) = \begin{bmatrix} 1 & y_{12} & y_{13} & y_{14} \\ -y_{12} & s_1 & y_{23} & y_{24} \\ -y_{13} & y_{23} & s_1 & y_{34} \\ -y_{14} & -y_{24} & -y_{34} & s_2 \end{bmatrix}$$

(4.11)

En prenant :

$$y_{12}=q_1, y_{13}=q_2, y_{14}=q_3, y_{23}=q_4, y_{24}=q_5 \text{ et } y_{34}=q_6,$$

Les coefficients  $d_{ij}$  peuvent être déduits comme :

$$d_{00} = (q_1q_6 - q_2q_5 + q_3q_4)^2,$$

$$d_{10} = q_5^2 + q_6^2,$$

$$d_{20} = q_3^2,$$

$$d_{01} = q_4^2,$$

$$d_{11} = q_1^2 + q_2^2,$$

$$d_{21} = 1.$$

Ainsi, quand  $I_1=2$  et  $I_2=2$ , on a [49] :

$$d_{00} = (q_5q_{10} + q_6q_9 + q_7q_8)^2,$$

$$d_{10} = (q_1q_{10} - q_3q_7 + q_4q_6)^2 + (q_2q_{10} - q_3q_9 + q_4q_8)^2,$$

$$d_{20} = q_{10}^2,$$

$$d_{01} = (q_1q_8 - q_2q_6 + q_3q_5)^2 + (q_1q_9 - q_2q_7 + q_4q_5)^2,$$

$$d_{11} = q_6^2 + q_7^2 + q_8^2 + q_9^2,$$

$$d_{21} = q_3^2 + q_4^2,$$

$$d_{02} = q_5^2,$$

$$d_{12} = q_1^2 + q_2^2,$$

$$d_{22} = 1,$$

Il est à noter que lorsque  $D(s_1, s_2)$  satisfait les conditions de stabilité, le  $D(z_1, z_2)$ , déduit par la transformation bilinéaire, satisfait les conditions de stabilité aussi [49] :

$$D(s_1, s_2) \Big|_{s_1 = \frac{(1-z_1^{-1})}{(1+z_1^{-1})}, s_2 = \frac{(1-z_2^{-1})}{(1+z_2^{-1})}} = D(z_1, z_2) \neq 0 \quad (4.12)$$

$$\text{pour } |z_1| \geq 1 \text{ et } |z_2| \geq 1$$

Cela signifie que [49], le polynôme  $D(z_1, z_2)$  n'a pas de zéros en dehors de la région :

$$\{|z_1| \cup |z_2|; |z_1| < 1 \text{ and } |z_2| < 1\} \quad (4.13)$$

Pour cette étude deux types de fonction de transfert  $H(z_1, z_2)$  seront exploités :

- Générale fonction de transfert de deuxième ordre.
- Fonction de transfert avec un numérateur polynôme miroir.

### 4.3.2 Fonction de transfert

Dans cette étude on utilise deux types de fonction.

-Fonction de transfert de deuxième ordre [1]:

$$H(z_2, z_2) = h_0 \frac{1 + a_{01}z_2 + a_{02}z_2^2 + a_{10}z_1 + a_{20}z_1^2 + a_{11}z_1z_2 + a_{12}z_1z_2^2 + a_{21}z_1^2z_2 + a_{22}z_1^2z_2^2}{(1 + b_1z_1 + c_1z_2 + d_1z_1z_2)(1 + b_2z_1 + c_2z_2 + d_2z_1z_2)} \quad (4.14)$$

L'amplitude est :

$$M(\omega_1, \omega_2) = |H(z_1, z_2)| = H_0 \left| \frac{A_R - jA_I}{(B_{1R} - jB_{1I})(B_{2R} - jB_{2I})} \right| \quad (4.15)$$

