ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

MÉMOIRE PRÉSENTÉ À
L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

COMME EXIGENCE PARTIELLE
À L'OBTENTION DE LA MAÎTRISE EN
GÉNIE DE LA PRODUCTION AUTOMATISÉE
M.Ing.

PAR
Mathieu BEAUCHEMIN-TURCOTTE

CONTRÔLE EN FIN DE CYCLE PAR APPRENTISSAGE ITÉRATIF VIA LA LOGIQUE FLOUE APPLIQUÉE AU CONTRÔLE EN TEMPÉRATURE D'UN FOUR DE THERMOFORMAGE

MONTRÉAL, LE 17 JUIN 2010

© Mathieu Beauchemin-Turcotte, 2010

PRÉSENTATION DU JURY

CE MÉMOIRE A ÉTÉ ÉVALUÉ

PAR UN JURY COMPOSÉ DE

M. Guy Gauthier, directeur de mémoire
Département de génie de la production automatisée à l'École de technologie supérieure

M. Robert Sabourin, Président du jury
Département de génie de la production automatisée à l'École de technologie supérieure

M. Benoit Boulet, examinateur externe
Département de génie électrique et informatique à l'université McGill

IL A FAIT L'OBJET D'UNE SOUTENANCE DEVANT JURY ET PUBLIC

LE 17 MAI 2010

À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

REMERCIEMENTS

Je désire tout d'abord remercier mon directeur de maîtrise Guy Gauthier, professeur au département de génie de la production automatisée à l'École de technologie supérieure, pour son support constant ainsi que pour sa flexibilité.

Mon projet d'étude a été réalisable grâce au financement du Conseil de recherches en sciences naturelles et en génie du Canada (CRSNG), du fond de recherche sur la nature et la technologie (FQRNT) ainsi que l'École de technologie supérieure.

Finalement, je désire remercier ma conjointe, Anne-Marie, pour son support et la révision de mes écrits.

CONTRÔLE EN FIN DE CYCLE PAR APPRENTISSAGE ITÉRATIF VIA LA LOGIQUE FLOUE APPLIQUÉE AU CONTRÔLE EN TEMPÉRATURE D'UN FOUR DE THERMOFORMAGE

Mathieu BEAUCHEMIN-TURCOTTE

RÉSUMÉ

Le procédé de thermoformage requiert le chauffage de feuilles de plastique pour ensuite les mouler par vacuum. L'opérateur ajuste les consignes de température des éléments chauffant manuellement. Il doit donc corriger continuellement les consignes au cours de son quart de travail, car la température ambiante change. Un mauvais chauffage de la pièce entraîne des problèmes de qualité et des pièces défectueuses. Par conséquent, le coût d'opération des compagnies augmente.

L'objectif principal du projet est d'automatiser les réglages faits par l'opérateur en concevant une commande en fin de cycle par apprentissage successif basée sur la logique floue. Un aspect intéressant de la logique floue est son caractère moins mathématique et plus intuitif. En effet, nous proposons dans ce mémoire de maîtrise une méthode de développement d'un module de découplage flou utilisant uniquement des données mesurées sur le procédé. Ce module simplifie l'implantation de la commande puisque le système équivalent devient un ensemble de systèmes simple entrée et simple sortie. L'approche proposée atténue l'effet des perturbations dans une étendue acceptable, sous certaines contraintes, en moins de cinq cycles. Ainsi, les problèmes de qualité devraient diminuer, car les paramètres du procédé seront constamment optimaux malgré le changement de l'environnement.

TERMINAL ITERATIVE LEARNING CONTROL BASED ON FUZZY LOGIC APPLIED TO TEMPERATURE CONTROL OF AN THERMOFORMING OVEN

Mathieu BEAUCHEMIN-TURCOTTE

ABSTRACT

The thermoforming process involves plastic sheet heating in order to mould it by vacuum. The operator sets heaters temperature manually. Therefore, the operator must continuously correct the settings during his workday because the ambient temperature varies. An inaccurate heating process of the plastic sheet leads to quality problems and flaws. Consequently, the operating cost increases.

The main goal of this project is to automate the settings usually made by the operator himself. To achieve this goal, we develop a terminal iterative learning control based on fuzzy logic. The intuitive character of fuzzy logic is a really interesting aspect since it is less mathematic. Indeed, we propose in this project a method to develop a fuzzy decoupling module. We only used the data from the thermoforming oven to obtain our module. This module simplifies the implantation of the controller since the equivalent systems become single-in and single-out. Under specific constrains, the proposed approach reduces perturbation effects within an acceptable range of 5°C in less than five heating cycles. Thus, the quality problems should decrease because the settings are continuously optimized even though the environment changes.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

Page

LISTE DES FIGURES

Page

LISTE DES ABRÉVIATIONS, SIGLES ET ACRONYMES

AG	Algorithme génétique
AGS	Algorithme génétique simple
FIS	« Fuzzy inference system »
ILC	« Iterative learning control »
IMC	« Internal model control »
LTI	« Linear time invariant »
TILC	« Terminal iterative learning control »
TSK	Takagi-Sugeno-Kwan

LISTE DES SYMBOLES ET DES VARIABLES

α_i	Coefficient du polynôme de régression obtenu par krigeage
β_i	Coefficient d'ajustement du n^{ième} test obtenu par krigeage
Δuk	Variation de la commande (TILC)
φ_i	Changement de variables d'entrée pour définir une position relative
a_i	Coefficient du polynôme décrivant le comportement de la i^{ème} sortie
b_i	Coefficient du polynôme décrivant le comportement de la i^{ème} sortie
c_i	Coefficient du polynôme décrivant le comportement de la i^{ème} sortie
d_i	Coefficient du polynôme décrivant le comportement de la i^{ème} sortie
f_k^i	$k^{ème}$ fonction d'appartenance du système d'inférence flou inverse
K_i	Le gain sur l'erreur au i^{ème} cycle antérieur de la commande de la commande TILC
L_i	Le gain sur la commande appliquée au i^{ème} cycle antérieur de la commande de la commande TILC
N	Ordre du contrôleur TILC
P	Coefficient proportionnel à la variance des données (krigeage)
r_k^i	$k^{ème}$ fonction définissant le conséquent du système d'inférence flou inverse
u_i	Vecteur d'entrée correspondant aux consignes de température
v_i	Vecteur d'entrée correspondant au consignes de température en amont du module de découplage
$x_{i,n}$	Consigne de température du i^{ème} élément chauffant du n^{ième} test
$\hat{y}$	Vecteur de sortie estimé par une fonction linéaire

y_d	Vecteur de sortie désirée correspondant au profil de température
y	Vecteur de sortie correspondant à la température à la surface de la feuille de plastique
y_n	Température mesuré pour une zone au $n^{ième}$ test
y_f	Vecteur de sortie de température mesurée à la fin du cycle

INTRODUCTION

Le procédé de thermoformage requiert le chauffage de feuilles de plastique. Pour le moment, l'opérateur ajuste les consignes de température des éléments chauffant manuellement. L'opérateur doit donc corriger continuellement les consignes, car la température ambiante change. Un mauvais chauffage de la pièce entraîne des problèmes de qualité et des pièces défectueuses. Par conséquent, le coût d'opération des compagnies augmente.

Le problème fondamental en contrôle est qu'il faut, la plupart du temps, obtenir le modèle mathématique avant de débuter le processus de conception. Cette constatation est surtout véridique dans le cas des systèmes à multiples entrées et multiples sorties. Dans ces cas, le système est, en plus, souvent non-linéaire. Donc, la modélisation de systèmes complexes devient très longue et demande une période de validation pour garantir une précision et une fiabilité suffisante.

L'objectif principal du projet est de concevoir des contrôleurs à commande en fin de cycle par apprentissages successifs, ou « Terminal Iterative Learning Control » (TILC), basés sur la logique floue. Un aspect intéressant de la logique floue est son caractère moins mathématique et plus intuitif. Malgré la méconnaissance du modèle mathématique, l'opérateur parvient à ajuster convenablement les paramètres du procédé. Avec cette approche, on reproduira le comportement de l'opérateur lors de l'ajustement des consignes.

Maintenant, introduisons les notions de base nécessaire à la bonne compréhension de nos travaux. Les deux principaux sujets identifiés au paragraphe précédent sont le système d'inférence floue, ou « fuzzy inference system » (FIS), et la commande TILC. Nous traiterons tout d'abord les FIS.

L'utilisation de la logique floue est répandue dans les biens de consommations usuelles tels que les lave-linges « intelligents ». Dans le milieu industriel, la commande floue est utilisée dans le traitement des eaux (Cordon *et al.*, 2001, p. 39). Munataka (1994, p. 70) relèvent

également plusieurs autres applications dans divers domaines, tel que la commande et la reconnaissance de formes. De plus, ils constatent une progression de l'utilisation de cette approche depuis 1986, soit 21 ans après les premiers développements théoriques. La principale motivation de cette approche est qu'elle ne nécessite pas de modèle mathématique du système à contrôler. Ainsi, le temps de développement est généralement plus court que les autres méthodes (Driankov, Hellendoorn et Reinfrank, 1996, p. 4). Nous espérons profiter de cet avantage dans le cadre de nos travaux.

Les deux types de FIS les plus populaires sont ceux de Mamdani et de Takagi-Sugeno-Kang (TSK). Dans le cadre de ce projet, nous avons décidé d'utiliser des FIS de type TSK, car ce sont des interpolateurs entre différentes conditions d'opérations (Babuška, Jager et Verbruggen, 1994). De plus, la continuité de la surface de commande est garantie lorsque l'on utilise le TSK et il est mieux adapté pour un apprentissage par les données (Mathworks, N.d.). Par ailleurs, ce type de FIS est beaucoup moins lourd que le Mamdani. Toutefois, les opérateurs utilisés sont différents et surtout, l'interprétation du fonctionnement est différente. La figure 0 illustre le processus d'évaluation d'une règle d'inférence.

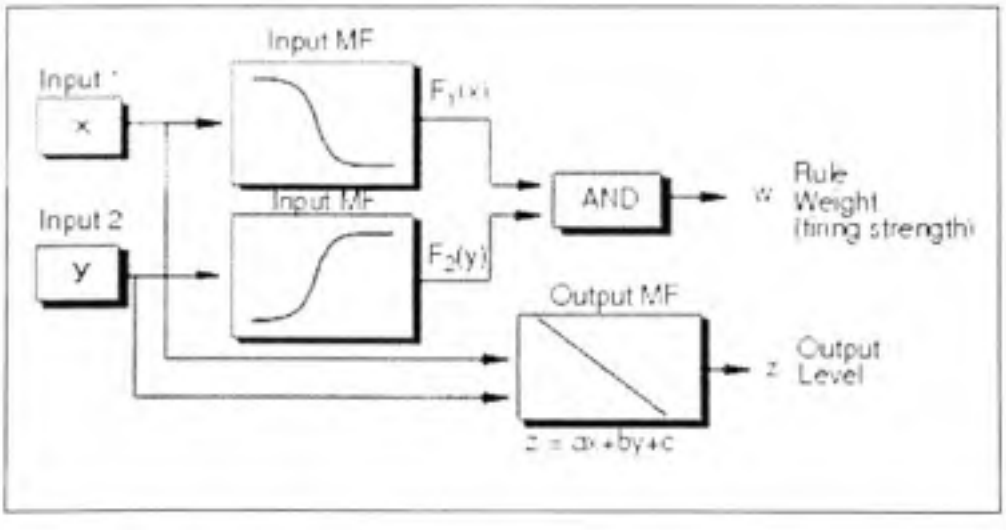

Figure 0.1 **Processus d'évaluation d'une règle d'inférence.**
Tirée de Mathworks (N.d.)

L'antécédent de la règle est utilisé pour déterminer un poids. Ce poids est utilisé pour pondérer la valeur de sortie obtenue par le calcul du conséquent. Par la suite, il s'agit de faire une moyenne de ces résultats pour obtenir la sortie exploitable. Pour plus de détails, nous vous invitons à consulter le livre de Driankov (1996, p. 186).

Le choix de l'ordre de la fonction située dans le conséquent de la règle dépend des exigences de l'utilisateur. Un ordre élevé permettrait un meilleur ajustement, mais la complexité augmente également. De plus, un FIS de type TSK d'ordre zéro est équivalent à un FIS type Mamdani ayant des singletons comme univers de sortie (Babuška et Verbruggen, 1996).

Les FIS sont souvent utilisés pour l'automatisation des procédés. Toutefois, il existe une autre application pour cette technologie. En effet, la modélisation de système est également étudiée (Babuška et Verbruggen, 1996). Dans ces travaux, les chercheurs ont caractérisé le rôle du FIS en tant qu'interpolateur. En effet, le FIS interpole les conditions définies par les règles du moteur d'inférence. Ainsi, il est considéré comme un approximateur de fonction universel (Hao *et al.*, 1999). Selon eux, n'importe quelle fonction peut être approximée au degré de précision voulu en utilisant la technique appropriée. Par ailleurs, cette qualité nous sera très utile dans le chapitre trois pour justifier l'utilisation des FIS.

Pour les besoins de nos travaux, nous devons explorer l'inversion de modèle floue. L'inversion de modèle permet de trouver des solutions à divers problèmes. Par exemple, en contrôle, on l'utilise dans la commande par modèle interne ou « internal model control » (IMC) (Boukezzoula, Galichet et Foulloy, 2003). Plusieurs chercheurs se penchent sur le problème d'inversion exacte d'un FIS de type Mamdani à multiples entrées et simple sortie (Baranyi *et al.*, 1998; Riid et Rüstern, 1998). Toutefois, leur méthode ne s'applique pas directement à un FIS de type TSK. Donc, d'autres efforts ont généré des solutions pour ce type de problème (Boukezzoula, Galichet et Foulloy, 2003; Salman et Anavatti, 2007; Xu et Shin, 2007). Toutefois, les méthodes se limitent aux FIS d'ordre zéro. L'équipe de Boukezzoula (2003) propose cependant une décomposition en sous-systèmes permettant de

résoudre le problème analytiquement. Cette décomposition sera utile dans l'élaboration de notre méthode d'inversion.

L'« Iterative Learning Control » (ILC) est utilisé dans plusieurs domaines (Bristow, Tharayil et Alleyne, 2006, p. 97). Cette approche se classifie dans les méthodes de commande cycle à cycle. Cette commande est adaptée aux procédés répétitifs. Par exemple, le problème classique résolu par l'ILC est le suivi de trajectoire d'un bras robotisé (Moore, 1998, p. 426). En effet, la commande en cours de cycle permettait de minimiser l'erreur sur la trajectoire en cours. Toutefois, aux cycles subséquents, le contrôleur reproduisait toujours cette erreur de suivi. Donc, l'idée est d'utiliser les erreurs du passé pour corriger la trajectoire suivante.

Pour une commande ILC discrète, la solution dépend de la période d'échantillonnage. En effet, si nous fixons cette période égale à la durée d'un cycle de chauffe du four de thermoformage, nous retrouvons le domaine des cycles (Gauthier, 2009, p. 5). De ce fait, les informations en cours de cycle ne sont plus disponibles pour l'évaluation de la commande (Xu *et al.*, 1999). Alors, la trajectoire n'a plus d'importance et le point d'arrivé devient la seule préoccupation. Ce cas particulier de l'ILC se nomme la commande en fin de cycle par apprentissages successifs, ou « Terminal Iterative Learning Control » (TILC). Puisque nous réalisons la mesure de température à la sortie du four, la commande TILC est plus adaptée à notre application.

L'utilisateur de l'ILC ainsi que du TILC accepte cinq hypothèses fondamentales à ces lois de commande appliqué à des systèmes linéaires variant dans le temps (Moore, 1998, p. 428) :

1. La condition initiale du système demeure constante d'un cycle à l'autre;
2. La dynamique peut varier dans le temps, mais elle ne peut pas varier sur la base des cycles. En d'autres termes, la variation ne peut pas être due à l'exécution d'un cycle;
3. La sortie désirée doit être faisable;
4. La solution pour obtenir la sortie désirée doit être unique;
5. La durée du cycle doit être constante.

L'hypothèse la plus contraignante est certainement la stationnarité des conditions initiales. En effet, nous n'avons pas de pouvoir réel sur ce paramètre contrairement au temps de cycle. Dans le cadre de nos travaux, nous considérerons cette variation comme une perturbation (Gauthier, 2009, p. 16).

Dans le cadre de ce projet, nous travaillerons avec un modèle existant présenté au chapitre un. Le modèle de simulation proposé reproduit fidèlement les différents phénomènes d'un four de thermoformage. Par exemple, les variations de température initiale des feuilles de plastique, les variations de la température ambiante et la variation des paramètres physiques de la feuille de plastique sont des éléments considérés. La simulation sera utilisée pour évaluer la stabilité du système et la convergence de la température de la feuille vers la consigne malgré les perturbations. La performance et la robustesse sont aussi évaluées.

Pratiquement, ce projet de recherche permettra de diminuer le taux de rejet des entreprises manufacturières et ainsi diminuer leur coût de production. De plus, le temps de démarrage des productions sera diminué, car le système atteindra plus rapidement la stabilité. Aussi, la conception modulaire de la commande ne sera basée que sur les données. En effet, nous espérons ainsi développer une méthode de conception éliminant le processus de modélisation mathématique pour une application future sur d'autres procédés. L'approche modulaire est souhaitable puisque nous pouvons travailler de manière indépendante sur les modules sans nécessairement affecter directement les autres. Plus concrètement, le premier module est utilisé pour effectuer le découplage des entrées du système et le second module est dédié à l'asservissement. L'objectif de la commande est de réduire l'erreur à moins de 5°C en moins de cinq cycles de chauffage. De plus, les courbes d'erreurs devront présenter des caractéristiques de stabilité. Alors, une convergence monotone est souhaitable.

Ce mémoire est divisé en quatre chapitres. Plus spécifiquement, nous présenterons le procédé à l'étude plus en détails dans le chapitre un. Nous en profiterons pour analyser certaines caractéristiques intéressantes du modèle mathématique utilisé pour les simulations. Le chapitre deux est le cœur de nos travaux. Il est consacré au module de découplage. La principale contribution de nos travaux est liée au processus d'inversion d'un FIS de type TSK d'ordre un. À notre connaissance, aucun auteur n'a publié d'article à cet effet. Sans ce module, la structure proposée dans le cadre de ces travaux ne serait pas réalisable. La formulation de la loi de commande floue de type TILC est décrite dans le chapitre trois. De plus, nous suggérerons deux façons de concevoir ce type de contrôleur. La méthode intuitive s'applique parfaitement au TILC d'ordre un. Pour les ordres supérieurs, nous suggérons l'utilisation d'algorithme génétique, car les combinaisons de conditions sont de plus en plus complexes à établir. Finalement, dans le chapitre quatre, nous appliquerons la structure proposée afin d'asservir le four de thermoformage. Nous étudierons trois configurations différentes.

CHAPITRE 1

PRÉSENTATION DU PROCÉDÉ

Ce chapitre est dédié à la présentation du four de thermoformage. Nous concentrerons nos efforts sur le modèle du procédé développé par (Gauthier, 2009). Cette analyse nous permettra de simplifier le développement de notre contrôleur. Plus exactement, nous détaillerons la configuration du four de thermoformage et nous présenterons rapidement le modèle.

1.1 Configuration du four de thermoformage

Les paramètres des modèles de simulation sont intimement liés à la construction physique de l'appareil de production. Toutefois, les comportements et les phénomènes intervenants dans le procédé demeurent semblables. Le modèle de simulation utilisé dans le cadre de ce projet est basé sur le four de thermoformage présenté à la figure 1.1.

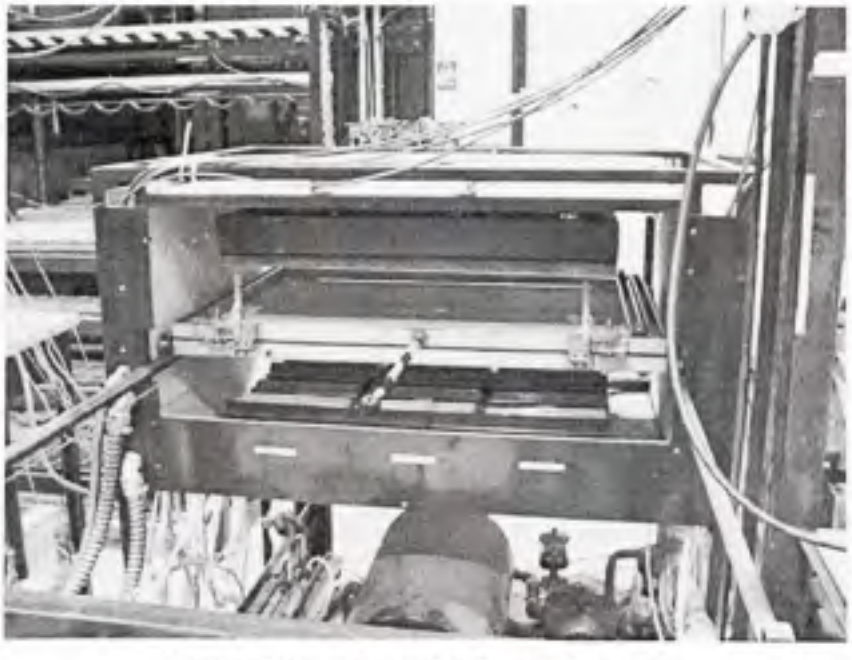

Figure 1.1 Four de thermoformage.
Tirée de Gauthier (2008)

Les éléments en céramique sont regroupés par trois pour former un sous-groupe. Les sous-groupes sont asservis par une boucle de contrôle PID pour maintenir une température égale à la consigne. Pour simplifier la lecture, nous utiliserons « élément chauffant » pour désigner les sous-groupes. La figure 1.2 illustre la disposition des actionneurs dans le four de thermoformage ainsi que la position de la feuille de plastique.

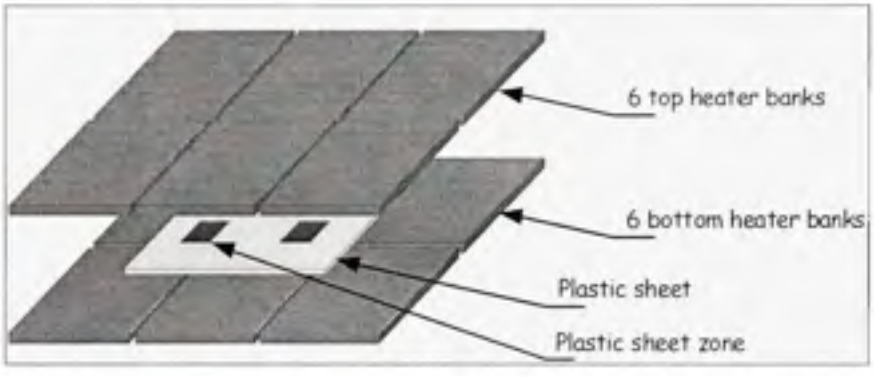

Figure 1.2 Configuration des éléments chauffants.
Tirée de Gauthier (2008)

Présentement, les prises de température sur les zones de la feuille de plastique sont effectuées par des capteurs situées à l'intérieur du four de thermoformage. La disposition ainsi que la numérotation des capteurs au dessus de la feuille sont illustré à la figure 1.3.

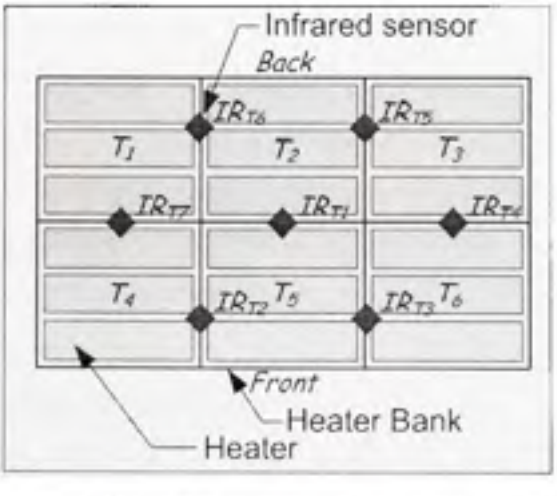

Figure 1.3 Disposition des capteurs de température.
Tirée de Gauthier (2009, p. 32)

En ce qui concerne les capteurs au dessous de la feuille, la disposition et la numérotation sont les mêmes. La figure 1.4 illustre la position des capteurs selon un vue de coupe transversale du four.

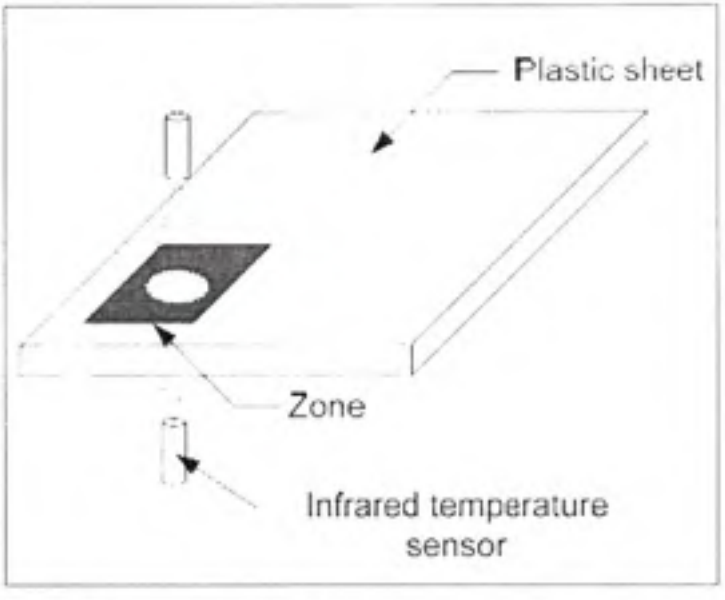

Figure 1.4 Position relative des capteurs du dessus et du dessous.
Tirée de Gauthier (2009, p. 32)

1.2 Modélisation

La première étape est de déterminer le système à l'étude. Précédemment, nous avons défini des zones de lecture pour les capteurs (figure 1.2). Donc, la coupe transversale de ces zones correspond aux systèmes à modéliser. Bien entendu, la température de la feuille de plastique est la grandeur d'intérêt. Toutefois, cette variable est distribuée, alors nous devons discrétiser pour faciliter le processus de modélisation. La figure 1.5 illustre le système discrétisé d'une zone de la feuille de plastique d'épaisseur z (Gauthier, 2009, p. 33). Le nombre N de nœuds dépend essentiellement du degré de précision à atteindre. Cependant, la complexité augmente également. Dans les travaux de Gauthier (2009), ce nombre est fixé à cinq. Ce nombre est un bon compromis entre complexité et précision.

