N° d'ordre : 3336

THESE

Présentée à

L'Université des Sciences et Technologies de LILLE

en vue de l'obtention du grade de

DOCTEUR DE L'UNIVERSITE

Discipline

Automatique et Informatique Industrielle

par

Christophe LURETTE

DEVELOPPEMENT D'UNE TECHNIQUE NEURONALE AUTO-ADAPTATIVE POUR LA CLASSIFICATION DYNAMIQUE DE DONNEES EVOLUTIVES. APPLICATION A LA SUPERVISION D'UNE PRESSE HYDRAULIQUE.

Soutenue le 17 septembre 2003 devant la commission d'examen

Présidente :	Mireille BAYART	*Professeur à Polytech'Lille*
Rapporteurs :	Stéphane CANU	*Professeur à l'INSA de Rouen*
	Gérard GOVAERT	*Professeur à l'UTC de Compiègne*
Examinateur :	Dimitris TSAPTSINOS	*Principal Lecturer à l'Université de Kingston (UK)*
Invité :	Didier DELLYS	*Représentant de la société Arc International*
Directeurs de thèse :	Stéphane LECOEUCHE	*Enseignant-Chercheur à l'Ecole des Mines de Douai* (*Co-encadrant)*
	Christian VASSEUR	*Professeur à l'Université des Sciences et Technologies de Lille*

Remerciements

Ce travail a été réalisé au sein de la direction Recherche & Développement de la société ARC International (AI), dans le cadre d'une convention CIFRE, avec le Laboratoire Interaction, Image & Ingénierie de la Décision (I3D) de l'Université de Lille 1 (FRE-CNRS-2497) et l'Ecole d'Ingénieurs du Pas-de-Calais (EIPC). A ce titre, j'adresse mes premiers remerciements à l'ensemble des personnes qui ont contribué à l'élaboration de cette convention. J'en profite également pour remercier l'Agence Nationale de la Recherche et de la Technologie (ANRT).

Pour avoir pu réaliser mes travaux de thèse au sein d'AI, je tiens à présenter mes remerciements les plus sincères à Monsieur E. Himpens, Directeur industriel, ainsi qu'à Monsieur D. Lalart, Directeur R&D. Par ailleurs, que Monsieur D. Dellys, Responsable du service Amélioration des Moyens de Production (AMP) au sein duquel j'ai réalisé mes travaux de thèse, trouve ici l'expression de mes plus profonds remerciements pour l'attention qu'il m'a témoignée durant ces trois années et pour la confiance qu'il m'a accordée lors de mon embauche.

Je tiens à exprimer ma gratitude à Monsieur S. Canu, Professeur à l'Institut National des Sciences Appliquées de Rouen (INSA) et à Monsieur G. Govaert, Professeur à l'Université de Technologie de Compiègne (UTC), pour avoir accepté de rapporter ce mémoire.

Je remercie Madame M. Bayart, Professeur à l'Ecole Polytechnique Universitaire de Lille (Polytech'Lille) et Monsieur D. Tsaptsinos de l'Université de Kingston, pour avoir accepté de participer au jury en tant qu'examinateurs. Par la même occasion, je remercie Monsieur D. Dellys pour sa participation en tant que représentant d'ARC International.

J'exprime mes plus vifs remerciements à Monsieur C. Vasseur, Professeur à l'Université des Sciences et Technologies de Lille (USTL) et Directeur du Laboratoire I3D, pour avoir dirigé ce travail de thèse. Je le remercie pour toute la confiance qu'il m'a accordée et pour l'intérêt qu'il a montré envers mes travaux durant toutes les discussions que nous avons pu avoir.

Une simple formule de remerciements ne suffit pas à exprimer toute ma gratitude à Monsieur S. Lecoeuche, ancien enseignant-chercheur à l'Ecole d'Ingénieurs du Pas-de-Calais (EIPC) et désormais en poste à l'Ecole Nationale Supérieure des Mines de Douai (EMD), pour avoir co-encadré et participé activement à la réalisation de ce travail. Je souhaite d'ailleurs remercier tout particulièrement l'EMD pour avoir accepté que Monsieur S. Lecoeuche poursuive son activité d'encadrement durant les derniers mois.

Je tiens également à remercier Messieurs S. Lalot et G. Mercere pour tout le temps qu'ils ont consacré à la relecture de mon manuscrit et pour tous les conseils qu'ils ont su me donner durant la rédaction de ce dernier.

Je n'oublie pas de remercier toutes les personnes qui par leurs conseils et leur soutien m'ont permis de mener à bien ces travaux de thèse. Mes premières pensées vont naturellement à mes parents. Je réserve toutefois une pensée plus affectueuse et chaleureuse pour l'amie que j'ai découverte durant ces trois années. Je la remercie pour tous les instants précieux où elle a su m'écouter et me comprendre, m'offrir sa gentillesse, partager sa bonne humeur, …, tous ces moments qui en font aujourd'hui ma meilleure amie.

Table des matières

Remerciements

Introduction générale 1

Chapitre I : Les nouveaux enjeux industriels de la maintenance

0. Introduction 5

1. Les nouveaux enjeux industriels 5

1.1. Les enjeux socio-économiques actuels 6
1.2. Les réponses et évolutions actuelles 6
1.3. Conclusion 8

2. L'évolution de la fonction maintenance 8

2.1. Définitions et concepts généraux 9
2.2. Les différentes formes de maintenance 10
2.2.1. La maintenance corrective *10*
2.2.2. La maintenance préventive *10*
2.3. La tendance actuelle 11
2.4. La maintenance prédictive : avantages et principes 12
2.5. Conclusion 13

3. Les techniques de mesures en diagnostic 13

3.1. Analyse vibratoire 14
3.2. Analyse thermographique 15
3.3. Analyse des lubrifiants et des particules d'usure 15
3.4. Analyse des paramètres de fonctionnement du procédé 16
3.5. Conclusion 17

4. Conclusion 18

Chapitre II : Les méthodes de diagnostic : Approche par Reconnaissance de Formes floue

0. Introduction **19**

1. Les différentes méthodes de diagnostic **20**

1.1. Les méthodes symboliques **20**
1.1.1. Méthodes inductives et déductives ***21***
1.1.2. AMDEC et Arbres de défaillances ***21***
1.1.3. Conclusion sur les méthodes symboliques ***22***
1.2. Les méthodes internes **22**
1.2.1. Génération de résidus ***23***
1.2.2. Décision ***24***
1.2.3. Conclusion sur les méthodes internes ***24***
1.3. Les méthodes externes **25**
1.3.1. Les principes de la RdF ***25***
1.3.2. Application au diagnostic ***26***
1.3.3. Conclusion sur les méthodes externes ***27***
1.4. Conclusion sur les méthodes de diagnostic **27**

2. Le diagnostic par l'approche RdF **28**

2.1. Phase d'analyse **29**
2.1.1. Détermination de l'espace de représentation ***29***
2.1.1.1. Pré-traitement des données *29*
2.1.1.2. Définition d'un vecteur observation *30*
2.1.1.3. Constitution d'une base d'apprentissage *30*
2.1.2. Détermination de l'espace de décision ***31***
2.2. Phase d'exploitation **33**
2.3. Conclusion sur le diagnostic par l'approche RdF **34**

3. Le diagnostic par l'approche RdF floue **35**

3.1. La logique floue **35**
3.2. Phase d'analyse **37**
3.2.1. Détermination de l'espace de décision flou ***37***
3.2.2. Représentativité des modes de fonctionnement ***38***
3.2.2.1. Gabarit des fonctions d'appartenance *38*
3.2.2.2. Approche monoprototype *39*
3.2.2.3. Approche multiprototype *39*
3.3. Phase d'exploitation **40**
3.3.1. Règle du maximum d'appartenance ***40***
3.3.2. Règle du rapport d'appartenance ***41***
3.4. Systèmes adaptatifs de RdF pour le diagnostic **42**
3.5. Conclusion sur le diagnostic par l'approche RdF floue **44**

4. Conclusion **44**

Chapitre III : Systèmes adaptatifs de RdF: Classification non supervisée, Classification dynamique de données évolutives

0. Introduction **46**

1. La classification non supervisée **47**

1.1. Algorithmes classiques de coalescence floue **48**
1.1.1. Algorithme des C-Moyennes floues ***49***
1.1.2. Algorithme possibiliste des C-moyennes ***50***
1.1.3. Algorithme FCMGK de Gustafson et Kessel ***52***
1.1.4. Validation d'une partition de données ***54***
1.2. Algorithme de coalescence floue sans connaissance *a priori* de K **55**
1.2.1. Algorithme Robust C-Prototypes ***55***
1.2.2. Algorithme Unconstrained Robust C-Prototypes ***58***
1.2.3. Algorithme Unsupervised Robust C-Prototypes ***58***
1.3. Algorithme de coalescence floue pour les classes de forme complexe **60**
1.3.1. Algorithme Unsupervised Graph Clustering ***60***
1.3.2. Algorithme Unsupervised Fuzzy Graph Clustering ***62***
1.4. Conclusion sur la classification non supervisée **64**

2. Situations d'adaptation d'un système de RdF **65**

2.1. Saut d'observations **65**
2.2. Evolution progressive des observations **66**
2.3. Evolution des classes **67**
2.4. Conclusion sur les situations d'adaptation **68**

3. La classification dynamique de données évolutives **68**

3.1. Le réseau de neurones Fuzzy Min-Max Clustering **71**
3.1.1. Architecture du réseau FMMC ***71***
3.1.2. Définition d'un prototype hypercubique flou ***72***
3.1.3. Processus d'apprentissage ***72***
3.2. Le réseau de neurones Cluster Detection and Labeling **75**
3.2.1. Architecture du réseau CDL ***75***
3.2.2. Définition d'un prototype ***76***
3.2.3. Processus d'apprentissage ***77***
3.3. Conclusion sur la classification dynamique de données évolutives **80**

4. Conclusion **80**

Chapitre IV : Réseau de neurones auto-adaptatif pour la classification dynamique de données évolutives : AUDyC

0. Introduction **82**

1. Définitions et principes **84**

2. Architecture auto-adaptative et mode d'apprentissage séquentiel **86**

2.1. Architecture auto-adaptative **86**
2.2. Mode d'apprentissage séquentiel **88**
2.2.1. Processus d'apprentissage cyclique ***90***
2.2.2. 1ère phase : Classification ***91***
2.2.3. 2ème phase : Fusion ***100***
2.2.4. 3ème phase : Evaluation ***101***
2.2.5. Bouclage des cycles d'apprentissage ***103***
2.3. Validation des capacités d'auto-adaptation de l'architecture **104**
2.3.1. Utilisation de l'algorithme neuronal AUDyC ***105***
2.3.2. Influence de l'ordre de présentation des données ***106***
2.3.3. Influence de la présence de bruit sur les données ***108***
2.3.4. Comparaison avec les algorithmes CDL et FMMC ***108***
2.4. Conclusion **109**

3. Architecture auto-adaptative et mode d'apprentissage continu **110**

3.1. Mode d'apprentissage continu **111**
3.2. Définition dynamique des prototypes et des classes **113**
3.2.1. Nouvelles règles d'adaptation des prototypes ***114***
3.2.2. Nouvelles relations prototypes-classes ***116***
3.3. Modélisation dynamique de données évolutives **116**
3.3.1. Etape n°1 : Initialisation ***117***
3.3.2. Etape n°2 : Apprentissage en ligne ***118***
3.3.2.1. Influence de l'horizon N_P des prototypes *119*
3.3.2.2. Influence de la largeur N_{fen} des fenêtres de données *120*
3.3.2.3. Création et fusion des classes *121*
3.4. Conclusion **123**

4. Conclusion **124**

Chapitre V : Système adaptatif de supervision de systèmes industriels. Application à une presse hydraulique

0. Introduction **125**

1. Proposition d'un système adaptatif de supervision **126**

1.1. Module de modélisation **128**
1.2. Module de surveillance **130**
1.2.1. Définition d'un « classifieur de tendances » ***130***
1.2.2. Détection de dérives « rapides » ***131***
1.2.3. Détection de dérives « lentes » ***132***
1.3. Module de diagnostic **133**
1.3.1. Identification du mode de fonctionnement actuel du procédé ***134***
1.3.2. Identification du mode de fonctionnement vers lequel évolue le procédé ***135***
1.3.3. Caractérisation d'un nouveau mode de fonctionnement ***136***
1.4. Conclusion **136**

2. Présentation de l'installation de pressage hydraulique **137**

2.1. Description générale **137**
2.1.1. Description structurelle ***137***
2.1.2. Description fonctionnelle ***138***
2.2. Exploitation sur le site de production **139**
2.2.1. Contexte d'utilisation ***139***
2.2.2. Analyse des défaillances ***141***
2.3. Supervision des systèmes hydrauliques **142**
2.3.1. En dehors de l'entreprise ***142***
2.3.2. Au sein de l'entreprise ***143***
2.4. Conclusion **143**

3. Supervision du pressage hydraulique **143**

3.1. L'installation expérimentale et son instrumentation **144**
3.2. Représentation de l'état de fonctionnement **145**
3.2.1. A partir des données de fabrication ***146***
3.2.2. A partir de séquences de test ***149***
3.2.3. Exploitation des deux espaces de représentation ***152***
3.3. Résultats de supervision du pressage hydraulique **154**
3.3.1. Caractérisation de la phase de mise en régime ***154***
3.3.2. Détection et diagnostic au niveau de l'accumulateur ***155***
3.3.2.1. Défaillance brutale de l'accumulateur *155*
3.3.2.2. Dégradation progressive de l'accumulateur *157*
3.3.3. Détection et diagnostic au niveau du servodistributeur ***158***
3.4. Conclusion **159**

4. Conclusion **160**

Conclusion générale **162**

Bibliographie **166**

Publications liées à la réalisation des travaux de thèse **175**

Annexe 1 : Démonstration des relations itératives d'adaptation de l'algorithme FCM **176**

Annexe 2 : Présentation de l'algorithme RCP avec la définition de la fonction ρ_k et des relations itératives d'adaptation **178**

Annexe 3 : Démonstration des relations d'adaptation des prototypes pour le réseau AUDyC **184**

Introduction générale

A l'heure actuelle, la plupart des entreprises ayant une activité de production sont fortement soumises à la concurrence du marché. Pour conforter leur position dans leur domaine d'activités, elles doivent satisfaire au mieux les ***attentes du client*** en terme de ***délais***, de ***qualité des produits*** et de ***prix***. De plus, pour accroître leur rentabilité, elles exploitent de plus en plus des ***systèmes de production de complexité croissante***. Pour satisfaire les demandes du client et réaliser des ***gains de productivité***, l'amélioration de la ***maîtrise des outils de production*** est donc devenue un ***enjeu capital*** pour la survie et la pérennité des entreprises. Ainsi, il est devenu indispensable d'assurer la ***disponibilité*** des biens de production, leur ***fiabilité***, et d'optimiser leurs conditions d'exploitation en vue d'une ***qualité optimale***.

Evidemment, ces nouveaux besoins ont un impact sur le ***rôle de la maintenance*** des outils de production. Dans ce sens, plutôt que d'attendre l'apparition d'une défaillance pour intervenir, il est préférable de ***suivre en permanence l'état de fonctionnement*** d'un procédé industriel afin de ***détecter au plus tôt ses dérives***. On est ainsi passé d'une ***maintenance curative*** à une ***maintenance prédictive*** pour laquelle la caractérisation de l'état de fonctionnement et des modes de fonctionnement du procédé nécessite l'utilisation d'***outils de modélisation***, de ***surveillance*** et de ***diagnostic.*** Dans cette optique, un système de diagnostic industriel peut être vu comme un système de Reconnaissance de Formes. L'***état de fonctionnement*** du procédé est représenté à l'aide d'un ***vecteur*** de caractéristiques et les ***modes de fonctionnement*** sont représentés par des ***régions*** particulières de l'espace de représentation défini par les caractéristiques sélectionnées. Pour cela, une première phase d'***apprentissage des modes de fonctionnement*** est nécessaire. Ensuite, il convient de prévoir une ***actualisation continue des différents modèles de modes de fonctionnement***, afin de prendre en compte la méconnaissance initiale d'informations sur d'autres modes de fonctionnement ainsi que l'évolution naturelle des modes de fonctionnement identifiés.

Les travaux présentés dans cette thèse résultent d'une convention CIFRE entre le ***Laboratoire Interaction, Image & Ingénierie de la Décision*** (I3D) de l'Université de Lille 1 (FRE-CNRS-2497), l'***Ecole d'Ingénieurs du Pas-de-Calais*** (EIPC) et le ***Service Amélioration des Moyens de Production*** (AMP) de la ***Direction Recherche & Développement de la société ARC International*** (AI). Ces travaux portent sur la mise en œuvre d'une ***structure adaptative de supervision d'une presse hydraulique*** utilisée au sein d'ARC International.

Avant de définir les objectifs de l'étude, il convient de préciser le contexte industriel propre à ***ARC International*** en termes d'activités et de besoins. Cette société réalise une activité de fabrication verrière sur l'ensemble du domaine des arts de la table et se positionne en leader mondial dans ce domaine. Sa production journalière s'élève à plus de 5,5 millions d'articles et couvre une large gamme de produits (assiettes, verres, …). Pour satisfaire ce volume de

production et cette diversité d'articles, l'entreprise s'est dotée d'une ***importante capacité de production*** répartie sur plusieurs secteurs fonctionnant 24h/24h. Ainsi, en raison de son importante activité de production et des impératifs de coûts et de délais liés à cette activité, ARC International a perçu la nécessité de mettre en œuvre une politique de maintenance prédictive de ses outils de production. Pour cela, il a été décidé de travailler à l'élaboration d'un système de supervision permettant une surveillance continue du fonctionnement des installations.

Le processus de fabrication d'un article met en œuvre une infrastructure importante à commencer par le ***four*** où l'on réalise la fusion des matières premières. La capacité d'un four permet alors de fournir du verre en fusion à plusieurs lignes de fabrications par des canaux d'alimentations appelés ***feeders*** (figure 1).

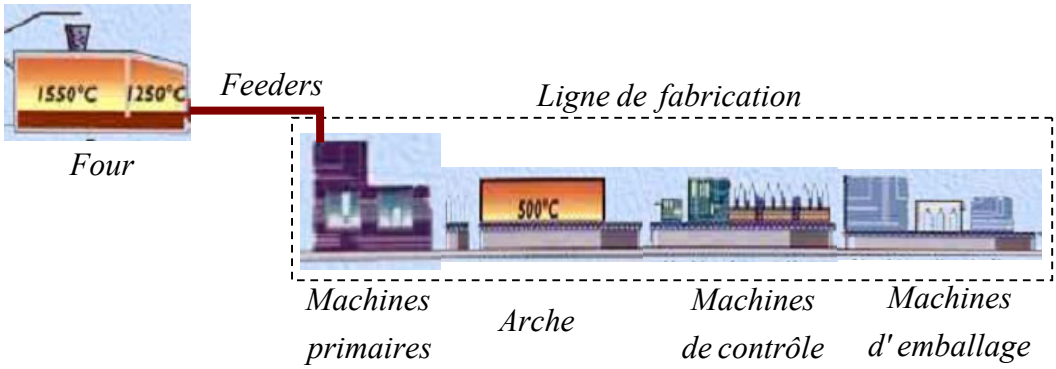


Figure 1 : Description générale d'une ligne de fabrication.

Chaque ligne dispose d'une structure spécifique pour la gamme d'articles à laquelle elle est dédiée. De façon générale, une ligne se décompose en 3 zones : le « primaire », le « parachèvement » et l'« emballage ». Le « ***primaire*** » regroupe l'ensemble des opérations de fusion (Four), de conditionnement et de canalisation du verre en fusion (Feeders), de formage des articles (Machines primaires), ainsi que les traitements thermiques tels que la recuisson ou la trempe (Arche) qui visent à donner aux articles des propriétés mécaniques ou thermiques spécifiques. Le « ***parachèvement*** » (non représenté figure 1) regroupe toutes les opérations apportant une valeur ajoutée à l'article dit « primaire » par des opérations de décor, de taille, ou encore de soudure pour la fabrication de verres à pieds. Enfin, l'« ***emballage*** » regroupe les opérations de contrôle de la qualité des articles (Machines de contrôle), d'emballage puis de palettisation (Machines d'emballage, robots). Par ailleurs, selon l'article fabriqué (dimension, forme, volume, …), différents procédés de formage sont possibles : le « ***pressé*** », le « ***pressé-soufflé*** » et le « ***centrifugé*** ».

Cette description du processus de fabrication permet de comprendre aisément que le simple dysfonctionnement d'un équipement au niveau de la ligne suffit à perturber l'ensemble du processus de fabrication. Il est donc évident que la ***non-disponibilité d'un équipement***, voire ses ***mauvaises performances*** suffisent à générer ***des coûts importants de non-qualité***.

Dans un premier temps, l'***automatisation*** des outils de production permet d'apporter une première réponse à ce problème, en facilitant la tâche de conduite de l'opérateur, au travers de la ***supervision***. Cette supervision permet à l'utilisateur d'accéder à un ensemble d'informations émanant de l'installation de production (synoptiques, interface Homme-Machine, …). De même,

pour le suivi de la qualité des articles, des ***outils de contrôles statistiques*** (SPC, …) permettent de suivre l'évolution de la qualité de production.

Dans un deuxième temps et pour une ***maîtrise totale de l'outil de production***, il est devenu nécessaire de réaliser une ***surveillance continue des installations***. C'est la raison pour laquelle, de plus en plus de fournisseurs proposent des outils, s'insérant dans la chaîne de supervision, et dédiés à la surveillance soit de composants particuliers (roulements à billes, …), soit de systèmes industriels spécifiques. Cependant, la plupart des systèmes proposés se limite à une surveillance restreinte autour d'un point de fonctionnement de l'organe ou de la machine étudiée. Or, au cours de l'exploitation, le ***point de fonctionnement d'une ligne de fabrication verrière subit de nombreuses discontinuités***, soit en raison des ***changements de fabrications***, soit en raison des variations de paramètres « extérieurs » tels que le ***conditionnement du verre*** ou les ***remplacements de moulerie***. De plus, étant donné la complexité des installations verrières, la connaissance de la totalité des modes de fonctionnement pouvant apparaître sur celles-ci est impossible. Par conséquent, cela nécessite de les apprendre au fur et à mesure de leur apparition. Evidemment, des systèmes industriels de diagnostic autorisent d'ores et déjà une prise en compte hors ligne de nouveaux modes de fonctionnement. Toutefois, cette considération hors ligne peut être catastrophique si le système dérive rapidement vers un nouveau mode de fonctionnement qualifié de dangereux.

L'objectif principal de la thèse est donc d'apporter une solution ***à la surveillance en continu des installations de production.*** Les concepts développés ont été mis en œuvre pour la surveillance d'une presse hydraulique utilisée pour le formage d'articles en verre. L'atteinte de cet objectif a nécessité la définition d'outils de représentation de l'état de fonctionnement, de modélisation des modes de fonctionnement, puis de surveillance et de diagnostic.

La première étape consiste à développer une méthode de ***modélisation dynamique des différents modes de fonctionnement*** à partir de techniques de ***classification dynamique de données évolutives.*** L'intérêt de cette démarche est d'une part de détecter et de caractériser les modes de fonctionnement inconnus initialement, et d'autre part, de suivre l'évolution temporelle des modes de fonctionnement déjà identifiés. En fait, il s'agit de réaliser un ***apprentissage non supervisé*** associé à un ***traitement en ligne*** des données recueillies sur le procédé.

La deuxième étape consiste à définir des ***outils de surveillance*** permettant de détecter toute évolution de l'état de fonctionnement. Pour cela, deux échelles de temps sont à considérer : la première pour des ***dérives « rapides »***, la seconde pour des ***dérives « lentes ».***

La dernière étape vise alors à définir un ***système adaptatif de supervision*** capable de fournir à l'utilisateur une aide à la décision, sur la base des outils de modélisation et de surveillance établis précédemment.

Ainsi, le mémoire est structuré en ***5 chapitres*** suivis d'une conclusion générale.

Le premier chapitre rappèle les principaux ***enjeux industriels actuels*** de la maîtrise et de la maintenance des outils de production. Ce chapitre se conclut par une présentation de la

maintenance prédictive et de ses avantages. Quelques techniques de mesures permettant de représenter l'état de fonctionnement d'un procédé industriel y sont également présentées.

Le deuxième chapitre aborde ***différentes approches pour l'élaboration d'un système de diagnostic industriel***. L'approche par ***Reconnaissance de Formes***, retenue dans le cadre de notre étude, est alors décrite plus amplement. Ce deuxième chapitre précise également les apports de la ***RdF floue*** sur la qualité de modélisation des modes de fonctionnement et insiste sur la nécessité de disposer d'un système adaptatif de RdF dans le cadre du diagnostic industriel.

Le troisième chapitre propose une description d'***algorithmes de classification non supervisée*** utilisés en RdF. Les premiers algorithmes décrits ne sont efficaces que si la base d'apprentissage initiale est exhaustive, ce qui est rarement le cas en diagnostic. Or, un système de diagnostic par RdF doit faire face à de nombreuses situations d'adaptation de sa connaissance des modes de fonctionnement, dont la prise en compte nécessite un ***apprentissage continu***. Pour cela, les réseaux de neurones ont su montrer leur avantage. Deux techniques neuronales issues de la littérature sont d'ailleurs décrites. Toutefois, même si celles-ci présentent des propriétés intéressantes pour la ***classification dynamique de données évolutives***, elles ne permettent pas d'envisager un apprentissage continu.

Le quatrième chapitre décrit alors ***le réseau AUDyC***, qui constitue la ***nouvelle technique neuronale auto-adaptative*** que nous avons développée ***pour la classification dynamique de données évolutives***. La particularité de l'architecture du réseau AUDyC repose sur ses ***capacités d'auto-adaptation.*** Deux versions sont proposées :

- la première considère un ***mode d'apprentissage séquentiel*** où l'architecture est actualisée de façon périodique,
- la seconde considère un ***mode d'apprentissage continu*** où l'architecture est actualisée en ligne, ce qui autorise la ***modélisation dynamique des classes***.

Le cinquième chapitre présente l'application du réseau AUDyC à l'***élaboration d'un système adaptatif de supervision*** composé de trois modules :

- un ***module de modélisation*** qui a pour rôle de caractériser en permanence les différents modes de fonctionnement du procédé,
- un ***module de surveillance*** qui a pour rôle de détecter toutes les évolutions de l'état de fonctionnement et/ou des caractéristiques du mode de bon fonctionnement,
- un ***module de diagnostic*** qui a pour rôle d'identifier le mode de fonctionnement du procédé ou celui vers lequel on se rapproche.

Les résultats de surveillance et de diagnostic, présentés à la fin de ce dernier chapitre, sont ceux obtenus sur une presse hydraulique utilisée au sein d'ARC International.

Finalement, la conclusion générale expose les perspectives industrielles qui seront données à ce travail dans le cadre de mon embauche au sein d'ARC International depuis le 1er février 2003.

Chapitre I : Les nouveaux enjeux industriels de la maintenance

0. Introduction

A l'heure actuelle, les entreprises de nombreux secteurs d'activité (automobile, sidérurgie, chimie, …) sont de plus en plus soumises à la concurrence du marché. Afin de répondre aux attentes du client en terme de quantité et qualité des produits, et de délais, nos sociétés technologiques engendrent des systèmes de production de complexité croissante et dont elles sont pleinement dépendantes. L'automatisation des processus industriels a eu pour vocation d'optimiser la productivité en implantant des commandes performantes. Malheureusement, cette complexification des systèmes a également engendré des dysfonctionnements qui viennent enrayer leur bon fonctionnement. Pour atteindre leurs objectifs de productivité et économiques, les entreprises se doivent de redéfinir leurs priorités, ce qui entraîne une mutation de certaines fonctions clés. Les tâches de maintenance ont ainsi pris une importance non négligeable dans la « bonne marche » des entreprises en garantissant le bon fonctionnement des outils de production.

Ainsi, dans une première partie, nous présentons les nouveaux enjeux des entreprises du monde industriel, tout en décrivant quelques réponses actuelles pour respecter ces nouvelles attentes. Nous décrivons alors l'évolution de la fonction maintenance qui a connu une forte mutation depuis qu'elle est considérée comme un des facteurs majeurs dans la maîtrise de l'outil de production et qui a désormais un rôle préventif dans le maintien de l'état de fonctionnement des systèmes de production. Pour assurer cette fonction, il est évident que l'on doit disposer d'un certain nombre d'informations sur le système dont on assure la maintenance. Ainsi, dans la dernière partie de ce chapitre, nous introduisons quelques techniques utilisées au sein des nouvelles formes de maintenance afin de représenter l'état de fonctionnement d'un système.

1. Les nouveaux enjeux industriels

La production en milieu industriel est caractérisée par une complexité toujours croissante des systèmes de production et des objectifs de plus en plus exigeants. Sur le plan économique, la considération des coûts de production, du rendement ou du respect des délais sont autant de facteurs qui, dans ce contexte actuel si concurrentiel, influent sur la compétitivité des entreprises. Sur le plan technique, les contraintes essentielles portent sur la diversité, la complexité et la qualité des produits, et donc sur le développement des technologies de l'informatisation et de l'automatisation. Par ailleurs, dans ce monde concurrentiel où l'on est à la recherche permanente d'une meilleure productivité et qualité à moindre coût, il ne faut pas oublier d'assurer la sûreté de fonctionnement qui a pour vocation de garantir la sécurité des biens, des personnes, mais aussi de l'environnement. Les entreprises doivent donc désormais faire face à différents enjeux socio-économiques pour assurer leur avenir.

1.1. Les enjeux socio-économiques actuels

Dans un monde industriel, où les gains de productivité représentent un souci quotidien pour les dirigeants, l'***amélioration permanente de la productivité*** de l'outil de production est un enjeu pour la survie et la pérennité des entreprises. On vise sans cesse à produire mieux, plus, à moindre coût et plus sûrement. La productivité d'une ressource de production s'exprime souvent en termes de disponibilité, de performance ou de fiabilité, et de taux de qualité. La ***disponibilité*** exprime l'aptitude d'un bien de production à être en état d'accomplir une fonction requise dans des conditions données, à un instant donné ou pendant un intervalle de temps donné. La ***performance*** ou plutôt la fiabilité exprime l'aptitude à accomplir une fonction requise, dans des conditions données, pendant un intervalle de temps donné. Et enfin, le taux de ***qualité***, exprime l'aptitude d'un matériel à fabriquer des produits dits de 1er choix.

Dans le prolongement de ce besoin permanent d'accroître la productivité, on assiste à une complexification des outils de production qui engendre une nécessité de ***sûreté de fonctionnement***. En effet, il s'agit de préserver à la fois la sécurité des opérateurs, mais aussi d'éviter la détérioration des matériels, tout en réduisant les coûts économiques qui découlent des perturbations et aléas qui surviennent en production. Dans ce sens, la complexité croissante des systèmes de production impose également une meilleure maîtrise de ceux-ci.

Pour répondre à ces enjeux, il faut d'une part maîtriser le fonctionnement de l'outil de production, et d'autre part suivre l'état de fonctionnement de celui-ci, afin de définir les actions correctives permettant de rétablir un état de fonctionnement optimal lors des situations de dysfonctionnement. Ces constats sur les besoins d'augmentation de la productivité, de maîtrise des procédés et de sûreté des biens et des personnes, ont naturellement conduit à la mise en œuvre de nouveaux outils de supervision des systèmes de production.

1.2. Les réponses et évolutions actuelles

Au cours de ces dernières années, des évolutions importantes ont marqué la conception des systèmes automatisés. Le terme « système industriel » a pris une toute autre dimension, on entend désormais par système, un ensemble d'éléments interagissant entre eux mais aussi avec l'extérieur dans le but d'assurer une mission globale qui va désormais au-delà du simple fait de produire. Dans ce sens, la tâche de supervision des systèmes industriels, qui vise à surveiller et gérer l'exécution des opérations d'un système, est devenue aujourd'hui une nécessité absolue pour les entreprises. Cette approche, qui considère les systèmes de production non seulement sous l'angle de leur commande mais également sous des aspects de supervision est couramment reprise sous la dénomination d'***Automatisation Intégrée*** [MAQ97], et se traduit par une structure hiérarchisée (figure I.1). Dans cette structure, les deux premiers niveaux correspondent respectivement aux couches traditionnelles d'***instrumentation***, et de ***régulation*** du système.

L'évolution des algorithmes de contrôle commande, avec l'apparition des Systèmes Numériques de Contrôle-Commande (SNCC) a permis la mise en place de postes de conduite où les informations sur le procédé sont accessibles à l'opérateur par des synoptiques. Toutefois, les SNCC sont essentiellement utilisés dans le cadre de la commande des processus continus et les

informations sont souvent restreintes au fonctionnement instantané du système, certains signaux étant directement comparés à des valeurs limites et traduits en terme d'alarmes.

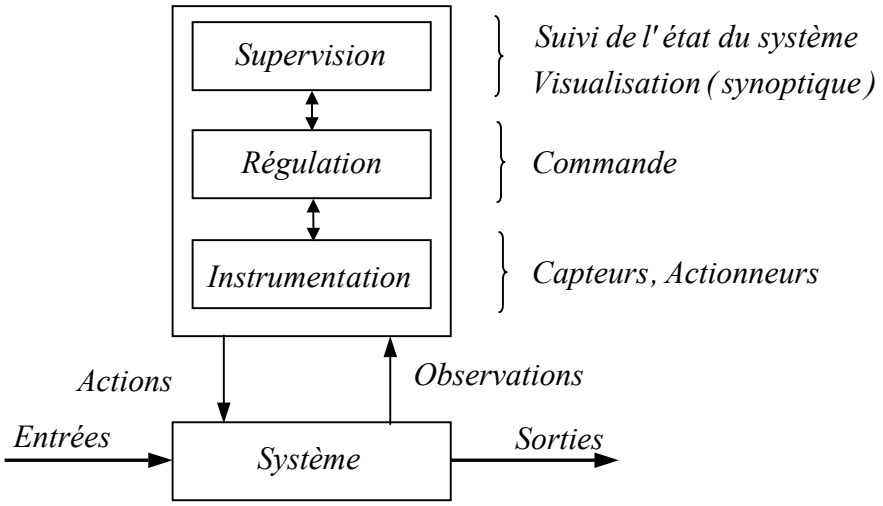


Figure I.1 : Automatisation Intégrée.

Le système SCADA (Supervisory Control And Data Acquisition), quant à lui, est un système de télécontrôle et de télécommande s'appuyant sur un ensemble d'outils informatiques interconnectés permettant d'automatiser et de surveiller à distance les opérations d'exploitation. Depuis le centre de contrôle, l'opérateur dispose alors d'informations sur le fonctionnement actuel du système (commandes, signaux de capteurs, …), mais aussi sur les rendements, les cadences, les temps d'arrêts ou encore la gestion des rebuts. Par ailleurs, au sein de ces systèmes, on dispose d'outils de contrôle tels que la maîtrise statistique des procédés (Statistical Process Control) qui vise à garantir le niveau de qualité requis et à mettre en place une traçabilité des produits, via des indices de capabilité, des cartes de contrôle…. Au milieu de toutes ces informations, l'opérateur choisit et visualise le synoptique et les informations dont il a besoin, puis après analyse, décide des actions à prendre pour maintenir la qualité de production et le bon état de fonctionnement du système : c'est ce que l'on appelle la ***supervision***.

L'introduction des SNCC ou des systèmes SCADA, auxquels on a adjoint les Interfaces Homme Machine (IHM), a considérablement amélioré le fonctionnement et la conduite des systèmes industriels. Toutefois, dans un premier temps, ces nouveaux systèmes se sont limités à un « simple partage » d'informations entre l'opérateur et le système de production. La demande croissante de rentabilité, de fiabilité, de respect des délais ou encore de sûreté de fonctionnement a alors justifié l'intérêt grandissant porté à des méthodes encore plus avancées de conduite incluant notamment des techniques de détection de défauts et de prise de décisions, afin de mieux maîtriser les périodes de production et de réduire les temps d'indisponibilités. Ainsi, les outils classiques de supervision doivent être complétés par des outils de ***surveillance***, de ***diagnostic*** et d'***aide à la décision*** qui s'intègrent à la supervision [TRA97] (figure I.2).

La distinction entre les concepts de surveillance et de supervision est au centre de réflexions au sein du Groupement pour la Recherche en Productique [GRP00]. La supervision consiste à gérer et surveiller l'exécution d'une opération ou d'un travail accompli par un homme ou une machine, puis à proposer des actions correctives si besoin. La surveillance est une opération de

recueil en continu des signaux et commandes d'un procédé afin d'en reconstituer l'état de fonctionnement réel. Ainsi, la ***surveillance*** utilise les données provenant du système pour représenter l'état de fonctionnement puis en détecter les évolutions. Le ***diagnostic*** identifie la cause de ces évolutions, puis le module d'***aide à la décision*** propose des actions correctives.

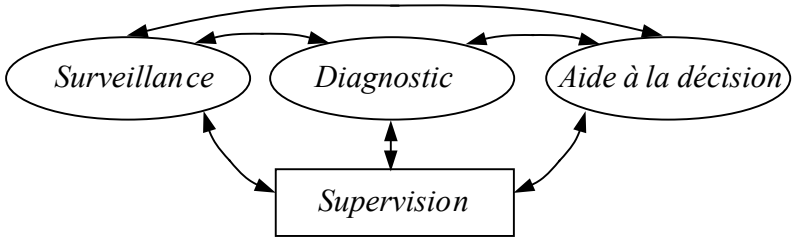


Figure I.2 : Introduction d'outils de surveillance, de diagnostic et d'aide à la décision au niveau de la supervision.

L'impératif de sûreté de fonctionnement, lié aux enjeux économiques en cas d'incidents ou de pannes, impose une maîtrise importante de ces techniques de surveillance et de diagnostic de défauts. En effet, en raison de la complexité des systèmes industriels, du principe de disponibilité maximale et de compétitivité des entreprises, la surveillance, le diagnostic et l'aide à la décision sont devenus des techniques très importantes et s'insèrent dans toute la chaîne de production d'un produit, de la conception à la maintenance.

1.3. Conclusion

Dans un monde concurrentiel dans lequel les outils de production sont de plus en plus complexes, les enjeux des entreprises s'expriment en terme de productivité, de fiabilité, de délais et de sûreté de fonctionnement. L'automatisation des systèmes industriels et le développement de systèmes de supervision ont permis d'améliorer la tâche de conduite de l'opérateur.

Pour garantir la productivité des ressources de production, la qualité des produits, et mettre en place un principe de sûreté de fonctionnement, il est devenu primordial de connaître à tout instant le mode de fonctionnement des systèmes de production. Pour cela, l'échange d'informations entre l'opérateur et le système a été amélioré au travers d'IHM plus développées et sur lesquelles l'opérateur a une image du fonctionnement de son outil de production.

Mais au-delà de ces évolutions, il faut désormais être capable de détecter un dysfonctionnement avant qu'il apparaisse concrètement en perturbant la bonne marche de l'outil de production, de le localiser, d'en identifier la cause probable, puis de proposer à l'opérateur des actions correctives. Les dysfonctionnements doivent donc être détectés le plus rapidement possible, voire anticipés puis identifiés de façon à réduire leurs conséquences néfastes tout en prenant en compte les considérations économiques, techniques, environnementales et humaines.

2. L'évolution de la fonction maintenance

La complexité croissante des matériels, l'évolution des modes de production et des comportements face aux enjeux des entreprises en terme de productivité, de qualité, de coûts et de délais ont favorisé considérablement le développement et l'essor de la fonction maintenance.

Longtemps considéré comme un poste de frais pour l'entreprise, la maintenance est désormais perçue comme un élément contribuant à favoriser la rentabilité de l'outil de production, un élément de qualité, et plus généralement comme un prolongement logique de l'activité de fabrication. La ***maintenance industrielle*** est ainsi devenue une fonction stratégique des entreprises, et comprend l'ensemble des techniques destinées à maintenir ou rétablir un bien dans un état ou des conditions données de sûreté pour accomplir une fonction requise [AFN94].

D'un point de vue organisationnel et structurel, on entend actuellement beaucoup parler de la Gestion Productive des Actifs (GPA), version enrichie des systèmes de Gestion de Maintenance Assistée par Ordinateur (GMAO). Celle-ci met l'accent sur le rendement des biens de production et leur capacité à produire des biens et des services de qualité, en d'autres termes sur l'obligation de résultats. De même, des outils comme la « Totale Productive Maintenance » (TPM) visent à sensibiliser tous les acteurs de la production à l'amélioration de la performance des ressources de production, toujours avec l'objectif d'obtenir un rendement maximum, d'optimiser les coûts d'exploitation et d'augmenter la maîtrise des outils de production. Toutes ces approches visent à une meilleure réactivité via la mise en place de ***différentes formes de maintenance***. Avant de présenter les différentes typologies de maintenance, il est toutefois utile de préciser quelques définitions et concepts généraux sur la terminologie utilisée.

2.1. Définitions et concepts généraux

La difficulté majeure rencontrée lors de la description des concepts et de la terminologie utilisée dans le monde de la maintenance provient du fait que l'on peut aborder ce problème de différentes manières selon l'origine et la formation des intervenants. De plus, les différences sont très subtiles et subjectives [ZWI95] et ces définitions peuvent différer d'une entreprise à l'autre.

Une ***défaillance*** est l'altération ou la cessation de l'aptitude d'un ensemble à accomplir sa ou ses fonctions requise(s) avec les performances définies dans les spécifications techniques.

Une ***dégradation*** d'un procédé caractérise alors le processus qui amène à un état défaillant du procédé.

Un ***défaut*** se définit comme une anomalie du comportement d'un système sans forcément remettre en cause sa fonction.

Une ***panne*** caractérise l'inaptitude d'un dispositif à accomplir une fonction requise. Un système est toutefois généralement considéré en panne dès l'apparition d'une défaillance.

La distinction entre ces définitions est établie entre les aspects comportementaux et fonctionnels [PLO98]. Ainsi, un défaut (comportement) n'entraîne pas forcément une défaillance (fonctionnelle) c'est-à-dire une impossibilité pour le procédé à accomplir sa tâche. Le défaut n'induit pas nécessairement une défaillance mais il en est la cause, et c'est donc bien la caractérisation de ces défauts qui nous intéresse ici afin de prévenir toute défaillance. Ainsi, une panne résulte toujours d'une ou plusieurs défaillance(s) qui elles-mêmes résultent d'un ou plusieurs défaut(s). Enfin, on utilise aussi le terme plus générique d'***anomalie*** pour évoquer une particularité non conforme à une référence comportementale ou fonctionnelle. Les défauts, défaillances et pannes sont des anomalies.

2.2. Les différentes formes de maintenance

D'après la norme AFNOR-NF-X60-010 [AFN94], la maintenance industrielle peut se décliner sous différentes formes selon les situations.

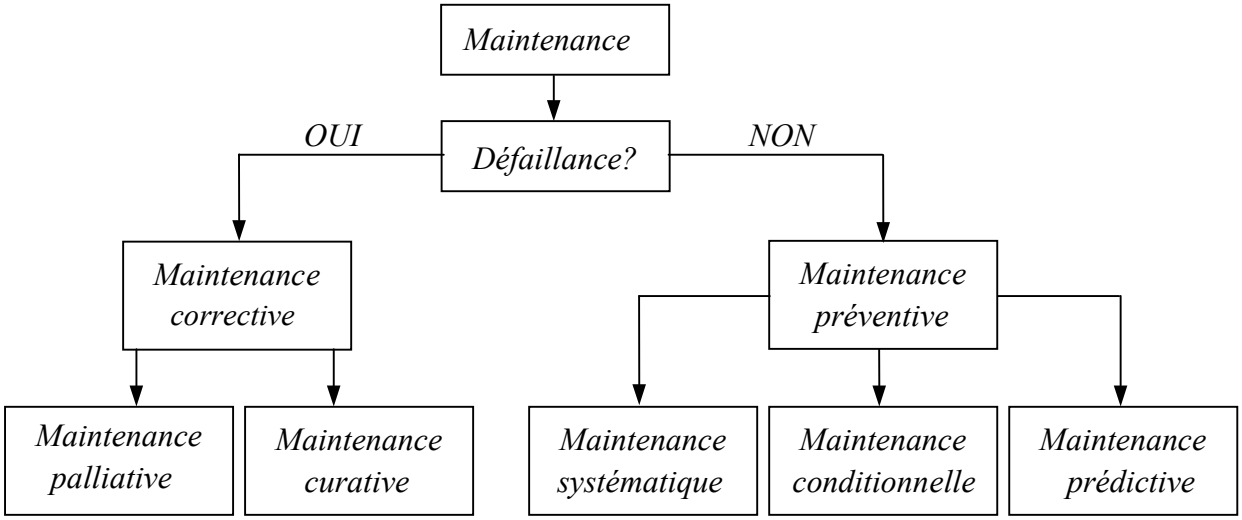


Figure I.3 : Les différentes formes de maintenance [AFN94].

Les différentes formes de maintenance sont réparties en deux catégories, figure I.3, selon la présence ou non d'une défaillance au moment considéré : on parle de ***maintenance corrective*** si une défaillance est *a priori* présente et de ***maintenance préventive*** sinon.

2.2.1. La maintenance corrective

La maintenance corrective est souvent perçue comme la forme primaire par excellence de la maintenance car l'intervention a lieu « en urgence » une fois la défaillance survenue. La logique de cette politique de maintenance est assez simple : lorsqu'une machine est défectueuse, il faut la réparer, ce qui sous-entend que si elle fonctionne, on n'y « touche » pas. Sur ce principe, la maintenance corrective regroupe l'ensemble des activités réalisées après la défaillance de l'outil de production. Cette politique regroupe une part importante des opérations de maintenance au cours desquelles le technicien de maintenance joue un rôle important puisque faute d'auto-diagnostic (aide à la décision), c'est lui qui établit un diagnostic et décide des actions correctives.

Sous cette forme de maintenance, on distingue généralement deux niveaux selon la nature des opérations réalisées. On parle de ***maintenance palliative*** lorsque l'intervention a un caractère provisoire dans le sens où elle nécessitera forcément une intervention ultérieure. Par opposition, une opération de ***maintenance curative*** se caractérise par la recherche des causes initiales de la défaillance et par la réalisation des opérations visant à rendre le système opérationnel et ainsi éviter toute nouvelle occurrence de cette défaillance.

2.2.2. La maintenance préventive

Par opposition à la maintenance corrective, la maintenance préventive regroupe les opérations de maintenance ayant pour objet de réduire la probabilité de défaillance de l'outil de production, opérations réalisées avant l'occurrence de toute défaillance qui viendrait entraver la production. Ce concept de maintenance est basé sur une inspection périodique de l'outil de production selon

des critères prédéterminés, afin de juger de ce bon état de fonctionnement. Parmi les techniques de maintenance préventive, on distingue trois niveaux. En ***maintenance préventive systématique***, l'entretien est réalisé selon un échéancier établi sur la base de critères d'usure tels que des heures de fonctionnement (en référence au MTTF (Mean Time To Failure) ou au MTBF (Mean Time Between Failure)) ou une quantité produite, qui permettent de déterminer des périodicités d'intervention. Les opérations d'entretien se traduisent par le remplacement systématique d'un certain nombre de composants identifiés par cet échéancier. En ***maintenance préventive conditionnelle***, le principe de périodicité des interventions est conservé, mais le remplacement des composants est conditionné par la comparaison du résultat de vérifications permettant d'évaluer le niveau de dégradation à un critère d'acceptation pré-établi. Les variables statistiques (MTBF, MTTF) utilisées en maintenance préventive systématique font donc place à des variables actualisées de l'état du système, telle que la valeur d'un jeu mécanique ou un niveau de vibrations. Enfin, la dernière forme de maintenance préventive est connue à la fois sous le nom de ***maintenance prévisionnelle*** et de ***maintenance prédictive***. Certains auteurs associent parfois à tort la dénomination de maintenance prédictive à la maintenance préventive conditionnelle, or en maintenance prédictive, le principe consiste à assurer un suivi continu, et non plus périodique, de l'état de fonctionnement de l'outil de production. L'objectif de la maintenance prédictive est alors de maîtriser au mieux les comportements passés et présents du système afin de prévoir les défaillances futures, et donc de maîtriser la globalité du processus de dégradation et de réduire les temps d'indisponibilités.

Pour la maintenance d'un outil de production, différentes stratégies de maintenance sont donc envisageables. Chaque forme de maintenance dispose toutefois de ses avantages et de ses inconvénients. Cependant, en raison de la complexification des systèmes industriels, la tendance actuelle est davantage au développement de la maintenance préventive et surtout de la maintenance prédictive [CAR01], même s'il y a toujours un compromis à établir en fonction du système surveillé, de la complexité des techniques à mettre en œuvre, et de leurs coûts.

2.3. La tendance actuelle

Dans le contexte actuel, la maintenance n'a plus aujourd'hui comme seul objectif de réparer l'outil de production, mais aussi et surtout de prévoir ses dysfonctionnements et d'en éviter les conséquences. Avec la complexité des systèmes actuels, l'activité de production est sujette à la présence d'aléas de fonctionnement, dont les conséquences sont assez variées mais quoique l'on en dise toujours néfastes pour la bonne marche de l'entreprise du point de vue économique, technique, et humain. L'arrêt ou le fonctionnement anormal de l'outil de production, qui se traduit par des temps d'indisponibilité, tout comme le non-respect des délais qui s'en suit, engendrent des coûts que les entreprises ne peuvent plus supporter.

De plus, la gestion de la maintenance vérifie chaque jour un vieil adage : les machines tombent toujours en panne au plus mauvais moment, i.e. lorsque la demande de production est à son niveau le plus élevé. Les entreprises ne peuvent plus se permettre d'attendre que la panne se produise pour y remédier mais doivent s'organiser pour procéder aux opérations préventives permettant de l'éviter ou tout au moins d'en limiter les conséquences. Ces dernières années, on

est ainsi passé d'une maintenance corrective à une maintenance préventive qui se caractérise par la définition de plans d'interventions sur l'outil de production, par le remplacement de certains composants en voie de dégradation, et par des opérations périodiques d'entretien.

L'ensemble de ces actions préventives était initialement réalisé de façon systématique sur la base d'un échéancier. Evidemment, ces opérations permettaient de réduire le taux de défaillances de l'outil de production, mais au détriment d'un alourdissement des coûts directs de maintenance puisque le remplacement systématique signifie que l'on change des composants qui auraient très bien pu continuer à remplir leur mission. On est alors passé à une approche conditionnelle pour laquelle le remplacement est assujetti à la comparaison d'indicateurs d'usure à des seuils pré-établis. Cette dernière évolution a essentiellement visé à optimiser les coûts de maintenance en ne cherchant à remplacer que les composants réellement dégradés, mais ne permet pas de prévoir l'apparition d'une défaillance en cours de fonctionnement en raison de son caractère ponctuel (périodique). Ainsi, ces dernières années, beaucoup de travaux ont été menés pour développer la maintenance prédictive qui vise à maîtriser la totalité du processus de dégradation en s'appuyant sur un suivi continu d'indicateurs de l'état de fonctionnement.

2.4. La maintenance prédictive : avantages et principes

Le processus de dégradation s'accélérant généralement de manière exponentielle, la détection précoce offerte par une politique de maintenance prédictive évite des coûts de réparations importants [MOB92]. Alors que les approches classiques de la maintenance reposent sur un entretien routinier et une réponse rapide face aux défaillances inattendues, la ***maintenance prédictive apporte une meilleure maîtrise de l'outil de production***. Une étude réalisée en 1988 par le Plant Performance Group, sur un échantillon de 500 entreprises (production d'électricité, l'industrie papetière, l'agroalimentaire, le textile, l'acier….) ayant adopté les premiers concepts de maintenance prédictive, a mis en évidence des améliorations sensibles de la fiabilité, de la disponibilité et des coûts de production [MOB92]. En fait, il s'agissait des premiers pas du passage de la maintenance préventive conditionnelle à la maintenance prédictive. Le développement réel de la maintenance prédictive nécessite encore beaucoup d'attention, et est encore aujourd'hui dans une phase de genèse.

Le développement d'une politique de maintenance prédictive nécessite la mise en œuvre d'un ***système de diagnostic industriel***. L'interprétation du mot diagnostic possède de nombreuses significations suivant les interlocuteurs. Les définitions utilisées ici sont celles établies au sein de la norme AFNOR-NF-X60-010 [AFN94], qui sont reprises par de nombreux auteurs [ZWI95], [BER95a] et [WEB99]. D'après l'AFNOR, l'opération de diagnostic consiste à identifier la cause probable de la (ou des) défaillance(s) à l'aide d'un raisonnement logique fondé sur un ensemble d'observations provenant d'une inspection, d'un contrôle ou d'un test. Il s'agit donc de travailler sur les relations de causalité liant les effets (symptômes observés sur le système) et les causes (défauts du système). L'objectif d'un système de diagnostic est alors de rendre compte de l'apparition d'un défaut le plus rapidement possible (c'est-à-dire avant qu'il n'entraîne des dommages importants au travers de défaillances). Les trois fonctions du diagnostic peuvent alors être décrites de la manière suivante :

- la ***détection*** : déterminer la présence ou non d'un défaut affectant le procédé en se basant sur l'analyse des effets sur le système (symptômes),
- la ***localisation*** : déterminer le type de défaut affectant le procédé en donnant des indications relatives à l'élément en défaut,
- l'***identification*** : déterminer exactement la cause de ses symptômes en identifiant la nature du défaut.

Sur ce principe, il est parfois difficile de distinguer les étapes de localisation et d'identification selon le niveau de décomposition donné aux modes de fonctionnement du système. A titre d'exemple, lorsque le voyant de température d'eau d'un véhicule s'allume, cela signifie qu'il y a eu détection d'un défaut, le symptôme étant la montée en température de l'eau. La présence d'un défaut étant établie, la fonction de localisation peut être la constatation que la pompe à eau ne tourne pas. L'identification du défaut se conclut par exemple par le fait de s'apercevoir que la courroie qui relie la pompe au moteur est cassée.

2.5. Conclusion

La fonction maintenance au sein des entreprises est en perpétuelle évolution. Pour répondre aux enjeux de productivité et de sûreté de fonctionnement, on se doit d'améliorer la disponibilité et la fiabilité des installations. La mise en œuvre d'une politique de maintenance prédictive se traduit par le développement d'un système de diagnostic capable, de détecter toute déviation de fonctionnement et / ou de comportement du procédé surveillé, de localiser le dysfonctionnement, puis d'identifier sa cause. Pour cela, le système de diagnostic doit disposer d'une connaissance sur l'état de fonctionnement du procédé surveillé. La mise en place d'une telle démarche nécessite d'acquérir des informations relatives à l'état de fonctionnement du procédé. A cet effet, différentes techniques d'acquisitions de données peuvent être mises en œuvre.

3. Les techniques de mesures en diagnostic

La mise en œuvre des nouvelles formes de maintenance nécessite l'utilisation de techniques de mesures permettant d'apprécier l'état de fonctionnement d'un système. Cette « image » de l'état de fonctionnement du procédé est souvent définie à l'aide de ***signatures***. Les descriptions suivantes sont inspirées des ouvrages très détaillés de [MOB92] et de [ZWI95].

Ces signatures doivent contenir l'information pertinente pour la détection d'une anomalie. Bien entendu, plus le système sera complexe, plus l'obtention et l'exploitation de ces signatures seront difficiles. Nombreuses sont les techniques de mesures qui peuvent être utilisées dans un programme de maintenance. Toutefois, nous ne tenons pas ici à faire une présentation exhaustive de celles-ci mais uniquement à sensibiliser le lecteur sur celles utilisées actuellement en maintenance. Dans ce sens, les principales techniques non destructives utilisées pour l'obtention des signatures sont l'***analyse vibratoire***, l'***analyse thermographique***, l'***analyse des lubrifiants et des particules d'usure***, et enfin l'***analyse des paramètres de fonctionnement*** du procédé.

3.1. Analyse vibratoire

L'analyse vibratoire s'est rapidement imposée comme la technique de mesures incontournables en maintenance, notamment des systèmes mécaniques. Le principe général consiste à utiliser l'information vibratoire émanant soit d'un équipement mécanique dynamique (machines tournantes, …) soit d'un matériel statique (structure, tuyauteries, …) afin d'en déterminer l'état. Les signaux vibratoires sont relevés à partir d'accéléromètres disposés au plus proche des endroits critiques et orientés selon l'information recherchée [CHI90].

Les techniques d'analyse vibratoire se sont montrées parfaitement fiables et précises pour la détection des comportements anormaux des systèmes de production. Toutefois, il ne faut pas oublier que les coûts de l'instrumentation nécessaire au recueil des données mais aussi d'analyses sont assez élevés, ce qui a pour effet de limiter la généralisation de ces techniques. Une première analyse consiste à déterminer l'énergie dissipée sous forme de vibrations dans une bande de fréquences donnée. La seconde consiste à travailler sur une bande de fréquences plus localisée, afin de ne considérer qu'une partie de l'outil de production. Enfin, une troisième catégorie d'analyses consiste à travailler sur une représentation spectrale de la vibration engendrée par l'outil de production ou l'un de ses composants : une ***signature vibratoire***.

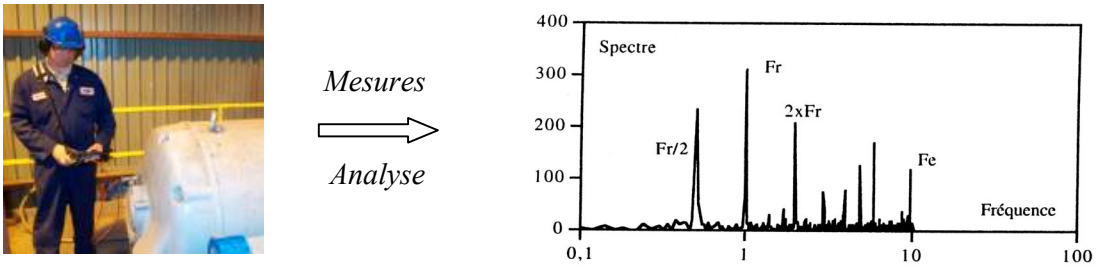


Figure I.4 : Principe de l'analyse vibratoire : a) Mesures des vibrations, b) Représentation du spectre vibratoire.

L'analyse vibratoire permet d'identifier une dégradation de l'outil de production avant qu'un incident ne se produise. Selon la qualité des relevés vibratoires et de la pertinence de l'analyse, elle permet de spécifier le ou les composant(s) qui commencent à se détériorer. [ZWI95] illustre le principe d'utilisation en considérant son application à un groupe motopompe composé d'un moteur électrique, d'un multiplicateur de vitesse et d'une pompe. L'enregistrement (figure I.4.a) des vibrations du système à des endroits spécifiques fournit un spectre vibratoire (figure I.4.b). Une analyse de celui-ci permet, par exemple, d'identifier les harmoniques de fréquences F_r (fréquence de rotation du moteur), $F_r / 2$ et $2.F_r$ (fréquences liées au nombre de dents du multiplicateur) et F_e (fréquence liée au nombre d'aubes de la pompe).

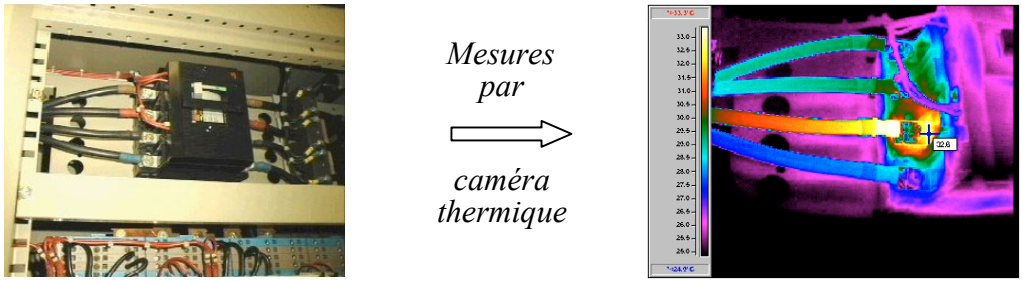
L'importance des systèmes mécaniques parmi les équipements industriels a favorisé l'essor des techniques d'analyse vibratoire dans les programmes de maintenance. Néanmoins, les coûts associés restent assez élevés pour en restreindre l'application aux équipements les plus importants de l'entreprise. Elles ont notamment été utilisées dans le cadre des travaux de thèse de [BEN99] pour la mise en place d'un système de diagnostic d'un train de laminoir à froid, en collaboration avec SOLLAC Montataire.

3.2. Analyse thermographique

L'analyse thermographique [PAJ94] fait également partie des techniques de mesures employées pour la surveillance des machines outils et des systèmes industriels. La thermographie permet la mesure de températures au moyen d'une caméra (mesure de rayonnements) et d'un thermographe (la température). L'analyse thermographie consiste à mesurer l'intensité des émissions de rayons infrarouges d'un système afin d'en déterminer les conditions opératoires.

Par conséquent, la détection de l'échauffement d'un composant (anomalie thermique telle qu'un point chaud, …) permet d'identifier un défaut, et dans certaines situations d'en localiser la cause. L'analyse thermographique peut s'appliquer à différents domaines industriels (mécaniques, électriques, électroniques, thermiques, …), mais nécessite, dans la plupart des situations, une connaissance digne du système étudié pour analyser l'***image thermique***.

Dans un programme de maintenance, on peut recourir à l'utilisation de la thermographie pour contrôler l'efficacité thermique de certains procédés, basés sur les transferts ou la conservation de la chaleur, notamment dans les industries chimiques et métallurgiques. La thermographie est ainsi utilisée pour la détection de pertes de chaleur ou de fuites sur des conduites ou sur des parois de fours calorifugés. Au-delà de ces applications de la thermographie à la maintenance des systèmes thermiques, il en existe des applications dans les domaines électriques et mécaniques.



a) *b)*

Figure I.5 : Principe de l'analyse thermographique : a) Disjoncteur électrique, b) Visualisation d'un défaut de connexion sur une image thermique.

Pour les systèmes électriques, la figure I.5 illustre la détection et la localisation d'un défaut à l'intérieur d'une armoire électrique. La partie a) représente l'image d'un disjoncteur vue par l'homme et sur laquelle aucune anomalie n'est perceptible, alors que la partie b) correspond à l'image obtenue par une caméra thermique, image sur laquelle une connexion défaillante est mise en évidence par l'apparition d'un point chaud. Pour les systèmes mécaniques, la thermographie a l'avantage de donner une vision rapide des effets d'un défaut, notamment des défauts d'engrenages, sans forcément en identifier la cause.

3.3. Analyse des lubrifiants et des particules d'usure

L'ensemble des techniques d'analyse des lubrifiants et d'analyse des particules d'usure est souvent repris sous la dénomination de ***tribologie*** [MOB92]. Ainsi, les analyses d'échantillons d'un fluide lubrifiant permettent de déterminer les caractéristiques physico-chimiques du lubrifiant et d'identifier de manière précoce une usure des éléments mécaniques en contact. Les

techniques d'analyse permettent de mettre en évidence deux facteurs prépondérants dans la qualité d'un lubrifiant : sa dégradation et sa contamination, voir figure I.6.

Echantillon

Analyse

	Dégradation	Contamination
Aspect	oui	oui
Viscosité	oui	
Oxydation	oui	
Eau en supension		oui
Insolubles		oui
Comptage de particules		oui
Ferrographie		oui

a) *b)*

Figure I.6 : Principe de la tribologie : a) Prélèvement d'un échantillon, b) Bilan de dégradation et de contamination du lubrifiant.