Avec  $z_1 = e^{-j\omega_1}$  et  $z_2 = e^{-j\omega_2}$

Et

$$A_R = a_{00} + a_{01}\cos(\omega_2) + a_{02}\cos(2\omega_2) + a_{10}\cos(\omega_1) + a_{20}\cos(2\omega_1) + a_{11}\cos(\omega_1 + \omega_2) \\ + a_{12}\cos(\omega_1 + 2\omega_2) + a_{21}\cos(2\omega_1 + \omega_2) + a_{22}\cos(2\omega_1 + 2\omega_2)$$

$$A_I = a_{01}\sin(\omega_2) + a_{02}\cos(2\omega_2) + a_{10}\sin(\omega_1) + a_{20}\sin(2\omega_1) + a_{11}\sin(\omega_1 + \omega_2) + a_{12}\sin(\omega_1 + 2\omega_2) + a_{21}\sin(2\omega_1 + \omega_2) + a_{22}\sin(2\omega_1 + 2\omega_2)$$

$$B_{1R} = 1 + b_1 \cos(\omega) + c_1 \cos(\omega_2) + d_1 \cos(\omega_1 + \omega_2)$$

$$B_{1I} = b_1 \sin(\omega_1) + c_1 \sin(\omega_2) + d_1 \sin(\omega_1 + \omega_2)$$

$$B_{2R} = 1 + b_2 \cos(\omega_1) + c_2 \cos(\omega_2) + d_2 \cos(\omega_1 + \omega_2)$$

$$B_{2I} = b_2 \sin(\omega_1) + c_2 \sin(\omega_2) + d_2 \sin(\omega_1 + \omega_2)$$

Fonction de transfert avec un numérateur polynôme miroir [49] :

$$H(z_1, z_2) = H_0 \left( \frac{z_1^{-1} z_2^{-1} (b_{11} + b_{01}(z_1 + z_2^{-1}) + b_{10}(z_2 + z_2^{-1}))}{D_1 \cdot D_2} + \frac{z_1^{-1} z_2^{-1} (b_{02}(z_1 z_2^{-1} + z_2 z_1^{-1}) + b_{00}(z_1 z_2 + z_1^{-1} z_2^{-1}))}{D_1 \cdot D_2} \right) \quad (4.16)$$

Ou

$$D_1 = 1 + b_1 z_1^{-1} + c_1 z_2^{-1} + d_1 z_1^{-1} z_2^{-1}$$

$$D_2 = 1 + b_2 z_1^{-1} + c_2 z_2^{-1} + d_2 z_1^{-1} z_2^{-1}$$

L'amplitude est :

$$M(\omega_1, \omega_2) = H_0 \left| \frac{A_r}{(B_{1r} + jB_{1i})(B_{2r} + jB_{2i})} \right| \quad (4.17)$$

Avec

$$A_r = b_{11} + 2b_{01} \cos(\omega_1) + 2b_{10} \cos(\omega_2) + 2b_{02} \cos(\omega_1 - \omega_2) + 2b_{00} \cos(\omega_1 + \omega_2),$$

$$B_{1r} = 1 + b_1 \cos(\omega_1) + c_1 \cos(\omega_2) + d_1 \cos(\omega_1 + \omega_2),$$

$$B_{1i} = b_1 \sin(\omega_1) + c_1 \sin(\omega_2) + d_1 \sin(\omega_1 + \omega_2),$$

$$B_{2r} = 1 + b_2 \cos(\omega_1) + c_2 \cos(\omega_2) + d_2 \cos(\omega_1 + \omega_2),$$

$$B_{2i} = b_2 \sin(\omega_1) + c_2 \sin(\omega_2) + d_2 \sin(\omega_1 + \omega_2).$$

#### 4.4. Résultats et discussion

Pour la simulation du programme. Les paramètres de l'algorithme Multi-PSO sont présentés dans le tableau 1 :

**Tableau 1.** Paramètres de l'algorithme Multi-PSO

Paramètres	Valeur
Nombre des essaim auxiliaires	10
Population de d'essaim auxiliaire	150
Itératives périodes pour le PSO auxiliaire	100
Itératives périodes pour le PSO principal	100
Coefficient d'inertie w	0.4 to 0.9
Coefficients d'accélération	C <sub>1</sub> =C <sub>2</sub> =2.05