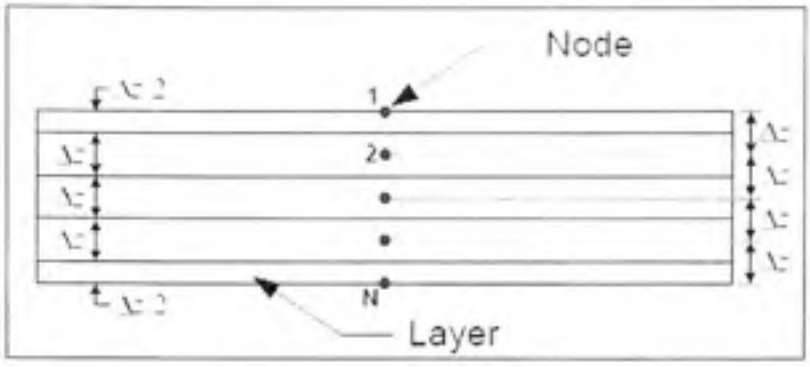

Figure 1.5 Système à modéliser.
Tirée de Gauthier (2009, p. 33)

1.2.1 Hypothèse

La première hypothèse, émise lors de l'élaboration du modèle, concerne la similarité entre les zones. En effet, on considère que les phénomènes sont les mêmes pour toutes les zones à un facteur près. En d'autres termes, d'un point de vue conceptuel, les zones sont identiques. La seconde hypothèse concerne la température ambiante dans le four, elle est considérée constante. La troisième hypothèse concerne les points situés à l'intérieur de la feuille. On considère que la feuille de plastique n'est pas opaque aux radiations. Donc, une partie des ondes sont transmises à l'intérieur de celle-ci. C'est la principale amélioration par rapport au modèle original (Gauthier, 2009, p. 25). Finalement, la déflexion de la feuille de plastique n'est pas considérée par le modèle.

1.2.2 Phénomène considéré dans le modèle

Le chauffage d'une feuille de plastique nécessite l'utilisation de notions de thermodynamique et de physique des ondes pour modéliser le comportement. En effet, les éléments chauffants émettent des radiations contribuant à l'augmentation de température qui est expliquée par la physique des ondes. La contribution de la thermodynamique dans le modèle est liée aux échanges de chaleur. Plus précisément, la conduction ainsi que la convection caractérisent ces échanges. Pour plus de détails, vous trouverez les équations différentielles ainsi que la

définition des variables dans Gauthier (2009). Ces phénomènes sont applicables à toutes les zones. Toutefois, nous savons très bien qu'une zone située au milieu de la feuille de plastique sera plus chaude que celle située sur les bords, car la somme des contributions des différents éléments chauffants est plus grande pour cette zone. En d'autres termes, la quantité d'énergie disponible pour augmenter la température de cette zone est plus grande. Donc, pour introduire cet aspect dans le modèle, le facteur géométrique doit être inclus dans le modèle d'une zone (Holmann, 2002). Ce facteur permet d'ajuster la quantité d'énergie transmise en fonction de la position relative de l'émetteur et du récepteur. Dans notre situation, les corps peuvent être considérés comme des surfaces rectangulaires parallèles puisque l'on ne tient pas compte de l'effet de déflexion de la feuille de plastique. La figure 1.6 illustre la position relative d'une paire d'éléments où z est la distance entre la feuille de plastique A_1 et les éléments chauffants A_2 (Ehlert et Smith, 1993).

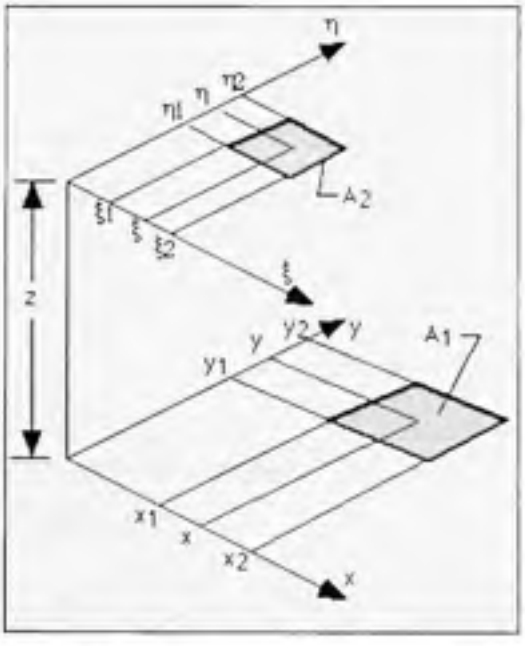

Figure 1.6 Paramètre du facteur géométrique pour deux surfaces parallèles.
Tirée de Ehlert et Smith (1993)

L'équation (1.1) permet de calculer le facteur correspondant à la paire d'éléments identifiée à la figure 1.6. Comme vous le constatez, le calcul est complexe et n'est valide que pour deux surfaces parallèles. La déflexion n'est pas considérée pour ces deux raisons.

$$F_{12} = \frac{1}{(x_2 - x_1)(y_2 - y_1)} \sum_{i=1}^{2}\sum_{k=1}^{2}\sum_{j=1}^{2}\sum_{s=1}^{2} (-1)^{(i+j+k+s)} G(x_i, y_j, \eta_k, \xi_s)$$

$$\text{où} \qquad G = \frac{1}{2\pi}\left\{(y-\eta)\sqrt{(x-\xi)^2 + z^2}\,\tan^{-1}\left(\frac{y-\eta}{\sqrt{(x-\xi)^2 + z^2}}\right)\right\}$$

$$+ \frac{1}{2\pi}\left\{(x-\xi)\sqrt{(y-\eta)^2 + z^2}\,\tan^{-1}\left(\frac{x-\xi}{\sqrt{(y-\eta)^2 + z^2}}\right)\right\}$$

$$- \frac{z^2}{4\pi}\ln\left((x-\xi)^2 + (y-\eta)^2 + z^2\right) \tag{1.1}$$

Pour plus d'informations sur le modèle mathématique, nous vous référons à la page 41 de Gauthier (2009).

1.2.3 Particularité du modèle : Symétrie des zones

En analysant les fonctions différentielles décrivant les zones, nous constatons que la seule différence entre celles-ci est le vecteur exprimant le facteur géométrique, communément appelé « view factor ». Par définition, ce facteur géométrique exprime l'arrangement des éléments chauffants dans le four. Donc, nous pouvons établir la symétrie entre les zones en comparant ces vecteurs.

Pour illustrer ce concept, nous proposons d'étudier le four selon une configuration à quatre entrées et à quatre sorties. Posons,

$$y_1 = IR_{T2} \quad y_2 = IR_{T5}$$
$$y_3 = IR_{B2} \quad y_4 = IR_{B5}$$
$$u_1 = T_1 = T_2 = T_3$$
$$u_2 = T_4 = T_5 = T_6$$
$$u_3 = B_1 = B_2 = B_3$$
$$u_4 = B_4 = B_5 = B_6$$

Par régression, il est possible de déterminer un polynôme (1.2) décrivant le comportement.

$$y_i = a_i \bullet u_1 + b_i \bullet u_2 + c_i \bullet u_3 + d_i \bullet u_4 \quad où\ i \in \{1,2,3,4\} \tag{1.2}$$

Dans les quatre équations, les coefficients seront identiques, mais ordonnés différemment. Nous proposons de faire le changement de variables suivant :

φ_1 := Température des éléments chauffants au dessus de la zone
φ_2 := Température des éléments chauffants en diagonale et au dessus de la zone
φ_3 := Température des éléments chauffants sous la zone
φ_4 := Température des éléments chauffants en diagonale et sous la zone

Pour clarifier le changement de variable, prenons un exemple définissant la position relative des groupes d'éléments chauffant par rapport à la zone lue par le capteur IR_{T5}. Les éléments T_1, T_2 et T_3 forme le regroupement φ_1 alors que B_1, B_2 et B_3 forme le regroupement φ_3. Les regroupements φ_2 et φ_4 sont formés respectivement des éléments T_4, T_5, T_6 et B_4, B_5, B_6. En utilisant une position relative plutôt qu'absolue, nous pouvons réécrire toutes les équations sous la forme suivante :

$$y_i = a \bullet \varphi_1 + b \bullet \varphi_2 + c \bullet \varphi_3 + d \bullet \varphi_4 \tag{1.3}$$

La symétrie nous permet donc d'utiliser le même résultat pour les quatre fonctions d'approximation en permutant simplement les entrées dans un ordre logique. Cette caractéristique nous permettra de réduire le temps nécessaire pour élaborer une stratégie de contrôle. Bien entendu, ce résultat est dû à la disposition physique des éléments dans le four réel. En pratique, la symétrie exacte entre le dessus et le dessous de la feuille n'existe pas. Il ne faut pas oublier que le modèle mathématique n'est qu'une représentation du système réel. En effet, des phénomènes non modélisés expliquent la différence de quelques degrés. Pour cette raison, nous éviterons, dans un premier temps, d'utiliser cette symétrie. Toutefois, nous

exploiterons cette caractéristique pour des systèmes de grandes dimensions en considérant la perte de précision engendrée par ce problème de modélisation.

Le modèle du four de thermoformage est un bon outil pour développer notre contrôleur. Le prochain chapitre est justement dédié à la première étape de conception d'un contrôleur à multiples entrées, soit le module de découplage.

CHAPITRE 2

LES FONCTIONS DE DÉCOUPLAGE

Dans ce chapitre, nous étudierons l'impact du couplage dans un système. Cette étude nous permettra de conclure que le couplage complexifie le développement de loi de commande. Donc, nous présenterons une méthode permettant d'éliminer ou d'atténuer ce comportement indésirable.

2.1 Description du phénomène de couplage

En premier lieu, il est nécessaire de définir le phénomène de couplage entre les variables d'un système multiples entrées. Supposons qu'il existe des fonctions (2.1) décrivant le comportement d'un système à n entrées et m sorties. Par exemple, si ce système est un réacteur chimique, les entrées peuvent être la concentration des réactifs et leurs températures et les sorties, la concentration et la température du produit.

$$\begin{aligned} & y_i = f_i(x_1, x_2, \cdots, x_n) \quad i \in \{1, 2, \cdots, m\} \\ & où \quad f_i : \mathbb{R}^n \to \mathbb{R} \end{aligned} \tag{2.1}$$

Pour simplifier la présentation, supposons maintenant que les fonctions sont linéaires. En appliquant les propriétés des applications linéaires, nous pouvons réécrire l'équation (2.1) sous la forme suivante :

$$\begin{aligned} & y_i = f_{i,1}(x_1) + f_{i,2}(x_2) + \cdots + f_{i,n}(x_n) \quad i \in \{1, 2, \cdots, m\} \\ & où \quad f_{i,j}(\bullet) \text{ est une fonction linéaire}, \quad i \in \{1, 2, \cdots, m\} \\ & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad j \in \{1, 2, \cdots, n\} \end{aligned} \tag{2.2}$$

La figure suivante illustre cette dernière équation sous la forme d'un schéma bloc.

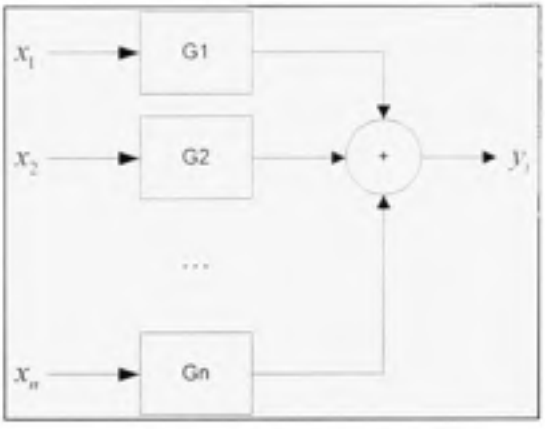

Figure 2.1 Schéma bloc décrivant un système.

Il devient évident que chacune des variables d'entrées apporte une contribution à la grandeur de sortie. Cette interaction entrées-sorties définit le couplage. En effet, l'indice de couplage est tout simplement la contribution d'une variable divisée par la contribution totale. Un couplage fort signifie qu'il n'y a pas de variable d'entrée dominante expliquant le comportement d'un système.

En contrôle, un couplage fort n'est pas souhaitable, car nous devons dans cette situation composer avec plusieurs entrées pour contrôler une sortie. La figure suivante illustre le problème de contrôle en rétroaction d'un système à deux entrées et à deux sorties (Gauthier, 2006).

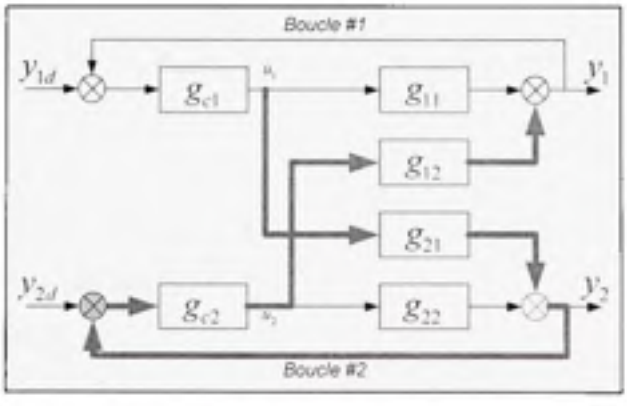

Figure 2.2 Schéma bloc illustrant l'interaction des entrées sur les sorties.

Le groupe g_{11} à g_{22} représente la dynamique du système. On remarque clairement l'effet de la commande sur les deux sorties. Donc, la conception de g_{c1} et g_{c2} (contrôleurs) n'est pas un problème trivial. En effet, les gains de ces contrôleurs doivent être déterminés simultanément.

Pour pallier à ce problème, il existe deux alternatives. La première est de modifier le procédé. Pour illustrer cette possibilité, prenons un réservoir d'eau ayant une entrée d'eau chaude et une entrée d'eau froide. Nous désirons contrôler avec ces deux grandeurs la température et le niveau dans le réservoir. Puisque les deux entrées ont un impact sur les deux sorties, nous pouvons nous attendre à un couplage important. Pour arriver au même objectif (contrôle du niveau et de la température dans le réservoir), nous pouvons modifier le procédé. En effet, nous pourrions ajouter un élément chauffant et retirer l'entrée d'eau chaude. Le système devient alors découplé. C'est-à-dire que le niveau n'est influencé que par le volume d'eau entrant dans le réservoir et la température, que par l'énergie transmise par l'élément chauffant.

Toutefois, il n'est pas toujours possible de modifier le procédé. La seconde alternative est donc de compenser les commandes pour éliminer synthétiquement les interactions. Dans le cadre de ce projet, nous avons choisi cette alternative puisqu'il n'est pas possible de modifier le procédé.

2.2 Méthodes existantes

Le contrôle de système multi variable impose aux développeurs le phénomène de couplage. Pour pallier ce phénomène, plusieurs travaux touchant les systèmes linéaires et invariants dans le temps (LTI) ont été effectués (Wang, 2003, p. 3). La figure 2.3 représente la structure proposée par (Wade, 2004).

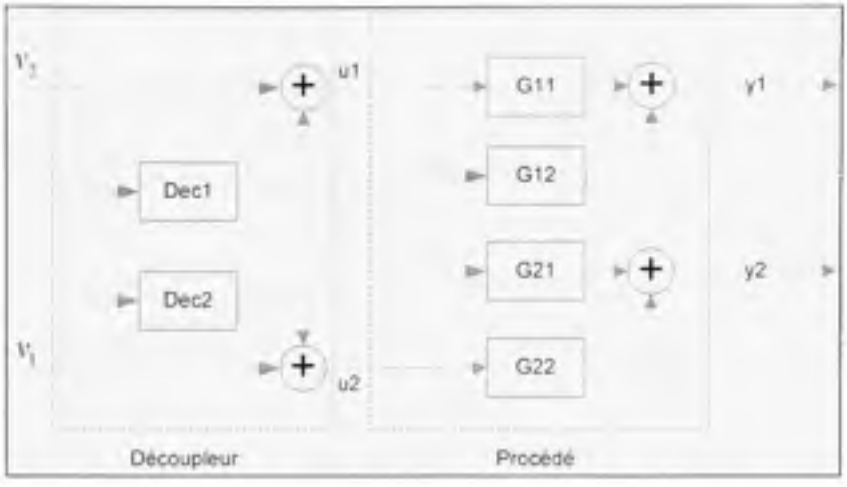

Figure 2.3 Structure de fonctions de découplage prédictif.

Cette figure présente un module de découplage prédictif puisque le calcul des commandes est effectué avec des informations connues à priori. La modularité de cette structure est sa qualité principale. Pour déterminer les fonctions de découplages Dec1 et Dec2, nous devons avoir un modèle mathématique LTI représenté par les fonctions *Gij*. Ce dernier aspect constitue en fait la contrainte la plus sévère puisque, la plupart du temps, les systèmes ne sont pas LTI. La linéarisation diminue le domaine de validité du modèle et limite ainsi la zone d'opération du système. Alors, les performances peuvent être grandement affectées à l'extérieur de cette zone.

Puisque le développement d'un modèle est un processus long et coûteux, d'autres auteurs privilégient l'utilisation de la logique floue. Par exemple, Mingyu (2008) propose une structure permettant d'intégrer les connaissances d'un expert. Contrairement aux méthodes analytiques, cette dernière proposition ne nécessite pas de modèle mathématique. Ainsi, la période de développement d'une solution est réduite considérablement. Toutefois, les interactions peuvent devenir très difficiles et longues à définir lorsque le nombre d'entrée est élevé. Pour remédier à ce problème, nous proposons une variantes inspirée par les méthodes de Wade (2004) et Mingyu (2008).

2.3 Méthode proposée

Théoriquement, il est possible d'éliminer l'interaction indésirable entre une entrée et une sortie lorsque le système peut être inversé. Pour atteindre cet objectif, nous tenterons de compenser l'effet de la commande appliquée à une entrée sur les autres entrées. En d'autres termes, nous ajoutons une couche d'interprétation entre le contrôleur et le système. Cette couche permet de simplifier le problème de contrôle puisque toutes les sorties sont expliquées par une seule entrée. Donc, toutes les théories applicables au système SISO peuvent être utilisées. Dans cette section, nous présenterons la méthode développée dans le cadre de ce projet.

2.3.1 Description générale

La fonction de découplage a pour but d'éliminer ou d'atténuer les interactions indésirables. Toutefois, il existe une infinité de fonctions permettant d'atteindre cet objectif. Cependant, nous pouvons être plus restrictifs sur l'objectif de la fonction de découplage. En effet, une fonction défaisant ce que fait le système éliminera également les interactions. Donc, nous utiliserons la fonction réciproque. Le découplage est simplement un problème de transformation entre deux espaces donnés. Le problème est bien connu en algèbre linéaire, mais beaucoup plus complexe pour des applications non-linéaires.

De plus, nous avons conclu que les méthodes existantes ont des lacunes. Malgré tout, elles offrent des caractéristiques complémentaires. Pour cette raison, nous proposons une méthode alternative utilisant la structure du module de découplage prédictif et le système d'inférence flou (FIS). La figure 2.4 présente la structure utilisée pour l'élaboration de cette méthode.

Le modèle du système est souvent inconnu au début du processus de conception d'un contrôleur. Pour cette raison, nous considérons que le modèle du système est de type « boîte noire ». De plus, la fonction de découplage est dorénavant modélisée par un système d'inférence flou comportant n entrées et n sorties. Pour simplifier la conception du module de

découplage. Mingyu (2008) propose une décomposition du FIS en élément singulier. La figure 2.5 présente la décomposition proposée.

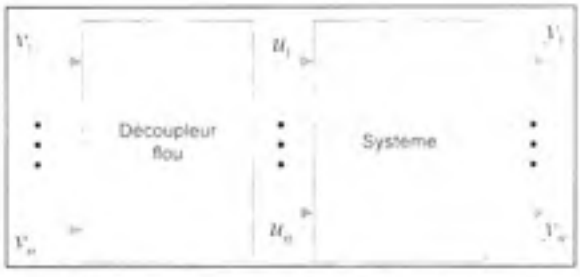

Figure 2.4 Structure de la fonction de découplage proposée.

Cette décomposition augmente la modularité du module de découplage. Cette caractéristique nous permettra de faire des simplifications dépendamment du système.

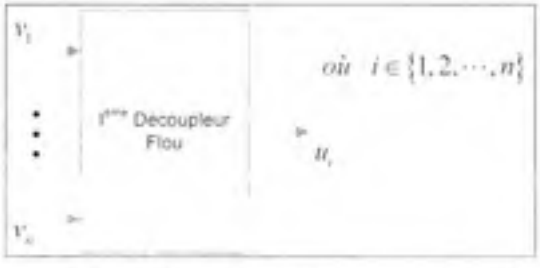

Figure 2.5 Structure décomposée utilisée.

Il y a deux alternatives aux problèmes de complexité et de connaissance limitée du système. La première alternative est d'effectuer un apprentissage en utilisant la base de données inversée. En d'autres termes, utiliser toutes les sorties associées à la $i^{ème}$ entrée pour déterminer la $i^{ème}$ fonction de découplage. Il existe des algorithmes permettant cet apprentissage (Babuška et Verbruggen, 1996). Puisque le comportement n'est pas connu à priori, il faut effectuer de nombreux tests pour couvrir adéquatement l'espace de sortie. Ces tests représentent des coûts importants, car ils doivent être réalisés sur le procédé réel. Donc, cette méthode devient longue, risquée et coûteuse. Driankov (1998, p. 103) propose une seconde alternative. En effet, il propose d'inverser le modèle direct pour obtenir le modèle inverse. Contrairement à la première alternative, la couverture adéquate du domaine du

modèle ne dépend pas du système. Donc, le nombre de tests est limité selon la précision que le concepteur désire. Alors, la méthode proposée se résume en deux grandes étapes :

1. Modélisation directe du système ;
2. Inversion du modèle direct.

Ces étapes seront décrites plus en détails dans les sections subséquentes.

2.3.2 Modélisation directe

La modélisation directe permet de caractériser le comportement du système. Ce modèle est obtenu de différentes façons. Classiquement, les lois physiques, mécaniques, chimiques et autres ont permis de développer de nombreux modèles. Toutefois, ce travail est simplifié par l'utilisation de systèmes d'inférence floue. En connaissant empiriquement le système, il est possible de déduire le modèle. En logique flou, il existe deux grands courants concernant l'apprentissage (Verbruggen et Babuška, 1999, p. 20). Le premier est l'apprentissage par un expert du domaine et le second est l'apprentissage par les données. Chacun comporte des avantages. Notamment, l'utilisation des connaissances d'un expert facilite le développement. Toutefois, lorsque la quantité d'information est trop importante, l'utilisation des données devient avantageuse. En effet, l'humain n'est pas capable de synthétiser une quantité importante d'information simultanément. Dans le cadre de ce projet, nous voulons déterminer des fonctions de découplages à plus de quatre entrées. Donc, la méthode d'apprentissage par les données est plus appropriée.

Essentiellement, le modèle a pour but de générer une courbe ou une surface (ou une hyper surface) conforme au comportement du système. En utilisant les données, nous générons en partie cette caractéristique de sortie. C'est d'ailleurs ce qu'exploite (Gauthier, 1997). La technique proposée s'approche d'une modélisation par élément fini. Il s'agit simplement de diviser l'espace en plusieurs sous-espaces de formes définies et de déterminer une fonction

linéaire décrivant le comportement moyen de la caractéristique. La figure suivante présente un résultat possible pour cette technique.