La ***dégradation*** d'un lubrifiant est généralement due à des actions combinées de l'oxygène de l'air et des températures élevées. L'analyse de la dégradation de l'huile s'intéresse à la quantification de ses propriétés caractéristiques telles que sa viscosité ou son acidité. Le suivi des propriétés du lubrifiant est essentiellement utile pour ajuster les périodes de renouvellement ou d'appoint sans pour autant permettre la prédiction d'usure des pièces mécaniques.

La ***contamination*** du lubrifiant provient quant à elle soit de particules d'usure des pièces en contact, soit d'une pollution par des particules solides en provenance de l'extérieur, ou encore par de l'eau. La présence de particules au sein du lubrifiant est souvent la conséquence d'usures anormales de certains composants du système. Les techniques permettant de mesurer le niveau de contamination d'un lubrifiant sont multiples. Nous ne citerons en exemple que la technique classique du comptage de particules qui constitue souvent la première analyse. L'avantage de l'étude de la contamination des huiles émane du fait qu'elle permet de déceler et de localiser l'usure prématurée d'un composant d'une installation à partir d'une analyse approfondie sur l'origine des particules d'usure présentes dans le lubrifiant. Cette analyse s'effectue par l'étude de la composition des particules, mais aussi de leur forme et de leur taille.

Toutefois, l'étape préalable à l'étude de la dégradation et/ou de la contamination du lubrifiant consiste à prélever un échantillon de lubrifiant. L'acquisition d'échantillons « valides » constitue souvent une limite à ces analyses. En effet, cette opération doit être effectuée avec soin, de façon à obtenir des échantillons très représentatifs de l'état du lubrifiant qui circule dans l'installation. Ainsi, même si toutefois on commence à voir apparaître des procédés d'analyse des lubrifiants utilisables en ligne, la méthode et la fréquence des prélèvements constituent toujours une des difficultés dans la mise en œuvre de ces analyses.

3.4. Analyse des paramètres de fonctionnement du procédé

Les techniques d'évaluation de l'état de fonctionnement d'un outil de production énumérées ci-dessus sont des techniques très spécifiques suivant l'équipement. A coté de celles-ci, il est possible d'analyser un ensemble de paramètres, directement dépendant de l'état de fonctionnement du procédé, et que l'on peut répertorier en deux catégories. Dans ce sens, il est important de noter que de nombreuses entreprises ne considèrent pas le ***rendement*** d'un outil de

production comme partie intégrante de la responsabilité de la fonction maintenance. Or, l'inefficacité est souvent le facteur limitant le plus dramatique dans une entreprise. Ainsi, des grandeurs comme le rendement constituent des indicateurs généraux sur l'état de fonctionnement d'un procédé. Bien entendu, une baisse de rendement ne peut pas être uniquement affectée à une défaillance, elle peut aussi être la conséquence d'un défaut de maîtrise. Toutefois, l'essentiel est de noter que son impact négatif sur la productivité et la rentabilité est souvent plus important que le coût direct des opérations de maintenance. Par ailleurs, l'indicateur de rendement ne peut être vu que de manière globale en permettant de détecter un dysfonctionnement, mais sans permettre de localiser et d'identifier la cause.

Au-delà de ces informations générales qui font parties des données dites de « gestion », tout système de production agit directement sur un ensemble de variables propres à son principe de fonctionnement : les grandeurs physiques. En effet, sur la majorité des procédés industriels, on dispose d'un ensemble de mesures et d'informations, notamment celles nécessaires à la commande du processus. Selon les procédés, des mesures de ***tensions***, ***pressions***, ***débits***, ***températures…*** constituent de bons indicateurs quant à l'évolution de l'état de fonctionnement d'un système dans le temps. De nombreux indicateurs pertinents peuvent alors être extraits de ces signaux par analyses temporelles ou fréquentielles. De plus, ces informations ont l'avantage indéniable d'être plus facilement interprétables et exploitables en terme de détection de dysfonctionnement, mais aussi et surtout de localisation et d'identification de la cause. Par ailleurs, quand ces informations sont nécessaires à la conduite du procédé, elles sont *a fortiori* présentes et sont donc accessibles à moindre coût.

Afin d'illustrer les possibilités d'exploitation de paramètres émanant du procédé, un exemple de développement d'un système de diagnostic utilisant des grandeurs physiques est présenté dans les travaux de [DEV99] pour la surveillance d'un processus d'injection plastique. Ce dernier représente l'état de fonctionnement au cours d'un cycle d'injection à l'aide notamment des mesures de pression d'injection, d'alimentation, de maintien, mais aussi des températures. De même, [GAD95] exploite les signaux provenant de capteurs de niveaux, de débits pour définir un système de surveillance et diagnostic appliqué à une installation hydraulique.

3.5. Conclusion

Dans cette partie, nous venons de présenter de manière non exhaustive quelques-unes des techniques de mesures permettant d'appréhender l'état de fonctionnement d'un système, et utilisées dans la plupart des programmes de maintenance préventive. L'objectif de ces techniques est de mettre en place des signatures caractéristiques de l'état de fonctionnement d'un système à partir de mesures réalisées sur celui-ci. L'utilisation de ces techniques en maintenance engendre des coûts de mise en œuvre qui ne sont pas toujours négligeables vis-à-vis des gains escomptés. Les techniques d'analyse vibratoire, de thermographie ou encore d'analyse des lubrifiants sont essentiellement utilisées de manière périodique sur des équipements pour lesquels les coûts imputés à un manque de fiabilité (perte de production, …) sont conséquents.

Les indicateurs obtenus par analyse des paramètres de fonctionnement du procédé ont l'avantage d'être plus facilement accessibles et à moindre coût lorsqu'ils font partie des

grandeurs nécessaires à la conduite du procédé. De plus, ces grandeurs sont généralement mesurables en ligne, ce qui facilite le suivi continu de l'état de fonctionnement du procédé.

4. Conclusion

Dans le monde industriel actuel, les entreprises sont soumises à des contraintes de plus en plus sévères pour assurer leur pérennité. Elles se doivent de redéfinir leurs priorités et donc leurs enjeux. Les enjeux sont nombreux mais se résument à la recherche d'une plus grande productivité de l'outil de production, ce qui se traduit par une meilleure disponibilité, fiabilité des équipements et qualité de la production, tout en assurant les besoins en sûreté de fonctionnement. Dans cette optique, les entreprises sont amenées à mettre en œuvre de plus en plus d'outils de contrôle, que ce soit au niveau de l'outil de production ou des produits (contrôles statistiques SPC). Mais plutôt que de favoriser le contrôle final des produits, il est toujours plus intéressant d'améliorer la maîtrise de l'outil de production et donc de travailler en amont. Ainsi, on assiste à une demande de nouveaux outils de supervision qui intègrent des modules de surveillance, diagnostic et d'aide à la décision : les systèmes de diagnostic industriel.

Dans ce sens, ces dernières années, la fonction de maintenance a vécu de nombreuses mutations. Aujourd'hui, l'objectif est d'être capable de détecter tout défaut d'un système ou de l'un de ses composants avant l'apparition d'une défaillance et/ou d'une panne afin d'en éviter au maximum les conséquences néfastes. On est ainsi passé d'une politique de maintenance corrective où on attendait que la panne survienne pour intervenir, à une maintenance prédictive où l'on cherche à analyser toute évolution jugée anormale du comportement d'un système pour une meilleure réactivité. Le principe de ces nouvelles formes de maintenance a engendré des besoins en outils de mesures. Pour détecter un défaut ou une dégradation et en analyser l'état d'avancement, il faut disposer d'une représentation de l'état de fonctionnement du système, ce qui se fait à l'aide de signatures spécifiques. Parmi les techniques les plus utilisées au sein des programmes de maintenance prédictive, on retrouve l'analyse vibratoire, la thermographie, l'analyse des lubrifiants, et l'analyse des paramètres de fonctionnement du procédé. Dans le cadre d'une surveillance permanente d'un procédé, la mise en œuvre des trois premières est assez coûteuse, l'analyse des paramètres du procédé a l'avantage d'offrir un ensemble de grandeurs plus facilement accessibles, de manière continue et à moindre coût. Toutefois, la maîtrise et la supervision d'un équipement ne se résument pas à ces relevés de signatures, il faut être capable de les analyser, d'en extraire les tendances, et d'identifier leurs causes, ….

Le développement de systèmes de diagnostic industriel est donc devenu indispensable et leur capacité se doit de suivre la complexité de nos systèmes. Tout ceci explique la grande diversité des méthodes de diagnostic industriel que l'on voit naître actuellement, et qui proposent des solutions pour la détection de défauts, mais aussi pour la localisation et l'identification des causes. Ainsi, tout au long du chapitre suivant, nous décrivons les différentes typologies de méthodes de diagnostic existantes.

***Chapitre II* : Les méthodes de diagnostic : Approche par Reconnaissance de Formes floue**

0. Introduction

Du fait de l'évolution permanente de la complexité des systèmes industriels, il existe des situations dans lesquelles l'être humain se trouve, en tant qu'utilisateur, face à des systèmes qui ne fonctionnent pas de la façon attendue. Selon l'ampleur et la nature de ces anomalies ou dysfonctionnements, les conséquences peuvent être plus ou moins graves, n'affecter que le système, ou alors pire, avoir un impact direct sur la sécurité du personnel. L'opérateur est de plus en plus dépendant des systèmes sur lesquels il travaille, ce qui nécessite de les maîtriser et ainsi de pouvoir leur faire confiance. Comme nous l'avons vu dans le chapitre précédent, l'automatisation des procédés a favorisé le développement des Interfaces Homme-Machine qui toutefois se limitent principalement aux situations de bon fonctionnement du procédé. Pourtant, dans un monde industriel où les procédés deviennent de plus en plus complexes et les lois du marché de plus en plus draconiennes, la ***garantie d'un bon fonctionnement*** des procédés et surtout la ***diminution des temps d'indisponibilité*** sont devenus des enjeux capitaux pour les industriels (chapitre 1).

L'amélioration des systèmes de ***supervision*** des machines de production et l'intégration des nouveaux outils de surveillance, de diagnostic et d'aide à la décision sont ainsi devenues des besoins. Dans ce sens, nous sommes passé dans une phase où la fonction maintenance connaît des évolutions notables pour atteindre le principe d'une maintenance dite prédictive. L'enjeu de la ***maintenance prédictive*** est de suivre en continu l'évolution de l'état de fonctionnement, et donc de ***détecter*** les défauts, de les ***localiser*** et d'***identifier*** leurs causes. La réalisation de ces trois fonctions se traduit par le développement d'un ***système de diagnostic industriel*** ayant pour mission d'informer à tout instant les opérateurs de production et de maintenance sur l'état de fonctionnement du système sur lequel ils opèrent, mais aussi de les renseigner lors de l'apparition de défauts ou dysfonctionnements, en leur proposant des actions correctives.

Le développement de systèmes de diagnostic est devenu un domaine d'activités technologiques et scientifiques en plein essor et impliqué dans de nombreux secteurs industriels. Dans l'automobile, l'apparition des nombreux calculateurs, notamment pour la gestion moteur, a rendu plus complexe la tâche du mécanicien; il n'est plus question de tendre l'oreille comme le faisaient autrefois les bons mécaniciens après des années d'expérience. [POU96] a d'ailleurs travaillé au développement d'un système de diagnostic par réseaux de neurones appliqué à des situations automobiles (défauts moteur, …) en collaboration avec la société ACTIA. Dans un tout autre secteur d'activité, [VIL98] et [BEN99] ont travaillé à la mise en œuvre de méthodes de diagnostic appliquées à des trains de laminoirs en sidérurgie.

A l'heure actuelle, il existe ***plusieurs typologies de méthodes de diagnostic*** qui ont chacune leurs spécificités. Pour cette raison, dans ce chapitre, nous dressons un état de l'art des méthodes de diagnostic. Dans une première partie, nous proposons une description des différentes

typologies de méthodes de diagnostic existantes : les ***méthodes symboliques***, les ***méthodes internes***, et les ***méthodes externes*** [ZWI95]. Dans une deuxième partie, nous décrivons de façon plus exhaustive les principes des ***méthodes basées sur l'approche Reconnaissance de Formes*** (RdF). Ces méthodes ont l'avantage de ne pas nécessiter la connaissance d'un modèle analytique du procédé contrairement aux méthodes internes. Enfin, nous verrons que l'introduction de la ***logique floue*** au sein de ces méthodes est avantageuse pour la définition des modes de fonctionnement et pour la prise en compte des phénomènes évolutifs.

1. Les différentes méthodes de diagnostic

Les méthodes de diagnostic sont nombreuses et diverses, les principales typologies sont détaillées dans cette première partie. Deux ouvrages ont contribué à la constitution de cet état de l'art : l'ouvrage très complet de [ZWI95] dédié aux méthodes de diagnostic en général, et l'ouvrage de [DUB90a] dédié aux méthodes de diagnostic par reconnaissance de formes. Cette bibliographie a été complétée par un rapport de recherche [BER95b], puis par diverses publications que nous évoquerons au fur et à mesure. Il convient de noter que cet état de l'art n'est pas exhaustif et a pour unique vocation de situer les méthodes entre elles. On distingue généralement trois catégories selon le type d'information traitée et la connaissance *a priori* sur le système. Lorsque les observations sont de type symbolique, on parle de ***méthodes symboliques*** dont fait partie l'approche par systèmes experts [FAR89]. Lorsque les observations sont numériques, on distingue deux approches : si l'on dispose d'un modèle mathématique du système, on s'oriente vers les ***méthodes internes***, sinon vers les ***méthodes externes***.

1.1. Les méthodes symboliques

Les méthodes de diagnostic présentées dans cette partie sont parfois reprises sous l'appellation de méthodes de diagnostic par modélisation fonctionnelle et matérielle [ZWI95]. Les informations utilisées sont principalement des données de haut niveau et la plupart du temps symboliques. Selon la nature et la disponibilité de ces informations, certains auteurs distinguent deux types de connaissances : de surface et profondes [BER95b].

Les ***connaissances de surface*** représentent l'ensemble des relations existantes et connues *a priori* entre les défauts et les causes, établies par des experts du système. Les premières méthodes symboliques de diagnostic ont représenté ces connaissances sous la forme de simples règles de production de type SI-ALORS, telles qu'elles ont été employées dans le système expert MYCIN pour une application de diagnostic de maladies infectieuses [SHO76].

Les ***connaissances profondes*** regroupent l'ensemble des connaissances structurelles, comportementales et fonctionnelles d'un système, constituant une description formelle du système sous forme d'arbres et de graphes non basés sur un système d'équations.

L'ensemble des méthodes symboliques de diagnostic donne une place importante à l'opérateur humain. Elles sont généralement constituées de deux éléments : une base de connaissance (ensemble d'états observés et de règles logiques) et un moteur d'inférence. La base de connaissances peut se présenter sous différentes formes. Les plus connues sont les ***arbres de***

défaillances et les ***Analyses des Modes de Défaillances, de leurs Effets, et de leurs Criticités*** (AMDEC) [POU96]. Le moteur d'inférence fonctionne, quant à lui, selon deux principes : l'***induction*** et la ***déduction*** [VEL98].

1.1.1. Méthodes inductives et déductives

Les ***méthodes inductives*** correspondent à une approche montante, dont le principe est décrit par la partie gauche de la figure II.1. Il s'agit de trouver un défaut (cause) à partir de l'observation de symptômes (effets) sur le système.

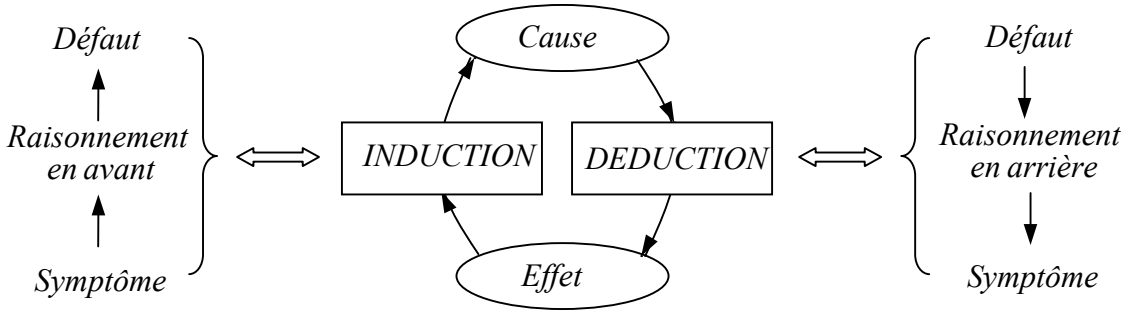


Figure II.1 : Principe des mécanismes d'induction et de déduction.

Les ***méthodes déductives*** correspondent à une approche descendante, dont le principe est décrit par la partie droite de la figure II.1. Il s'agit à partir de la connaissance *a priori* d'un défaut (cause) de déterminer tous les symptômes (effets) dont il peut être responsable. Une vérification des « effets trouvés » par rapport aux « effets possibles » permet alors de confirmer ou d'infirmer l'existence du dit défaut [ISE93].

1.1.2. AMDEC et Arbres de défaillances

Les ***AMDEC*** sont des techniques déductives et qualitatives avec lesquelles les effets des défauts sont systématiquement identifiés, elles sont très utilisées en sûreté de fonctionnement. La démarche consiste à comprendre comment et pourquoi les fonctions du système étudié risquent de ne plus être assurées correctement, en définissant ce que l'on appelle l'ensemble des modes de défaillances. Pour chaque mode de défaillance, on recherche les causes possibles de son apparition, et on définit pour chaque combinaison (cause, mode de défaillance), les effets sur le système et sur l'opérateur, leurs niveaux possibles de détection et leurs criticités sur la base de critères de gravité, de fréquence d'apparition et de probabilité de non-détection. Les résultats de l'analyse sont alors présentés sous forme d'une grille récapitulative. Dans le cadre d'un système de diagnostic industriel, les résultats de l'AMDEC sont utilisés comme outils d'identification de la cause d'une défaillance. Pour sa part, [BER95a] décrit l'AMDEC comme une méthode d'analyse critique consistant à identifier de façon inductive et systématique les risques de dysfonctionnement d'un système puis à en rechercher les causes et les conséquences.

Les ***arbres de défaillances*** correspondent aux démarches inductives : on identifie toutes les combinaisons d'événements possibles (défauts) qui entraînent la réalisation d'un événement indésirable (défaillance). Les conditions et les événements sont alors organisés sous la forme d'un arbre à plusieurs niveaux à l'aide d'opérateurs logiques. A un niveau donné, chaque événement est généré à partir de combinaisons logiques d'événements des niveaux inférieurs.

Cette procédure est réitérée jusqu'à atteindre les événements élémentaires, les fautes, qui traduisent les causes possibles de l'événement indésirable considéré. Sur le même principe que les arbres de défaillances, certains auteurs utilisent ce que l'on appelle les ***arbres de tests***. Dans ce sens, [DUF94] présente dans ses travaux de thèse une méthodologie de génération automatique d'arbres de tests intéressante pour le traitement de systèmes complexes, et qu'il a appliqué au domaine automobile au sein de la société ACTIA.

1.1.3. Conclusion sur les méthodes symboliques

L'idée des méthodes symboliques de diagnostic est basée sur la mise en œuvre d'un modèle fonctionnel ou comportemental du système étudié visant à reproduire la démarche de l'homme lorsqu'il se trouve confronté au problème de diagnostic. Deux principes de raisonnement sont envisagés selon que l'on part du défaut ou de sa cause : la déduction et l'induction. Les AMDEC et les arbres de défaillances permettent de structurer la connaissance utilisée par ces méthodes symboliques. Malheureusement, pour les systèmes industriels complexes, l'exploitation et l'analyse de la base de connaissances deviennent vite fastidieuses. Cependant, certains auteurs travaillent sur des méthodes informatisées qui permettent par exemple la génération de la base de connaissances d'un système expert à partir des données d'une AMDEC [SUB92].

Ainsi, même si ces méthodes sont parfois présentées comme des méthodes de diagnostic à part entière, en pratique elles sont essentiellement utilisées pour identifier les relations de causalité défauts / symptômes. Par ailleurs, ces méthodes ne permettent pas de suivre l'apparition d'une défaillance en raison de la nature binaire des événements considérés.

1.2. Les méthodes internes

Les méthodes internes sont souvent reprises sous la dénomination de ***méthodes à base de modèles analytiques***. Par opposition aux méthodes symboliques, les méthodes internes traitent des données numériques obtenues à partir d'un ***modèle*** du système. De façon générale, ces méthodes sont basées sur un principe de comparaison qui vise à surveiller un système en comparant les paramètres du système réel à ceux d'un modèle de celui-ci [WEB99]. La mise en place d'un modèle d'un système physique fait appel à différentes techniques d'identification et de modélisation des procédés [LJU99]. On distingue alors deux phases lors de l'utilisation de ces méthodes : la première est fondée sur l'utilisation d'un modèle analytique du système et consiste à comparer ce modèle au système réel et à engendrer des ***résidus*** sensibles aux défauts, c'est la ***génération de résidus***; la seconde vise à détecter la présence ou non d'un défaut, puis à localiser et identifier le défaut, c'est la phase de ***décision***.

Cette famille de méthodes est dérivée des techniques utilisées par les automaticiens et est souvent reprise dans la littérature sous la dénomination de FDI (Fault Detection and Isolation). De nombreux travaux de synthèse ([WIL76], [ISE84], [GER88], [FRA90], [PAT94]) et ouvrages ([BRU90], [GER98]) traitent de ce sujet et témoignent de la diversité des approches proposées. Les paragraphes suivants décrivent les principes de génération des résidus, puis de décision.

1.2.1. Génération de résidus

Les méthodes à base de modèles nécessitent une étape importante d'élaboration d'un modèle analytique du système, nécessaire à la ***génération des résidus***. Plusieurs types de modélisation sont possibles (discret, continu, linéaire, non linéaire…) [WAL94]. Nous ne rappelons ici que la distinction fondamentale entre le modèle de connaissance et le modèle de représentation.

L'écriture d'un ***modèle de connaissance*** repose sur la connaissance des lois fondamentales de la physique qui régissent le procédé. Ses paramètres sont directement liés aux éléments physiques du système, ce qui le rend approprié pour le diagnostic. Hélas, pour les systèmes complexes, sa constitution nécessite une lourde investigation.

L'élaboration d'un ***modèle de représentation*** consiste à modéliser les relations entrées-sorties du système. Les paramètres perdent alors une grande partie de leur sens physique, et sont donc plus difficilement interprétables en diagnostic. Cette modélisation est simple à mettre en œuvre et reste la seule approche pour les systèmes de plus grande complexité.

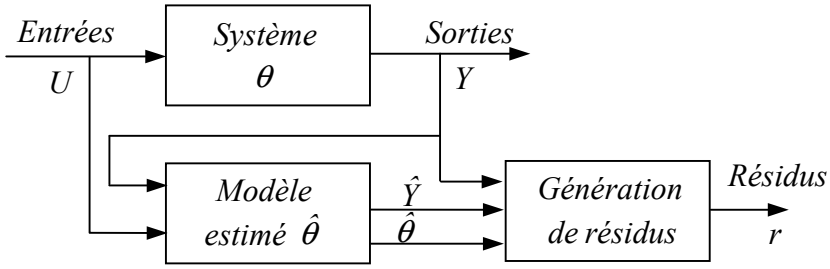


Figure II.2 : Principe de génération des résidus.

Le modèle estimé $\hat{\theta}$ du système étant élaboré [LJU99], la phase de génération des résidus (r) peut avoir lieu. La figure II.2 illustre ainsi de façon simplifiée le concept des méthodes internes de diagnostic avec notamment la phase de génération de résidus. L'objectif d'un résidu est d'être sensible aux défauts. Un résidu est donc proche d'une valeur de référence (couramment nulle) si aucun défaut n'affecte le système, et s'écarte de cette référence dès l'occurrence d'un défaut. En général, les résidus résultent de la comparaison entre des mesures effectuées sur le système et les informations fournies par le modèle. Plusieurs méthodes existent pour engendrer des résidus. Les paragraphes suivants, inspirés de l'ouvrage très complet de [GER98] présentent brièvement les trois principales méthodes :

- ***Equations de parité*** [PAT91][GER90] : Le principe repose sur la vérification de l'égalité des deux membres d'une équation, faisant uniquement intervenir des variables mesurables du système, et obtenue par un principe de redondance. Le résidu est défini par la soustraction des deux membres de l'équation. Une relation ou équation qui génère un résidu est appelée relation ou équation de parité. La mise en place des équations de parité nécessite bien entendu la connaissance de la structure et du comportement du système.

- ***Estimation paramétrique*** [ISE93] : Les résidus sont généralement engendrés par la différence entre les estimations en ligne des paramètres du modèle du procédé et les paramètres nominaux du modèle définis pour un fonctionnement normal (modèle de référence). Le pouvoir explicatif des paramètres estimés est d'autant plus grand que ceux-ci ont un sens physique. Toutefois, les conditions d'estimation des paramètres sont

contraignantes (excitation permanente), et le retour aux paramètres physiques n'est pas toujours possible. Dans ses travaux, [SCH99] propose une application de cette technique pour le développement d'un système de surveillance de machines asynchrones.

- ***Estimation d'état*** [PAT89] [FRA90] : Les résidus sont définis en considérant l'erreur de prédiction d'un observateur, ou d'un filtre de Kalman [WIL76]. Le modèle de référence du procédé est alors forcément sous la forme d'état, ce qui permet de connaître tous les états internes du procédé. L'un des avantages de cette approche de génération des résidus est la possibilité de construire un ensemble d'observateurs sensibles à un nombre restreint de défauts, afin d'assurer l'isolabilité des défauts. [GAD95] applique ces méthodes d'estimation d'état au diagnostic d'un système hydraulique.

Les différentes méthodes de générations de résidus exposées ci-dessus présentent selon certains auteurs, des équivalences, bien qu'en réalité elles se montrent souvent complémentaires. La tendance actuelle est de les faire « cohabiter » pour exploiter leur complémentarité [WEB99].

1.2.2. Décision

Les résidus ayant été générés, il faut désormais les exploiter dans le cadre d'un système de diagnostic. Ils doivent être quantifiés, c'est ce qu'on appelle la phase d'***évaluation des résidus*** ou de ***détection***. L'évaluation a pour objectif de décider ou non de la présence d'un défaut dans le système. La présence d'un défaut étant vérifiée, la phase de décision se poursuit avec les étapes de ***localisation*** et d'***identification*** du défaut.

Les méthodes de détection ou d'évaluation peuvent être établies de différentes façons : par l'utilisation d'un seuil fixe, ou par des tests d'hypothèses plus sophistiqués comme le test de somme cumulée CUSUM (dit de Page Hinkley) [BAS88], ou le test du maximum de vraisemblance généralisé GLR [WIL76] (chapitre 4 de [ZWI95]). Ces méthodes définissent un vecteur de cohérence binaire (une signature) où les résidus ayant permis la détection sont repérés. Si un seul résidu sensible à tous les défauts est suffisant pour la détection, plusieurs résidus sont nécessaires à la localisation du défaut [GK93] par analyse de tous ces résidus.

Dans ce sens, indépendamment de la méthode utilisée pour la génération des résidus, la connaissance liée à la structure choisie est généralement stockée dans une matrice binaire appelée ***matrice d'incidence.*** Cette matrice décrit les relations entre les résidus et les défauts. La comparaison du vecteur de cohérence aux différentes colonnes de la matrice d'incidence permet de localiser le défaut. Selon la structure de la matrice, on peut juger des capacités de localisation. On dit par exemple qu'une matrice d'incidence est non localisante si deux de ces colonnes sont identiques, c'est-à-dire que deux défauts ont la même signature vis-à-vis des résidus.

1.2.3. Conclusion sur les méthodes internes

Les méthodes internes de diagnostic sont fondées sur l'utilisation d'une modélisation du système étudié. Ce modèle sert généralement de référence du mode de bon fonctionnement du système. La surveillance est alors basée sur la constitution d'un vecteur de résidus, où chaque résidu représente l'écart entre des estimations faites à l'aide du modèle et les grandeurs réelles. Plusieurs méthodes existent pour cette phase de génération de résidus : les méthodes d'espace de

parité, les méthodes d'estimation d'état, les méthodes d'estimation paramétrique, …. L'étape de détection consiste à déceler une évolution de la valeur d'un des résidus. La localisation s'établit alors à l'aide d'une matrice d'incidence construite sur la base des relations connues entre les résidus et les défauts.

Enfin, la théorie du diagnostic à base de modèles est construite sur le principe qu'un modèle analytique du système est disponible, mais dans la réalité, la tâche de loin la plus délicate est justement la construction de ce modèle. De plus, celui-ci étant établi, il faut s'assurer de la sensibilité des résidus aux défauts pour éviter les risques de fausses alarmes ou de non-détection. Pour pallier cette difficulté, la section suivante est consacrée à une catégorie de méthodes ne nécessitant pas l'élaboration d'un tel modèle.

1.3. Les méthodes externes

Les méthodes de diagnostic dites « externes » s'appliquent dans les situations où la modélisation du procédé à surveiller n'est pas réalisable ou nécessite un travail très laborieux, ce qui est souvent le cas pour les systèmes industriels complexes. Contrairement aux méthodes internes, l'utilisation des méthodes externes ne nécessite pas de modèle du procédé et donc minimise la ***connaissance a priori sur celui-ci***. L'état de fonctionnement du système est déduit à partir de mesures pertinentes relevées sur le procédé via son instrumentation. Ces mesures portent couramment le nom de **« *signatures externes* »**. Un état de fonctionnement est alors représenté par un ensemble de caractéristiques extraites de ces signatures. Ceci permet de définir un mode de fonctionnement en regroupant l'ensemble des états de fonctionnement qu'il caractérise. Les systèmes de diagnostic par approche externe ont pour mission d'associer un état de fonctionnement à l'un des modes de fonctionnement du système, démarche similaire à celle des techniques de ***Reconnaissance de Formes*** (RdF). Dans le cadre de ce mémoire, les travaux ont porté sur ces méthodes en raison de leur aptitude à être appliquées aux systèmes complexes.

Ainsi, nous commençons par rappeler les principes de la reconnaissance de formes, puis nous décrivons les similarités entre les démarches du diagnostic et de la RdF qui permettent d'introduire les concepts des ***méthodes de diagnostic par reconnaissance de formes*** [DUB90a].

1.3.1. Les principes de la RdF

La reconnaissance des formes est la science de définition d'algorithmes permettant de déterminer à quelle forme un objet observé est similaire, ou en d'autres termes à quelle classe connue d'objets il peut être associé, une classe étant définie par un ensemble de formes-types. La connaissance *a priori* nécessaire à tout système de RdF est alors la définition d'une forme, d'une forme-type, et d'une classe. Dans la littérature, on distingue deux types de RdF :

- la ***RdF structurelle*** [FU74] : représentation des formes à l'aide de grammaires,
- la ***RdF statistique*** [DUB90a] : représentation numérique des formes.

Nous n'évoquerons ici que le principe des techniques de RdF statistique qui sont les plus exploitées dans le cadre du diagnostic. En RdF statistique, on définit une forme par un ensemble de D paramètres, appelés aussi caractères ou composantes.

Une ***forme*** est décrite par un vecteur, communément repris sous les termes de ***vecteur forme*** ou ***observation***, noté $X^i = \left[x_1^i \cdots x_D^i\right]^T$ dans un espace à D dimensions. Une observation est représentée par un point dans cet espace qui est connu sous le nom d'***espace de représentation.*** Dans le cas où toutes les composantes sont à valeurs réelles, considération adoptée par la suite, l'espace de représentation est noté $\Re^D$.

Une ***forme-type*** caractérise un point représentatif de l'espace de représentation et est décrite par un vecteur noté $P^j = \left[p_1^j \cdots p_D^j\right]^T$ de dimension D. Une forme-type est communément appelée ***prototype***. En raison du « bruit » sur les observations, à chaque prototype est associée une « zone géométrique » qui permet la définition des ***classes*** décrivant des régions particulières de l'espace de représentation. L'ensemble des classes $C = \left\{C^k, k = 1 \ldots K\right\}$ définit l'***espace de décision***. La définition de l'espace de décision requiert une phase d'apprentissage nécessitant un ensemble X d'observations initiales et des ***méthodes de classification***.

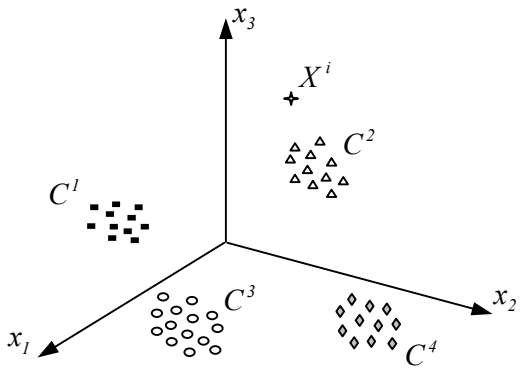


Figure II.3 : Le problème de la reconnaissance de formes : A quelle classe associer l'observation X^i ?

Le problème de la reconnaissance de formes, illustré par la figure II.3, est alors de savoir associer toute nouvelle observation $X^i \in \Re^D$ à une classe de l'espace de décision. L'association d'un vecteur forme X^i à l'une des K classes notées C^1, …, C^K désigne une opération de ***classement*** ou de ***discrimination***.

1.3.2. Application au diagnostic

L'opération de diagnostic consiste à identifier la cause probable de la (ou des) défaillance(s) affectant un système à l'aide d'un raisonnement logique fondé sur un ensemble d'observations provenant d'une inspection, d'un contrôle ou d'un test (chapitre 1). En d'autres termes, le diagnostic a pour mission d'identifier le mode de fonctionnement d'un système à partir d'observations sur celui-ci. La ressemblance entre les problèmes de diagnostic et de RdF est alors évidente si l'on considère leurs définitions respectives. C'est cette similitude qui a donné naissance aux ***méthodes de diagnostic par l'approche Reconnaissance de Formes***. Cette analogie est basée sur le postulat qu'un mode de fonctionnement peut être représenté par une classe, c'est-à-dire une région de l'espace de représentation. Dans ce sens, la figure II.4 illustre de façon schématique le principe d'élaboration d'un vecteur observation de D caractéristiques de l'état de fonctionnement du système définissant un espace de représentation.

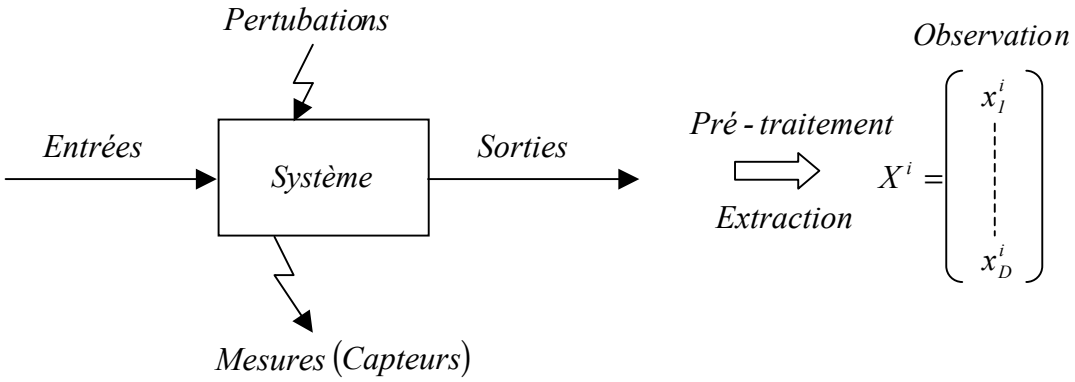


Figure II.4 : Extraction d'un vecteur observation représentatif de l'état de fonctionnement du procédé étudié.

Ces caractéristiques sont issues de traitements effectués sur les mesures acquises sur le procédé via son instrumentation. On utilise en général des méthodes issues de l'analyse de données [THI97] ou du traitement de signal pour extraire les caractéristiques pertinentes. Sur le même principe, les différentes classes identifiées par les méthodes de RdF sur la base d'un échantillon d'observations sur le procédé peuvent être assimilées aux différents modes de fonctionnement de celui-ci, ceux-ci définissant alors l'espace de décision.

Le principe des méthodes de diagnostic par les techniques de RdF consiste alors à associer toute nouvelle observation sur le système à l'un des modes de fonctionnement définis dans l'espace de décision.

1.3.3. Conclusion sur les méthodes externes

Les méthodes de diagnostic par reconnaissance de formes ont l'avantage de ne pas utiliser de modèle analytique du système et donc de minimiser la connaissance *a priori* sur celui-ci. Ces méthodes utilisent une représentation de l'état de fonctionnement établie à partir de mesures effectuées sur le système, d'où leur appellation de méthodes externes. A tout instant, l'état de fonctionnement du système est représenté par une observation définie par un ensemble de caractéristiques. Les modes de fonctionnement du système sont caractérisés par des classes définissant des régions de l'espace de représentation et constituant un espace de décision.

L'objectif de ces méthodes de diagnostic par RdF est alors d'analyser la position des observations dans l'espace de représentation et d'associer les nouvelles observations sur le système à l'un de modes de fonctionnement défini dans l'espace de décision, lui-même établi lors de la phase d'apprentissage du système de RdF. De nombreux travaux ont déjà permis de montrer l'intérêt du diagnostic par RdF [DUB94][MOU93] et notamment dans le cadre de systèmes complexes. Sur ce dernier point, [RIB02] présente une approche de diagnostic par RdF appliquée à un accélérateur à induction.

1.4. Conclusion sur les méthodes de diagnostic

L'objectif de cette première partie a été de proposer un bref état de l'art sur les principes des différentes méthodes de diagnostic couramment utilisées. Trois catégories de méthodes ont été présentées : les méthodes symboliques, les méthodes internes, et les méthodes externes.

Les ***méthodes symboliques*** constituent davantage des outils d'aide au diagnostic que des méthodes propres au développement d'un système industriel de diagnostic. En effet, il s'agit plutôt d'outils d'analyse des relations entre les causes (défauts) et les effets (symptômes) utilisables après la détection de la présence d'une défaillance sur le procédé.

Les ***méthodes internes*** de diagnostic sont très intéressantes lorsque l'on dispose d'un modèle interne du procédé ou que celui-ci est facilement accessible par modélisation. Des indicateurs de l'état de fonctionnement du procédé, nommés résidus, sont établis par comparaison entre l'état théorique du système, estimé à l'aide du modèle, et son état mesuré. Un ensemble d'outils d'analyse des résidus permet d'accomplir les tâches du diagnostic. Toutefois, l'étape la plus contraignante dans la mise en œuvre de ces méthodes est l'établissement du modèle et l'estimation de ses paramètres, qui est d'autant plus délicate pour les systèmes complexes.

Les ***méthodes de diagnostic par Reconnaissance de Formes***, par opposition aux méthodes internes, n'utilisent pas de modèle analytique du procédé. En effet, les méthodes par RdF s'appuient sur une représentation de l'état de fonctionnement d'un système à l'aide d'un ensemble de caractéristiques extraites des mesures sur le procédé. L'ensemble des caractéristiques et l'ensemble des modes de fonctionnement traduisent respectivement l'espace de représentation et l'espace de décision. L'opération de diagnostic consiste à associer toute observation sur le procédé à l'un des modes de fonctionnement.

Ainsi, le ***choix d'une méthode de diagnostic*** dépend essentiellement de la connaissance dont on dispose du procédé étudié, sans toutefois oublier les considérations techniques et économiques. Cependant, parmi les méthodes présentées, les plus appropriées pour le diagnostic des systèmes complexes sont celles basées sur l'approche par RdF puisqu'elles ont l'avantage de minimiser le besoin de connaissance *a priori*.

2. Le diagnostic par l'approche RdF

Les méthodes de diagnostic par Reconnaissance de Formes présentent quelques avantages dans le cadre des systèmes complexes. En effet, sur des problèmes tels que le nucléaire, l'industrie automobile, sidérurgique, chimique, … où la modélisation des procédés utilisés est souvent laborieuse, on privilégie l'approche par Reconnaissance de Formes [ZWI95]. Les applications de ces méthodes à des situations réelles sont de plus en plus nombreuses et présentes dans de nombreux domaines d'activité. [THO96] a ainsi proposé une méthode de diagnostic par RdF appliquée au domaine automobile, tout comme [BEN99] pour une application de diagnostic d'un laminoir.

L'élaboration d'un système de diagnostic par RdF se réalise en deux phases. La première, la ***phase d'analyse*** comprend l'analyse des données (pré traitement, …), la définition d'un ***espace de représentation***, et la constitution d'un ***espace de décision*** au sein duquel sont définis les différents modes de fonctionnement. La seconde phase est la ***phase d'exploitation*** où l'objectif est d'associer aux nouvelles observations les modes de fonctionnement.

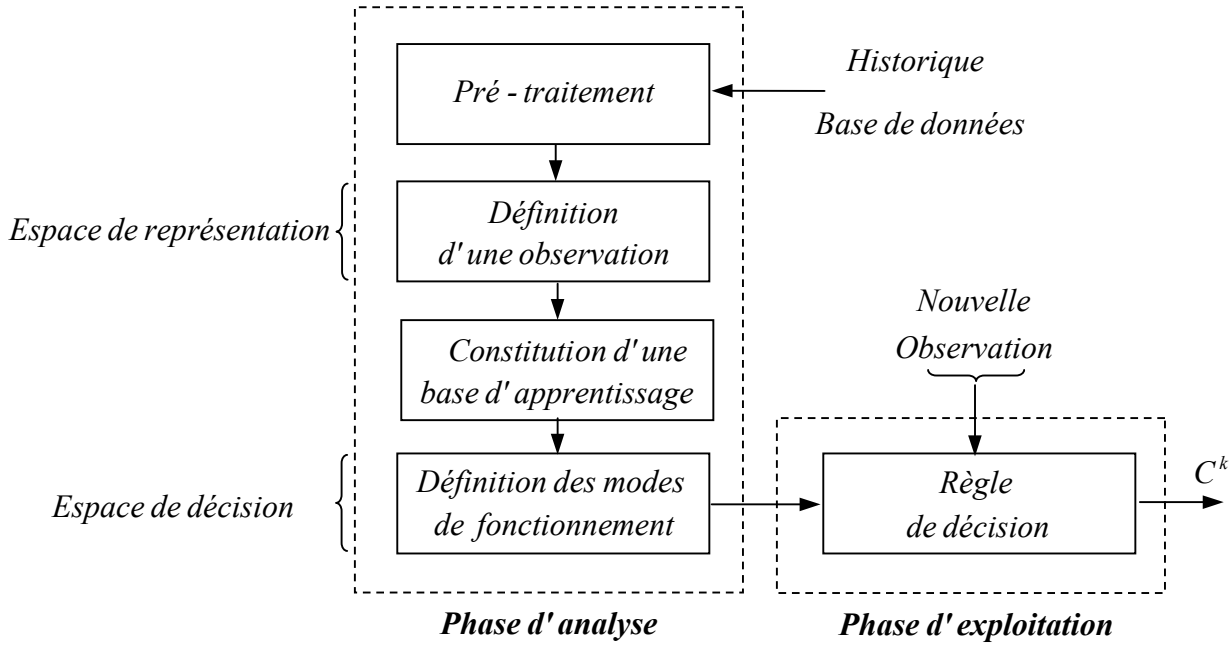


Figure II.5 : Phases d'élaboration d'un système de diagnostic par RdF.

La démarche conduisant à l'élaboration d'un système de diagnostic par RdF est illustrée par la figure II.5. Cette seconde partie est consacrée à la description de la phase d'analyse, puis de la phase d'exploitation d'un système de diagnostic par Reconnaissance de Formes.

2.1. Phase d'analyse

La phase d'analyse est une étape très importante qu'il convient de mener avec méthode et rigueur. Son objectif est de ***déterminer*** d'une part l'***espace de représentation*** $\Re^D$ au sein duquel sont décrites les observations sur le système à surveiller, et d'autre part l'***espace de décision*** C correspondant aux différents modes de fonctionnement caractérisés par des classes.

2.1.1. Détermination de l'espace de représentation

La définition de l'espace de représentation se réalise en plusieurs étapes : le pré-traitement des données contenues au sein d'un historique du procédé; la définition d'un ensemble de caractéristiques représentatives de l'état de fonctionnement du procédé et définissant les dimensions d'un espace de représentation; la constitution d'une base d'apprentissage regroupant les observations caractéristiques des différents modes de fonctionnement connus dans l'historique; et enfin une étape de sélection et d'extraction des caractéristiques permettant de réduire la dimension de l'espace de représentation.

2.1.1.1. Pré-traitement des données

Pour surveiller un système, il est nécessaire de disposer d'informations de celui-ci, généralement délivrées par des capteurs. Pour ce faire, un travail d'instrumentation du système est nécessaire. Le choix des capteurs est fonction de la pertinence des informations recherchées. Pour cela, il est courant de s'appuyer sur les historiques de la MAO (Maintenance Assistée par Ordinateur) afin d'une part de cerner les défaillances courantes et pénalisantes et d'autre part

d'identifier leurs symptômes (variations de pression, hausse de température, …). Au-delà de cette phase d'instrumentation, on établit une phase de pré-traitement faisant appel aux techniques de traitement du signal et de l'analyse de données.

En pratique, le choix des capteurs et le pré-traitement des informations délivrées ont une importance considérable pour le développement et surtout la réussite d'un système de diagnostic. Néanmoins, ces étapes restent étroitement liées au domaine d'application et il est par conséquent difficile d'établir une théorie générale (seulement une méthodologie [DUB90a]).

2.1.1.2. Définition d'un vecteur observation

La détermination du vecteur observation représentatif de l'état de fonctionnement du procédé à surveiller constitue une étape primordiale et toujours délicate dans l'élaboration d'un système de diagnostic. En effet, les caractéristiques sélectionnées sur les informations obtenues à l'issue de l'étape de pré-traitement et utilisées pour définir le vecteur observation conditionnent l'efficacité du système de diagnostic, car celles-ci définissent l'espace de représentation. Il s'agit en fait de définir à partir des données prétraitées D caractéristiques pertinentes vis-à-vis des différents modes de fonctionnement connus. Aucune règle n'existe, toutefois il faut trouver un compromis entre la dimension D du vecteur et la pertinence des caractéristiques. Très souvent, c'est l'expertise physique du procédé qui guide ce choix.

2.1.1.3. Constitution d'une base d'apprentissage

Les caractéristiques reprises dans le vecteur observation étant définies, l'étape suivante vise à mettre en place un ensemble constitué de vecteurs observations définis à partir de l'historique du procédé. Cet ensemble est alors noté $X = \{X^1 \dots X^n \dots X^N\}$ où X^n représente la $n^{ième}$ observation et constitue une ***base d'apprentissage***. Evidemment, cette base est rarement exhaustive surtout en diagnostic. Dans la pratique, les techniques actuelles prennent souvent l'hypothèse que l'on dispose d'informations sur l'ensemble des modes de fonctionnement du procédé.

A ce stade, nous disposons d'un espace de représentation $\Re^D$ des données ainsi que d'une base d'apprentissage pour notre système de diagnostic. Toutefois avant de conclure sur la détermination de l'espace de représentation, il nous semble opportun d'évoquer quelques méthodes utilisées pour réduire la dimension de celui-ci en éliminant les informations corrélées ou non représentatives tout en conservant la pertinence des informations. Il s'agit en fait de trouver un espace de dimension réduite $D' < D$ à l'aide de méthodes de sélection ou d'extraction [KLE98] sans pour autant perdre trop d'informations :

- ***Sélection de caractéristiques*** : ces méthodes ont pour but de trouver le meilleur sous-ensemble de caractéristiques parmi celles existantes. Un critère souvent utilisé dans le cadre de ces méthodes est le critère de Fisher [DOC81]. Une autre approche consiste à retenir les caractères les plus discriminants en cherchant à rendre les classes associées aux différents modes de fonctionnement les plus « éloignées » et « compactes » possibles. Cette approche utilise des critères à base de matrice de covariance [DUB90a].

- ***Extraction de caractéristiques*** : ces méthodes visent à définir des nouvelles caractéristiques à partir de celles initiales, tout en conservant un maximum d'informations. Parmi ces méthodes, il y a l'Analyse en Composantes Principales [SAP90] qui permet d'extraire des nouvelles caractéristiques d'après l'étude des corrélations linéaires entre les données initiales, ou plus récemment l'Analyse en Composantes Curvilignes [JAU99] qui étend l'ACP à l'analyse des corrélations non linéaires entre les caractéristiques.

D'un point de vue théorique, les méthodes d'extraction sont optimales, toutefois en pratique, les méthodes de sélection ont l'avantage de conserver l'interprétabilité des caractéristiques ce qui est très intéressant pour le diagnostic. Par ailleurs, en diagnostic, une simplification trop importante de l'espace de représentation lors de la phase d'analyse peut être néfaste lors de la phase d'exploitation. En effet, une simplification trop hâtive génère un risque de non-détection de modes de fonctionnement non présents dans la base d'apprentissage.

2.1.2. Détermination de l'espace de décision

A ce stade de la phase d'analyse, l'espace de représentation $\Re^D$ dans lequel les états de fonctionnement du procédé surveillé sont décrits est défini. Cette dernière étape consiste à définir l'espace de décision $C = \{C^1, ..., C^K\}$, i.e. à déterminer les régions de l'espace $\Re^D$ représentatives des modes de fonctionnement. L'opération visant à séparer en régions l'ensemble des observations de la base d'apprentissage X, fait appel à des méthodes de ***classification***. Le type d'une méthode de classification se décline généralement en deux familles : le mode supervisé et le mode non supervisé. En RdF, cette distinction se fait en introduisant les notions d'***apprentissage supervisé*** ou d'***apprentissage non supervisé*** pour la définition d'un classifieur :

- ***Apprentissage supervisé*** : lorsque l'on connaît *a priori* les modes réels de fonctionnement des observations contenues dans la base d'apprentissage X, on parle alors d'apprentissage en mode supervisé. Le classifieur est paramétré sous le contrôle d'un expert grâce aux éléments de la base d'apprentissage dont le mode de fonctionnement est parfaitement connu, ce qui permet de définir l'espace de décision C. Cette classification est couramment reprise sous la dénomination de ***classification supervisée.***

- ***Apprentissage non supervisé*** : malheureusement, il n'est pas toujours possible de disposer *a priori* de la connaissance du mode de fonctionnement de chaque observation de la base d'apprentissage. En diagnostic, il s'agit de situations où l'étiquetage des observations d'apprentissage est une opération longue et fastidieuse. Il s'agit alors de mettre en évidence une structure de classes dans un ensemble d'observations pour lesquelles on ne dispose d'aucune information *a priori* quant à leur appartenance à un mode de fonctionnement. On parle d'apprentissage en mode non supervisé, ce qui se traduit par l'utilisation d'algorithmes de ***classification non supervisée*** ou ***automatique***. Lors d'un apprentissage non supervisé, il est important de noter que la ***labelisation des classes en terme de modes de fonctionnement*** reste très délicate mais indispensable au diagnostic. Cette tâche est généralement réalisée par un expert du procédé.

Par ailleurs, on peut noter qu'entre ces deux types extrêmes de connaissance *a priori* sur le système, il existe une troisième situation d'apprentissage pour laquelle on dispose d'une

information partielle sur les données d'apprentissage. On utilise alors des méthodes partiellement supervisées [PED90]. Par la suite, nous ne considérons que l'apprentissage non supervisé, étant donné que dans le cadre du diagnostic, il est souvent difficile de disposer d'une connaissance *a priori* complète et/ou fiable sur l'ensemble des modes de fonctionnement. En pratique, la base d'apprentissage est rarement exhaustive en raison des modes de fonctionnement inconnus ou pour lesquels il n'a pas été possible de collecter d'informations.

La phase de détermination de l'espace de décision C se conclut par la définition de la ***règle de décision*** utilisée pour le classement des nouvelles observations. Ainsi, la fonction de décision (ou fonction discriminante), notée $\hat{u}$ (II.1), définit une partition de l'espace de représentation en autant de régions que de modes de fonctionnement et permet d'associer toute nouvelle observation X^i à l'un des K modes de fonctionnement :

$$\begin{array}{rccc} \hat{u}: & \Re^D & \rightarrow & \{1,\cdots,K\} \\ & X^i & \mapsto & \hat{u}\left(X^i\right) \end{array} \quad \text{avec} \quad \hat{u}\left(X^i\right) = k \text{ si } X^i \in C^k \qquad \text{(II.1)}$$

Lors de la définition de cette règle de décision, on parle couramment de ***frontières de décision*** entre les modes de fonctionnement. L'obtention de ces frontières peut se faire suivant deux approches : analytique ou statistique [AMB97][DUB90a]. Toutefois, ces deux approches ne sont pas cloisonnées et sont parfois équivalentes dans certaines circonstances :

- ***Approche analytique*** : on détermine directement la fonction discriminante en estimant les paramètres d'une fonction mathématique de manière à séparer au mieux les modes de fonctionnement, i.e. à minimiser la probabilité de mauvais classement. Le choix du modèle pour la fonction mathématique dépend de la complexité de la frontière de décision à mettre en place. En général, on commence par utiliser des fonctions linéaires [HO65], puis si les performances de classement ne sont pas satisfaisantes, on fait appel à des fonctions quadratiques [CHA83], ou encore à des modèles neuronaux. Dans ce sens, les Machines à Vecteurs de Support (SVM en anglais) recherchent les frontières de décision maximisant la marge séparatrice entre les classes [GUN98].

- ***Approche statistique*** : on considère que chaque observation X^i de $\Re^D$ suit, dans chaque mode de fonctionnement C^k, une loi de probabilité notée $f\left(X^i / C^k\right)$. Les concepts issus de la théorie statistique de la décision [FER67][FUK90] et de l'analyse discriminante [LAC79] peuvent alors être utilisés pour établir les frontières de décision. On distingue généralement deux catégories de méthodes selon le type de modélisation :

  - les ***modèles paramétriques*** pour lesquels $f\left(X^i / C^k\right)$ est supposée connue pour chaque mode de fonctionnement C^k, tout comme les probabilités *a priori* $P\left(C^k\right)$. La règle de décision optimale est alors définie à partir de la théorie Bayésienne de la décision qui permet de déterminer $P\left(C^k / X^i\right)$,

  - les ***modèles non paramétriques*** pour lesquels les lois de probabilités sont inconnues. Soit on estime $f\left(X^i / C^k\right)$ à l'aide de méthodes comme les noyaux de Parzen [DUB90a], soit on estime directement les probabilités *a posteriori* $P\left(C^k / X^i\right)$ à l'aide de la méthode des k Plus Proches Voisins (k-PPV) [COV67].

Par ailleurs, à mi-chemin entre ces deux types de modèles, on distingue les ***modèles de mélange*** [AMB97]. L'algorithme le plus populaire pour l'estimation des paramètres des modèles de mélange est l'algorithme d'***Expectation-Maximization*** (EM) [BIE99a].

A l'issue de la phase d'analyse, l'espace $\Re^D$ de représentation des données sur le système et l'espace de décision C définissant K régions de décision au sein de $\Re^D$ sont déterminés. De plus, une règle de décision (frontières entre ces régions) permet d'associer toute nouvelle observation X^i sur le procédé à l'un des K modes de fonctionnement.

2.2. Phase d'exploitation

La phase d'analyse a permis d'élaborer le système de diagnostic par RdF ce qui permet de passer désormais en phase d'exploitation, i.e. au classement d'observations n'appartenant pas à la base d'apprentissage. Toutefois, il faut se rappeler que lors de la phase d'apprentissage, une hypothèse a été prise sur la connaissance *a priori* d'observations représentatives de tous les modes de fonctionnement du système. Dans ce sens, l'approche du diagnostic par RdF souffre d'un problème épineux puisqu'en pratique la connaissance *a priori* est rarement exhaustive. Tous les modes de fonctionnement ne sont pas forcément identifiés. C'est le cas par exemple pour les systèmes évoluant dans le temps ou pour ceux où le recueil de données sur certains modes de fonctionnement est impossible. Cette non exhaustivité de la base d'apprentissage engendre alors ***deux inconvénients majeurs*** lors de l'écriture de la ***règle de décision*** :

- Lorsqu'une observation à classer se trouve « près » d'une frontière de décision, elle se trouve en situation d'ambiguité entre deux ou plusieurs modes de fonctionnement et son affectation à l'un d'eux induit un risque de mauvais classement.
- Lorsqu'une observation à classer se trouve dans une région de l'espace de représentation « éloignée » de tous les modes de fonctionnement connus, son affectation au mode « le plus proche » induit généralement une erreur de classement.

Les méthodes classiques de RdF présentées jusqu'ici ont alors besoin d'être étendues pour prendre en compte ces situations. En effet, la remise en cause du système de décision, et donc de la règle de décision, est nécessaire lors de l'apparition de nouveaux modes de fonctionnement.

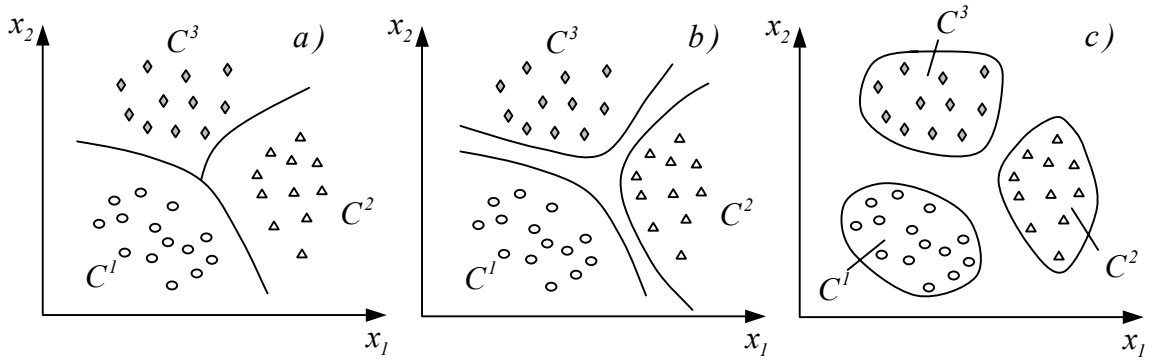


Figure II.6 : Définition des frontières de décision entre les classes : a) sans rejets, b) avec rejet d'ambiguïté, c) avec rejet d'ambiguïté et de distance.

Une première réponse à ce problème a été d'introduire respectivement les notions de ***rejet d'ambiguïté*** [CHO70] et de ***rejet de distance*** [DUB90b] qui constituent la base essentielle

de toute règle de décision d'un système de diagnostic par RdF, et auxquels on a associé les classes de rejet d'ambiguïté C^{amb} et de rejet en distance C^{dist}.

La figure II.6.a présente ainsi les frontières de décision obtenues sans ajout de rejets dans la règle de décision, la figure II.6.b avec l'introduction de la notion de rejet d'ambiguïté, et enfin la figure II.6.c montre la définition des frontières de décision en utilisant conjointement les notions de rejets d'ambiguïté et de distance.

2.3. Conclusion sur le diagnostic par l'approche RdF

Après avoir décrit les différentes méthodes de diagnostic, cette deuxième partie de chapitre a permis de clarifier les ***liens entre le diagnostic et la RdF***, en décrivant les phases d'élaboration d'un système de diagnostic par RdF.

La ***phase d'analyse*** débute par la définition d'un ***espace de représentation*** des données caractéristiques du système étudié, extraites des signaux relevés via des capteurs. A partir de l'historique du procédé et de la connaissance disponible sur les différents modes de fonctionnement, une ***base d'apprentissage*** est élaborée. Une phase d'***apprentissage*** faisant appel à des ***techniques de classification*** supervisée ou non supervisée est réalisée afin de déterminer un ***espace de décision*** définissant des régions de l'espace de représentation associées aux différents modes de fonctionnement identifiés. Cet apprentissage donne naissance à une ***règle de décision*** traduisant les frontières entre les modes de fonctionnement.

La ***phase d'exploitation*** permet alors à l'aide de la règle de décision d'associer de nouvelles observations aux modes de fonctionnement identifiés dans l'espace de décision. Toutefois, étant donné la non-exhaustivité de la base d'apprentissage (en raison des modes de fonctionnement inconnus initialement), la règle de décision doit être pourvue de ***principes de rejets*** permettant d'écarter les observations « ambiguës » ou « éloignées ». L'intégration des notions de rejets fut une première réponse mais hélas celle-ci n'est pas toujours suffisante. En effet, l'espace de décision a parfois besoin d'être remis en cause et adapté suite aux évolutions du procédé. L'adaptabilité des systèmes de diagnostic par RdF sera abordée plus en détails dans le cadre de la troisième partie de ce chapitre.

Cependant, avant de développer cet aspect adaptatif, il est intéressant de se pencher sur la qualité de représentation des différents modes de fonctionnement. La modélisation de ceux-ci, associée à la nature imparfaite des informations utilisées limite les performances de l'approche dite « classique ». En effet, l'affectation d'une observation au modèle d'un mode de fonctionnement est définie de façon binaire avec la règle de décision, tout comme la modélisation des modes de fonctionnement au travers d'ensembles classiques. Pour la surveillance d'un système industriel, certains ***phénomènes évolutifs*** peuvent se présenter, le passage d'un mode de fonctionnement vers un autre étant rarement instantané. Avec la RdF classique, la modélisation des modes de fonctionnement ne permet pas de « quantifier » les évolutions des observations, or un système de diagnostic doit pouvoir les détecter.

Ces contraintes de représentation ou de modélisation sont solutionnées avec l'introduction de la théorie des sous-ensembles flous et plus généralement de la ***logique floue*** : on parle de ***reconnaissance des formes floue***.

3. Le diagnostic par l'approche RdF floue

Dans certaines situations, le ***concept de modes de fonctionnement*** est fondamentalement ***imprécis*****,** d'où la difficulté de modélisation. Prenons par exemple le problème de suivi de l'usure d'un outil de coupe [ZIE95]. Initialement l'outil est bon puis il se dégrade jusqu'à atteindre un état d'usure inacceptable : il y a bien transition d'un état « bon » vers un état « mauvais ». Bien qu'il soit difficile de donner une vérité binaire aux propositions que l'outil soit « bon » ou « mauvais », il semble envisageable d'y associer une valeur de vérité sur $[0,1]$. En fait, la définition des modèles de modes de fonctionnement, tout comme l'affectation d'une observation, sont sources d'imperfections [PED90], que ce soit l'imprécision lors de la création du modèle ou l'incertitude lors de la labelisation des modes de fonctionnement ou de l'affectation d'une observation. De manière courante, la notion d'***imprécision*** concerne le doute placé sur le contenu d'une proposition tandis que l'***incertitude*** est relative au doute sur la véracité même de la proposition [GAC97].

La première manifestation de la volonté de prise en compte des connaissances incertaines fut le développement de la ***théorie des probabilités*** à partir du XVIIème siècle [DUD73]. Toutefois, l'ignorance totale ou partielle est très mal prise en compte par le modèle probabiliste. On a alors assisté aux développements de la ***théorie de l'évidence*** qui offre un cadre plus général et où les raisonnements sont effectués à partir de deux mesures duales : crédibilité et plausibilité. Celles-ci permettant de mieux appréhender les concepts d'incertitude et d'imprécision [SHA76][PET99].

L'introduction de la logique floue en RdF a apporté une nouvelle solution aux problèmes d'imperfections. En fait, il s'agit d'une théorie établie en deux parties et qui a vu le jour en 1965 avec la ***théorie des sous-ensembles flous*** [ZAD65], puis en 1978 avec la ***théorie des possibilités*** [ZAD78]. La logique floue permet la modélisation de données imprécises et incertaines. La reconnaissance de formes intégrant les principes de la logique floue a donné lieu à la ***RdF floue*** [BEZ81a]. Celle-ci a trouvé de nombreuses applications dans le domaine du diagnostic. Ainsi, pour poursuivre avec l'exemple introductif, [ZIE95] propose une méthode de suivi d'un phénomène évolutif, basée sur des techniques de RdF floue, et appliquée à la détection d'usure d'outils de coupe sur des machines outils. De même, [THO96] propose une méthode de surveillance par approche de RdF floue appliquée à des situations issues du domaine automobile.