- Générale fonction de transfert de deuxième ordre :

$$H(z_1, z_2) = h_0 \frac{1 + a_{01}z_2 + a_{02}z_2^2 + a_{10}z_1 + a_{20}z_1^2 + a_{11}z_1z_2 + a_{12}z_1z_2^2 + a_{21}z_1^2z_2 + a_{22}z_1^2z_2^2}{(1 + b_1z_1 + c_1z_2 + d_1z_1z_2)(1 + b_2z_1 + c_2z_2 + d_2z_1z_2)} \quad (4.18)$$

Le vecteur optimisé x :

$$x(a_{01} \ a_{02} \ a_{10} \ a_{11} \ a_{12} \ a_{20} \ a_{21} \ a_{22} \ b_1 \ c_1 \ d_1 \ b_2 \ c_2 \ d_2 \ h_0)$$

$$x(0.4083-0.3521 \ 0.2272 \ 0.3112-0.7478-1.114 \ 0.7196 \ 1.7258 \ -0.9279-0.9274 \ 0.8765-0.4518-0.4249-0.0175 \ 0.001)$$

L'erreur minimale obtenue après l'optimisation est :

$$\text{Error1} = 1.2216$$

- Fonction de transfert avec un numérateur polynôme miroir :

$$H(z_1, z_2) = H_0 \left( \frac{z_1^{-1}z_2^{-1}(b_{11} + b_{01}(z_1 + z_2^{-1}) + b_{10}(z_2 + z_2^{-1}))}{D_1 \cdot D_2} + \frac{z_1^{-1}z_2^{-1}(b_{02}(z_1z_2^{-1} + z_2z_1^{-1}) + b_{00}(z_1z_2 + z_1^{-1}z_2^{-1}))}{D_1 \cdot D_2} \right) \quad (4.19)$$

ou

$$D_1 = 1 + b_1z_1^{-1} + C_1z_2^{-1} + d_1z_1^{-1}z_2^{-1}$$

$$D_2 = 1 + b_2z_1^{-1} + C_2z_2^{-1} + d_2z_1^{-1}z_2^{-1}$$

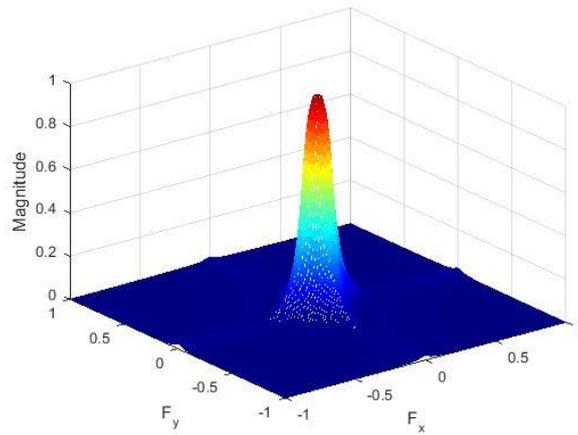
L'erreur minimale obtenue après optimisation :

$$\text{Error2} = 1.1428$$

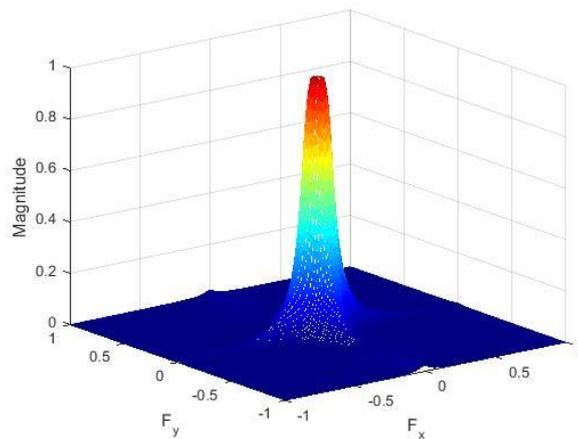
Le vecteur  $x$

$$x(b_{00} \ b_{01} \ b_{02} \ b_{10} \ b_{11} \ b_1 \ c_1 \ d_1 \ b_2 \ c_2 \ d_2 \ H_0)$$