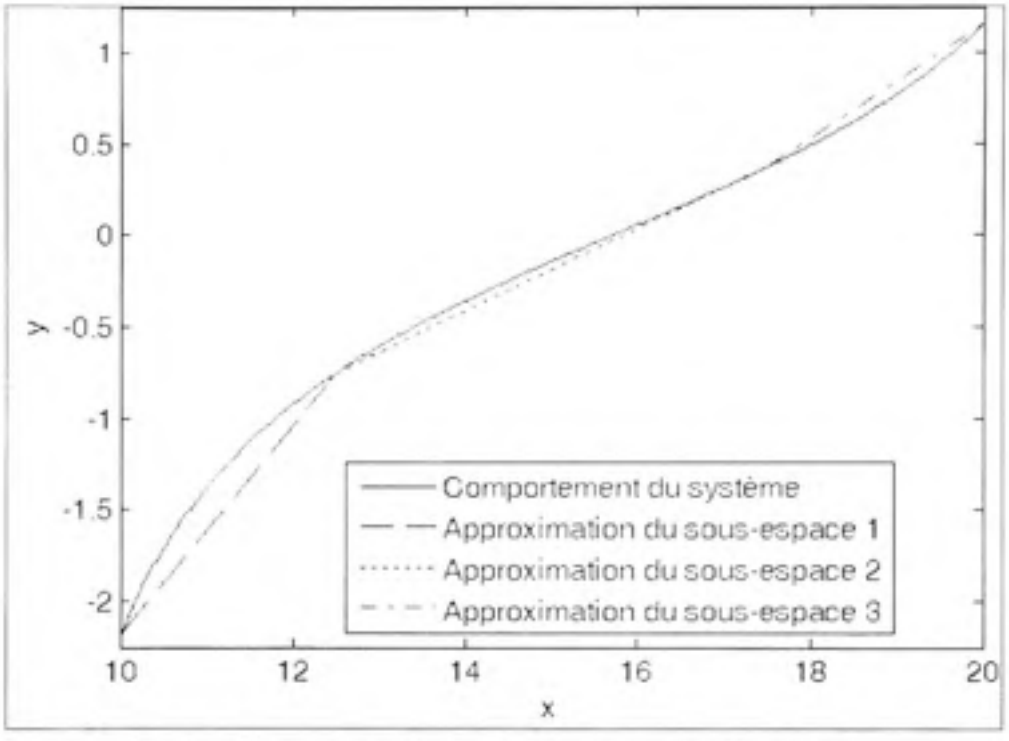

Figure 2.6 Exemple de modélisation par élément fini.

Une approximation d'ordre plus élevé permettrait de diminuer l'erreur d'estimation. Toutefois, l'utilisation de droite affine est une solution simple et nous pouvons diviser l'espace en plus petits sous-espaces pour améliorer la précision. De plus, le choix des courbes d'approximation est important. En effet, il est avantageux de définir les paramètres qui minimisent l'erreur d'approximation.

Pour diminuer cette erreur, nous suggérons de lisser la courbe pour exploiter l'information partagée entre deux sous-espaces. Un bon moyen pour atteindre cet objectif est d'utiliser un système d'inférence flou. La section suivante présente cet outil en mettant l'emphase sur son rôle d'interpolateur.

Système d'inférence flou

Nous avons décidé d'utiliser le type TSK, car ce type est considéré comme un interpolateur entre différentes conditions d'opérations (Babuška, Jager et Verbruggen, 1994). De plus, la continuité de la surface de commande est garantie lorsque l'on utilise le TSK et il est mieux adapté pour un apprentissage par les données. Par ailleurs, ce système d'inférence flou est beaucoup moins lourd que le Mamdani. Précédemment, nous avons choisi d'utiliser une fonction affine. Alors, l'ordre de la fonction situé dans le conséquent de la règle est un.

Pour illustrer le comportement d'interpolation de ce type de FIS, nous avons repris l'exemple de la figure 2.6. La figure 2.7 présente le résultat obtenu suite au traitement flou.

Pour obtenir ce résultat, nous devions définir les fonctions d'appartenance du système d'inférence. Puisqu'à priori, nous avons une connaissance limitée du système, nous avons choisi d'utiliser des fonctions d'appartenance triangulaire pour couvrir l'univers de discours. Pour ce cas précis, les intervalles ont été défini tels que présentés à la figure 2.8.

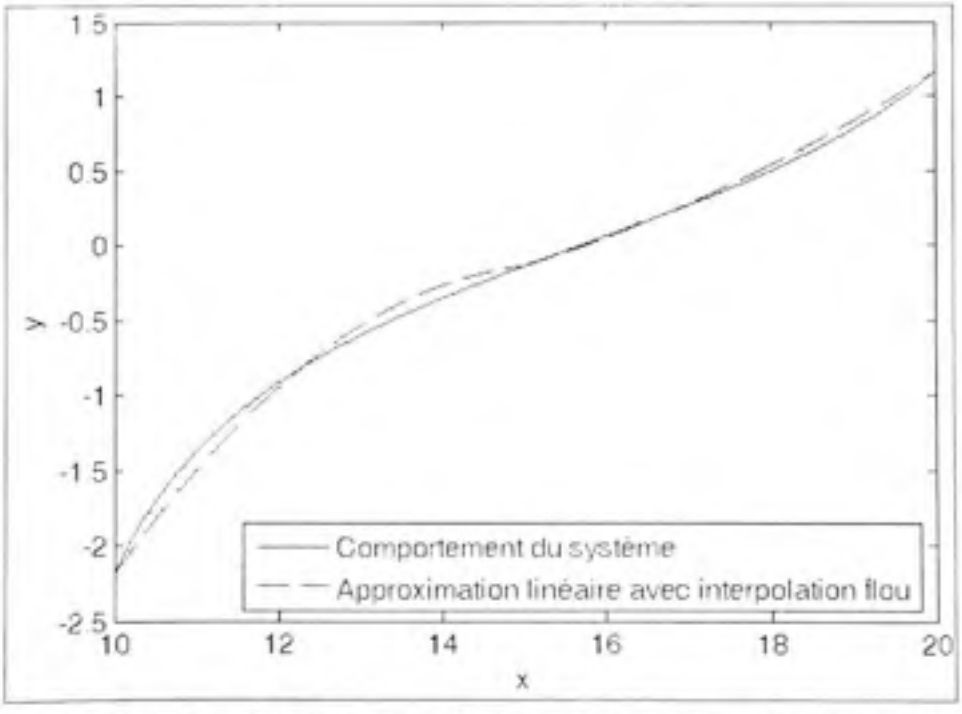

Figure 2.7 Exemple de lissage effectué par le système d'inférence flou.

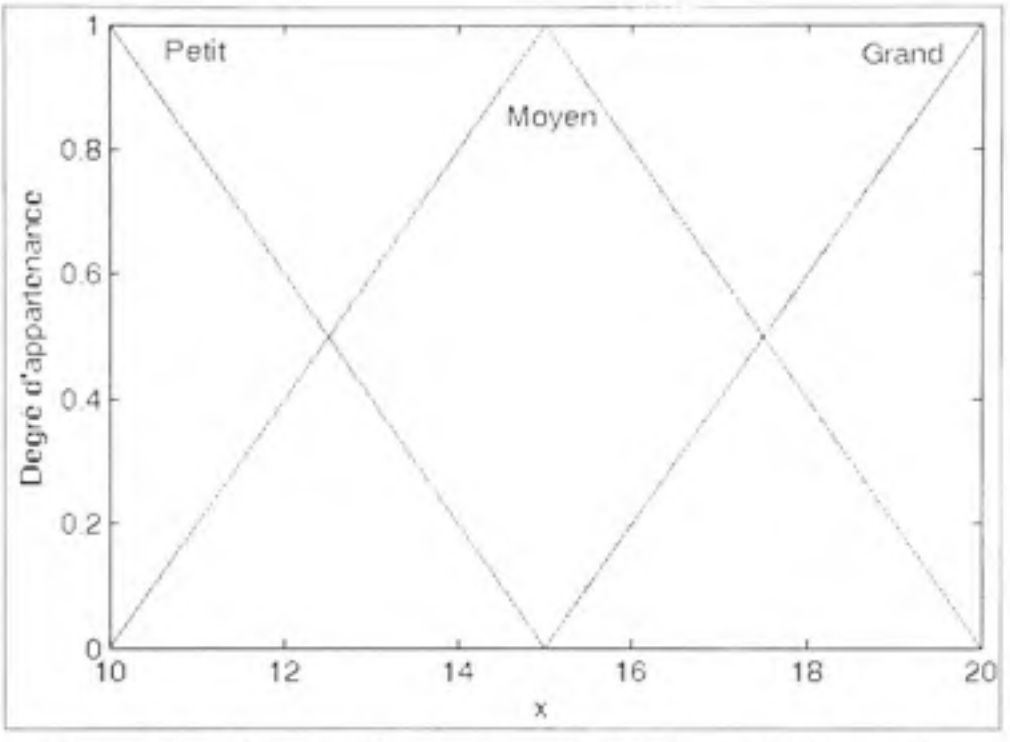

Figure 2.8 Fonction d'appartenance du système d'inférence flou.

Le nombre de fonction d'appartenance est directement lié au nombre de facettes approximant le comportement du système. Dans notre cas, nous avons choisi de diviser chacune des dimensions en trois facettes. Donc, il y a trois fonctions d'appartenance par entrée. La complexité du système d'inférence est directement liée à ce nombre. En effet, l'équation suivante permet de déterminer le nombre maximum de règles n.

$$n = nb.\ fonction\ d'appartenance^{nb.\ d'entrées} \tag{2.3}$$

De plus, la quantité de données nécessaire à l'apprentissage est directement liée à la quantité de paramètres à déterminer. Le choix des fonctions situées dans le conséquent est également critique au bon fonctionnement de l'interpolateur. En effet, (Babuška, Jager et Verbruggen, 1994) démontrent que le TSK est un mauvais interpolateur de fonction locale. Pour illustrer ce comportement, nous proposons d'utiliser les fonctions linéaires caractérisant parfaitement le comportement pour un degré d'appartenance de 100%. Ces fonctions ainsi que le comportement de l'application sont présenté à la figure 2.9.

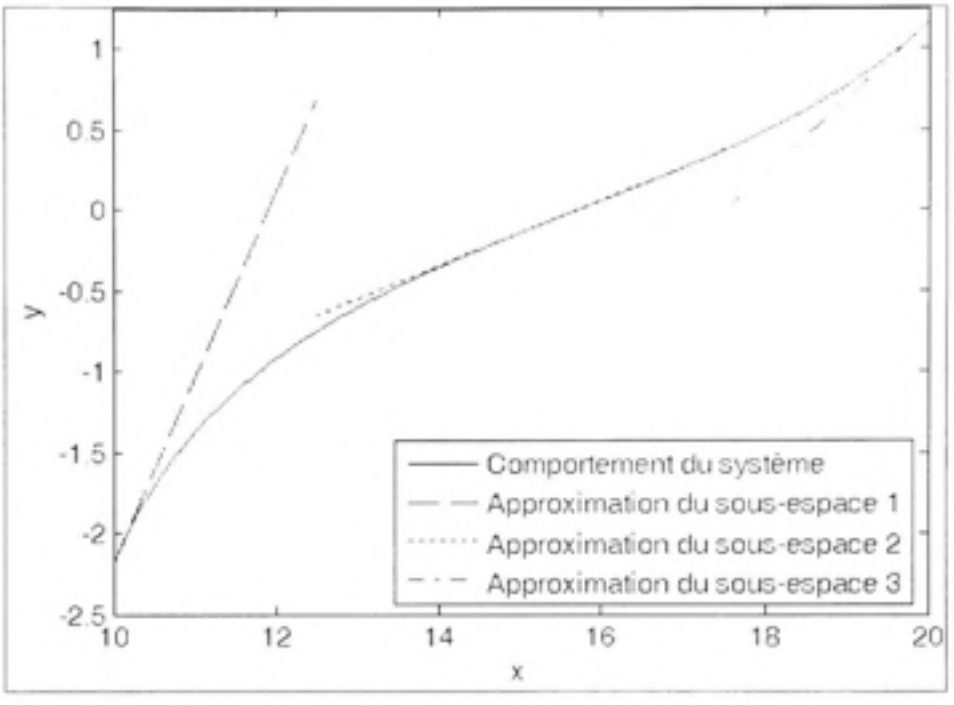

Figure 2.9 Droite d'approximation locale.

La figure 2.10 illustre la sortie du système floue utilisant les fonctions de la figure 2.9.

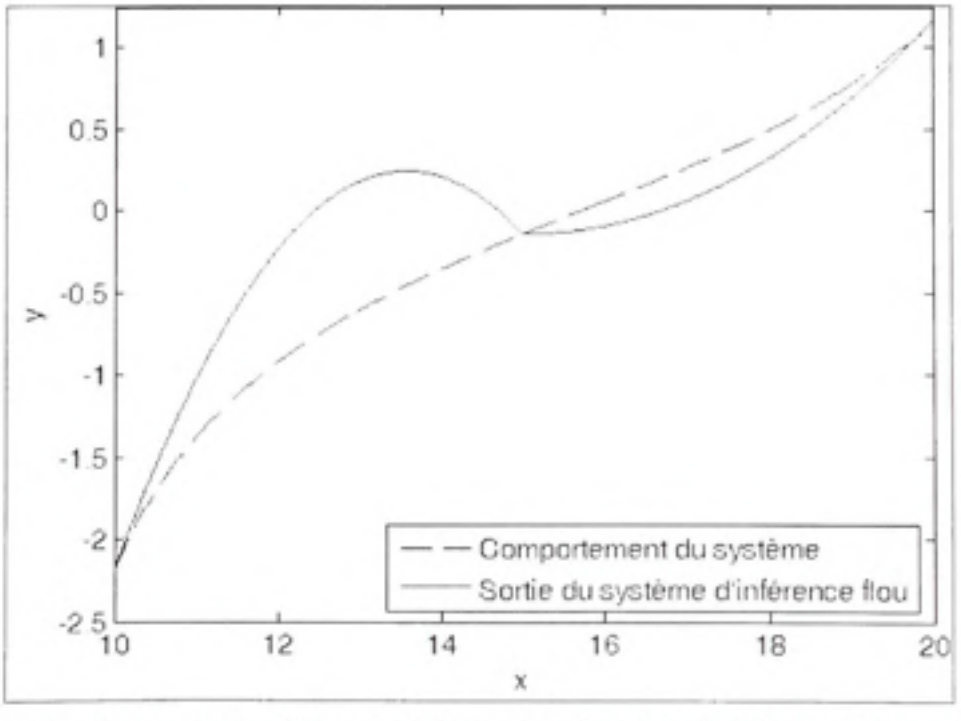

Figure 2.10 Interpolation floue avec des approximations locales du comportement.

C'est pour cette raison que les auteurs suggèrent d'utiliser la droite moyenne pour définir ces fonctions comme l'illustre la figure 2.7. Vous trouverez à l'annexe I le script pour définir un FIS à deux entrées et à une sortie dans le logiciel « Matlab ». Dans la prochaine section, nous discuterons de la méthode pour obtenir ces fonctions. Nous présenterons un outil flexible pour réaliser une régression linéaire.

Krigeage Dual

Le krigeage est une méthode optimale d'interpolation spatiale utilisée dans plusieurs domaines tels qu'en géostatistique, l'océanologie et la météorologie (Gratton, 2002). Cette méthode introduit l'aspect de dépendance spatiale entre les données. En effet, la covariance des données ne doit dépendre que de la distance euclidienne entres les points. Cette première hypothèse est le fondement de la méthode. Elle est également la principale distinction par rapport à la régression linéaire. Le krigeage dual est la forme explicite du krigeage universel. Le passage entre ces deux formulations ainsi que les détails théoriques sont présentés à l'annexe de l'article de Trochu (1993). Nous avons choisi cet outil, car il est plus souple que la régression multi-variable. De plus, l'effort de calcul est réduit par sa formulation sous la forme matricielle.

Dans le cadre de ce projet, nous nous intéressons seulement à la portion de dérive qui définit la tendance moyenne de la caractéristique. Par ailleurs, nous avons défini antérieurement que nous devions déterminer le polynôme minimisant l'écart entre la valeur réelle et la valeur estimée. Ce polynôme doit être d'ordre un. Donc, nous concentrerons nos efforts sur un cas limite du krigeage dual. Jusqu'à maintenant, la variabilité sur les données n'était pas considérée malgré que l'on suppose les données normales. L' « effet pépite » introduit ce concept. L'objectif est d'ajouter l'aspect d'erreur de mesure à l'analyse (Trochu, 1993). Ainsi, nous retrouvons la définition de la régression si nous considérons que la covariance est constante ou égale à zéro. Ce cas limite du krigeage dual est représenté par :

$$\begin{bmatrix} P & \cdots & 0 & 1 & x_{1,1} & \cdots & x_{i,1} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & P & 1 & x_{1,n} & \cdots & x_{i,n} \\ 1 & \cdots & 1 & 0 & 0 & \cdots & 0 \\ x_{1,1} & \cdots & x_{1,n} & 0 & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{i,1} & \cdots & x_{i,n} & 0 & 0 & \cdots & 0 \end{bmatrix} \bullet \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_n \\ \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_{i-1} \end{bmatrix} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{2.4}$$

où $\{x_{1,n}, \cdots, x_{i,n}\}$:= $n^{ième}$ ensemble des consignes de température

y_n := Température de la feuille d'une zone du $n^{ième}$ ensemble

α_i := Coefficient du polynôme de régression

β_n := Coefficient d'ajustement $n^{ième}$ ensemble

P := Coefficient proportionel à la variance des données

Les couples représentés par $(x_{1,n}\ x_{2,n} \cdots x_{i,n}\ y_n)$ constituent les coordonnées des points utilisés pour déterminer les paramètres α_i du polynôme de régression. Les coefficients β_i représentent quant à eux le produit de l'écart des points à la courbe par le coefficient P. Ce paramètre ne fait qu'une mise à l'échelle des coefficients β_i et n'affecte pas les paramètres α_i. Toutefois, si le paramètre P variait selon les coordonnées, les paramètres α_i seraient affectés. Donc, nous pourrions jouer avec ces valeurs pour modifier l'application. Par exemple, supposons que la valeur de P soit plus grande pour un point donnée, la courbe passera plus près de ce point que pour une valeur de P inférieure. Dans le cadre de ce projet, nous considérons que l'erreur de mesure est constante puisque nous utiliserons une simulation pour obtenir les coordonnées nécessaires. Donc, le paramètre P sera constant. Nous le poserons égal à un pour simplifier l'implantation. Ainsi, le polynôme obtenu par krigeage (2.5) correspond exactement à celui obtenu par régression ayant comme critère de minimisation l'erreur quadratique.

$$\hat{y} = \alpha_0 + \alpha_1 \bullet x_1 + \cdots + \alpha_i \bullet x_i \tag{2.5}$$

Ce polynôme définira le conséquent de la règle flou correspondant au sous-espace. Vous trouverez à l'annexe II le script « Matlab » mettant en œuvre le krigeage présenté ci-haut.

Maintenant, nous devons identifier les points $(x_{1,n}\ x_{2,n}\ \cdots\ x_{i,n}\ \ y_n)$. Lors de la présentation du système d'inférence flou, nous avons montré que la fonction affine devait correspondre au comportement moyen de la caractéristique. Pour ce faire, nous devons inclure les données sur un sous-espace suffisamment grand. La question suivante se pose : Comment définir les limites de ce sous-espace ? Pour répondre, nous devons nous rappeler le rôle d'interpolation du système d'inférence flou. En effet, la sortie du FIS est une composition linéaire des fonctions locales. De plus, nous savons que le degré d'appartenance maximum est égal à un selon la définition des univers de discours. Donc, lorsque le degré d'appartenance à une fonction est supérieur à 50%, c'est la fonction appartenant à ce sous-espace qui domine. Alors, il est raisonnable de dire que la fonction affine représente bien cette zone. Donc, nous devons inclure les points correspondant à un degré d'appartenance de 50% dans la liste des points à utiliser.

Jusqu'à maintenant, nous avons obtenu un modèle du système. Toutefois, ce dernier n'est pas exploitable directement pour réaliser le découplage des entrées. En effet, il faut maintenant inverser le modèle.

2.3.3 Inversion de modèle direct

Le modèle direct flou obtenu doit maintenant être inversé pour déterminer la fonction de découplage. Idéalement, nous voulons identifier une méthode analytique exacte. En effet, ce type de méthode ne dégraderait pas nos résultats et serait plus efficace qu'un apprentissage. Toutefois, l'inversion de système d'inférence flou TSK d'ordre un n'est pas documenté. En fait, il existe des méthodes d'inversion de modèle linguistique (Baranyi *et al.*, 1998) et de modèle de type TSK ayant un conséquent constant (Boukezzoula, Galichet et Foulloy, 2003; Galichet, Boukezzoula et Foulloy, 2004; Xu et Shin, 2007). Donc, nous proposons une

méthode approximative s'inspirant à la fois des deux catégories existantes. Voyons tous d'abord le principe général des deux classes de méthodes.

L'inversion de modèle linguistique, comportant autant d'entrées que de sorties, est un processus simple. En effet, il s'agit simplement de permuter les univers de discours des entrées avec les sorties et d'adapter les règles d'inférence. Puisque le type TSK n'utilise pas de fonction d'appartenance en sortie, alors ce processus n'est pas applicable.

La seconde classe de méthode applicable aux modèles flous de type TSK d'ordre zéro propose une décomposition. En effet, le domaine des fonctions du conséquent du moteur d'inférence est lié directement au degré d'appartenance des variables en cause. En d'autres termes, les règles ont une influence limitée à un sous-espace sur le résultat du modèle flou. La superposition de ces sous-espaces est définie par l'univers de discours d'une règle. Donc, un intervalle donné de la sortie n'est caractérisé que par un sous-ensemble de règles connu à priori. Alors, il s'agit d'inverser ces sous-ensembles pour obtenir l'inversion complète du modèle. La principale lacune de cette méthode réside dans l'ordre résultant de la décomposition. En effet, un modèle TSK d'ordre zéro ayant des fonctions d'appartenance triangulaire génère un système d'équation d'ordre un. Ce système est facilement inversible et le résultat de l'inversion demeure d'ordre un. Ainsi, nous pouvons implanter ce résultat dans un moteur d'inférence de type TSK d'ordre un. Cependant, ce processus n'est pas applicable à un modèle TSK d'ordre supérieur, car le résultat de l'inversion ne peut pas être implanté à nouveau. Un exemple présenté dans les pages qui suivent illustre ce problème. Malgré tout, le concept d'inversion par sous-ensemble de règles demeure applicable au problème d'ordre un et nous l'exploiterons différemment. Maintenant, nous pouvons définir plus exactement le processus d'inversion proposé.

Afin de mieux comprendre le problème, déterminons le modèle inverse analytiquement. Illustrons ce processus par un exemple simple synthétique à une entrée et à une sortie comportant deux règles d'inférence. Ce modèle flou peut-être interprété comme étant un sous-système d'un modèle plus grand. C'est d'ailleurs cette propriété de décomposition que

les techniques d'inversion de modèle de type TSK existantes utilisent. L'univers de discours de l'entrée est décrit par la figure 2.11.

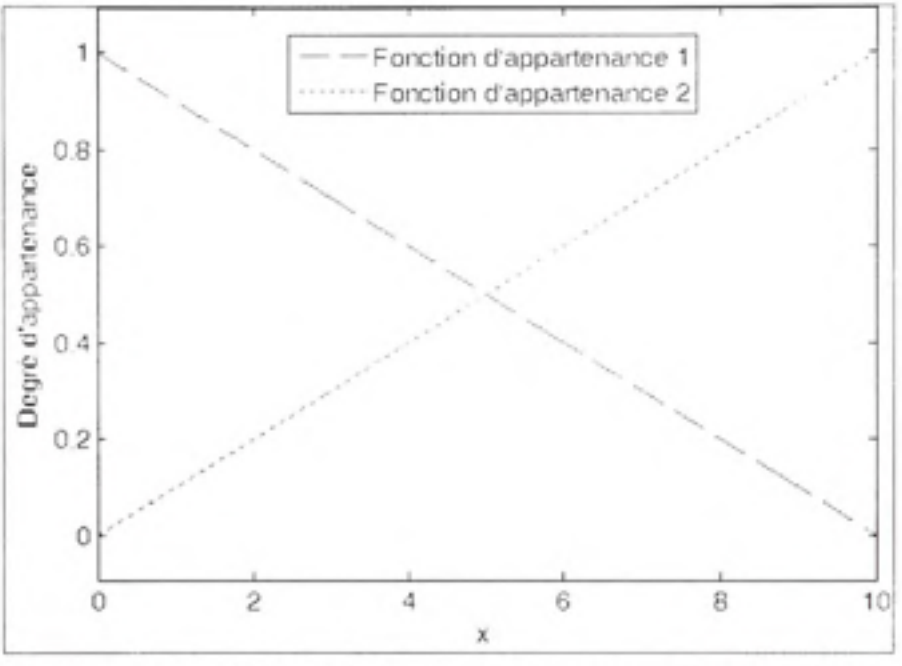

Figure 2.11 Définition de l'univers de discours du modèle direct.

Nous pouvons représenter mathématiquement les deux courbes de la figure 2.11 par les relations linéaires suivantes :

$$f_1(x) = -\frac{1}{10}x + 1$$
$$f_2(x) = \frac{1}{10}x \tag{2.6}$$

Les équations suivantes définissent le conséquent des deux règles d'inférences présenté respectivement dans l'ordre :

$$r_1(x) = 2x + 10$$
$$r_2(x) = x + 15 \tag{2.7}$$

On suppose que ces deux équations définissent le comportement moyen de l'application sur l'intervalle définit par leur fonction d'appartenance respective.

À partir de ces deux ensembles d'équations précédentes, nous pouvons déterminer l'équation de la sortie du système d'inférence flou suivante :

$$y = f_1(x) \bullet r_1(x) + f_2(x) \bullet r_2(x) \tag{2.8}$$

La figure 2.12 présente la valeur de sortie, la contribution un $\left(f_1(x) \bullet r_1(x)\right)$ et la contribution deux $\left(f_2(x) \bullet r_2(x)\right)$ du système d'inférence flou.

Comme vous le remarquez, l'ordre du système d'inférence flou dépend à la fois de l'ordre de la fonction d'appartenance et de celle du conséquent de la règle. Dans le cas présent, l'ordre est de deux. Nous pouvons obtenir l'inverse de cette fonction analytiquement. Toutefois, il existe deux solutions, mais une seule d'entre elle correspond au domaine de la fonction directe. Nous conserverons cette dernière pour poursuivre l'analyse. La figure 2.13 présente le résultat de l'inversion suivie d'une restriction sur le domaine.