Dans cette troisième partie du chapitre, nous présentons les concepts des méthodes de diagnostic par RdF floue, en rappelant les ***principes généraux de la logique floue***, puis en précisant les évolutions respectives des ***phases d'analyse*** et d'***exploitation***. Un dernier paragraphe consacré aux ***systèmes adaptatifs de diagnostic par RdF*** apporte un complément aux problèmes de non-exhaustivité de la base d'apprentissage.

3.1. La logique floue

La logique floue a pour but de raisonner à partir de connaissances imparfaites (imprécises et incertaines) pour lesquelles la modélisation en logique classique pose des difficultés. Il est généralement admis que les prémices de la logique floue datent de 1965 avec l'introduction de la

théorie des sous-ensembles flous par Zadeh [ZAD65], bien que les travaux de Black datant de 1937 [BLA37] contiennent plusieurs des idées de Zadeh.

La ***théorie des sous-ensembles flous*** est basée sur la notion d'appartenance d'un objet à un ensemble par l'ajout d'une fonction d'appartenance graduelle aux ensembles : on parle alors d'ensembles flous. Un ***ensemble flou*** A d'univers Ω, i.e. un sous-ensemble flou A de Ω, est défini par sa ***fonction d'appartenance*** μ_A qui à chaque élément X^i de Ω associe une valeur $\mu_A(X^i)$ sur $[0,1]$ traduisant le degré d'appartenance de X^i à A.

$$\mu_A : \Omega \rightarrow [0,1] \text{ avec } \begin{cases} \mu_A(X^i) = 0 & si\ X^i \notin A \\ 0 < \mu_A(X^i) < 1 & si\ X^i \in \text{ partiellement à } A \\ \mu_A(X^i) = 1 & si\ X^i \in \text{ totalement à } A \end{cases} \qquad \text{(II.2)}$$

Selon cette définition, la théorie des sous-ensembles flous permet de modéliser essentiellement l'imprécision sur les données plus que l'incertitude. L'avantage de la théorie des sous-ensembles flous réside dans son aptitude à pouvoir traiter des notions imprécises et à y apporter une modélisation. Par opposition à la théorie des ensembles classiques, elle caractérise les ensembles en leur associant une fonction d'appartenance, qui confère aux objets d'un ensemble un degré d'appartenance à valeurs sur $[0,1]$ et non plus binaires.

Comme indiqué en introduction, une des premières manifestations pour la prise en compte de l'incertitude fut la ***théorie des probabilités***. Toutefois, le choix de cette approche probabiliste ne permet pas de modéliser l'ignorance sur la réalisation d'un événement (observation) en raison de la contrainte de normalisation introduite par la définition des distributions de probabilités.

Une nouvelle théorie a ainsi vu le jour en 1978 : la ***théorie des possibilités*** [ZAD78] qui fut très développée par [DUB87] et constitue le deuxième volet de la logique floue. Cette théorie utilise les mesures de possibilité et de nécessité qui ne sont autres que des cas particuliers des mesures de plausibilité et de crédibilité introduites en théorie de l'évidence afin de modéliser l'incertain. La théorie des possibilités permet de gérer l'incertitude sur la réalisation d'événements, tout en considérant l'ignorance, en « levant » la contrainte de normalisation pour la définition des distributions de possibilités.

Ainsi, l'association de la théorie des sous-ensembles flous et de la théorie des possibilités permet de modéliser une connaissance imparfaite à l'aide d'ensembles flous dont la fonction d'appartenance n'est autre qu'une distribution de possibilités. Ces théories constituent une réponse naturelle adaptée à la problématique de modélisation des modes de fonctionnement par les algorithmes de classification utilisés en RdF.

Avant de conclure sur ce paragraphe introductif à la logique floue, il est important de noter que l'opposition entre ***les possibilités et les probabilités*** est toujours source de vives polémiques [FIO99], notamment pour l'interprétation des ensembles flous. Quoi qu'il en soit, la théorie des probabilités et la logique floue (dont la théorie de possibilités fait partie) sont deux outils pour la modélisation de l'incertain et il n'y a pas de cloisonnement précis entre les deux disciplines [ZAD95]. Ainsi, les sous-ensembles flous peuvent avoir une interprétation probabiliste ou possibiliste.

Ainsi, alors que les méthodes usuelles de RdF (2ème partie du chapitre) ne permettent pas de modéliser l'imprécision (bruit, …) sur les données issues du système, tout comme l'incertitude (doute, ignorance, …) sur leur affectation aux différents modes de fonctionnement, la RdF floue apporte une solution intéressante. Les paragraphes suivants sont ainsi dédiés à la présentation des évolutions des phases d'analyse et d'exploitation pour l'élaboration d'un système de diagnostic par RdF floue.

3.2. Phase d'analyse

Dans le cadre de l'utilisation des techniques de RdF floue en diagnostic, la principale modification de la phase d'analyse intervient lors de l'étape de détermination de l'espace de décision qui devient flou. En effet, les modes de fonctionnement sont désormais modélisés par des ensembles flous auxquels on associe une fonction d'appartenance à valeurs sur $[0,1]$. La phase d'apprentissage vise à déterminer les paramètres de ces fonctions d'appartenance à partir des échantillons de la base d'apprentissage. Nous décrivons, ci-après, le principe de ***détermination de l'espace de décision flou***, puis les possibilités de modélisation des modes de fonctionnement à l'aide de sous-ensembles flous, ce qui permet de modifier la ***représentativité des modes de fonctionnement***.

3.2.1. Détermination de l'espace de décision flou

La détermination de l'espace de décision flou vise à définir à l'aide de méthodes de classification floue, les paramètres des fonctions d'appartenance, notées ψ_k, modélisant les différents modes de fonctionnement. Tout comme en RdF classique, plusieurs stratégies sont envisageables en fonction de la connaissance disponible sur les observations d'apprentissage.

En ***mode supervisé***, les observations d'apprentissage sont déjà étiquetées en terme de mode de fonctionnement. Il s'agit alors de réaliser un apprentissage supervisé pour définir des modèles de modes de fonctionnement en établissant pour chacun d'entre eux une fonction d'appartenance.

En ***mode non supervisé***, comme dans le cas classique, on fait appel à des algorithmes de classification non supervisée. De nombreuses méthodes existent selon la structure de classification recherchée (partition, hiérarchie, …). Parmi les algorithmes les plus connus, on peut citer celui des C-moyennes floues (Fuzzy C-Means en anglais) qui permet la construction d'une partition floue des données d'apprentissage [BEZ81a].

Dans le cadre du diagnostic, on se situe généralement dans une situation d'apprentissage non supervisé faisant appel aux ***algorithmes de classification floue non supervisée,*** détaillés plus amplement au chapitre 3. Il s'agit d'identifier parmi les données d'apprentissage des « formes-types » nommées « prototypes ». Chaque mode de fonctionnement est caractérisé par un ou plusieurs prototypes P^j auxquels on associe une fonction d'appartenance μ_j. Par la suite, le terme « ***prototype*** » définira l'ensemble formé par le point représentatif et sa fonction d'appartenance, d'où la notion de ***modèle de distribution***. Le modèle du mode de fonctionnement n°k est établi par la définition de la fonction d'appartenance ψ_k des observations à la classe C^k. Celle-ci est généralement établie à partir d'une fonction de la

distance d'une observation X^i aux prototypes P^j de C^k [MAS96]. Plusieurs possibilités sont offertes pour la représentativité des modes de fonctionnement selon le type de modélisation retenu.

3.2.2. Représentativité des modes de fonctionnement

L'espace de décision flou est défini par des sous-ensembles flous associés aux classes caractérisant les K modes de fonctionnement et pour lesquels on détermine une fonction d'appartenance. En fait, les modes de fonctionnement sont modélisés par des classes, elles-mêmes définies par des prototypes assimilés à des sous-ensembles flous. En RdF floue, la première étape consiste à choisir le gabarit ***des fonctions d'appartenance*** aux prototypes.

3.2.2.1. Gabarit des fonctions d'appartenance

Plusieurs gabarits sont envisageables et proposés dans la littérature pour la définition de la fonction d'appartenance μ_j au prototype P^j. Cette fonction doit toutefois satisfaire deux contraintes : être maximale et au voisinage de *1* pour les observations qui appartiennent de façon certaine au prototype P^j, et décroître vers *0* au fur et à mesure que les observations s'en éloignent. Ces fonctions se différencient alors par la forme et la rapidité de leur décroissance. La fonction la plus simple et la plus employée est la ***fonction exponentielle*** :

$$\mu_j(X^i, P^j) = e^{-\lambda_j . d^2(X^i, P^j)} \tag{II.3}$$

où X^i traduit une observation quelconque de l'espace de représentation, λ_j une constante utilisée pour ajuster la « raideur » de la courbe exponentielle, et $d(X^i, P^j)$ une distance définie dans l'espace de représentation $\Re^D$. Les prototypes ainsi définis sont assimilés à des modèles gaussiens. De la même manière, une autre fonction parfois utilisée est la ***fonction inverse*** :

$$\mu_j(X^i, P^j) = \frac{1}{1 + \lambda_j . d(X^i, P^j)} \tag{II.4}$$

L'emploi d'une fonction d'appartenance est subordonné au choix d'une mesure de distance d dans $\Re^D$. Il s'agit d'une mesure de proximité entre une observation X^i et le centre M_{P^j} du prototype P^j. La distance la plus simple est la ***distance Euclidienne*** :

$$d^2(X^i, P^j) = (X^i - M_{P^j})^T (X^i - M_{P^j}) \tag{II.5}$$

Toutefois, cette mesure de distance ne prend pas en compte la dispersion des données selon les composantes des observations et on lui préfère souvent la ***distance de Mahalanobis***.

$$d^2(X^i, P^j) = (X^i - M_{P^j})^T \Sigma_{P^j}^{-1} (X^i - M_{P^j}) \tag{II.6}$$

où Σ_{P^j} définit la matrice de covariance des observations associées à P^j.

Par ailleurs, deux possibilités sont offertes pour définir l'association des prototypes aux classes modélisant les modes de fonctionnement : une ***approche monoprototype*** et une ***approche multiprototype***.

3.2.2.2. Approche monoprototype

L'***approche monoprototype*** considère qu'un mode de fonctionnement (classe) peut être modélisé par un unique prototype, on a donc $\psi_k = \mu_k$.

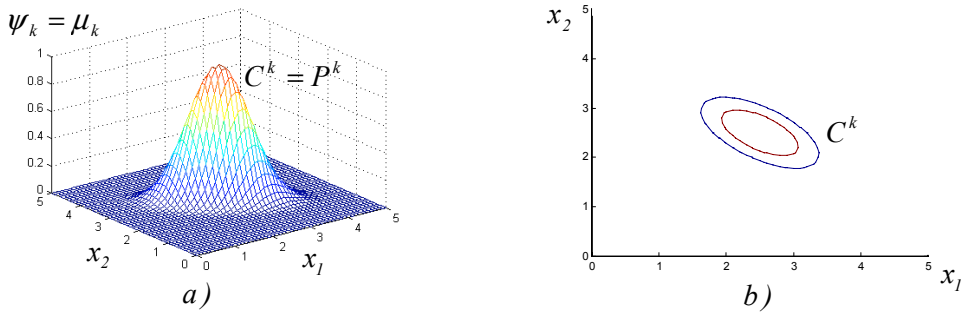


Figure II.7 : Approche monoprototype : a) Fonction d'appartenance $\psi_k = \mu_k$ à la classe C^k, b) Courbes de niveau pour deux valeurs d'appartenance ψ_k à la classe C^k.

La figure II.7 illustre l'approche monoprototype avec sur la partie a) la fonction d'appartenance à la classe C^k modélisant le mode de fonctionnement n° k, et sur la partie b) deux courbes de niveaux d'appartenance à C^k.

3.2.2.3. Approche multiprototype

L'approche monoprototype suppose toutefois que les modes de fonctionnement peuvent être modélisés par une classe de forme simple. Si la forme de la classe est plus complexe, il devient nécessaire de définir plusieurs prototypes pour représenter le mode de fonctionnement, on parle d'***approche multiprototype***. La fonction d'appartenance ψ_k à la classe C^k peut alors être définie à partir des fonctions d'appartenance μ_j aux J prototypes représentatifs de C^k :

$$\psi_k(X^i) = min(1, \sum_{j=1}^{J} \mu_j(X^i, P^j)) \qquad \text{(II.7)}$$

où chaque fonction μ_j est construite à partir de l'une des fonctions types précédentes.

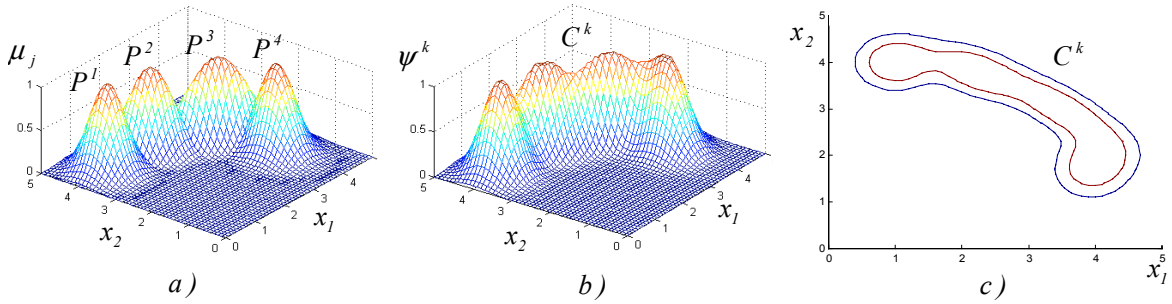


Figure II.8 : Approche multiprototype : a) Fonctions d'appartenance μ_j aux prototypes P^j, b) Fonction d'appartenance ψ_k à la classe C^k, c) Courbes de niveau pour deux valeurs d'appartenance ψ_k à la classe C^k.

Cette approche est illustrée par la figure II.8 où l'on a représenté d'une part les fonctions d'appartenance aux 4 prototypes associés à la classe C^k modélisant le mode de fonctionnement n° k, partie a), et d'autre part la fonction d'appartenance à la classe C^k, partie b), déterminée à partir de l'expression (II.7). Enfin, la partie c) représente les courbes de niveaux associées à deux valeurs d'appartenance à la classe C^k.

3.3. Phase d'exploitation

Comme en RdF classique, la phase d'exploitation a pour objet d'associer une nouvelle observation à l'un des K modes de fonctionnement. Toutefois, l'espace de décision est désormais défini par un ensemble de classes avec leur fonction d'appartenance graduelle. Ainsi, la phase d'exploitation utilise une ***règle de décision floue*** qui, à partir des différents degrés d'appartenance aux modèles de modes de fonctionnement identifiés dans l'espace de décision flou, permet l'étiquetage d'une observation correspondant à l'un des modes de fonctionnement. Sur ce principe, la figure II.9 illustre la définition d'un vecteur ψ des degrés d'appartenance respectifs d'une observation X^i aux K classes de l'espace de décision C.

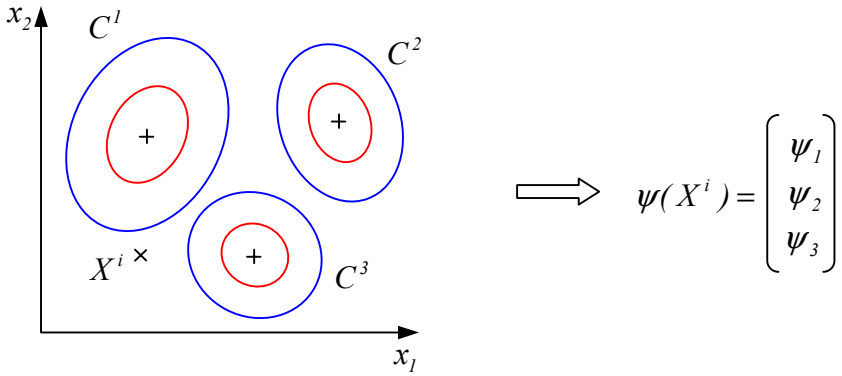


Figure II.9 : Définition d'un vecteur ψ des degrés d'appartenance d'une observation X^i à 3 modèles de modes de fonctionnement.

En fait, la différence fondamentale avec le cas classique provient du fait que la définition de la règle de décision floue se fait dans l'***espace des fonctions d'appartenance*** $[0,1]^K$ et non plus dans l'espace de représentation [DUB94]. La réussite de la phase d'exploitation se situe dans l'élaboration de la règle de décision floue qui doit par ailleurs intégrer les principes de rejets.

3.3.1. Règle du maximum d'appartenance

La règle la plus couramment utilisée est la ***règle du maximum d'appartenance*** introduite par [PAL77] et qui consiste à affecter l'observation X^i à la classe C^k pour laquelle la fonction d'appartenance ψ_k renvoie une valeur maximale :

$$X^i \in C^k \quad si \quad \psi_k(X^i) = max_{j=1,K}(\psi_j(X^i)) \tag{II.8}$$

Cette règle de décision est très utilisée en raison de sa simplicité d'utilisation. Toutefois, elle peut faire apparaître des cas d'indécisions, notamment lorsque les degrés d'appartenance d'une

observation à plusieurs classes sont proches ou encore lorsque le degré maximal d'appartenance est faible. En fait, il s'agit des situations de rejets évoquées au paragraphe 2.2. Dans le cadre d'une interprétation probabiliste des ensembles flous, l'utilisation de la contrainte de normalisation introduite par les distributions de probabilité définissant les fonctions d'appartenance suppose un caractère exhaustif des modes de fonctionnement [DUB87], ce qui exclut l'utilisation de rejet d'appartenance (en distance). En fait, le ***rejet d'appartenance n'est possible que dans le cadre d'une approche possibiliste***.

3.3.2. Règle du rapport d'appartenance

Avec l'approche possibiliste, les notions de rejets peuvent être incluses dans la règle de décision floue. Parmi les règles les plus connues, la ***règle du rapport d'appartenance*** s'exprime à l'aide d'un seuil ψ_{min} pour le ***rejet d'appartenance*** et d'un seuil T_r pour le ***rejet d'ambiguïté*** :

$$X^i \text{ affecté à } \begin{cases} C^{app} & si \ \ Card(A) = 0 \\ C^k & si \ \ Card(A) = 1 \ \ ou \ \ R_p < T_r \\ C^{amb} & si \ \ Card(A) > 1 \ \ et \ \ R_p > T_r \end{cases} \qquad \text{(II.9)}$$

où A désigne l'ensemble des classes vérifiant l'expression (II.10) :

$$A = \left\{ C^j \ / \ \psi_j(X^i) > \psi_{min} \right\} \qquad \text{(II.10)}$$

et R_p désigne le rapport d'appartenance (II.11) utilisé pour l'option de rejet d'ambiguïté :

$$R_p = \frac{\Psi_p(X^i)}{\Psi_k(X^i)} \ avec \begin{cases} k \ tel \ que \ \Psi_k(X^i) = max_{j=1,K}(\Psi_j(X^i)) \\ \Psi_p(X^i) = max_{j=1,K, j\neq k}(\Psi_j(X^i)) \end{cases} \qquad \text{(II.11)}$$

Ainsi, une observation est rejetée en appartenance si son degré d'appartenance au sens d'une possibilité est inférieur au seuil ψ_{min} pour l'ensemble des classes. De même, une observation est dite ambiguë si ses deux coefficients ou possibilités d'appartenance les plus grands ont des valeurs proches, i.e. si leur rapport d'appartenance R_p est supérieur au seuil T_r.

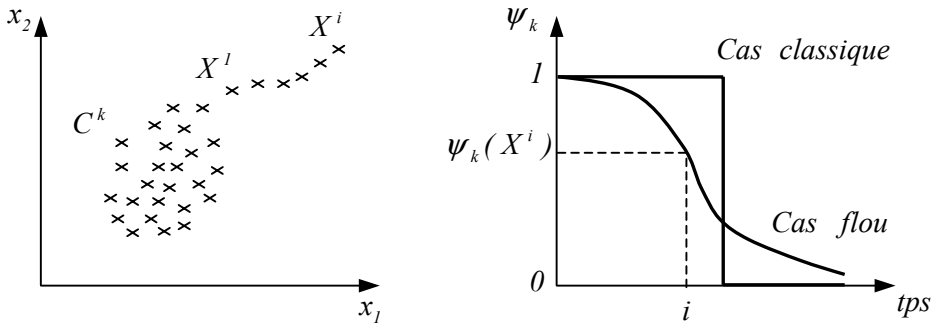


Figure II.10 : Modélisation de l'évolution des observations.

A première vue, il peut paraître paradoxal de revenir à une décision binaire d'appartenance aux modes de fonctionnement après avoir adopté une modélisation floue de ceux-ci. Mais en réalité, le fait de prendre une décision stricte n'empêche nullement le diagnosticien ou l'expert

de considérer les différents degrés d'appartenance et ainsi d'étudier l'évolution de l'état de fonctionnement du système.

En effet, l'introduction des fonctions d'appartenance à valeurs graduelles offre la possibilité très intéressante de suivre l'évolution des nouvelles observations sur le procédé par rapport aux différents modèles des modes de fonctionnement et c'est bien là un des avantages indéniables de l'utilisation de la RdF floue (figure II.10). En fait, il s'agit du premier pas vers l'établissement d'une prédiction sur l'évolution future du système [FRE92] et non plus d'une simple association à un mode de fonctionnement.

Ces dernières années de nombreux travaux se sont intéressés à l'***aspect pronostic*** [FRE92][GAN87] tout comme à l'***aspect adaptatif*** [BOU98][DEV99][THO96] des systèmes de diagnostic. L'opération de pronostic (ou de prédiction) vise à informer sur l'évolution future du procédé et constitue souvent la finalité dans la mise en œuvre des programmes de maintenance prévisionnelle, alors que l'adaptabilité du système de diagnostic traduit la capacité de celui-ci à prendre en compte les évolutions de la connaissance sur le procédé étudié. Avant de pouvoir réaliser un pronostic, il est intéressant de considérer l'aspect adaptatif. Ainsi avant de conclure sur l'approche du diagnostic par RdF floue, une brève présentation sur les ***systèmes adaptatifs de diagnostic*** est proposée.

3.4. Systèmes adaptatifs de RdF pour le diagnostic

Dans les paragraphes précédents, tous les modes de fonctionnement du procédé sont considérés connus *a priori* et donc représentés au sein de l'historique de ce dernier. Or, rien ne permet de garantir l'exhaustivité de la base d'apprentissage, ne serait-ce que pour des raisons d'impossibilités techniques d'acquisitions de données sur des modes anormaux. De même, au cours de la vie du procédé, des nouveaux modes de fonctionnement peuvent apparaître ou ceux existants évoluer.

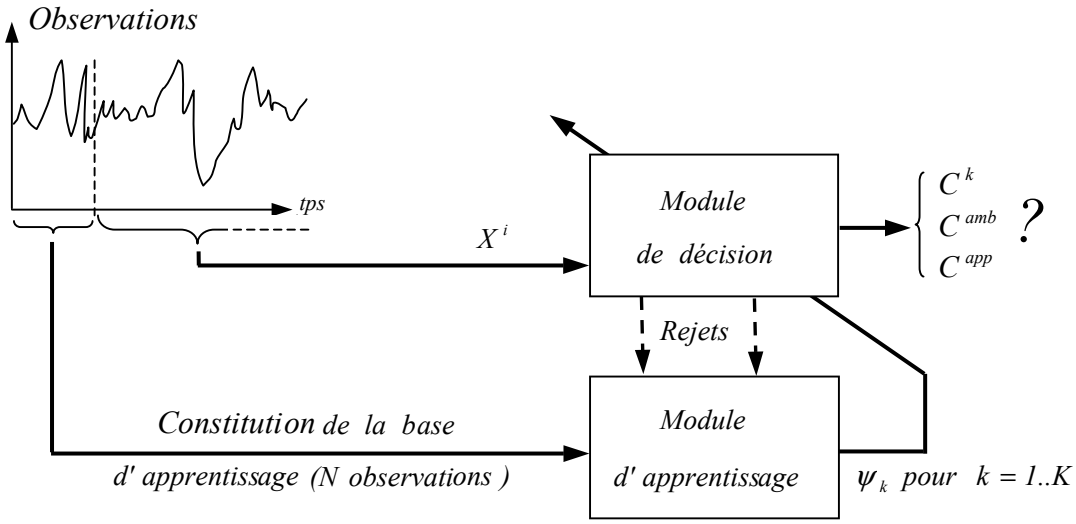


Figure II.11 : Structure de base d'un système adaptatif de diagnostic.

Ainsi, il est devenu impératif d'élaborer un ***système adaptatif de diagnostic*** afin d'autoriser l'apparition de nouveaux modes de fonctionnement et l'évolution des modes existants, en

remettant en cause le nombre K de modèles de modes de fonctionnement identifiés ainsi que les paramètres de ces modèles.

Sur ce principe, [FRE92] a proposé une des premières structures adaptatives illustrées sur la figure II.11. Le système de RdF est initialisé au cours d'une phase d'apprentissage initial sur la base d'un ensemble fini d'observations connues sur le système. Lors de la phase d'exploitation, le module de décision permet alors d'affecter les observations présentées aux classes existantes C^k ou aux classes de rejets C^{app} et C^{amb}. Sur ce principe, dès l'occurrence d'un nombre N_{app} prédéterminé de rejets en appartenance ($\left|C^{app}\right| > N_{app}$), l'apprentissage est remis en question. Cette approche reste toutefois assez limitée. En effet, le caractère adaptatif proposé est souvent incomplet étant donné que la remise en cause de l'espace de décision n'est considérée que périodiquement. En réalité, différentes typologies d'apprentissage sont envisageables : l'***apprentissage hors ligne*** réalisé de manière périodique, et l'***apprentissage en ligne*** réalisé en continu. En apprentissage hors ligne, on distingue deux situations. Dans les deux cas, on ajoute les observations ayant provoquées le besoin d'adaptation à l'ensemble X puis, soit on relance totalement l'apprentissage, soit on adapte l'espace de décision en ne considérant que les nouvelles observations. En apprentissage en ligne, toutes les observations présentées au module de décision sont prises en compte pour l'adaptation de l'espace de décision.

Evidemment, que ce soit en RdF classique ou floue, des principes de rejets en appartenance et en ambiguïté ont été intégrés à la règle de décision comme première solution. La quasi-totalité des systèmes adaptatifs actuels de diagnostic s'appuie sur la structure de [FRE92] à laquelle des améliorations ont été apportées. [PEL93] a développé un système de diagnostic d'évolution basé sur un principe de rejet auquel est associé un critère de voisinage spatial et temporel. Les observations rejetées sont agglomérées en groupes selon ce critère, un groupe permettant la création d'un nouveau modèle de fonctionnement dès lors qu'il atteint un effectif suffisant. L'apprentissage est continuellement remis en cause, mais l'adaptation du classifieur se fait toujours hors ligne. [THO96] donne de plus amples explications sur cette approche. De son coté, [GOM95] a mis en place une architecture neuronale utilisant des fonctions à base radiale permettant de modéliser chaque mode de fonctionnement par un prototype de forme hypersphérique. Les prototypes sont adaptés en ligne, ce qui permet d'intégrer rapidement l'occurrence d'un nouveau mode de fonctionnement. Une expérimentation de cette approche est d'ailleurs proposée dans [GOM96]. Plus récemment, [BOU98] a proposé un algorithme dynamique d'apprentissage pour l'élaboration en ligne des fonctions d'appartenance des différents modes de fonctionnement, auquel il a associé un module de détection permettant d'écarter les états transitoires et de détecter la stabilisation dans un mode connu ou inconnu. Enfin, [DEV99] présente un système où le module d'apprentissage propose une adaptation hors ligne des fonctions d'appartenance lorsque les classes de rejets atteignent un effectif donné, mais également un apprentissage en ligne intégrant l'enrichissement de la base de données lorsqu'une observation nouvelle est affectée à l'une des classes déjà identifiées. En fait, encore peu de travaux traitent concrètement des problèmes d'adaptation en ligne des systèmes de diagnostic, et c'est bien à ce niveau que se positionne notre travail.

3.5. Conclusion sur le diagnostic par l'approche RdF floue

Pour conclure sur cette troisième partie, il faut se rappeler que le diagnostic par RdF est très approprié pour la surveillance des systèmes complexes, mais que l'utilisation des techniques classiques de RdF limite les capacités de modélisation des modes de fonctionnement. L'introduction des principes de la logique floue aux méthodes de RdF a alors donné naissance à la RdF floue permettant la modélisation de données imparfaites. La logique floue repose en réalité sur deux théories : la théorie des sous-ensembles flous et la théorie des possibilités.

L'utilisation de la ***théorie des sous-ensembles flous*** lors de l'élaboration d'un système de diagnostic par RdF vise à modéliser les modes de fonctionnement à l'aide d'ensembles flous associés à des fonctions d'appartenance à valeurs sur $[0,1]$. Ce principe permet de définir une modélisation de données imprécises en associant à chaque observation un degré d'appartenance entre 0 et 1 aux différents modèles de modes de fonctionnement.

La ***théorie des possibilités*** permet, quant à elle, de donner une interprétation particulière à la fonction d'appartenance afin de gérer le problème d'incertitude sur les données. Historiquement, une des premières réponses au problème d'incertitude a été la théorie des probabilités. Toutefois, cette théorie ne permet pas de considérer l'ignorance sur la réalisation d'une observation en raison de la contrainte de normalisation placée sur les distributions de probabilités. En effet, cette contrainte suppose une connaissance exhaustive des modes de fonctionnement (i.e. des distributions), ce qui est difficilement envisageable en diagnostic.

L'utilisation des techniques de RdF floue pour l'élaboration d'un système de diagnostic suit la même démarche que pour les techniques classiques. Lors de la phase d'**analyse**, on structure les observations contenues dans la base d'apprentissage en caractérisant des modèles de modes de fonctionnement à l'aide de fonctions d'appartenance associées à des sous-ensembles flous, ce qui permet de déterminer l'espace de décision flou. En phase d'***exploitation***, l'association des nouvelles observations aux modes de fonctionnement se fait par une règle de décision floue utilisant les degrés d'appartenance aux différents modèles. Celle-ci intègre les concepts de ***rejets d'appartenance et d'ambiguïté*** nécessaires à la définition d'un système adaptatif de diagnostic. La RdF floue apporte une meilleure modélisation des modes de fonctionnement et permet de représenter l'évolution progressive de l'état de fonctionnement d'un système à partir de son mode de bon fonctionnement. Le problème est alors celui d'une ***analyse de tendances*** des degrés d'appartenance aux différents modèles de modes de fonctionnement.

4. Conclusion

Ce deuxième chapitre a eu pour vocation de positionner les principes des méthodes de diagnostic actuelles. Ainsi, dans une première partie, les ***différentes typologies de méthodes de diagnostic*** (symboliques, internes, externes) ont été exposées, puis ensuite les techniques de ***diagnostic par RdF puis par RdF floue*** ont été décrites de manière plus approfondie.

Les ***méthodes symboliques*** sont essentiellement utilisées comme outils d'aide au diagnostic. Ces méthodes s'appuient sur la connaissance d'informations symboliques pour caractériser les différentes relations causes (défauts) / effets (symptômes).

Les ***méthodes internes*** sont intéressantes pour la qualité du diagnostic fourni, mais nécessitent hélas la connaissance ou l'élaboration d'un modèle mathématique du procédé, ce qui est parfois très laborieux notamment pour les systèmes complexes. Ces méthodes s'appuient sur la définition de résidus caractérisant l'« écart » entre le système réel et son modèle estimé. Des outils de décision permettent alors d'accomplir les tâches du diagnostic.

Enfin, la troisième catégorie de méthodes présentées est connue sous le nom de ***méthodes externes*** avec les méthodes de diagnostic par ***approche de reconnaissance de formes***. Ces méthodes ont l'avantage de minimiser la connaissance *a priori* sur le procédé. L'état de fonctionnement est représenté à l'aide d'un vecteur de caractéristiques extraites de signaux relevés sur le procédé, celles-ci définissant l'espace de représentation. L'idée générale est de caractériser les modes de fonctionnement par des classes, la tâche de diagnostic consistant à résoudre le problème de classement des nouvelles observations dans ces classes à l'aide d'une règle de décision. Pour cela, une ***phase d'apprentissage*** des classes modélisant les modes de fonctionnement est nécessaire et aboutit à la création d'un espace de décision. Celle-ci fait appel à des algorithmes de classification supervisée ou non, et est menée sur la base d'un historique sur le procédé. Enfin, il est important de noter que pour réaliser un diagnostic lors du classement de nouvelles observations, il faut passer par une phase de labelisation des classes identifiées au sein de l'espace de décision.

L'introduction de la ***logique floue*** aux méthodes de RdF a permis d'accroître leurs qualités de modélisation en caractérisant les modes de fonctionnement par des classes définies à l'aide de sous-ensembles flous et de leur fonction d'appartenance. La phase d'apprentissage vise à déterminer les paramètres des modèles des différents modes de fonctionnement. Une ***règle de décision floue***, intégrant les concepts de ***rejet d'ambiguïté*** et de ***rejet d'appartenance,*** est alors établie dans l'espace des fonctions d'appartenance afin de permettre l'affectation d'une observation à un mode de fonctionnement. De plus, au-delà de cette décision d'affectation, ces méthodes ont l'avantage de permettre le suivi des observations par rapport aux différents modèles des modes de fonctionnement.

L'ensemble de ce chapitre a permis de mettre en évidence les ***aptitudes de la RdF floue*** pour l'élaboration d'un système de diagnostic, que ce soit du point de vue de la modélisation des modes de fonctionnement ou de la caractérisation des phénomènes évolutifs. Dans ce sens, les méthodes de diagnostic par RdF floue semblent très prometteuses et vouées à de nombreux développements à venir. Toutefois, il serait intéressant maintenant d'étudier les techniques d'***apprentissage d'un système de RdF*** et plus particulièrement les méthodes de classification utilisées pour l'élaboration d'un classifieur. L'apprentissage initial d'un système de RdF est généralement établi hors ligne. Toutefois, comme nous l'avons vu précédemment, un système de diagnostic par RdF doit être adaptatif.

Ainsi, le chapitre suivant est dédié aux ***systèmes adaptatifs de RdF.*** Dans un premier temps, des ***algorithmes de classification non supervisée*** sont présentés. Ceux-ci peuvent être utilisés lors d'apprentissages hors ligne. Différentes situations, où le ***besoin d'adaptation*** d'un système de RdF peut se faire ressentir, sont alors décrites. Des ***algorithmes de classification dynamique*** sont alors exposés pour définir un module d'***apprentissage en ligne***.

***Chapitre III* : Systèmes adaptatifs de RdF : Classification non supervisée, Classification dynamique de données évolutives**

0. Introduction

Après avoir détaillé les principes, la démarche d'élaboration, et les avantages de l'approche du diagnostic par des techniques de RdF floue, il est intéressant de décrire la phase essentielle pour la réussite d'un système de RdF floue : ***la phase d'apprentissage***. En fait, celle-ci consiste en la construction d'un classifieur caractérisant une structure de classes parmi un ensemble X d'observations (définies au sein de l'espace de représentation $\Re^D$) et traduisant l'espace de décision floue. A chaque classe est associée une fonction d'appartenance à valeurs sur $[0,1]$. Le paramétrage du classifieur est obtenu à l'aide de méthodes de classification floue, à apprentissage supervisé ou non, selon la connaissance *a priori* sur l'appartenance des observations de X aux classes. Par la suite, nous ne considérons que le mode d'apprentissage non supervisé. La phase d'exploitation vise alors à associer toute nouvelle observation X^i à l'une des classes identifiées au sein de l'espace de décision C. Ce classement s'effectue à l'aide d'une règle de décision floue qui s'appuie sur les degrés d'appartenance de l'observation X^i aux différentes classes identifiées.

Cette démarche, qui repose essentiellement sur le résultat de la phase d'apprentissage, soulève quelques questions sur l'***exhaustivité*** et la ***validité*** de la ***base d'apprentissage*** utilisée. Toutes les classes sont-elles correctement représentées parmi les observations disponibles initialement? Toutes les classes sont-elles connues *a priori*? N'y a t-il pas des classes pour lesquelles l'acquisition de données est difficile voire impossible? Une classe sera t-elle toujours valide si l'on considère la dimension temporelle des observations associées à cette dernière? En fait, rien ne permet d'assurer l'exhaustivité de la connaissance initiale et donc de garantir la pérennité de l'apprentissage d'un système de RdF. Au fur et à mesure de la présentation de nouvelles observations, de nouvelles classes peuvent apparaître dans la structure initiale. De même, les classes existantes peuvent évoluer en fusionnant avec d'autres, en se scindant, en se déformant ou en se déplaçant au sein de l'espace de représentation. Dès la création d'un système de RdF, il est donc nécessaire de lui intégrer un caractère adaptatif : on parle de ***systèmes adaptatifs de RdF***. Deux principes d'apprentissage sont possibles : un ***apprentissage hors ligne*** et un ***apprentissage en ligne***. L'apprentissage hors ligne peut être utilisé pour la phase d'apprentissage initial ou pour des remises en question périodiques de l'espace de décision. L'apprentissage en ligne permet, quant à lui, de modéliser en continu l'espace de décision tout au long de la phase d'exploitation du système de RdF.

Ainsi, la première partie de ce chapitre est consacrée à la présentation de quelques ***algorithmes de classification non supervisée.*** Ceux-ci permettent de caractériser une structure

de classes au sein d'un nuage fini d'observations et peuvent être utilisés lors d'apprentissage hors ligne. Dans une deuxième partie, différentes situations, où le ***besoin d'adaptation des systèmes de RdF*** est mis en évidence, sont décrites. Ces situations sont illustrées au travers d'évolutions possibles des observations et des classes. Enfin, la troisième partie est dédiée à une présentation d'algorithmes de ***classification dynamique de données évolutives*** permettant une adaptation de l'espace de décision d'un système de RdF à partir d'observations nouvelles. En fait, les algorithmes présentés disposent d'une architecture évolutive et de règles d'apprentissage non supervisé, ce qui leur confèrent des capacités d'auto-adaptation.

1. La classification non supervisée

D'un point de vue général, les méthodes de classification ont pour but de regrouper les éléments d'un ensemble $X = \{X^1, ..., X^n, ..., X^N\}$ en un nombre K optimal de classes selon leurs ressemblances. Les techniques de ***classification non supervisée*** ont l'avantage de permettre la formation de classes dans un ensemble d'observations sur lesquelles on ne dispose pas d'informations quant à leur appartenance aux éventuelles classes. Il existe un grand nombre de typologies de méthodes de classification que l'on peut regrouper en plusieurs catégories.

Une première distinction consiste à séparer les techniques ***statistiques*** de celles qui sont ***floues***. Toutefois, cette distinction n'est pas toujours évidente puisqu'il a été montré que l'algorithme EM, qui fait partie des techniques statistiques, peut également être interprété comme un algorithme de classification floue [CEL94]. De son coté, [BIE99b] présente cinq catégories de méthodes : les ***méthodes statistiques***, les ***méthodes hiérarchiques***, les ***méthodes par graphes***, les ***méthodes itératives*** et les ***méthodes connexionnistes***. Par ailleurs, la définition des classes et des relations entre celles-ci dépend de la structure de classification utilisée. Dans ce sens, les principales structures sont la ***hiérarchie*** et la ***partition*** [THI97].

En fait, les techniques de classification les plus utilisées sont celles qui produisent une partition floue d'un ensemble d'observations. Ces méthodes sont connues sous le nom de ***méthodes de coalescence floue*** (regroupement). Ces méthodes optimisent itérativement un critère de classification afin d'établir une partition floue d'un ensemble de données. La notion de ***partition floue*** qui s'est avérée très utile pour le développement des techniques floues de RdF a été introduite par Ruspini [RUS69]. Au sens de Ruspini, une K-partition floue d'un ensemble X peut être obtenue en définissant K sous-ensembles flous de X tel que la somme des degrés d'appartenance pour chaque observation de X soit unitaire. En fait, on associe à chaque observation $X^n \in X$ un vecteur de degrés d'appartenance aux différentes classes. La juxtaposition de ces vecteurs pour l'ensemble des N observations de X amène à la définition d'une matrice d'appartenance U (dimension $K \times N$) où l'élément u_{kn} représente le coefficient d'appartenance de l'observation X^n à la classe C^k. Cette matrice établit une relation d'ordre floue et traduit l'idée d'une partition floue en K classes. La matrice d'appartenance U est également appelée matrice de partition floue et respecte les propriétés suivantes :

$$u_{kn} \in [0,1] \ \forall k \ \ \forall n \tag{III.1}$$

$$0 < \sum_{n=1}^{N} u_{kn} < N \ \ \forall k \tag{III.2}$$

ainsi que la relation suivante qui traduit une condition de normalisation :

$$\sum_{k=1}^{K} u_{kn} = 1 \ \ \forall n \tag{III.3}$$

Toutefois, avec ce concept de partition floue, une certaine ambiguïté subsiste en raison de l'amalgame entre les degrés d'appartenance et les probabilités. En effet, la condition de normalisation (III.3) est d'inspiration probabiliste. Cependant, bien que cette approche de la définition d'une partition floue est la plus utilisée, différents auteurs [KRI93][PED96][KHO97] présentent des approches où la contrainte probabiliste est levée. Il s'agit d'approches possibilistes de la définition d'une partition floue (la distinction entre les probabilités et les possibilités a été introduite au §3.1 du chapitre 2).

Ainsi, dans les paragraphes suivants, un ensemble d'algorithmes de coalescence floue est présenté. Parmi ceux-ci, le plus connu est l'algorithme des C-Moyennes floues, en anglais ***Fuzzy C-Means*** (FCM). Deux extensions de celui-ci sont proposées, une version possibiliste et une version permettant la détection des formes propres à chaque classe. Toutefois, ces premiers algorithmes de coalescence floue nécessitent une phase de validation pour le choix du nombre optimal de classes. Une deuxième partie est alors consacrée à un algorithme de coalescence floue sans connaissance *a priori* du nombre de classes : l'algorithme ***Unsupervised Robust C-Prototypes*** (URCP). Le FCM et ses extensions tout comme l'URCP n'étant pas approprié pour la caractérisation de classes de forme complexe (voir l'exemple de l'approche multiprototype illustrée par la figure II.8), une dernière partie est dédiée à la présentation d'une méthode de coalescence permettant de pallier cet inconvénient : l'algorithme ***Unsupervised Fuzzy Graph Clustering*** (UFGC).

1.1. Algorithmes classiques de coalescence floue

Les algorithmes de coalescence floue visent à réaliser une classification d'un ensemble de données en établissant une partition floue des observations en un nombre donné de classes. Comme indiqué en introduction, l'algorithme Fuzzy C-Means (FCM), issu des travaux de [DUN74], est de loin le plus connu. Il a d'ailleurs été amélioré par [BEZ81a] et constitue une référence parmi les méthodes de coalescence floue. Toutefois, cet algorithme présente quelques inconvénients, ce qui a donné naissance à plusieurs variantes. Ainsi, dans cette partie, l'***algorithme FCM*** est décrit ainsi que deux de ses extensions. La première est une variante possibiliste, connue sous le nom d'***algorithme PCM***, qui autorise l'introduction de la notion de rejet d'appartenance et donc la caractérisation des observations éloignées. La seconde est une version connue sous le nom d'***algorithme FCMGK*** et utilisée pour la détection des formes propres à chacune des classes. Enfin, étant donné que ces algorithmes nécessite le choix *a priori* du nombre de classes, un dernier paragraphe est consacré à la présentation de quelques ***critères de validation d'une partition***.

1.1.1. Algorithme des C-Moyennes floues

L'algorithme FCM réalise une partition floue en K classes d'un ensemble X de N observations de dimension D. L'obtention de cette partition est établie selon un principe itératif par minimisation d'un critère J_{FCM} permettant d'obtenir des classes homogènes et les plus distinctes possibles :

$$J_{FCM}(U,G)=\sum_{k=1}^{K}\sum_{n=1}^{N}\left((u_{kn})^m.d_A^2\left(X^n,G^k\right)\right) \qquad \text{(III.4)}$$

où U est la matrice de partition floue (dimension $K\times N$),

$G=\{G^1,...,G^k,...,G^K\}$ avec G^k centre du modèle de la classe C^k,

$u_{kn}\in[0,1]$ définit le coefficient d'appartenance de X^n à C^k,

$m\in[1,\infty[$ désigne le degré de flou de la partition,

$d_A^2\left(X^n,G^k\right)=d_{A,kn}^2=\left(X^n-G^k\right)^T.A.\left(X^n-G^k\right)$ définit la mesure de distance entre l'observation X^n et le centre G^k au sens de la métrique A (dimension $D\times D$),

La minimisation du critère J_{FCM} en fonction des centres G^k des modèles des différentes classes permet de définir la relation de mise à jour suivante (voir annexe 1) :

$$G^k=\frac{\sum_{n=1}^{N}\left((u_{kn})^m.X^n\right)}{\sum_{n=1}^{N}(u_{kn})^m}\quad\forall k\in\{1,2,...,K\} \qquad \text{(III.5)}$$

De même, la minimisation du critère J_{FCM} en fonction des coefficients de la matrice de partition floue U permet de définir la relation (III.6) permettant la mise à jour des coefficients u_{kn}. En fait, l'expression (III.6) est obtenue par la ***méthode des multiplicateurs de Lagrange*** [BEZ80] pour tenir compte de la contrainte de normalisation probabiliste (III.3) (voir annexe 1) :

$$u_{kn}=\frac{1}{\sum_{j=1}^{K}\left(\frac{d_{A,kn}^2}{d_{A,jn}^2}\right)^{\frac{1}{m-1}}}\quad\forall k\in\{1,2,...,K\}\text{ et }\forall n\in\{1,2,...,N\} \qquad \text{(III.6)}$$

Le processus itératif s'arrête lorsque la partition devient stable, c'est-à-dire lorsqu'elle n'évolue plus entre deux itérations successives. Ceci se traduit de manière générale par la vérification de l'expression (III.7) où le terme de gauche traduit une norme matricielle et le coefficient ξ définit le seuil de convergence :

$$\left\|U^{it+1}-U^{it}\right\|<\xi \qquad \text{(III.7)}$$

Le processus d'apprentissage aboutit à la définition de la matrice de partition U en K classes et des différents centres G^k. Dans ce sens, la figure III.1 illustre les modèles de classes établis à partir de la partition obtenue avec l'algorithme FCM ($K = 3$ et $m = 2$) en présence d'un nuage d'observations comportant trois classes.

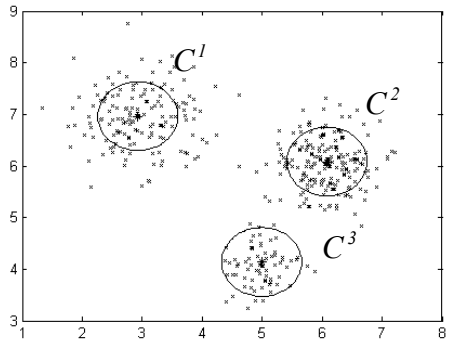


Figure III.1 : Illustration des modèles de classes représentés par leurs centres et des courbes de niveau d'appartenance établis à partir de la partition obtenue par l'algorithme FCM.

Algorithme FCM

Définir la métrique A *,*
Fixer le nombre K *de classes,*
Fixer le degré de flou m *,*
Initialiser la matrice U *de partition,*
Initialiser les centres G^k *des classes,*
REPETER
    Mettre à jour les centres G^k *des classes (Eq.(III.5))*
    Mettre à jour la matrice U *de partition (Eq.(III.6))*
JUSQU'À obtenir la stabilité de U *(Eq.(III.7))*

L'***algorithme FCM*** réalise une partition floue d'un ensemble d'observations en un nombre connu *a priori* de classes. Bien que la convergence de cette méthode soit démontrée [BEZ80], le résultat obtenu dépend de l'initialisation des centres des classes et du choix de la métrique employée. De plus, en fonction du choix de K, la partition obtenue n'est pas forcément optimale. Ceci nécessite de réitérer plusieurs fois l'algorithme avec des valeurs différentes, puis de sélectionner ultérieurement la meilleure partition au cours d'une phase de validation (§1.1.4).

De plus, l'introduction de la contrainte de normalisation probabiliste pose problème pour la caractérisation des observations isolées. En effet, cette condition traduit le fait que le coefficient d'appartenance maximal d'une observation ne peut être inférieur à $1/K$ or les seuils de rejet en appartenance (voir §3.3.2. du chapitre 2) sont généralement inférieurs à cette grandeur.

1.1.2. Algorithme possibiliste des C-moyennes

[KRI93] propose une solution au problème des observations isolées en s'appuyant sur une hypothèse possibiliste pour la définition de la partition. En fait, il propose de lever la contrainte de normalisation probabiliste, ce qu'il réalise au travers de l'algorithme Possibiliste des C-

Moyennes (PCM). Les coefficients d'appartenance s'interprètent comme des possibilités d'appartenance et rendent compte de la typicité d'une observation aux classes. La possibilité d'appartenance d'une observation à une classe est d'autant plus grande que celle-ci est proche du centre de la classe. En fait, cette approche a l'avantage de considérer l'appartenance d'une observation à une classe indépendamment de son appartenance aux autres classes. Le critère d'optimisation de l'algorithme PCM s'exprime par :

$$J_{PCM}(U,G) = \sum_{k=1}^{K}\sum_{n=1}^{N}\left((u_{kn})^m . d_A^2\left(X^n, G^k\right)\right) + \sum_{k=1}^{K}\left(\eta_k \cdot \sum_{n=1}^{N}(1-u_{kn})^m\right) \qquad \text{(III.8)}$$

en considérant seulement les propriétés (III.1) et (III.2) de la définition d'une partition floue donnée par [RUS69] et où la contrainte de normalisation est remplacée par :

$$max_{k=1,K}(u_{kn}) > 0 \quad \forall n$$

Le premier terme du critère J_{PCM} correspond au critère de l'algorithme FCM. Le second vise à imposer aux possibilités d'appartenance des valeurs les plus grandes possibles. Par ailleurs, le terme η_k pondère chaque classe C^k et est déterminé selon l'expression suivante :

$$\eta_k = \frac{\sum_{n=1}^{N}\left((u_{kn})^m . d_{A,kn}^2\right)}{\sum_{n=1}^{N}(u_{kn})^m} \quad \forall k \in \{1,2,...,K\} \qquad \text{(III.9)}$$

Le processus de minimisation du critère est toujours itératif. L'algorithme PCM est initialisé à partir des résultats du FCM. La relation de mise à jour des centres G^k des classes s'exprime comme pour l'algorithme FCM avec l'expression (III.5). Par contre, la relation de mise à jour de la partition floue U prend la forme suivante :

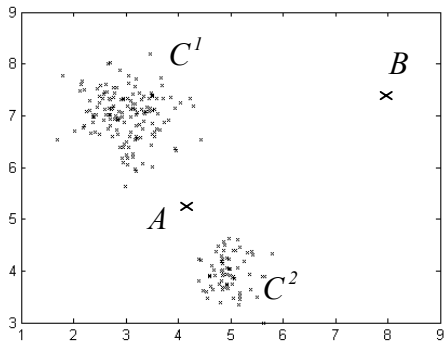
$$u_{kn} = \frac{1}{1+\left(\frac{d_{A,kn}^2}{\eta_k}\right)^{\frac{1}{m-1}}} \quad \forall k \in \{1,2,...,K\} \text{ et } \forall n \in \{1,2,...,N\} \qquad \text{(III.10)}$$

Algorithme PCM

Définir la métrique A,
Fixer le nombre de classes K,
Fixer le degré de flou m,
Initialiser la matrice U de partition et les centres G^k des classes avec l'algorithme FCM,
Calculer η_k pour chaque classe C^k (Eq.(III.9))
REPETER
    Mettre à jour les centres G^k des classes (Eq.(III.5))
    Mettre à jour la matrice U de partition (Eq.(III.10))
JUSQU'À obtenir la stabilité de U (Eq.(III.7))

Selon cette approche possibiliste, une observation éloignée de toutes les classes voit ses possibilités d'appartenance aux classes prendre des valeurs très faibles. En effet, par opposition à

l'algorithme FCM, la caractérisation d'une observation éloignée est envisageable grâce à la définition de coefficients d'appartenance de nature possibiliste, rendant possible l'introduction des notions de rejet (figure III.2). De même, cet algorithme a l'avantage de permettre la classification au sein d'un nuage bruité sans que les observations bruitées perturbent la définition des modèles de classes réelles.



		u_{FCM}	u_{PCM}
A	C^1	0.5	0.3
	C^2	0.5	0.3
B	C^1	0.5	0.01
	C^2	0.5	0.02

Figure III.2 : Caractérisation d'un point B éloigné par rapport à un point A ambigu par l'algorithme PCM par opposition à l'algorithme FCM.

L'***algorithme PCM*** permet la prise en compte des observations éloignées en leur associant une possibilité d'appartenance plus faible que le coefficient d'appartenance de nature probabiliste défini par l'algorithme FCM. D'autres algorithmes ont été développés afin d'améliorer la robustesse des algorithmes de coalescence floue aux points éloignés. Pour sa part, [OHA84] considère une classe de points éloignés (classe de rejet) et modifie en conséquence la définition du critère d'optimisation.

Malheureusement, tout comme le FCM, l'algorithme PCM souffre de quelques inconvénients. En effet, bien que le choix de la métrique A soit libre, celle-ci impose un modèle unique pour la forme des classes de la partition et ne permet pas de définir un modèle propre à chaque classe, tout comme elle ne permet pas de caractériser des classes de forme complexe. L'utilisation de ces algorithmes est restreinte à la caractérisation de classes de forme hypersphérique et hyperelliptique ayant la même orientation.

1.1.3. Algorithme FCMGK de Gustafson et Kessel

L'extension de l'algorithme FCM proposée par Gustafson et Kessel (FCMGK) a l'avantage de tenir compte de la forme propre à chacune des classes [GUS79]. L'algorithme FCMGK cherche à reconnaître la forme de chaque classe plutôt que de définir un modèle unique (hypersphérique ou hyperelliptique) pour toutes les classes. Pour cela, pour chaque classe C^k, la mesure de distance est énoncée à l'aide d'une métrique A_k :

$$d^2_{A_k}\left(X^n, G^k\right) = \left(X^n - G^k\right)^T . A_k . \left(X^n - G^k\right) \qquad \text{(III.11)}$$

où A_k est une matrice propre à la classe C^k et pour laquelle [GUS79] montre que la minimisation du critère d'optimisation conduit à la valeur de A_k suivante :

$$A_k = (\delta_k . det(\Sigma_k))^{1/D} . \Sigma_k^{-1} \quad \forall k \in \{1, 2, ..., K\} \tag{III.12}$$

où Σ_k est la matrice de covariance floue de la classe C^k et s'exprime par :

$$\Sigma_k = \frac{S_k}{\sum_{n=1}^{N} (u_{kn})^m} \tag{III.13}$$

avec S_k la matrice de dispersion floue de la classe C^k :

$$S_k = \sum_{n=1}^{N} \left((u_{kn})^m . (X^n - G^k)(X^n - G^k)^T \right) \tag{III.14}$$

Le critère d'optimisation J_{FCMGK} de l'algorithme FCMGK s'exprime de la même façon que celui de l'algorithme FCM. De même, les relations de mise à jour des coefficients d'appartenance et des centres des classes conservent l'écriture des expressions (III.5) et (III.6). Toutefois, la distance est désormais calculée à partir de la relation (III.11) faisant appel à la matrice A_k. Par ailleurs, dans l'expression (III.12), le terme δ_k exprime une contrainte sur le volume de la classe C^k permettant d'optimiser la forme de celle-ci [GUS79] :

$$\delta_k = det(A_k) = Cste \quad pour\ chaque\ classe\ C^k \tag{III.15}$$

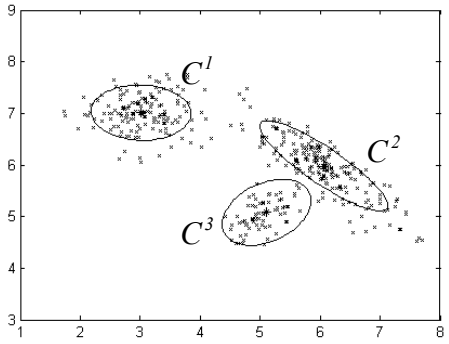


Figure III.3 : Illustration des modèles de classes représentés par leurs centres et des courbes de niveau d'appartenance établis à partir de la partition obtenue par l'algorithme FCMGK.

La figure III.3 illustre l'aptitude de l'algorithme FCMGK à modéliser les différentes classes en reconnaissant pour chacune d'entre elles leur forme et leur orientation.

Algorithme FCMGK

Fixer le nombre de classes K *,*
Fixer le degré de flou m *,*
Initialiser la matrice de partition U *et les centres* G^k *des classes avec l'algorithme FCM,*
Fixer une valeur de δ_k *pour chaque classe* C^k *,*
REPETER
    Mettre à jour les centres G^k *des classes (Eq.(III.5))*
    Mettre à jour les matrices A_k *des classes (Eq.(III.12))*

> *Mettre à jour la matrice* U *de partition (Eq.(III.6) en utilisant la distance* d_{A_k}
> *JUSQU'À obtenir la stabilité de* U *(Eq.(III.7))*

L'***algorithme FCMGK*** permet la caractérisation de classes de formes et orientations différentes au sein d'un nuage de points grâce à l'utilisation d'une métrique propre à chacune des classes. Toutefois, cette approche reste limitée aux classes de forme hyperelliptique sans autoriser la caractérisation de classes de forme complexe (non convexe). De plus, comme les algorithmes précédents, la difficulté sur le choix de K est toujours présente.

1.1.4. Validation d'une partition de données

Le développement des méthodes de coalescence floue a entraîné la nécessité de définir des critères permettant de valider une partition floue U en K classes d'un ensemble X de N observations. Ces critères visent à définir le nombre optimal K^* de classes en sélectionnant la partition U^* en K^* classes qui optimise le critère sélectionné.

Critères	*Calcul des critères*	*Règles d'utilisation*
Coefficient de partition	$C_{CP}(U,K)=\frac{1}{N}\sum_{n=1}^{N}\sum_{k=1}^{K}(u_{kn})^2$	$(U^*,K^*)=max_{K=2,\ldots,N-1}(C_{CP}(U,K))$
Entropie de classification	$C_{EC}(U,K)=\frac{1}{N}\sum_{n=1}^{N}\sum_{k=1}^{K}\mu_{kn}\,log(u_{kn})$	$(U^*,K^*)=min_{K=2,\ldots,N-1}(C_{EC}(U,K))$
Exposant de proportion	$C_{EP}(U,K)=\frac{1}{N}\sum_{n=1}^{N}max_{k=1,\ldots,K}(u_{kn})$	$(U^*,K^*)=max_{K=2,\ldots,N-1}(C_{EP}(U,K))$
Hypervolume flou	$C_{HF}(U,K)=\sum_{k=1}^{K}(det(\Sigma_k))^{1/2}$	$(U^*,K^*)=min_{K=2,\ldots,N-1}(C_{HF}(U,K))$

Table III.1 : Critères de validation d'une partition.

La table III.1 illustre quatre critères couramment utilisés pour la validation d'une partition. Chaque critère est supposé atteindre sa valeur optimale lorsque la valeur de K correspond au nombre de classes réelles. En fait, la recherche d'une partition optimale n'est pas du tout chose facile puisque, bien que disposant du meilleur critère possible, il reste un bon nombre de partitions envisageables.

Cette première partie a permis de présenter l'algorithme FCM et deux de ses extensions. D'***autres variantes de l'algorithme FCM*** ont été proposées dans la littérature, notamment pour prendre en compte des classes de formes plus variées. Dans ce sens, l'algorithme ***Fuzzy C-Varietes*** (FCV) n'utilise plus les centres des classes mais des variétés linéaires d'ordre r avec $0 \le r \le D$ [BEZ81a]. Les lignes, les plans ou hyperplans représentant les classes permettent alors

de définir des classes allongées et inclinées en utilisant une distance orthogonale entre une observation et la variété de la classe considérée. Malheureusement, en présence de classes colinéaires même bien séparées, l'algorithme FCV échoue. Ce problème est résolu par l'algorithme ***Fuzzy C-Elliptotypes*** (FCE) [BEZ81b] qui utilise une distance combinant la distance entre une observation et le centre de la classe considérée, et la distance entre cette même observation et la variété linéaire caractérisant la classe.

L'inconvénient majeur des différents algorithmes de coalescence floue présentés ici provient du fait qu'ils nécessitent tous une phase de validation. En effet, la partition obtenue dépend directement du choix de K. La phase de validation permet de sélectionner, parmi différentes partitions en des nombres différents de classes, celle qui semble optimale au sens d'un critère de validité. Afin de pallier cette contrainte, d'autres algorithmes ont été proposés. Ainsi, des auteurs ont présenté des approches où K vaut 1 initialement et croit au cours des itérations, la valeur optimale étant définie par un indice de performance [WON01]. Dans la partie suivante, nous verrons que d'autres auteurs ont préféré une approche où l'on surestime le nombre de classes *a priori* puis l'algorithme tend vers la valeur optimale lors de son processus itératif.

1.2. Algorithme de coalescence floue sans connaissance *a priori* de K

Afin de pallier la difficulté quant au choix du nombre de classes *a priori*, Frigui et Krishnapuram ont proposé l'algorithme ***Unsupervised Robust C-Prototypes*** (URCP) [FRI96]. La première étape de cet algorithme consiste à définir une partition floue des données en un nombre K_p de classes avec au moins $K_p > K$ où K est le nombre réel de classes. La seconde étape permet de fusionner certaines classes sur la base d'une mesure de similarité entre classes. En fait, le processus d'apprentissage de l'algorithme URCP est construit à partir de deux algorithmes : le ***Robust C-Prototypes*** (RCP) et l'***Unconstrained Robust C-Prototypes*** (UcRCP)**.**

1.2.1. Algorithme Robust C-Prototypes

L'***algorithme Robust C-Prototypes*** (RCP), introduit par [FRI95], a été défini pour caractériser des classes dans un nuage bruité d'observations, c'est-à-dire contenant des observations ambiguës et/ou isolées. Le principe global d'apprentissage du RCP est similaire aux algorithmes classiques de coalescence floue. La particularité du RCP provient du fait qu'il utilise simultanément deux ensembles de poids ayant deux interprétations en terme d'appartenance aux classes. Les premiers sont des coefficients d'appartenance u_{kn} (approche probabiliste) et traduisent le degré d'appartenance des observations aux classes. Les seconds sont des possibilités d'appartenance w_{kn} (approche possibiliste) et dénotent de la typicité des observations aux classes. Les coefficients u_{kn} permettent de forcer la création de K classes au sein du nuage d'observations alors que les coefficients w_{kn} visent à améliorer la modélisation de celles-ci en conférant aux observations éloignées une pondération nulle $w_{kn} = 0$. Pour cela, le critère d'optimisation J_{RCP} (III.16) est défini en utilisant une fonction de perte ρ_k propre à chaque classe C^k (voir annexe 2) :

$$J_{RCP}(U,G)=\sum_{k=1}^{K}\sum_{n=1}^{N}\left((u_{kn})^{m}.\rho_k\left(d^2\left(X^n,G^k\right)\right)\right) \qquad \text{(III.16)}$$

avec $u_{kn}\in[0,1]\ \forall k\ \ \forall n$ et la contrainte $\sum_{k=1}^{K}u_{kn}=1\ \ \forall n\in\{1,2,...,N\}$.