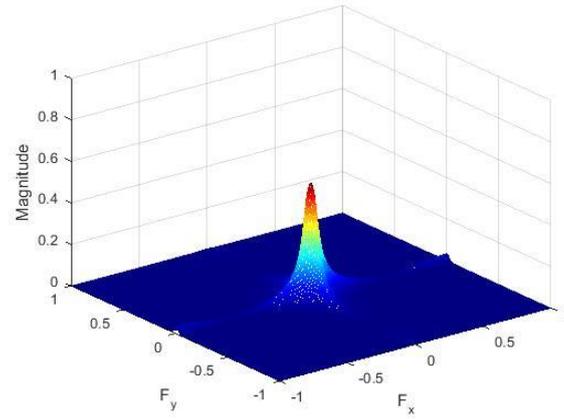
$$x(0.2444 \ -0.9507 \ -0.3804 \ -1.4545 \ -0.1707 \ -0.0008 \ -0.1742 \ -0.6908 \ -0.3747 \ 0.1565 \ -0.5511 \ 0.0065)$$



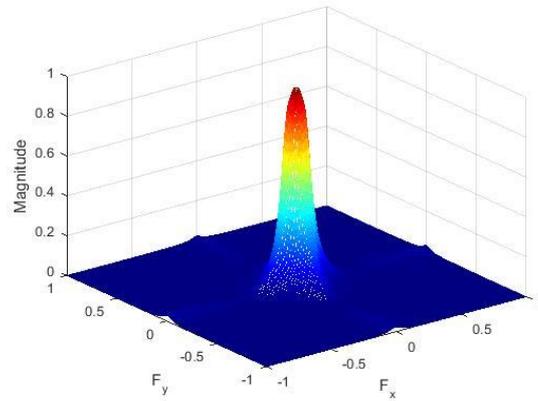
(a) HMAPSO



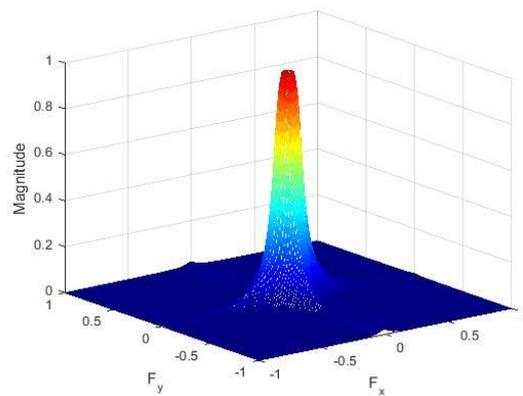
(b) Algorithme proposé (equation 4.18)



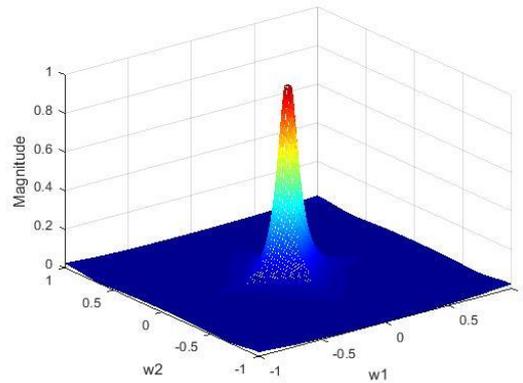
(c) MEPSO



(d) SA-PSO



(e) Algorithme proposé (équation 4.19)



(f) BBO-PSO

**Figure 4.4** Comparaison des réponses en amplitude  $M(\omega_1, \omega_2)$  en utilisant différents algorithmes (MEPSO, HMAPSO, SA-PSO, BBO – PSO et algorithme proposé)

**Note :** dans Figure 4  $F_x = \omega_1$  and  $F_y = \omega_2$

Afin de prouver l'efficacité de notre algorithme proposé[2], il est comparé à quelques travaux importants : MEPSO [50], HMAPSO [51], BBO – PSO [52] et SA-PSO [1] utilisent l'algorithme PSO et ils ont prouvé leurs bonnes performances par rapport aux autres algorithmes.