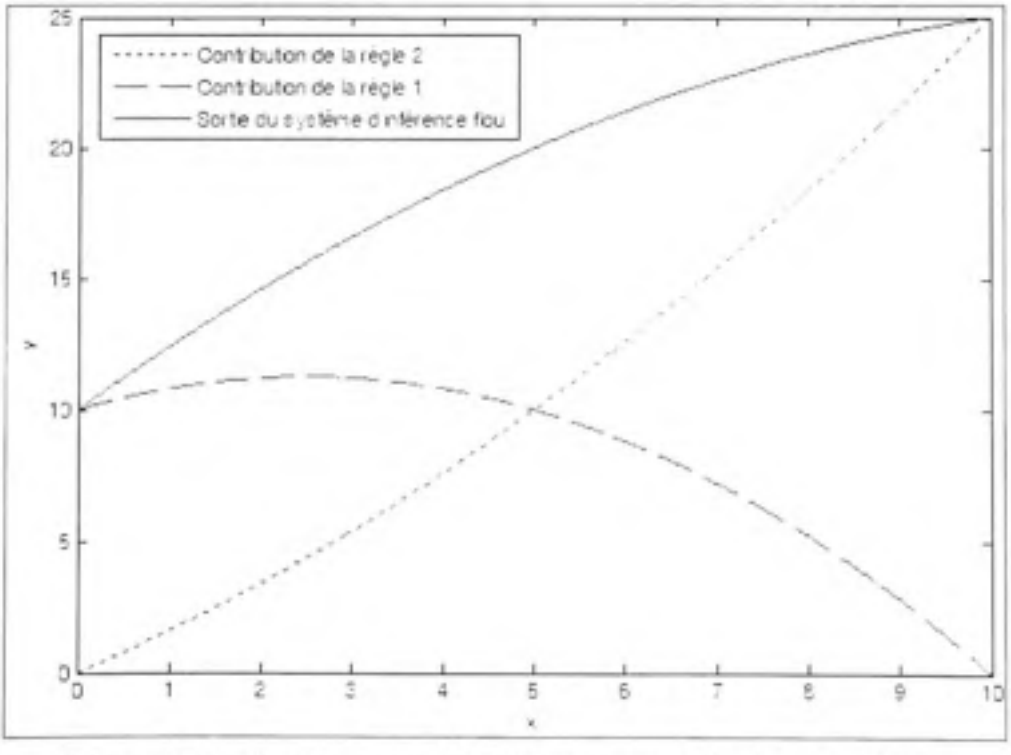

Figure 2.12 Sortie du modèle directe et contribution des règles.

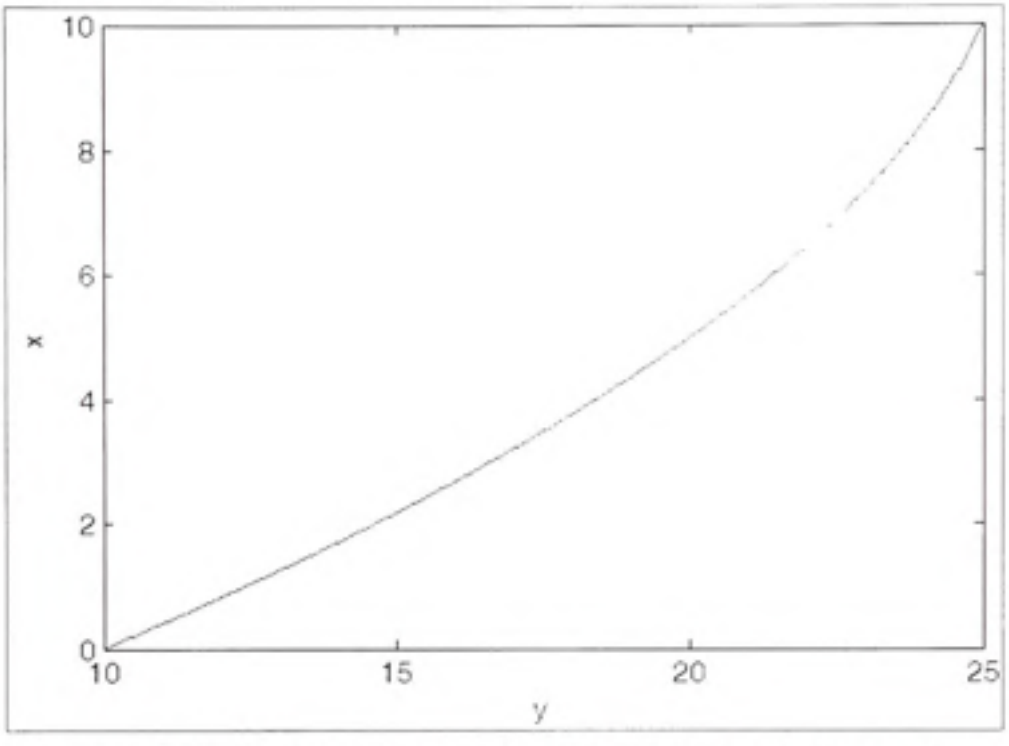

Figure 2.13 Fonction réciproque exacte obtenue analytiquement.

La méthode proposée dans le cadre de ce projet se décompose en deux étapes :

1. Inverser le conséquent de chacune des règles;
2. Définir les univers de discours des entrées.

Poursuivons l'exemple précédent en appliquant ces étapes. Inversons d'abord les fonctions du conséquent des règles (2.6).

$$r'_1(y) = (y-10)/2$$
$$r'_2(y) = y-15 \qquad (2.9)$$

Nous désirons utiliser un système d'inférence flou de type TSK d'ordre un et nous venons de définir le conséquent des deux règles du modèle inverse. Maintenant, nous devons définir les fonctions d'appartenance de l'entrée afin que le modèle flou soit égal à l'inversion analytique. L'équation suivante permet de calculer la sortie du modèle inverse :

$$x = f'_1(y) \bullet r'_1(y) + f'_2(y) \bullet r'_2(y) \tag{2.10}$$

où $f'_i(x) := k^{x/inc}$ fonction d'appartenance du modèle inverse

$r'_i(x) := k^{x/inc}$ fonction définissant le conséquent du modèle inverse

Toutefois, nous ne connaissons pas les fonctions d'appartenance du modèle indirect. Pour résoudre le problème, il faut éliminer un degré de liberté en imposant une contrainte.

$$f'_1(y) + f'_2(y) = 1 \tag{2.11}$$

Donc,

$$x = f'_1(y) \bullet r'_1(y) + \left(1 - f'_1(y)\right) \bullet r'_2(y) \tag{2.12}$$

En utilisant le résultat analytique déterminé plus tôt et en résolvant l'équation précédente, nous pouvons déterminer la fonction d'appartenance exacte. La figure 2.14 présente ces deux fonctions permettant d'obtenir l'inverse exact du modèle flou direct.

Le maximum de ces courbes correspond à la projection des fonctions d'inférence du modèle directe vers le codomaine. En d'autres termes, ces points sont les images des maximums d'appartenance du domaine du modèle direct. Malheureusement, les fonctions standards (par exemple, trapézoïdale, triangulaire et gaussienne) ne peuvent pas représenter fidèlement ces courbes. Donc, nous avons choisi de les approximer par des fonctions triangulaires. Nous avons choisi cette fonction pour les raisons suivantes :

1. Simplicité d'implantation;
2. Diminution de la complexité du système d'inférence.

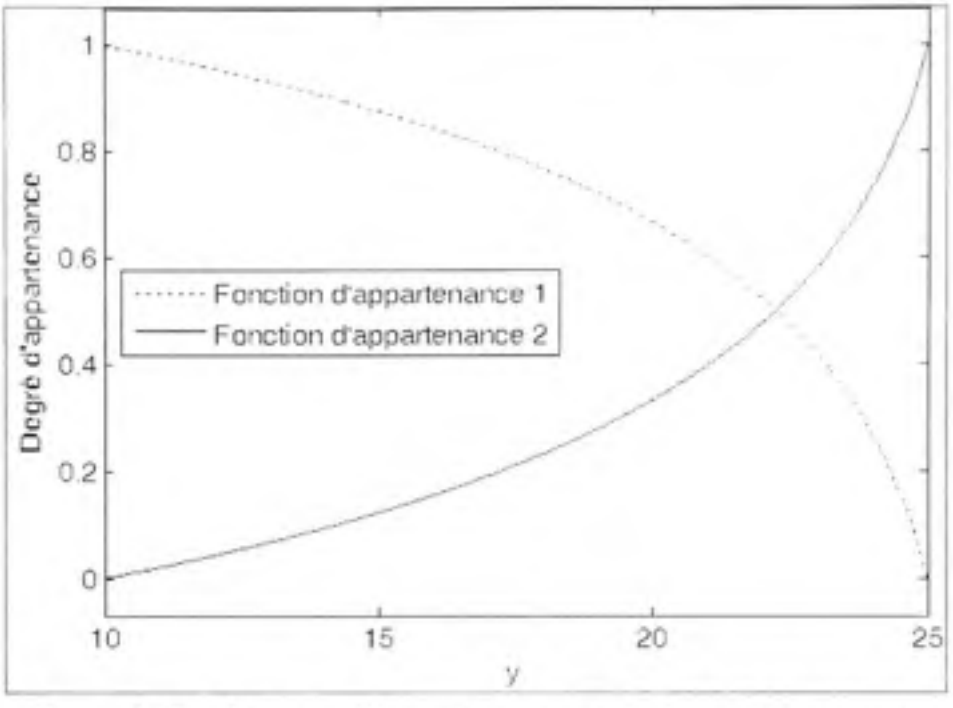

Figure 2.14 Fonction d'appartenance exacte du modèle inverse.

Pour définir une fonction triangulaire, il faut identifier trois points. Dans l'ordre, nous devons connaître le premier minimum, le maximum ainsi que le dernier minimum. Puisque la somme des degrés d'appartenance est égal à un, le premier minimum correspond au maximum de la fonction précédente et le dernier minimum, au maximum de la fonction suivante. Donc, pour définir complètement l'univers de discours d'une entrée du modèle inverse, il suffit de calculer les images correspondant aux points maximum des fonctions d'appartenance du modèle direct. La figure 2.15 illustre l'approximation proposée.

La principale lacune est le décalage du point de coupure comme l'illustre la figure 2.15. Pour éviter ce problème, et ainsi améliorer l'approximation, nous pourrions utiliser une fonction trapézoïdale. Cependant, nous devrions déterminer deux paramètres supplémentaires. De plus, pour localiser ces points, nous devrions connaître le comportement des fonctions d'appartenances exactes. C'est pour cette raison que les fonctions triangulaires sont plus simples d'implantation et que nous préférons utiliser ces dernières. Maintenant, nous avons

tous les éléments pour mettre en application la méthode de découplage sur une application industrielle.

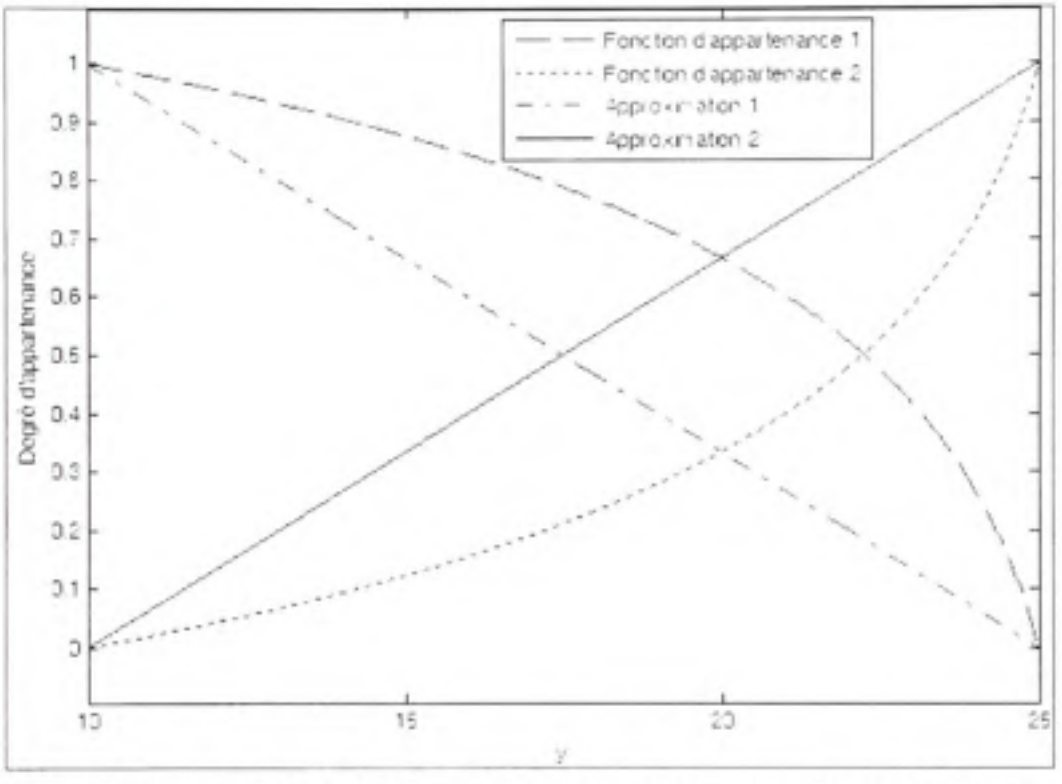

Figure 2.15 Approximation des fonctions d'appartenance du modèle inverse.

2.4 Application de la méthode de découplage au four de thermoformage

Le four de thermoformage est une application industrielle s'adaptant parfaitement à la méthode élaboré. Dans cette section, nous présenterons les étapes du processus de conception appliqué à ce procédé.

2.4.1 Définition des univers de discours des entrées

La première étape du processus de développement est de déterminer le modèle flou direct du système à l'étude. Pour ce faire, nous devons définir les univers de discours des entrées. Les

consignes de température des différents éléments chauffants constituent les entrées de ce système. Alors, les paramètres suivant doivent être identifiés pour toutes entrées :

1. La plage du domaine couverte (la température minimum et maximum de l'élément chauffant par exemple);
2. Le nombre de fonction d'appartenance.

Dans le cadre de ce projet, nous avons choisi de fixer le nombre de fonctions d'appartenance à trois pour diminuer la complexité du système d'inférence flou. De plus, nous proposons de placer les maximums d'appartenance au centre de la plage, puisque nous n'avons pas à priori de connaissances précises du système.

Une caractéristique particulière à ce procédé nous permet de simplifier grandement cette étape. En effet, toutes les entrées sont homogènes. En d'autres termes, ce sont tous des éléments chauffants et la seule distinction est leur localisation dans le four. Donc, tous les univers de discours peuvent utiliser la même définition. L'étape suivante, pour obtenir le modèle direct, est d'identifier l'information nécessaire à l'apprentissage.

2.4.2 Construction de la base de données

Pour effectuer l'apprentissage, nous devons détenir certaines informations sur le comportement du système. Dans le cadre de ce projet, nous effectuons un contrôle en fin de cycle pour des raisons économiques. Alors, la base de données est constituée d'informations obtenues à la fin du cycle de chauffage. Normalement, les tests sont effectués sur un système réel. Par conséquent, la méthode expérimentale doit être adaptée pour tenir compte de la variabilité des données. Dans notre cas, ce processus n'est pas nécessaire, car les données sont obtenues par la simulation d'un modèle mathématique.

Les uplets[1] de la base de données sont constituées des informations suivantes :

1. La consigne de température appliquée à toutes les entrées du système;
2. La température de toutes les sorties à la fin du cycle.

Pour s'assurer de bien représenter l'espace, toutes les intersections des univers de discours ainsi que la combinaison des limites des plages de ces univers doivent être incluses. Par exemple pour un système à deux entrées et utilisant trois fonctions triangulaire pour définir l'univers de discours (semblable à la figure 2.8, mais ayant une étendue de 425°K à 625°K), le nombre de uplets minimum est égal à seize. La figure 2.16 illustre ce cas.

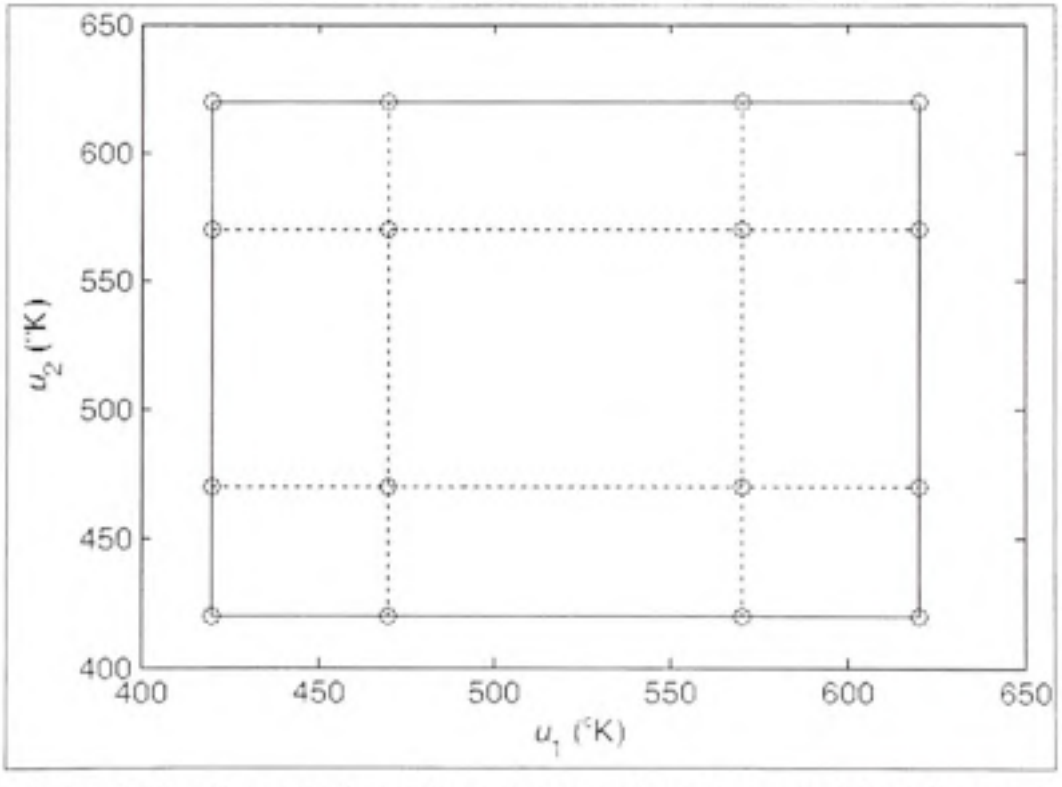

Figure 2.16 Exemple du nombre minimum de uplets pour définir l'espace.

[1] Ensemble d'attributs définissant une entrée d'une base de données.

Les lignes en pointillé correspondent à l'intersection des fonctions d'appartenance dans l'espace et les lignes pleines, les limites des plages d'entrées. Les uplets nécessaires sont identifiés par des cercles.

2.4.3 Identification des fonctions linéaires du conséquent

Pour terminer la modélisation floue directe du système, il faut identifier, à partir des données, les fonctions décrivant le comportement moyen du système pour chacun des sous-espaces correspondant aux règles respectives. Le krigeage dual utilise seulement une partie des données associées à ces sous-espaces puisque nous voulons que la fonction représente à plus de 50% le comportement du système. Il faut effectuer ce calcul autant de fois qu'il y a de règles multiplié par le nombre de sorties. Le nombre de règles dépend quant à lui du nombre d'entrées élevé au nombre de fonctions d'appartenance d'un univers de discours.

2.4.4 Inversion du modèle flou direct

Essentiellement, la méthode reste la même que pour l'exemple à une entrée et à une sortie. La seule différence est dans l'inversion des fonctions linéaire du conséquent des règles. Pour simplifier le développement, nous avons proposé une décomposition de la structure du module de découplage comme l'illustre la figure 2.5. Pour l'occasion, nous revenons à la forme précédente pour clarifier l'inversion. Ainsi, le conséquent de la règle possède n fonctions linéaires associées à n sorties et l'antécédent possède n entrées. Par la structure du problème, nous avons une application linéaire bijective. Donc, nous pouvons obtenir facilement l'inverse de ce système pour chacune des règles comme nous l'avons fait pour le cas SISO. De plus les fonctions d'appartenance du nouveau modèle peuvent être calculées de la même manière qu'illustrée par cet exemple à l'exception que l'application est multidimensionnelle.

2.5 Remarque particulière appliquée au four de thermoformage

Nous avons présenté le processus pour obtenir un module de découplage permettant seulement d'atténuer les interactions puisque la méthode n'est pas exacte. Cependant, l'interprétation du module est un aspect intéressant. En effet, le module détermine le gain réciproque du procédé le plus approprié à une situation. En d'autres terme, c'est comme si nous avions effectué plusieurs linéarisations du système à différentes conditions d'opération et, en exploitation, nous avions sélectionné la meilleure approximation. Donc, notre module exploite une stratégie à gain adaptatif pour pallier au problème de non-linéarité. Ce résultat est dû simplement à la sélection des données et des entrées du système. En effet, les données utilisées génèrent un module de découplage « statique en fin de cycle ». Si nous voulions introduire un aspect dynamique, nous opterions plutôt pour des données incluant les états du système. Toutefois, elle est plus adaptée au domaine temporel discret. Par conséquent, le comportement résultant du module dépend essentiellement de la manière d'utiliser l'approche proposée.

Dans ce chapitre, nous avons établi une méthode permettant de déterminer les fonctions de découplage des entrées. Cette méthode propose d'utiliser les données caractérisant le comportement du système. La première étape est de déterminer un modèle direct du système puisque l'apprentissage est plus simple et beaucoup moins long. La seconde étape est d'inverser le modèle direct. Dans la littérature, nous n'avons pas trouvé de méthode d'inversion exacte permettant l'inversion du modèle tel que construit. C'est pour cette raison que nous avons proposé une procédure permettant d'obtenir une inversion approximative. Nous avons également établi qu'il est possible d'effectuer l'inversion exacte, mais les fonctions d'appartenance résultantes sont complexes à implanter. Ironiquement, la fonction de découplage obtenu n'est rien de moins qu'un contrôleur en boucle ouverte. Donc, la méthode n'est pas robuste aux perturbations. En effet, le contrôleur n'a pas la capacité de récupérer l'erreur en régime permanent. Pour remédier à ce problème, nous proposons d'ajouter une boucle de régulation en utilisant la théorie du contrôle en fin de cycle abordé par Gauthier (2009).

CHAPITRE 3

COMMANDE FLOUE TILC

La commande en fin de cycle par apprentissage successif, ou « Terminal Iterative Learning Control » (TILC), permet l'asservissement d'un procédé dont les grandeurs mesurables ne peuvent pas être évaluées en cours de cycle. Donc, la correction de la commande utilise seulement l'information disponible à la fin du cycle de production (Gauthier, 2009, p. 47). La principale motivation, quant à la prise de mesure en fin de cycle, est d'éliminer les onéreux capteurs de température à l'intérieur du four. Ces derniers seraient remplacés par un rideau optique situé à la sortie du four de thermoformage. Ainsi, une plus grande flexibilité serait possible quant au point de mesure de la température à la surface de la feuille à thermoformer.

Dans ce chapitre, nous présenterons la structure de commande proposée pour asservir le four de thermoformage. Par la suite nous introduirons la commande en fin de cycle par apprentissage successif flou. Par ailleurs, la logique floue a déjà été utilisée pour concevoir des approches d'asservissement ILC (Chien, Hsu et Yao, 2004). Ce projet étendra l'usage de la logique floue à l'approche TILC. Finalement, nous décrirons les différentes étapes menant à l'élaboration d'un contrôleur.

3.1 Structure de la commande

La structure de commande proposée pour implanter la loi de commande s'inspire des méthodes numériques de simulation utilisant un prédicteur-correcteur pour améliorer la précision. En effet, le module de découplage développé au chapitre précédent nous permet d'estimer les entrées à appliquer pour obtenir le profil de température désiré puisqu'il est en fait la fonction réciproque du système. Malgré les perturbations et les erreurs de modélisation, nous avons plus de chance d'obtenir une petite erreur à la première itération. Pour corriger l'erreur résiduelle, nous devons introduire une rétroaction sur les cycles dans la structure. L'asservissement d'un système requiert toujours le calcul d'erreurs entre la sortie

désirée et la sortie obtenue. Donc, sans perte de généralité, la figure 3.1 présente le processus d'évaluation de la commande.

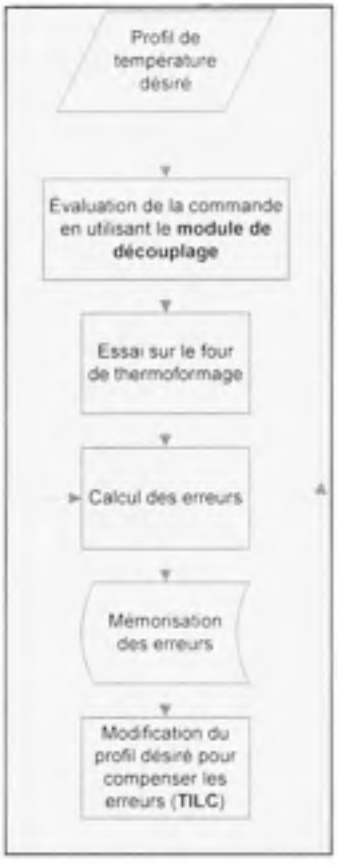

Figure 3.1 Processus d'évaluation de la commande.

Le principal avantage de cette structure est la modularité. En effet, le module de découplage est complètement indépendant du module de modification de la commande. Cette approche offre plus de flexibilité au développeur puisqu'il peut choisir d'autres techniques pour chacun des modules.