La technique des multiplicateurs de Lagrange permet d'obtenir la relation de mise à jour des coefficients d'appartenance minimisant J_{RCP} selon U (voir annexe 2) :

$$u_{kn}=\frac{1}{\sum_{j=1}^{K}\left(\frac{\rho_k\left(d_{kn}^2\right)}{\rho_j\left(d_{jn}^2\right)}\right)^{\frac{1}{m-1}}}\quad \forall k\in\{1,2,...,K\}\ \text{et}\ \forall n\in\{1,2,...,N\} \qquad \text{(III.17)}$$

La minimisation du critère J_{RCP} s'effectue également en fonction des paramètres β_k des classes. L'écriture des relations d'optimisation de ces paramètres nécessite de choisir une structure de modèle, i.e. d'identifier les paramètres des classes. Pour une classe C^k de forme hypersphérique, les paramètres β_k se résument au centre G^k de la classe, le volume étant interprété au sens de la distance euclidienne. Par contre, pour une classe C^k de forme hyperelliptique, les paramètres β_k sont définis par le centre G^k de la classe et une matrice A_k définissant la métrique propre à la classe C^k. Sur ce principe, la minimisation du critère J_{RCP} en fonction des paramètres β_k du modèle de la classe C^k permet de définir les relations de mise à jour des paramètres β_k des différents modèles de classes (voir annexe 2).

Par ailleurs, la définition d'une fonction ρ_k (différentes fonctions sont proposées dans [FRI97], voir annexe 2) pour chaque classe C^k permet d'adjoindre à la définition du modèle de cette dernière un ensemble de pondérations w_{kn} de nature possibiliste. Ces pondérations w_{kn} sont réunies au sein d'une matrice W et sont mises à jour en même temps que les paramètres des classes selon l'expression suivante :

$$w_{kn}=\frac{\partial\rho_k\left(d_{kn}^2\right)}{\partial d_{kn}^2}\quad \forall k\in\{1,2,...,K\}\ \text{et}\ \forall n\in\{1,2,...,N\} \qquad \text{(III.18)}$$

Pour la détection de classes hyperelliptiques, l'utilisation de la matrice A_k nécessite la même contrainte que pour l'algorithme FCMGK, i.e. s'assurer que le terme $\delta_k=det(A_k)$ reste constant. La mise à jour des centres G^k des classes s'effectue alors selon l'expression :

$$G^k=\frac{\sum_{n=1}^{N}\left((u_{kn})^{m}.w_{kn}.X^n\right)}{\sum_{n=1}^{N}(u_{kn})^{m}.w_{kn}}\quad \forall k\in\{1,2,...,K\} \qquad \text{(III.19)}$$

La relation de mise à jour des matrices A_k s'obtient quant à elle par la technique des multiplicateurs de Lagrange en tenant compte de la contrainte $\delta_k = det(A_k) = Cste$ sur chacune des classes C^k et s'écrit sous la forme suivante :

$$A_k = (\delta_k . det(\Sigma R_k))^{\frac{1}{d}} \Sigma R_k^{-1} \tag{III.20}$$

où la matrice ΣR_k définit la matrice de covariance floue robuste de la classe C^k :

$$\sum R_k = \frac{\sum_{n=1}^{N} \left((u_{kn})^2 . w_{kn} . (X^n - G^k)(X^n - G^k)^T \right)}{\sum_{n=1}^{N} \left((u_{kn})^m . w_{kn} \right)} \tag{III.21}$$

Comme l'indique les expressions (III.19)(III.20)(III.21), la mise à jour itérative des paramètres $\beta_k = \{G^k, A_k\}$ des différents modèles de classes dépend à la fois des coefficients d'appartenance u_{kn} et des possibilités d'appartenance w_{kn}.

L'utilisation simultanée des deux pondérations permet de définir correctement les paramètres des classes tout en identifiant les observations parasites (bruit, …). L'introduction de la fonction ρ_k a pour effet de donner aux classes une connotation possibiliste en associant aux observations X^n éloignées de la classe C^k une possibilité d'appartenance w_{kn} proche de 0, voire nulle. Hélas, l'utilisation de l'algorithme RCP nécessite toujours la connaissance *a priori* du nombre K de classes, et c'est pourquoi [FRI97] propose d'utiliser cet algorithme en spécifiant un nombre $K_p > K$ de classes.

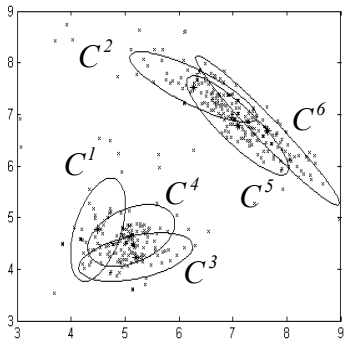


Figure III.4 : Partition obtenue après plusieurs itérations de l'algorithme RCP pour un nombre K_p spécifié de 6 classes.

Dans ce sens, la figure III.4 illustre le résultat obtenu avec l'algorithme RCP pour la définition d'une partition en $K_p = 6$ classes au sein d'un nuage bruité d'observations. Toutefois, le fait de surestimer le nombre de classes provoque un éclatement des régions denses en plusieurs classes. Cette séparation de classes réelles en plusieurs petites classes est due à la contrainte de normalisation dont l'objectif est justement de définir une partition en un nombre spécifié de classes. De même, des classes « parasites » peuvent apparaître dans des régions bruitées de l'espace de représentation.

1.2.2. Algorithme Unconstrained Robust C-Prototypes

Le fait de surestimer le nombre de classes lors de l'utilisation du RCP a pour effet de séparer certaines classes réelles en plusieurs petites classes. Pour pallier cette difficulté, il existe une version de l'algorithme RCP, l'algorithme ***Unconstrained Robust C-Prototypes*** (UcRCP), où la contrainte de normalisation est levée [FRI97]. Cette opération a pour objet de permettre aux modèles des petites classes de s'étendre pour couvrir toute une région dense. Il suffit d'ignorer les coefficients d'appartenance ($u_{kn} = 1 \ \forall k \ \forall n$) et donc d'optimiser le critère réduit suivant :

$$J_{Unconstrained\ RCP}(U,G) = \sum_{k=1}^{K}\sum_{n=1}^{N} \rho_k \left(d^2\left(X^n, G^k\right)\right) \qquad \text{(III.22)}$$

Les relations de mise à jour des paramètres des classes sont alors les mêmes que pour l'algorithme RCP mais en considérant $u_{kn} = 1 \ \forall k \ \forall n$. L'utilisation de l'algorithme UcRCP permet d'améliorer la partition obtenue avec le RCP. En effet, l'application de cet algorithme permet, d'une part, de faire migrer les classes parasites vers des régions plus denses, et d'autre part, d'accroître la taille des classes adjacentes de sorte que chacune d'entre elles couvre toute une région dense. Ainsi, l'application de cet algorithme a pour effet de tendre vers l'obtention d'une partition où plusieurs classes modélisent les mêmes régions denses.

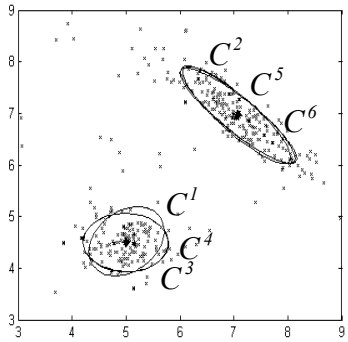


Figure III.5 : Partition obtenue après quelques itérations de l'algorithme UcRCP pour un nombre K_p spécifié de 6 classes.

La figure III.5 illustre la partition en $K_p = 6$ classes obtenue après plusieurs itérations de l'algorithme UcRCP initialisé avec le résultat de l'algorithme RCP (figure III.4). Les différentes classes représentant les deux régions denses se sont étendues. Chaque région dense est désormais caractérisée par 3 classes « similaires ». Toutefois, il est important de noter que si l'expansion des classes n'est pas contrôlée, certaines classes distinctes risquent d'être caractérisées par une seule et même classe. A cet effet, et comme l'indique [FRI97], l'expansion peut être contrôlée puis limitée lors de l'adaptation des fonctions ρ_k après chaque itération (voir annexe 2).

1.2.3. Algorithme Unsupervised Robust C-Prototypes

L'algorithme ***Unsupervised Robust C-Prototypes*** (URCP) a été introduit en 1996 par [FRI96] dans le but de réaliser une coalescence floue sans connaître *a priori* le nombre K de classes d'un nuage d'observations. L'URCP est un algorithme de nature itérative qui optimise le nombre de

classes de la partition en fusionnant les classes similaires. Son principe consiste à rechercher dans un premier temps une partition en un nombre K_p de classes avec $K_p > K$ à l'aide de l'algorithme RCP. Dans un second temps, l'algorithme UcRCP est utilisé pour favoriser l'extension des classes afin qu'elles caractérisent l'intégralité des régions denses. L'algorithme URCP fusionne alors les classes similaires pour tendre vers le nombre K optimal de classes.

Dans ce sens, l'algorithme URCP utilise une mesure de similarité floue s_{ij} définie pour chaque couple de classes $\left(C^i, C^j\right)$ et exprimée par :

$$s_{ij} = 1 - \frac{\sum_{X^n \in EW_{ij}} \left| u_{in} - u_{jn} \right|}{\left| EW_i \right| + \left| EW_j \right|} \qquad \text{(III.23)}$$

où les notations utilisées sont définies par :

$EW_i = \left\{ X^n / w_{in} > 0 \right\}$ est l'ensemble des observations qualifiées de typiques pour la définition de la classe C^i, i.e. non éloignées de celle-ci,

$EW_{ij} = \left\{ X^n / w_{in} > 0 \ \ ou \ \ w_{jn} > 0 \right\}$ est l'ensemble des observations qualifiées de typiques pour la définition de C^i ou de C^j,

$\left| EW_i \right| = \sum_{X^n \in E_i} u_{in}$ est la cardinalité robuste de l'ensemble EW_i.

La comparaison des mesures de similarité floue entre les différents couples de classes à un seuil de similarité s_{min} permet de fusionner les classes similaires au fur et à mesure des itérations de l'algorithme URCP.

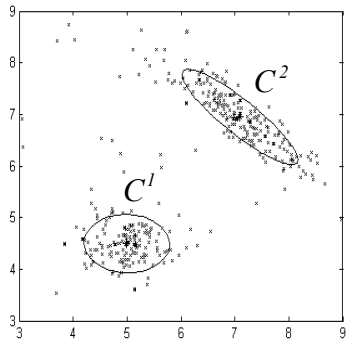


Figure III.6 : Partition obtenue après plusieurs itérations de l'algorithme URCP pour un nombre K_p spécifié de 6 classes.

La figure III.6 présente la partition obtenue avec l'URCP sur la base des données utilisées précédemment et en fixant $s_{min} = 0.5$. En fait, le processus d'apprentissage itératif de l'algorithme peut être résumé par la chronologie des figures III.4, III.5 et III.6.

Algorithme Unsupervised RCP

> *Fixer le nombre de classes* $K_p > K$,
> *Fixer le degré de flou* m ,

Fixer le seuil de similarité s_{min} ,
Initialiser la matrice U *de partition et les paramètres* β_k *des classes,*
Initialiser les fonctions ρ_k ,
REPETER
    Utiliser l'algorithme RCP,
    Utiliser l'algorithme Unconstrained RCP ($u_{kn} = 1$*),*
    Mettre à jour la matrice U *de partition (Eq.(III.17)),*
    POUR chaque paire de classes C^i *et* C^j ,
        Calculer la similarité s_{ij} *(Eq.(III.23)),*
        SI $s_{ij} > s_{min}$
        ALORS Fusionner les classes C^i *et* C^j ,
    FIN
    Mettre à jour le nombre de classes K_p
JUSQU'À Plus de fusions réalisées

L'***algorithme Unsupervised Robust C-Prototypes*** permet d'obtenir une partition floue dans un nuage bruité d'observations sans précision sur le nombre de classes. Toutefois, l'algorithme URCP souffre de quelques inconvénients : le nombre initial de classes doit être supérieur au nombre réel de classes ce qu'il n'est pas évident d'assurer; il n'existe pas de méthodes particulières pour fixer le seuil de similarité floue pour la fusion; et enfin comme pour les algorithmes de coalescence floue présentés au paragraphe précédent, l'utilisation est restreinte aux classes de forme hyperelliptique, aucune solution n'est apportée aux problèmes de caractérisation des classes de forme complexe.

1.3. Algorithme de coalescence floue pour les classes de forme complexe

Pour combler les lacunes des algorithmes de coalescence en présence de classes de forme complexe, certains auteurs ont développé d'autres méthodes de classification. De manière générale, le principe de ces dernières consiste à associer aux techniques de coalescence floue des méthodes hiérarchiques qui produisent une suite de partitions emboîtées représentables sous la forme d'un graphe. Deux approches de ce type sont présentées ici : l'***algorithme Unsupervised Graph Clustering*** et l'***algorithme Unsupervised Fuzzy Graph Clustering***.

1.3.1. Algorithme Unsupervised Graph Clustering

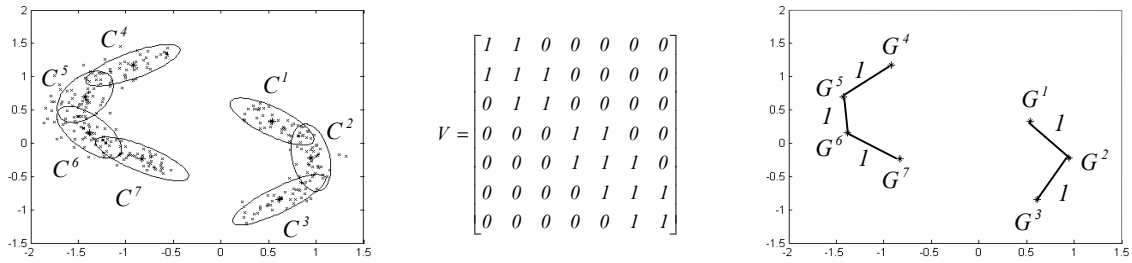
L'algorithme ***Unsupervised Graph Clustering*** (UGC) proposé par [BIL98] permet de résoudre les problèmes de caractérisation d'une structure de classes de forme complexe au sein d'un nuage de points, sans connaissance *a priori* du nombre de classes. En fait, la première étape vise à décomposer le nuage d'observations en un nombre K_p important de sous-classes avec un algorithme classique de coalescence floue. En fait, il s'agit de caractériser des classes de forme complexe selon une approche multiprototype (terminologie introduite au chapitre 2), bien que [BIL98] qualifie les prototypes de sous-classes. Par la suite, nous conservons cependant la terminologie de [BIL98] en notant qu'une sous-classe n'est autre qu'une classe de forme hyperelliptique assimilable à un prototype.

Le processus d'apprentissage de l'algorithme UGC débute par la définition d'une matrice de voisinage V ayant pour objet d'identifier les sous-classes connexes (voisines). Deux sous-classes C^i *et* C^j sont dites connexes s'il existe une observation ambiguë entre celles-ci. L'écriture de la matrice de voisinage s'établit alors en utilisant un ***indice de voisinage booléen*** :

$$v_{ij} = \begin{cases} 1 & si \quad \exists X^n / \left| u_{in} - u_{jn} \right| < S_{amb} \\ 0 & sinon \end{cases} \qquad \text{(III.24)}$$

où S_{amb} est un seuil d'ambiguïté.

La matrice de voisinage étant écrite, un ***graphe de voisinage*** entre les sous-classes est déterminé. Les sommets et les arcs de celui-ci caractérisent respectivement les centres des sous-classes et l'existence de zones de voisinage entre les sous-classes, une zone de voisinage étant définie par l'existence de sous-classes connexes. Les arcs sont pondérés par les coefficients de la matrice de voisinage



a) *b)* *c)*

Figure III.7 : Résultats de l'algorithme UGC : a) Définition d'une partition en 7 sous-classes, b) Matrice de voisinage définie selon l'indice de voisinage et c) Graphe de voisinage associé.

La figure III.7 illustre le résultat obtenu en appliquant l'algorithme UGC en présence d'un nuage de points constitué de deux classes clairement séparées. A partir d'une partition en $K_p = 7$ sous-classes (figure III.7.a) obtenue à l'aide d'un algorithme de coalescence floue, la matrice de voisinage est établie (figure III.7.b). Celle-ci permet de mettre en évidence une structure de classes au sein du nuage d'observations. En effet, au niveau du graphe de voisinage, cela se traduit par l'apparition de ***composantes connexes*** caractérisées par la présence de plusieurs sommets liés entre eux par des arcs de pondération unitaire (figure III.7.c).

La fusion des sous-classes présentes dans chaque ***composante connexe*** conduit alors à la reconstruction des classes complexes réelles. Les coefficients d'appartenance aux classes réelles sont obtenus par une somme bornée des coefficients des sous-classes connexes ce qui permet d'élaborer les modèles des différentes classes complexes au sens de l'approche multiprototype.

Ainsi, l'algorithme ***Unsupervised Graph Clustering*** permet de calculer une partition floue dans un nuage de points comportant des classes de forme complexe, sans connaissance *a priori* du nombre de classes. Toutefois, deux inconvénients résident lors de son utilisation. D'une part, l'indice de voisinage utilisé est de nature booléenne, ce qui rend l'algorithme très sensible au bruit et par conséquent suppose l'existence de classes clairement séparées; et d'autre part, la détermination de celui-ci nécessite le choix d'un seuil d'ambiguïté S_{amb}.

1.3.2. Algorithme Unsupervised Fuzzy Graph Clustering

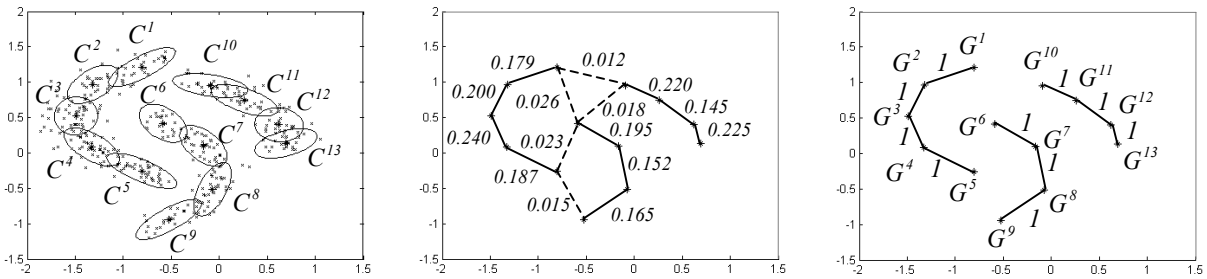
L'algorithme ***Unsupervised Fuzzy Graph Clustering*** (UFGC) a été développé par [DEV99] afin de pallier la difficulté à laquelle est confronté l'algorithme UGC lors de la présence de points isolés entre les classes réelles. En fait, l'inconvénient de l'algorithme UGC réside dans son principe de caractérisation des composantes connexes. Evidemment, une première solution à ce problème consiste à introduire un nombre n_{amb} de points ambigus dans sa définition, d'où l'écriture de l'***indice d'ambiguité.*** La valeur de l'indice d'ambiguité γ_{ij} entre les sous-classes C^i et C^j est définie selon (III.25) où la notion d'ambiguité d'une observation est identique au cas précédent :

$$\gamma_{ij} = \begin{cases} 1 - \dfrac{S_{amb}}{n_{amb}} \;\; si \;\; n_{amb} \geq 1 \\ 0 \;\; si \;\; n_{amb} = 0 \end{cases} \qquad \text{(III.25)}$$

Ainsi, pour un seuil d'ambiguité S_{amb} fixé, γ_{ij} est d'autant plus grand qu'il y a d'observations ambiguës entre les sous-classes C^i et C^j. Toutefois, l'évolution de cet indice reste assez brutale et c'est pourquoi on lui préfère souvent l'***indice de similarité floue*** :

$$s_{ij} = 1 - \frac{\sum_{X^n \in C^i \; ou \; X^n \in C^j} \left| u_{in} - u_{jn} \right|}{\sum_{X^n \in C^i} \left| u_{in} \right| + \sum_{X^n \in C^j} \left| u_{jn} \right|} \qquad \text{(III.26)}$$

Cet indice est similaire à celui défini par [FRI96], énoncé au §1.2.3, et a l'avantage par rapport à l'indice d'ambiguité de ne pas requérir le choix de seuils supplémentaires. Par ailleurs, par opposition à l'indice de voisinage booléen, il est défini sur $[0,1]$ d'où une meilleure interprétation du rapprochement entre les sous-classes.



a) *b)* *c)*

Figure III.8 : Résultats de l'algorithme UFGC : a) Définition d'une partition en 13 sous-classes, b) Graphe flou de proximité défini selon l'indice de similarité floue c) Graphe de voisinage associé.

L'expression de l'indice de similarité floue étant définie, la figure III.8 illustre les étapes d'apprentissage de l'algorithme UFGC. Comme pour l'algorithme UGC, la première étape consiste à définir une partition en un nombre $K_p > K$ de classes avec un algorithme classique de coalescence floue (figure III.8.a). L'utilisation de l'indice de similarité floue permet alors l'écriture de la ***matrice de proximité floue*** S avec l'ensemble des valeurs de similarités s_{ij}.

Celle-ci donne lieu à la construction d'un ***graphe flou de proximité*** où les poids des arcs sont définis par les valeurs de similarité floue (figure III.8.b). Pour des raisons de clarté, seuls les arcs intéressants ont été représentés. A partir de ce graphe, une hiérarchie est définie et indicée selon les valeurs de similarité floue. En fait, le rapprochement entre les modèles hiérarchiques et les relations d'ordre flou a été introduit dans les années 1980 et est présenté par [RUS98].

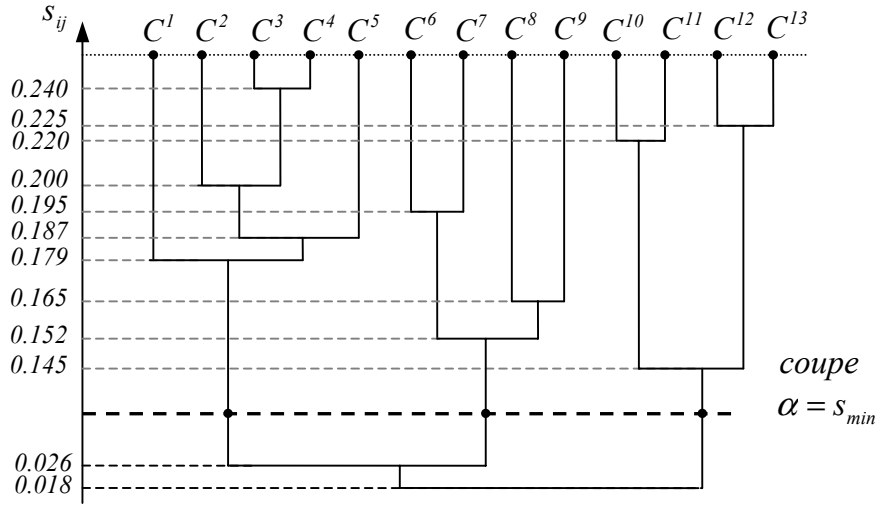


Figure III.9 : Hiérarchie établie à partir du graphe flou de proximité.

Dans le cas du graphe flou de proximité présenté à la figure III.8.b), la hiérarchie est définie par la figure III.9. Le retour aux classes réelles nécessite la réalisation d'une coupe à un niveau $\alpha = s_{min}$ (α-coupe) permettant d'écrire un graphe de voisinage ainsi que la matrice de voisinage V associée. En l'occurrence, pour notre exemple, celui-ci peut être obtenu pour une valeur de $s_{min} = 0.08$ (figure III.8.c). En fait, à chaque niveau de coupe correspond une partition floue du nuage de points. [DEV99] définit la coupe optimale pour la valeur minimale du critère K_z :

$$K_z = \left| 1 - \frac{\frac{1}{N}\sum_{k=1}^{K}\sum_{n=1}^{N}(u_{kn}.d_{kn})^2}{\frac{1}{K}\sum_{k=1}^{K}\frac{\sum_{n=1}^{N}(u_{kn}.d_{kn})^2}{\sum_{n=1}^{N}u_{kn}}} \right| \qquad \text{(III.27)}$$

L'écriture de ce critère est basée sur l'hypothèse que le rapport de la compacité globale du nuage de points sur la compacité moyenne (somme des compacités par classe) tend vers 1 pour la valeur optimale du nombre de classes.

Algorithme UFGC

Fixer le nombre de sous-classes $K_p > K$,
Fixer le degré de flou m ,
Déterminer une partition U *en* K_p *sous-classes,*
Déterminer la matrice de proximité flou S *(Eq.(III.26)),*

z=1
POUR s_{min} *allant de* $min(S)$ *à* $max(S)$,
    POUR *i=1 à* K_p,
        POUR *j=1 à* K_p,
            SI $s_{ij} > s_{min}$
            ALORS $v_{ij} = 1$
            SINON $v_{ij} = 0$
        FIN
    FIN
    Définir le graphe de voisinage et la matrice de voisinage V,
    Fusionner les sous-classes connexes,
    Mettre à jour la matrice de partition U,
    Calculer le critère K_z *(Eq.(III.27))*
    z=z+1
FIN
Déterminer la coupe optimale pour la valeur minimale de K_z

L'algorithme ***Unsupervised Fuzzy Graph Clustering*** permet d'obtenir une partition floue dans un nuage bruité d'observations sans précision sur le nombre de classes et en présence de classes de forme complexe. De plus, par opposition à l'algorithme UGC, l'utilisation d'un indice de similarité floue rend l'algorithme UFGC robuste aux données bruitées.

1.4. Conclusion sur la classification non supervisée

Cette première partie du chapitre a permis de mettre l'accent sur quelques méthodes de classification non supervisée pouvant être utilisées lors de phases d'apprentissage hors ligne d'un système de RdF floue. En fait, les méthodes décrites constituent un ensemble d'algorithmes de coalescence floue. Elles visent à définir une structure en un nombre prédéfini de classes dans un nuage d'observations en déterminant une partition floue de celui-ci. L'apprentissage est mené au cours d'un processus itératif de minimisation d'un critère d'optimisation de la partition.

L'algorithme ***Fuzzy C-Means*** (FCM) est certainement l'algorithme de coalescence floue le plus connu. Toutefois, celui-ci souffre de quelques limitations ayant donné lieu à diverses extensions. Ainsi, une version possibiliste (PCM) est née pour la caractérisation des observations éloignées tout comme une version permettant la détection des formes propres à chacune des classes (FCMGK). Le choix de la partition optimale, au sens du nombre de classes, se fait au cours d'une phase de validation où on sélectionne la partition optimale parmi un ensemble de résultats à l'aide d'un critère de validité.

L'algorithme ***Unsupervised Robust C-Prototypes*** (URCP) a été défini afin de définir une structure de classes au sein d'un nuage bruité d'observations sans avoir de connaissance *a priori* sur le nombre de classes. La définition de cet algorithme s'appuie sur la minimisation d'un critère similaire à celui des algorithmes classiques de coalescence floue mais faisant intervenir simultanément des coefficients d'appartenance de nature probabiliste et d'autres de nature possibiliste, ce qui lui permet de réunir les avantages du FCM et du PCM. Le choix initial du

nombre de classes de la partition est surestimé. L'algorithme URCP utilisant un apprentissage itératif, le nombre de classes est optimisé à chaque itération par fusion des classes similaires.

Enfin, l'algorithme ***Unsupervised Fuzzy Graph Clustering*** (UFGC) permet l'élaboration d'une partition dans un nuage bruité d'observations sans précision sur le nombre de classes et en présence de classes de forme complexe. En fait, cet algorithme combine les avantages des algorithmes de coalescence et des méthodes hiérarchiques. A partir d'une partition en un nombre surestimé de classes, un graphe flou de proximité est construit. Les sommets et les arcs du graphe correspondent respectivement aux classes et aux liens entre celles-ci au sens d'un indice de similarité floue. La détection de composantes connexes au sein du graphe permet alors de caractériser la présence de classes de forme complexe.

L'ensemble de ces algorithmes, indépendamment de leurs qualités respectives, sont intéressants pour un ***apprentissage hors ligne*** d'un système de RdF. En fait, toutes ces méthodes s'appuient sur la connaissance d'une base d'apprentissage reposant sur la définition d'un ***ensemble fini*** d'observations. L'utilisation de ces méthodes présente toutefois ***un inconvénient*** dans le sens où elle suppose la ***connaissance a priori de toutes les observations nécessaires à la caractérisation des classes***. Or, durant l'exploitation d'un système de RdF, différentes possibilités d'évolutions des données peuvent apparaître et apporter des informations complémentaires d'où la nécessité de remettre en cause l'espace de décision. Dans ce sens, les paragraphes suivants proposent une description de quelques-unes de ces situations en mettant en évidence les besoins d'adaptation.

2. Situations d'adaptation d'un système de RdF

Au cours de l'exploitation d'un système de RdF, de nombreuses situations peuvent nécessiter la remise en question de la structure de classes initiale. Ces situations apparaissent notamment lors de la présentation de nouvelles observations au module de décision. On distingue généralement trois types d'évolutions. La première traduit un ***saut*** des observations en dehors de la classe actuelle. La deuxième traduit un ***éloignement progressif*** des observations de la classe actuelle. Enfin, la troisième considère diverses ***évolutions possibles d'une classe initiale.***

2.1. Saut d'observations

Lors de l'arrivée successive des observations, il est possible qu'à l'instant t_{i+1}, l'observation X^{i+1} présente des caractéristiques très différentes de celles de l'observation précédente. Celle-ci vient alors se positionner dans une région de l'espace de représentation éloignée de celle des observations antérieures. Ce phénomène peut être interprété de différentes façons : soit l'observation X^{i+1} a rejoint une classe déjà identifiée au sein de l'espace de décision (figure III.10.a), soit l'observation X^{i+1} est située loin de toutes les classes connues (figure III.10.b). Dans le premier cas, l'observation est affectée à la classe, dans le second cas, elle est affectée à une classe de rejet en appartenance. Alors que la première situation ne pose pas de problème, la seconde soulève quelques interrogations sur l'adaptation du système de RdF.

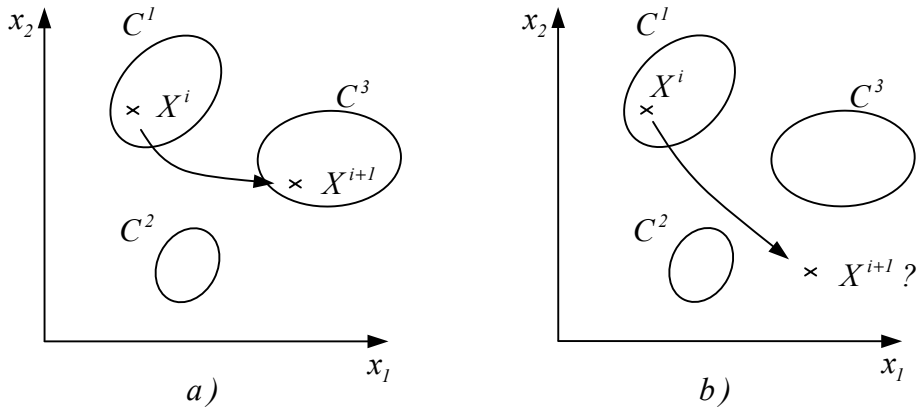


Figure III.10 : a) Saut d'une classe connue vers une classe connue, b) Saut d'une classe connue vers une région à étiquetage inconnu.

L'apparition d'une observation dans une région non définie de l'espace de représentation peut être révélatrice, soit d'une observation aberrante (bruit sur les données d'entrée), soit d'une classe non caractérisée initialement. Ce dernier cas traduit la non-exhaustivité de la base d'apprentissage et nécessite une adaptation du nombre de classes.

2.2. Evolution progressive des observations

Par opposition à la situation précédente, il peut arriver que les observations s'éloignent progressivement d'une classe connue vers une autre classe connue (figure III.11.a) ou vers une région de l'espace de représentation à étiquetage inconnu (figure III.11.b). Il s'agit d'une ***dérive d'observations***.

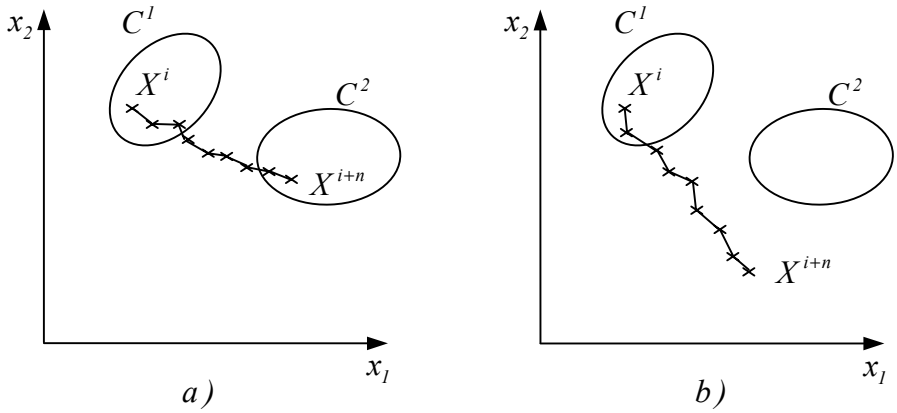


Figure III.11 : a) Evolution progressive vers une classe connue, b) Evolution progressive vers une région à étiquetage inconnu.

Dans ces situations, le problème consiste à détecter de manière précoce l'évolution des observations ainsi que l'occurrence d'une stabilisation éventuelle dans une région identifiée ou non de l'espace de représentation. En fait, toute la difficulté est de savoir distinguer un ***phénomène transitoire*** d'un ***état de stabilisation*** et donc de définir un seuil sur l'évolution des observations. Pour cela, le système de RdF sera d'autant plus efficace que la détection de la dérive des observations est précoce. Par ailleurs, l'apprentissage ne doit être remis en question que s'il y a stabilisation dans une nouvelle région de l'espace de représentation.

2.3. Evolution des classes

Enfin, un certain nombre de phénomènes peuvent entraîner une modification des paramètres du modèle d'une classe. Comme l'illustre la figure III.12, le modèle d'une classe peut perdre sa validité au cours du temps. Par exemple, la classe C^k à l'instant initial t_i (figure III.12.a) peut être amenée à subir ***une modification de forme*** (dispersion, orientation…) (figure III.12.b) et/ou ***un déplacement*** au sein de l'espace de représentation (figure III.12.c).

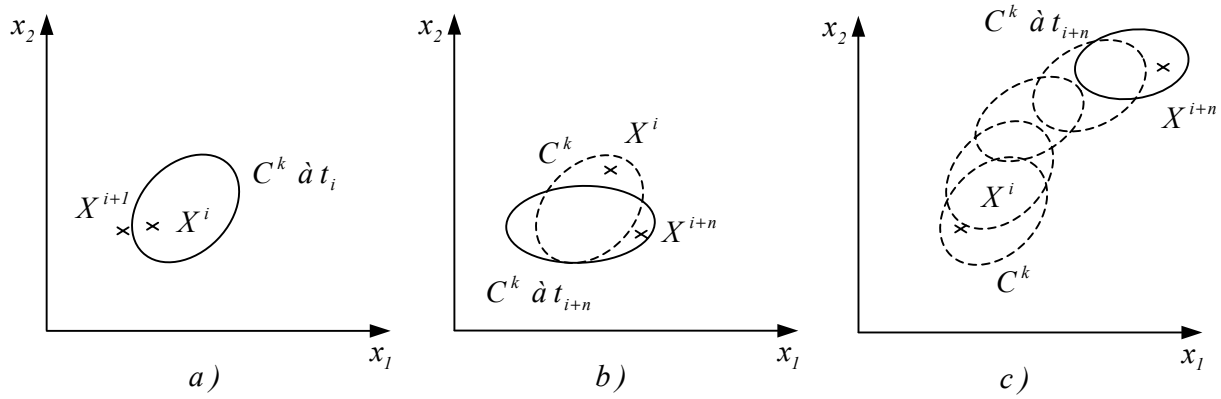


Figure III.12 : a) Classe C^k à l'instant t_i, b) Déformation de la classe C^k à l'instant t_{i+N}, c) Déplacement de la classe C^k à l'instant t_{i+N}.

Par ailleurs, d'autres situations peuvent nécessiter la remise en question de l'apprentissage. Il s'agit par exemple de situations où certaines classes fusionnent suite à l'information apportée par les dernières observations (figure III.13.a) ou se scindent en raison du caractère obsolète d'observations anciennes (figure III.13.b). Ceci nécessite une adaptation du nombre de classes.

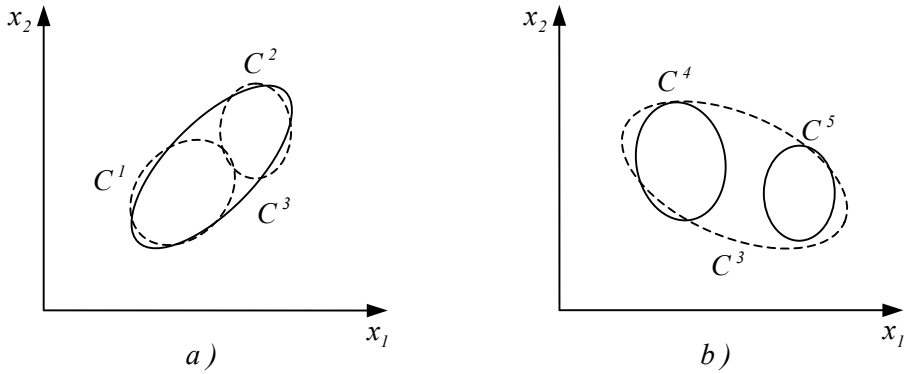


Figure III.13 : a) Fusion des classes C^1 et C^2 pour former la classe C^4, b) Scission de la classe C^3 pour former les classes C^4 et C^5.

Pour les situations évoquées dans ce dernier paragraphe, la caractérisation des évolutions à l'aide d'un apprentissage hors ligne et des concepts de rejet (§3.3 du chapitre 2) ne peut pas procurer une totale satisfaction. En fait, la solution idéale consisterait à définir un apprentissage en ligne du système de RdF afin de prendre en compte continuellement les évolutions des caractéristiques des différentes classes. Evidemment, on comprend bien que la mise en place d'un tel apprentissage nécessite le choix de seuils pour la fusion, la scission…, ce qui n'est pas toujours évident.

2.4. Conclusion sur les situations d'adaptation

Cette partie a eu pour objectif de ***sensibiliser le lecteur aux nécessités d'adaptation*** d'un système de RdF au cours de son exploitation. Celles-ci sont mises en évidence au travers de quelques situations où le besoin d'adaptation de l'espace de décision est décrit. La présentation successive des observations au système de RdF lors de son exploitation peut traduire des situations (saut d'observations, dérive d'observations, dérive de classes) entraînant une modification de la structure initiale de classes et donc des modèles associés à ces dernières.

La ***première réponse*** à ces besoins d'adaptation d'un système de RdF consiste souvent à introduire des ***principes de rejets*** permettant essentiellement la prise en compte de nouvelles classes (voir §3.4 du chapitre 2). Toutefois, l'inconvénient de cette solution provient du fait que le caractère adaptatif de cette approche des systèmes de RdF émane la plupart du temps d'une remise en question périodique de l'espace de décision au travers d'un ***apprentissage hors ligne***. Or, ce principe d'adaptation n'autorise pas la considération des phénomènes de dérives progressives des observations ou des classes, tout au moins de manière précoce. En fait, toute la réussite d'un système adaptatif de RdF floue réside dans son aptitude, d'une part, à modéliser les classes, et d'autre part, à intégrer l'évolution respective de celles-ci au fur et à mesure de la présentation de nouvelles observations. Pour cela, il est nécessaire de définir un ***module d'apprentissage en ligne*** autorisant une actualisation des modèles de classes. Afin de répondre à ce besoin, la troisième partie de ce chapitre est consacrée à une description d'algorithmes intéressants pour la ***classification dynamique de données évolutives.*** En fait, l'introduction de ***capacités d'auto-adaptation*** au sein de ces méthodes permet d'envisager leur utilisation pour la constitution d'un système adaptatif de RdF.

3. La classification dynamique de données évolutives

L'ensemble des algorithmes de classification présentés précédemment réalisent un regroupement en classes d'observations connues *a priori* dans un ensemble fini. Or, comme nous venons de l'évoquer, lors de son exploitation, un système de RdF est confronté à l'apparition de ***nouvelles observations***. Le problème de RdF peut alors être vu comme un problème de ***classification dynamique de données évolutives***. En fait, nous dirons qu'il s'agit d'un problème de classification de ***données évolutives*** dès lors que les caractéristiques des classes évoluent avec le temps, ce qui traduit forcément une non-exhaustivité de la base d'apprentissage initiale. L'utilisation d'***algorithmes dynamiques*** permet alors de pouvoir adapter les résultats de classification, au fur et à mesure que de nouvelles informations apparaissent et sans reconsidérer l'ensemble des données initiales. En fait, par opposition aux techniques de classification présentées au sein de la première partie, les algorithmes dynamiques disposent de capacités d'auto-adaptation permettant d'adapter la structure de classes existante par présentation d'ensembles d'observations nouvelles.

Afin d'illustrer cette problématique, prenons l'exemple d'un système de RdF utilisé pour la classification d'une population (ensemble d'individus) en 3 classes que l'on peut labelliser avec les termes « jeunes », « adultes » et « vieux ». Chaque individu est défini au sein d'un espace de

représentation par ses caractéristiques personnelles telles que son poids et sa taille. D'après les caractéristiques définissant un individu, il est évident qu'un apprentissage des différentes classes en 1900 ou en 2000 ne produira pas la même définition de l'espace de décision. En effet, en un siècle, la morphologie des individus présents au sein de la population a évolué, et ce dans toutes les tranches d'âges. Ainsi, un système de RdF élaboré à partir des données de 1900 n'est plus valide en 2000, bien que les différentes classes continuent d'exister. La définition d'un tel système de RdF et donc des classes considérées nécessite une adaptation vis-à-vis de l'évolution temporelle de la population. En résumé, l'***évolution temporelle*** des observations (individus en l'occurrence) peut conduire à des situations où les modèles des classes existantes doivent être modifiés ou que des nouvelles classes doivent être caractérisées (voir les paragraphes précédents). Ainsi, si l'on considère la dimension temporelle des observations, un système de RdF doit disposer d'un module d'***apprentissage permettant la prise en compte de la nature évolutive des données***.

Peu de travaux traitent concrètement de ce problème de ***classification de données évolutives***. Si l'on considère le chapitre 7 de l'ouvrage de [DAZ96], une seule méthode est présentée en solution : la méthode des centres mobiles [DID71] qui a par ailleurs donné naissance à plusieurs algorithmes de coalescence floue dont l'algorithme FCM. En fait, elle puise son unique aspect « évolutif » dans le fait qu'elle optimise le résultat de classification d'un ensemble fini d'observations au cours d'un processus d'optimisation itératif. Pour leur part, les ***réseaux de neurones artificiels*** (RNA) ont davantage su montrer leur intérêt face à ce problème de classification de données évolutives.

Les prémices des travaux sur les RNA remontent aux années 1940 où les premières réflexions sur le sujet se basaient sur des fondements biologiques [MCC43], bien que la première réelle architecture neuronale, le perceptron, date du début des années 1960 et ait été présentée dans les travaux de Rosenblatt [ROS62]. En fait, les RNA visent à modéliser les capacités d'apprentissage du cerveau humain. Prenons l'évolution d'un enfant de sa naissance à l'âge adulte : celui-ci est confronté à différentes situations qu'il observe et à partir desquelles il réalise un apprentissage. Ainsi, au cours de sa vie, il profite de toutes les situations rencontrées pour améliorer, adapter ou bien même compléter et organiser ses connaissances au sein de sa mémoire. Après une vague d'effervescence, l'enthousiasme autour des RNA s'estompe puis on commence à reparler des RNA avec les ***travaux de Hopfield*** [HOP82]. Celui-ci introduit les réseaux récurrents (feedback) qui présentent une architecture complètement interconnectée et deviennent rapidement très utiles pour la compréhension et la modélisation des processus de la mémoire. Parallèlement à ces travaux, le développement de nouvelles architectures neuronales a ravivé l'intérêt porté aux RNA. Parmi les plus connus, l'algorithme des ***cartes auto-organisatrices de Kohonen*** (***Self-Organizing Map (SOM)***) [KOH84] est l'un des tous premiers RNA présentant un apprentissage non supervisé par réseaux de neurones. Les neurones sont organisés sous la forme d'un maillage : une carte. En fait, cette architecture neuronale puise son appellation dans sa capacité à auto-organiser les pondérations de son architecture à chaque présentation d'une nouvelle observation. Le réseau suit alors continuellement un apprentissage non supervisé de type compétitif où le neurone le plus sensible et ses voisins voient leurs

pondérations adaptées. De la même manière, le réseau ***Learning Vector Quantization*** (LVQ) a montré son aptitude à la classification [KOH86]. Le principe des techniques de quantification vectorielle consiste à caractériser les classes par un ensemble de prototypes. L'apprentissage permet d'adapter les pondérations de l'architecture avec chaque nouvelle observation. Au travers de ces quelques exemples, on perçoit ***les capacités d'auto-organisation des RNA***. Malheureusement, les différentes architectures évoquées ici sont figées en terme de nombre de neurones ce qui n'autorise pas la création de nouvelles classes correspondant à la prise en compte de nouvelles informations.

Des réseaux à ***architecture évolutive*** ont alors vu le jour en proposant des capacités d'auto-adaptation de leur structure. Dans ce sens, [REM97] rappelle les deux grandes stratégies envisageables : une ***stratégie incrémentale*** pour laquelle on part d'un réseau sous-dimensionné auquel on rajoute des neurones et des connexions durant l'apprentissage; une ***stratégie décrémentale*** pour laquelle on part d'un réseau sur-dimensionné auquel on supprime des neurones et des connexions durant l'apprentissage (principe d'élagage ou de pruning) afin d'optimiser la dimension de l'architecture. La première approche est évidemment plus intéressante dans les situations où des observations non considérées initialement se présentent. Le réseau ***Restricted Coulomb Energy*** (RCE) [REI82] est certainement le réseau le plus connu en ce qui concerne sa technique d'apprentissage où l'architecture du réseau est construite de façon incrémentale. En effet, les neurones des couches cachées et de sortie sont créés au cours de la présentation de nouvelles observations. Plus récemment, [PLA91] présente le réseau ***Resource Allocating Network*** (RAN) qui permet l'allocation de nouveaux neurones et l'adaptation des paramètres de ceux existants. En fait, dans le cadre de ces architectures neuronales, la création d'un nouveau neurone est conditionnée par un critère de distance entre la nouvelle observation et les prototypes ou les classes existantes. Différents auteurs ont proposé des évolutions de ce type de critères de ré-allocation [KAD94], [MCL97] ou encore plus récemment [NGO98]. Par ailleurs, les différents algorithmes neuronaux à architecture évolutive évoqués ici suivent un processus d'apprentissage supervisé. Or, en classification de données évolutives, il est assez rare de disposer d'informations *a priori* sur l'appartenance des observations et on doit souvent faire appel à des ***règles d'apprentissage non supervisé***. Dans ses travaux, [HAM97] présente un réseau de neurones à architecture évolutive basé sur l'utilisation de règles d'apprentissage non supervisé. Le réseau présenté est utilisé pour l'estimation de la fonction de densité de probabilité. En fait, il s'agit d'un réseau probabiliste dont l'objectif est similaire aux réseaux Probabilistic Neural Network (PNN) [SPE90] et Gaussian Potential Function Network (GPFN) [LEE91].

Dans cet esprit, cette dernière partie du chapitre 3 est consacrée à la présentation de deux algorithmes de classification à architecture neuronale évolutive selon des règles d'apprentissage non supervisé. La présentation se limitera à deux algorithmes neuronaux qui réunissent la plupart des propriétés nécessaires à un algorithme de classification dynamique de données évolutives. Par ailleurs et par opposition aux travaux de [HAM97], les algorithmes présentés font davantage partie de l'approche métrique du problème de classification. Le premier est l'algorithme neuronal ***Fuzzy Min Max Clustering*** (FMMC) [SIM93] qui appartient aux techniques de coalescence floue agglomérative. Par opposition aux méthodes de classification présentées dans

la première partie du chapitre, les classes sont créées au fur et à mesure que de nouvelles observations apparaissent. Sur ce principe, la première observation donne lieu à la création de la première classe. Le second algorithme est le réseau de neurones ***Cluster Detection and Labeling*** (CDL) [ELT98]. Les principaux avantages de ce réseau proviennent des capacités d'auto-adaptation de son architecture et de l'utilisation de règles d'apprentissage non supervisé permettant la définition de nouveaux prototypes et/ou de classes par la création de nouveaux neurones.

3.1. Le réseau de neurones Fuzzy Min-Max Clustering

En 1992, l'algorithme neuronal Fuzzy Min-Max Classification a été introduit par [SIM92]. Cet algorithme dispose d'une architecture évolutive mais utilise des règles d'apprentissage supervisé. Pour étendre son application au domaine de l'apprentissage non supervisé, [SIM93] propose l'***algorithme neuronal Fuzzy Min-Max Clustering.*** Il s'agit d'une méthode de coalescence floue et agglomérative à base de prototypes flous. La principale particularité de ce réseau réside dans sa capacité à faire évoluer son architecture au cours de son utilisation. Les classes sont caractérisées par un ensemble de prototypes flous hypercubiques. L'apprentissage est mené selon un processus en trois phases comprenant des étapes d'expansion, de contraction et de test de recouvrement entre prototypes.

3.1.1. Architecture du réseau FMMC

Le réseau de neurones FMMC se présente sous la forme d'une architecture de type feedforward en 3 couches successives (figure III.14). La couche d'entrée comporte autant de neurones que de composantes au sein de l'espace de représentation $\Re^D$. Chaque neurone de la couche de sortie caractérise une des K classes de l'espace de décision C. Enfin, chaque neurone de la couche cachée modélise un des J prototypes créés. Les sorties des neurones de la couche cachée et de la couche de sortie définissent alors respectivement les fonctions d'appartenance b_j aux prototypes et B_k aux classes.

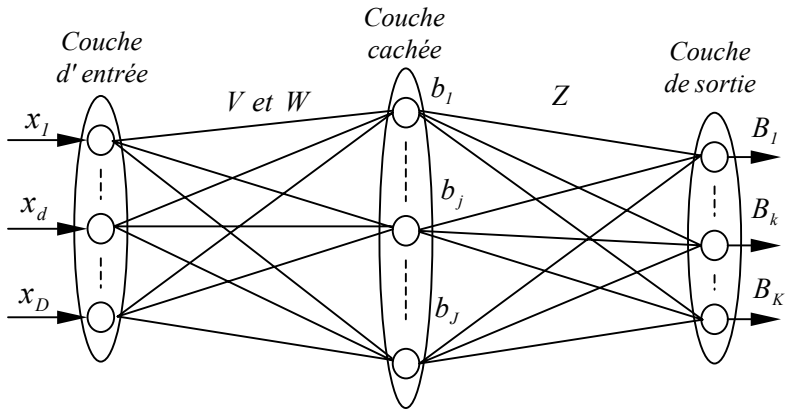


Figure III. 14 : Architecture neuronale du réseau FMMC.

Les connexions entre la couche d'entrée et la couche cachée sont pondérées par deux ensembles de poids permettant de mémoriser les caractéristiques des différents prototypes de la

couche cachée. Ces poids sont mémorisés au sein de deux matrices de pondération notées V et W. De même, les connexions entre la couche cachée et la couche de sortie sont pondérées par les coefficients de la matrice Z caractérisant l'association des prototypes aux classes.

3.1.2. Définition d'un prototype hypercubique flou

Comme nous venons de le voir, chaque classe est représentée par un ensemble de prototypes. Il s'agit de prototypes flous de nature hypercubique. Chaque prototype P^j définit un sous-ensemble flou représenté dans l'espace de représentation $\Re^D$, d'une part, par un couple de points (V^j, W^j) caractérisant la zone d'influence du prototype, et d'autre part, par une fonction d'appartenance b_j.

Les vecteurs V^j et W^j, extraits des matrices V et W, définissent respectivement les coordonnées des points Min et Max de l'hypercube associé au prototype P^j (figure III.15).

Pour la fonction d'appartenance d'une observation $X^i \in \Re^D$ à un prototype P^j, [SIM93] propose l'écriture suivante:

$$b_j(X^i) = \prod_{d=1}^{D} \left[1 - f_\gamma(v_d^j - x_d^i) - f_\gamma(x_d^i - w_d^j) \right] \quad \text{avec} \quad f_\gamma(x) = \begin{cases} 1 & si\ x.\gamma > 1 \\ x.\gamma & si\ 0 \le x.\gamma \le 1 \\ 0 & si\ x.\gamma < 0 \end{cases} \qquad \text{(III.28)}$$

La notation x_d^i exprime la $d^{ème}$ composante du vecteur X^i, et le paramètre γ permet de régler la pente de la fonction d'appartenance en dehors de l'hypercube.

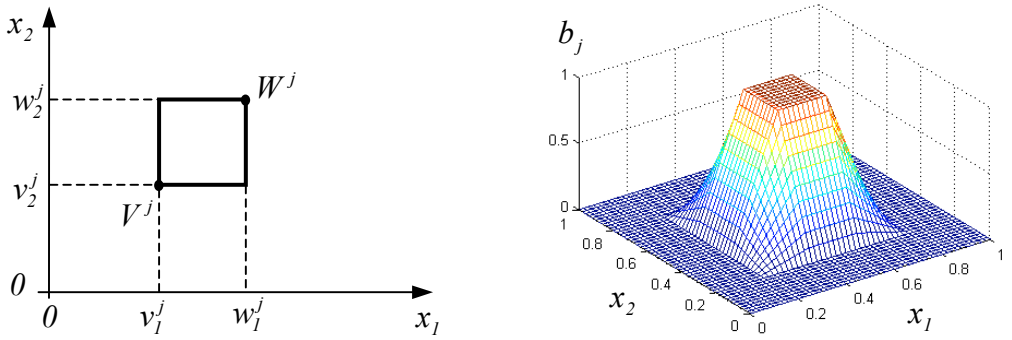


Figure III. 15 : Illustration 2D de l'allure d'un prototype hypercubique flou avec $V^j = [0.4 \ \ 0.4]$, $W^j = [0.6 \ \ 0.6]$ *et* $\gamma = 5$.

La figure III.15 illustre, en dimension $D = 2$, l'allure d'un prototype hypercubique où sont illustrées la zone d'influence décrite par le couple de points (V^j, W^j) et l'allure de la fonction d'appartenance b_j exprimée par (III.28).

3.1.3. Processus d'apprentissage

Le processus d'apprentissage du réseau de neurones Fuzzy Min-Max Clustering consiste à définir un ensemble de prototypes flous hypercubiques permettant de caractériser une structure de classes au sein d'un ensemble X d'observations. L'apprentissage est réalisé à l'aide d'un

algorithme séquentiel non supervisé. La principale caractéristique de cet algorithme se situe dans son caractère agglomératif qui présente l'avantage de ne pas nécessiter la connaissance préalable du nombre de classes.

L'initialisation du processus d'apprentissage est obtenue en créant le premier prototype noté P^1 à partir de l'observation X^1 de X, ce qui se traduit par la création d'un neurone sur la couche cachée. Le processus d'apprentissage se décline alors en 3 phases : ***une phase de dilatation***, ***une phase de test de recouvrement*** et ***une phase de contraction***.

Pour chaque nouvelle observation X^i présentée à l'entrée du réseau, ses degrés d'appartenance aux différents prototypes existants sont évalués à l'aide de l'expression (III.28). Le prototype hypercubique P^j présentant le plus grand degré d'appartenance est alors identifié. Celui-ci subit un ***processus d'expansion*** qui permet de définir le prototype P^{j^*} caractérisé par le couple de points (V^{j^*}, W^{j^*}) et obtenu selon les expressions suivantes :

$$v_d^{j^*} = min\left(v_d^j, x_d^i\right) \ pour\ tout\ d \in \{1, \cdots, D\} \tag{III.29}$$

$$w_d^{j^*} = max\left(w_d^j, x_d^i\right) \ pour\ tout\ d \in \{1, \cdots, D\} \tag{III.30}$$

Toutefois, cette adaptation du prototype P^j en P^{j^*} doit être validée par la vérification d'un ***test de vigilance*** où le volume du prototype résultant P^{j^*} est évalué. Ce test de vigilance vise à maîtriser l'expansion des prototypes et se traduit par la définition d'un ***critère limite d'expansion*** :

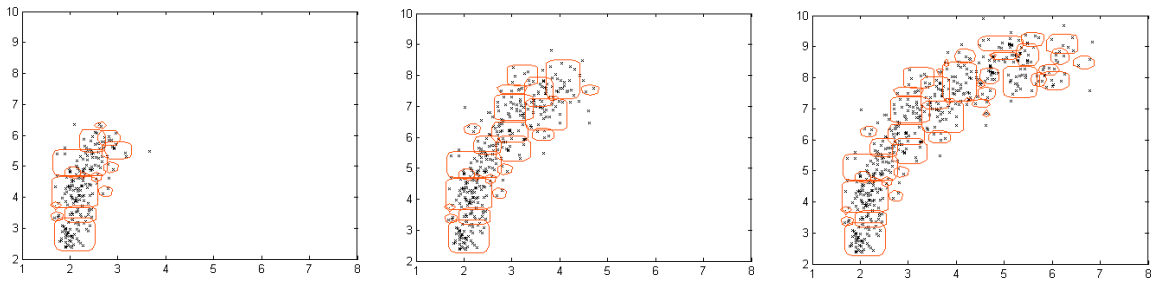
$$\frac{1}{D}\sum_{d=1}^{D}\left(w_d^{j^*} - v_d^{j^*}\right) \leq \theta \tag{III.31}$$

Le terme θ représente le ***seuil de vigilance ou d'expansion***. Si ce critère d'expansion n'est pas vérifié pour le prototype P^j, alors un nouveau prototype P^l est créé à partir du couple de points (V^l, W^l) défini de la façon suivante :

$$V^l = X^i \quad \text{et} \quad W^l = X^i \tag{III.32}$$

A l'issue de cette première phase, si l'observation X^i a donné lieu à la création d'un nouveau prototype, alors on passe à l'observation suivante, sinon on ***teste les recouvrements*** possibles entre le prototype adapté P^{j^*} et l'ensemble des prototypes. Si des recouvrements sont détectés, ceux-ci sont éliminés au cours d'une ***phase de contraction***. Plusieurs types de recouvrements peuvent être distingués. [SIM93] en propose une description détaillée avec pour chaque type de recouvrement un principe de contraction adaptée. Cette phase de contraction donne lieu à une modification des matrices de pondération V *et* W.

Après avoir présenté toutes les observations de X, le processus d'apprentissage est répété jusqu'à atteindre une stabilisation de la définition des prototypes. Une ***phase de validation*** est alors menée afin de labelliser les prototypes et de les réunir en classes à l'aide d'algorithmes hiérarchiques. L'aboutissement de cette étape se traduit par la définition de la couche de sortie et de la matrice de pondération Z associant les prototypes aux classes.



a) *b)* *c)*

Figure III. 16 : Exemple d'application de l'algorithme neuronal FMMC à la classification dynamique de données évolutives.

La figure III.16 illustre un exemple de classification dynamique de données évolutives établi avec l'algorithme neuronal FMMC. En fait, les différentes figures illustrent les prototypes obtenus suite à ***trois apprentissages*** au cours desquels on a présenté ***séquentiellement*** au réseau des ensembles d'observations nouvelles. Le processus d'apprentissage a été initialisé en considérant $\gamma = 4$ pour la définition des prototypes et en prenant comme seuil de vigilance $\theta = 0.9$. La figure III.16.a représente les 15 premiers prototypes obtenus (en rouge sur la figure III.16). Les figures III.16.b et c décrivent alors l'évolution du nombre de prototypes avec respectivement 24 et 44 prototypes créés suite à la présentation de deux ensembles de données nouvelles.

Algorithme FMMC

Fixer une valeur de γ *pour la définition des prototypes,*
Fixer le seuil de vigilance θ,
Initialiser le réseau avec la première observation X^1 *(Eq.(III.32)),*
REPETER
POUR chaque nouvelle observation $X^i \in X$,
Identifier le prototype P^j *ayant le degré* $b_j\left(X^i\right)$ *maximal (Eq.(III.28)),*
Dilater le prototype P^j *en* P^{j^*} *(Eq.(III.29)(III.30)),*
SI la taille du prototype P^{j^*} *est inférieure à* θ *(Eq.(III.31)),*
ALORS
$P^j = P^{j^*}$,
Rechercher les recouvrements entre P^j *et les prototypes existants,*
Eliminer ces recouvrements par contraction des prototypes,
SINO N Créer un nouveau prototype P^l *(Eq.(III.32)),*
FIN
JUSQU'A Stabilisation de la définition des prototypes,
Labelliser les prototypes obtenus et définir la matrice Z,
Définir les fonctions d'appartenance aux classes.

L'***algorithme neuronal FMM-Clustering*** a la capacité d'élaborer une structure de classes à partir d'un ensemble d'observations en créant des prototypes de façon incrémentale. Cet algorithme fait partie des méthodes d'apprentissage associatif dynamique. L'apprentissage est réalisé de manière non supervisé et sans connaissance *a priori* du nombre de classes au fur et à mesure de la présentation de nouvelles données. La capacité d'auto-adaptation de son

architecture en terme de prototypes lui confère une aptitude pour la classification dynamique de données évolutives. Il existe une version généralisée qui associe les variantes supervisées et non supervisées, connue sous le nom d'algorithme General Fuzzy Min-Max (GFMM) [GAB00].

Malheureusement, l'utilisation de l'algorithme FMMC comporte différents inconvénients. La création des classes a lieu en différée lors d'une phase de validation. La modélisation des prototypes n'est pas très précise en raison de l'utilisation de prototypes hypercubiques qui ne permettent pas toujours de représenter correctement la dispersion des observations. Evidemment, leur définition peut être ajustée avec le choix du paramètre γ et du seuil de vigilance θ, mais une valeur de θ trop faible accroît le nombre de prototypes créés.

3.2. Le réseau de neurones Cluster Detection and Labeling

Le ***réseau Cluster Detection and Labeling*** (CDL) est un réseau de type feedforward, développé en 1998 par T. Eltoft et Rui. J. P. Defigueiredo [ELT98]. Le réseau CDL dispose d'une architecture évolutive, comme l'algorithme FMMC, et de règles d'apprentissage non supervisé qui lui confèrent des capacités d'auto-adaptation. Les classes sont caractérisées par un ou plusieurs prototypes. A chaque présentation d'une nouvelle observation, sa ressemblance aux prototypes existants est évaluée sur la base d'une mesure de similarité. Selon les valeurs de similarité, le processus d'apprentissage permet alors notamment de définir des nouveaux prototypes et/ou des nouvelles classes en adaptant l'architecture existante.

3.2.1. Architecture du réseau CDL

Le réseau de neurones CDL est construit autour d'une architecture de type feed-forward, constituée de 3 couches principales et d'un réseau annexe, figure III.17. La couche d'entrée comporte autant de neurones que de composantes au sein de l'espace de représentation $\Re^D$.