Tableau 2. Résultats obtenus (coefficients de filtre et erreur minimale) avec différents algorithmes

Coefficients	BBO-PSO	SA-PSO	MEPSO	HMAPSO	Algorithme Proposé.
$a_{01}$	0.3840	0.3069	0.3061	0.5815	0.4083
$a_{02}$	-0.8816	-0.9806	0.9949	0.2207	-0.3521
$a_{10}$	-0.4478	0.1681	0.3935	0.4387	0.2272
$a_{11}$	-0.7609	-0.0431	-0.0338	0.4045	0.3112
$a_{12}$	1.1371	-0.1820	0.6481	-1.4084	-0.7478
$a_{20}$	-0.1803	-0.7270	1.2345	-0.5720	-1.1144
$a_{21}$	0.2938	-0.3249	0.5030	-0.8418	0.7196
$a_{22}$	0.2473	1.6358	0.4481	2.277	1.7258
$b_1$	-0.1249	-1.4201	-1.0239	-0.9078	-0.9279
$b_2$	-1.0553	-0.9178	0.0342	-0.9058	-0.9274
$c_1$	-0.1890	-0.6530	-0.9605	0.8373	0.8765
$c_2$	-1.0594	-0.9127	-0.0371	-0.9075	-0.4518
$d_1$	-0.8986	1.0081	0.9523	-0.9101	-0.4249
$d_2$	1.1366	0.8545	-0.9056	0.8406	-0.0175
$h_0$	0.0013	0.0022	0.00034	0.00024	0.001
Min.Err	2.8518	2.629	9.0005	1.58657	1,221

- le test de la condition de stabilité [1, 50, 51,52] :

$$|b_k + c_k| - 1 < d_k < 1 - |b_k - c_k|$$

Afin de vérifier la stabilité des filtres obtenus par MEPSO, HMAPSO, BBO – PSO et SA-PSO, nous utilisons les valeurs de  $b_k$ ,  $c_k$  et  $d_k$  ( $k = 1, 2$ ) dans le tableau 2

Tableau 3. Tableau de contrôle de stabilité.

ALGORITHME	Filtre résultat
MEPSO	Instable
SA-PSO	Instable
BBO-PSO	Stable
HMAPSO	Stable
<b>Algorithme</b> (Multi-PSO)	Stable

À partir du tableau 2, il est facile de vérifier que notre algorithme proposé (Multi-PSO) est mieux classé en termes de bonne optimisation quantifiée par la valeur d'erreur minimale (erreur = 1,2216), au deuxième rang, vient le HMAPSO avec une valeur d'erreur (error = 1,586576) et en dernier BBO – PSO avec MEPSO.

On peut remarquer que pour les algorithmes SA-PSO et MEPSO, les auteurs affirment que le problème d'instabilité est résolu par une bonne optimisation, alors que leurs filtres résultants sont instables lors de la vérification de la condition de stabilité (tableau 3). D'autre part, notre algorithme proposé[2] assure toujours la stabilité du filtre.

Selon les résultats obtenus, il est clair que les réponses en amplitude de fréquence dans les deux méthodes (équation (4.18) et (4.19)) de notre algorithme proposé (voir figure 4.4) sont très proches du filtre du modèle (voir figure 4.1).

De plus, les erreurs minimales obtenues par notre modèle après optimisation (erreur1 et erreur2) sont très faibles par rapport aux résultats donnés par les algorithmes HMAPSO, SA-PSO, BBO – PSO et MEPSO. De plus, notre modèle est toujours stable.

## 4.5 Conclusion

Dans cette étude, un nouvel algorithme Multi-PSO pour la conception de filtre numérique récursif bidimensionnel est étudié. Il se caractérise par la diversité des particules et l'utilisation de deux idées. La première consiste à utiliser deux types d'essaims, un principal et des

auxiliaires. La deuxième idée est la partition de l'espace de recherche en plusieurs zones. Ainsi, conduisant à une meilleure exploration du problème. Les résultats obtenus avec Multi-PSO sont comparés à d'autres algorithmes précédemment rapportés. Ils indiquent que les méthodes basées sur Multi-PSO présentent de meilleures performances dans toutes les expériences et fournissent la meilleure solution optimale pendant le mécanisme de recherche. En conclusion, ces nouvelles idées présentées ci-dessus ont ouvert la porte à de nouvelles méthodes qui pourraient être appliquées à d'autres algorithmes d'optimisation, tels que les algorithmes évolutionnaires. Mais le Multi-PSO proposé prend beaucoup de temps en simulation, ce qui est un inconvénient.