Pour arrimer la commande TILC flou à cette structure, il suffit d'utiliser le domaine des cycles. Plus précisément, l'algorithme sera implanté dans le bloc nommé « Modification du profil désiré pour compenser les erreurs ». Un exemple de mise en œuvre de la commande sur le modèle du four en configuration à douze capteurs et à douze éléments chauffant est présenté à l'annexe III. La prochaine section présente la commande TILC flou.

3.2 Commande TILC floue

Dans le cadre de ce projet, nous proposons d'exploiter les systèmes d'inférences flous pour mettre en œuvre le TILC. Cette loi de commande devra permettre d'éliminer les erreurs entre le profil de chauffe et le profil désiré tout en présentant des caractéristiques de stabilité et des performances satisfaisantes.

L'ordre du contrôleur TILC dépend de l'information utilisée. Par exemple, une commande TILC d'ordres N utilisera les N itérations précédentes. L'équation (3.1) permet de calculer la commande à appliquer à l'itération suivante pour un contrôleur d'ordre N.

$$u[k+1] = \sum_{i=0}^{N} \left(L_{i+1} u[k-i] + K_{i+1} \left(y_d[k-i] - y_T[k-i] \right) \right) \tag{3.1}$$

où $u[k]$:= La commande appliquée au cycle k;
N := L'ordre du contrôleur;
$y_d[k]$:= La sortie désirée au cycle k;
$y_T[k]$:= La sortie mesurée au cycle k;
L_i := Le gain sur la commande appliquée au $i^{ème}$ cycle antérieur;
K_i := Le gain sur l'erreur au $i^{ème}$ cycle antérieur.

Pour l'asservissement du four de thermoformage nous avons choisi, pour débuter, d'implanter un TILC d'ordre un (3.2).

$$u[k+1] = u[k] + \Delta u[k] \qquad \text{où} \quad \Delta u[k] = K_1 \left(y_d - y_T[k] \right) \tag{3.2}$$

Par la suite, nous avons choisi d'implanter un TILC d'ordre deux (3.3) pour améliorer la robustesse de la commande (Gauthier, 2009, p. 56). Vous obtiendrez davantage de détails dans la section dédiée à l'application au four de thermoformage. Maintenant, nous devons inclure un système d'inférence flou dans ces deux algorithmes.

$$u[k+1] = L_1 \cdot u[k] + L_2 \cdot u[k-1] + \Delta u[k]$$
$$\text{où} \quad \Delta u[k] = K_1 (y_d - y_1[k]) + K_2 (y_d - y_1[k-1]) \tag{3.3}$$

La variation de commande $\Delta u[k]$ est une composition linéaire des erreurs des cycles antérieurs. Dépendamment des conditions d'opération, le gain optimal peut varier, car le système à l'étude est non-linéaire. Donc, nous devons choisir un gain qui assure la stabilité du système dans toutes ces conditions. Toutefois, ce choix implique une dégradation certaine des performances. Pour remédier à ce problème, nous proposons de calculer la variation de la commande $\Delta u[k]$ en utilisant un système d'inférence flou.

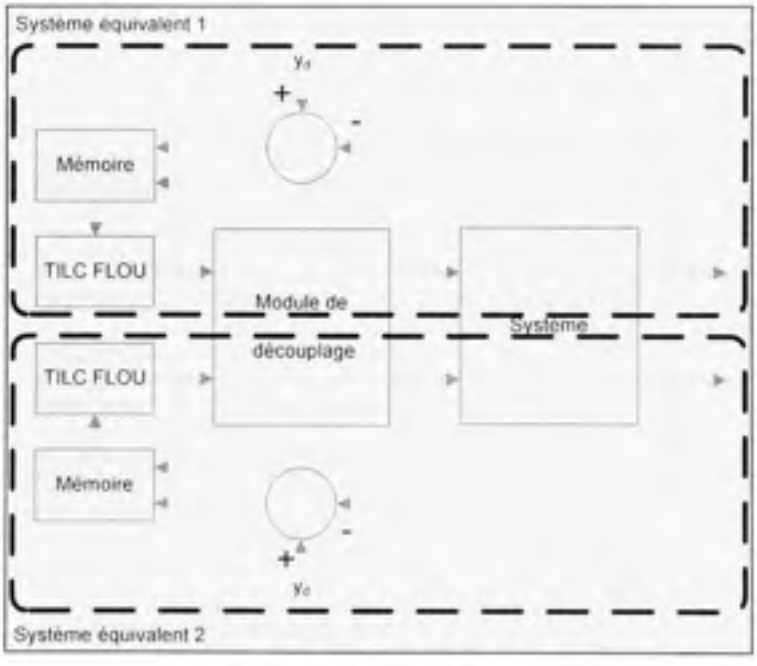

Figure 3.2 Exemple d'implantation de la commande TILC floue.

Pour mettre en œuvre un TILC d'ordre un, il s'agit de concevoir un système d'inférence simple entrée et simple sortie (SISO). Dans le cas du second ordre, nous devons concevoir un système d'inférence multiple entrées et simple sortie (MISO). Comme vous le remarquez, nous n'avons, dans les deux cas, qu'une seule sortie même si nous traitons un problème MIMO. Rappelons-nous le rôle du module de découplage. Ce module est utilisé pour

éliminer, ou atténuer, les interactions existantes. Donc, le résultat de l'assemblage de ce module au système est la décomposition en plusieurs systèmes SISO équivalents. Alors, nous n'avons qu'à asservir ces différents systèmes pour atteindre notre objectif de commande du système MIMO. De plus, dans le cadre de ce projet, les grandeurs à asservir sont toutes identiques. En effet, nous pouvons utiliser la même commande TILC pour les différents systèmes équivalents. La figure 3.2 illustre ce concept.

3.3 Méthode de conception d'un TILC flou

Jusqu'à maintenant, nous avons défini la structure de la commande. Nous avons préalablement établi que la variation de la commande est calculée par un système d'inférence flou. Dans cette section, nous présenterons le processus pour développer cette portion du contrôleur.

3.3.1 Système d'inférence flou

Comme vous le remarquez, le contrôleur de l'étude préliminaire n'est pas très différent. Par ailleurs, un parallèle entre les contrôleurs TILC et les contrôleurs PID est souligné par Chen (1997). Pour cette raison, les outils développés dans le cadre de cette étude peuvent être utilisés. Vous remarquez beaucoup de similitude entre les deux en ce qui concerne le système d'inférence flou.

Définition du type de système d'inférence flou

Nous avons décidé d'utiliser le type TSK d'ordre zéro. La continuité de la surface de commande est garantie lorsqu'on utilise le TSK. Dans notre cas, c'est un avantage indéniable puisque nous utiliserons un algorithme génétique. De plus, ce système d'inférence flou est beaucoup moins lourd que celui de Mamdani.

Nous avons décidé d'utiliser un ordre zéro pour faciliter l'interprétation de notre contrôleur. En effet, ce cas particulier peut être comparé à un système d'inférence flou de Mamdani avec des « singletons » comme fonction d'appartenance de sortie. Ceci signifie simplement que la fonction située dans le conséquent des régles d'inférence est une valeur constante.

Définition des ensembles flous

La définition des ensembles flous se divise en deux grandes étapes. La première étape est de définir le nombre de fonctions d'appartenance définissant l'univers de discours. La seconde étape est de définir les fonctions d'appartenance.

Ensembles flous d'entrées

La complexité du moteur d'inférence croît rapidement avec l'ajout de fonctions d'appartenance en entrée. Ce choix est plutôt arbitraire. Toutefois, avec un minimum de connaissance du système, nous avons décidé de limiter le nombre de fonctions d'appartenance à cinq. Cependant, pour obtenir plus de flexibilité nous pourrions opter pour sept fonctions. De plus, nous avons normalisé l'univers de discours de -1 à 1. Pour ce projet, nous avons choisi des fonctions triangulaires. Pour définir la position des fonctions, il existe deux possibilités. La première possibilité est d'interroger un expert et la seconde, est de placer uniformément les fonctions sur l'intervalle (Cordon *et al.*, 2001, p. 39). Nous avons choisi la seconde solution, car nous allons de toute façon adapter l'ensemble de sortie. La figure suivante présente l'ensemble flou d'entrée utilisé pour ce projet. Le nombre d'entrées dépend de l'ordre du contrôleur TILC.

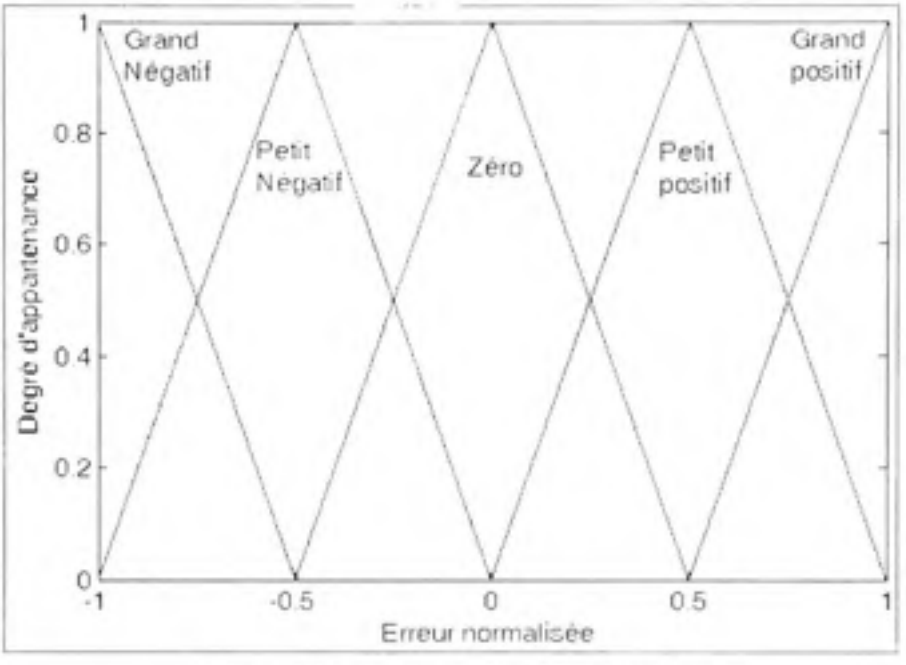

Figure 3.3 Ensemble flou d'entrée.

Ensemble flou de sortie

Comme nous l'avons précisé plus tôt, nous avons choisi un moteur d'inférence flou de type TSK d'ordre zéro. Donc, l'ensemble de sortie est une série de singletons. Pour offrir plus de capacité d'adaptation, nous avons fixé le nombre de singletons égal au nombre de règles d'inférence.

Définition des opérateurs

Le type de moteur d'inférence nous impose l'opérateur d'implication et d'agrégation. Respectivement, nous devons utiliser le produit et la somme pour effectuer ces traitements. Pour la fonction « ET » nous avons choisi d'utiliser l'opérateur minimal. Nous pourrions également utiliser le produit. La fonction « OU » n'est pas utilisée dans la définition des règles. Nous l'avons imposée arbitrairement à l'opérateur maximum. Pour la défuzzification, nous avons sélectionné la moyenne pondérée des sorties.

Définition des règles d'inférence

La définition des règles d'interférence est très simple. En optimisant l'ensemble de sorties, la topologie du moteur d'inférence demeure constante. Précédemment, nous avons choisi d'utiliser des fonctions linéaires d'ordre zéro dans le conséquent des règles. En d'autres termes, nous utilisons un terme constant. Pour faciliter le travail, toutes les N règles seront associées à un paramètre P_i. Notez que les variables P_i où $i = \{1,2,\ldots, N\}$ correspondent à la position du $i^{ème}$ singleton.

3.3.2 Méthode intuitive pour définir la position des singletons

La première méthode proposée repose sur les connaissances d'un expert. Cette méthode consiste simplement à définir le conséquent de la règle en formulant une question. Nous privilégions cette approche pour l'apprentissage d'un contrôleur TILC d'ordre un, car elle est rapide et abordable. La complexité du second ordre, due au grand nombre de combinaisons, rend cependant la tâche plus ardue. Donc, nous proposons une seconde méthode permettant d'atteindre des objectifs concurrents tels que la rapidité et la convergence monotone.

3.3.3 Algorithme génétique pour définir la position des singletons

Les algorithmes génétiques (AG) sont des outils d'optimisation plus efficaces qu'une recherche exhaustive de l'espace, car ils convergent plus vite vers de bonnes solutions. Par ailleurs, la recherche exhaustive est parfois irréalisable. De plus, l'utilisation des AG n'impose pas de conditions contraignantes quant à la fonction à optimiser. Par exemple, la fonction n'a pas besoin d'être dérivable ou linéaire.

Pour notre application, nous avons jusqu'à 26 paramètres à déterminer simultanément. Donc, l'espace de recherche est très grand et il est impossible d'évaluer toutes les combinaisons dans un temps raisonnable avec les outils informatiques actuels. Cette dernière affirmation ne signifie pas obligatoirement que les AG représentent la solution universelle aux problèmes

d'optimisation. Les caractéristiques de l'espace de recherche affectent également l'efficacité de ce genre d'algorithme.

Dans cette section, nous présenterons l'algorithme utilisé pour automatiser le processus de définition du conséquent des règles d'inférence. Nous justifierons les choix de la méthode de codage ainsi que la définition de la fonction de coût. Nous indiquerons également la procédure pour définir les paramètres et les opérateurs génétiques.

Présentation de l'algorithme

Pour l'automatisation de la définition du conséquent des règles d'inférence, nous avons choisi une variante de l'algorithme génétique simple (AGS). Cette variante est présentée à la figure 3.4. Un AGS est exécuté plusieurs fois en insérant la meilleure solution de l'itération précédente. Cette approche permet de raffiner la solution d'une itération à l'autre.

Pour l'implantation, nous avons décidé d'utiliser « MATLAB » puisque nos simulations sont exécutées sur cette plateforme. Nous avons décidé d'utiliser un « toolbox » externe (Chipperfield *et al.*, N.d.) qui offre des fonctions pour mettre en œuvre rapidement des algorithmes génétiques. Le script et les fonctions pour l'implantation de cet algorithme sont présentés en annexe IV de ce document.

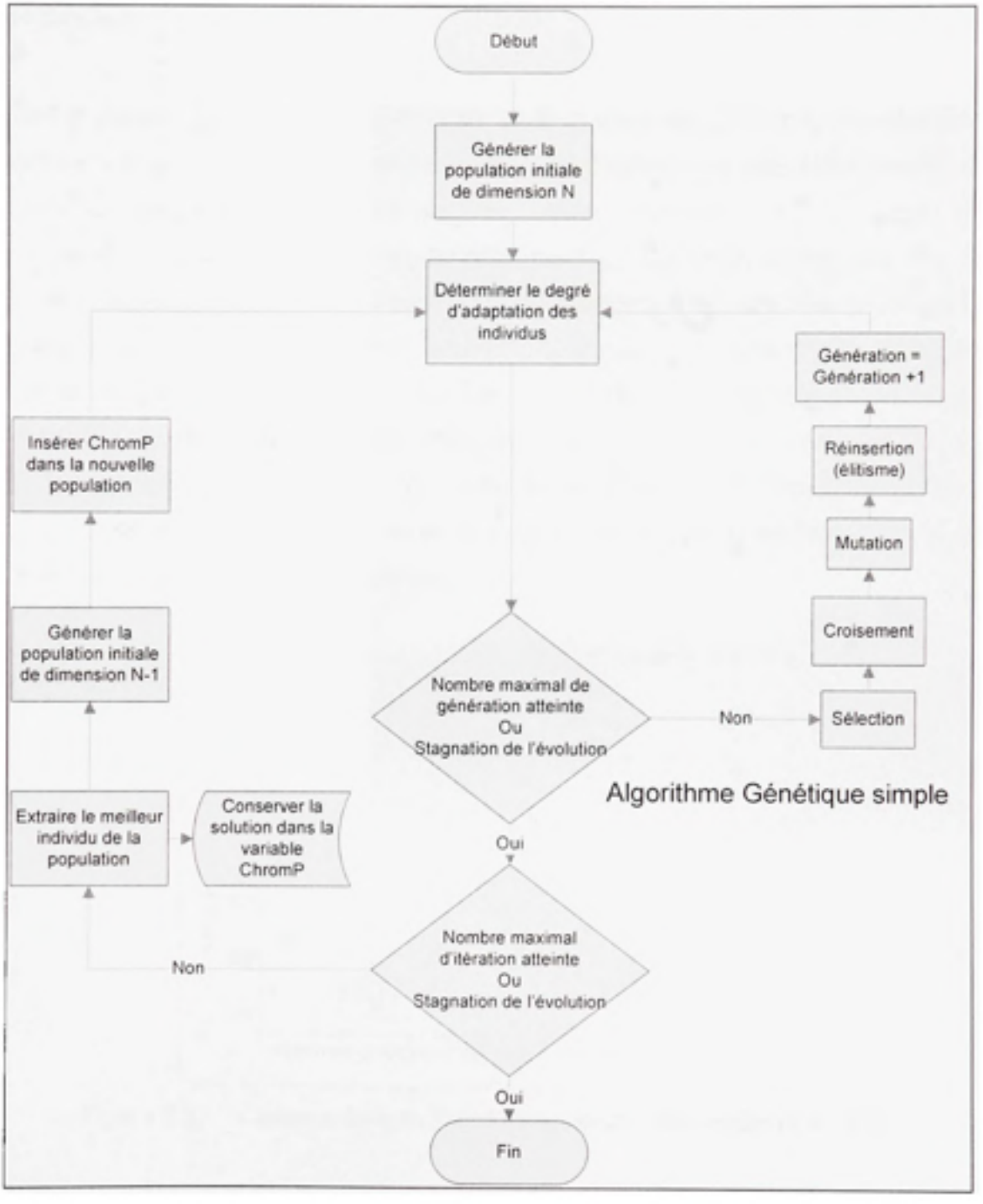

Figure 3.4 Algorithme génétique proposé.

Justification

Avant d'adopter cette approche, nous avons tenté d'utiliser un AGS pour résoudre notre problème lors de l'étude préliminaire (voir annexe V). Les premiers tests étaient concluants et nous arrivions à trouver des solutions intéressantes. Toutefois, l'AGS ne donnait pas systématiquement de bonnes solutions. En effet, parfois, l'algorithme convergeait vers des optimums locaux indésirables. En consultant les chromosomes des dernières générations de ces cas, nous avons remarqué qu'il n'y avait plus beaucoup de diversité. Donc, nous avons tenté de changer les probabilités de mutation pour remédier à ce problème. Nous avons observé une sensible amélioration. En effet, une probabilité de 0.5 permettait d'obtenir dix fois sur vingt un résultat acceptable. Une probabilité de 0.7 permettait d'obtenir neuf fois sur vingt un résultat acceptable. Nous avons également observé que la population de départ influençait la convergence de l'algorithme.

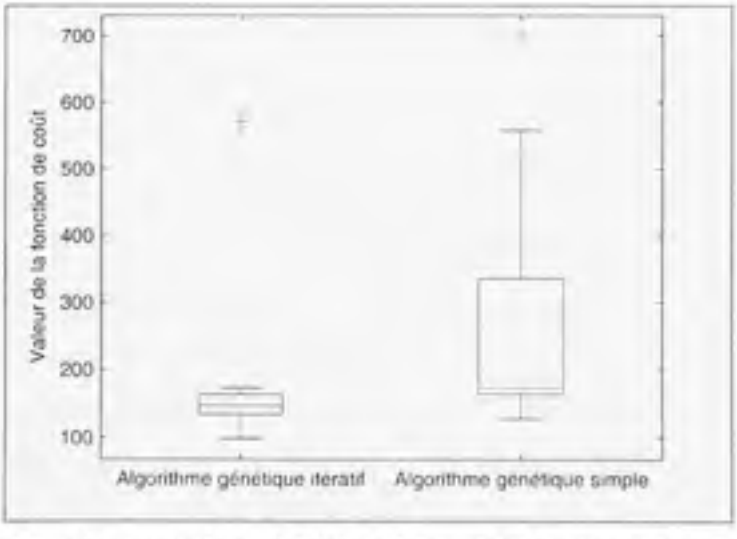

Figure 3.5 Comparaison de l'algorithme génétique simple et itératif.

Dans la littérature, il est suggéré d'introduire dans la population initiale des individus connus pour améliorer la rapidité de convergence (Benahmed, 2002). Dans notre cas, nous ne connaissons pas nécessairement de bonnes solutions avant l'exécution. Donc, nous avons déterminé une première solution avec notre AGS, pour ensuite l'introduire dans la seconde

itération. Ainsi, nous espérons affiner, au fur et à mesure, la solution obtenue à la première itération. En effet, le bagage génétique est renouvelé à chacune des itérations. Donc, nous améliorons nos chances d'obtenir de bonnes solutions. La figure 3.5 montre la différence entre l'AGS et l'algorithme itératif suggéré précédemment. Nous avons exécuté vingt fois les algorithmes et identifié la meilleure solution de ces réplications.

Cette méthode diminue l'impact de la population initiale et l'effet de la convergence vers des optimums locaux. De plus, la variabilité obtenue est significativement inférieure en utilisant l'algorithme génétique itératif. Veuillez noter que les paramètres utilisés pour l'exécution seront justifiés subséquemment. Puisque l'AGS est un cas particulier de notre algorithme, nous utiliserons l'AGS pour définir les opérateurs génétiques et leurs paramètres. Nous utiliserons le problème de l'étude préliminaire (annexe V) pour effectuer les tests.

Méthode de codage

La méthode de codage est le premier élément à déterminer. En effet, les autres opérateurs dépendent de ce choix. Pour représenter notre chromosome, nous avons le choix entre une représentation réelle ou binaire. Pour le codage binaire, il est possible d'utiliser une représentation standard ou bien une représentation suivant le code Gray. Cette dernière diminue l'effet de la discontinuité de l'espace de représentation (Herrera, Lozano et Verdegay, 1998). Donc, le codage binaire standard est éliminé, car le risque d'une convergence prématurée vers un optimum local est plus élevé. Pour la représentation binaire, la longueur d'un gène dépend de la précision ou de la résolution que l'on désire obtenir. Donc, la longueur du chromosome grandit rapidement pour améliorer la précision de la solution. Nous savons que la rapidité de convergence dépend également de la longueur de ce chromosome. De plus, les bornes inférieures et supérieures doivent être connues. Dans notre situation, nous avons défini la plage selon les limites raisonnables de fonctionnement de notre procédé. Cette dernière limitation ne nous affecte pas. Toutefois, lors de l'évolution, l'algorithme optimise tous les paramètres des gènes en simultanés. Nous savons que les chiffres les moins significatifs de la représentation ne sont pas importants au début du

processus. Alors, des efforts inutiles sont déployés au départ pour optimiser ces paramètres (Herrera, Lozano et Verdegay, 1998). La représentation réelle ne rencontre pas ce problème. De plus, cette représentation est plus naturelle selon ces auteurs. Donc, nous avons choisi cette représentation comme méthode de codage. L'implantation de cette méthode est également beaucoup plus simple et n'exige pas de conversion. De plus, ce nombre de paramètres est diminué puisque nous n'avons pas à définir la longueur du gène. Plus précisément, un gène correspond à un singleton d'une règle de notre système d'inférence flou ou à la constante d'une fonction de sortie d'un TSK d'ordre zéro. Donc, la valeur réelle d'un gène est directement utilisé dans le FIS. La longueur du chromosome est directement liée à la taille du FIS. En effet, nous avons choisit d'assigner un singleton par règle d'inférence pour permettre plus de flexibilité. Par exemple, dans le cas de notre commande en fin de cycle par apprentissage itératif flou d'ordre deux que nous présenterons ultérieurement, nous avons 25 règle ou singletons en plus d'un paramètre de pondération des entrées antérieures. Donc, le chromosome se compose de 26 gènes.

Opérateurs génétiques

Les opérateurs génétiques sélectionnés ont un impact important sur le comportement de l'algorithme. Nous vous rappelons que les opérateurs utilisés dans le cadre de ce projet sont les mêmes pour les deux algorithmes présentés plus tôt.

Fonction d'évaluation

La fonction d'évaluation que nous avons choisie est le classement. Plus précisément, la méthode assigne une valeur entre zéro et deux selon la position relative de l'individu dans la population. « Cette méthode permet d'éviter la convergence prématurée et d'accélérer la recherche quand la population converge vers une solution » (Benahmed, 2002). Pour cette raison, nous croyons que cette méthode est plus appropriée que la mise à l'échelle linéaire.

Sélection

Deux outils de sélection sont disponibles. Le premier est la roulette biaisée et le second, est le « stochastic universal sampling ». La roulette biaisée est la plus utilisée pour la mise en œuvre des AG (K.F. Man, 1999). Par ailleurs, cet outil correspond exactement au rôle de la sélection théorique, soit de favoriser les individus qui ont une meilleure capacité d'adaptation sans toutefois éliminer les mauvais candidats. Donc, cette méthode correspond à nos besoins et sera utilisée.

Mutation

Le choix de la représentation réelle comme méthode de codage nous impose le choix de la méthode de mutation. L'outil utilisé offre seulement la fonction « mutbga » pour la mutation sur des populations à valeurs réelles (Chipperfield *et al.*, N.d., p. 2.17). L'équation suivante montre le formalisme de la méthode de mutation.

$$
\begin{aligned}
&V_f = V_i + R \times Shrink \times Delta \times S \\
&\text{où } V_i := \textit{Variable avant mutation;} \\
&\quad R := \tfrac{1}{2} \textit{ du domaine de recherche;} \\
&\quad Shrink := \textit{Paramètre de restriction sur l'amplitude de la mutation;} \\
&\quad S = \pm 1, \textit{ selon la probabilité de mutation;} \\
&\quad Delta = \sum_{i=0}^{m-1} \alpha_i 2^{-i},\ \alpha_i = 1 \textit{ avec la probabilité } \tfrac{1}{20} \textit{ sinon } 0.
\end{aligned}
\tag{3.4}
$$

Selon Chipperfield, cette méthode permet de visiter le voisinage de la variable avec une probabilité plus forte pour des changements de faible amplitude à l'intérieur d'un hyper-cube.