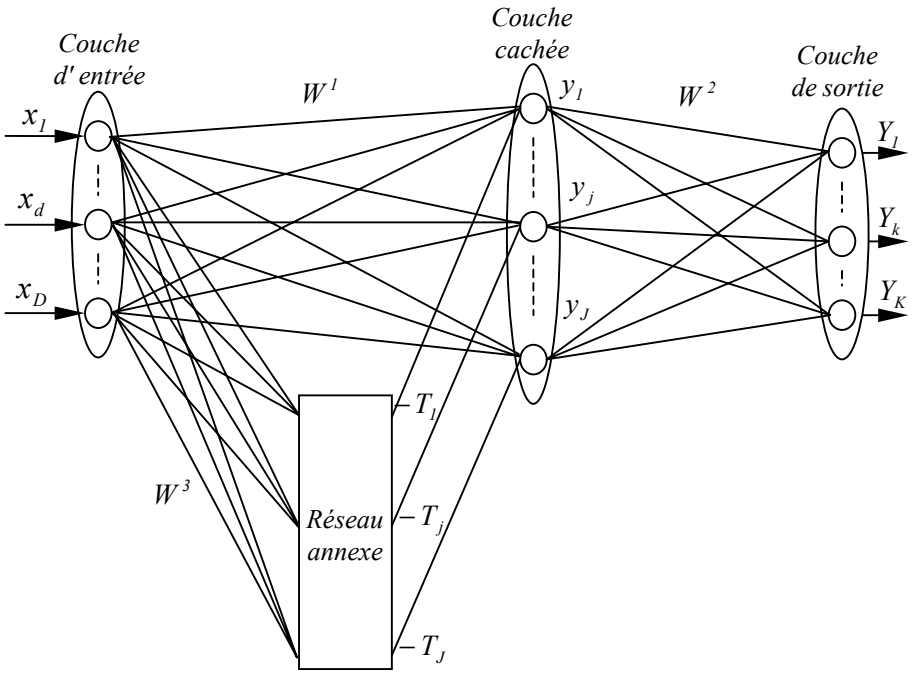


Figure III. 17 : Architecture du réseau CDL.

Les neurones de la couche cachée et de la couche de sortie caractérisent respectivement les prototypes et les classes. Les connexions entre la couche d'entrée et la couche cachée, stockées dans la matrice de pondération W^1, mémorisent les paramètres de définition des prototypes (leurs centres). Les connexions entre la couche cachée et la couche de sortie définissent quant à elles les associations entre les prototypes et les classes selon la matrice binaire W^2.

A chaque présentation d'une observation X^i de l'ensemble X des observations à classer, la sortie y_j du neurone j de la couche cachée prend la valeur 1 s'il y a similarité entre l'observation X^i et le prototype P^j et 0 sinon. De même, la sortie Y_k du neurone k de la couche de sortie vaut 1 si l'observation X^i est affectée à la classe C^k et 0 sinon. Cette appréciation de la similarité entre une observation et un prototype est établie grâce à l'utilisation d'une mesure de similarité et de deux seuils notés ξ_{min} et ξ_{max}. En fait, ce test de similarité nécessite la définition d'un réseau annexe dont les sorties représentent des critères de similarité $T_j = \{T_j^{min}, T_j^{max}\}$ définis pour chaque prototype P^j et calculés à partir des seuils cités précédemment.

L'écriture de ces critères de similarité dépend du choix de la mesure de similarité, décrite au paragraphe suivant. Le réseau annexe est entièrement connecté à la couche d'entrée par une matrice de pondération $W^3 = W^1$.

3.2.2. Définition d'un prototype

Le principe d'apprentissage du réseau CDL nécessite la définition d'une mesure de similarité entre une observation $X^i \in \Re^D$ et un prototype P^j. Dans les travaux de [ELT98], cette mesure de similarité s_{ij} est définie comme étant l'inverse de la distance euclidienne au carré entre l'observation X^i et le centre M_{P^j} du prototype P^j (ici noté simplement P^j), bien que d'autres mesures soient envisageables :

$$s(P^j, X^i) = s_{ij} = \frac{1}{d(X^i, P^j)^2} \qquad \text{(III.33)}$$

Un prototype est alors caractérisé par son centre M_{P^j} et une région de forme hypersphérique qui traduit la zone d'influence du prototype dans l'espace de représentation.

La figure III.18 illustre la mesure de similarité associée aux différents prototypes. Celle-ci met en évidence un problème de sensibilité au voisinage proche du prototype puisqu'elle est non bornée. Sur ce principe, la mesure de similarité définie par (III.33) ne peut être vue comme une fonction d'appartenance et donc l'algorithme ne permet pas de définir des fonctions d'appartenance aux classes.

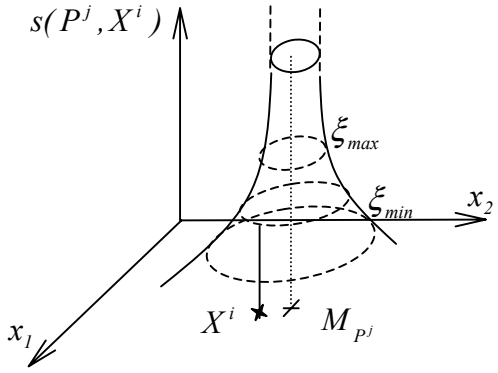


Figure III. 18 : Illustration 2D de l'allure d'un prototype caractérisé par son centre et la mesure de similarité.

Enfin, pour revenir aux critères de similarité T_j^{min} et T_j^{max} propres à chaque prototype, le calcul et l'écriture de ces derniers s'appuient sur une reformulation matricielle de (III.33) où la mesure de similarité est écrite en faisant intervenir le produit scalaire $X^{iT} P^j$ entre une observation X^i et le centre M_{P^j} d'un prototype P^j (que l'on note ici P^j par souci de simplification). Ainsi, d'un point de vue pratique, la similarité peut être interprétée par la comparaison du produit scalaire $X^{iT} P^j$ aux critères T_j^{min} et T_j^{max} qui ne sont que des reformulations des seuils ξ_{min} et ξ_{max} :

$$T_j^{min} = \frac{\xi_{min} \cdot (\|X^i\|^2 + \|P^j\|^2) - 1}{2.\xi_{min}} \quad \text{et} \quad T_j^{max} = \frac{\xi_{max} \cdot (\|X^i\|^2 + \|P^j\|^2) - 1}{2.\xi_{max}} \qquad \text{(III.34)}$$

3.2.3. Processus d'apprentissage

Le réseau de neurones CDL est utilisé pour regrouper en classes les observations d'un ensemble X d'observations. Pour cela, l'algorithme suit un processus d'apprentissage cyclique au cours duquel chaque cycle est décomposé en 3 phases : ***classification sans fusion***, ***fusion de classes*** et ***évaluation des classes***. Toutefois, avant de débuter ce processus cyclique, le réseau est initialisé avec la première observation en créant le premier prototype (premier neurone de la couche cachée) ainsi que la première classe (premier neurone sur la couche de sortie). Par ailleurs, il faut également fixer les valeurs des seuils de similarité ξ_{min} et ξ_{max}.

Au cours de la phase de ***classification sans fusion,*** l'ensemble des observations à classer sont présentées à l'architecture neuronale. Les observations sont présentées successivement et pour chacune d'entre elles leur similarité par rapport aux prototypes existants est évaluée. Différentes situations peuvent se présenter par comparaison de cette similarité aux seuils de similarité ξ_{min} et ξ_{max}. Pour chacune d'entre elles, les effets sont différents et peuvent aller de la simple affectation de l'observation à une classe à la création d'une nouvelle classe et d'un nouveau prototype en passant par la création d'un nouveau prototype pour une classe existante. Si l'observation X^i est jugée trop éloignée des prototypes existants ($s_{ij} < \xi_{min} \ \forall P^j$), alors un nouveau prototype et une nouvelle classe sont créés. Si l'observation X^i est jugée proche d'un

prototype P^j d'une classe donnée mais pas assez pour être associé à ce prototype ($\xi_{min} < s_{ij} < \xi_{max}$), alors un nouveau prototype est créé et associé à cette même classe. Si l'observation X^i est jugée très proche d'un prototype P^j d'une classe donnée ($s_{ij} > \xi_{max} > \xi_{min}$), alors X^i est simplement affectée à la classe considérée. Enfin, il se peut que l'observation X^i présente une similarité avec des prototypes de classes différentes ($s_{ij} > \xi_{min}$ et $s_{il} > \xi_{min}$ avec P^j et P^l associés à des classes différentes). L'observation est alors considérée ambiguë et écartée dans un ensemble X_{ec}.

La phase de ***fusion de classes*** permet de traiter l'ensemble des observations écartées en réalisant une fusion des classes pour lesquelles elles ont soulevé une ambiguïté.

La phase d'***évaluation des classes*** permet de supprimer les classes contenant trop peu d'observations vis-à-vis d'un seuil de cardinalité N_{min}^C. Les observations concernées sont alors rejetées dans un ensemble X_{rej}.

De plus amples informations sur les règles d'apprentissage utilisées au sein de ces trois phases pourront être consultées dans les travaux de [ELT98].

A l'issue d'un cycle du processus d'apprentissage, on dispose d'un ensemble X_{cl} d'observations classées, d'un ensemble X_{ec} d'observations écartées pour ambiguïté et d'un ensemble X_{rej} d'observations rejetées pour des raisons de cardinalité. Un nouveau cycle du processus d'apprentissage est alors lancé après modification des seuils de similarité :

$$\xi^{cycle+1} = \alpha * \xi^{cycle} \quad \text{avec} \quad \frac{1}{2} \leq \alpha < 1 \tag{III.35}$$

Ceci a pour effet de diminuer les seuils ξ_{min} et ξ_{max} et donc d'accroître la zone d'influence des prototypes entre les différents cycles. Les cycles du processus d'apprentissage sont alors répétés jusqu'à atteindre un pourcentage P_{min} d'observations classées.

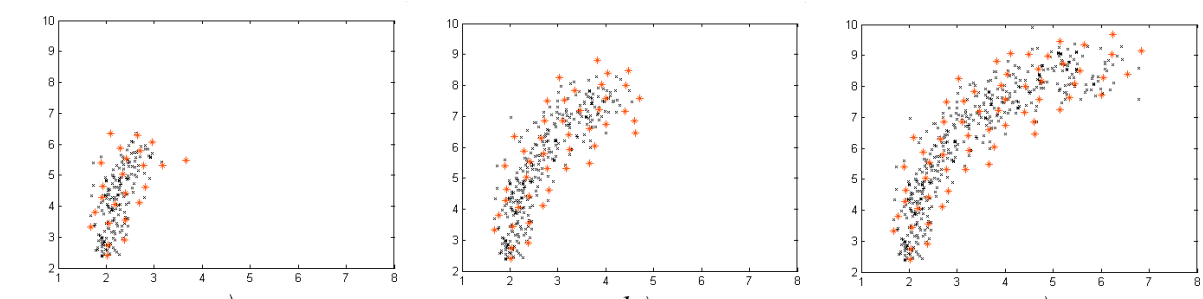


Figure III. 19 : Exemple d'application de l'algorithme neuronal CDL à la classification dynamique de données évolutives.

La figure III.19 illustre un exemple de classification dynamique de données évolutives établi avec l'algorithme neuronal CDL. Les données utilisées sont les mêmes que celles utilisées avec l'algorithme FMMC (figure III.16) et sont présentées au réseau selon le même principe. Le processus d'apprentissage a été initialisé en prenant les valeurs de seuils $\xi_{min} = 4$ et $\xi_{max} = 8$,

puis en fixant les paramètres suivants $\alpha = 0.9$, $N_{min}^{C} = 10$ et $P_{min} = 90$. La figure III.19.a représente les 24 premiers prototypes obtenus (en rouge sur la figure III.19). Les figures III.19.b et c décrivent alors l'évolution du nombre de prototypes avec respectivement 46 et 63 prototypes créés suite à la présentation de deux ensembles de données nouvelles.

Algorithme CDL

Fixer les seuils ξ_{min} *et* ξ_{max} *,*

Fixer les paramètres N_{min}^{C} *et* P_{min} *,*

Initialiser le réseau avec la première observation X^1 *,*
REPETER

POUR chaque nouvelle observation X^i *,*

Calculer sa similarité s_{ij} *avec l'ensemble des prototypes* P^j *(Eq.(III.33)),*

SI $s_{ij} < \xi_{min}$ *pour tous les* P^j *,*

ALORS Créer un nouveau prototype et une nouvelle classe ($X^i \rightarrow X_{cl}$*),*

SINON SI $s_{ij} > \xi_{min}$ *pour des prototypes de la même classe* C^k *,*

ALORS SI $s_{ij} < \xi_{max}$ *, créer un nouveau prototype pour* C^k *(*$X^i \rightarrow X_{cl}$*),*

SINON SI $s_{ij} > \xi_{min}$ *pour des prototypes de classes différentes,*

ALORS Ecarter l'observation X^i *en ambiguïté (*$X^i \rightarrow X_{ec}$*),*

Fusionner les classes pour lesquelles au moins une ambiguïté a été soulevée,

Evaluer les classes et éclater les classes ayant une cardinalité < à N_{min}^{C} *(*$X^i \rightarrow X_{rej}$*),*

FIN

Adapter les seuils de similarités ξ_{min} *et* ξ_{max} *(Eq.(III.35)),*

Définir l'ensemble $X = \{X_{ec}, X_{rej}\}$

TANT que le pourcentage d'observations classées est inférieur à P_{min}

Le ***réseau de neurones CDL*** utilise des règles d'apprentissage non supervisé qui lui confèrent de réelles capacités d'auto-adaptation. En effet, celles-ci autorisent la création de nouveaux neurones afin de caractériser des nouveaux prototypes mais aussi de nouvelles classes. Ainsi, contrairement au réseau FMMC, le réseau CDL présente une architecture totalement évolutive et donc intéressante pour la classification dynamique de données évolutives.

Toutefois, la définition de ce réseau est limitée par l'utilisation d'une mesure de similarité qui d'une part n'est pas sensible au voisinage des prototypes et d'autre part ne permet pas la définition de fonctions d'appartenance aux classes. De plus, un prototype est défini par une unique observation définissant son centre. Ceci induit la création de nombreux prototypes qui par ailleurs peuvent apparaître sur la périphérie des classes réelles. Bien entendu, la sensibilité des prototypes peut être ajustée par le choix des seuils de similarité ξ_{min} et ξ_{max}. Enfin, le principe de la mesure de similarité nécessite le calcul de critères de similarités à chaque nouvelle présentation d'une observation ce qui requiert la présence d'un réseau annexe supplémentaire.

3.3. Conclusion sur la classification dynamique de données évolutives

Dans cette troisième partie, il a été montré qu'actuellement peu de méthodes existent pour la classification dynamique de données évolutives bien que les réseaux de neurones semblent présenter certaines aptitudes vis-à-vis de ce type de problème. Par le passé, les RNA ont déjà su montrer leurs capacités d'auto-organisation en permettant d'améliorer continuellement leur apprentissage. Plus récemment, des réseaux à architecture évolutive sont apparus et ont accru les capacités d'auto-organisation des réseaux de neurones en autorisant la création de nouveaux neurones au cours de l'apprentissage. Sur ce principe, cette partie a mis l'accent sur la présentation de deux algorithmes neuronaux disposant de capacités d'auto-adaptation de leur architecture via l'utilisation de règles d'apprentissage non supervisé.

Le premier, l'***algorithme neuronal Fuzzy Min Max Clustering,*** fait partie des méthodes de coalescence floue agglomérative pour lesquelles la création des prototypes a lieu au fur et à mesure de la présentation des observations. Durant l'apprentissage, les prototypes existants sont adaptés ou de nouveaux sont créés. La zone d'influence des prototypes est caractérisée par une région de forme hypercubique à laquelle on associe une fonction d'appartenance. Ce choix de définition des prototypes, bien qu'ayant l'avantage de simplifier le processus d'adaptation de ces derniers, restreint la finesse de caractérisation des classes. De plus, les fonctions d'appartenance aux classes ne sont définies qu'à l'issue d'une phase de validation réalisée après le processus d'apprentissage des prototypes.

Le second, l'***algorithme neuronal Cluster Detection and Labeling,*** a pour sa part l'avantage de réunir à la fois des capacités d'apprentissage continu de prototypes et de classes. En effet, le processus d'élaboration de ce réseau est construit à partir d'un ensemble de règles d'apprentissage non supervisé autorisant la création de nouveaux neurones sur la couche cachée (prototypes) et sur la couche de sortie (classes). Ces règles sont établies sur la base d'une mesure de similarité entre les observations et les prototypes. La comparaison du degré de similarité d'une observation aux prototypes à deux seuils spécifiques permet alors de décider de la création de nouveaux prototypes et/ou de nouvelles classes. Hélas, l'utilisation d'une fonction de similarité non bornée n'autorise pas l'élaboration de fonctions d'appartenance aux prototypes et aux classes. De plus, celle-ci entraîne un problème de sensibilité au proche voisinage des prototypes. Par ailleurs, un prototype est défini par une seule observation caractérisant son centre. Sur ce principe, des prototypes sont créés en bordure de classes et le nombre de prototypes créés est important. Enfin, à chaque nouvelle présentation d'une observation, la mesure de similarité nécessite la détermination de critères de similarité à l'aide d'un réseau annexe.

4. Conclusion

Les différents besoins d'apprentissage d'un système de RdF ont été mis en évidence au travers des différentes parties de ce chapitre. L'élaboration d'un système de RdF débute par une ***phase d'apprentissage initial***. Celle-ci est menée sur la base d'un ensemble d'observations prédéfinies. Dans le cadre d'un système de RdF floue, l'apprentissage s'établit à l'aide de

méthodes de classification floue. Si l'on suppose ne pas connaître l'appartenance des observations initiales, ces méthodes doivent suivre un apprentissage non supervisé.

Sur ce principe, la première partie du chapitre a été dédiée à une description de différents ***algorithmes de classification floue*** pouvant être utilisés pour initialiser un système de RdF. Différents algorithmes classiques de coalescence floue ont été exposés. Ceux-ci permettent de réaliser une partition d'un nuage d'observations en un nombre donné de classes. Un algorithme de classification permettant d'établir une partition sans connaissance *a priori* du nombre de classes a alors été exposé, ainsi qu'un algorithme permettant de détecter des classes de forme complexe au sein d'un nuage d'observations.

Toutefois, l'apprentissage réalisé avec ces méthodes ne considère que la connaissance présente dans l'ensemble initial de données qui est rarement exhaustif. Or, le système de RdF ne doit pas être figé à l'issue de son apprentissage initial. En effet, la deuxième partie du chapitre a proposé une description d'un ensemble de situations où les ***nécessités d'adaptation d'un système de RdF*** sont perceptibles. Ces situations sont décrites par des séquences d'évolutions des observations pouvant traduire des évolutions des modèles de classes et donc nécessiter une adaptation de la modélisation de ces dernières.

La définition d'un module d'apprentissage autorisant l'adaptation des modèles de classes nécessite la définition de ***méthodes de classification dynamique de données évolutives*** disposant de capacités d'auto-adaptation indispensables à la prise en compte de nouvelles informations. A cet effet, nous avons vu que les ***réseaux de neurones*** présentent des pré-dispositions. Parmi les algorithmes neuronaux, certains proposent des architectures évolutives associées à l'utilisation de règles d'apprentissage non supervisé. Dans ce sens, notre présentation s'est portée sur ***deux architectures évolutives*** disposant de propriétés intéressantes telles que l'introduction de fonctions d'appartenance aux classes, de règles d'adaptation des prototypes, de création de prototypes mais aussi de classes…. En fait, les deux techniques présentées suivent un ***mode d'apprentissage séquentiel*** permettant d'actualiser régulièrement la structure de classes par la présentation d'un ensemble contenant des nouvelles observations.

En conclusion, il est évident qu'un système de RdF ne doit pas être figé à l'issue de la phase d'apprentissage initial ne serait-ce qu'en raison du caractère incomplet de la base de données initiales. Pour cela, des méthodes de classification à architecture évolutive existent mais ne présentent pas forcément l'ensemble des qualités requises. Le chapitre suivant est donc consacré à la présentation d'***une nouvelle architecture neuronale auto-adaptative pour la classification dynamique de données évolutives***. En fait, deux modes d'apprentissage, ***séquentiel*** ou ***continu***, sont envisageables pour l'adaptation de l'architecture en présence de données évolutives. Dans un premier temps, nous proposons un mode d'***apprentissage séquentiel*** pour lequel l'adaptation est réalisée ***périodiquement*** lors de la présentation d'un ensemble de données comportant les observations nouvelles, comme pour le FMMC et le CDL. Ensuite, nous décrivons un mode d'***apprentissage continu*** où les données sont considérées en ligne pour l'adaptation de l'architecture. Cette seconde approche est plus intéressante pour intégrer la nature évolutive des données et fournit ***une modélisation totalement dynamique des classes*** via l'introduction d'une ***notion d'oubli permettant de limiter l'influence des observations anciennes***.

***Chapitre IV* : Réseau de neurones auto-adaptatif pour la classification dynamique de données évolutives : AUDyC**

0. Introduction

La mise en œuvre d'un système de Reconnaissance de Formes, permettant une ***modélisation dynamique des classes*** au fur et à mesure de la présentation de nouvelles observations, nécessite la définition d'un module d'***apprentissage en ligne***. Le module d'apprentissage doit intégrer des capacités d'auto-adaptation autorisant l'évolution des modèles des classes existantes ainsi que la caractérisation de nouvelles classes. En fait, il est nécessaire de considérer le caractère « innovant » des nouvelles observations présentées au système de RdF et par opposition le caractère « obsolète » des observations anciennes. La construction du module d'apprentissage du système de RdF requiert alors l'élaboration d'une méthode de ***classification dynamique de données évolutives*** adaptée à la ***modélisation dynamique des classes***. Différentes propriétés nécessaires à la définition de ce type de méthodes de classification sont détaillées ci-après.

Une première propriété concerne la ***qualité de modélisation des classes***. En effet, la caractérisation de classes de forme complexe nécessite d'utiliser une ***approche multiprototype***. L'appartenance d'une observation à une classe est alors interprétée à partir de son niveau d'appartenance aux différents prototypes représentatifs de la classe. Selon ce principe, la qualité de modélisation des classes est fortement liée à la qualité de modélisation des prototypes et donc à la ***définition de leur zone d'influence*** et à la ***sensibilité de leur fonction d'appartenance.***

Une deuxième propriété vise à associer des ***possibilités d'adaptation de la structure de classes ainsi définie*** à la technique de classification. En effet, la détermination initiale du nombre de classes présentes au sein des données considérées constitue souvent la première difficulté des techniques de classification. De même, il n'est pas évident de définir initialement le nombre de prototypes nécessaire à la caractérisation d'une classe, tout comme leur position, leur forme et leur orientation au sein de cette dernière. Une solution consiste alors à ***modifier le nombre de classes et de prototypes et à adapter les paramètres de ces derniers*** au fur et à mesure de la présentation de nouvelles observations. Ainsi, la structure de classes est caractérisée de façon progressive via l'utilisation de règles d'apprentissage spécifiques. Si aucune connaissance *a priori* n'est disponible quant à l'appartenance des observations à classer, alors ces ***règles*** relèvent de l'***apprentissage non supervisé***. On parle de ***règles d'auto-adaptation*** de la structure de classes.

Une troisième propriété, indispensable à la ***modélisation dynamique de données évolutives,*** vise à intégrer l'***évolution temporelle des observations***. Pour cela, l'introduction de capacités d'auto-adaptation de la structure de classes évoquée ci-dessus est évidemment indispensable. En effet, le nombre et la définition des prototypes et des classes peuvent évoluer au fur et à mesure de la présentation temporelle des données. Leur remise en cause est donc nécessaire à la ***caractérisation de « classes dynamiques »***. Une complète considération du problème de modélisation dynamique de données évolutives n'est alors possible qu'avec l'utilisation d'un

mode d'***apprentissage continu.*** Les données évolutives sont considérées en ligne au travers de fenêtres glissantes. De plus, il doit être possible d'intégrer une capacité d'***oubli des informations les plus anciennes***.

Dans ce sens, nous avons vu à la fin du chapitre 3 que les réseaux de neurones présentaient des aptitudes pour la définition d'une technique de classification dynamique de données évolutives. Toutefois, bien que disposant de capacités d'auto-adaptation de leur architecture, aucune des deux techniques présentées ne vérifie l'ensemble des propriétés énumérées précédemment. En effet, les réseaux CDL et FMMC proposent une solution au problème de classification dynamique de données évolutives mais se limitent à l'utilisation d'un mode d'***apprentissage séquentiel*** au cours duquel les modèles de classes ne peuvent être actualisés que périodiquement suite à la présentation d'un nouvel ensemble de données au réseau.

Afin de combler cette lacune, nous présentons dans ce chapitre un nouvel algorithme neuronal, ***le réseau AUDyC (AUto-adaptive and Dynamical Clustering)*** développé pour la ***classification dynamique de données évolutives.*** En fait, la présentation du réseau AUDyC est réalisée dans un premier temps en considérant un mode d'***apprentissage séquentiel***, puis ensuite un mode d'***apprentissage continu*** afin d'autoriser la modélisation dynamique des classes.

La ***première partie*** présente l'architecture neuronale du réseau AUDyC puis décrit ***son mode d'apprentissage séquentiel*** au travers de l'utilisation d'un processus d'apprentissage cyclique. En fait, cette première approche du problème de classification de données évolutives peut être assimilée à un problème de ***classification dynamique*** de données stationnaires pour lequel l'intégralité des classes connues n'est pas décrit initialement dans les données d'apprentissage. Le résultat est donc optimisé périodiquement lors de la présentation d'un nouvel ensemble de données. L'avantage des algorithmes dynamiques par rapport à une approche classique (1ère partie du chapitre 3) provient du fait qu'il n'est pas nécessaire de reconsidérer toute la base d'apprentissage mais seulement les nouvelles données. Ainsi, cette première partie porte l'accent sur les ***capacités d'apprentissage*** de notre algorithme neuronal en présentant, d'une part, son ***architecture auto-adaptative,*** et d'autre part, son ***processus d'apprentissage cyclique***.

La ***seconde partie*** s'appuie sur l'architecture neuronale développée précédemment mais propose un ***mode d'apprentissage continu*** afin d'établir une modélisation dynamique des classes. L'algorithme reprend l'ensemble des capacités d'auto-adaptation de l'architecture du réseau AUDyC mais les applique en ligne à l'ensemble des données. Pour cela, un ***processus d'apprentissage en ligne***, s'appuyant sur des fenêtres glissantes sur les données évolutives, est présenté. Par ailleurs, les règles d'apprentissage sont complétées afin de considérer un degré d'***oubli des informations les plus anciennes***, ce qui permet de mieux apprécier l'évolution temporelle des classes. Les capacités de modélisation dynamique de données évolutives sont alors évaluées au travers de différentes expérimentations où les avantages du mode d'apprentissage continu sont mis en évidence.

En préambule de ces deux parties, un premier paragraphe décrit les principes et concepts utilisés. Par ailleurs, ce premier paragraphe introduit la terminologie employée tout au long de ce chapitre.

1. Définitions et principes

La démarche d'élaboration d'une technique de classification doit tenir compte de la nature des données à classer : ***stationnaires*** ou ***évolutives (non stationnaires)***. Dans le cadre de données stationnaires, l'objet de la méthode de classification est de définir une structure de classes à partir d'un ensemble de données caractérisant des classes « stables ». Par opposition, dans le cadre de données évolutives, la technique de classification doit établir une structure de classes à partir de données disponibles au fur et à mesure de l'évolution temporelle des classes. Chaque technique de classification dispose donc d'un mode d'apprentissage spécifique.

En ***classification de données stationnaires***, l'apprentissage peut être envisagé de différentes façons selon la disponibilité de l'ensemble des observations, i.e. de l'exhaustivité de la base d'apprentissage. Si l'on considère que la base d'apprentissage est complète, alors les observations peuvent être considérées ***de manière simultanée***. Le résultat de classification est alors optimisé au cours d'un processus itératif (FCM, …) où toutes les observations sont considérées simultanément. Si la base d'apprentissage est incomplète, alors il est plus judicieux de s'orienter vers des algorithmes dynamiques ou les observations sont considérées individuellement. Cette deuxième approche permet d'optimiser la structure de classes initiales ***de manière séquentielle*** en présentant périodiquement un ensemble d'observations nouvelles.

En ***classification de données évolutives***, la base d'apprentissage est nécessairement incomplète d'où l'importance de faire appel à des algorithmes de classification dynamique pour autoriser l'adaptation des classes existantes et la création de nouvelles classes. En fait, l'utilisation d'algorithmes dynamiques requiert la mise en place de différentes règles d'auto-adaptation au travers d'un mode d'***apprentissage séquentiel*** ou ***continu***. Le mode séquentiel est similaire à celui évoqué pour les données stationnaires. Le mode continu a pour vocation de réaliser une ***modélisation dynamique des classes*** à partir de l'ensemble des données évolutives et requiert dans ce sens des aptitudes particulières (oubli des observations anciennes, …).

Ceci étant, différentes capacités d'apprentissage existent au sein des algorithmes neuronaux dédiés à la classification. On parle d'ailleurs d'***auto-organisation*** ou d'***auto-adaptation*** de leur architecture. La capacité d'auto-organisation vise à ajuster les pondérations du réseau sans en modifier la structure, alors que la capacité d'auto-adaptation permet d'adapter les pondérations tout en modifiant la structure du réseau en terme de nombre de neurones. Dans le cadre du développement de notre technique neuronale dédiée à la classification dynamique de données évolutives, nous avons donc préféré opter pour des règles d'auto-adaptation, ce qui permet de définir une ***architecture totalement évolutive***.

Ces quatre premiers paragraphes ont permis de préciser quelques définitions utilisées par la suite. Dans les paragraphes suivants, nous indiquons les différents principes retenus afin de conférer à notre technique de classification, les propriétés énumérées en introduction.

Le principe de base consiste à s'appuyer sur une ***approche multiprototype*** permettant la modélisation de classes de forme complexe. Les classes de forme complexe, i.e. non convexe (figure IV.1), peuvent être décomposées en une ***combinaison d'entités élémentaires : les prototypes***. Ceux-ci sont définis par des modèles gaussiens ayant l'avantage d'approcher la

distribution réelle des données au sein des entités élémentaires considérées. L'appartenance d'une observation à une classe est alors interprétée par rapport à ses degrés d'appartenance aux prototypes associés à cette même classe.

La caractérisation d'une structure de classes s'établit alors au cours d'un processus d'***apprentissage des prototypes et des classes*** sur la base des données évolutives présentées au réseau. Selon l'appartenance de ces observations aux prototypes et classes existants, de nouveaux prototypes et/ou de nouvelles classes sont créés. De plus, les modèles des prototypes ne sont pas optimaux dès leur création. Ceux-ci doivent donc être adaptés au fur et à mesure de la présentation des observations. Ainsi, pour améliorer la qualité de modélisation, ***des règles d'adaptation des paramètres des prototypes*** ont été définies et permettent d'ajuster la position, la forme et l'orientation de ces derniers.

Enfin, afin d'éviter la création d'un nombre excessif de prototypes, nous avons complété les règles d'apprentissage par un principe de ***fusion de prototypes*** dès lors que ceux-ci sont jugés proches. De même, il existe des situations où des classes, caractérisées initialement séparément, ne définissent en réalité qu'une seule et même classe suite à l'apparition de nouvelles observations dans une région proche de plusieurs classes distinctes. Un principe de ***fusion de classes*** a donc été également défini. Celui-ci permet de réunir les prototypes des différentes classes concernées et de les associer à une classe unique.

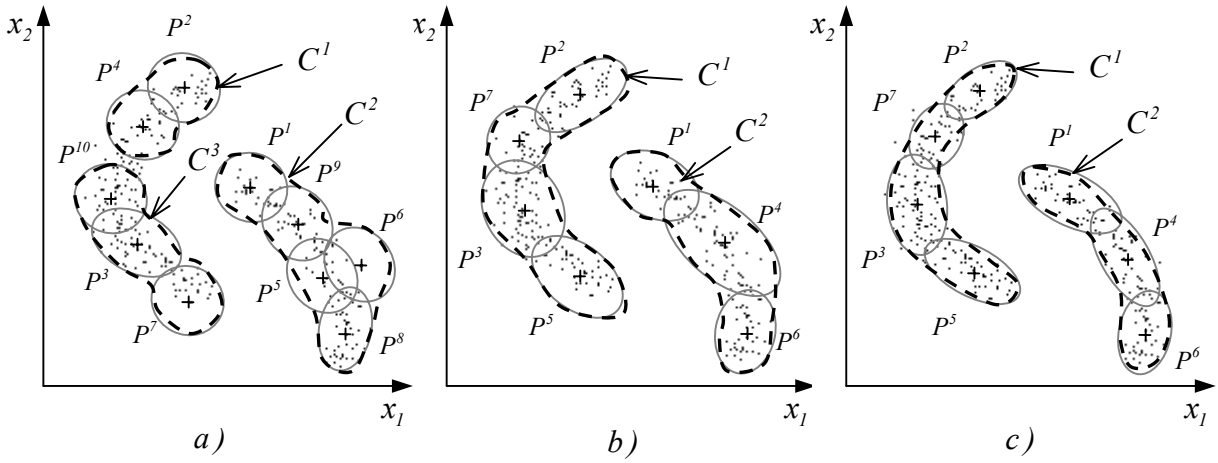


Figure IV.1 : Illustration de la démarche proposée pour l'apprentissage d'une structure de classes : a) création initiale de prototypes et de classes, b) fusion de prototypes et de classes, c) adaptation des prototypes.

Les différents principes retenus et évoqués ci-dessus sont illustrés au travers des trois étapes d'apprentissage décrites figure IV.1. Les modèles initiaux des prototypes et des classes ne sont pas optimaux (figure IV.1.a). Les prototypes P^6 et P^7 ne caractérisent pas une région dense. De même, une des classes réelles est caractérisée par la classe C^1 et par la classe C^3. La figure IV.1.b décrit alors une première phase d'adaptation avec la fusion des classes C^1 et C^3, ainsi que la fusion de différents prototypes. La figure IV.1.c expose le résultat de modélisation après une complète adaptation des paramètres des prototypes. Enfin, si l'on prend l'exemple du prototype P^4 (figure IV.1.b), on remarque que celui tend à se repositionner et à se déformer afin de mieux représenter les données qui lui sont associées (figure IV.1.c).

2. Architecture auto-adaptative et mode d'apprentissage séquentiel

Cette première partie propose une description détaillée de l'***architecture auto-adaptative*** du ***réseau AUDyC*** puis présente son ***mode d'apprentissage séquentiel*** en insistant sur ses capacités d'adaptation des modèles de classes au fur et à mesure de l'apprentissage. Les capacités de classification dynamique du réseau AUDyC sont alors validées par des expérimentations sur un ensemble de données stationnaires.

2.1. Architecture auto-adaptative

Afin d'introduire l'***approche multiprototype*** pour la caractérisation des classes de forme complexe, l'architecture proposée pour le réseau AUDyC définit une structure où les prototypes et les classes sont représentés respectivement par les neurones des couches cachée et de sortie. L'architecture proposée est alors de type feed-forward et se compose de 3 couches (figure IV.2).

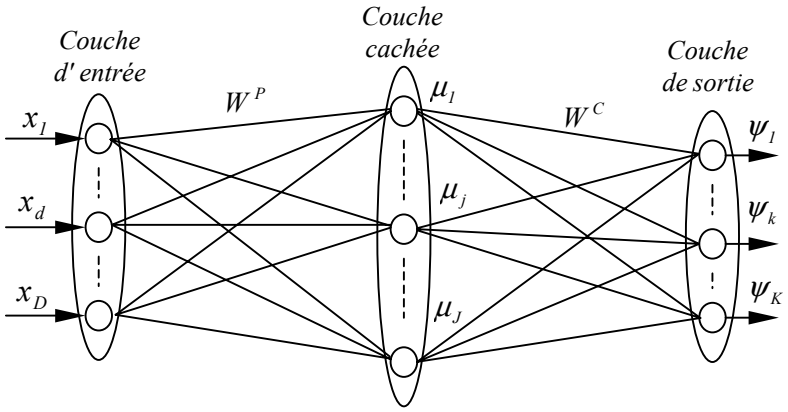


Figure IV.2 : Architecture proposée pour le réseau AUDyC.

Cette architecture est similaire à celle d'un réseau RBF (Radial Basis Function Network). En fait, la particularité de notre architecture se situe dans ses ***capacités d'auto-adaptation*** au cours de l'apprentissage. Il s'agit d'une ***architecture évolutive*** dont les 3 couches sont définies de la façon suivante :

- *Couche d'entrée*

La ***couche d'entrée*** comporte autant de neurones que de composantes au sein du vecteur observation $X^i = \left[x_1^i \cdots x_d^i \cdots x_D^i\right]^T$ défini dans l'espace $\Re^D$. La dimension de cette couche est donc fixée initialement, par opposition aux dimensions de la couche cachée et de la couche de sortie qui évoluent en fonction de la création de prototypes et de classes.

- *Couche cachée*

La ***couche cachée*** comporte autant de neurones que de prototypes existants à un instant donné. Chaque prototype est caractérisé par un neurone sur la couche cachée. La fonction d'activation du neurone j traduit la fonction d'appartenance μ_j des observations au prototype P^j (figure IV.2). Ainsi, le terme $\mu_j\left(X^i\right)$ exprime le degré d'appartenance de l'observation X^i au prototype P^j et sa valeur est déterminée par :

$$\mu\left(P^j, X^i\right) = \mu_j\left(X^i\right) = exp\left(-\frac{d\left(X^i, P^j\right)^2}{2}\right) \qquad \text{(IV.1)}$$

Le choix de cette fonction d'activation correspond à la définition de ***prototypes gaussiens***. Par ailleurs, afin de respecter la distribution des données dans toutes les directions de l'espace de représentation, la distance utilisée dans l'expression (IV.1) est la distance de Mahalanobis :

$$d\left(X^i, P^j\right)^2 = \left(X^i - M_{P^j}\right)^T \Sigma_{P^j}^{-1}\left(X^i - M_{P^j}\right) \qquad \text{(IV.2)}$$

où les termes M_{P^j} et Σ_{P^j} représentent respectivement le centre et la matrice de covariance du prototype P^j. Cette matrice de covariance (pleine) permet de caractériser une ***zone d'influence de forme hyperelliptique*** (figure IV.3 avec $D = 2$).

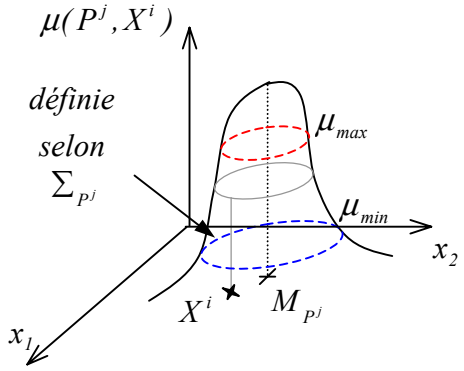


Figure IV.3 : Illustration 2D d'un prototype gaussien avec sa région d'influence hyperelliptique et sa fonction d'appartenance gaussienne.

Trois niveaux d'appartenance sont identifiés par comparaison de $\mu_j\left(X^i\right)$ à deux seuils notés μ_{min} et μ_{max} et illustrés figure IV.3. ***Le seuil μ_{min} peut être interprété comme le seuil minimum d'appartenance à une classe. Le seuil μ_{max} peut, quant à lui, être interprété comme le seuil minimum d'appartenance à un prototype***.

Une observation $X^i \in \Re^D$ est alors affectée à la classe à laquelle est associé le prototype P^j si son degré d'appartenance $\mu_j\left(X^i\right)$ est supérieur à μ_{min}. De plus, si $\mu_j\left(X^i\right)$ est supérieur à μ_{max} alors l'observation X^i est associée au prototype P^j et contribue à sa définition.

La couche cachée est entièrement connectée à la couche d'entrée. Ces connexions sont pondérées par les coefficients de la matrice $W^P = \left[w_{jd}^P\right]_{j\in[1\ldots J],\, d\in[1\ldots D]}$ qui mémorisent les coordonnées des centres des J prototypes existants. ***La dimension J de la couche cachée n'est pas figée et évolue avec le nombre de prototypes***.

➢ *Couche de sortie*

La ***couche de sortie*** comporte autant de neurones que de classes connues. Chaque classe est caractérisée par un neurone sur la couche de sortie. La fonction d'activation du neurone k traduit la fonction d'appartenance ψ_k des observations à la classe C^k. Cette fonction utilise un opérateur de type t-conorme (max, somme bornée, …) basé sur les fonctions d'appartenance μ_j

des prototypes P^j caractérisant la classe C^k. Ainsi, le terme $\psi_k\left(X^i\right)$ exprime le degré d'appartenance de l'observation X^i à la classe C^k. Dans le cadre du développement de notre algorithme, l'opérateur « somme bornée » (IV.3) a été préféré à l'opérateur « max » afin de limiter l'apparition d'« îlots d'appartenance » au sein des classes composées de plusieurs prototypes.

$$\psi\left(C^k, X^i\right) = \psi_k\left(X^i\right) = min\left(1, \sum_{P^j \in C^k} \mu_j\left(X^i\right)\right) \qquad \text{(IV.3)}$$

La figure IV.4 justifie ce choix en illustrant la définition des modèles des prototypes (figure IV.4.a) caractérisant une seule et même classe, et en exposant la définition du modèle de la classe en considérant l'opérateur « max » (figure IV.4.b) puis l'opérateur « somme bornée » (figure IV.4.c).

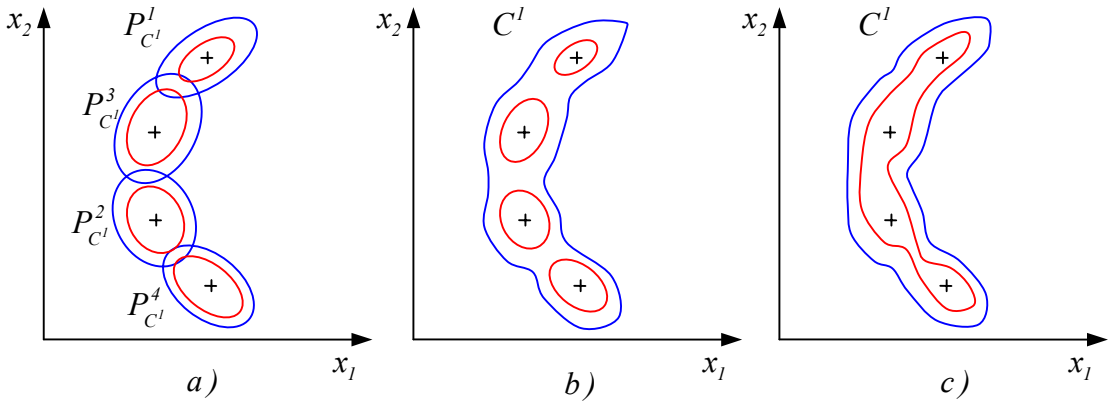


Figure IV.4 : Illustration de deux niveaux d'appartenance : a) pour les prototypes P^j de la classe C^1, puis pour la classe C^1 selon b) l'opérateur "max" et c) l'opérateur "somme bornée".

La couche de sortie est entièrement connectée à la couche cachée à l'aide de la matrice binaire $W^C = \left[w^C_{kj}\right]_{k\in[1\ldots K],\, j\in[1\ldots J]}$ qui mémorisent les associations des J prototypes existants aux K classes existantes. L'association du prototype P^j à la classe C^k, notée par la suite $P^j_{C^k}$, se traduit alors par l'égalité $w^C_{kj} = 1$ et sinon $w^C_{kj} = 0$. Tout comme pour la couche cachée, la dimension K de la couche de sortie n'est pas figée et évolue avec le nombre de classes.

2.2. Mode d'apprentissage séquentiel

L'architecture proposée pour la définition du réseau AUDyC a la particularité d'être ***évolutive***. Les dimensions de la couche cachée et de la couche de sortie ne sont pas fixées initialement. Le nombre de neurones constituant ces couches évolue en fonction de la caractérisation des prototypes et/ou des classes lors de la présentation d'observations nouvelles. Cette adaptation progressive de l'architecture est rendue possible grâce à l'introduction de règles d'auto-adaptation et à l'exploitation d'un ***processus d'apprentissage cyclique***. Ces règles autorisent notamment la création de nouveaux neurones sur la couche cachée et sur la couche de sortie. De plus, elles intègrent des principes d'adaptation des paramètres des prototypes.

L'architecture initiale du réseau est définie figure IV.5 en considérant uniquement la première observation. En fait, en l'absence de classes et de prototypes, la présentation de l'observation X^1 donne lieu à la définition de la première classe C^1 caractérisée par un unique prototype P^1. La définition de P^1 puis de C^1 se traduit alors respectivement par la création du premier neurone sur la couche cachée et du premier neurone sur la couche de sortie.

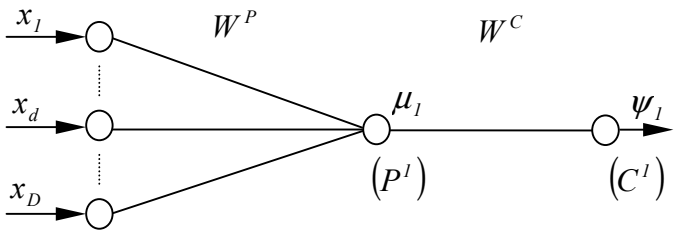


Figure IV.5 : Architecture initiale du réseau.

Les paramètres M_{P^1} et $\sum_{P^1}$ du prototype P^1 sont initialisés selon les expressions suivantes :

$$M_{P^1} = X^1 \quad \text{et} \quad \sum_{P^1} = \sum_{ini} \tag{IV.4}$$

Les coordonnées du centre M_{P^1} du prototype P^1 sont définies à l'aide de l'observation X^1. La matrice de covariance $\sum_{P^1}$ du prototype P^1 est quant à elle initialisée par une matrice $\sum_{ini} = \sigma_{ini}^2 . I_D$ (où la notation I_D exprime la matrice identité en dimension D) en considérant le même écart-type σ_{ini} dans toutes les directions. Les coordonnées du centre de P^1 sont mémorisées au sein de la matrice de pondération W^P :

$$W^P = \left[M_{P^1}\right]^T = \left[X^1\right]^T \tag{IV.5}$$

L'association du prototype P^1 à la classe C^1 ($P^1_{C^1}$) est mémorisée par la matrice de pondération W^C :

$$W^C = [1] \tag{IV.6}$$

L'architecture est alors créée de façon progressive au fur et à mesure de la présentation des observations. Dans ce sens, les ***cycles du processus d'apprentissage*** sont définis en respectant les différents principes et concepts exposés au §1. Les paragraphes suivants présentent l'ensemble des règles d'apprentissage non supervisé utilisées au sein de ce processus cyclique. Pour cela, nous avons développé notamment des procédures de ***création de neurones*** (prototypes et/ou classes) et d'***adaptation des paramètres*** des prototypes existants ce qui nécessite de ***modifier les pondérations du réseau***. De plus, afin d'améliorer la modélisation des classes, la ***fusion de prototypes*** a été introduite tout comme la ***fusion des classes*** qualifiées proches. Enfin, la ***suppression*** de neurones permet d'éliminer certains prototypes ou classes peu représentatifs afin d'optimiser la dimension de l'architecture.

L'exploitation de ce processus cyclique permet alors de mettre en œuvre un mode d'apprentissage séquentiel, en réitérant plusieurs fois celui-ci avec des ensembles de données nouvelles.

2.2.1. Processus d'apprentissage cyclique

Le processus d'apprentissage du réseau AUDyC, inspiré de celui de l'algorithme CDL [ELT98], suit un ***processus cyclique*** décomposé en trois phases : une phase de ***classification***, une phase de ***fusion*** et une phase d'***évaluation***. Pour chaque ensemble de données considéré, ce processus cyclique permet une optimisation progressive de la structure de classes. A chaque cycle d'apprentissage (figure IV.6), un ensemble X^{Cycle} d'observations est considéré, puis traité successivement par les 3 phases définies précédemment. Cet ensemble X^{Cycle} est alors actualisé à l'issue de chaque cycle en fonction des résultats de classification. Lors du premier cycle d'apprentissage, X^{Cycle} correspond à l'ensemble X des observations à classer.

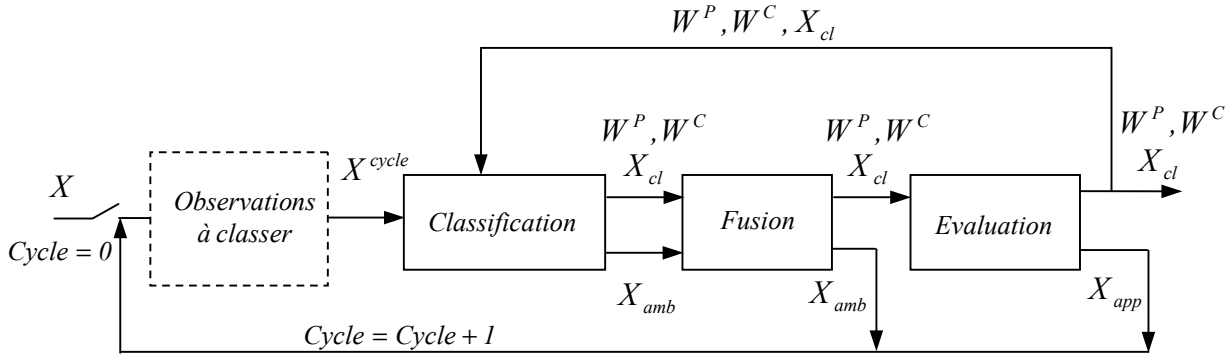


Figure IV.6 : Processus d'apprentissage cyclique du réseau AUDyC.

Pour chaque cycle ($Cycle$), les observations de X^{Cycle} sont dans un premier temps exploitées au cours de la ***phase de classification***. Celles-ci sont présentées au réseau les unes après les autres. Ainsi, lors de la présentation de l'observation $X^i \in X^{Cycle}$, ses degrés d'appartenance aux prototypes existants sont évalués. Par comparaison aux seuils μ_{min} et μ_{max}, différentes situations sont envisageables selon le niveau d'appartenance de X^i. Chacune de ces situations est gérée par un ensemble de règles d'apprentissage non supervisé permettant soit de créer des nouveaux neurones (prototypes et/ou classes), soit d'adapter les paramètres des prototypes existants. A l'issue de cette première phase, l'architecture neuronale définit plusieurs classes caractérisées par plusieurs prototypes. La définition des prototypes est mémorisée à l'aide de la matrice W^P et des matrices de covariance $\sum_{P^j}$ propres à chaque prototype. De même, la définition des classes est mémorisée au sein de W^C. Un ***ensemble d'observations classées*** X_{cl} est défini, ainsi qu'un ***ensemble d'observations écartées*** X_{amb}. Ce dernier est constitué des observations qui n'ont pu être traitées lors de la première phase d'apprentissage en raison de leurs ressemblances à plusieurs classes.

L'***architecture neuronale est alors adaptée en considérant l'ensemble*** X_{amb} des observations écartées précédemment. Cette deuxième phase se traduit par une ***phase de fusion*** au cours de laquelle les classes proches sont regroupées. Bien que plusieurs critères soient envisageables pour juger de cette proximité, en pratique, nous avons utilisé un critère basé sur le nombre d'observations ambiguës. Après fusion des classes, les observations ambiguës sont regroupées dans X_{amb} afin d'être reconsidérées lors du cycle suivant ($Cycle + 1$).

Enfin, chaque cycle d'apprentissage se conclut par une ***phase d'évaluation*** permettant d'***éliminer les prototypes et les classes qualifiés de peu représentatifs*** de la structure de classes. A cet effet, en pratique, nous avons considéré des critères de cardinalité. ***Les observations initialement affectées aux prototypes et aux classes éliminées sont rejetées au sein d'un ensemble*** X_{app}, puis sont reconsidérées lors du cycle suivant ($Cycle+1$).

Après le premier cycle d'apprentissage, l'architecture du réseau AUDyC est paramétrée par les coefficients des matrices W^P et W^C. Pour chaque prototype, une matrice de covariance mémorise sa forme et son orientation. Enfin, on dispose d'un ensemble d'observations X_{cl} labellisées en terme de classes alors que les observations de $\{X_{amb}, X_{app}\}$ ne sont pas classées. Celles-ci doivent être présentées à nouveau jusqu'à ce que toutes les observations de X ou tout au moins un pourcentage soient classées, ou que les ensembles X_{amb} et X_{app} se stabilisent.

2.2.2. 1ère phase : Classification

La phase de ***classification*** a pour objectif de ***définir et d'adapter l'architecture du réseau*** afin d'optimiser la modélisation des classes en fonction des observations présentées. Pour cela, la particularité de cette première phase se situe dans sa capacité à ***adapter les paramètres des prototypes existants*** et/ou à ***créer des nouveaux neurones*** (prototypes et classes) si l'observation présentée apporte une information supplémentaire.

Pour chaque observation $X^i \in X^{Cycle}$ présentée à l'entrée du réseau, on évalue ses degrés d'appartenance $\mu_j(X^i)$ aux différents prototypes P^j connus. Comme nous l'avons indiqué précédemment, par comparaison aux seuils μ_{min} et μ_{max} (figure IV.3), l'interprétation du niveau d'appartenance de X^i donne lieu à différentes situations d'apprentissage. Afin de mieux apprécier la description des règles d'apprentissage propres à chacune de ces situations, considérons qu'avant la présentation de l'observation X^i ($D=2$), la structure de classes est composée de 3 classes caractérisées par 4 prototypes (figure IV.7.a). Ceci signifie que l'architecture est composée d'une couche d'entrée à 2 neurones, d'une couche cachée à 4 neurones et d'une couche de sortie à 3 neurones (figure IV.7.b).

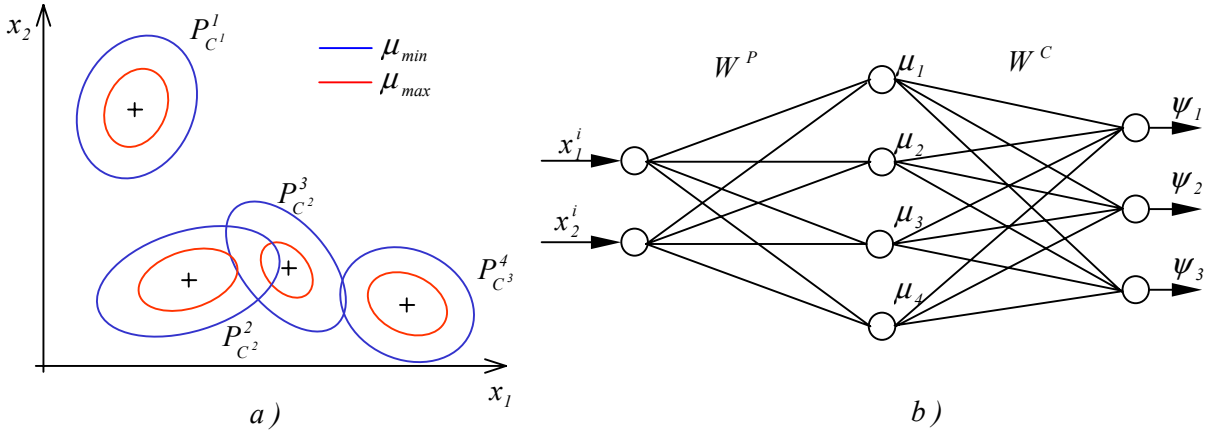


Figure IV.7 : a) Illustration d'une structure en 3 classes et 4 prototypes, b) Architecture du réseau avant la présentation de l'observation X^i.

La figure IV.7.a illustre les 3 classes modélisées à l'aide de 4 prototypes pour lesquels seuls les centres et les contours (niveaux d'appartenance correspondant aux deux seuils d'appartenance) ont été représentés. Le prototype P^1 est l'unique représentant de la classe C^1. Les prototypes P^2 et P^3 caractérisent la classe C^2. Le prototype P^4 est l'unique représentant de la classe C^3. Les matrices de pondération W^P et W^C mémorisent respectivement les centres des 4 prototypes et les liens de ces 4 prototypes aux 3 classes. Celles-ci sont définies par :

$$W^P = \left[M_{P^1} \quad M_{P^2} \quad M_{P^3} \quad M_{P^4} \right]^T \tag{IV.7}$$

$$W^C = \begin{matrix} (P^1) & (P^2) & (P^3) & (P^4) \end{matrix} \begin{matrix} \\ \\ \\ \end{matrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{matrix} (C^1) \\ (C^2) \\ (C^3) \end{matrix} \tag{IV.8}$$

2.2.2.1. <u>1er cas</u> : $\mu(P^j, X^i) < \mu_{min} \quad \forall \; P^j$

Cette première situation illustre le cas où l'observation X^i présentée au réseau est trop éloignée des prototypes existants. Son degré d'appartenance $\mu_j(X^i)$ est inférieur à μ_{min} quel que soit le prototype P^j considéré. En d'autres termes, celle-ci ne présente ***aucune ressemblance avec les classes existantes*** et ne peut donc être affectée à aucune d'entre elles. Une ***nouvelle classe*** est alors créée. Celle-ci est caractérisée par un ***nouveau prototype*** dont la position est définie par X^i (expression IV.4). Par ailleurs, l'observation X^i est affectée à la nouvelle classe puis mémorisée au sein de l'ensemble X_{cl} des observations classées. Le principe d'adaptation de l'architecture est décrit en reprenant la situation initiale de la figure IV.7 et en considérant que l'observation X^i présentée se situe dans une région de l'espace de représentation éloignée de celles caractérisées par les 3 classes existantes (figure IV.8.a).

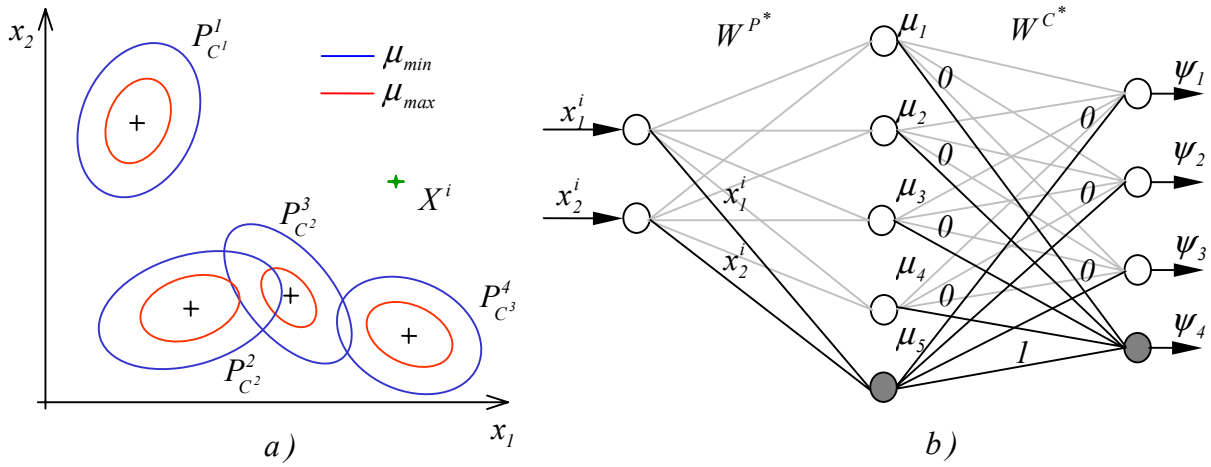


Figure IV.8 : a) Présentation d'une observation X^i éloignée des classes existantes, b) Création d'un nouveau prototype et d'une nouvelle classe.

Une nouvelle classe C^4 est définie par la création d'un $4^{ième}$ neurone sur la couche de sortie (en gris foncé). Celle-ci est modélisée par un unique prototype P^5 caractérisé par un $5^{ième}$

neurone sur la couche cachée (en gris foncé). La position du prototype P^5 est définie par les coordonnées de l'observation X^i et sa matrice de covariance est initialisée par une matrice initiale $\sum_{ini}$ (expression IV.4). D'un point de vue structurel, la création du prototype et de la classe engendre une adaptation de l'architecture neuronale en provoquant l'ajout d'un neurone sur la couche cachée et sur la couche de sortie (figure IV.8.b). Cette adaptation de l'architecture neuronale s'accompagne d'une modification des matrices de pondération W^P et W^C :

$$W^{P^*} = \begin{array}{c} \begin{matrix} x_1 & x_2 \end{matrix} \\ \left[\begin{array}{cc} & \\ \multicolumn{2}{c}{W^P} \\ & \\ \hline x_1^i & x_2^i \end{array}\right] \end{array} \begin{matrix} \\ \\ \\ \\ (P^5) \end{matrix} \qquad \text{(IV.9)}$$

$$W^{C^*} = \begin{array}{c} \\ \left[\begin{array}{ccc|c} & & & 0 \\ & W^C & & \vdots \\ & & & 0 \\ \hline 0 & \cdots & 0 & 1 \end{array}\right] \end{array} \begin{matrix} (P^5) \\ \\ \\ \\ (C^4) \end{matrix} \qquad \text{(IV.10)}$$

Les expressions (IV.9) et (IV.10) décrivent respectivement l'adaptation de W^P pour mémoriser les coordonnées du centre du nouveau prototype P^5, et de W^C pour mémoriser l'association entre le nouveau prototype P^5 et la nouvelle classe C^4.

2.2.2.2. <u>2ème cas</u> : $\mu_{min} < \mu(P^j, X^i) < \mu_{max}$ pour un ou plusieurs prototypes appartenant tous à la classe C^k

Par opposition au paragraphe précédent, ce deuxième cas illustre la situation où l'observation X^i présentée est proche d'un ou plusieurs prototypes P^j associés à une unique classe C^k. En fait, le degré d'appartenance $\mu_j(X^i)$ est supérieur à μ_{min} pour des prototypes $P^j_{C^k}$, mais reste inférieur à μ_{max} quel que soit le prototype $P^j_{C^k}$ considéré. En d'autres termes, l'***observation*** X^i ***est proche de ces prototypes*** $P^j_{C^k}$ et donc de la classe C^k, ***mais pas suffisamment proche pour être associée à ces derniers et donc contribuer à leur définition***. L'observation est toutefois affectée à la classe C^k et à l'ensemble X_{cl} des observations classées. Par ailleurs, l'observation X^i est considérée comme apportant une information supplémentaire à la définition de la classe C^k et engendre par conséquent la définition d'un ***nouveau prototype*** pour la classe C^k. Pour illustrer cette situation, reprenons l'exemple de la structure de classes décrite figure IV.7 en considérant que l'observation X^i présentée apparaît cette fois-ci ***à proximité du prototype*** P^1 associé à la classe C^1 (en gris clair sur la figure IV.9.b). L'observation X^i présente donc bien un degré d'appartenance au prototype P^1 supérieur à μ_{min} mais celui-ci reste inférieur à μ_{max} (figure IV.9.a).

Un nouveau prototype P^5 est alors défini par la création d'un $5^{ième}$ neurone sur la couche cachée (en gris foncé), puis associé à la classe C^1. La position du prototype P^5 est définie par les coordonnées de l'observation X^i et sa matrice de covariance est initialisée par une matrice initiale $\sum_{ini}$ (expression IV.4). La création du prototype P^5 vise à compléter le modèle de la classe C^1.

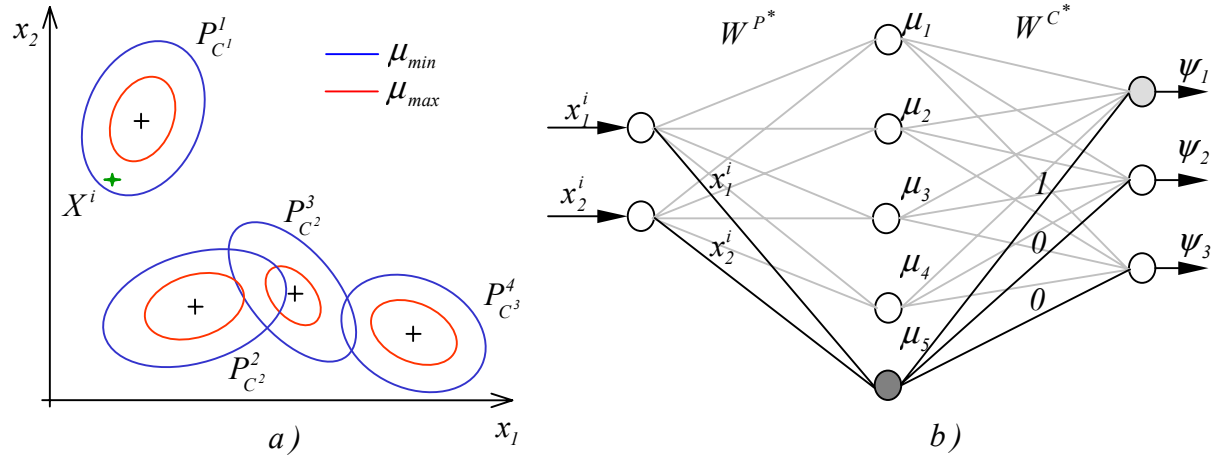


Figure IV.9 : a) Présentation d'une observation X^i proche du prototype P^1 de la classe C^1 mais pas assez pour être associée au prototype P^1, b) Création d'un prototype P^5 pour la classe C^1.