## Conclusion générale

Dans cette étude, nous essayons une nouvelle variante de l'algorithme original d'optimisation par essaim des particules, qui est le Muti-PSO dans lequel deux idées ont été ajoutées : la première idée est l'utilisation de deux types d'essaims, un principal et des auxiliaires. La deuxième idée est la partition de l'espace de recherche en plusieurs zones. L'algorithme utilise une nouvelle stratégie d'initialisation. Cette stratégie est basée sur deux types d'essaims : l'essaim principal désigné par  $S_1$ , et de  $s$  essaims auxiliaires notés  $S_{2i}$ , où  $1 \leq i \leq s$ . Dans l'algorithme Multi-PSO, chaque essaim auxiliaire est destiné à découvrir une zone de l'espace de recherche en effectuant un certain nombre d'itérations et son rôle se termine lorsque la meilleure position de la zone est atteinte. Les meilleures positions obtenues par tous les essaims auxiliaires sont regroupées dans un tableau de taille égale au nombre d'essaims initialisés et aussi au nombre de zones. Les positions dans ce tableau composeront alors la population initiale de l'essaim principal. Une fois initialisé, il continue la recherche de la même manière que l'algorithme PSO standard et les essaims auxiliaires ne sont plus utilisés.

Les résultats obtenus avec Multi-PSO sont comparés à d'autres algorithmes précédemment rapportés. Ils indiquent que les méthodes basées sur Multi-PSO présentent de meilleures performances dans toutes les expériences et fournissent la meilleure solution optimale pendant le mécanisme de recherche. En conclusion, ces nouvelles idées présentées ci-dessus ont ouvert la porte à des nouvelles méthodes qui pourraient être appliquées à d'autres algorithmes d'optimisation, tels que les algorithmes évolutionnaires.

### Références bibliographiques

- [1] Dhabal, S., Venkateswaran, P. "Two-dimensional IIR filter design using simulated annealing-based particle swarm optimization". *Journal of Optimization*, 2014. <https://doi.org/10.1155/2014/239721>
- [2] Lakhdar Kaddouri, Amel B.H. Adamou-Mitiche, Lahcene Mitiche."Design of two-dimensional recursive digital filter using multi particle swarm optimization algorithm". *Journal Européen des Systèmes Automatisés* Vol. 53, No. 4, pp. 559-566, August 2020.
- [3] Lu, W.S., Antoniou, A. *Two-dimensional digital filters*. CRC Press, :1-4,124-127,1992.
- [4] Jae S.Lim "two dimensional signal and image processing" Prentice Hall, signal processing series Alan V.Oppenheim series 1990.
- [5] P.A. Ramamoorthy and L.T. Bruton, "Frequency domain approximation of stable multi-dimensional discrete recursive filters" *Proc.of Int'l Symp. on Circuits and Systems*, pp. 654-657, April 1977.
- [6] S.A.H. Aly and M.M. Fahmy, "Design of two-dimensional recursive digital filters with specified magnitude and group delay characteristics," *IEEE Trans. on Circuits and Systems, CAS-25*, pp. 908-915, November 1978.
- [7] T. O'Connor and Thomas S. Huang "Stability of General Two-Dimensional recursive Digital Filters" *IEEE transactions on acoustics, speech, and signal processing, VOL.26*, pp.550-560, December 1978.
- [8] Liejun Xie "A Criterion for Hurwitz Polynomials and its Applications" *I.J.Modern Education and Computer Science*, pp.38-40,2011.
- [9] J. M. Costa and A. N. Venetsanopoulos," Design of circularly symmetric two-dimensional recursive filters", *IEEE Tram. Acoust., Speech, Signal Process.*, vol. 22, Dec 1974.
- [10] T. Hinamoto and S. Maekawa, "Design of two-dimensional recursive digital filters using mirror image polynomials," *IEEE Trans. on Circuits and Systems*, vol.33, 1986.
- [11] N. Mastorakis, I. F. Gonos, and M. N. S. Swamy, "Design of two-dimensional recursive filters using genetic algorithms" *IEEE Transactions on Circuits and Systems*, vol.50,2003.
- [12] V. M. Mladenov and N. E. Mastorakis, "Design of two-dimensional recursive filters by using neural networks," *IEEE Transactions on Neural Networks*, vol. 12, 2001.