Croisement

Pour définir l'opérateur génétique de croisement, nous avons comparé expérimentalement par rapport à notre problème de sélection les différents opérateurs disponibles. Nous avons utilisé l'AGS pour produire ces résultats. Nous avons limité nos expérimentations aux outils offerts dans le « toolbox ». En effet, trois algorithmes de croisement sont proposés. Le premier est le croisement discret. L'enfant généré par cet algorithme est obligatoirement dans l'espace formé par les différentes combinaisons de gènes des parents. En d'autres termes, l'enfant est formé par les gènes prélevé sur les parents. Le deuxième est le croisement ligne. Les nouveaux gènes sont obtenus en additionnant le gène du premier parent à proportion de la différence des gènes des deux parents. La proportion est constante pour tous les gènes calculés. Donc, l'enfant généré peut sortir de l'espace défini par les parents. Le troisième est le croisement intermédiaire. Cet algorithme est très semblable à la recombinaison ligne. La seule différence est la variation du facteur de proportionnalité. En effet, pour toutes les paires de gènes des parents ce facteur change. La description complète des opérateurs est disponible dans le manuel l'accompagnant (Chipperfield *et al.*, N.d.). Les figures suivantes présentent les meilleurs individus de vingt réplications en fixant la probabilité de mutation à zéro. La méthode « recdis » est éliminée, car elle n'offre pas de bons résultats en moyenne et converge trop rapidement vers des optimums locaux. Le choix entre les autres méthodes est cependant moins évident. En effet, il n'y a pas de différence très significative. Nous avons préféré « recint », car cette méthode semble donner généralement de meilleurs résultats. L'équation suivante présente la méthode d'évaluation du croisement « recint ».

$$
\begin{aligned}
&O_1 = P_1 \times \alpha\left(P_2 - P_1\right) \\
&\text{où } O_1 := \textit{Enfant;} \\
&\quad P_1 := \textit{Premier parent;} \\
&\quad P_2 := \textit{Second parent;} \\
&\quad \alpha := \textit{Facteur de mise à l'échelle variant aléatoirement } [-0.25;1.25] \\
&\qquad \textit{pour toutes les paires de gènes des parents.}
\end{aligned}
\tag{3.5}
$$

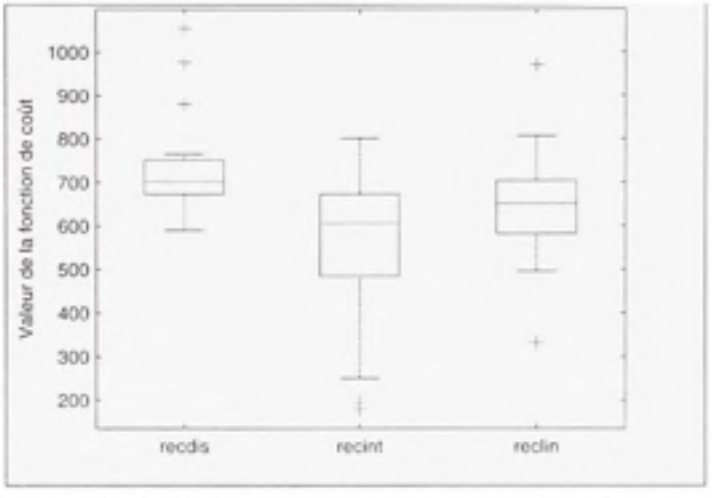

Figure 3.6 Comparaison des opérateurs de croisement selon leur coût.

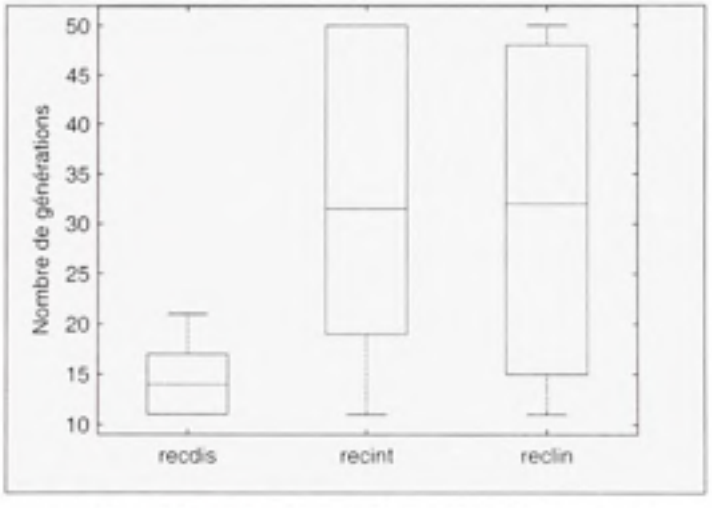

Figure 3.7 Comparaison des opérateurs de croisement selon leur moment d'arrêt.

Réinsertion

Pour augmenter la rapidité de la convergence, il est souhaitable d'adopter le principe d'élitisme. En effet, la propagation du meilleur individu d'une génération à l'autre permet de conserver un focus vers la solution optimale. Lors de nos tests préliminaires, nous avions constaté qu'il était préférable de propager un seul individu. En effet, la diversité des n-

meilleurs individus était faible. Donc, la propagation de plusieurs individus favorisait une convergence prématurée à des optimums locaux.

Fonction de coût

La fonction de coût doit être adaptée au problème à résoudre. En effet, la recherche est guidée par cette fonction. Selon notre expérience, l'objectif d'optimisation doit être clair pour améliorer les chances de convergence. Donc, la fonction de coût doit explicitement définir le comportement à optimiser. Par exemple, si un dépassement est inadmissible, un coût important doit y être associé. Donc, une fonction de coût bien définie n'a pas d'impact sur l'algorithme génétique, mais elle a un impact important sur le lieu de convergence. Dans notre cas, nous avons deux fonctions de coût à définir. La première est utilisée pour effectuer l'étude préliminaire alors que la seconde, pour la définition des contrôleurs TILC.

Étude préliminaire

Le but de l'optimisation du conséquent des règles d'inférence dans le cas de l'étude préliminaire (annexe V) est d'obtenir les meilleures performances. En contrôle, il existe des mesures pour qualifier les capacités d'un système. En effet, le temps de réponse, le pourcentage de dépassement et l'erreur en régime permanent sont tous des exemples de critère. Donc, nous utiliserons ces paramètres pour qualifier le degré d'adaptation d'un individu. De plus, nous imposerons une contrainte sur la commande. L'objectif de cette contrainte est de réduire les changements brusques de la position de la valve. Ainsi, nous pourrons augmenter la durée de vie utile de l'équipement. Nous avons choisi de normaliser entre 0 et 1 les mesures. Le pourcentage de dépassement correspond déjà à ce critère. L'erreur en régime permanent est normalisée par rapport à la consigne. Le temps de réponse est normalisé par rapport au temps de simulation. Cette dernière normalisation peut paraître un peu étrange. Toutefois, il ne faut pas oublier que toutes les solutions sont comparées suivant le même critère. Alors, même s'il n'y a pas de signification physique à cette normalisation, elle a une signification relative par rapport aux autres solutions. Malgré tout, il

faut être prudent dans le choix du temps de simulation. En effet, nous devons privilégier un temps raisonnablement long afin d'atteindre le régime permanent. À l'inverse, le temps de simulation trop long diminuerait la sensibilité de la normalisation. La pénalité sur la commande est quant à elle déterminée suivant la relation suivante :

$$Pénalité\ sur\ la\ commande = \min\left(1; e^{(0.02 \cdot Nombre\ de\ dépassement\ de\ seuil)} - 1\right) \tag{3.6}$$

Le coefficient de 0.02 a été déterminé pour autoriser un nombre acceptable de transgression au seuil défini. Nous avons choisi un seuil de 0.1 qui correspond à un taux de variation de l'ouverture de la valve de 10% par seconde. Finalement, la fonction de coût est simplement la somme de ces mesures multipliée par cent pour donner plus de sensibilité à l'algorithme génétique :

$$C = \left(D + {T_{2\%}}/{T_{sim}} + P + {y_f}/{y_d}\right) \cdot 100$$

où $D :=$ *% de dépassement*
$T_{2\%} :=$ *Temps de réponse à 2%*
$T_{sim} :=$ *Temps de simulation*
$P :=$ *Pénalité sur la commande*
$y_f :=$ *Valeur de la sortie en régime permanent*
$y_d :=$ *Valeur désirée*

(3.7)

Nous avons choisi de ne pas appliquer de pression plus importante sur l'un ou l'autre des critères. Toutefois, selon le contexte, nous pourrions assigner plus d'importance à un des éléments en le multipliant par un poids supérieur à 1.

Contrôleur TILC

Les objectifs du contrôleur TILC sont semblables au précédent cas. En effet, l'erreur en régime permanent doit être également minimisée ainsi que le temps de réponse. Toutefois, nous utilisons le domaine des cycles plutôt que le domaine continu. Donc, la granularité est

grossière. Alors, nous proposons d'utiliser l'erreur quadratique cumulée pour remplacer ces deux mesures. Cette nouvelle mesure est adaptée et conserve l'objectif des deux anciennes. En effet, si une erreur en régime permanent subsiste, l'erreur cumulée augmente. De plus, une réponse lente aura une erreur cumulée plus grande qu'une réponse rapide.

Le four de thermoformage impose également une contrainte fonctionnelle. En effet, il n'est pas souhaitable qu'une feuille de plastique soit surchauffée, car elle risque de coller sur les éléments chauffants situés sous cette dernière. Donc, un dépassement n'est pas admissible. Nous ajouterons, au coût de la solution, une forte pénalité lorsque cette situation se présentera. L'équation suivante présente le calcul effectué.

$$C = |E|_2 + n \cdot 1000$$
$$\text{où } E := \textit{Vecteur d'erreur d'une sortie} \qquad (3.8)$$
$$n := \textit{nombre de dépassement de la consigne}$$

Choix des paramètres

Les paramètres des opérateurs génétiques ont un impact important sur les algorithmes génétiques. De plus, il n'y a pas de solution universelle pour résoudre tous les problèmes d'optimisation. Donc, les paramètres sont obtenus expérimentalement. Toutefois pour certains d'entre eux, nous avons imposé une valeur pour diminuer le temps de développement. En effet, les paramètres ne sont pas indépendants (Eiben, Hinterding et Michalewicz, 1999). De plus, la validation expérimentale requiert beaucoup de temps de calcul. Donc, il est impossible de déterminer la meilleure combinaison de paramètres dans un temps raisonnable.

Taille de la population

La taille de la population est un paramètre important. En effet, si elle est trop petite, l'algorithme risque de converger vers des optimums locaux. Toutefois, si elle est trop grande,

la convergence sera très lente. Il est recommandé d'utiliser une taille de population comprise entre vingt et trente individus (Eiben, Hinterding et Michalewicz, 1999). Pour notre problème, nous avons choisi la borne inférieure. Nous sommes cependant conscients que notre problème de convergence vers des optimums locaux peut être causé par ce choix, mais le temps de calcul raisonnable demeure un objectif important. Il ne faut pas oublier que nous tentons d'optimiser un petit contrôleur flou. Alors, si nous voulons éventuellement traiter des contrôleurs à multiples sorties, notre outil d'aide à la conception doit être très performant sur le problème actuel.

Nombre maximal de générations

Le nombre maximal de générations est probablement le paramètre le moins critique. En effet, il s'agit qu'il soit suffisamment grand pour permettre l'obtention de la solution. Lors de notre étude préliminaire, nous avons fixé ce paramètre à 100 générations. Toutefois, nous avons remarqué que l'évolution se stabilisait à moins de 50 générations. Donc, nous avons choisi cette dernière valeur comme nombre maximal de générations. Cette valeur est cependant très discutable. Normalement, le nombre de générations varie entre 1000 et 10 000 (Benahmed, 2002). Toutefois, nous avons remarqué que la diversité diminuait rapidement, même avec une probabilité de mutation élevée. Donc, il devenait inutile de poursuivre. Généralement, le critère de stagnation de l'évolution stoppait l'algorithme avant l'atteinte du nombre maximal de générations.

Mutation

La méthode de mutation utilisée offre deux paramètres. Le premier paramètre correspond à la restriction de la plage de mutation admissible et le second, au taux de mutation.

Restriction de la plage de mutation

Dans le cadre de ce projet, nous avons choisi de modifier ce paramètre en fonction de la génération. La relation est inversement proportionnelle. Cette valeur restreint l'effet de la mutation sur le gène. Donc, nous voulons favoriser l'exploration au début de l'algorithme et favoriser l'exploitation à la fin du processus. Toutefois, nous avons imposé une borne minimale à ce paramètre afin de conserver l'effet de la mutation tout au long du processus d'optimisation. L'équation suivante montre la relation déterminant ce paramètre.

$$Shrink = \max\left(\frac{-1}{Nombre\ Maximum\ de\ générations} \cdot génération + 1\ ;\ 0.5\right) \tag{3.9}$$

Probabilité de mutation

La probabilité de mutation a été sélectionnée sur une base expérimentale. En effet, nous avons effectué plusieurs essais avec des probabilités différentes. Nous avons utilisé les paramètres et les opérateurs définis préalablement pour effectuer ces essais. Pour chacun d'eux, nous avons exécuté 20 réplications pour valider notre choix. La figure 3.8 présente les meilleurs individus en termes d'adaptation des différents essais effectués. Nous avons choisit de représenté ce résultant sur forme de quartile. Donc, la ligne séparant la boîte illustre la valeur de la médiane.

En consultant ce graphique, nous constatons que la meilleure probabilité se situe entre 0,2 et 0,7. Donc, nous avons refait le test avec les mêmes paramètres en se concentrant sur cette plage. La figure 3.9 présente les résultats obtenus.

Nous pouvons éliminer la probabilité à 0,3 car elle ne donne pas de résultat satisfaisant. Il n'y a pas de différence très significative entre les autres probabilités. En effet, les médianes sont

toutes alignées. De plus, il ne faut pas oublier que le nombre d'échantillons est relativement restreint. Toutefois, nous avons préféré utiliser la probabilité de 0,5 car elle est au milieu de l'intervalle des bonnes solutions et semble donner généralement de meilleurs résultats.

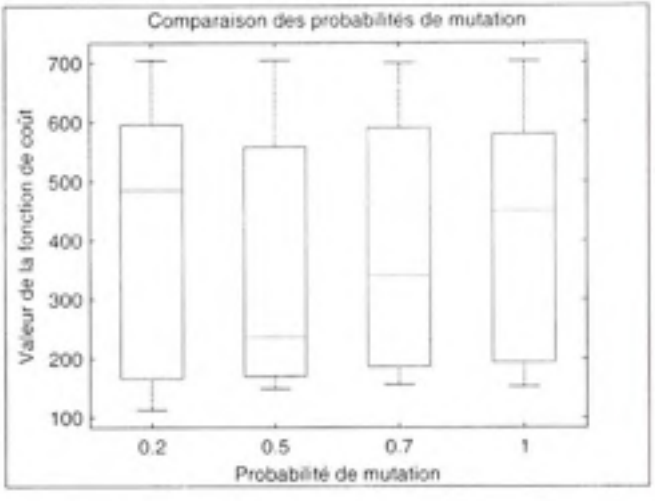

Figure 3.8 Comparaison des probabilités de mutation.

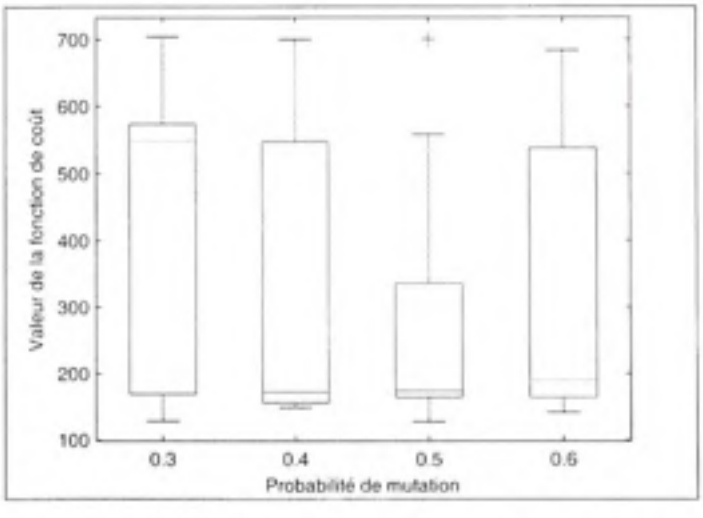

Figure 3.9 Comparaison des probabilités de mutation sur une plage réduite.

Maintenant, nous avons tous les éléments pour réaliser l'asservissement du four de thermoformage selon différentes configurations. Le chapitre suivant est consacré à la présentation des résultats d'expérimentation sur le modèle de simulation.

CHAPITRE 4

RÉSULTATS DES SIMULATIONS

Ce dernier chapitre sera consacré à la présentation et à l'analyse des résultats de simulations. Nous avons testé plusieurs configurations du four de thermoformage en débutant par un cas simple. Pour chacune des configurations, nous présenterons tout d'abord le module de découplage développé avec la méthode expliquée plus tôt. Par la suite, nous présenterons le module de contrôle assurant l'asservissement en température à la surface de la feuille.

4.1 Configuration à deux éléments chauffants et à deux capteurs

La configuration élémentaire consiste au regroupement des éléments du dessus et de ceux du dessous. Ces deux groupes seront associés respectivement aux capteurs IR_{T1} et IR_{B1}. Les éléments chauffant du premier groupe, tout comme ceux du second groupe, partageront la même consigne de température.

4.1.1 Module de découplage

La première étape du développement de la commande est la conception du module de découplage. En appliquant exactement la méthode proposée au chapitre 1, nous obtenons ce module de découplage. La figure 4.1 présente la distribution des erreurs et le tableau 4.1, quelques statistiques. Ces résultats sont obtenus par simulation du modèle en posant v_1 et v_2 à la même valeur. En d'autres termes, nous désirons atteindre la même température des deux côtés de la feuille. Nous avons fait ce choix pour comparer la qualité des différentes configurations en conservant un temps de simulation raisonnable. De plus, l'ensemble de test est constitué de tous les entiers entre 400°K et 450°K incluant ces valeurs.

Tableau 4.1 Statistiques des erreurs de découplage de la configuration 2x2

no. de l'entrée	maximum (°C)	moyenne (°C)	médiane (°C)
1	4,6945	2,4156	2,1073
2	4,6945	2,4156	2,1073

Comme vous le remarquez, les statistiques des deux composantes du module de découplage sont identiques. Cette observation s'explique par la symétrie suivant le plan de la feuille de plastique existant dans le modèle de simulation. Sur l'étendue de 50°C, l'erreur maximale est inférieure à 5°C. Actuellement, le travail des opérateurs est de maintenir les erreurs dans cette bande. Donc, avant même d'effectuer un asservissement, nous obtenons une prédiction acceptable pour les opérations actuelles.

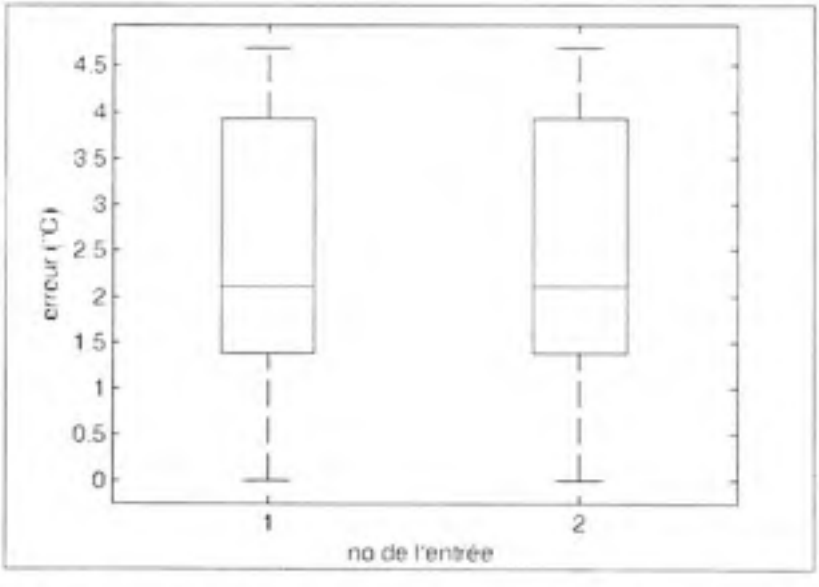

Figure 4.1 Distribution des erreurs (°C) de découplage de la configuration 2x2.

La figure 4.2 illustre la répartition de l'erreur en fonction de la température désirée. Comme vous le remarquez, l'erreur maximale apparaît au milieu de l'intervalle. Cette observation n'est pas étrangère à la définition des sous-espaces des univers de discours. En effet, l'espace couvert par la règle correspondante au centre illustré à la figure 2.16 est beaucoup plus grand. Alors, l'approximation par un plan de cette zone est beaucoup plus grossière.

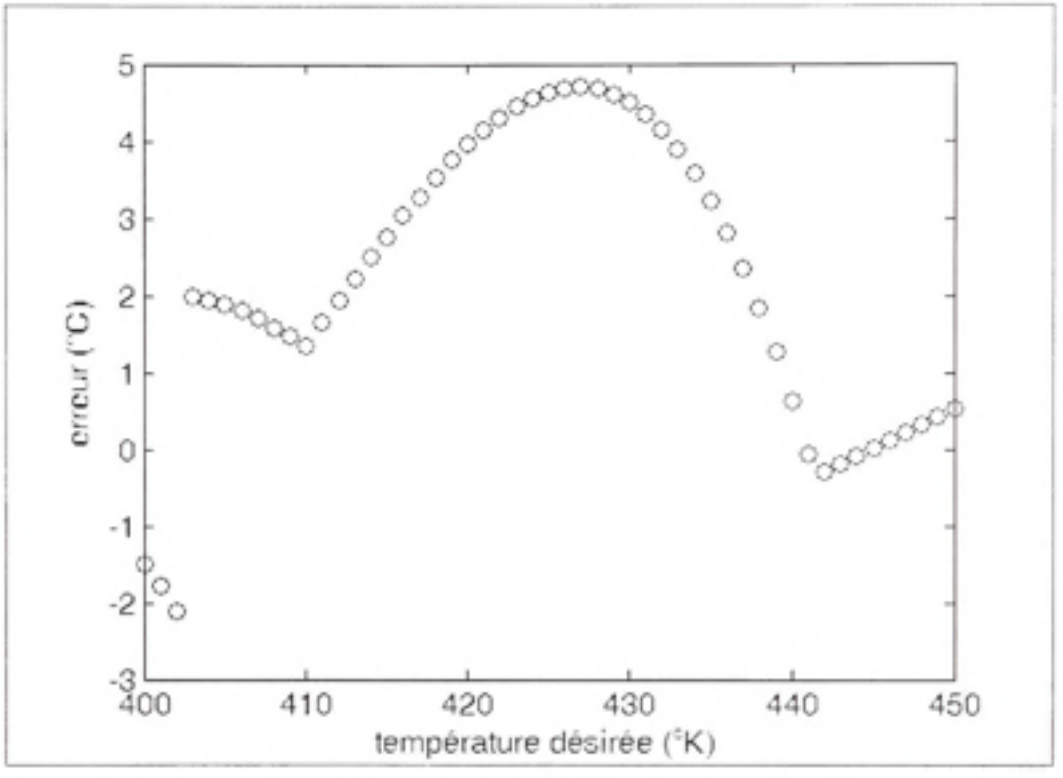

Figure 4.2 Répartition des erreurs de découplage pour la configuration 2x2.

La figure 4.3 présente la surface générée par une des deux composantes du module de découplage. En regardant cette figure, nous pouvons avoir une idée générale de l'espace de faisabilité. En effet, lors de nos tests, nous avons constaté qu'un écart de 50°C entre le dessous et le dessus n'était pas possible comme l'illustre cette figure.

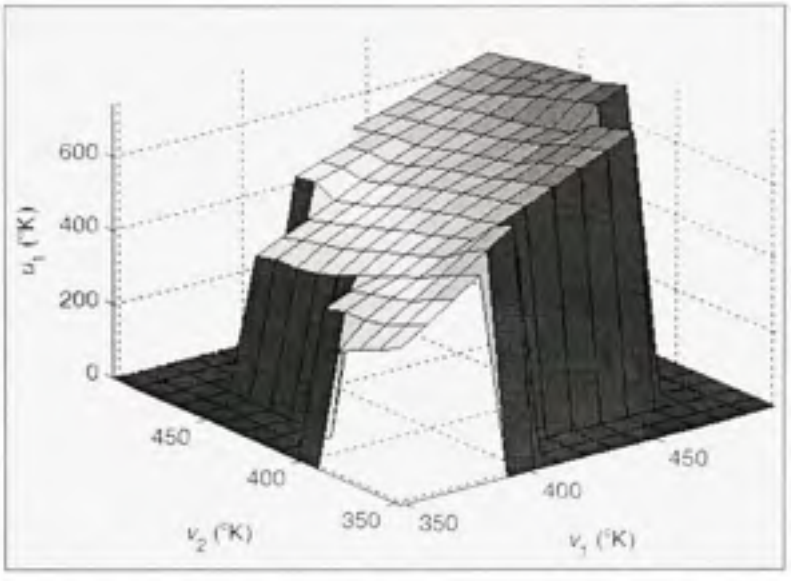

Figure 4.3 Surface de découplage des entrées en configuration 2x2.