D'un point de vue structurel, la définition du prototype P^5 engendre une adaptation de l'architecture neuronale afin d'ajouter un neurone sur la couche cachée. Cette adaptation de l'architecture s'accompagne d'une modification des matrices W^P et W^C :

$$W^{P^*} = \begin{matrix} x_1 & x_2 \\ \left[\begin{array}{cc} W^P & \\ \hline x_1^i & x_2^i \end{array}\right. & \left.\begin{array}{c} \\ \\ P^5 \end{array}\right. \end{matrix} \qquad \text{(IV.11)}$$

$$W^{C^*} = \begin{matrix} & P^5 \\ \left[\begin{array}{cc} W^C & \\ & \end{array}\right. & \left.\begin{array}{c} 1 \\ 0 \\ \vdots \\ 0 \end{array}\right] \end{matrix} \begin{matrix} \\ C^1 \end{matrix} \qquad \text{(IV.12)}$$

Les expressions (IV.11) et (IV.12) décrivent respectivement le principe d'adaptation de W^P pour mémoriser la position du centre du prototype P^5 et de W^C pour mémoriser l'association entre le nouveau prototype P^5 et la classe C^1 existante.

2.2.2.3. $3^{ème}$ cas : $\mu_{min} < \mu_{max} < \mu(P^j, X^i)$ pour un ou plusieurs prototypes appartenant tous à la classe C^k

La situation décrite ici peut être vue comme une extension de la précédente. En effet, cette troisième situation reprend le cas où l'observation X^i présentée au réseau est proche d'un ou plusieurs prototypes P^j associés à une unique classe C^k dans le sens où son degré d'appartenance $\mu_j(X^i)$ est supérieur à μ_{min} pour les prototypes $P^j_{C^k}$ considérés. Toutefois, par opposition à la situation précédente, le degré d'appartenance $\mu_j(X^i)$ est supérieur à μ_{max} pour certains prototypes $P^j_{C^k}$. En d'autres termes, l'***observation*** X^i ***est proche de prototypes*** $P^j_{C^k}$ et donc de la classe C^k, et en plus elle est ***suffisamment proche de certains prototypes*** $P^j_{C^k}$ (ceux pour lesquels elle présente un degré d'appartenance supérieur à μ_{max}) ***pour participer à leur définition***. Par conséquent, l'observation est affectée à la classe C^k ainsi qu'à l'ensemble X_{cl} des observations classées. De plus, elle ne provoque pas la création d'un nouveau prototype mais engendre une ***adaptation des paramètres des prototypes*** $P^j_{C^k}$ pour lesquels elle présente un degré d'appartenance supérieur à μ_{max}. Sur ce point, deux cas de figures peuvent être distingués : soit l'observation présente un degré d'appartenance supérieur à μ_{max} ***pour un unique prototype*** de la classe C^k, soit l'observation présente un degré d'appartenance supérieur à μ_{max} ***pour plusieurs prototypes*** de la classe C^k.

➢ *Pour un unique prototype P^j de la classe C^k*

La figure IV.10 reprend la structure de classes décrite figure IV.7 en considérant l'apparition d'une observation X^i ***au proche voisinage du prototype*** P^1 associé à la classe C^1 (en gris clair sur la figure IV.10.b). L'observation X^i présente donc un degré d'appartenance supérieur à μ_{max} pour l'unique prototype P^1 de la classe C^1.

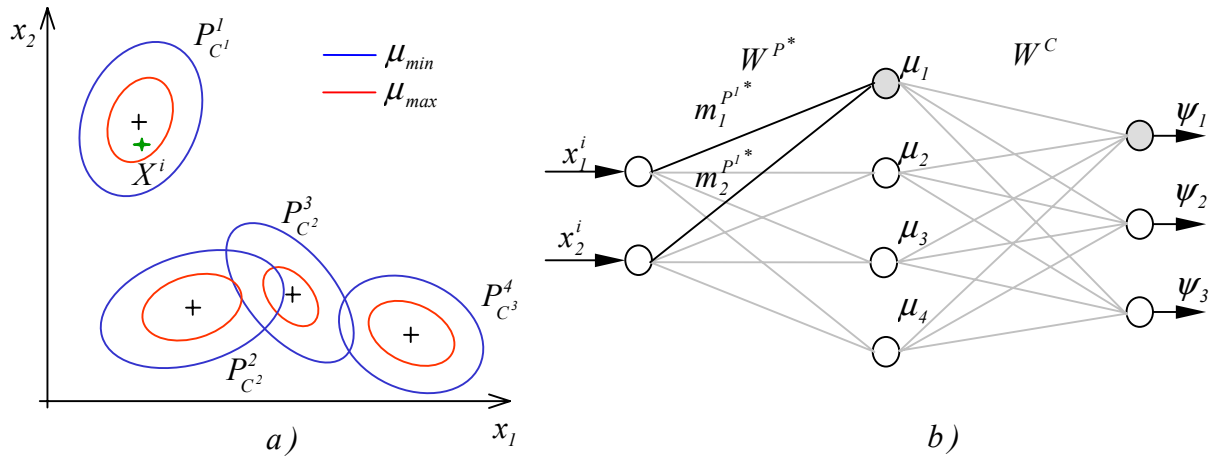


Figure IV.10 : a) Présentation d'une observation X^i suffisamment proche du prototype P^1 pour être associée à celui-ci et affectée à la classe C^1 qu'il caractérise, b) Adaptation des paramètres du prototype P^1.

Dans cette situation, les paramètres du prototype P^l sont adaptés avec l'information apportée par l'observation X^i. D'un point de vue général, les règles d'adaptation des paramètres M_{P^j} et $\sum_{P^j}$ d'un prototype P^j avec une observation X^i s'expriment à l'aide des relations itératives suivantes (voir annexe 3) :

$$M_{P^j}^{(nb+1)} = \frac{nb}{nb+1}.M_{P^j}^{(nb)} + \frac{1}{nb+1}X^i \qquad \text{(IV.13)}$$

$$\sum_{P^j}^{(nb+1)} = \frac{nb-1}{nb}.\sum_{P^j}^{(nb)} + \frac{1}{nb+1}.\left(X^i - M_{P^j}^{(nb)}\right)\left(X^i - M_{P^j}^{(nb)}\right)^T \qquad \text{(IV.14)}$$

Ces relations sont en fait des estimateurs non biaisés de la moyenne et de la matrice de covariance d'un prototype. Les notations $P^{j(nb)}$ et $P^{j(nb+1)}$ permettent de distinguer le prototype avant et après adaptation. Le terme nb spécifie quant à lui le nombre d'observations associées à P^j avant son adaptation avec X^i. La figure IV.11 illustre en dimension $D=2$ le ***principe d'adaptation du prototype*** P^j avec une observation X^i.

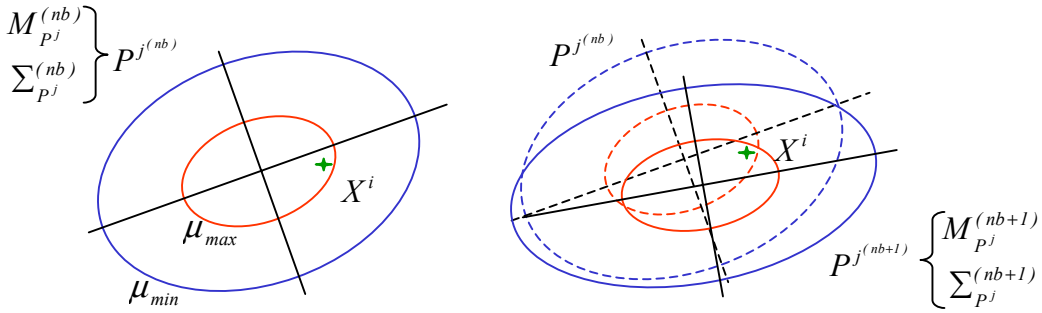


Figure IV.11 : Illustration de l'adaptation d'un prototype P^j avec une nouvelle observation X^i.

Les règles d'adaptation proposées pour les prototypes ont l'avantage d'être présentées sous une forme itérative. Chaque observation associée à un prototype P^j contribue à améliorer la définition de celui-ci en terme de position (M_{P^j}), de forme et d'orientation ($\sum_{P^j}$). Ces règles d'adaptation permettent d'améliorer en continu la qualité de modélisation des prototypes. Par ailleurs, suite à l'adaptation d'un prototype, aucune modification de la structure du réseau n'est requise, seule la matrice W^P est adaptée afin de tenir compte de l'évolution du prototype P^l :

$$W^{P^*} = \begin{array}{c} \begin{matrix} x_1 & x_2 \end{matrix} \\ \begin{bmatrix} m_1^{P^{l^*}} & m_2^{P^{l^*}} \\ m_1^{P^2} & m_2^{P^2} \\ m_1^{P^3} & m_2^{P^3} \\ m_1^{P^4} & m_2^{P^4} \end{bmatrix} \end{array} \begin{matrix} (P^{l^*}) \\ (P^2) \\ (P^3) \\ (P^4) \end{matrix} \qquad \text{(IV.15)}$$

➢ *Pour plusieurs prototypes P^j de la classe C^k*

Considérons maintenant que l'observation X^i présente un degré d'appartenance supérieur à μ_{max} pour plusieurs prototypes de la classe C^k. Ceci signifie que la classe C^k est modélisée par

plusieurs prototypes $P_{C^k}^j$ dont certains (ceux présentant une valeur d'appartenance pour X^i supérieure à μ_{max}) se « chevauchent » ou se « recouvrent ». ***Différentes solutions sont alors envisageables*** : soit on adapte uniquement le prototype pour lequel le degré d'appartenance de l'observation est le plus grand parmi les $P_{C^k}^j$ concernés, soit on adapte l'ensemble des prototypes $P_{C^k}^j$ concernés, soit on fusionne l'ensemble des prototypes $P_{C^k}^j$ concernés.

La ***première solution*** est probablement la plus simple. En effet, le fait de ne considérer que le prototype offrant le plus grand degré d'appartenance pour l'observation présentée revient à la situation précédente où l'observation a un degré d'appartenance supérieure à μ_{max} pour un unique $P_{C^k}^j$. Le prototype « vainqueur » est adapté, mais aucune modification n'est apportée aux autres prototypes concernés. Rien ne justifie toutefois de privilégier le prototype « vainqueur ». De plus, cette solution ne résout pas le problème de chevauchement des prototypes concernés.

La ***deuxième solution*** consiste à adapter l'ensemble des prototypes $P_{C^k}^j$ concernés à l'aide des relations (IV.13) et (IV.14). Cette décision signifie que l'on considère que l'observation présentée contribue à l'adaptation des paramètres de l'ensemble des prototypes concernés et ceci avec la même importance pour chaque prototype. Si cette situation se présente plusieurs fois au cours d'un cycle d'apprentissage alors les prototypes concernés vont tendre à modéliser une même région de données et donc devenir similaires. La deuxième étape de cette solution consiste alors à ne conserver qu'un seul des prototypes similaires après complète adaptation puis à éliminer les autres. Cependant, cette approche nécessite d'élaborer une procédure permettant de spécifier si des prototypes sont similaires, de la même façon que l'algorithme URCP (§1.2.3 du chapitre 3) évalue la similarité entre classes.

La ***troisième solution*** s'appuie davantage sur le fait que l'occurrence d'une observation dans une région d'appartenance supérieure au seuil μ_{max} pour plusieurs prototypes d'une même classe signifie que ceux-ci modélisent une région commune de la classe. Il est alors légitime de remettre en cause la nécessité de l'existence des différents prototypes actuels. Dans ce sens, la solution proposée vise à favoriser la définition d'un nouveau prototype en remplacement des différents prototypes concernés. Celui-ci est défini par fusion des prototypes $P_{C^k}^j$ considérés. Cette approche a l'avantage d'optimiser le nombre de prototypes pour la caractérisation d'une classe et propose un traitement immédiat des situations de chevauchement des prototypes.

Ces trois solutions ont été expérimentées puis comparées en terme de qualité de modélisation et de nombre de prototypes créés. Suite à ces expérimentations, ***nous avons opté pour le principe de fusion des prototypes*** qui optimise la forme et le nombre de prototypes.

La figure IV.12 illustre la fusion de deux prototypes P^{j1} et P^{j2} associés à la même classe et présentant une zone de recouvrement pour des degrés d'appartenance supérieurs à μ_{max} (zone hachurée). La procédure de fusion donne lieu à la définition d'un nouveau prototype noté P^{j1^*} dont les paramètres sont obtenus par adaptation des paramètres de P^{j1} avec l'ensemble des observations du prototype P^{j2} à l'aide des relations (IV.13) et (IV.14).

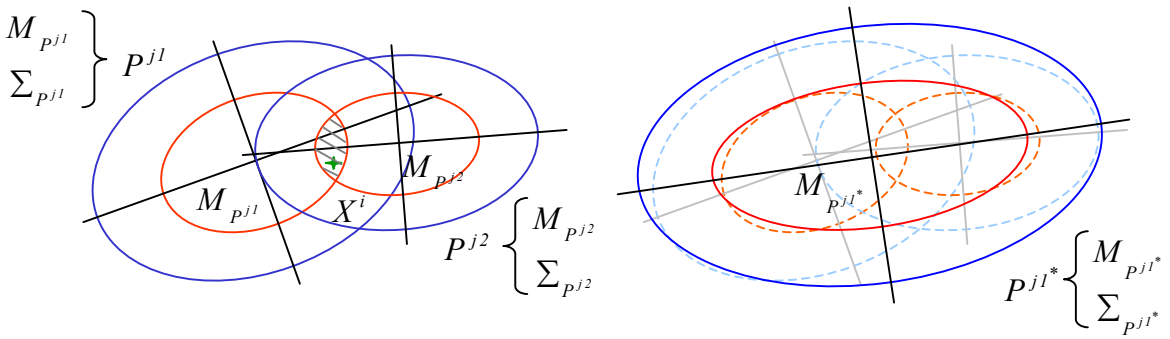


Figure IV.12 : Illustration de la fusion d'un prototype P^{j1} et d'un prototype P^{j2} pour former l'unique prototype P^{j1^*}.

L'architecture neuronale est adaptée en supprimant les neurones des prototypes concernés (P^{j1}, P^{j2}), puis en créant un neurone pour le prototype résultant P^{j1^*} et en associant ce dernier à la même classe que les anciens prototypes. Les matrices W^P et W^C sont alors adaptées.

2.2.2.4. *4ème cas* : $\mu_{min} < \mu(P^j, X^i)$ *pour des prototypes de plusieurs classes*

Cette dernière règle illustre le cas où l'observation X^i présentée au réseau est qualifiée de proche de plusieurs classes dans le sens où son degré d'appartenance $\mu_j(X^i)$ est supérieur à μ_{min} pour des prototypes P^j de classes différentes. Cette situation soulève alors une ambiguïté d'affectation entre plusieurs classes.

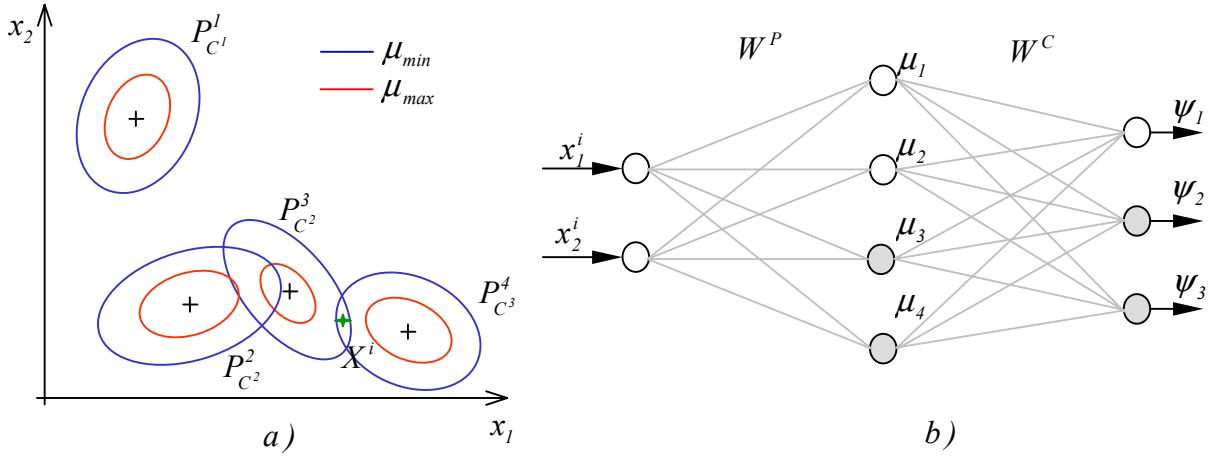


Figure IV.13 : a) Présentation d'une observation X^i proche de prototypes de classes différentes, b) L'observation X^i est dite ambigüe puisqu'elle active simultanément les sorties de plusieurs classes (C^2 et C^3).

La figure IV.13 illustre cette situation à partir de la structure de classes décrite figure IV.7 en considérant que l'observation X^i présente un degré d'appartenance supérieur à μ_{min} pour les prototypes P^3 et P^4 appartenant respectivement aux classes C^2 et C^3 (en gris clair sur la figure IV.13.b). Cette proximité simultanée de l'observation X^i vis-à-vis de deux classes traduit une ambiguïté d'affectation. L'observation est alors écartée au sein d'un ensemble noté X_{amb} qui est constitué de toutes les observations ambiguës puis considéré lors de la phase de fusion.

A l'issue de la phase de classification, un ensemble de prototypes et de classes est défini. Ceci se traduit par une architecture du réseau AUDyC où la couche cachée et la couche de sortie ont évolué (ainsi que les matrices de pondération W^P et W^C) afin d'intégrer des nouvelles connaissances. Par ailleurs, sur l'ensemble des données présentées au réseau, un ensemble X_{cl} d'observations a été classé et un ensemble X_{amb} d'observations a été écarté pour des raisons d'ambiguïté. L'étape suivante consiste donc à traiter ces différents problèmes d'ambiguïté. Pour cela et pour poursuivre avec notre exemple, considérons qu'à l'issue de la phase de classification, l'architecture du réseau AUDyC est composée de 4 classes caractérisées par 6 prototypes.

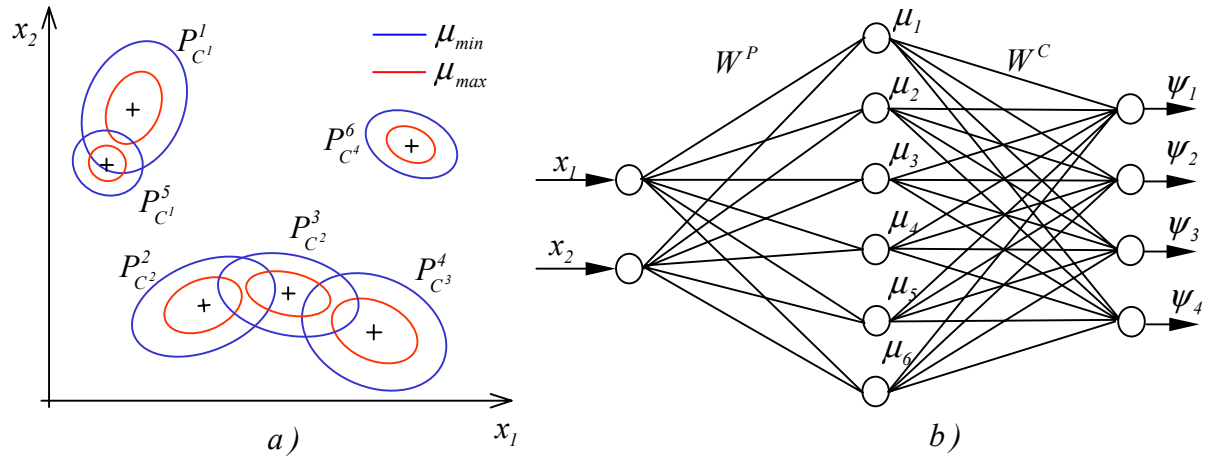


Figure IV.14 : a) Illustration de la structure en 4 classes et 6 prototypes obtenue à l'issue de la phase de classification, b) Définition de l'architecture neuronale associée.

La figure IV.14 donne une représentation en terme de position et de forme des 6 prototypes créés avec leur étiquette d'appartenance aux 4 classes. Comme on peut le voir les prototypes initiaux (figure IV.7) ont évolué en terme de position, d'orientation et de forme en fonction des données qui leur ont été associées. De plus, de nouveaux prototypes sont apparus pour compléter le modèle de certaines classes ou tout simplement pour créer une nouvelle classe. La classe C^1 est désormais modélisée par les prototypes P^1 et P^5. La classe C^2 est modélisée par les prototypes P^2 et P^3. La classe C^3 est modélisée par l'unique prototype P^4. Une nouvelle classe est apparue, la classe C^4 modélisée par l'unique prototype P^6. En accord avec la définition de cette structure de classes, les matrices de pondération W^P et W^C sont désormais définies de la façon suivante :

$$W^P = \begin{bmatrix} M_{P^1} & M_{P^2} & M_{P^3} & M_{P^4} & M_{P^5} & M_{P^6} \end{bmatrix}^T \qquad \text{(IV.16)}$$

$$W^C = \begin{array}{c} \begin{matrix} (P^1) & (P^2) & (P^3) & (P^4) & (P^5) & (P^6) \end{matrix} \\ \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \end{array} \begin{matrix} (C^1) \\ (C^2) \\ (C^3) \\ (C^4) \end{matrix} \qquad \text{(IV.17)}$$

Sur la base de l'architecture obtenue à l'issue de la phase de classification (figure IV.14), le processus d'apprentissage se poursuit avec les phases de fusion et d'évaluation.

2.2.3. 2ème phase : Fusion

La phase de classification a permis d'élaborer une structure de classes en créant un ensemble de prototypes caractérisant les différentes classes. Toutefois, toutes les observations ne sont pas classées à l'issue de cette première phase, certaines d'entre elles, reprises dans X_{amb}, présentent une ambiguïté d'appartenance vis-à-vis de plusieurs classes. Une ***phase de fusion*** a donc été élaborée afin de gérer ces situations d'ambiguïté.

La présence d'observations ambiguës dans une région située entre des classes distinctes peut être révélatrice d'une probable erreur de modélisation lors des étapes précédentes. En effet, après la phase de classification, une classe réelle peut être caractérisée par plusieurs petites classes en raison du principe de création des prototypes et des classes. En fait, il suffit qu'une observation d'une classe donnée apparaisse dans une région non identifiée de cette classe pour que celle-ci soit considérée comme la première occurrence d'une nouvelle classe.

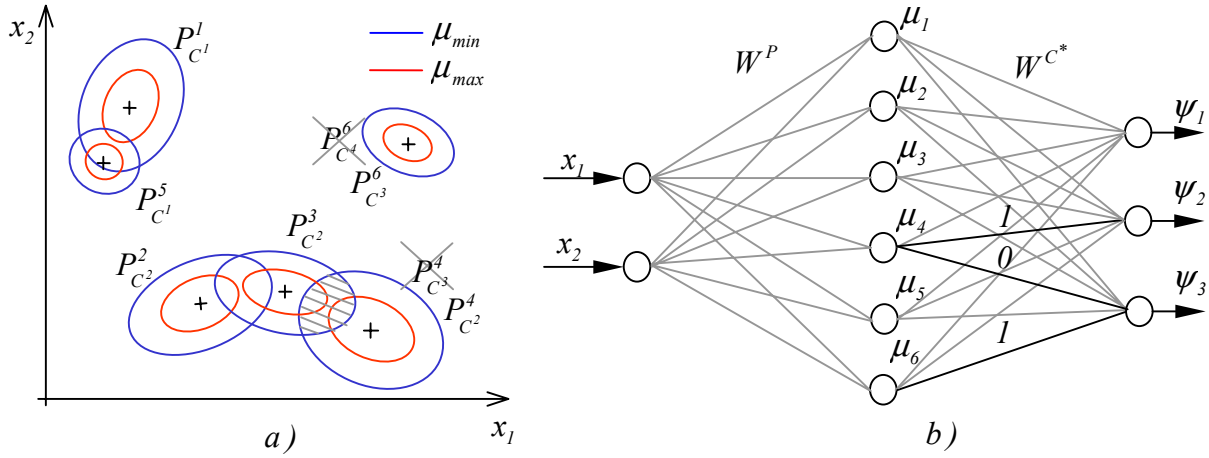


Figure IV.15 : a) Traitement d'une situation d'ambiguité entre deux classes (C^2 et C^3), b) Adaptation de l'architecture par fusion des 2 classes.

Dans le cadre de notre exemple, la figure IV.15.a illustre l'existence d'une zone d'ambiguïté entre les classes C^2 et C^3 (zone hachurée). Cette zone a été identifiée sur la base des observations associées à l'ensemble X_{amb}. La présence d'une seule observation dans cette zone ne justifie pas la fusion des deux classes. En fait, comme introduit au §2.2.1, la détermination d'une ambiguïté entre deux classes est établie à partir d'un seuil n_{amb} sur le nombre d'observations ambiguës entre celles-ci bien que d'autres critères soient envisageables. On dit alors qu'il y a ambiguïté entre les classes C^{k1} et C^{k2} si :

$$Card\left(\left\{X^i \in X_{amb} \,/\, X^i \in C^{k1} \text{ et } X^i \in C^{k2}\right\}\right) \geq n_{amb} \qquad \text{(IV.18)}$$

Si l'expression (IV.18) est vérifiée alors le processus de fusion des classes C^{k1} et C^{k2} est activé. Dans la situation de la figure IV.15, cela se traduit par la fusion des classes C^2 et C^3 pour former la « nouvelle » classe C^2. Par ailleurs, afin de respecter la numérotation des classes,

la classe C^4 devient la classe C^3 mais conserve les mêmes caractéristiques. Par conséquent, l'architecture du réseau est adaptée (figure IV.15.b). Le prototype P^4 de l'ancienne classe C^3 est associé à la nouvelle classe C^2 ce qui se traduit par les relations $w_{3,4}^{C\,*} = 0$ et $w_{2,4}^{C\,*} = 1$. L'ancienne classe C^3 est supprimée ce qui se traduit par la suppression du neurone correspondant sur la couche de sortie. L'ancienne classe C^4 devient alors la nouvelle classe C^3 ($w_{3,6}^{C\,*} = 1$). La matrice W^P est inchangée par contre la matrice W^C est modifiée afin de tenir compte de la fusion des classes C^2 et C^3 :

$$W^{C^*} = \begin{matrix} \\ (C^1) \\ (C^2) \\ (C^3) \end{matrix} \begin{matrix} \begin{matrix} (P^1) & (P^2) & (P^3) & (P^4) & (P^5) & (P^6) \end{matrix} \\ \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \end{matrix} \qquad \text{(IV.19)}$$

D'un point de vue pratique, l'écriture de la matrice W^{C^*} est obtenue en remplaçant la ligne 2 de W^C (IV.17) par l'union des lignes 2 et 3 de W^C (IV.17). Cette opération traduit le fait que l'ensemble des prototypes des classes fusionnées appartienne au modèle de la classe résultante. Par ailleurs, l'ancienne ligne 4 de W^C devient la ligne 3 de W^{C^*}. Ceci étant, les observations de X_{amb} sont reconsidérées lors du cycle d'apprentissage suivant ($Cycle + 1$).

2.2.4. 3ème phase : Evaluation

La phase de fusion propose une solution aux situations d'ambiguïté entre les classes mises en évidence lors de la phase de classification. Après fusion, la structure de classes n'est toutefois pas encore optimale puisqu'elle comporte toujours des prototypes et des classes peu représentatifs. Cette situation s'explique par le fait qu'un prototype tout comme une classe peuvent être créés à partir d'une seule et unique observation (§2.2.2.1). Cependant, l'existence réelle de telles classes est difficilement envisageable, ne serait-ce que par le fait que leur prise en compte engendre une dimension importante de l'architecture neuronale. Afin d'éviter une telle situation, une ***phase d'évaluation des prototypes et des classes*** a été élaborée.

Plusieurs critères d'évaluation sont possibles (densité, éloignement, …), toutefois comme indiqué au §2.2.1, nous avons retenu des critères de cardinalité. La cardinalité $Card(P^j)$ d'un prototype P^j est égale au nombre d'observations associées à celui-ci, i.e. ayant contribué à l'adaptation de ses paramètres. Sur ce principe, la cardinalité $Card(C^k)$ d'une classe C^k est égale à la somme des cardinalités de ses prototypes. Ces cardinalités sont évaluées pour l'ensemble des prototypes et des classes puis comparées respectivement à deux seuils notés N_{min}^P et N_{min}^C. Les prototypes et les classes vérifiant les expressions (IV.20) et (IV.21) sont qualifiés de peu représentatifs et sont éliminés.

$$Card(P^j) < N_{min}^P \qquad \text{(IV.20)}$$

$$Card(C^k) < N_{min}^C \qquad \text{(IV.21)}$$

Cette élimination de prototypes et de classes se traduit par une simplification de l'architecture neuronale par suppression des neurones correspondants sur la couche cachée et la couche de sortie. Les observations associées aux prototypes et classes éliminées sont rejetées au sein d'un ensemble noté X_{app} et sont retirées de l'ensemble X_{cl}. Ceci étant, les observations de X_{app} sont reconsidérées lors du cycle d'apprentissage suivant ($Cycle+1$).

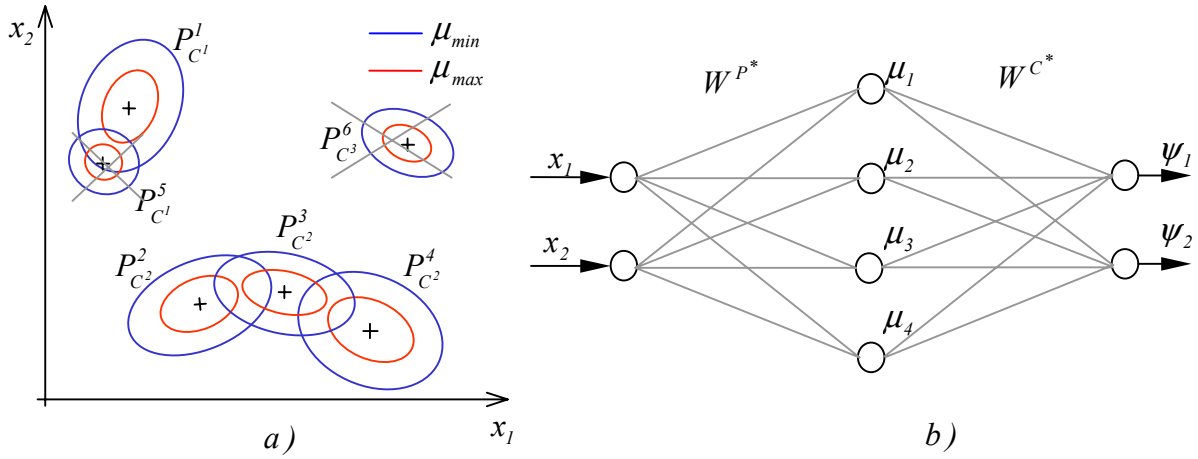


Figure IV.16 : a) Evaluation de la cardinalité des prototypes et des classes, b) Adaptation de l'architecture par suppression des prototypes P^5 et P^6 et de la classe C^3 trop peu représentatifs.

Le processus d'évaluation puis de suppression de neurones est décrit à l'aide de l'exemple de la figure IV.16 où les 6 prototypes illustrés sont ceux mis en évidence après la phase de fusion (figure IV.15). Sur la figure IV.16.a, les prototypes et les classes ne satisfaisant pas les contraintes de cardinalité (IV.20) et (IV.21) sont identifiés et « repérés » à l'aide d'une croix. Ainsi, la cardinalité du prototype P^5 est inférieure à N_{min}^P. De même, la cardinalité de la classe C^3 est inférieure à N_{min}^C bien que la cardinalité du prototype $P_{C^3}^6$ puisse être supérieure au seuil N_{min}^P. La phase d'évaluation optimise alors le résultat de modélisation en supprimant le prototype P^5 et la classe C^3 ainsi que le prototype P^6 associé à cette dernière. Par conséquent, la classe C^1 n'est plus représentée que par l'unique prototype P^1. Par ailleurs, la suppression du prototype P^6 dont l'effectif est toutefois supérieur au seuil N_{min}^P est réalisée puisque celui-ci n'est plus associé à l'une des classes. La suppression des neurones correspondants aux prototypes et aux classes éliminés provoque une simplification de l'architecture (figure IV.16.b). Les neurones 5 et 6 de la couche cachée (figure IV.15.b) ont été supprimés ainsi que le neurone 3 de la couche de sortie (figure IV.15.b). Enfin, les matrices de pondération sont adaptées en fonction des prototypes et classes éliminés :

$$W^{P^*} = \begin{bmatrix} M_{P^1} & M_{P^2} & M_{P^3} & M_{P^4} \end{bmatrix}^T \qquad \text{(IV.22)}$$

$$W^{C^*} = \begin{matrix} (P^1) & (P^2) & (P^3) & (P^4) \end{matrix} \\ \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix} \begin{matrix} (C^1) \\ (C^2) \end{matrix} \qquad \text{(IV.23)}$$

2.2.5. Bouclage des cycles d'apprentissage

Les trois paragraphes précédents ont permis d'exposer les différentes ***règles d'apprentissage non supervisé*** utilisées lors des phases de classification, de fusion et d'évaluation caractérisant un ***cycle d'apprentissage*** du réseau AUDyC (figure IV.6).

Lors de la ***phase de classification***, les règles exploitées autorisent l'adaptation des paramètres de la modélisation existante, mais aussi la création de prototypes et de classes (nouveaux neurones) pour intégrer l'apparition de nouvelles informations et ainsi compléter la modélisation. En fait, ces règles traitent successivement les observations de X^{cycle} selon leur ordre de présentation à la couche d'entrée du réseau. Lors des ***phases de fusion et d'évaluation***, les règles d'apprentissage optimisent les résultats de classification et de modélisation en regroupant les classes proches et en supprimant les prototypes et les classes peu représentatifs.

Toutefois, après la réalisation du premier cycle d'apprentissage, selon les 3 phases décrites précédemment, toutes les observations de l'ensemble X initial ne sont pas forcément étiquetées. Des observations (X_{amb}) ont été écartées pour des raisons d'ambiguïté entre classes lors de la phase de classification. De même, des observations (X_{app}) ont été rejetées suite à l'élimination des prototypes ou des classes auxquelles elles étaient associées lors de la phase d'évaluation.

Afin de considérer ces observations, ***le processus d'apprentissage adopte une structure bouclée.*** En effet, pour un même ensemble de données, les cycles d'apprentissage se succèdent jusqu'à ce que ***toutes les observations ou tout au moins un pourcentage de celles-ci soient étiquetées ou que les résultats de classification convergent***, i.e. que les ensembles X_{amb} et X_{app} conservent la même définition après plusieurs cycles consécutifs. A chaque nouveau cycle d'apprentissage, les observations présentées au réseau sont regroupées au sein d'un ensemble X^{cycle} constitué des ensembles X_{amb} et X_{app} établis au cycle précédent. Ainsi, la structure cyclique du processus d'apprentissage a pour effet d'accroître le taux de classification après chaque cycle, en complétant l'ensemble X_{cl} des observations labellisées et en optimisant l'architecture neuronale.

Cependant, bien que le taux de classification soit correct à l'issue du processus d'apprentissage, ***les modèles obtenus pour la définition des classes ne sont pas toujours optimaux***. En effet, la réalisation de plusieurs cycles d'apprentissage ne permet pas forcément d'identifier immédiatement les modèles réels des classes. Ce phénomène peut s'expliquer par le fait que la convergence des relations d'adaptation (IV.13) et (IV.14) vers les valeurs réelles de la moyenne et de la matrice de covariance d'un prototype n'est assurée que si le nombre nb d'observations associées à celui-ci est suffisant (§2.2.2.3). En fait, la principale explication émane de la possible non exhaustivité de la base de données initiales ou de la taille réduite de cette dernière par rapport au nombre de classes représentées.

La solution consiste alors à exploiter les ***capacités d'auto-adaptation*** de l'architecture. Pour cela, il suffit d'exploiter plusieurs fois le ***processus d'apprentissage cyclique*** en présentant régulièrement au réseau un ensemble de données nouvelles. Dans le cadre de données stationnaires, ce principe a pour effet d'optimiser la position et la forme des prototypes tout en

forçant une certaine stabilisation de leurs paramètres et d'améliorer le résultat de classification entre deux réalisations consécutives du processus d'apprentissage. Par ailleurs, ce mode d'apprentissage, que l'on peut qualifier de séquentiel, permet de limiter l'influence de l'ordre de présentation des observations au réseau. De plus, cela tend à positionner les prototypes sur les régions de données les plus denses et donc à toujours écarter *a priori* les mêmes observations après convergence de la modélisation.

2.3. Validation des capacités d'auto-adaptation de l'architecture

Cette troisième partie a pour but de ***valider les qualités d'apprentissage de l'architecture*** du réseau AUDyC en éprouvant ses capacités d'auto-adaptation au cours de différents cycles de son processus d'apprentissage. Pour cela, nous avons utilisé un ensemble de données stationnaires, décrit figure IV.17. Celui-ci est constitué de 411 observations ($D = 2$) structurées en 3 classes présentant des formes de « croissant ». Cette structure de classes a été choisie afin d'éprouver les capacités de l'algorithme d'apprentissage du réseau AUDyC à définir une classification et une modélisation correcte en présence de classes de forme complexe.

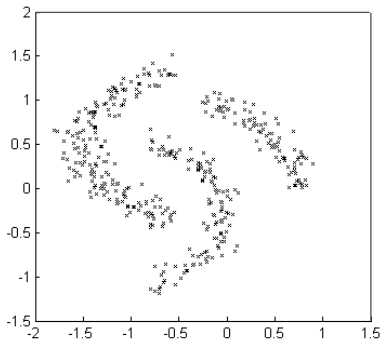


Figure IV.17 : Données expérimentales.

Le choix des différents paramètres et seuils nécessaires à l'utilisation du réseau AUDyC est réalisé en tenant compte de l'influence de chacun d'entre eux et de la nature des données considérées. Ainsi, les ***seuils d'appartenance*** μ_{min} ***et*** μ_{max} et le ***paramètre initial*** σ_{ini} sont notamment déterminés en fonction de la variance au sein des différentes classes et traduisent la taille de la zone d'influence des prototypes. Un premier paragraphe décrit alors le principe d'***utilisation du réseau AUDyC*** dans le cadre des données de la figure IV.17. L'aspect évolutif de l'architecture du réseau AUDyC est illustré à l'aide de la présentation de l'évolution des résultats de classification et de modélisation après chaque cycle du processus d'apprentissage. Un deuxième paragraphe expose ***l'influence de l'ordre de présentation des données*** sur la convergence de la classification. Ce test permet également de vérifier la stabilisation puis la convergence de la modélisation après plusieurs présentations du même ensemble de données. Un troisième paragraphe ***évalue la sensibilité du réseau AUDyC à la présence de données bruitées***. Enfin, un dernier paragraphe ***compare les résultats du réseau AUDyC à d'autres algorithmes*** dynamiques tels que les réseaux de neurones CDL et FMMC décrits au chapitre 3.

2.3.1. Utilisation de l'algorithme neuronal AUDyC

Ce premier paragraphe présente les résultats de classification des données de la figure IV.17, obtenus à l'aide de l'algorithme AUDyC en considérant les paramètres suivants : $\mu_{min} = 0.1$; $\mu_{max} = 0.3$; $\sigma_{ini} = 0{,}13$; $n_{amb} = 2$; $N^{P}_{min} = 7$ et $N^{C}_{min} = 10$. Le tableau IV.1 décrit ***les résultats de classification et de modélisation après les deux premiers cycles du processus d'apprentissage, puis après le 5ème cycle***.

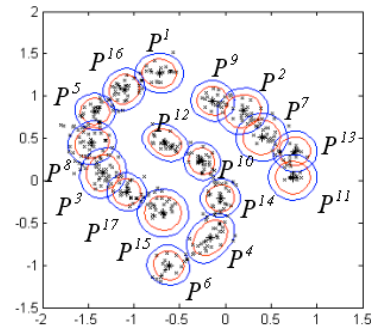
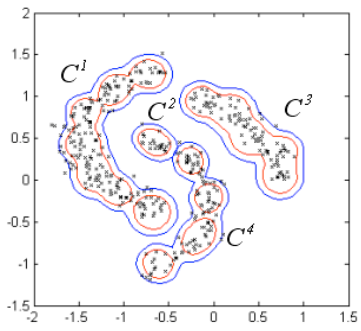
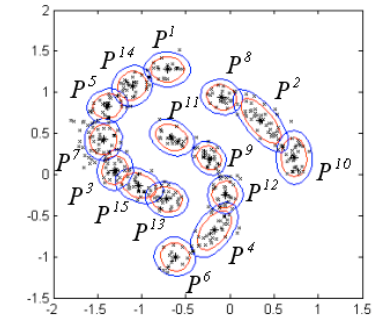
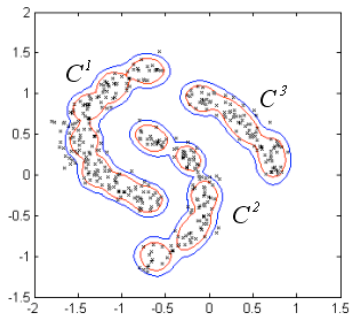
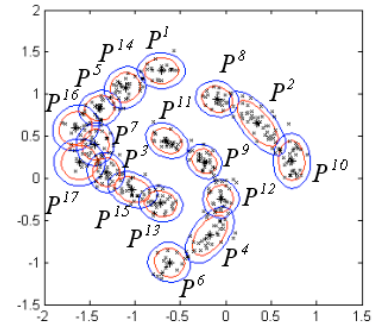
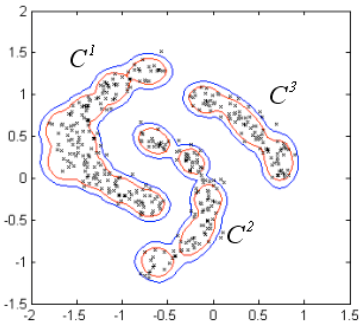
Processus d'apprentissage n°1		
	 	*Cycle n°1* *Nb prototypes* { 32 Classification ; 32 Fusion ; 17 Evaluation } *Nb classes* { 23 Classification ; 5 Fusion ; 4 Evaluation } $X_{cl} : 56\%$
	 	*Cycle n°2* *Nb prototypes* { 31 Classification ; 31 Fusion ; 15 Evaluation } *Nb classes* { 9 Classification ; 6 Fusion ; 3 Evaluation } $X_{cl} : 86\%$
	⋮	⋮
	 	*Cycle n°5* *Nb prototypes* { 32 Classification ; 32 Fusion ; 17 Evaluation } *Nb classes* { 8 Classification ; 4 Fusion ; 3 Evaluation } $X_{cl} : 94\%$

Tableau IV.1 : Résultats de classification et de modélisation à différents stades de l'apprentissage du réseau AUDyC.

Les résultats présentés indiquent une évolution du taux de classification après chaque cycle d'apprentissage ainsi qu'une adaptation progressive des modèles des prototypes et des classes. La colonne de droite précise, pour chaque cycle, les évolutions du nombre de prototypes et de classes entre les différentes phases d'apprentissage. A l'issue du 5ème cycle du processus d'apprentissage, ***les résultats de classification ont convergé pour atteindre un taux de classification de 94%***. En fait, si l'on considère les modèles obtenus et une règle de décision basée sur le maximum d'appartenance, alors on peut affecter correctement la totalité des

observations. ***La structure finale est formée de 3 classes caractérisées par 17 prototypes***. Cependant, certains prototypes ont conservé leur forme hypersphérique initiale, notamment au niveau de la classe C^1. Prenons l'exemple des prototypes P^{16} et P^{17} (cycle n°5), ceux-ci ont été créés lors des derniers cycles d'apprentissage afin d'intégrer une nouvelle connaissance propre à la classe C^1, mais non pas achevés leur processus d'adaptation. Comme indiqué au §2.2.5, l'apprentissage peut être poursuivi en présentant à nouveau l'ensemble des données considérées, ce qui a pour effet d'***améliorer la modélisation***. En fait, ce principe correspond à un ***mode d'apprentissage séquentiel*** de l'architecture permettant une optimisation périodique de cette dernière.

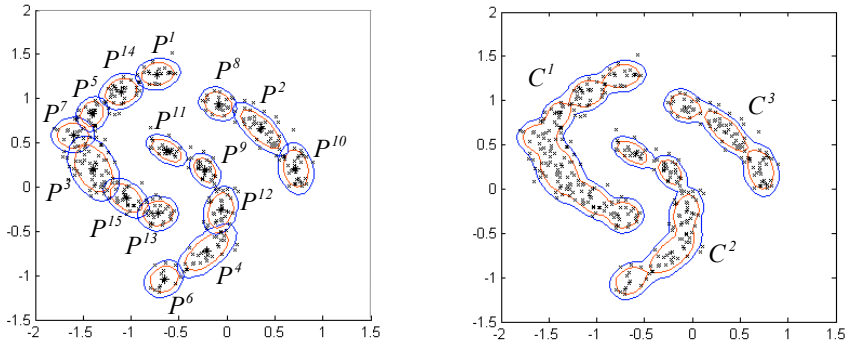


Figure IV.18 : Résultats obtenus après une 3ème présentation des données : a) Modèles des prototypes, b) Modèles des classes.

Ainsi, la figure IV.18 illustre les résultats de modélisation à l'issue de la ***troisième réalisation du processus d'apprentissage***. Les 3 classes sont désormais caractérisées par un ensemble de 15 prototypes, optimisés en terme de position et de forme. Ces derniers ont l'avantage de modéliser plus fidèlement les frontières des classes.

2.3.2. Influence de l'ordre de présentation des données

Les capacités d'auto-adaptation de l'architecture évolutive développée pour le réseau AUDyC viennent d'être illustrées en vérifiant, de façon expérimentale, la convergence des résultats de classification et de modélisation. Afin de conforter les qualités de notre réseau, ce deuxième paragraphe expose une expérimentation au cours de laquelle nous avons testé l'***influence de l'ordre de présentation des données*** sur les résultats précédents.

En ***terme de classification***, les différentes expériences réalisées ont montré que quel que soit l'ordre de présentation des données, ***l'algorithme d'apprentissage converge vers un résultat de classification identique***.

En ***terme de modélisation***, le tableau IV.2 décrit les modèles obtenus en considérant 3 ordres de présentation des observations. La colonne de gauche expose les modèles initiaux des prototypes, i.e. obtenus à l'issue de la 1ère réalisation du processus d'apprentissage. La colonne de droite expose les modèles finaux des classes, i.e. obtenus après plusieurs réalisations du processus d'apprentissage, soit jusqu'à stabilisation et convergence de la modélisation.

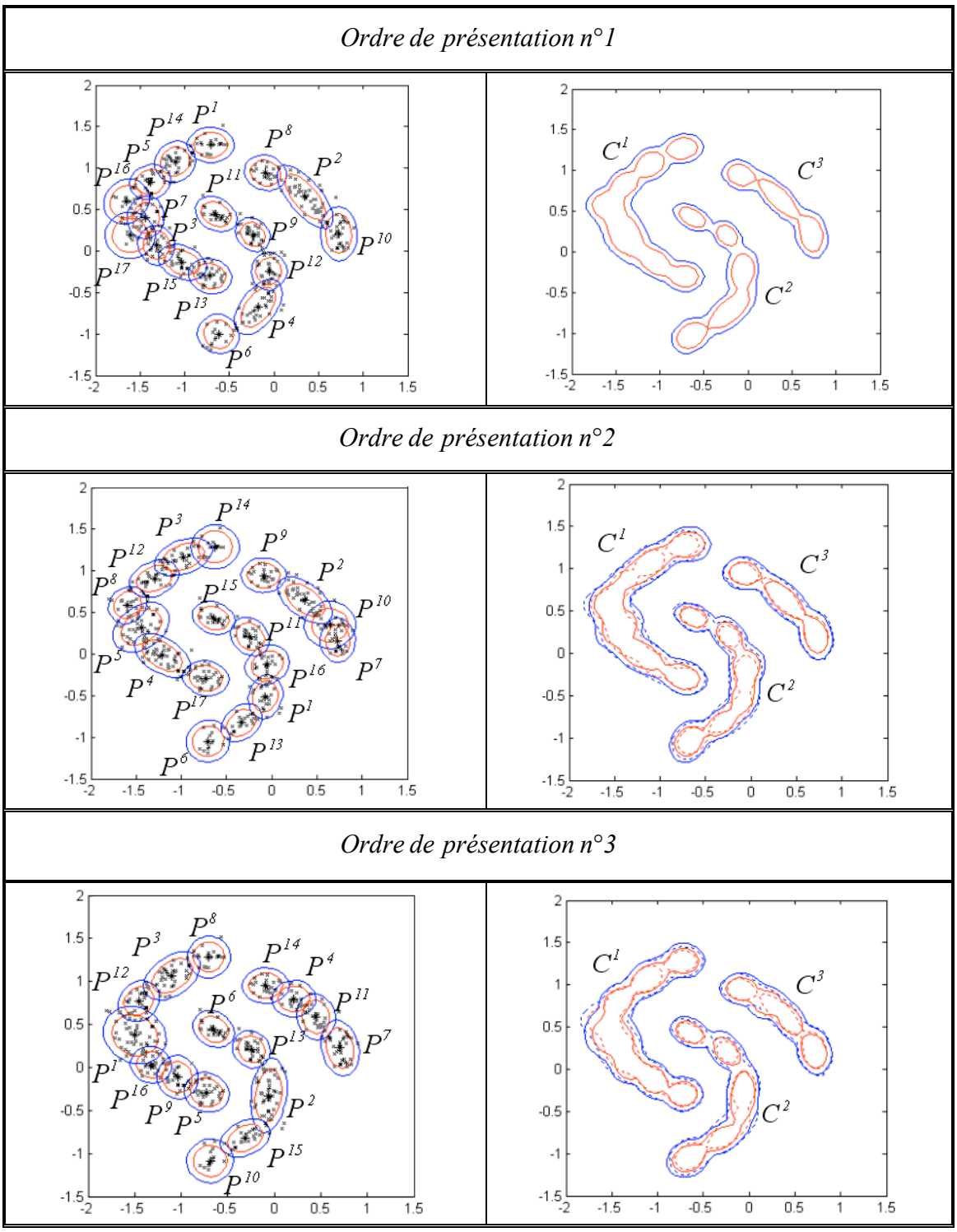


Tableau IV.2 : Convergence de la modélisation du réseau AUDyC pour trois ordres différents de présentation des données.

Comme l'illustre le tableau IV.2, en dépit des différences de création initiale des prototypes, après plusieurs exécutions du processus d'apprentissage, ***les modèles de classes obtenus dans les cas n°2 et n°3 sont très proches de ceux obtenus lors du cas n°1 (en pointillés).*** Cette expérimentation décrit les capacités de ***convergence de la classification des observations*** et de ***stabilisation de la modélisation des classes*** du réseau AUDyC, quel que soit l'ordre de présentation des observations.

2.3.3. Influence de la présence de bruit sur les données

Cette troisième expérimentation vise à évaluer la capacité de l'algorithme d'apprentissage du réseau AUDyC à produire une classification correcte d'un ensemble de données en présence d'observations bruitées, tout en identifiant correctement les modèles de classes. Pour cela, nous avons utilisé les observations de la figure IV.17 auxquelles nous avons adjoint environ 150 données bruitées (illustrées en rouge sur la figure IV.19) ce qui représente un ***niveau de bruit de l'ordre de 30%***. Dans une telle situation, le seuil d'ambiguïté n_{amb} révèle toute son importance et évite de fusionner des classes réelles. Par ailleurs, en présence d'observations bruitées, il peut également être intéressant de définir un seuil d'évaluation N_{min}^{P} plus important. En fait, ceci permet d'éviter la création intempestive de prototypes en périphérie des classes, ceux-ci pouvant induire une fusion indésirable de classes réelles.

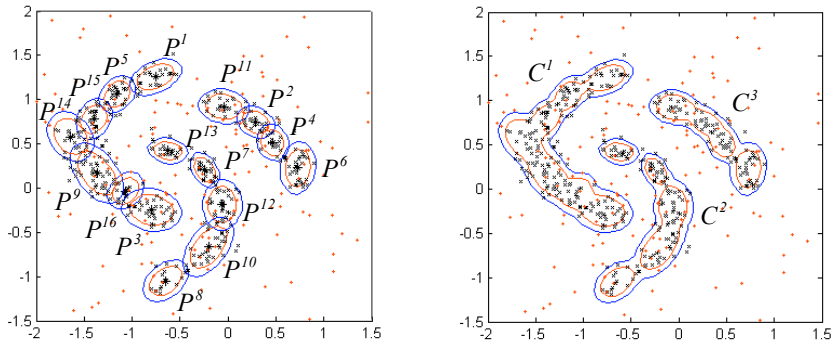
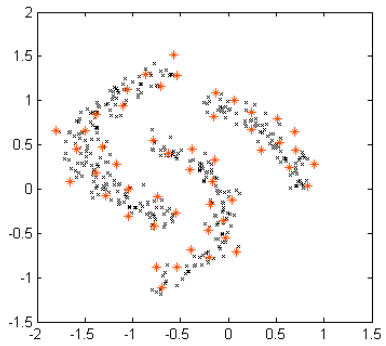


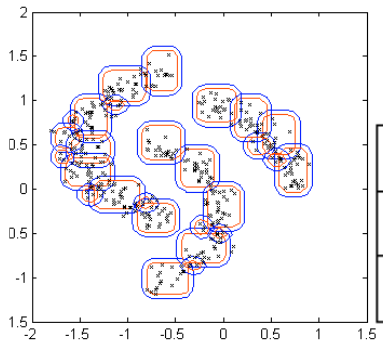
Figure IV.19 : Modélisation obtenue en présence de bruit sur les données : a) Modèles des prototypes, b) Modèles des classes.

La figure IV.19 illustre les prototypes et classes obtenus en initialisant l'algorithme avec les mêmes paramètres que précédemment mais ***en initialisant n_{amb} à 6*** en raison du niveau de bruit au sein des données. Le ***taux de classification*** au sein de X_{cl} atteint un niveau de 70%, ce qui correspond en réalité à près de ***90% des observations réelles***. Les 3 classes sont correctement caractérisées (figure IV.19.b) et modélisées par 16 prototypes (figure IV.19.a). Cependant, on peut constater l'apparition d'un îlot d'appartenance au niveau de la classe C^2 avec le prototype P^{13}. En fait, ceci peut s'expliquer par la faible densité d'observations dans cette région de C^2 par rapport au reste de la classe.

2.3.4. Comparaison avec les algorithmes CDL et FMMC

Cette dernière expérimentation a pour but de comparer les performances de l'algorithme AUDyC à celles de deux autres algorithmes dynamiques disposant de capacités d'auto-adaptation : les algorithmes CDL et FMMC (chapitre 3). La figure IV.20 illustre les prototypes obtenus à l'aide de l'algorithme CDL (figure IV.20.a) puis avec l'algorithme FMMC (figure IV.20.b) sur la base du meilleur compromis lors du choix de leurs paramètres d'initialisation. La figure IV.20.c compare alors les performances du CDL, du FMMC et du réseau AUDyC développé.





	Nombre de prototypes	Taux de classification
CDL	50	99%
FMMC	27	93%
AUDyC	15	94%

a) *b)* *c)*

Figure IV.20 : Comparaison des performances du réseau AUDyC à celles des réseaux neuronaux CDL et FMMC.

L'***algorithme CDL*** est initialisé à l'aide des seuils suivants : $\xi_{min} = 15$ et $\xi_{max} = 30$ pour les seuils de similarité et $N_{min}^{C} = 10$ pour l'évaluation des classes. L'apprentissage des trois classes entraîne la définition de ***50 prototypes*** illustrés en rouge sur la figure IV.20.a. Le ***taux de classification est bon*** (figure IV.20.c), mais en raison du ***nombre important de prototypes créés***, l'architecture neuronale présente une couche cachée de grande dimension. Cette quantité de prototypes est due au fait que ***nombre d'entre eux sont créés sur la périphérie des classes*** et ne fournissent pas une réelle modélisation de ces dernières.

L'***algorithme FMMC*** est initialisé en prenant $\gamma = 4$ pour le paramètre de pente des modèles des prototypes hypercubiques et $\theta = 0.3$ pour le seuil de vigilance nécessaire au principe d'expansion. L'apprentissage des trois classes entraîne la définition de ***81 prototypes*** illustrés par deux niveaux d'appartenance sur la figure IV.20.b. En fait, seuls les 27 prototypes pour lesquels le processus d'expansion a été activé sont représentés. Si l'on ne considère que les prototypes ayant atteint un effectif significatif, ***le taux de classification est de 93%***. Cependant, d'une part, le ***nombre de prototypes créés est important,*** et d'autre part, parmi les 27 prototypes ayant suivi le processus d'expansion, ***certains sont peu significatifs de la distribution des données***. En fait, le processus d'expansion des prototypes vise davantage à étendre leur zone d'influence afin qu'ils « englobent » l'ensemble des données qui leur sont associées, qu'à modéliser la distribution de données. De plus, la forme hypercubique des prototypes ne permet pas d'identifier précisément les contours propres à chacune des classes.

En conclusion, bien que les réseaux CDL et FMMC disposent de bonnes capacités de classification (figure IV.20.c), ils restent moins performants que le réseau AUDyC du point de vue de la modélisation des classes.

2.4. Conclusion

Les deux premiers paragraphes de cette deuxième partie ont proposé une description du réseau de neurones auto-adaptatif développé : ***le réseau AUDyC***. L'***architecture du réseau*** (§2.1) et les ***règles d'apprentissage*** (§2.2) ont été établies en tenant compte des propriétés énoncées en introduction du chapitre. La plupart des propriétés sont vérifiées et l'accent a été porté sur l'introduction de ***capacités d'auto-adaptation*** ce qui a permis de définir une

architecture évolutive. Les prototypes et les classes sont créés lors d'un processus d'apprentissage cyclique décomposé en 3 phases : ***classification***, ***fusion*** et ***évaluation***. Les règles d'apprentissage permettent notamment la création de nouveaux prototypes et de nouvelles classes ainsi que l'adaptation des prototypes et des classes existants. De plus, des règles de fusion de prototypes et/ou de classes ont été définies, tout comme des règles d'élimination des prototypes et des classes peu représentatifs de la distribution de données.

Suite à la description de l'architecture et du processus d'apprentissage cyclique du réseau AUDyC, le paragraphe 2.3 a permis de ***valider les capacités de classification et de modélisation de l'architecture proposée*** sur un ensemble de données stationnaires. Après différentes expérimentations, il a été vu que ***l'algorithme AUDyC converge vers une classification correcte des observations*** et qu'après plusieurs présentations d'un ensemble de données (mode d'apprentissage séquentiel), ***les modèles de classes convergent vers une modélisation correcte des classes***. Par ailleurs, il a été montré, de façon expérimentale, que le résultat de classification du réseau AUDyC est peu sensible à ***l'ordre de présentation des données***. Une troisième expérimentation a confirmé ***ces aptitudes en présence d'un nuage d'observations bruitées***. Enfin, par opposition aux algorithmes CDL et FMMC, l'algorithme AUDyC a l'avantage de ***restreindre le nombre de prototypes créés*** et permet ***une meilleure modélisation des classes***.

Toutefois, bien que disposant de capacités d'auto-adaptation de son architecture, le réseau AUDyC n'est pas à ce stade adapté pour la ***modélisation dynamique de données évolutives***. En effet, les règles d'apprentissage non supervisé utilisées permettent uniquement de modéliser un ensemble de données, puis d'adapter périodiquement la modélisation obtenue par présentation de nouveaux ensembles d'observations selon un mode d'apprentissage séquentiel. En fait, l'utilisation d'un processus d'apprentissage cyclique exclut toute définition d'un ***mode d'apprentissage continu***. Pour combler cette lacune, la troisième partie de ce chapitre présente un mode ***d'apprentissage continu du réseau AUDyC*** visant à autoriser la prise en compte en ligne de données évolutives. ***Une évolution du principe de définition et d'adaptation des prototypes*** permet, par ailleurs, d'introduire des ***capacités de modélisation dynamique*** pour le réseau AUDyC tout en intégrant une ***notion d'oubli des observations les plus anciennes.***

3. Architecture auto-adaptative et mode d'apprentissage continu

Idéalement, l'élaboration d'une technique de ***classification dynamique de données évolutives*** nécessite l'utilisation d'un ***processus d'apprentissage en ligne*** permettant de considérer l'***évolution temporelle*** des données dès leur apparition, et ainsi de réaliser une ***modélisation dynamique des classes***. Au cours de la partie précédente, nous avons essentiellement mis l'accent sur la description des capacités d'auto-adaptation de l'architecture et des différentes étapes du processus d'apprentissage cyclique du réseau AUDyC, ce qui a permis de présenter un ***mode d'apprentissage séquentiel***. Cette seconde phase de présentation est, quant à elle, axée sur la présentation d'***un mode d'apprentissage continu***, visant à une meilleure considération de l'évolution temporelle des données évolutives au travers de « ***classes dynamiques*** ». Pour cela, les règles d'apprentissage doivent permettre d'***acquérir les connaissances nouvelles*** mais

également d'***oublier les connaissances anciennes,*** ce qui est rendu possible via l'utilisation d'un ***processus d'apprentissage en ligne***. Les règles développées précédemment proposent, d'ores et déjà, d'adapter et/ou de modifier l'architecture neuronale au cours de l'apprentissage. En effet, en fonction de la pertinence des observations présentées, soit de nouveaux prototypes et/ou de nouvelles classes sont créés, soit les prototypes et/ou les classes existants sont adaptés. Toutefois, aucune règle intègre la possibilité d'oublier les observations les plus anciennes. Pour y remédier, cette seconde phase de développement propose également une ***modification du processus d'adaptation des prototypes***.

Cette dernière partie du chapitre est donc composée de trois paragraphes. Le premier expose le ***mode d'apprentissage continu du réseau AUDyC*** dédié au traitement des ***données évolutives***. Le second décrit la ***définition dynamique conférée aux prototypes et aux classes***, ainsi que la ***modification des relations d'adaptation*** des paramètres des prototypes, ceci afin de favoriser la modélisation dynamique des classes. Dans ce sens, un dernier paragraphe ***illustre les capacités de modélisation dynamique*** du réseau au travers de différentes expérimentations sur des ensembles de données évolutives.

3.1. Mode d'apprentissage continu

Une totale considération du problème de classification de données évolutives requiert un ***traitement en ligne*** des données via l'utilisation d'un ***mode d'apprentissage continu***. Or, sous sa forme actuelle, le processus d'apprentissage cyclique précédent (figure IV.6) ne répond pas à cette attente. En effet, sa structure bouclée autorise uniquement une remise en cause périodique des modèles de classes au travers l'utilisation d'un mode d'apprentissage séquentiel.