- [13] J. Kennedy and R. C. Eberhart. "Particle Swarm Optimization". IEEE International Conference on Neural Networks, vol. IV, pp. 1942–1948, November 1995.
- [14] Y. Shi, R.C. Eberhart, Empirical study of particle swarm optimization, in: Proceedings of Congress on Evolutionary Computation, Washington, DC, pp. 1945–1949, 1999.
- [15] Particle Swarm Central. <http://particleswarm.info>
- [16] S. Das, A. Abraham, and A. Konar. "Particle swarm optimization and differential evolution algorithms: technical analysis, applications and hybridization perspectives". *Computational Intelligence (SCI)*, Vol. 116, pp. 1–38, 2008.
- [17] Y. Shi, R.C. Eberhart, A modified particle swarm optimizer, in: Proceedings of IEEE International Conference on Evolutionary Computation, Anchorage, AK, pp. 69–73, May 1998.
- [18]. Niu, B., Zhu, Y., He, X., Wu, H.: MCPSO: a multi-swarm cooperative particle swarm optimizer. *Appl.Math. Comput.* 185, 1050–1062, 2007.
- [19] René Garello "Two-Dimensional Signal Analysis" Hermes Science Publications, PP 17-50, 2001.
- [20] DUDGEON D. E., MERSEREAU R. M., *Multidimensional Digital Signal Processing*, Englewood Cliffs, NJ, Prentice-Hall, Signal Processing Series, 1984.
- [21] GUYON X., "Random Fields on a Network. Modeling, Statistics, and Applications", Springer-Verlag, 1995.
- [22] JAE S.LIM "Two-dimensional signal and image processing", PH, Year: 1989.
- [23] R.Fletcher, *Practical Methods of Optimization*, 2nd ed., Chichester: Wiley, 1987.
- [24] D. G.Luenberger, *Linear and Nonlinear Programming*, 2nd ed., Reading, Mass.: Addison-Wesley, 1984.
- [25] P. K. Rajan and M. N. S. Swamy, Quadrantal symmetry associated with two-dimensional digital transfer functions, *IEEE Trans. Circuits Syst.*, vol. CAS- 29, pp. 340-343, June 1983.
- [26] C. Charalambous, A unified review of optimization, *IEEE Trans. Microwave Theory and Techniques*, vol. "IT-22, pp. 289-300, March 1974.
- [27] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to algorithms*, chapitre 16 :Greedy Algorithms. MIT Press and McGraw-Hill, 1st Ed., 1990.