Pour mieux apprécier l'espace de faisabilité des solutions, nous avons superposé les données réelles obtenues par la simulation du modèle à la figure 4.4.

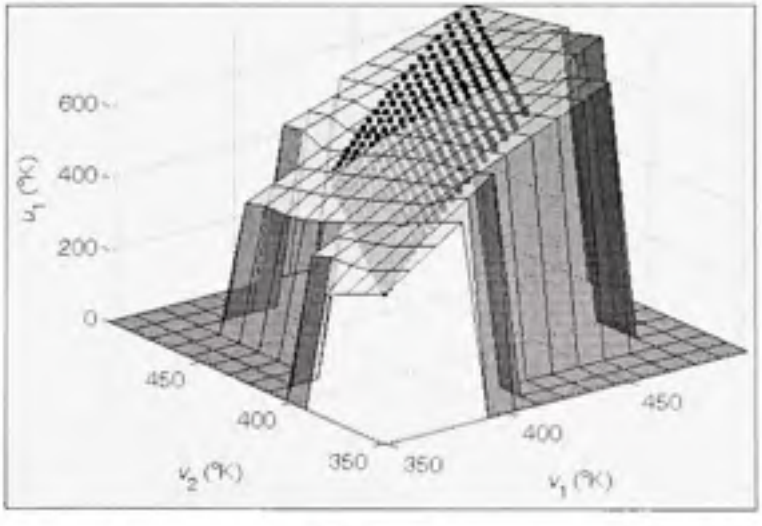

Figure 4.4 Superposition de la surface exacte et du module flou.

Vous pourrez également apprécier l'approximation du module de découplage dans son ensemble. La surface exacte représentée par les carrés est obtenue par simulation du modèle de thermoformage. Plus précisément, nous avons utilisé les deux sorties du modèle comme

variables indépendantes et une variable d'entrée du modèle comme variable dépendante. En d'autres termes, cette surface est réciproque à la surface caractérisant le comportement direct. Lors de la simulation, nous avons couvert l'espace compris entre les valeurs de 420 à 620°K sur toutes les entrées du modèle avec un incrément de 10°K. Vous remarquerez à la figure 4.4 que la sensibilité augmente avec la température. Donc, l'asservissement pour de faibles températures est plus facile et moins critique que pour les hautes températures. Toutefois, nous n'utiliserons pas toute cette surface. En effet, notre objectif est d'asservir le système pour des températures situées entre 400 et 450°K.

4.1.2 Asservissement

Pour réduire l'erreur, nous asservirons la température en utilisant une commande TILC d'ordre un. L'ordonné des ronds illustré à la figure 4.5 correspondent aux cinq constantes des fonctions de sortie, situé dans le conséquent des règles, du TSK d'ordre zéro. La courbe représente la caractéristique de sortie du contrôleur TILC flou d'ordre un. Nous avons utilisé la méthode intuitive pour déterminer la valeur de ces constantes. Nous avons fait plusieurs tests pour affiner la position des singletons afin d'obtenir des performances satisfaisantes.

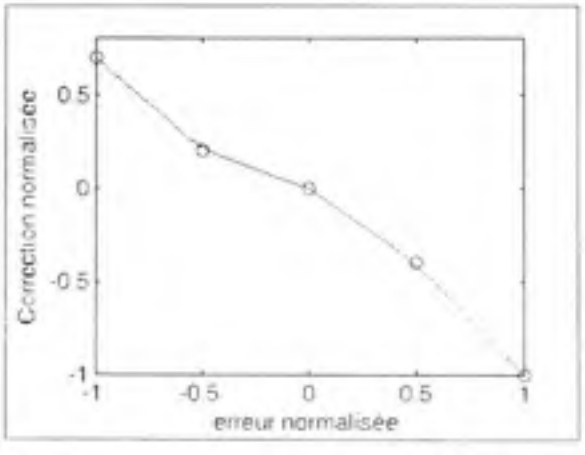

Figure 4.5 Surface de commande du TILC d'ordre un.

Puisque le maximum d'incertitude du module de découplage se situe à une valeur de consigne de 425°K, nous concentrerons nos tests sur cette valeur. Rappelons que les deux

températures désirées sont identiques. Nous savons également que le modèle mathématique n'est pas parfait. Nous ajoutons donc une perturbation pour valider la capacité du système à compenser le phénomène. Nous avons choisi de modifier la température initiale de la feuille de plastique. L'apprentissage a été effectué avec une température de 300°K. La figure 4.6 illustre l'évolution de l'erreur en fonction de l'itération. En deux itérations, nous arrivons à réduire l'écart à moins de 2°C.

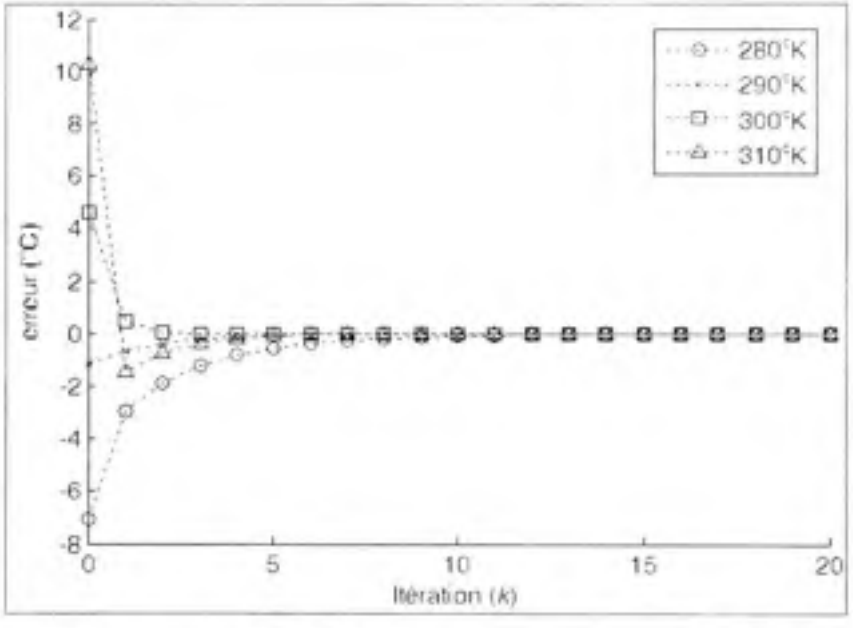

Figure 4.6 Évolution de l'erreur du TILC d'ordre un pour différentes températures initiales appliquées au cas 2x2 pour une consigne à 425°K.

La courbe correspondante à une température initiale de 310°K présente un dépassement. En effet, lorsque l'erreur est plus grande que zéro, notre contrôleur est plus agressif. Nous avons fait ce choix, car la surchauffe de la feuille n'est pas souhaitable en raison de la déflexion de cette dernière. Dans ce cas, la feuille pourrait toucher les éléments chauffant du bas. La figure 4.7 montre la convergence des consignes pour les quatre conditions d'opération. Nous remarquons que la convergence est monotone et rapide.

Les performances du contrôleur développé pour la configuration à deux éléments chauffant et à deux capteurs sont satisfaisantes. Maintenant, nous pouvons traiter un cas plus complexe.

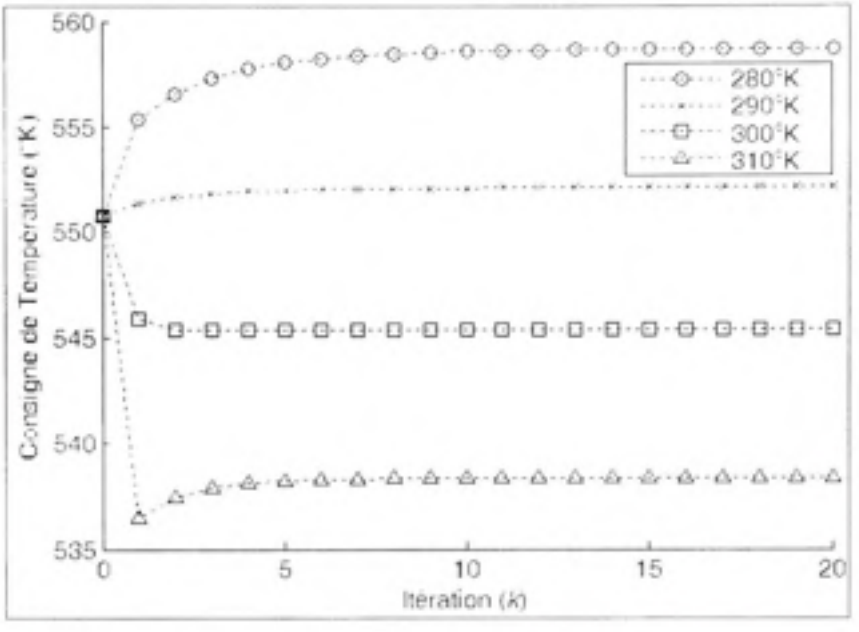

Figure 4.7 Consigne de température pour différentes températures initiales pour une consigne de 425°K appliquée au cas 2x2.

4.2 Configuration à quatre capteurs et à quatre éléments chauffant

La configuration à quatre capteurs et à quatre éléments chauffant nous permettra d'étudier le phénomène de symétrie. Cette seconde configuration consiste à effectuer les regroupements suivants :

1. T_1-T_2-T_3 IR_{T5};
2. T_4-T_5-T_6 IR_{T2};
3. B_1-B_2-B_3 IR_{B5};
4. B_4-B_5-B_6 IR_{B2}.

Dans cette section, nous présenterons le module de découplage disponible en deux versions et le module d'asservissement.

4.2.1 Modules de découplage

Pour cette configuration, nous avons développé deux modules de découplage. La première version respecte en tout point la méthode proposée. En d'autres termes, chacune des sorties possède une composante de découplage. Comme vous vous en doutez, l'apprentissage du module est beaucoup plus long, car l'ensemble de test est plus grand que pour le cas précédent. Donc, ce facteur limitatif nous restreint aux configurations inférieures à six capteurs et à six éléments chauffant. Pour pallier ce facteur, nous pouvons réduire la complexité du module de découplage en exploitant la symétrie dans le plan de la feuille de plastique. La seconde version exploite ce phénomène.

Version standard

La version standard est obtenue en appliquant à la lettre la méthode proposée. Pour illustrer les performances de ce module de découplage, nous avons évalué l'erreur de prédiction en utilisant le modèle de simulation. La figure 4.8 présente la distribution des erreurs et le tableau 4.2, quelques statistiques. Encore un fois, nous avons imposé les valeurs de consignes égales des deux côtés de la feuille de plastique. Toutefois, l'ensemble de test est constitué de toutes les paires situées entre 400°K et 450°K, par incrément de cinq, en incluant ces dernières valeurs. Les performances, pour l'ensemble de test, sont très satisfaisantes. En effet, 50% des consignes testées présentent une erreur inférieure à 1,5°C.

Tableau 4.2 Statistiques des erreurs de découplage de la configuration 4x4

no. de l'entrée	maximum (°C)	moyenne (°C)	médiane (°C)
1	5,689	1,982	1,509
2	5,689	1,982	1,509
3	5,689	1,982	1,509
4	5,689	1,982	1,509

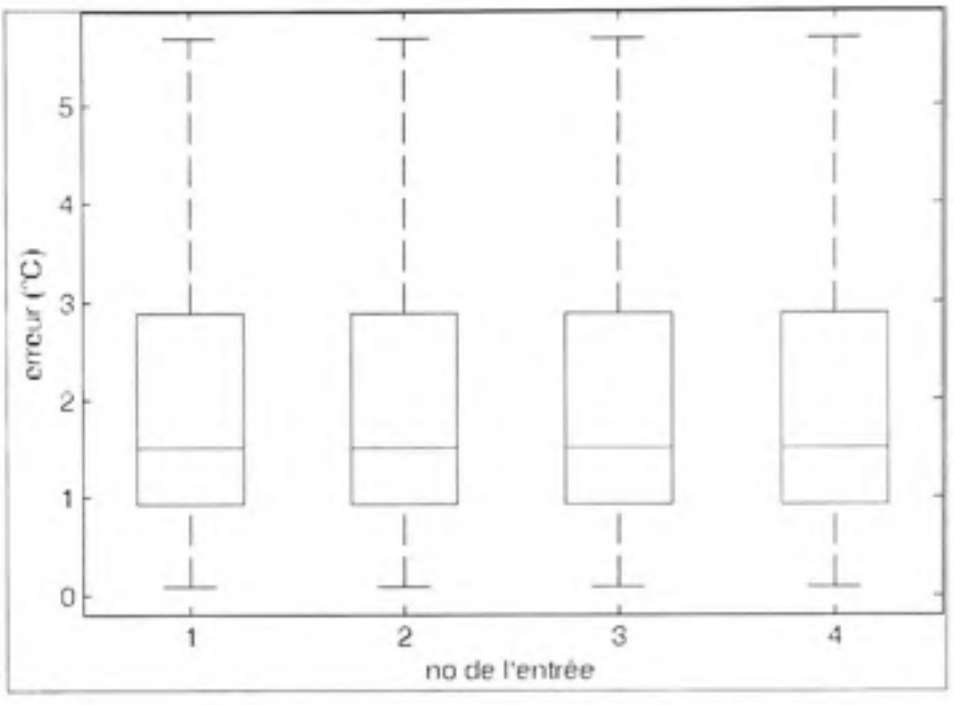

Figure 4.8 Distribution des erreurs de découplage de la configuration 4x4.

Nous avions établi antérieurement la présence de nombreuses symétries dans le modèle de simulation pour ce cas. Nous avons l'illustration de ce phénomène dans nos résultats. Donc, en position relative les composantes de découplage développées sont toutes identiques. Cette constatation nous conduit directement à la seconde version du module de découplage

Version symétrique

Jusqu'à maintenant, nous n'avons pas tenté de diminuer la complexité de notre module de découplage. Comme nous l'avons vu, il existe dans le modèle mathématique plusieurs symétries. Toutefois, il en existe une seule applicable à toutes les configurations. En effet, la symétrie dans le plan de la feuille est toujours existante. Donc, nous utiliserons cette dernière pour simplifier le module de découplage.

Pour atteindre cet objectif, nous devons modifier légèrement la méthode proposée. Grâce à la flexibilité de cette dernière, les changements sont mineurs. En effet, il ne s'agit que de

modifier la base de données destinée à l'apprentissage. Plus concrètement, nous imposons simplement la même température des deux cotées de la feuille, puisque le modèle est symétrique. La valeur mesurée des deux côtés de la feuille est identique. Donc, la base de données ne comporte que les consignes appliquées et les sorties correspondantes au dessus de la feuille. Par la suite, nous appliquons la même méthode pour poursuivre la conception du contrôleur. Le module de découplage résultant a seulement deux entrées et deux sorties. La figure 4.9 montre la nouvelle configuration adaptée. Vous remarquerez que le même module de découplage est utilisé deux fois.

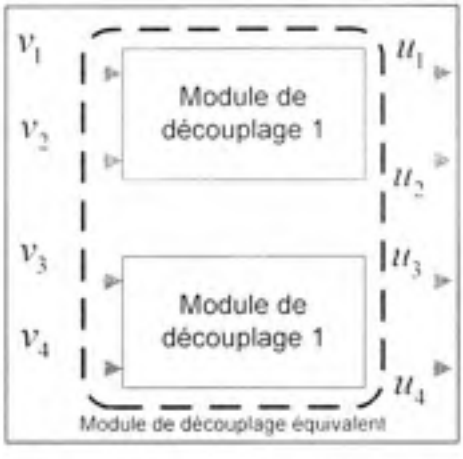

Figure 4.9 Schéma du module de découplage équivalent.

La figure 4.10 présente la distribution des erreurs et le tableau 4.3, quelques statistiques. Nous avons utilisé le même ensemble de test que la version standard.

Tableau 4.3 Statistiques des erreurs de découplage de la configuration 4x4 symétrique

no. de l'entrée	maximum (°C)	moyenne (°C)	médiane (°C)
1	3,492	1,482	1,275
2	3,492	1,482	1,275

Nous observons une amélioration des performances. Il faut toutefois relativiser ce résultat. En effet, l'ensemble d'apprentissage est très semblable à l'ensemble de test. Donc, nous

avons spécialisé notre module sur ce cas particulier. Toutefois, pour les cas où la température désirée diffère entre le dessus et le dessous de la feuille, nous nous attendons à une dégradation des performances. Ces cas seront observables en asservissement.

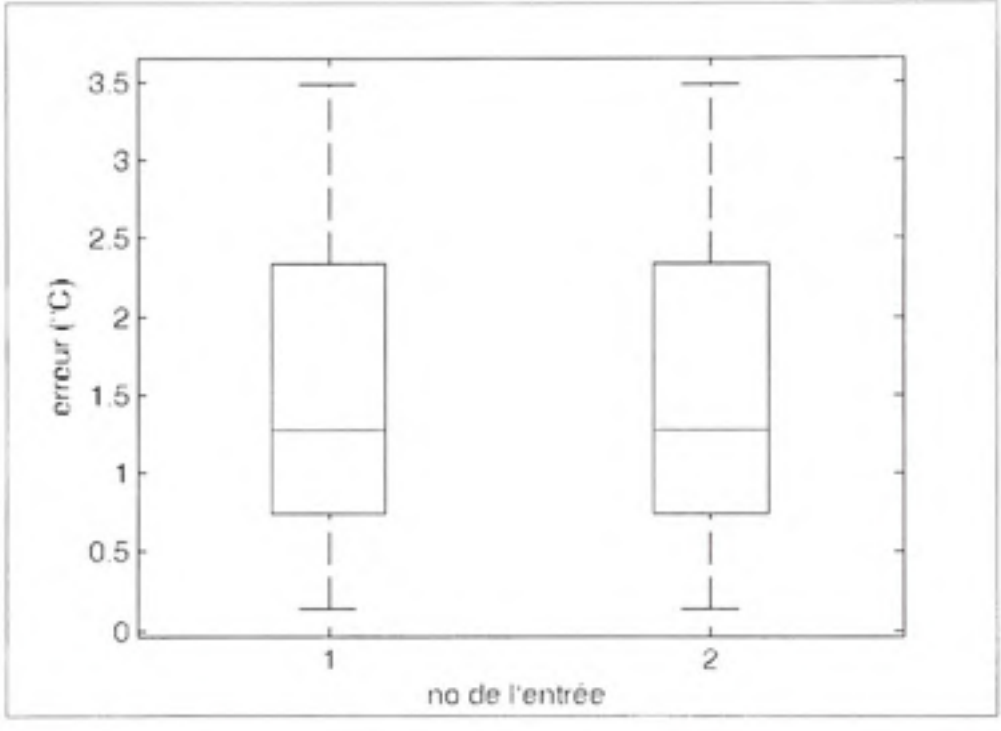

Figure 4.10 Distribution des erreurs de découplage de la configuration 4x4 symétrique.

Comparaison des deux versions

Les deux versions développées répondent au besoin quant à la prédiction de la consigne à appliquer pour obtenir le profil de chauffage désiré. Toutefois, nous avons relevé que la version symétrique est spécialisée sur un problème précis. De plus, le four réel n'est pas parfaitement symétrique. Malgré tout, cette dernière option demeure intéressante. Le tableau 4.4 présente le temps de calcul et le nombre de points dans la base de données nécessaire au développement des deux solutions.

Tableau 4.4 Comparaison de la version standard et de la version symétrique

Version	Temps de développement (s)	Nombre de points
Standard	45.576	256
Symétrique	2.858	16

Comme vous le remarquez, le coût de calcul de la version symétrique est beaucoup plus faible à cause du nombre réduit d'entrées du système d'inférence flou et de la petite taille de la base de données d'apprentissage. Pour cette raison, nous préférerons, pour les configurations plus complexes, la version symétrique malgré ces lacunes. En effet, nous croyons que le module d'asservissement récupèrera les erreurs d'approximations causées par cette version. La prochaine section nous permettra de vérifier cette hypothèse expérimentalement.

4.2.2 Asservissement

Pour éliminer les effets des perturbations et réduire les erreurs, nous proposons d'utiliser une commande TILC d'ordre un. Puisque le système est découplé, la commande à réaliser est semblable à la configuration à deux capteurs et à deux éléments chauffant. Nous avons seulement à ajouter deux boucles de régulation. Donc, nous utiliserons le même module d'asservissement pour cette configuration. Dans cette section, nous présenterons les résultats de ce module associé avec les deux différentes versions du module de découplage.

Version standard

Pour vérifier la capacité du contrôleur, nous l'avons soumis à différentes températures initiales. La température désirée demandée aux quatre points est identique. Nous posons cette température à 425°K. La figure 4.11 présente l'évolution de l'erreur et la figure 4.12 montre l'évolution de la consigne en amont du module de découplage.

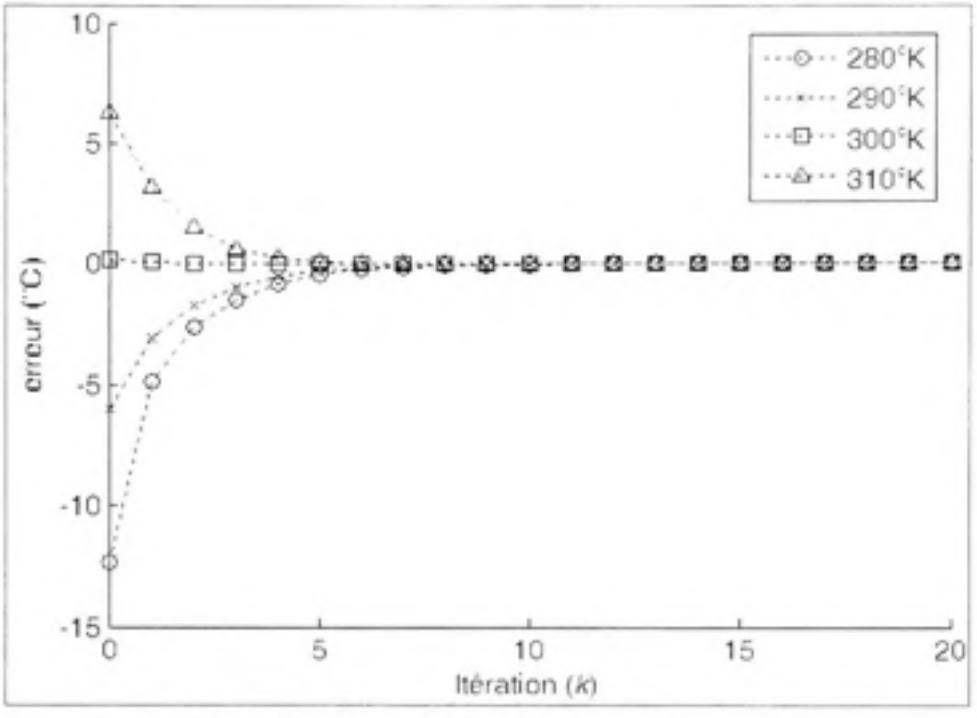

Figure 4.11 Évolution de l'erreur du TILC d'ordre un pour différentes températures initiales appliquées au cas 4x4 pour une consigne à 425°K.

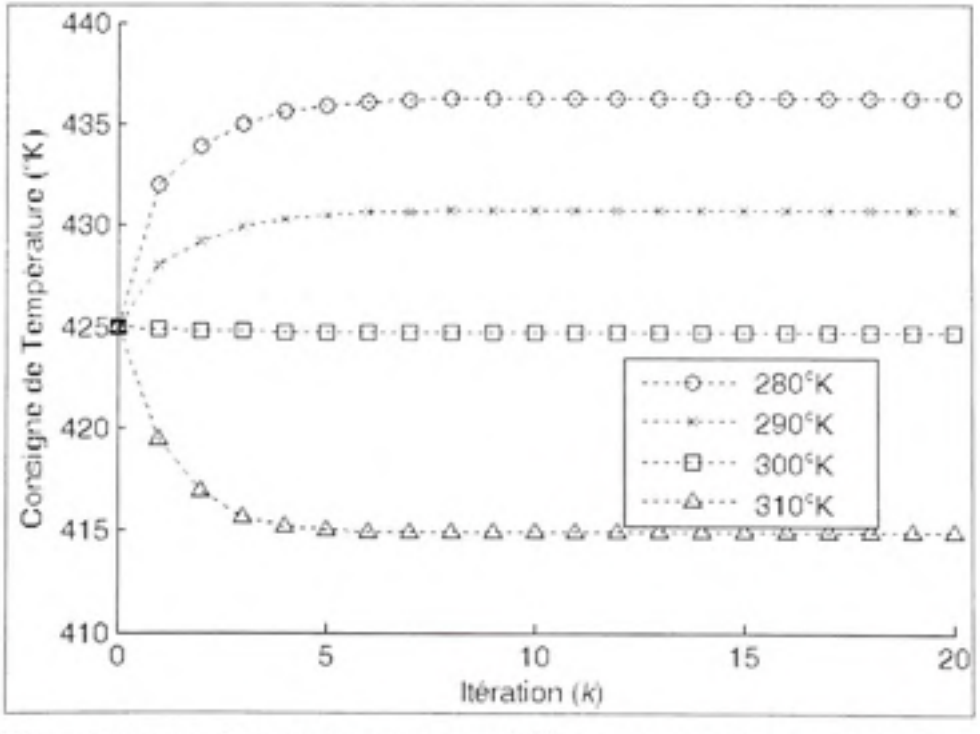

Figure 4.12 Consigne de température pour différentes températures initiales pour une consigne de 425°K appliquée au cas 4x4.