Le ***mode d'apprentissage continu,*** proposé ci-après, associe ***les capacités d'auto-adaptation du réseau AUDyC au caractère évolutif des observations***, en présentant ces dernières à l'aide de ***fenêtres glissantes***. Dans les paragraphes précédents, l'utilisation du processus d'apprentissage cyclique visait à accroître le taux de classification et à optimiser la modélisation, après chaque cycle, pour un même ensemble de données. Afin d'établir un mode d'apprentissage continu, il est indispensable de disposer d'un ***processus d'apprentissage en ligne*** permettant de traiter toutes les données évolutives au travers de fenêtres glissantes sur ces dernières. Cette présentation permanente de « nouveaux ensembles » de données permet une optimisation continue de la modélisation des classes.

Ainsi, le processus d'apprentissage en ligne proposé conserve ***la structure en 3 phases d'un cycle d'apprentissage*** (§2.2), ***mais sans le bouclage avec les observations non classées***. A partir de l'observation courante X^t, la définition d'une ***fenêtre de largeur*** N_{fen} ***sur les données*** « passées » permet de constituer l'ensemble d'observations présenté au réseau à l'instant t. Celui-ci est noté $X_t^{N_{fen}} = \left\{X^{t-N_{fen}+1}, \ldots, X^t\right\}$. Le choix de la largeur N_{fen} détermine l'horizon de données considérées lors de l'apprentissage. L'ensemble suivant est défini en décalant la fenêtre actuelle d'un nombre donné d'observations, noté Off .

Par opposition au processus d'apprentissage cyclique décrit figure IV.6, après la présentation de l'ensemble des données d'une fenêtre $X_t^{N_{fen}}$, les observations rejetées au sein de X_{amb} et

X_{app} ne sont plus reconsidérées. Toutefois, la phase de fusion continue à regrouper les classes proches, tout comme la phase d'évaluation continue à éliminer les prototypes et classes peu représentatifs, créés lors de la présentation des données de $X_t^{N_{fen}}$. Sur ce principe, à chaque instant t, la création d'une nouvelle classe ne peut être effective que si la fenêtre $X_t^{N_{fen}}$ comporte suffisamment d'observations caractéristiques de la nouvelle classe en raison des critères de cardinalité utilisés. Les seuils utilisés pour la phase d'évaluation doivent donc être définis en accord avec la dimension N_{fen} de la fenêtre d'observations considérée.

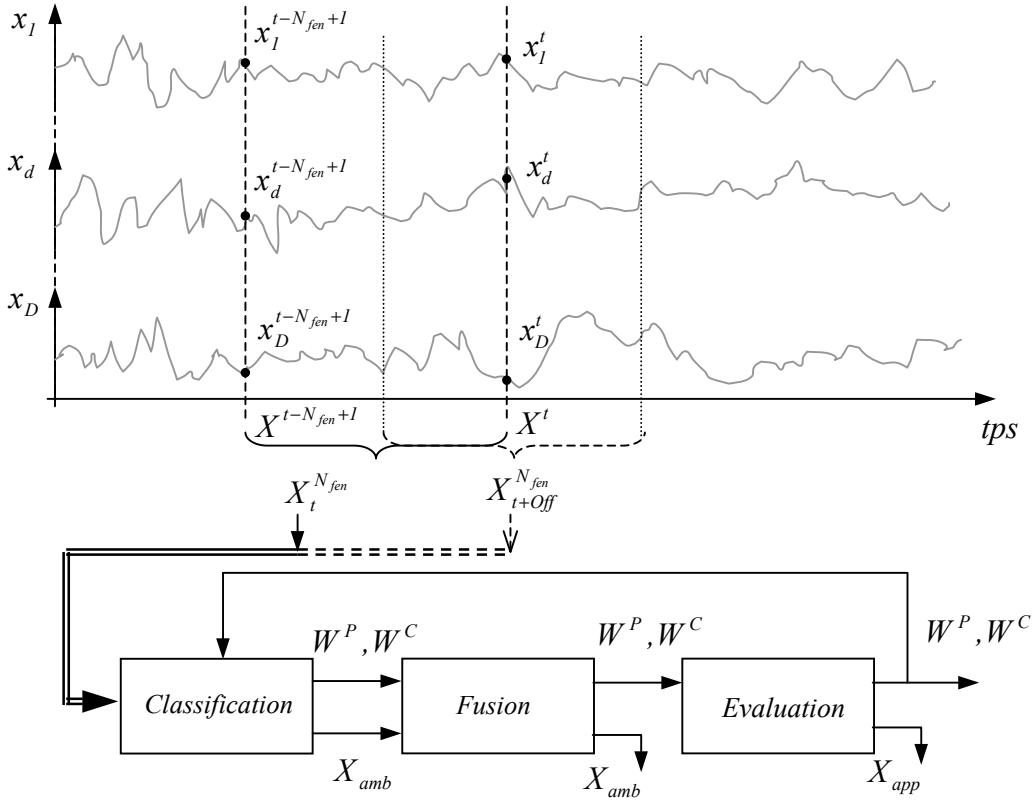


Figure IV.21 : Processus d'apprentissage en ligne du réseau AUDyC utilisé en présence de données évolutives.

La figure IV.21 illustre le ***processus d'apprentissage en ligne*** du réseau AUDyC. Toutefois, il existe des situations pour lesquelles un ensemble de données initiales est disponible. Dans ce cas, il est parfois préférable d'établir une modélisation précise de celui-ci avant d'entamer l'apprentissage en ligne. Pour cela, une étape d'initialisation a été introduite au sein de la procédure d'apprentissage (figure IV.22).

En fait, lors de l'étape d'initialisation, le processus d'apprentissage est le même puisqu'il suffit de considérer que la fenêtre de données présentée au réseau contient les mêmes observations à chaque présentation. Sur ce principe, la figure IV.22 expose l'intégralité du mode d'apprentissage continu du réseau AUDyC avec son ***étape d'initialisation*** et son ***étape d'apprentissage en ligne***.

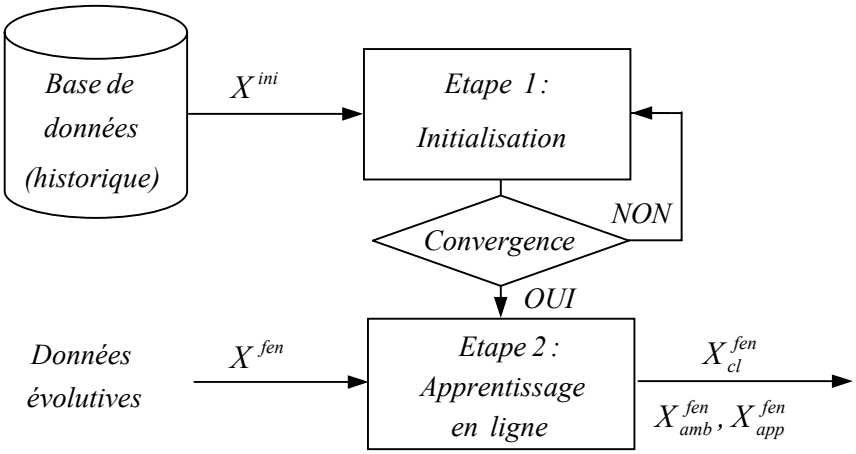


Figure IV.22 : Définition du mode d'apprentissage continu du réseau AUDyC en présence de données évolutives.

L'***étape d'initialisation*** réalise un apprentissage initial du réseau AUDyC afin de modéliser la connaissance relative aux observations contenues au sein d'un historique. Cette étape est réalisée en présentant plusieurs fois l'ensemble de données caractérisant la base de données initiales. Après convergence de cette étape d'initialisation, l'***étape d'apprentissage en ligne*** permet d'actualiser en continu les modèles des classes. Toutefois, cette étape d'initialisation n'est pas obligatoire, en raison des capacités d'auto-adaptation de notre architecture, et a uniquement pour vocation d'intégrer un ensemble de connaissances disponibles *a priori*.

3.2. Définition dynamique des prototypes et des classes

Le processus d'apprentissage décrit figure IV.21 permet au réseau AUDyC de traiter en ligne des problèmes de classification de données évolutives. Toutefois, en raison de la définition actuelle des prototypes et des classes, ***une modélisation dynamique des classes*** n'est pas possible. En effet, les relations d'adaptation (IV.13) et (IV.14) (§2.2.2.3) permettent d'adapter les paramètres des prototypes (centre et matrice de covariance) avec l'information apportée par les nouvelles observations, mais ne considèrent pas l'oubli des informations les plus anciennes. Dans ce sens, à chaque instant t, les modèles des prototypes et des classes intègrent l'ensemble des connaissances qui leur ont été associées jusqu'à l'instant considéré. Par conséquent, si les caractéristiques d'une classe ont évolué depuis son apprentissage initial, alors le modèle de celle-ci est forcément perturbé par la considération des observations les plus anciennes. De plus, au-delà de cette perturbation éventuelle des modèles de classes, les règles d'adaptation utilisées jusqu'ici engendrent une stabilisation des modèles des prototypes dès lors que ceux-ci comportent un nombre important d'observations.

Ainsi, nous proposons un ***nouveau principe d'adaptation des prototypes*** afin que ces derniers oublient les observations anciennes au fur et à mesure que leur définition intègre de nouvelles observations. De même, le principe d'association des prototypes aux classes est complété, afin de donner à chaque prototype, un degré de représentativité au sein de la classe à laquelle il est associé. Ceci a pour effet d'***améliorer la modélisation dynamique des classes***, notamment lors de la création de nouveaux prototypes pour cette dernière.

3.2.1. Nouvelles règles d'adaptation des prototypes

L'introduction d'une ***capacité d'oubli des observations anciennes*** lors de la définition des prototypes a pour but de rendre totalement dynamique la modélisation des prototypes et des classes vis-à-vis de l'évolution temporelle des données. La ***nouvelle définition des prototypes et de leur principe d'adaptation***, que nous proposons, vise à paramétrer leurs modèles uniquement sur la base des dernières observations qui leur ont été associées, i.e. à partir des observations les plus récentes. Evidemment, plusieurs solutions sont envisageables selon le poids que l'on souhaite accorder aux nouvelles observations associées à un prototype.

Une ***première solution*** consiste à ***introduire un facteur d'oubli*** α à valeurs dans l'intervalle $]0,1[$. L'adaptation des paramètres d'un prototype P^j est alors réalisée en pondérant du facteur α l'information relative à la nouvelle observation X^i associée au prototype P^j, et d'un facteur $1-\alpha$, les paramètres initiaux M_{P^j} et $\sum_{P^j}$ de ce dernier. La nouvelle moyenne $M^*_{P^j}$ et la nouvelle matrice de covariance $\sum^*_{P^j}$ du prototype sont alors déterminées de la façon suivante :

$$M^*_{P^j} = (1-\alpha).M_{P^j} + \alpha.X^i \tag{IV.24}$$

$$\sum\nolimits^*_{P^j} = (1-\alpha).\sum\nolimits_{P^j} + \alpha.\left(X^i - M^*_{P^j}\right).\left(X^i - M^*_{P^j}\right)^T \tag{IV.25}$$

Ces règles d'adaptation considèrent que chaque observation associée au prototype P^j est affectée d'une pondération donnée. Ainsi, l'observation la plus récente X^i est affectée du poids α et la pondération des autres observations diminue de façon exponentielle avec l'ancienneté des observations. Ainsi, plus le facteur α est grand, plus l'oubli des observations est rapide, et inversement, plus le facteur α est petit, plus l'oubli des observations est lent.

Une ***seconde solution*** consiste à introduire une fenêtre de largeur N_P pour la définition du prototype P^j. Celui-ci est alors paramétré sur la base des observations qui lui sont associées mais qui appartiennent encore à sa fenêtre de définition. Lors de l'adaptation du prototype P^j, on considère alors, d'une part, l'opération d'ajout de la nouvelle observation X^i, et d'autre part, celle de retrait de l'observation la plus ancienne notée X^{i-N_P+1}. Par opposition aux règles (IV.24) et (IV.25), les observations présentes sur la fenêtre N_P sont toutes pondérées de la même façon lors de la détermination des paramètres du prototype. La nouvelle moyenne $M^*_{P^j}$ et la nouvelle matrice de covariance $\sum^*_{P^j}$ du prototype sont alors déterminées par (voir annexe 3) :

$$M^*_{P^j} = M_{P^j} + \frac{1}{N_P}.\left(\delta X^+ - \delta X^-\right) \tag{IV.26}$$

$$\sum\nolimits^*_{P^j} = \sum\nolimits_{P^j} + \Delta X.Q^{-1}.\Delta X^T \tag{IV.27}$$

$$\text{où } \begin{cases} \delta X^+ = X^i - M_{P^j} \\ \delta X^+ = X^{i-N_P+1} - M_{P^j} \\ \Delta X = \left[\delta X^+ \quad \delta X^-\right] \end{cases} \text{ et } Q^{-1} = \frac{1}{N_P}\begin{bmatrix} 1 & \frac{1}{N_P-1} \\ \frac{1}{N_P-1} & -\frac{N_P+1}{N_P-1} \end{bmatrix}$$

Dans la pratique, ***nous avons retenu ce deuxième principe d'adaptation*** qui délimite parfaitement l'horizon de définition des prototypes. Par ailleurs, afin de faciliter le calcul en ligne des degrés d'appartenance aux prototypes (expression IV.2), un calcul itératif de la matrice de covariance inverse $\sum_{P^j}^{-1}$ de chaque prototype P^j est proposé (voir annexe 3) :

$$\sum\nolimits_{P^j}^{*\,-1} = \sum\nolimits_{P^j}^{-1} - K.\Delta X^T.\sum\nolimits_{P^j}^{-1} \qquad \text{(IV.28)}$$

avec $K = \sum_{P^j}^{-1}.\Delta X.\left(\Delta X^T.\sum_{P^j}^{-1}.\Delta X + Q\right)^{-1}$

Ces nouvelles règles d'adaptation ne s'appliquent pas dès la création des prototypes, mais uniquement lorsque ceux-ci ont atteint un effectif de N_P observations. Dans ce sens, on distingue ***trois phases d'apprentissage*** des prototypes (figure IV.23).

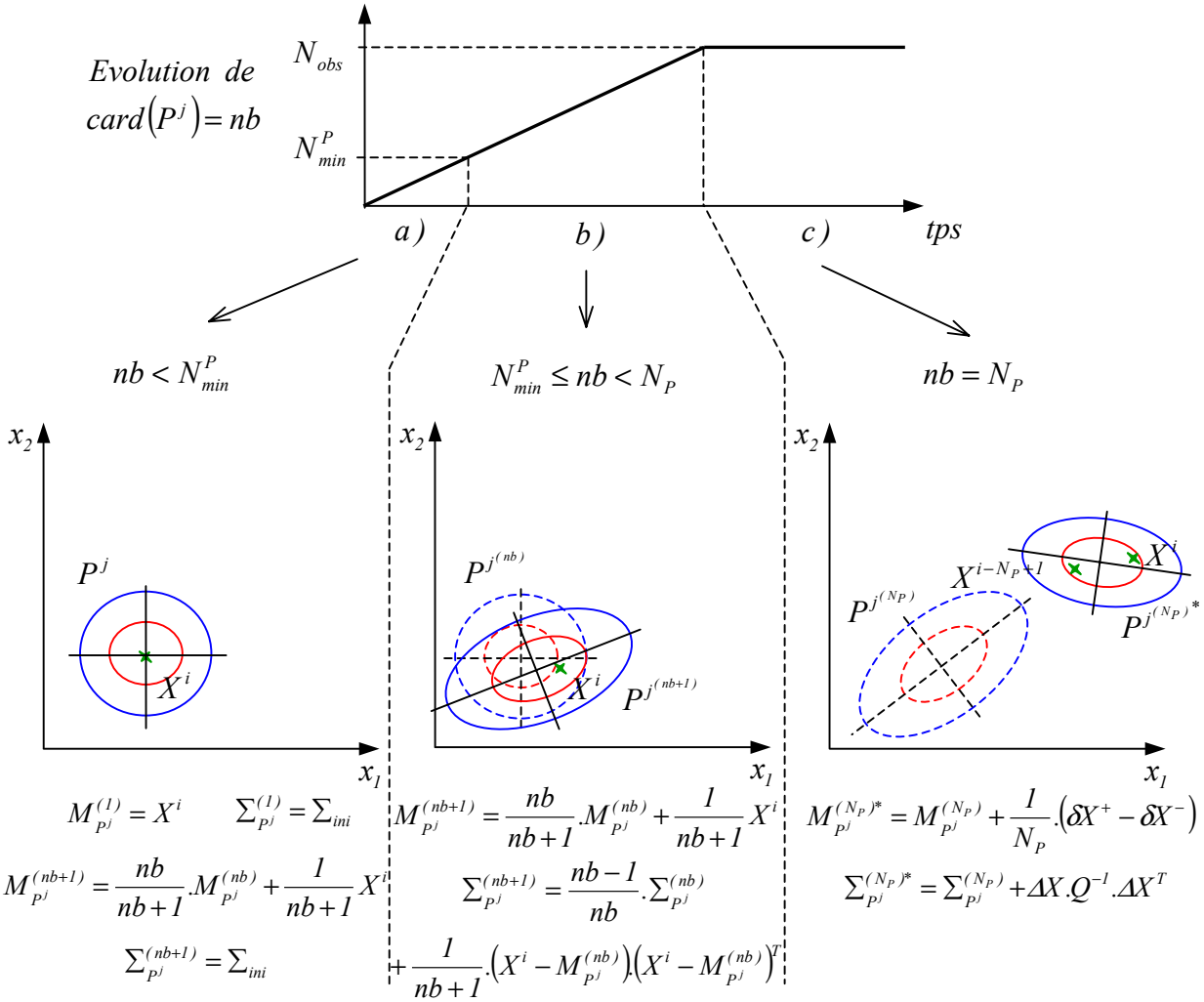


Figure IV.23 : Les 3 phases d'apprentissage d'un prototype : a) Phase de définition initiale, b) Phase d'adaptation avec ajout, c) Phase d'adaptation avec ajout et oubli.

La première phase (a) dure de la création du prototype jusqu'au moment où la cardinalité de celui-ci atteint le seuil N_{min}^P. Durant toute cette période, l'effectif du prototype augmente, et on adapte uniquement sa position (IV.13). La deuxième phase (b) poursuit l'apprentissage du prototype jusqu'à un effectif de N_P observations. Les paramètres du prototype sont adaptés à

l'aide des relations (IV.13) et (IV.14). Enfin, la troisième phase d'apprentissage (c) permet la modélisation dynamique du prototype sur la base des N_P dernières observations qui lui sont associées. Les paramètres du prototype sont adaptés à l'aide des relations (IV.26) et (IV.27).

3.2.2. Nouvelles relations prototypes-classes

D'après la définition initiale de l'architecture du réseau AUDyC (figure IV.2), la fonction d'appartenance ψ_k associée à la classe C^k est déterminée à partir des fonctions d'appartenance μ_j des différents prototypes $P^j_{C^k}$ de cette même classe. Toutefois, chaque prototype de la classe est supposé avoir le même ***degré de représentativité***, aucune distinction n'étant réalisée entre les prototypes d'une même classe. En effet, la matrice de pondération W^C, traduisant les relations entre la couche cachée (prototypes) et la couche de sortie (classes), réalise uniquement une association binaire des prototypes aux classes.

Afin d'améliorer ces relations prototypes-classes, ***la définition de la matrice de pondération*** W^C est modifiée afin d'associer à chaque prototype, un degré de représentativité par rapport à la classe qu'il caractérise. Pour cela, nous avons profité de l'introduction de l'horizon de définition N_P des prototypes, en considérant qu'un prototype est d'autant plus caractéristique d'une classe donnée que son effectif est proche de N_P. La matrice W^C est alors déterminée telle que ***le coefficient*** w^C_{kj} ***traduise le degré de représentativité du prototype*** P^j ***au sein de la classe*** C^k. Ainsi, celui-ci s'exprime de la manière suivante :

$$w^C_{kj} = \frac{Card(P^j_{C^k})}{N_P} \qquad \text{(IV.29)}$$

Par conséquent, la matrice W^C est modifiée après chaque adaptation d'un prototype. De plus, le degré d'appartenance $\psi_k(X^i)$ de l'observation X^i à la classe C^k, expression (IV.3), voit sa définition évoluée :

$$\psi_k(X^i) = min\left(1, \sum_{P^j \in C^k} w^C_{kj}.\mu_j(X^i)\right) \qquad \text{(IV.30)}$$

Cette nouvelle définition de la fonction d'appartenance ψ_k permet d'***atténuer l'influence des prototypes peu représentatifs*** lors de la modélisation des classes. Evidemment, dès que tous les prototypes d'une classe ont atteint un horizon de N_P observations, la relation (IV.29) devient obsolète et la relation (IV.30) redevient similaire à l'expression (IV.3). En fait, l'approche proposée vise essentiellement à limiter l'influence des prototypes n'ayant pas atteint leur horizon de définition, i.e. n'ayant pas réalisé la totalité de leur processus d'adaptation.

3.3. Modélisation dynamique de données évolutives

Les paragraphes précédents ont permis d'exposer un mode d'apprentissage continu du réseau AUDyC. Celui-ci comporte une ***étape d'initialisation*** et une ***étape d'apprentissage en ligne*** (figure IV.22). De plus, le réseau dispose désormais d'une capacité d'oubli des observations

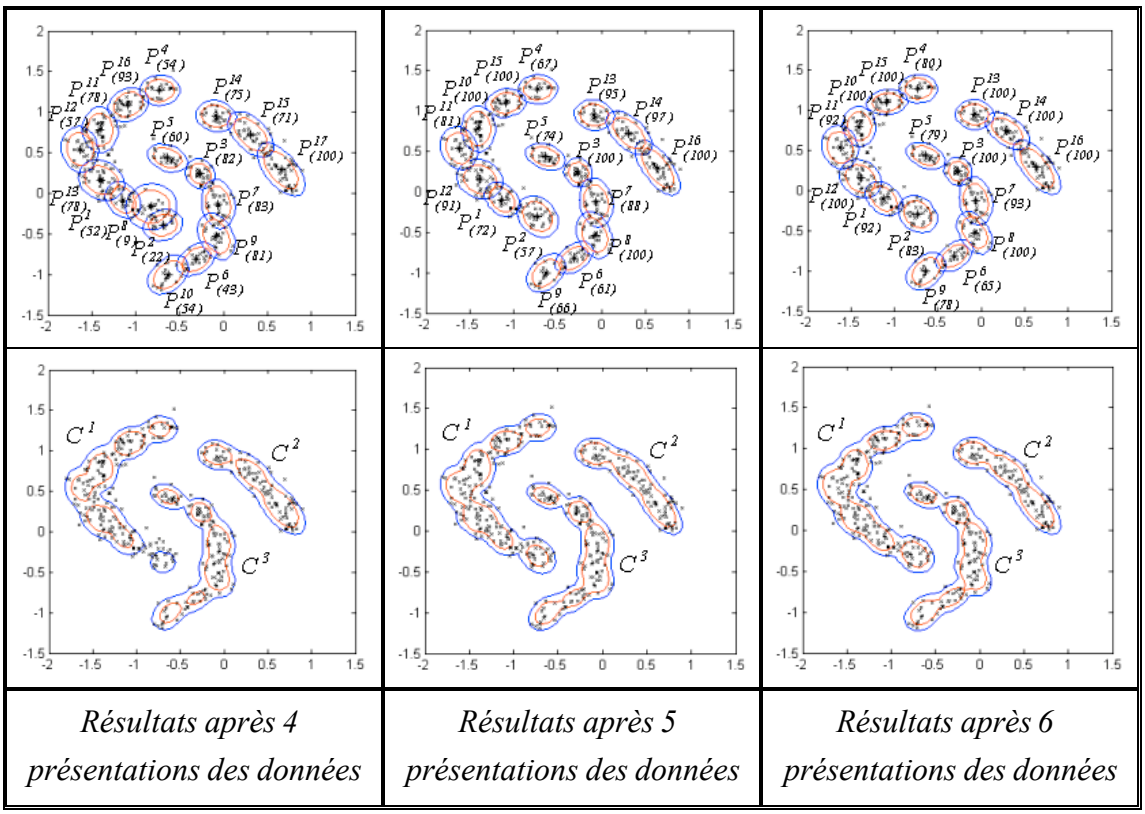
anciennes grâce à une ***nouvelle définition des prototypes.*** Enfin, une nouvelle relation prototypes-classes permet d'améliorer la qualité de modélisation des classes. L'ensemble de ces modifications confère au réseau une capacité d'apprentissage continu de données évolutives tout en améliorant ses qualités de modélisation.

Cette dernière partie du chapitre a donc pour objectif de valider les capacités de ***modélisation dynamique de l'architecture neuronale*** en présence de données évolutives. Avant cela, un premier paragraphe illustre le fonctionnement de l'***étape d'initialisation*** en considérant les données de la figure IV.17. Un deuxième paragraphe présente la ***capacité d'apprentissage en ligne*** du réseau AUDyC en décrivant, dans un premier temps, l'influence des grandeurs N_P et N_{fen}, puis ensuite, une situation où les ***classes se déplacent, se déforment, fusionnent ou encore de nouvelles classes apparaissent***.

3.3.1. Etape n°1 : Initialisation

L'étape d'initialisation a été définie afin de modéliser l'ensemble des connaissances disponibles *a priori*. La réalisation de cette étape d'initialisation dépend donc de l'existence d'un historique sur les données et n'est pas obligatoire en raison des capacités d'auto-adaptation en ligne du réseau AUDyC. Toutefois, ce paragraphe en propose une illustration en considérant les données de la figure IV.17.

Comme indiqué au §3.1, l'ensemble de données (411 observations) est présenté plusieurs fois au réseau, ce qui revient à considérer successivement la présentation de la même fenêtre de données de largeur $N_{fen} = 411$. Les paramètres et seuils initiaux conservent les valeurs établies précédemment et l'horizon de définition des prototypes est fixé à $N_P = 100$. Les résultats obtenus sont décrits au niveau du tableau IV.3 qui illustre les prototypes et les classes après 4, 5 et 6 présentations de l'ensemble de données. Au niveau des modèles de prototypes (1ère ligne), la notation $P^j_{(nb)}$ précise la cardinalité du prototype P^j à différents stades de l'étape d'initialisation. Cette illustration permet également de constater l'effet de l'introduction du degré de représentativité pour les prototypes. En effet, suite à la 4ème boucle de l'étape d'initialisation (1ère colonne), on peut voir que la classe C^2 n'est pas perturbée par la création du prototype P^8 qui a pourtant à peine débuté son processus d'adaptation. En fait, la création de ce prototype a été nécessaire, dans le sens où il a permis de modéliser des données non considérées précédemment, bien que d'après les itérations suivantes de l'étape d'initialisation, on remarque que son existence est temporaire puisqu'il fusionne avec le prototype P^2.



Résultats après 4 présentations des données	*Résultats après 5 présentations des données*	*Résultats après 6 présentations des données*

Tableau IV.3 : Convergence de l'initialisation du réseau AUDyC sur la base des données de la figure IV.17 après plusieurs présentations.

L'étape d'initialisation s'achève après stabilisation de la modélisation des classes (figure IV.22). A l'issue de celle-ci, les prototypes ont généralement complètement atteint leur horizon de définition de N_P observations. L'application de l'étape d'apprentissage en ligne permet alors d'actualiser les connaissances acquises.

3.3.2. Etape n°2 : Apprentissage en ligne

L'étape d'apprentissage en ligne traduit la capacité de ***modélisation dynamique de données évolutives*** pour le réseau AUDyC. Les aptitudes réelles de cette étape d'apprentissage en ligne du réseau sont évaluées au cours de trois expérimentations.

La première précise l'influence du choix de l'horizon N_P utilisé pour la définition des prototypes. La deuxième vise à apprécier l'importance du choix de la largeur N_{fen} des fenêtres de présentation des données. Enfin, la troisième décrit un exemple d'application du réseau AUDyC à une situation de modélisation dynamique d'un ensemble de données évolutives.

Pour ces différentes expérimentations, nous avons conservé les seuils précédents en modifiant toutefois la valeur de σ_{ini} par rapport aux données utilisées, soit $\sigma_{ini} = 0{,}7$. La fenêtre de données X^t_{fen} présentée au réseau est constituée, à chaque instant t, des observations allant de $X^{t-N_{fen}+1}$ jusqu'à l'observation courante X^t, soit un ensemble de N_{fen} observations appartenant indifféremment à l'une des classes considérées. Enfin, entre chaque fenêtre présentée au réseau, nous avons considéré un décalage de $Off = 4$ observations.

3.3.2.1. Influence de l'horizon N_P des prototypes

D'après le §3.2.1, la grandeur N_P a été introduite afin d'associer à chaque prototype un horizon de définition. En fait, N_P délimite l'horizon des prototypes et détermine la dynamique d'évolution de ces derniers et donc des classes. Pour un prototype donné, plus la valeur de N_P est faible plus celui-ci est sensible à l'évolution de la distribution de données qu'il caractérise et inversement, plus la valeur de N_P est grande, plus le prototype y est insensible. Prenons l'exemple du tableau IV.4 où nous avons appliqué le réseau AUDyC en considérant d'une part une fenêtre glissante de largeur $N_{fen} = 100$ sur les 1000 observations définissant la phase d'évolution de la classe C^1 et d'autre part 3 valeurs distinctes pour N_P.

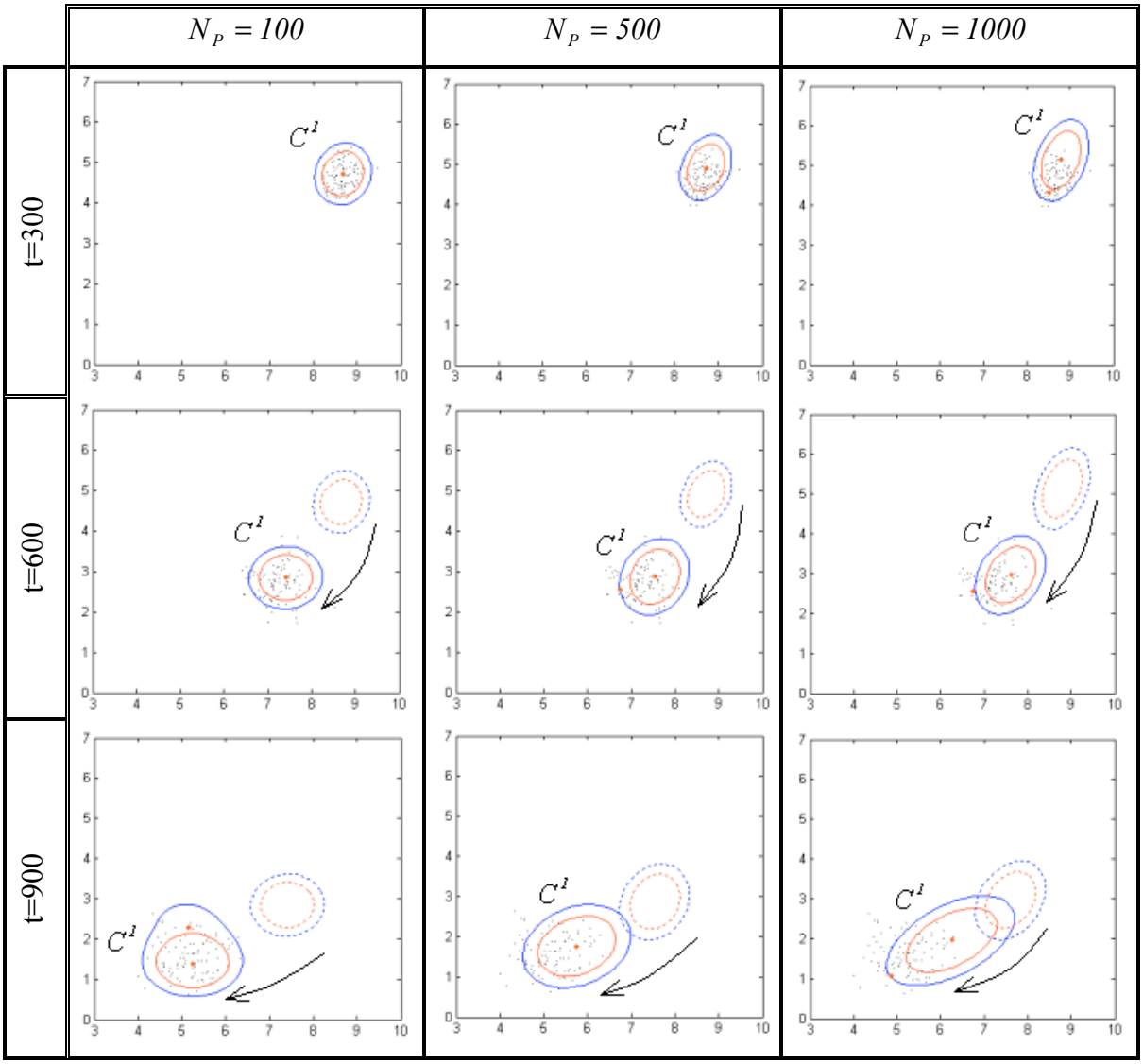


Tableau IV.4 : Illustration de l'influence de l'horizon N_P pour la définition des prototypes sur leur dynamique d'adaptation.

Pour $N_P = 100$ (1ère colonne), la dynamique d'adaptation des prototypes est suffisante pour caractériser l'évolution de la classe C^1. En effet, l'adaptation des prototypes est suffisamment « rapide » pour suivre le déplacement de la classe. Par opposition, pour les valeurs de N_P de 500

ou de 1000 (2ème et 3ème colonne), on remarque que le modèle obtenu pour la classe C^1 est constamment en « retard » par rapport à la dernière fenêtre de données (représentées par des points). De plus, pour ces deux dernières valeurs, on peut remarquer qu'à plusieurs reprises un prototype supplémentaire est créé au niveau des observations les plus récentes afin d'améliorer la caractérisation de la classe. Toutefois, celui-ci n'influe guère sur le modèle global de la classe en raison de son effectif restreint vis-à-vis de la valeur N_P. Dans le cas extrême, i.e. pour une valeur de N_P très grande vis-à-vis de la vitesse d'évolution de la classe, il est possible qu'un prototype ne puisse « suivre » cette évolution ce qui entraîne la création d'un nouveau prototype. Par conséquent et de façon idéale, l'horizon N_P de définition des prototypes doit être choisi en fonction de la dynamique d'évolution des observations modélisées.

3.3.2.2. Influence de la largeur N_{fen} des fenêtres de données

Après avoir étudié l'influence de N_P sur la dynamique de modélisation des prototypes, ce paragraphe évalue l'influence de la largeur de fenêtre N_{fen}. En fait, le choix de N_{fen}, traduisant un horizon sur les données présentées au réseau, n'a de réelle influence que si les caractéristiques des classes évoluent. Ainsi, plus N_{fen} est grand, plus le modèle d'une classe couvrira une large définition de l'évolution de cette dernière. Par ailleurs, le choix des seuils d'évaluation des prototypes et des classes doit être établi en accord avec la grandeur N_{fen}.

Le tableau IV.5 illustre l'évolution de 2 modèles de classes au fur et à mesure de la présentation des données représentatives de leur évolution. Pour évaluer concrètement l'influence de N_{fen}, l'algorithme a été appliqué en considérant dans un premier temps $N_{fen} = 200$, puis $N_{fen} = 500$, en conservant la même valeur pour l'horizon de définition des prototypes ($N_P = 100$). La colonne de gauche résume l'évolution du nombre de prototypes au sein de chacune des classes, alors que la colonne de droite décrit l'évolution des modèles de classes pour les deux valeurs de N_{fen} retenues.

Dans les deux cas, les modèles de classes représentent correctement les données présentées au réseau ce qui justifie la dynamique d'adaptation accordée aux prototypes avec le choix de $N_P = 100$. Par contre, à un même instant t, les modèles obtenus pour $N_{fen} = 500$ caractérisent une plus grande période de l'évolution des classes que les modèles obtenus pour $N_{fen} = 200$. Enfin, les modèles de classes obtenus avec $N_{fen} = 500$ nécessitent la définition de plusieurs prototypes afin de couvrir l'ensemble de la phase d'évolution des classes caractérisées par les données présentes sur la fenêtre.

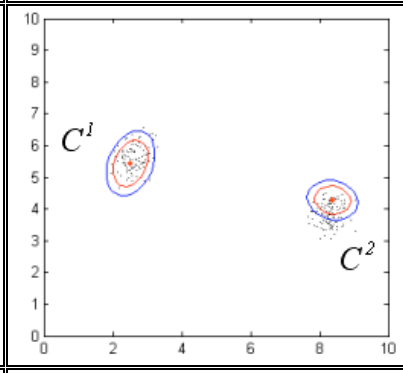
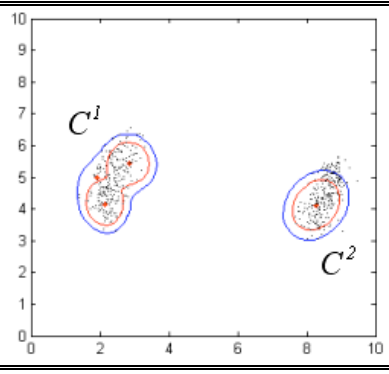
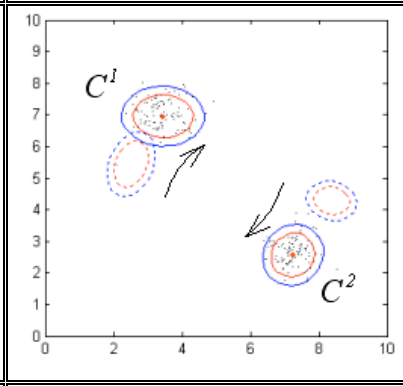
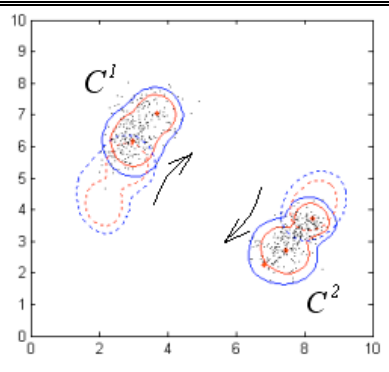
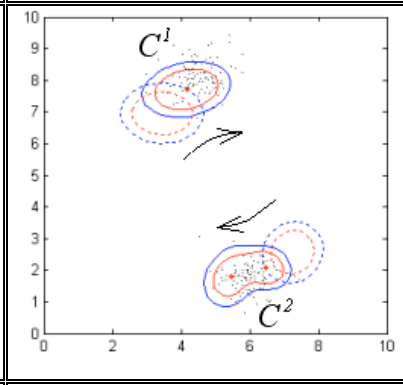
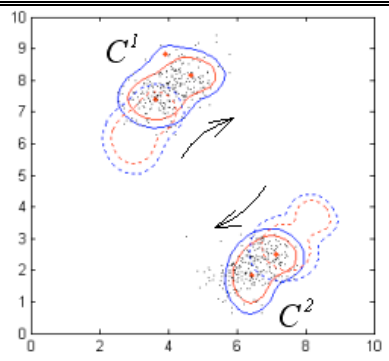
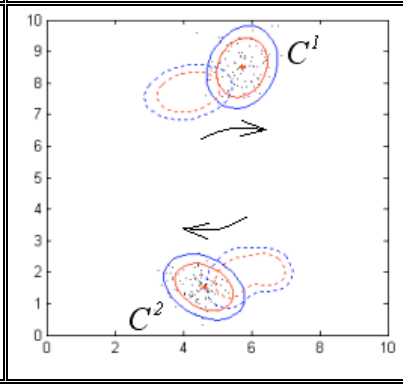
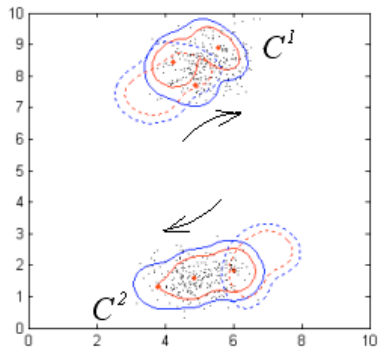
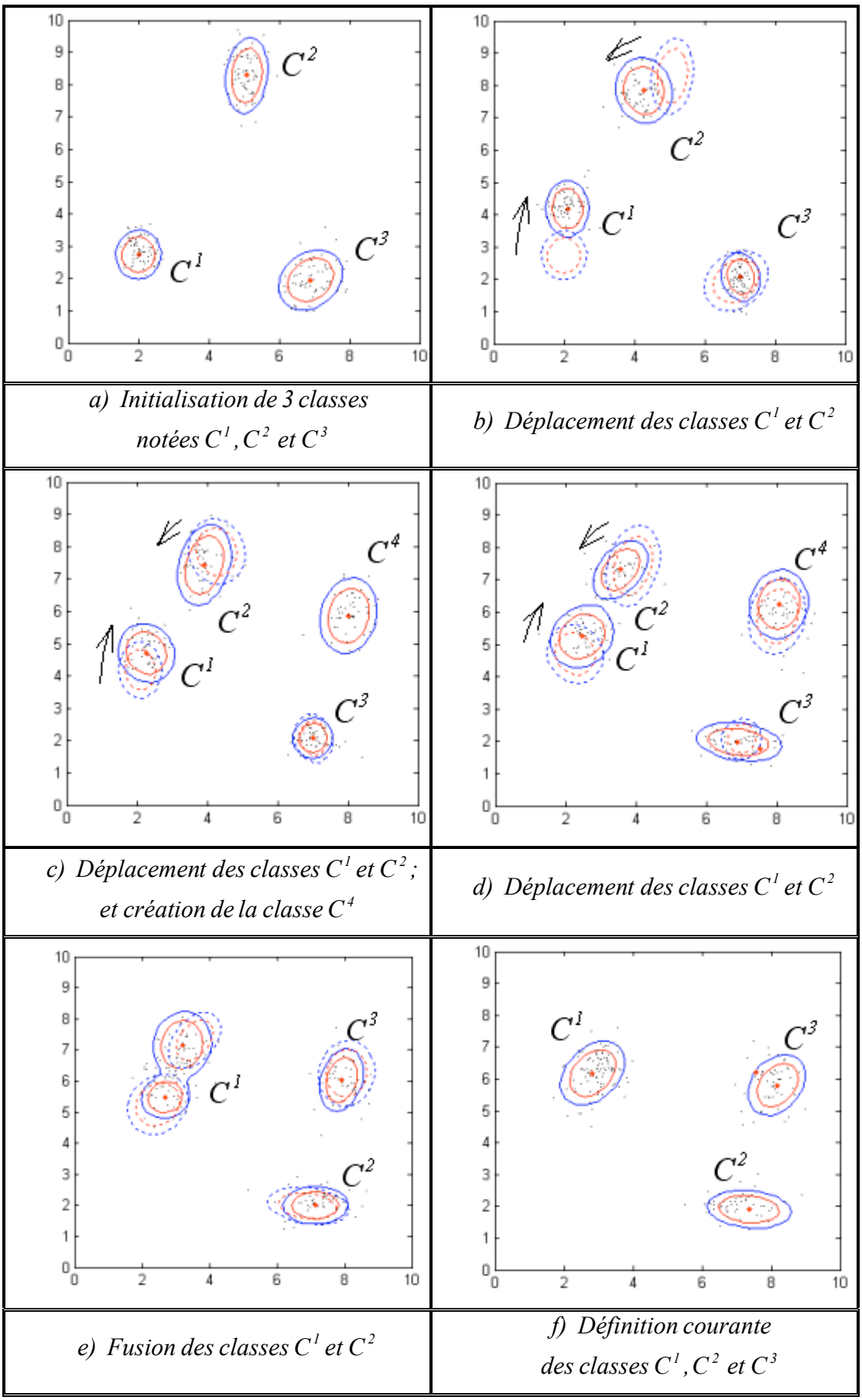
	$N_{fen} = 200$	$N_{fen} = 500$
$t = 800$ $N_{fen} = 200$ { C^1 1 prototype ; C^2 1 prototype } $N_{fen} = 500$ { C^1 3 prototypes ; C^2 1 prototype }		
$t = 1200$ $N_{fen} = 200$ { C^1 1 prototype ; C^2 1 prototype } $N_{fen} = 500$ { C^1 2 prototypes ; C^2 3 prototypes }		
$t = 1600$ $N_{fen} = 200$ { C^1 1 prototype ; C^2 2 prototypes } $N_{fen} = 500$ { C^1 3 prototypes ; C^2 2 prototypes }		
$t = 2000$ $N_{fen} = 200$ { C^1 1 prototype ; C^2 1 prototype } $N_{fen} = 500$ { C^1 3 prototypes ; C^2 3 prototypes }		

Tableau IV.5 : Illustration de l'influence de la largeur de fenêtre N_{fen} sur les classes de données.

3.3.2.3. Création et fusion de classes

Ce troisième et dernier paragraphe résume sur un seul exemple (tableau IV.6) l'ensemble des capacités d'apprentissage en ligne et de modélisation dynamique du réseau AUDyC.



a) Initialisation de 3 classes notées C^1, C^2 et C^3

b) Déplacement des classes C^1 et C^2

c) Déplacement des classes C^1 et C^2 ; et création de la classe C^4

d) Déplacement des classes C^1 et C^2

e) Fusion des classes C^1 et C^2

f) Définition courante des classes C^1, C^2 et C^3

Tableau IV.6 : Illustration des capacités de modélisation dynamique du réseau AUDyC dans des situations d'évolution, de création et de fusion de classes.

Suite aux remarques effectuées précédemment, nous avons réalisé cette expérience en prenant un choix judicieux de N_{fen} et de N_P permettant d'accorder une dynamique convenable aux modèles des prototypes et des classes. Les différentes figures reprises au sein du tableau IV.6 illustrent alors l'évolution des classes à différentes étapes de l'apprentissage en ligne. Trois classes sont initialement créées (a) puis les classes C^1 et C^2 évoluent l'une vers l'autre (b et c). A l'étape (c), une nouvelle classe est caractérisée, la classe C^4. Lors de ces premières étapes, on remarque également que le modèle de la classe C^2 évolue au gré de la modification des caractéristiques de la classe. A l'étape (e), on assiste à la fusion des classes C^1 et C^2 pour atteindre finalement la modélisation représentée à l'étape (f).

3.4. Conclusion

Cette troisième partie a permis de présenter une seconde version du réseau ***AUDyC*** à laquelle nous avons associé un ***mode d'apprentissage continu*** plus intéressant pour la classification et la modélisation de données évolutives. Cette modification du mode d'apprentissage s'est traduite par la définition d'un ***processus d'apprentissage en ligne*** et s'est accompagnée d'une ***nouvelle définition des prototypes et des classes*** afin d'améliorer la qualité de modélisation dynamique.

L'élaboration du mode d***'apprentissage continu*** du réseau s'est appuyée à la fois sur les capacités d'auto-adaptation de l'architecture développée précédemment et sur la présentation continue de données au réseau. Ceci a permis de définir une ***étape d'apprentissage en ligne*** où l'on présente des fenêtres de données successives au réseau. Cette étape d'apprentissage en ligne s'accompagne d'une ***étape d'initialisation*** permettant d'acquérir la connaissance relative à une base de données initiales.

La ***nouvelle définition des prototypes*** a permis d'introduire une notion d'oubli des observations anciennes lors de la détermination des paramètres des prototypes. Pour cela, l'horizon de définition de chaque prototype a été restreint aux observations les plus récentes et de ***nouvelles règles d'adaptation des prototypes*** ont été établies. Cette nouvelle définition des prototypes confère à ces derniers une certaine dynamique de modélisation. Par ailleurs, la modélisation des classes a été améliorée via l'introduction pour chaque prototype d'un ***degré de représentativité*** au sein de la classe qu'il caractérise.

Les aptitudes d'apprentissage en ligne et de modélisation dynamique ***du réseau AUDyC*** ont alors été illustrées au travers de différentes expérimentations. Dans un premier temps, celles-ci ont permis d'apprécier l'***influence de l'horizon de définition des prototypes*** ainsi que de la ***largeur de fenêtre utilisée sur les données évolutives***. Ensuite, une dernière expérimentation a permis de vérifier l'ensemble des capacités du réseau AUDyC pour la ***modélisation dynamique de données évolutives***, en illustrant sur un même exemple, la caractérisation de l'évolution des modèles existants, mais aussi la création de nouvelles classes ou encore la fusion de classes existantes.

4. Conclusion

Ce quatrième chapitre a permis de présenter ***une nouvelle architecture neuronale pour la classification dynamique de données évolutives*** permettant de définir un ***système adaptatif de Reconnaissance de Formes***. La technique neuronale développée est le ***réseau AUDyC*** (AUto-adaptive and Dynamical Clustering). L'élaboration du réseau a été réalisée en tenant compte d'un ensemble de contraintes liées à la modélisation dynamique de données évolutives avec notamment la définition d'une architecture auto-adaptative via l'utilisation de règles d'apprentissage non supervisé. L'adaptation des modèles de classes est alors réalisée au cours soit d'un mode d'***apprentissage séquentiel,*** soit d'un mode d'***apprentissage continu***.

Dans le premier cas, l'accent a été porté sur l'élaboration ***de l'architecture auto-adaptative*** du réseau. Ceci a nécessité l'introduction d'un ensemble de règles spécifiques d'apprentissage non supervisé autorisant l'adaptation des paramètres des prototypes et classes existants mais également la création de nouveaux prototypes et/ou de nouvelles classes. Celles-ci ont été complétées par des règles de fusion et d'évaluation des prototypes et/ou des classes. Le principe d'apprentissage de l'architecture neuronale s'articule alors autour d'un processus cyclique composé de 3 phases. Les capacités de classification et de modélisation offertes par cette architecture neuronale ont été validées sur un ensemble de données stationnaires caractérisant 3 classes de forme complexe. Une comparaison à d'autres algorithmes dynamiques de classification disposant d'une architecture auto-adaptative a permis de confirmer les performances de l'architecture développée.

Dans le second cas, l'objectif a été d'étendre ***les capacités d'auto-adaptation*** de l'architecture à la ***modélisation dynamique de données évolutives*** via la définition d'***un mode d'apprentissage continu*** autorisant le traitement en ligne de données évolutives. L'apprentissage est alors réalisé au cours d'un processus en ligne où l'on présente au réseau des fenêtres de données que l'on décale au fur et à mesure de l'apparition des observations. De plus, une ***nouvelle définition des prototypes*** autorise l'***oubli des observations anciennes*** ce qui permet d'appréhender complètement l'aspect évolutif des données. Enfin, cette nouvelle définition des prototypes a été complétée par l'introduction d'un ***degré de représentativité*** des prototypes aux classes ce qui a permis d'améliorer la modélisation de ces dernières. Les capacités d'apprentissage en ligne du réseau AUDyC ont alors été expérimentées pour la modélisation dynamique de données évolutives.

Ce quatrième chapitre a eu pour objet de présenter une nouvelle architecture neuronale, le réseau AUDyC, développée dans le cadre de nos travaux. Celle-ci dispose d'un ensemble d'aptitudes pour les problèmes de classification et de modélisation de données évolutives. A l'aide de son mode d'apprentissage continu, le réseau AUDyC permet d'élaborer un système adaptatif de diagnostic par RdF (voir chapitre 2) capable d'établir une ***modélisation dynamique des modes de fonctionnement*** d'un procédé. Dans ce sens, le chapitre suivant expose la définition d'un système de supervision utilisant notre technique neuronale et décrit son application à la surveillance et au diagnostic d'un système hydraulique industriel.

***Chapitre V* : Système adaptatif de supervision de procédés industriels. Application à une presse hydraulique**

Conclusion générale

Les travaux menés au cours de cette thèse ont permis de définir un système adaptatif de supervision pour les procédés industriels. Ils s'inscrivent dans la stratégie de la société ARC International, qui vise à ***accroître la maîtrise de l'état de fonctionnement de ses outils de production***, afin de mieux répondre aux exigences actuelles d'un marché très concurrentiel. Pour cela, l'objectif du système de supervision proposé au sein de ce mémoire, est de réaliser une ***surveillance continue de l'état de fonctionnement*** d'un procédé industriel, de ***détecter les évolutions brutales ou progressives*** de celui-ci et d***'élaborer un diagnostic***. Bien que plusieurs approches soient possibles, en raison de la complexité de modélisation mathématique des procédés de fabrication verrière, nous nous sommes orientés vers une ***approche externe*** exploitant des ***techniques de reconnaissance de formes***.

L'approche retenue se base sur la ***modélisation en ligne des différents modes de fonctionnement*** du procédé. Nous utilisons une ***représentation de l'état de fonctionnement*** obtenue à partir de caractéristiques extraites des signaux prélevés sur le procédé. La ***surveillance*** consiste alors à analyser l'évolution des observations par rapport aux différents modèles de modes de fonctionnement. Le ***diagnostic*** permet ensuite de caractériser l'état de fonctionnement en cours en référence aux différents modes appris. Evidemment, la réussite de cette approche repose sur la ***qualité de modélisation des modes de fonctionnement***. C'est la raison pour laquelle nos travaux se sont appuyés sur des techniques de classification permettant de modéliser en ligne et de manière non supervisée ces différents modes de fonctionnement.

La plupart des ***techniques actuelles de classification***, présentées dans la littérature, nécessitent généralement la connaissance *a priori* d'observations sur l'ensemble des modes de fonctionnement du procédé. Cette condition impose l'exhaustivité de la connaissance initiale, ce qui en pratique n'est pas envisageable pour une application de diagnostic. En effet, au cours de la vie d'un procédé industriel, on peut assister à l'***apparition de nouveaux modes de fonctionnement*** ou encore à l'***évolution des caractéristiques des modes de fonctionnement existants***. Sur ce principe, l'apprentissage des modèles de modes de fonctionnement requiert l'utilisation d'une technique de ***classification dynamique de données évolutives*** permettant une analyse en ligne des observations relevées sur le procédé.

Par conséquent, la technique de classification, que nous avons développée, s'appuie sur une nouvelle architecture neuronale, ***le réseau AUDyC*** qui dispose de ***capacités d'auto-adaptation*** rendant son architecture totalement évolutive. Par ailleurs, la classification est basée sur une ***approche multiprototype*** offrant la possibilité de modéliser des classes de forme complexe à partir de ***prototypes gaussiens***. Un ensemble de règles d'apprentissage non supervisé a alors été proposé, puis a permis d'élaborer un processus d'apprentissage décomposé en trois phases. Une première phase, dite ***phase de classification***, vise à évaluer la ressemblance des nouvelles observations aux classes et aux prototypes existants. Celle-ci donne lieu à la caractérisation de nouvelles classes par des règles de ***création de nouveaux neurones*** ou à l'adaptation des

modèles des classes existantes. Cette adaptation est réalisée soit par des règles d'***adaptation itérative des paramètres des prototypes***, soit par des règles de ***fusion de prototypes***. Une seconde phase, dite ***phase de fusion***, permet de fusionner les classes pour lesquelles un nombre donné d'ambiguïtés a été atteint. Enfin, une troisième phase, dite ***phase d'évaluation***, permet d'éliminer les classes et les prototypes peu représentatifs sur la base de critères de cardinalité. L'architecture neuronale est alors simplifiée par suppression des neurones correspondants. En s'appuyant sur ces différentes règles, ***deux modes d'apprentissage*** du réseau AUDyC ont été proposés, un ***mode d'apprentissage séquentiel*** et un ***mode d'apprentissage continu*** :

- le ***mode d'apprentissage séquentiel*** s'appuie sur une ***utilisation cyclique*** du processus d'apprentissage. Lors de la présentation d'un ensemble de données au réseau, celles-ci sont traitées successivement par chacune des phases, puis les observations non classées sont à nouveau présentées lors du cycle suivant. De plus, à chaque nouveau cycle, il est envisageable de présenter un ensemble de données complémentaires visant à affiner le modèle obtenu. Cette propriété caractérise le mode d'apprentissage séquentiel en autorisant une actualisation périodique des modèles des classes, ce qui permet de pallier la non exhaustivité de la base de données initiale. Différentes expérimentations ont permis de valider les qualités de modélisation et de classification du réseau AUDyC, en insistant sur ses capacités d'auto-adaptation et sur la convergence de la structure de classes obtenue.
- le ***mode d'apprentissage continu*** s'appuie sur une ***utilisation en ligne*** du processus d'apprentissage. L'objectif de ce second mode d'apprentissage est de réaliser une ***modélisation dynamique de données évolutives*** en proposant un traitement continu des données. Pour cela, les observations sont présentées au réseau à l'aide de ***fenêtres successives sur les données évolutives***. Les phases de classification, de fusion et d'évaluation sont organisées au sein d'un ***processus en ligne***, sans bouclage des cycles, et en y intégrant une ***définition dynamique des prototypes et des classes***. Enfin, le principe d'adaptation des prototypes a été complété par une ***capacité d'oubli des observations les plus anciennes***. Différentes expérimentations ont permis de valider les capacités d'apprentissage en ligne et de modélisation dynamique du réseau AUDyC au travers d'exemples où des classes apparaissent, évoluent et fusionnent.

En s'appuyant sur les propriétés du réseau AUDyC, un ***nouveau système adaptatif de supervision*** a été élaboré. Ses aptitudes pour la ***surveillance et le diagnostic*** de l'état de fonctionnement d'une installation ont été ***expérimentées sur une presse hydraulique*** d'ARC International. Ce système de supervision est composé de trois modules. Le premier module s'apparente au réseau AUDyC dans son mode d'apprentissage continu et réalise une ***modélisation dynamique des modes de fonctionnement*** du procédé étudié. Lors de la présentation d'une nouvelle fenêtre d'observations, le ***module de modélisation*** évalue l'appartenance des observations aux modèles de fonctionnement, puis actualise la connaissance sur les modes de fonctionnement. Un second module, dédié à la tâche de ***surveillance***, permet de ***détecter toutes les évolutions possibles de l'état de fonctionnement***. Cette surveillance est réalisée à l'aide d'un ***classifieur de tendances*** capable de distinguer différentes formes

d'évolutions au sein d'un signal. Les ***dérives rapides*** (brutales ou progressives) de l'état de fonctionnement d'un mode de fonctionnement à un autre sont détectées en suivant l'évolution des degrés d'appartenance aux différents modèles. Par ailleurs, en raison du principe d'adaptation continue des modèles de modes de fonctionnement, une surveillance des caractéristiques de ces derniers permet de détecter des ***dérives lentes*** de l'état de fonctionnement. Enfin, un ***module de diagnostic*** complète le système de supervision en ***informant l'utilisateur du mode de fonctionnement dans lequel se situe le procédé ou vers lequel il tend à se rapprocher***.

Un ensemble d'expérimentations sur un ***banc hydraulique*** a permis d'évaluer les capacités du système de supervision proposé. Au travers des différentes expérimentations réalisées, nous avons pu voir que notre approche permet la prise en compte des nouveaux modes de fonctionnement, mais également leur adaptation continue. Ainsi, et par exemple, nous avons pu mettre en évidence les périodes de mise en régime de l'installation, l'usure des composants, …. En expérimentant différentes défaillances provoquées sur l'installation hydraulique, les capacités du module de surveillance ont été éprouvées pour la détection de dérives lentes et de dérives rapides de l'état de fonctionnement. Enfin, il a été vu que le module de diagnostic apporte une aide à la décision pour l'utilisateur en lui indiquant le mode de fonctionnement actuel de l'installation ou celui vers lequel elle tend à se rapprocher.

Plusieurs améliorations de notre technique de classification de données évolutives sont envisageables. Tout d'abord, l'utilisation du réseau AUDyC requiert actuellement le choix d'un ensemble de seuils nécessaires à son fonctionnement optimal. Afin de ***limiter cette « contrainte »***, il serait intéressant de pouvoir éliminer une partie de ces paramètres de réglage en proposant des critères automatiques de choix. Par exemple, l'écart-type initial des prototypes pourrait être établi en fonction de la densité des observations déterminée à partir des données initiales. De même, l'élaboration de nouveaux critères pourrait permettre d'améliorer les phases de fusion et d'évaluation en cherchant à mieux quantifier la proximité des classes ou tout simplement leur représentativité au sein de la distribution de données. Par ailleurs, dans le cadre de la modélisation dynamique des modes de fonctionnement d'un procédé industriel, il serait peut-être intéressant de conférer une ***dynamique propre à chacun des modèles de fonctionnement***. En effet, la fréquence d'occurrence d'observations caractéristiques du mode de bon fonctionnement est plus importante (et heureusement) que celles des observations des modes de dysfonctionnement, et par conséquent, le modèle associé ne nécessite par forcément la même rapidité d'adaptation.

D'un autre coté, pour le ***système de supervision proposé***, il serait intéressant d'***élargir son champ d'application à d'autres procédés industriels*** afin de mieux apprécier ses aptitudes en présence de phénomènes de dérives plus ou moins rapides. En effet, d'après nos premières expérimentations sur le banc de pressage hydraulique, nous avons pu remarquer que le principe et la qualité de détection d'une dérive dépendent de la vitesse d'évolution des observations, mais aussi de la dynamique d'adaptation accordée aux modèles de modes de fonctionnement. Dans ce sens, une ***expérimentation sur un procédé industriel sujet à des phénomènes de dérives***, variées

en terme de type et de vitesse d'évolution, serait riche d'enseignements. Ainsi, une étude plus approfondie des phénomènes de dérives permettrait de parfaire notre approche de supervision et de donner une meilleure réactivité lors de la prise de décision.

Enfin, au-delà de l'application à la maintenance prédictive, il serait intéressant d'envisager une exploitation des techniques développées sur ***des données industrielles*** de nature différente. Dans ce sens, un premier travail pourrait consister à réaliser un suivi de la ***qualité des articles produits***. Pour cela, il suffirait de disposer de caractéristiques pertinentes des articles pour établir une modélisation des classes de qualité de ces derniers. La démarche consisterait alors à informer l'utilisateur dès lors qu'une dérive de la qualité de fabrication serait détectée. Enfin, cette étude pourrait être poursuivie par une analyse des ***corrélations entre les paramètres de réglages et des caractéristiques des modèles de classes de qualité, lors de l'apparition des différentes formes de rebuts***. L'objectif final de cette dernière étude serait alors de prévenir l'apparition éventuelle d'une non-qualité des articles à partir de l'analyse des évolutions des paramètres de réglage au niveau de la machine primaire.

Tous ces aspects ouvrent un champ d'investigation qui sera poursuivi dans le cadre des attributions qui m'ont été confiées lors de mon embauche au sein d'ARC International depuis le 1er février 2003. Fort de cette première expérience réussie et enrichissante, de nouvelles collaborations avec le laboratoire I3D, notamment pour la poursuite des travaux engagés, devraient être tout aussi fructueuses.

Bibliographie

[AFN94] AFNOR Norme expérimentale X60-010. *Maintenance - Concepts et définition des activités de maintenance*. AFNOR, premier tirage, Paris, 1994.

[AMA02] H. Amadou Boubacar. *Outils pour la détection de changements de comportement d'un système*. Rapport de DEA de l'université de Sciences et Technologies de Lille, 2002.

[AMB97] C. Ambroise. *Introduction à la reconnaissance statistique des formes*. Cours de l'Ecole des Mines de Paris, Mars 1997.

[BAS88] M. Basseville. « Detecting changes in signal and systems. A survey », *Automatica*, Vol. 24, n°3, pp309-339, 1988.

[BAS93] M. Basseville, I.V. Nikiforov. *Detection of abrupt change : Theory & application*, Prentice Hall, New Jersey, 1993.

[BEN99] R. Ben Ouaghram. *Contribution au Diagnostic des Machines Tournantes Complexes. Application à un Laminoir*. Thèse de l'université de Compiègne, France, 1999.