- [28] R. A. DeVore and V. N. Temlyakov. “Some remarks on greedy algorithms”. *Advances in Computational Mathematics*, Vol. 5, No. 1, pp. 173–187, 1996.
- [29] F. Glover. “Future paths for integer programming and links to artificial intelligence”. *Computers and Operations Research*, Vol. 13, pp. 533–549, 1986.
- [30] Y. Collette and P. Siarry. *Optimisation multiobjectif*. Eyrolles, 2002.
- [31] D. E. Goldberg and J. Richardson. “Genetic algorithms with sharing for multimodal function optimization”. In: *Proceedings of the Second International Conference on Genetic Algorithms on Genetic algorithms and their application*, pp. 41–49, Lawrence Erlbaum Associates., Mahwah, NJ, USA, 1987.
- [32] J. Branke. *Evolutionary Optimization in Dynamic Environments*. Kluwer Academic, 2002.
- [33] E. G. Talbi. “A Taxonomy of Hybrid Metaheuristics”. *Journal of Heuristics*, Vol. 8, No. 5, pp. 541–564, 2002.
- [34] E. Alba. *Parallel Metaheuristics: A New Class of Algorithms*. John Wiley & sons, 2005.
- [35] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. “Optimization by Simulated Annealing”. *Science*, Vol. 220, 4598, pp. 671–680, 1983.
- [36] V. Cerny. “Thermodynamical Approach to the Travelling Salesman Problem: An Efficient Simulation Algorithm”. *Journal of Optimization Theory and Applications*, Vol. 45, pp. 41–51, 1985.
- [37] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. “Equation of State Calculations by Fast Computing Machines”. *The Journal of Chemical Physics*, Vol. 21, No. 6, pp. 1087–1092, 1953.
- [38] W. K. Hastings. “Monte-Carlo sampling method using Markov chains and their applications”. *Biometrika*, Vol. 57, 1970.
- [39] M. Creutz. “Microcanonical Monte Carlo simulation”. *Physical Review Letters*, Vol. 50, No. 19, pp. 1411–1415, 1983.
- [40] Y. Okamoto and U. H. E. Hansmann. “Thermodynamics of helix-coil transitions studied by multicanonical algorithms”. *J. Phys. Chem.*, No. 99, pp. 11276–11287, 1995.

- [41] S. Goss, S. Aron, J. L. Deneubourg, and J. M. Pasteels. “Self-organized shortcuts in the Argentine ant”. *Naturwissenschaften*, Vol. 76, No. 12, pp. 579–581, 1989.
- [42] R. Storn and K. Price. “Differential Evolution - A Simple and Efficient Heuristic for global Optimization over Continuous Spaces”. *Journal of Global Optimization*, Vol. 11, No. 4, pp. 341–359, 1997.
- [43] A. S. Fraser. “Simulation of genetic systems by automatic digital computers”. *Australian Journal of Biological Science*, Vol. 10, pp. 484–491, 1957.
- [44] R. Eberhart, P. Simpson, and R. Dobbins. *Computational Intelligence PC Tools*. AP Professional, 1996.
- [45] Kennedy, J. The behavior of particles. LNCS, Springer, In Proceedings of the 7th Conference on Evolutionary Computation, pp. 581–589, 1998. <https://doi.org/10.1007/BFb0040809>
- [46] Ozcan, E. Mohan, C. Particle Swarm Optimization: surfing the waves. *IEEE Congress on Evolutionary Computation*, pp. 1939-1944, 1999. <https://doi.org/10.1109/CEC.1999.785510>
- [47] Clerc, M., Kennedy, J. The particle swarm: explosion, stability, and convergence in multidimensional complex space. *IEEE Transactions on Evolutionary Computation*, 20(1), 2002: 1671-1676, 2002. <https://doi.org/10.1109/4235.985692>.
- [48] R. Eberhart and Y. Shi. “Comparing inertia weights and constriction factors in particle swarm optimization”. In: *Proceedings of the 6th IEEE Congress on Evolutionary Computation*, pp. 84–88, IEEE Press, 2000.
- [49] Sheno, B.A. *Magnitude and Delay Approximation of 1-D and 2-D Digital Filters*. Springer Verlag Berlin Heidelberg, 190-195. 1st edition, 1999.
- [50] Das, S., Konar, A. A swarm intelligence approach to the synthesis of two-dimensional IIR filters. *Engineering Applications of Artificial Intelligence*, 20(8):1086-1096, 2007.
- [51] Kumar, R., Kumar, A. Design of two-dimensional infinite impulse response recursive filters using hybrid multiagent particle swarm optimization. *Applied Artificial Intelligence*, 24(4): 295-312, 2010. <https://doi.org/10.1080/08839511003715204>

- [52] Lv, C., Yan, S., Cheng, G., Xu, L., Tian, X.Y. Design of two-dimensional IIR digital filters by using a novel hybrid optimization algorithm. *Multidim. Syst. Sign. Process*, 28: 1267-1281, 2016. <https://doi.org/10.1007/s11045-016-0397-0>