[BER95a] M. Bergot, L. Grudzien. « Sûreté et diagnostic des systèmes industriels. Principaux concepts, méthodes, techniques et outils », *Diagnostic et sûreté de fonctionnement*, Vol. 5 n°3, pp317-344, 1995.

[BER95b] E. Bernauer, H. Demmou. *Les méthodes de détection, localisation et diagnostic*. Rapport LAAS n°95473, Toulouse, France, 1995.

[BEZ80] J.C. Bezdek. « A convergence theorem for the fuzzy Isodata clustering algorithms », *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 2, n°1, pp1-8, 1980.

[BEZ81a] J.C. Bezdek. « Pattern recognition with fuzzy objective function algorithms », *Plenum Publishing Corporation*, New York, 1981.

[BEZ81b] J.C. Bezdek, C. Coray, R. Gunderson, J. Watson. « Detection and characterization of cluster substructure », *SIAM Journal of Applied Mathematics*, Vol. 40, pp339-372, 1981.

[BIE99a] C. Biernacki. *Choix de modèles en classification*. Thèse de l'Université Technologique de Compiègne, France, 1999.

[BIE99b] P. Biela. *Classification automatique d'observations multidimensionnelles par réseaux de neurones compétitifs*. Thèse de l'Université des Sciences et Technologies de Lille, France, 1999.

[BIL98] P. Billaudel, A. Devillez, G. Villermain Lecolier. « An unsupervised classification algorithm for the non elliptic classes », *EURISCON'98*, Grèce, 1998.

[BLA37] M. Black. « Vagueness : an exercise in logical analysis ». *Philosophy of science*, Vol. 4, pp427-455, 1937.

[BOU98] N. Boudaoud. *Conception d'un système de diagnostic adaptatif en ligne pour la surveillance des systèmes évolutifs*. Thèse de l'Université Technologique de Compiègne, France, 1998.

[BRU90] J. Brunet, M. Labarrère, D. Jaume, A. Rault, M. Vergé. *Détection et diagnostic de pannes.* Traité des nouvelles technologies, série Diagnostic et Maintenance, Hermès, 1990.

[CAR01] B. Cart, V. Gosseaume, F. Kogut-Kubiak, M-H. Toutin. *La maintenance industrielle. Une fonction en évolution, des emplois en mutation*. Centre d'Etudes et Recherche sur les Qualifications. Bref n°174, Avril 2001.

[CEL94] G. Celeux, G. Govaert. « Fuzzy clustering and mixture models », *In Proceedings of CompStat*, 1994.

[CHA83] C. Chabanon, B. Dubuisson. « Discrimination quadratique en reconnaissance des formes. Obtention de frontières du type hyper quadratiques de révolution », *RAIRO Automatic Systems Analysis and Control*, Vol. 17, 1983.

[CHI90] M. Chiollaz, B. Favre. *Caractérisation fine de bruits moteur par analyse temps-fréquence de Wigner-Ville*, Journées organisées au CETIM à Senlis, France, 1990.

[CHO70] C.K. Chow. « On optimum recognition error and reject tradeoff », *IEEE Transactions on Information Theory*, Vol. 16, pp41-46, 1970.

[COV67] T.M. Cover, P.E. Hart. « Nearest neighbour pattern classification », *IEEE Transactions on Information Theory*, Vol. 13, pp21-27, 1967.

[DAZ96] F. Dazy, J.F. Le Barzic, G. Saporta, F. Lavallard. *L'analyse des données évolutives. Méthodes et applications*. Editions Technip. Paris, 1996.

[DEV99] A. Devillez. *Contribution à la classification floue de données comportant des classes de forme quelconque. Application au développement d'un module d'aide à la décision.* Thèse de l'Université de Reims Champagne-Ardenne, France, 1999.

[DID71] E. Diday. « La méthode des nuées dynamiques », *Rev. Stat. Appliquée*, Vol. 19, n°2, p19-34, 1971.

[DOC81] P.G. Doctor, T.P. Harrington, T.J. Davis, C.J. Morris, D.W. Fraley. *Pattern recognition methods for classifying and sizing flaws using eddy-current data, eddy-current characterization of materials and structures.* ASTM STP 722, George Birnbaum and George Free, Eds., American society for testing and materials, pp464-483, 1981.

[DUB87] D. Dubois, H. Prade. *Théorie des possibilités. Application à la représentation des connaissances en informatique.* Deuxième édition, Masson, 1987.

[DUB90a] B. Dubuisson. *Diagnostic et reconnaissance des formes. Traité des nouvelles technologies*, Série Diagnostic et Maintenance, Hermès, 1990.

[DUB90b] B. Dubuisson, « Decision with reject options ». *In 5th European signal processing conference*, Barcelone, Espagne, 1990.

[DUB94] B. Dubuisson, M.H. Masson, C. Frélicot. *Diagnostic par une approche reconnaissance des formes*. Journées CNRS Sûreté, Surveillance, Supervision, 1994.

[DUD73] R.O. Duda, P.E. Hart. *Pattern Classification and Scene Analysis.* John Willey and Sons, Inc, New York, 1973.

[DUF94] O. Duffaut. *Problématique multi-modèle pour la génération d'arbres de test. Application au domaine automobile.* Thèse de l'Ecole Nationale Supérieure de l'Aéronautique et de l'Espace, 1994.

[DUN74] J. C. Dunn. « A fuzzy relative of the ISODATA process and its use in detecting compact well separated clusters ». *Journal of cybernetics*, Vol. 3, pp32-57, 1974.

[ELT98] T. Eltoft, J.P. Rui Defigueiredo. « A new neural network for Cluster-Detection-and-Labeling », *IEEE Transactions on Neural Networks,* Vol. 9, n°5, pp1021-1035, 1998.

[FAR89] H. Farreny. *Les systèmes experts. Principes et exemples.* Cépadues, 1989.

[FER67] T.S. Ferguson. *Mathematical statistics. A decision theory approach.* Academic Press Inc, New York, 1967.

[FIO99] A. Fiordaliso. *Systèmes flous et prévision de séries temporelles*. Hermès, Paris, France, 1999.

[FRA90] P.M. Frank. « Fault diagnosis in dynamic systems using analytical and knowledge-based redundancy », *Automatica*, Vol. 26, n°3, pp459-474, 1990.

[FRE92] C. Frélicot. *Un système adaptatif de diagnostic prédictif par reconnaissance des formes floues.* Thèse de l'Université Technologique de Compiègne, France, 1992.

[FRI95] H. Frigui, R. Krishnapuram. « A robust clustering algorithm based on the M-estimator », *In Proceedings of Neural, Parallel and Scientific Computations*, Atlanta, Georgia, 1995.

[FRI96] H. Frigui, R. Krishnapuram. « A robust algorithm for automatic extraction of unknown number of cluster from noisy data », *Pattern Recognition Letters*, Vol. 17, pp1223-1232, 1996.

[FRI97] H. Frigui. *New approaches for robust clustering and for estimating the optimal number of clusters*. Thèse de l'Université de Missouri-Columbia, 1997.

[FU74] K.S. Fu. *Syntactic methods in pattern recognition.* Academic Press. New York, 1974.

[FUK90] K. Fukunaga. *Introduction to statistical pattern recognition.* Academic Press, 2nd edition, New York, 1990.

[GAB00] B. Gabrys, A. Bargiela. « General fuzzy min-max neural network for clustering and classification », *IEEE Transactions on neural networks*, Vol. 11, n°3, pp769-783, 2000.

[GAC97] L. Gacogne. *Eléments de logique floue*. Editions Hermès, Paris, 1997.

[GAD95] B. Gaddouna. *Contribution au diagnostic des systèmes linéaires invariants à entrées inconnues. Application à un processus hydraulique.* Thèse de l'Institut National Polytechnique de Lorraine. Nancy 1, France, 1995.

[GAN87] K. Gana. *Suivi d'évolution et aide au pronostic en maintenance de système industriel.* Thèse de l'Université de Valenciennes et du Hainaut Cambrésis, France, 1987.

[GER88] J. Gertler. « Survey of model-based failure detection and isolation to complex plants », *IEEE Control Systems Magazine*, Vol. 8, n°6, pp3-11, 1988.

[GER90] J. Gertler, D. Singer. « A new structural framework for parity equation-based failure detection and isolation », *Automatica*, Vol. 26, n°2, pp. 381-388, 1990.

[GER98] J. Gertler. *Fault detection and diagnosis in Engineering systems.* Marcel Dekker, Inc., New York, USA, 1998.

[GK93] J. Gertler et M.M. Kunwer. « Optimal Residual Decoupling for Robust Fault Diagnosis », *Tooldiag'93 : International Conference on Fault Diagnosis*, Vol. 2, pp. 436-452. Toulouse, France, Avril 1993.

[GOM95] J.B. Gomm. « Process fault diagnosis using a self-adaptive neural network with on-line learning capabilities », *IFAC'95, Workshop on fault detection and supervision in the Chemical Process Industries*, pp78-83, NewCastle, Angleterre, 1995.

[GOM96] J.B. Gomm. « On-line learning for fault classification using adaptive neuro-fuzzy network », *IFAC'96*, pp175-180, San-Francisco, USA, 1996.

[GRP00] Groupement de Recherche en Productique. Rapport d'activité, 1997-2000. http://www.univ-valenciennes.fr/LGIL/GRP/sever.htm et http://www.laas.fr/~combacau/SPSF/sursup.html.

[GUN98] S. Gunn. *Support Vector Machines for classification and regression.* ISIS Rapport technique de l'Université de Southampton, 1998.

[GUS79] D. E. Gustafson, W. C. Kessel. « Fuzzy clustering with a fuzzy covariance matrix », *Proceedings of IEEE-CDC*, San Diego, 1979.

[HAM97] D. Hamad. *Réseaux de neurones pour la classification non supervisée.* HDR de l'Université des Sciences et Technologies de Lille, France, 1997.

[HO65] Y.C. Ho, R.L. Kashyap. « An algorithm for linear inequalities an its applications », *IEEE Transactions on Electronic Computers*, EC-14, pp683-688, 1965.

[HOP82] J.J. Hopfield. « Neural networks and physical systems with emergent collective computational abilities », *Proceedings of the National Academy of Sciences*, USA, Vol. 79, pp2554-2558, 1992.

[HRL00] Hydro Renée Leduc. *Accumulateurs hydropneumatiques. Précautions maintenance*. Documentation constructeur, Réf AQ55 G, France, 2000.

[ISE84] R. Isermann. « Process fault detection based on modeling and estimation methods », *Automatica*, Vol. 20, n°4, pp387-404, 1984.

[ISE93] R. Isermann. « Fault diagnosis of machines via parameter estimation and knowledge processing. Tutorial paper », *Automatica*, Vol. 29, n°4, pp815-835, 1993.

[JAU99] C. Jausions-Picaud. *Analyse en Composantes Curvilignes et représentation de données multidimensionnelles. Application au routage adaptatif.* Thèse de l'Institut National Polytechnique de Grenoble, France, 1999.

[KAD94] V. Kadirkamanathan. « A statistical inference growth criterion for the rbf network », *In proceedings of the IEEE Workshop on Neural Networks for Signal Processing*, Vol. 4, pp12-21, 1994.

[KAY99] M. Kaye. *Cavitation monitoring in hydraulic machines by vibration analysis.* Thèse de l'Université de Lausanne, rapport n°2112, Suisse, 1999.

[KHO97] L. Khodja. *Contribution à la classification floue non supervisée.* Thèse de l'Université de Savoie, France, 1997.

[KRI93] R. Krishnapuram, J.M. Keller. « A possibilistic approach to clustering », *IEEE Transactions of fuzzy systems*, Vol. 1, n°2, pp98-110, 1993.

[KLE98] F. Klein, *Etude comparative de méthodes de classification floues.* Thèse de l'Université de Reims Champagne-Ardenne, France, 1998.

[KOH84] T. Kohonen. *Self-organization and associative memory.* Springer Verlag, 1984.

[KOH86] T. Kohonen. *Learning Vector Quantization for pattern recognition.* Rapport technique de l'Université de Technologie d'Helsinki, Ref. TKK-F-A601, Finlande, 1986.

[LAC79] P. A. Lachenbruch, M. Goldenstein. « Discriminant analysis », *Biometrics*, Vol. 35, pp69-85, 1979.

[LEE91] S. Lee, R.M. Kil. « A gaussian potential function network with hierarchically self-organizing learning », *IEEE Transactions on Neural Networks*, Vol. 4, pp207-224, 1991.

[LJU99] L. Ljung. *System identification. Theory for the user.* Practice Hall Information, 2nd édtition, 1999.

[MAQ97] D. Maquin. *Diagnostic à base de modèles des systèmes technologiques.* Thèse de l'Institut National Polytechnique de Lorraine, Nancy, France, 1997.

[MAS96] M.H. Masson, B. Dubuisson, C. Frélicot. « Conception d'un module de reconnaissance des formes floues pour le diagnostic », *Journal Européen des Systèmes Automatisés, RAIRO-APII-JESA*, Vol. 30, n°2, pp319-341, 1996.

[MCC43] W.S. McCulloch, W. Pitts. « A logical calculus of the ideas immanent in nervous activity », *Bulletin of Mathematical Biophysics*, Vol. 9, pp127-147, 1943.

[MCL97] A. McLachlan. « An improved novelty criterion for ressource allocating networks », *In proceedings of the fifth International Conference on Artificial Neural Networks*, Vol. 440, p48-52, 1997.

[MOU93] G. Mourot. *Contribution au Diagnostic de Systèmes Industriels par Reconnaissance des Formes.* Thèse de l'Université Technologique de Compiègne, France, 1993.

[MOU01] D. Mourre, R. Burton. « Investigation of a neural network/statistical condition monitoring technique for a proportional solenoid valve », *Bath Workshop on power transmission and motion control*, 12-14 septembre 2001, Université de Bath, Canada, 2001.

[MOB92] R.K. Mobley. *La maintenance prédictive. Organisation industrielle.* Editions Masson. Paris, 1992.

[NGO98] J. Ngole, L. Tarassenko. « On-line resource reallocation in RBF classifiers », *In Proceedings of ICSC'98*, Tenerife, Espagne, 1998.

[OHA84] Y. Ohashi. « Fuzzy clustering and robust estimation », *In 9th Meeting of SAS Users Group International*, Floride, 1984.

[PAJ94] D. Pajani. *La thermographie infrarouge.* Les techniques de l'Ingénieur. Traité Mesures et Contrôles. R2740 et R2741, Avril 1985. Réactualisation de cette page en 1992 et 1994.

[PAL77] S. Pal. « Fuzzy sets and decision making approaches in vowel speaker recognition », *IEEE Transactions on Systems Man and Cybernetics*, Vol.7, pp625-629, 1977.

[PAT89] R.J. Patton, P.M. Frank, R.N. Clark. *Fault diagnosis in dynamic systems – Theory and application.* Prentice Hall, 1989.

[PAT91] R.J. Patton, J. Chen. « A review of parity space approaches to fault diagnosis », *In IFAC Safeprocess'91*, Vol. 1, pp239-255. Baden Baden, Allemagne, 1991.

[PAT94] R.J. Patton. « Robust model-based fault diagnosis. the state of the art », *Proceedings of IFAC Symposium on Fault Detection and Supervision and Safety for technical processes. Safeprocess'94*, pp1-24, 1994.

[PED90] W. Pedrycz. « Fuzzy sets in pattern recognition. Methodology and methods », *Pattern Recognition*, Vol. 23, n°1/2, pp121-146, 1990.

[PED96] W. Pedrycz. « Conditionnal Fuzzy C-means », *Pattern Recognition Letters*, Vol. 17, pp625-631, 1996.

[PEL93] M.A. Peltier, B. Dubuisson. « A human operator monitoring process based on a fuzzy approach », *Tooldiag'93 : International Conference on Fault Diagnosis*, Toulouse, 1993.

[PET99] S. Petit Renaud. *Application de la théorie des croyances et des systèmes flous à l'estimation fonctionnelle en présence d'informations incertaines ou imprécises.* Thèse de l'Université de Technologie de Compiègne, France, 1999.

[PHA98] D.T. Pham, A.B., Chan. « Control chart pattern recognition using a new type of self organizing neural networks », *Proc. Instn, Mech Engrs*, Vol. 212, n°1, pp115-127, 1998.

[PLA91] J. Platt. « A ressource-allocating network for function interpolation », *Neural Computation*, Vol. 3, n°2, pp213-225, 1991.

[PLO98] S. Ploix. *Diagnostic des systèmes incertains. Approche bornante.* Thèse de l'Université Henri Poincaré, CRAN, Nancy 1, France, 1998.

[POU96] H. Poulard. *Statistiques et réseaux de neurones pour un système de diagnostic. Application au diagnostic de pannes automobiles.* Thèse de l'Université Paul Sabatier, LAAS, Toulouse, France, 1996.

[REI82] D.L. Reilly, L.N. Cooper, C. Erlbaum. « A neural model for category learning», *Biological Cybernetics*, Vol. 45, pp35-41, 1982.

[REM97] J.F. Remm. *Extraction de connaissances par réseaux neuronaux. Application au domaine du radar.* Thèse de l'Université Henri Poincaré, Nancy 1, France, 1997.

[RIB02] J.C. Ribes, G. Delaunay, J. Delvaux, E. Merle, M. Mouillet. « Diagnostic par reconnaissance de formes de l'état de fonctionnement de l'accélérateur AIRIX », *Journal Européen des Systèmes Automatisés, APII-JESA*, 2002.

[ROS62] F. Rosenblatt. *Principles of neurodynamics. Perceptrons and the theory of brain mechanisms.* Spartan books, Washington DC, 1962.

[RUS69] E.H. Ruspini. « A new approach to clustering », *Information and Control*, Vol. 15, pp22-32, 1969.

[RUS98] E.H. Ruspini, P.P. Bonissone, W. Pedricz. *Handbook of fuzzy computation*, IOP Publishing Ltd, 1998.

[SAN97] J. Sandt, J. Rinkinen, J. Laukka. « Particle and water on-line monitoring for hydraulic system diagnosis », *The fifth Scandinavian International Conference on fluid power 1997*, Vol. 3, pp257-268. Linköping, Suède, 1997.

[SAP90] G. Saporta. *Probabilités, analyse de données et statistique.* Editions Technip, 1990.

[SCH99] E. Schaeffer. *Diagnostic des machines asynchrones. Modèles et outils paramétriques dédiés à la simulation et à la détection de défauts.* Thèse de l'Université de Nantes, France, 1999.

[SHA76] G. Shafer. *A mathematical theory of evidence.* Princeton University Press, Princeton, New Jersey, 1976.

[SHO76] E. Shortliffe. *Computer-based medical consultation : MYCIN.* Elsevier, New-York, USA, 1976.

[SIM92] P.K. Simpson. « Fuzzy min-max neural networks. Part 1 : classification », *IEEE Transactions on Neural Networks*, Vol. 3, pp776-786, 1992.

[SIM93] P.K. Simpson. « Fuzzy min-max neural networks. Part 2 : clustering », *IEEE Transactions on Fuzzy Systems*, Vol. 1, n°1, pp32-45, 1993.

[SPE90] D.F. Specht. « Probabilistic neural networks », *IEEE Transactions on Neural Networks*, Vol. 3, pp109-118, 1990.

[STE98] J.S. Stecki. *Total contamination control*. Fluid Power Net Publication, 177 pages, Australie, 1998.

[SUB92] M.C. Subner, M. Gabriel, L. Claude. « Utilisation de l'AMDEC pour générer la base de connaissances d'un système expert d'aide au diagnostic », *Diagnostic et sûreté de fonctionnement*, Vol. 2, n°2, pp133-153, 1992.

[THI97] S. Thiria, Y. Lechevallier, O. Gascuel, S. Canu. *Statistique et méthodes neuronales.* 2ème cycle et écoles d'ingénieurs. Editions Dunod. 1997.

[THO96] J.H. Thomas. *Etude de méthodes de diagnostic par reconnaissance de formes floues. Application à deux situations issues du domaine automobile.* Thèse de l'Université de Compiègne, France, 1996.

[TRA97] L. Travé-Massuyès, P. Dague, F. Guerrin. *Le raisonnement qualitatif.* Hermès, France, 1997.

[VEL98] L. G. Vela Valdes. *Etude et élaboration d'une approche fonctionnelle pour la localisation de défauts en diagnostic. Application à la simulation d'un moteur à courant continu.* Thèse de l'Université Henri Poincaré, Nancy 1, France, 1998.

[VIL98] L. Vilain. *Conception et Evaluation d'un Système de Surveillance. Application au Serrage Hydraulique.* Thèse de l'Université des Sciences et Technologies de Lille, 1998.

[WAL94] E. Walter, L. Pronzato. *Identification de modèles paramétriques à partir de données expérimentales. Modélisation Analyse Simulation Commande*, Ed. Masson, Paris, France, 1994.

[WEB99] P. Weber. *Diagnostic de procédé par l'analyse des estimations paramétriques de modèles de représentation à temps discret.* Thèse de l'Institut National Polytechnique de Grenoble, France, 1999.

[WIL76] A.S. Willsky. « A survey of design methods for failure detection in dynamic systems », *Automatica*, Vol. 12, pp601-611, 1976.

[WON01] C.C Wong, C.C. Chen, M.C. Su. « A novel algorithm for data clustering », *Pattern Recognition*, Vol. 34, pp425-442, 2001.

[ZAD65] L.A. Zadeh. « Fuzzy sets », *Informations and control*, Vol. 8, pp338-353, 1965.

[ZAD78] L.A. Zadeh. « Fuzzy set as a basis for a theory of possibility », *Fuzzy sets and systems*, Vol. 1, pp3-28, 1978.

[ZAD95] L. Zadeh. « Discussion : probability theory and fuzzy logic are complementary rather than competitive », *Technometrics*, Vol. 37, pp271-276, 1995.

[ZAV97] M.K. Zavarehi. *On-line condition monitoring and fault diagnosis in hydraulic system components using parameter estimation and pattern classification.* Thèse de l'université de British Columbia. Département d'ingénierie mécanique, Canada, 1997.

[ZIE95] S. Zieba. *Une méthode de suivi d'un phénomène évolutif. Application au diagnostic de la qualité d'usinage.* Thèse de l'Université Technologique de Compiègne, France, 1995.

[ZWI95] G. Zwingelstein, *Diagnostic des défaillances – Théorie et pratique pour les systèmes industriels.* Traité des Nouvelles Technologies, série Diagnostic et Maintenance. Hermès, Paris, 1995.

Publications liées à la réalisation des travaux de thèse

Année 2000

[LEC00] S. Lecoeuche, C. Lurette, S. Lalot. « Online identification and diagnostic method applied to thermal systems », *Proceedings of the International Conference on Engineering Applications of Neural Networks, EANN 2000*, pp155-162, 17-19 juillet, Kingston, Angleterre, 2000.

Année 2001

[LUR01a] C. Lurette, S. Lalot, S. Lecoeuche. « Detection and Diagnostic of Drifts by Unsupervised and Auto-adaptative Neural Network », *Proceedings of the International Conference on Engineering Applications of Neural Networks, EANN 2001*, pp239-246, 16-18 juillet, Cagliari, Sardaigne, 2001.

[LUR01b] C. Lurette, S. Lecoeuche. « Improvement of Cluster Detection and Labeling Neural Network by introducing elliptical basis functions », *Proceedings of the International Conference on Artificial Neural Networks, ICANN 2001*, pp196-202, 21-25 Août, Vienne, Autriche, 2001.

Année 2002

[LEC02a] S. Lecoeuche, « Unsupervised classifier for monitoring and diagnostic of time series », *Proceedings of the European Symposium on Artificial Neural Networks, ESANN 2002*, pp419-424, 24-26 avril, Bruges, Belgique, 2002.

[LEC02b] S. Lecoeuche, C. Lurette. « Classification non supervisée de données évolutives par architecture neuronale autoadaptative », *9ème rencontres de la Société Francophone de Classification, SFC 2002*, 16-18 septembre, Toulouse, France, 2002.

Année 2003

[LEC03a] S. Lecoeuche, C. Lurette. « Auto-Adaptative and Dynamical Clustering Neural Network », *International Conference on Artificial Neural Networks, ICANN 2003,* 26-29 juin, Istanbul, Turquie, 2003. Accepté pour présentation au format "Long paper".

[LUR03a] C. Lurette, S. Lecoeuche. « New supervision architecture based on on line modelization of non-stationary data », *International Conference on Engineering Applications of Neural Networks, EANN 2003*, 8-10 septembre, Malaga, Espagne, 2003. Accepté pour présentation.

[LUR03b] C. Lurette, S. Lecoeuche. « An unsupervised and auto-adaptive neural architecture for on-line monitoring of an hydraulic process », *Revue Engineering Applications of Artificial Intelligence*. Accepté pour publication en 2003.

Annexe 1 : Démonstration des relations itératives d'adaptation de l'algorithme FCM

La démonstration des relations itératives utilisées par l'algorithme FCM traduit la résolution du problème de minimisation du critère d'optimisation suivant :

$$J_{FCM}(U,G)=\sum_{k=1}^{K}\sum_{n=1}^{N}\left((u_{kn})^{m}.d_{A,kn}^{2}\right) \tag{1}$$

sous la contrainte de normalisation

$$\sum_{k=1}^{K}u_{kn}=1\ \ \forall n \tag{2}$$

1. Démonstration de la relation de mise à jour des centres G^k

La minimisation du critère J_{FCM} selon les centres G^k des classes s'obtient directement en annulant le gradient de J_{FCM} par rapport aux différents centres G^k :

$$\frac{\partial J_{FCM}(U,G)}{\partial G^{k}}=\sum_{n=1}^{N}\left((u_{kn})^{m}.\left(2.d_{A,kn}\right)\right)=0 \tag{3}$$

soit

$$\sum_{n=1}^{N}\left((u_{kn})^{m}.X^{n}\right)=\sum_{n=1}^{N}\left((u_{kn})^{m}.G^{k}\right) \tag{4}$$

et donc on obtient

$$G^{k}=\frac{\sum_{n=1}^{N}(u_{kn})^{m}}{\sum_{n=1}^{N}\left((u_{kn})^{m}.X^{n}\right)} \tag{5}$$

2. Démonstration de la relation de mise à jour des coefficients u_{kn}

Il s'agit ici d'un problème de minimisation du critère J_{FCM} sous la contrainte d'égalité (2). Pour cela, on utilise la technique des multiplicateurs de Lagrange qui est une méthode d'optimisation de critères sous des contraintes d'égalité ou d'ordre. L'utilisation de cette méthode se traduit par la minimisation de l'expression suivante où les termes λ_n sont les coefficients multiplicateurs de Lagrange :

$$J(U,\lambda)=\sum_{k=1}^{K}\sum_{n=1}^{N}\left((u_{kn})^{m}.d_{A,kn}^{2}\right)-\sum_{n=1}^{N}\left(\lambda_{n}.\left(\sum_{k=1}^{K}u_{kn}-1\right)\right) \tag{6}$$

Etant donné que les colonnes de la matrice de partition floue U sont indépendantes les unes des autres, on peut réduire le problème de minimisation à chacune des observations (colonnes), i.e. à N problèmes d'optimisation indépendants :

$$J_n(U_n, \lambda_n) = \sum_{k=1}^{K} (u_{kn})^m d^2_{A,kn} - \lambda_n \cdot \left(\sum_{k=1}^{K} u_{kn} - 1 \right) \tag{7}$$

où la notation U_n représente la $n^{ième}$ colonne de U. Ainsi, en annulant le gradient de J_n par rapport aux termes λ_n et u_{kn} on obtient respectivement les expressions :

$$\frac{\partial J_n(U_n, \lambda_n)}{\partial \lambda_n} = \sum_{k=1}^{K} u_{kn} = 1 \tag{8}$$

$$\frac{\partial J_n(U_n, \lambda_n)}{\partial u_{kn}} = m.(u_{kn})^{m-1}.d^2_{A,kn} - \lambda_n = 0 \tag{9}$$

L'égalité (9) permet alors d'exprimer le terme u_{kn} de la manière suivante :

$$u_{kn} = \left(\frac{\lambda_n}{m.d^2_{A,kn}} \right)^{1/m-1} \tag{10}$$

d'où en remplaçant dans (8) :

$$\sum_{j=1}^{K} u_{jn} = \sum_{j=1}^{K} \left(\frac{\lambda_n}{m.d^2_{A,jn}} \right)^{1/m-1} = 1 \quad \text{soit encore} \quad \left(\frac{\lambda_n}{m} \right)^{1/m-1} = \frac{1}{\sum_{j=1}^{K} \left(\frac{1}{d^2_{A,jn}} \right)^{1/m-1}} \tag{11}$$

et donc en combinant (10) et (11) on obtient :

$$u_{kn} = \frac{\left(\frac{1}{d^2_{A,kn}} \right)^{1/m-1}}{\sum_{j=1}^{K} \left(\frac{1}{d^2_{A,jn}} \right)^{1/m-1}} \tag{12}$$

soit

$$u_{kn} = \frac{1}{\sum_{j=1}^{K} \left(\frac{d^2_{A,kn}}{d^2_{A,jn}} \right)^{1/m-1}} \tag{13}$$

L'expression (13) traduit ainsi la relation de mise à jour de la matrice de partition permettant de minimiser le critère d'optimisation (1) sous la contrainte (2) selon les coefficients de la matrice de partition floue U.

Annexe 2 : Présentation de l'algorithme RCP avec la définition de la fonction ρ_k et des relations itératives d'adaptation

L'algorithme RCP fait parti des méthodes de coalescence au sein desquelles l'apprentissage est réalisé au cours d'un processus itératif de minimisation d'un critère d'optimisation. Toutefois, l'algorithme RCP présente la particularité de combiner les capacités de définition d'une partition en un nombre donné de classes de l'algorithme FCM (approche probabiliste) et les capacités de caractérisation des observations éloignées de l'algorithme PCM (approche possibiliste). Cette association des avantages respectifs des deux algorithmes est réalisée par la définition d'un nouveau critère d'optimisation noté J_{RCP} :

$$J_{RCP}(U,G) = \sum_{k=1}^{K}\sum_{n=1}^{N}\left((u_{kn})^m.\rho_k\left(d^2\left(X^n,G^k\right)\right)\right) \quad (1)$$

$$\text{avec} \begin{cases} u_{kn} \in [0,1] \ \forall k \ \ \forall n \\ \sum_{k=1}^{K} u_{kn} = 1 \ \ \forall n \in \{1,2,...,N\} \end{cases}$$

L'écriture de ce critère est assez proche de celui de l'algorithme FCM. La différence provient de l'introduction d'une fonction de perte ρ_k définie pour chacune des classes C^k. Cette fonction est adaptée à chaque itération au même titre que les coefficients u_{kn} et les paramètres β_k des classes. En fait, elle permet d'associer à chaque observation des possibilités d'appartenance w_{kn} en plus des coefficients d'appartenance u_{kn} traditionnels. Les pondérations w_{kn} ont la particularité d'être nulles pour les observations X^n trop éloignées du centre de la classe C^k.

1. Choix de la fonction ρ_k

Comme nous venons de le voir, la fonction ρ_k introduite lors de la définition du critère J_{RCP} a pour vocation de compléter le modèle de chacune des classes en lui associant des pondérations w_{kn} de nature possibiliste pour la prise en compte des observations éloignées. La fonction ρ_k est choisie telle que les pondérations w_{kn} traduisent pour chaque classe C^k une fonction de pondération w_k qui décroît avec la distance au centre des classes jusqu'à atteindre la valeur 0 à partir d'un certain seuil. En fait, comme nous le verrons dans les développements suivants, la fonction w_k n'est autre que la dérivée de la fonction ρ_k par rapport à la distance d. Cette dernière est définie de façon à respecter les contraintes suivantes sur la fonction w_k :

- $w_k\left(d^2\right)$ est une fonction monotone décroissante,
- $w_k\left(d^2\right)=0$ pour $d^2 > T_k + cS_k$,
- $w_k(0)=1$, $w_k(T_k)=0.5$, et $w_k'(0)=0$.

Les termes T_k et S_k traduisent respectivement des estimations de la moyenne et de l'écart type des distances au carré entre les observations affectées à la classe C^k et son centre G^k. Par ailleurs, le terme c est un facteur constant qui permet d'influer sur la zone d'influence des fonctions de pondération. A chaque itération du processus d'apprentissage, l'estimation des valeurs de T_k et S_k pour chaque classe C^k est très importante puisque celles-ci conditionnent l'allure de la fonction de pondération.

Un exemple de couple de fonctions ρ_k et w_k remplissant les différentes contraintes citées ci-dessus est présenté au travers des expressions suivantes puis illustré par la figure 1 :

$$w_k\left(d^2\right)=\begin{cases} 1-\dfrac{d^4}{2T_k^2} & si\ d^2 \in [0,T_k] \\ \dfrac{\left(d^2-(T_k+cS_k)\right)^2}{2c^2S_k^2} & si\ d^2 \in (Tk,T_k+cS_k] \\ 0 & si\ d^2 > T_k+cS_k \end{cases} \tag{2}$$

$$\rho_k\left(d^2\right)=\begin{cases} d^2-\dfrac{d^6}{6T_k^2} & si\ d^2 \in [0,T_k] \\ \dfrac{\left(d^2-(T_k+cS_k)\right)^3}{6c^2S_k^2} & si\ d^2 \in (Tk,T_k+cS_k] \\ \dfrac{5T_k+cS_k}{6}+K_k & si\ d^2 > T_k+cS_k \end{cases} \tag{3}$$

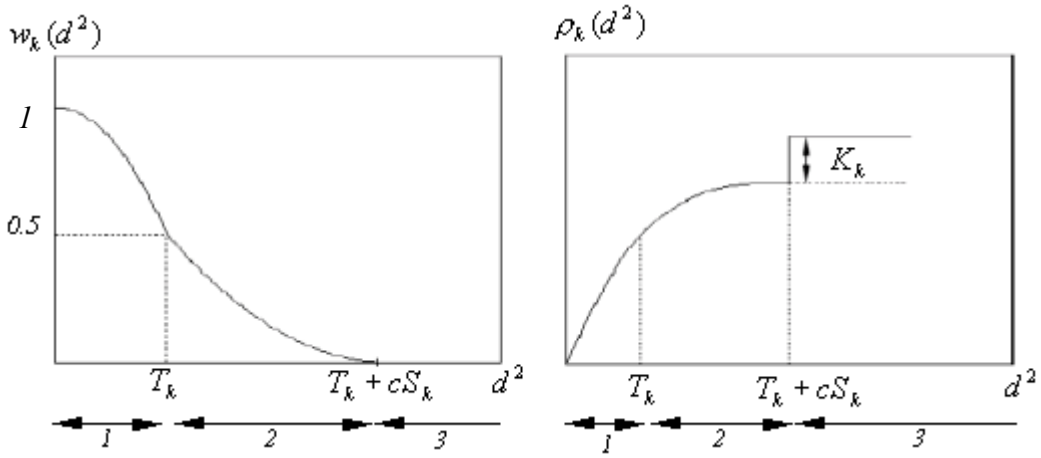


Figure 1 : Illustration des fonctions w_k et ρ_k pour la classe C^k.

Dans les expressions (2) et (3), le terme K_k est une constante définie pour chaque classe C^k et calculée selon l'expression :

$$K_k = max_{1 \le i \le K}\left(\frac{5T_i + cS_i}{6}\right) - \frac{5T_k + cS_k}{6} \qquad (4)$$

Cette constante permet d'éviter l'affectation d'observations bruitées à une classe en particulier en forçant chaque fonction ρ_k a la même valeur maximale.

Par ailleurs, les 3 zones repérées sur la figure 1 correspondent respectivement à une zone forte (zone 1) où les possibilités w_{kn} d'appartenance à la classe C^k sont supérieures à 0.5, une zone de doute (zone 2) où les possibilités d'appartenance à la classe C^k sont inférieures à 0.5 et une zone de rejet (zone 3) où les observations sont trop éloignées ($w_{kn} = 0$).

Enfin, pour conclure sur le choix de la fonction ρ_k, on peut noter que différentes situations peuvent se présenter :

- Si $\rho_k\left(d^2\right) = d^2 \ \forall k$ alors le critère J_{RCP} devient celui de l'***algorithme FCM***,
- Si on considère que toutes les classes sont indépendantes les unes des autres, alors on peut éliminer les coefficients d'appartenance u_{kn} et alors en choisissant pour ρ_k la fonction suivante :

$$\rho_k\left(d_{kn}^2\right) = \left(1 + \left(\frac{d_{kn}^2}{\eta_k}\right)^{1/m-1}\right)^{1-m} . d_{kn}^2$$

où le terme η_k se définit de la même façon que celui décrit pour l'algorithme PCM, on obtient un critère similaire à celui de l'***algorithme PCM***.

L'existence de ces deux cas particuliers montre bien que l'algorithme RCP est d'une certaine manière une généralisation des algorithmes FCM et PCM. Le choix de la fonction ρ_k étant établi, les trois paragraphes suivants exposent d'une part les relations de mise à jour des coefficients d'appartenance u_{kn} et des paramètres β_k des classes et d'autre part l'évolution de la fonction ρ_k entre chaque itération.

2. Mise à jour des coefficients u_{kn}

L'utilisation de la technique des multiplicateurs de Lagrange (voir annexe 1) permet d'établir la relation de mise à jour des coefficients d'appartenance u_{kn} minimisant J_{RCP} selon U sous la contrainte de normalisation :

$$u_{kn} = \frac{1}{\sum_{j=1}^{K}\left(\frac{\rho_k\left(d_{kn}^2\right)}{\rho_j\left(d_{jn}^2\right)}\right)^{\frac{1}{m-1}}} \quad \forall k \in \{1, 2, ..., K\} \text{ et } \forall n \in \{1, 2, ..., N\} \qquad (5)$$

3. Mise à jour des paramètres β_k des classes

La minimisation du critère J_{RCP} en fonction des paramètres β_k de la classe C^k est obtenue en fixant les coefficients de la matrice U et en annulant le gradient de J_{RCP} par rapport aux paramètres β_k, d'où l'expression :

$$\sum_{n=1}^{N}\left((u_{kn})^m . w_{kn} . \frac{\partial d_{kn}^2}{\partial \beta_k} \right) = 0 \quad \forall k \in \{1, 2, ..., K\} \text{ et } \forall n \in \{1, 2, ..., N\} \tag{6}$$

$$\text{où } w_{kn} = \frac{\partial \rho_k \left(d_{kn}^2 \right)}{\partial d_{kn}^2}$$

Ainsi, la détermination des relations de mise à jour des paramètres des classes fait apparaître des pondérations w_{kn} définissant des possibilités d'appartenance. La simplification de l'équation (6) nécessite le choix de la fonction ρ_k et de la distance d_{kn}. Le premier point a été abordé précédemment. Pour le second, deux cas de figure peuvent être envisagés selon que les classes sont de forme hypersphérique ou de forme hyperelliptique.

3.1. RCP pour des classes hypersphériques

Pour la détection de classes de forme hypersphérique, les paramètres β_k des classes sont les centres G^k. La distance utilisée est la distance euclidienne. L'équation (6) correspondant à la minimisation du critère J_{RCP} par rapport aux paramètres β_k de la classe C^k s'écrit alors :

$$\sum_{n=1}^{N}\left((u_{kn})^m . w_{kn} . \frac{\partial \left\| X^n - G^k \right\|^2}{\partial G^k} \right) = 0 \tag{7}$$

La relation de mise à jour des centres G^k des classes s'exprime alors de la façon suivante :

$$G^k = \frac{\sum_{n=1}^{N} \left((u_{kn})^m . w_{kn} . X^n \right)}{\sum_{n=1}^{N} (u_{kn})^m . w_{kn}} \quad \forall k \in \{1, 2, ..., K\} \tag{8}$$

3.2. RCP pour des classes hyperelliptiques

Pour la détection de classes hyperelliptiques, les paramètres β_k des classes sont les centres G^k et les matrices A_k puisque dans cette situation la distance utilisée est une A_k *-norm.* Cette distance est la même que celle utilisée pour l'algorithme FCMGK et nécessite la contrainte $\delta_k = det(A_k) = Cste$. Par conséquent, la mise en place des relations de mise à jour des paramètres $\beta_k = \{G^k, A_k\}$ des classes ne s'établit plus par la seule simplification de l'équation

(6) mais nécessite l'ajout d'une contrainte dans le critère J_{RCP}. La mise à jour des centres G^k des classes s'effectue toujours selon l'équation (8), mais la relation de mise à jour des matrices A_k s'obtient par la technique des multiplicateurs de Lagrange avec la contrainte supplémentaire $\delta_k = det(A_k)$ qui permet de définir l'équation (9) :

$$A_k = (\delta_k . det(\Sigma R_k))^{\frac{1}{d}} \Sigma R_k^{-1} \tag{9}$$

où la matrice ΣR_k définit la matrice de covariance floue robuste :

$$\sum R_k = \frac{\sum_{n=1}^{N} \left((u_{kn})^2 . w_{kn} . (X^n - G^k)(X^n - G^k)^T \right)}{\sum_{n=1}^{N} \left((u_{kn})^m . w_{kn} \right)} \tag{10}$$

Les relations (8) et (10) de mise à jour des paramètres β_k des classes font donc intervenir les coefficients d'appartenance u_{kn} mais également les possibilités d'appartenance w_{kn}. L'utilisation simultanée de ces deux ensembles de pondération permet de définir correctement les paramètres des classes tout en identifiant les observations éloignées. En effet, l'utilisation des fonctions ρ_k permet l'introduction des pondérations w_{kn} qui prennent une valeur nulle à partir d'un certain éloignement des centres des classes.

4. Mise à jour de la fonction ρ_k

Comme nous l'avons vu lors de la définition des fonctions de pondération, la définition des fonctions ρ_k nécessite pour chaque classe C^k l'estimation des grandeurs T_k et S_k. Celle-ci est réalisée à partir des observations affectées à la classe considérée. [FRI97] propose deux estimateurs couramment employés :

$$T_k = Med_{X^n \in X^{Ck}} \{ d_{kn}^2 \} \tag{11}$$

$$S_k = Med_{X^n \in X^{Ck}} \{ | d_{kn}^2 - Med_k | \} \tag{12}$$

où la notation Med définit la valeur médiane et la notation X^{Ck} représente l'ensemble des observations associées à la classe C^k défini par :

$$X^{Ck} = \{ X^n / d_{kn}^2 < d_{jn}^2 \ \forall k \neq j \} \tag{13}$$

Enfin, au-delà des termes T_k et S_k, un coefficient c est utilisé pour la définition des régions d'acceptation des fonctions ρ_k. Celui- ci prend généralement une valeur entre 4 et 12. Si c est trop grand, les fonctions de pondération w_k ont une région d'acceptation plus dilatée, c'est à dire que les observations éloignées ont des poids faibles mais non nuls ce qui perturbe l'estimation des paramètres des classes. D'un autre coté, si c est trop petit, peu d'observations sont

considérées et l'algorithme risque de converger prématurément vers un minimum. Pour pallier cette difficulté [FRI97] considère qu'il faut partir avec une initialisation de c à une valeur importante puis le faire décroître lors des itérations suivantes. Ceci a pour effet de permettre une plus forte mobilité lors des premières itérations, puis au cours des dernières itérations de stabiliser les estimations des paramètres des classes. La fonction de décroissance du coefficient c entre les itérations It et $It+1$ peut alors s'établir de la façon suivante :

$$c^{It+1} = max\left\{c_{min}, c^{It} - \Delta c\right\} \tag{14}$$

avec $\begin{cases} c^0 = 12 \\ c_{min} = 4 \\ \Delta c = 1 \end{cases}$

5. Contrôle de l'expansion des classes avec l'UcRCP

En relachant la contrainte de normalisation au sein de l'algorithme UcRCP, on autorise une expansion des classes. Toutefois, celle-ci doit être contrôlée afin d'éviter que certaines classes « incompatibles » se réunissent pour en former une seule. Pour cela, [FRI97] propose de limiter l'expansion en définissant une région d'acceptation au niveau de la fonction de pondération plus petite que celle utilisée avec le RCP. Ceci est obtenu en prenant par exemple $c^0 = 12$ et $c_{min} = 4$ comme paramètres initiaux lors de l'utilisation de l'UcRCP.

Annexe 3 : Démonstration des relations d'adaptation des prototypes pour le réseau AUDyC

Certaines règles d'apprentissage du réseau AUDyC (chapitre 4) font appel à des relations d'adaptation dynamique des paramètres (M^N, Σ^N) des prototypes. Les relations d'adaptation proposées dans un premier temps permettent uniquement d'ajuster les paramètres d'un prototype lors de l'ajout d'information à ce dernier. Afin de pallier cet inconvénient et d'associer un horizon temporel pour la définition des prototypes, de nouvelles règles ont été définies. Celles-ci permettent d'ajuster le modèle d'un prototype en tenant compte de l'ajout d'une nouvelle observation mais aussi du retrait de l'observation la plus ancienne.

1. Ajout d'une observation à un prototype

Soit un prototype défini par un ensemble de N observations $\{X^i, i = 1,\ldots,N\}$. Son centre M^N et sa matrice de covariance Σ^N sont déterminés selon les expressions :

$$M^N = \frac{1}{N}\sum_{i=1}^{N} X^i \tag{1}$$

$$\Sigma^N = \frac{1}{N-1}\sum_{i=1}^{N}\left(X^i - M^N\right)\left(X^i - M^N\right)^T \tag{2}$$

L'ajout d'une observation X^{N+1} à un prototype nécessite une adaptation des paramètres (M^{N+1}, Σ^{N+1}) de ce dernier afin d'intégrer la nouvelle information :

$$M^{N+1} = \frac{1}{N+1}\sum_{i=1}^{N+1} X^i \tag{3}$$

$$\Sigma^{N+1} = \frac{1}{N}\sum_{i=1}^{N+1}\left(X^i - M^{N+1}\right)\left(X^i - M^{N+1}\right)^T \tag{4}$$

Pour faciliter l'adaptation des paramètres d'un prototype lors de l'ajout d'une observation à celui-ci, il est toutefois plus intéressant de disposer d'une écriture de ces relations sous forme itérative. Le nouveau centre M^{N+1} et la nouvelle matrice de covariance Σ^{N+1} sont alors calculés directement en fonction de M^N, Σ^N et de l'observation supplémentaire X^{N+1}.

1.1. Mise à jour itérative de la moyenne ($M^N \rightarrow M^{N+1}$)

La relation de mise à jour itérative du centre M^{N+1} d'un prototype est obtenue en réécrivant l'expression (3) sous la forme :

$$M^{N+1} = \frac{1}{N+1}\left(\sum_{i=1}^{N} X^i + X^{N+1}\right)$$

soit $M^{N+1} = \frac{1}{N+1}\left(N.M^N + X^{N+1}\right)$

$$M^{N+1} = M^N + \frac{1}{N+1}\left(X^{N+1} - M^N\right) \tag{5}$$

1.2. Mise à jour itérative de la matrice de covariance ($\sum^N \rightarrow \sum^{N+1}$)

La relation de mise à jour itérative de la matrice de covariance $\sum^{N+1}$ d'un prototype est obtenue en présentant l'expression (4) sous la forme :

$$\sum^{N+1} = \frac{1}{N}\left(\sum_{i=1}^{N}\left(X^i - M^{N+1}\right).\left(X^i - M^{N+1}\right)^T + \left(X^{N+1} - M^{N+1}\right).\left(X^{N+1} - M^{N+1}\right)^T\right)$$

$$\text{soit } \sum^{N+1} = \frac{1}{N}(A+B) \tag{6}$$

$$\text{avec } \begin{cases} A = \sum_{i=1}^{N}\left(X^i - M^{N+1}\right).\left(X^i - M^{N+1}\right)^T \\ B = \left(X^{N+1} - M^{N+1}\right).\left(X^{N+1} - M^{N+1}\right)^T \end{cases}$$

1.2.1. Calcul du terme A

$$A = \sum_{i=1}^{N}\left(X^i - M^{N+1}\right).\left(X^i - M^{N+1}\right)^T$$

$$A = \sum_{i=1}^{N}\left(X^i - M^N - \frac{1}{N+1}\left(X^{N+1} - M^N\right)\right).\left(X^i - M^N - \frac{1}{N+1}\left(X^{N+1} - M^N\right)\right)^T$$

$$\begin{aligned} A = & \sum_{i=1}^{N}\left(X^i - M^N\right).\left(X^i - M^N\right)^T \\ & - \frac{2}{N+1}\sum_{i=1}^{N}\left(X^i - M^N\right).\left(X^{N+1} - M^N\right)^T \\ & + \frac{N}{(N+1)^2}\left(X^{N+1} - M^N\right).\left(X^{N+1} - M^N\right)^T \end{aligned}$$

et donc en remarquant que $\sum_{i=1}^{N}\left(X^i - M^N\right) = 0$, on obtient :

$$A = (N-1)\sum^N + \frac{N}{(N+1)^2}\left(X^{N+1} - M^N\right).\left(X^{N+1} - M^N\right)^T \tag{7}$$

1.2.2. Calcul du terme B

$$B = \left(X^{N+1} - M^{N+1}\right).\left(X^{N+1} - M^{N+1}\right)^T$$

$$B = \left(X^{N+1} - M^N - \frac{1}{N+1}\left(X^{N+1} - M^N\right)\right).\left(X^{N+1} - M^N - \frac{1}{N+1}\left(X^{N+1} - M^N\right)\right)^T$$

$$B = \frac{N^2}{(N+1)^2}\left(X^{N+1} - M^N\right)\left(X^{N+1} - M^N\right)^T$$

et donc d'après (6), on a :

$$\Sigma^{N+1} = \frac{N-1}{N}\Sigma^{N} + \frac{1}{N}\left(\frac{N}{(N+1)^2} + \frac{N^2}{(N+1)^2}\right)\left(X^{N+1} - M^{N}\right)\left(X^{N+1} - M^{N}\right)^{T}$$

soit après simplification :

$$\Sigma^{N+1} = \frac{N-1}{N}\Sigma^{N} + \frac{1}{N+1}\left(X^{N+1} - M^{N}\right)\left(X^{N+1} - M^{N}\right)^{T} \tag{8}$$

2. Ajout et retrait d'une observation

Soit un prototype défini sur un horizon de largeur N regroupant les observations $\{X^i, i = k-N+1,\ldots,k\}$. Son centre M^N et sa matrice de covariance Σ^N sont déterminés par :

$$M^{N} = \frac{1}{N}\sum_{i=k-N+1}^{k} X^{i} \tag{9}$$

$$\Sigma^{N} = \frac{1}{N-1}\sum_{i=k-N+1}^{k}\left(X^{i} - M^{N}\right)\left(X^{i} - M^{N}\right)^{T} \tag{10}$$

L'ajout d'une nouvelle observation X^{k+1} à ce prototype, accompagné du retrait de l'observation X^{k-N+1} la plus ancienne, nécessitent une adaptation des paramètres $\left(M^{N^*}, \Sigma^{N^*}\right)$ de ce dernier :

$$M^{N^*} = \frac{1}{N}\sum_{i=k-N+2}^{k+1} X^{i} \tag{11}$$

$$\Sigma^{N^*} = \frac{1}{N-1}\sum_{i=k-N+2}^{k+1}\left(X^{i} - M^{N^*}\right)\left(X^{i} - M^{N^*}\right)^{T} \tag{12}$$

2.1. Mise à jour itérative de la moyenne ($M^N \rightarrow M^{N^*}$)

La relation de mise à jour itérative du centre M^{N^*} d'un prototype est obtenue en récrivant l'expression (11) sous la forme :

$$M^{N^*} = \frac{1}{N}\left(\sum_{i=k-N+1}^{k} X^{i} + X^{k+1} - X^{k-N+1}\right)$$

$$M^{N^*} = \frac{1}{N}\left(N.M^{N} + X^{k+1} - X^{k-N+1}\right)$$

$$M^{N^*} = M^{N} + \frac{1}{N}\left(X^{k+1} - X^{k-N+1}\right) \tag{13}$$

soit encore :

$$M^{N^*} = M^{N} + \frac{1}{N}\left(\delta X^{+} - \delta X^{-}\right) \tag{14}$$

en posant $\begin{cases} \delta X^+ = X^{k+1} - M^N \\ \delta X^- = X^{k-N+1} - M^N \end{cases}$

2.2. Mise à jour itérative de la matrice de covariance ($\sum^N \rightarrow \sum^{N^*}$)

La relation de mise à jour itérative de la matrice de covariance $\sum^{N^*}$ d'un prototype est obtenue en récrivant l'expression (12) sous la forme :

$$\begin{aligned} \sum^{N^*} = \frac{1}{N-1}\Bigg(& \sum_{i=k-N+1}^{k} \left(X^i - M^{N^*}\right).\left(X^i - M^{N^*}\right)^T \\ & + \left(X^{k+1} - M^{N^*}\right).\left(X^{k+1} - M^{N^*}\right)^T \\ & - \left(X^{k-N+1} - M^{N^*}\right).\left(X^{k-N+1} - M^{N^*}\right)^T \Bigg) \end{aligned}$$

$$\text{soit } \sum^{N^*} = \frac{1}{N-1}(A + B - C) \qquad (15)$$

$$\text{avec } \begin{cases} A = \sum_{i=k-N+1}^{k} \left(X^i - M^{N^*}\right).\left(X^i - M^{N^*}\right)^T \\ B = \left(X^{k+1} - M^{N^*}\right).\left(X^{k+1} - M^{N^*}\right)^T \\ C = \left(X^{k-N+1} - M^{N^*}\right).\left(X^{k-N+1} - M^{N^*}\right)^T \end{cases}$$

2.2.1. Calcul du terme A

$$A = \sum_{i=k-N+1}^{k} \left(X^i - M^{N^*}\right).\left(X^i - M^{N^*}\right)^T$$

$$A = \sum_{i=k-N+1}^{k} \left(X^i - M^N - \frac{1}{N}\left(\delta X^+ - \delta X^-\right)\right).\left(X^i - M^N - \frac{1}{N}\left(\delta X^+ - \delta X^-\right)\right)^T$$

$$\begin{aligned} A = & \sum_{i=k-N+1}^{k} \left(X^i - M^N\right).\left(X^i - M^N\right)^T \\ & - \frac{2}{N} \sum_{i=k-N+1}^{k} \left(X^i - M^N\right).\left(\delta X^+ - \delta X^-\right)^T \\ & + \frac{1}{N^2} \sum_{i=k-N+1}^{k} \left(\delta X^+ - \delta X^-\right).\left(\delta X^+ - \delta X^-\right)^T \end{aligned}$$

et donc en remarquant que $\sum_{i=1}^{N} \left(X^i - M^N\right) = 0$, on obtient :

$$A = \sum_{i=k-N+1}^{k} \left(X^i - M^N\right).\left(X^i - M^N\right)^T + \frac{1}{N}\left(\delta X^+ - \delta X^-\right).\left(\delta X^+ - \delta X^-\right)^T$$

soit après développement :

$$A = (N-1).\sum^N + \frac{1}{N}\left(\delta X^+.\delta X^{+T} - \delta X^+.\delta X^{-T} - \delta X^-.\delta X^{+T} + \delta X^-.\delta X^{-T}\right) \qquad (16)$$

2.2.2. Calcul du terme B

$$B = \left(X^{k+1} - M^{N^*}\right).\left(X^{k+1} - M^{N^*}\right)^T$$

$$B = \left(X^{k+1} - M^N - \frac{1}{N}\left(\delta X^+ - \delta X^-\right)\right).\left(X^{k+1} - M^N - \frac{1}{N}\left(\delta X^+ - \delta X^-\right)\right)^T$$

$$B = \left(\delta X^+ - \frac{1}{N}\left(\delta X^+ - \delta X^-\right)\right).\left(\delta X^+ - \frac{1}{N}\left(\delta X^+ - \delta X^-\right)\right)^T$$

$$B = \frac{1}{N^2}\left((N-1).\delta X^+ + \delta X^-\right).\left((N-1).\delta X^+ + \delta X^-\right)^T$$

$$B = \frac{1}{N^2}\left((N-1)^2.\delta X^+.\delta X^{+T} + (N-1).\delta X^+.\delta X^{-T} + (N-1).\delta X^-.\delta X^{+T} + \delta X^-.\delta X^{-T}\right) \qquad (17)$$

2.2.3. Calcul du terme C

$$C = \left(X^{k-N+1} - M^{N^*}\right).\left(X^{k-N+1} - M^{N^*}\right)^T$$

$$C = \left(X^{k-N+1} - M^N - \frac{1}{N}\left(\delta X^+ - \delta X^-\right)\right).\left(X^{k-N+1} - M^N - \frac{1}{N}\left(\delta X^+ - \delta X^-\right)\right)^T$$

$$C = \left(\delta X^- - \frac{1}{N}\left(\delta X^+ - \delta X^-\right)\right).\left(\delta X^- - \frac{1}{N}\left(\delta X^+ - \delta X^-\right)\right)^T$$

$$C = \frac{1}{N^2}\left((N+1).\delta X^- - \delta X^+\right).\left((N+1).\delta X^- - \delta X^+\right)^T$$

$$C = \frac{1}{N^2}\left((N+1)^2.\delta X^-.\delta X^{-T} - (N+1).\delta X^-.\delta X^{+T} - (N+1).\delta X^+.\delta X^{-T} + \delta X^+.\delta X^{+T}\right) \qquad (18)$$

et donc d'après (15), on a :

$$\begin{aligned}\Sigma^{N^*} = \Sigma^N &+ \frac{1}{N-1}\left(\frac{1}{N} + \frac{(N-1)^2}{N^2} - \frac{1}{N^2}\right).\delta X^+.\delta X^{+T} \\ &+ \frac{1}{N-1}\left(\frac{1}{N} + \frac{1}{N^2} - \frac{(N+1)^2}{N^2}\right).\delta X^-.\delta X^{-T} \\ &+ \frac{1}{N-1}\left(-\frac{1}{N} + \frac{N-1}{N^2} + \frac{N+1}{N^2}\right).\delta X^+.\delta X^{-T} \\ &+ \frac{1}{N-1}\left(-\frac{1}{N} + \frac{N-1}{N^2} + \frac{N+1}{N^2}\right).\delta X^-.\delta X^{+T}\end{aligned}$$

$$\begin{aligned}\Sigma^{N^*} = \; & \Sigma^{N} \\ & + \frac{1}{N}.\delta X^{+}.\delta X^{+^T} \\ & - \frac{N+1}{N(N-1)}.\delta X^{-}.\delta X^{-^T} \\ & - \frac{1}{N(N-1)}.\delta X^{+}.\delta X^{-^T} \\ & - \frac{1}{N(N-1)}.\delta X^{-}.\delta X^{+^T}\end{aligned}$$

on obtient alors l'écriture simplifiée suivante :

$$\Sigma^{N^*} = \Sigma^{N} + \Delta X.Q^{-1}.\Delta X^{T} \tag{19}$$

avec $Q^{-1} = \frac{1}{N}\begin{pmatrix} 1 & \frac{1}{N-1} \\ \frac{1}{N-1} & -\frac{N+1}{N-1} \end{pmatrix}$ et $\Delta X = \begin{bmatrix} \delta X^{+} & \delta X^{-} \end{bmatrix}$

2.2.4. Mise à jour itérative de la matrice de covariance inverse ($\Sigma^{N-1} \rightarrow \Sigma^{N-1^*}$)

De plus, afin de faciliter le calcul du degré d'appartenance d'une observation à un prototype selon la distance de Mahalanobis, une relation itérative peut être définie pour la matrice de covariance inverse. Cette relation est obtenue en utilisant le lemme d'inversion matricielle :

$$A^{-1} = B^{-1} + C^{T}D^{-1}C \;\Rightarrow A = B - BC^{T}\left(D + CBC\right)^{-1}CB \tag{20}$$

Ainsi, selon l'analogie suivante

$$\begin{cases} A^{-1} = \Sigma^{N^*} \\ B^{-1} = \Sigma^{N} \\ C = \Delta X^{T} \\ D^{-1} = Q^{-1} \end{cases}$$

on obtient la relation :

$$\Sigma^{N-1^*} = \Sigma^{N-1} - \Sigma^{N-1}.\Delta X.\left(Q + \Delta X^{T}.\Sigma^{N-1}.\Delta X\right)^{-1}.\Delta X^{T}.\Sigma^{N-1} \tag{21}$$

soit encore

$$\Sigma^{N-1^*} = \Sigma^{N-1} - K.\Delta X^{T}.\Sigma^{N-1} \tag{22}$$

avec $K = \Sigma^{N-1}.\Delta X.\left(Q + \Delta X^{T}.\Sigma^{N-1}.\Delta X\right)^{-1}$ (23)

Développement d'une technique neuronale auto-adaptative pour la classification dynamique de données évolutives. Application à la supervision d'une presse hydraulique.

Résumé :

Le caractère évolutif des procédés industriels rend leur supervision délicate puisque les modes de fonctionnement existants évoluent et que d'autres peuvent apparaitre. L'étude présentée porte sur la conception d'une structure adaptative de supervision permettant la surveillance, la détection et le diagnostic de systèmes évolutifs.

La structure proposée est constituée de trois modules : modélisation, surveillance et diagnostic. En raison de l'aspect évolutif de la connaissance sur les modes de fonctionnement, l'accent a été porté sur la définition d'une technique de modélisation dynamique basée sur une approche par reconnaissance de formes. L'architecture neuronale développée, le réseau AUDyC, autorise la classification dynamique de données évolutives via un ensemble de règles d'apprentissage non supervisées permettant son auto-adaptation.

La première partie introduit les enjeux industriels. La deuxième partie expose des approches d'élaboration d'un système de diagnostic industriel avec en particulier celle par reconnaissance de formes. La troisième partie présente des techniques de classification à apprentissage hors ligne, puis deux techniques neuronales autorisant la prise en compte de données évolutives. La quatrième partie décrit l'architecture évolutive du réseau AUDyC construite selon une approche multiprototype. Les règles d'apprentissage autorisent l'adaptation, la création, la fusion et la suppression de prototypes ou de classes, et sont structurées en trois phases : classification, fusion et évaluation. La cinquième partie expose la structure adaptative de supervision élaborée à partir du réseau AUDyC, puis décrit les performances de celle-ci pour la supervision d'une presse hydraulique.

Mots-clés :

Classification dynamique / Réseau de neurones / Architecture auto-adaptative / Données évolutives / Apprentissage non supervisé / Reconnaissance de formes / Supervision / Procédé hydraulique.

Development of an auto-adaptive neural technique for the dynamical clustering of evolutionary data. Application to supervision of a hydraulic press.

Abstract :

The evolutionary aspect of industrial processes makes their supervision delicate due to the evolution of the existing functioning modes and the apparition of new modes. The presented study deals with the conception of an adaptive structure of supervision that allows the monitoring, the detection and the diagnostic of non-stationary systems.

The developed structure is composed of three modules: modeling, monitoring and diagnostic. Due to the evolutionary aspect of the knowledge on the functioning modes, the attention has been focused on the definition of a technique of dynamical modeling based on an approach by pattern recognition. The developed neural architecture, the AUDyC network, authorizes the dynamical clustering of evolutionary data via a set of unsupervised learning rules allowing its auto-adaptation.

The first part introduces the industrial stakes. The second part explains different approaches of elaboration of an industrial diagnostic system with in particular the one by pattern recognition. The third part presents techniques of clustering with off-line learning and two neural techniques authorizing the consideration of evolutionary data. The fourth part describes the evolutionary architecture of the AUDyC network built according to a multiprototype approach. The learning rules authorize the adaptation, the creation, the fusion and the elimination of prototypes or classes, and are structured in three stages : classification, merging and evaluation. The fifth part exposes the adaptive structure of supervision elaborated from the AUDyC network, and describes its performances for the supervision of a hydraulic press.

Keywords :

Dynamical clustering / Neural network / Auto-adaptive architecture / Evolutionary data / Unsupervised learning / Pattern recognition / Supervision / Hydraulic process.