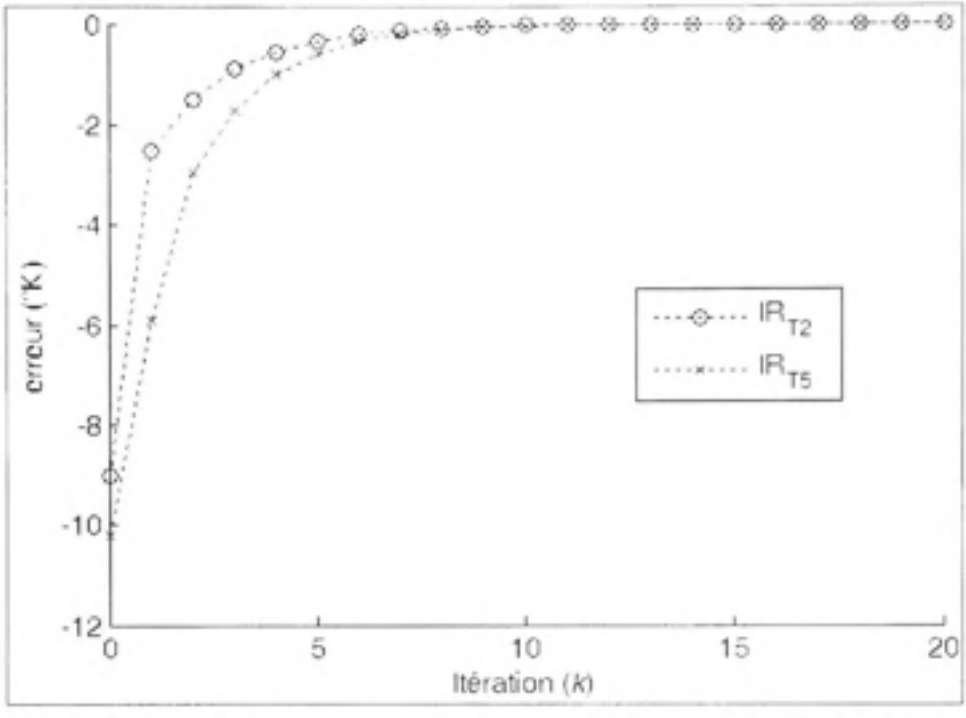

Figure 4.13 Évolution de l'erreur pour une température initiale de 280°K appliquée au cas 4x4 pour une consigne à 410°K et 440°K.

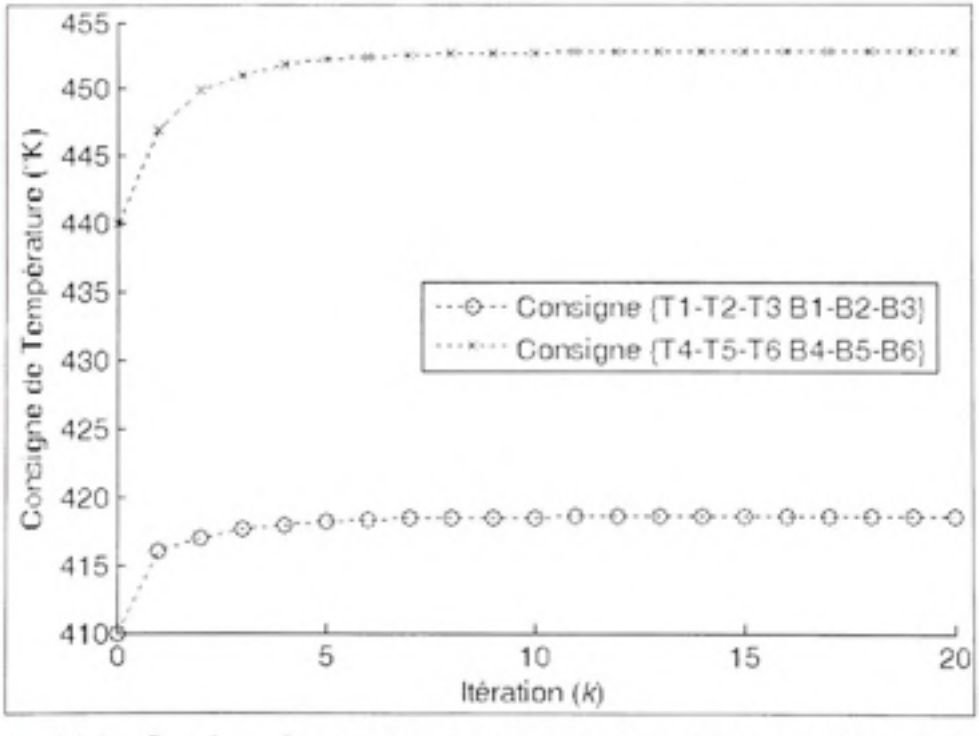

Figure 4.14 Consigne de température pour une température initiale de 280°K appliquée au cas 4x4 pour une consigne à 410°K et 440°K.

Veuillez noter que la figure 4.12 illustre la consigne d'un seul groupe d'éléments chauffant. En effet, le modèle de simulation étant symétrique, toutes les consignes sont identiques pour cette configuration.

Jusqu'à maintenant, nous avons demandé un patron de chauffage simple. La figure 4.13 présente l'évolution de l'erreur pour une température désirée sur IR_{T2}-IR_{B2} de 410°K et 440°K sur IR_{T5}-IR_{B5}. La température initiale de la feuille est fixée à 280°K pour ajouter une perturbation. Cette fois, il y a deux courbes puisque les températures demandées sont différentes. La figure 4.14 montre l'évolution de la commande en amont du module de découplage.

La version standard du module de découplage associée à une commande TILC d'ordre un nous permet de réduire l'erreur sur le profil de température en moins de deux itérations à l'intérieur d'une bande de 3°C. Vérifions maintenant les performances de la version symétrique.

Version symétrique

Pour vérifier la capacité du contrôleur, nous l'avons soumis à différentes températures initiales. La température désirée demandée aux quatre points est identique. Nous posons cette température à 425°K. La figure 4.15 présente l'évolution de l'erreur et la figure 4.16 montre l'évolution de la consigne en amont du module de découplage.

Veuillez noter que la figure 4.16 illustre la consigne d'un seul groupe d'éléments chauffant. En effet, le modèle de simulation étant symétrique, toutes les consignes sont identiques pour cette configuration.

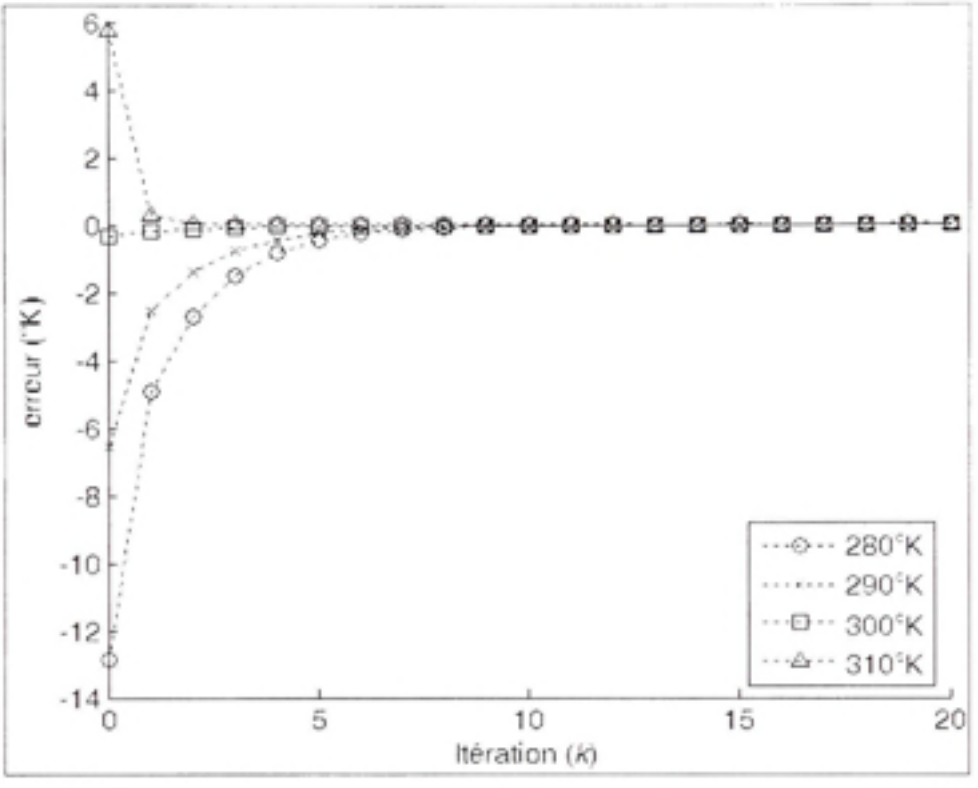

Figure 4.15 Évolution de l'erreur pour une consigne à 425°K (version symétrique).

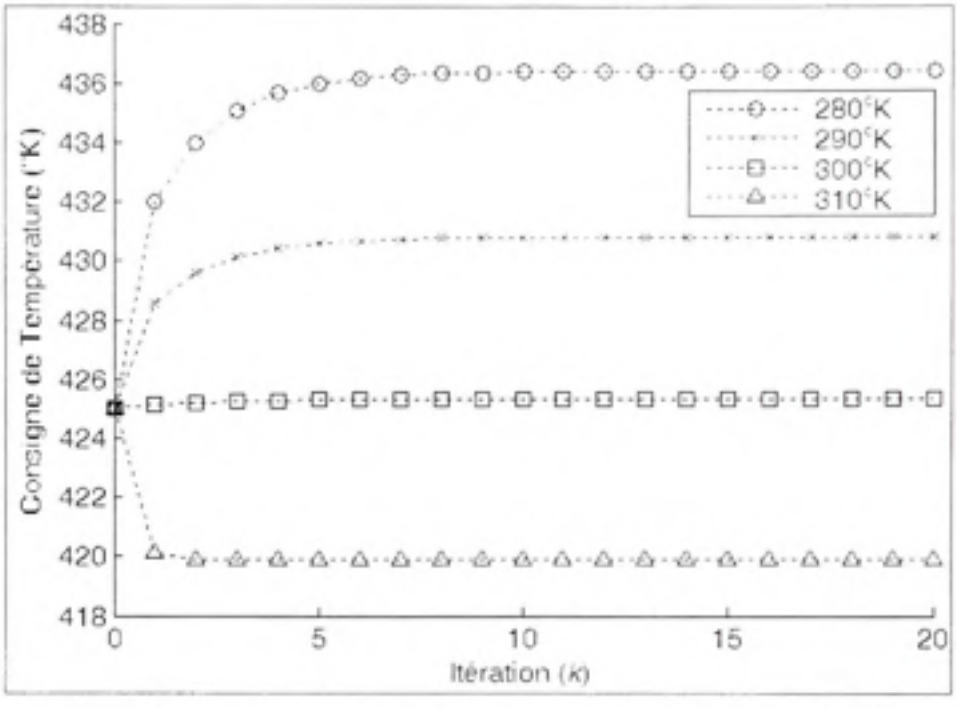

Figure 4.16 Consigne de température pour une consigne à 425°K (version symétrique).

Jusqu'à maintenant, nous avons demandé un patron de chauffage simple. La figure 4.17 présente l'évolution de l'erreur pour une température désirée sur IR_{T2}-IR_{B2} de 410°K et 440°K sur IR_{T5}-IR_{B5}. La température initiale de la feuille est fixée à 280°K pour ajouter une perturbation. Cette fois, il y a deux courbes puisque les températures demandées sont différentes. La figure 4.18 montre l'évolution de la commande en amont du module de découplage.

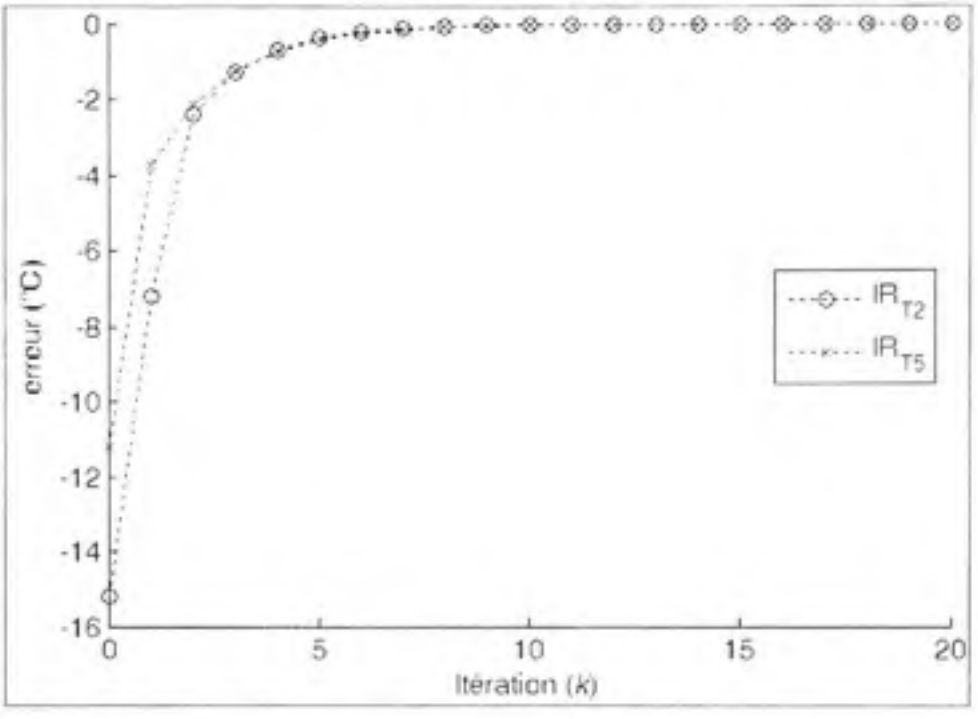

Figure 4.17 Évolution de l'erreur pour une consigne à 410°K et 440°K (version symétrique).

Jusqu'à maintenant, la version symétrique semble plus performante que la version standard. En effet, l'erreur après deux itérations est à environ 2,5°C. Toutefois, nous avions déterminé plus tôt que notre module de découplage perdrait de la performance dans le cas où la température sous la feuille serait différente de celle du dessus. Pour effectuer ce test, nous avons fait une modification sur le modèle du four de thermoformage. Nous avons tout simplement ajouté un décalage à toutes les lectures de températures situées sous la feuille. Nous avons fixé ce décalage à 5°C. Ainsi, nous pouvons toujours demander le même profil de température que le test précédant. Ce décalage nous permet également de briser la symétrie existante dans le modèle du four pour vérifier la robustesse de notre contrôleur au variation du modèle. Les figures 4.19 et 4.20 illustrent respectivement l'évolution de l'erreur et l'évolution de la consigne en amont du module de découplage. Dans ces conditions, les performances se sont dégradées. En effet, en quatre itérations, l'erreur est inférieure à 3°C. Donc, nous avons besoin de deux itérations supplémentaires pour obtenir le même résultat que la version standard. Toutefois, le développement de cette solution est beaucoup plus

rapide. Elle nous permet de traiter des configurations beaucoup plus complexes. Alors, nous privilégierons cette approche pour le prochain module de découplage.

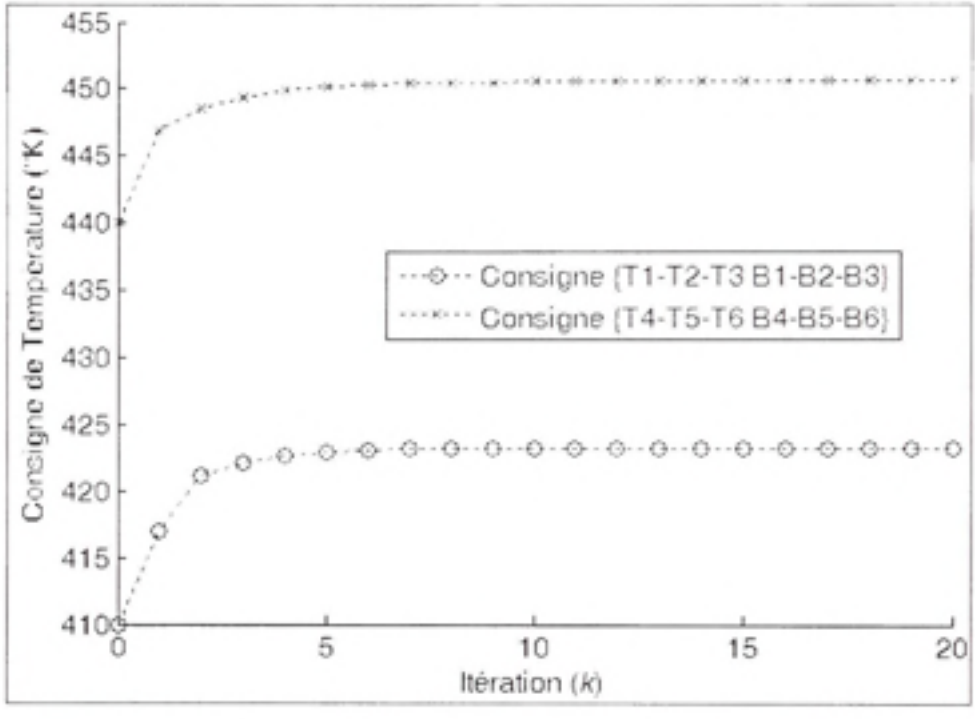

Figure 4.18 Consigne de température pour une consigne à 410°K et 440°K (version symétrique).

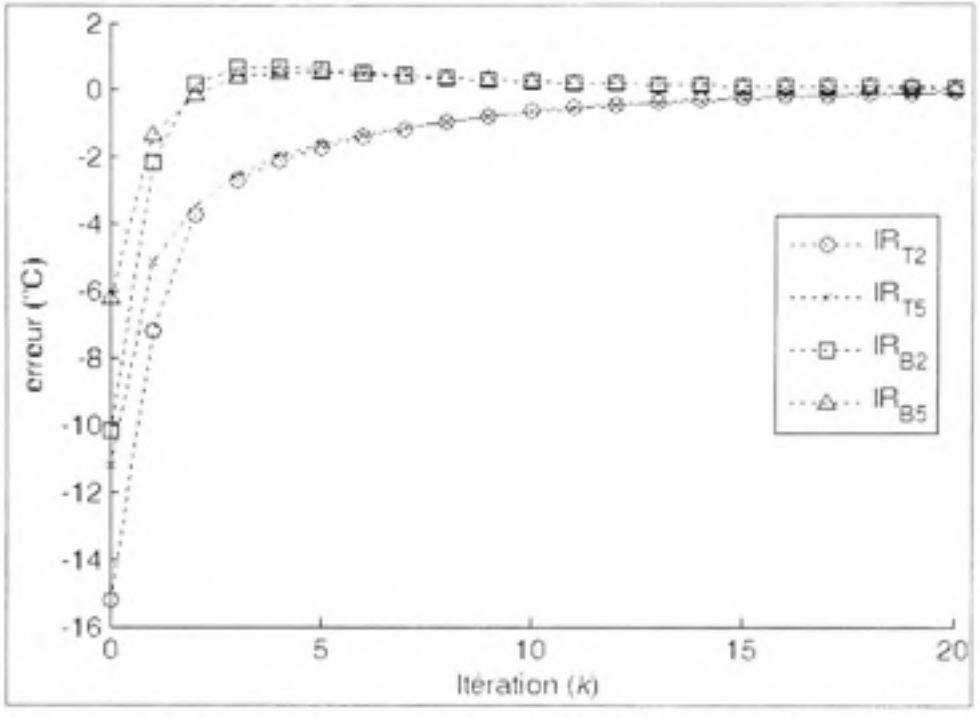

Figure 4.19 Évolution de l'erreur pour une consigne à 410°K et 440°K (version symétrique et asymétrie du modèle).

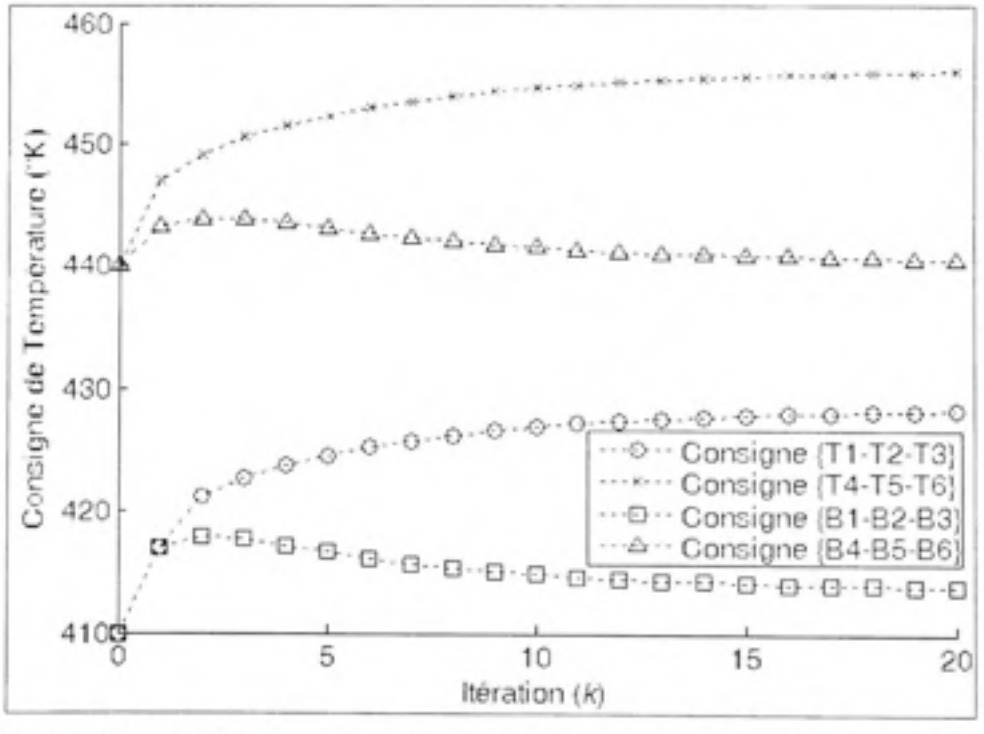

Figure 4.20 Consigne de température pour une consigne à 410°K et 440°K (version symétrique et asymétrie du modèle).

4.3 Configuration à douze capteurs et à douze éléments chauffants

La configuration à douze capteurs et à douze éléments chauffant est la commande la plus complexe réalisée dans le cadre de nos travaux. Plus précisément, nous associons de la manière suivante les capteurs aux éléments chauffants :

1. T_1-IR_{T7};
2. T_2-IR_{T6};
3. T_3-IR_{T5};
4. T_4-IR_{T2};
5. T_5-IR_{T3};
6. T_6-IR_{T4}.

Le même agencement est utilisé pour les capteurs et les éléments chauffants situés sous la feuille de plastique. Dans cette section, nous présenterons les résultats relatifs au module de découplage en version symétrique et le module de commande TILC.

4.3.1 Module de découplage

Pour la réalisation du module de découplage nous avons suivi la méthode proposée en tenant compte de la symétrie dans le modèle pour accélérer le processus de développement. Toutefois, nous avons dû faire quelques modifications. Plus précisément, ces modifications touchent les fonctions situées dans le conséquent du système d'inférence flou. Par le passé, cette fonction linéaire comportait un coefficient constant. Toutefois, pour le cas qui nous intéresse, la présence de ce coefficient génère un problème mal conditionné qui rend l'inversion instable. Pour mesurer le conditionnement, il existe une mesure basée sur les valeurs propres. La première étape est d'écrire le système d'équation sous la forme matricielle suivante :

$$Y = A \bullet X \tag{4.1}$$

Par la suite, nous déterminons les valeurs singulières minimale et maximale de la matrice A. Le calcul suivant permet d'obtenir la mesure $\kappa(A)$ nommée « condition number » :

$$\kappa(A) = \frac{\sigma_{max}(A)}{\sigma_{min}(A)} \tag{4.2}$$

Par exemple, si nous n'avions pas effectué la modification des fonctions linéaires, $\kappa(A)$ serait de l'ordre de 10^{15} pour les pires systèmes à inverser. Si nous effectuons la modification, nous observons en générale une diminution de l'ordre de 10^2. Cette amélioration n'est toujours pas suffisante. En effet, un problème bien conditionné aurait une valeur de $\kappa(A)$ près de un. Tout comme Gauthier (2009, p.168), nous éliminons les valeurs singulières près de zéro. Plus précisément, toutes les valeurs cent fois inférieures à la valeur singulières la plus grande sont forcées à zéro. Par contre, le rang de la matrice A devient déficient. Alors, nous utilisons le pseudo inverse de Moore-Penrose. En effectuant ces modifications lors de la phase d'inversion, nous parvenons à stabiliser le problème en contrôlant sa sensibilité.

Ces modifications faites, nous avons testé le module de découplage. L'ensemble de test est constitué de toutes les combinaisons de 420°K, 430°K, 440°K sur chacune des six entrées correspondant aux éléments du dessus. Le même patron de chauffage est appliqué au dessous. Le tableau 4.5 et la figure 4.21 montrent les résultats de ce test.

Tableau 4.5 Statistiques des erreurs de découplage (cas 12x12 symétrique)

no. de l'entrée	maximum (°C)	moyenne (°C)	médiane (°C)
1	135,124	7,740	4,253
2	133,774	6,671	3,277
3	133,775	6,762	3,372
4	133,776	6,715	3,046
5	133,777	7,501	3,285
6	135,124	5,734	2,521

Comme vous le remarquez, l'erreur maximum n'est pas acceptable. Toutefois, cette configuration étant complexe, les cas testés ne sont pas tous réalisables. Les points représentés par le symbole de l'addition sur la figure 4.21 sont certainement infaisables. Nous avons repris le test avec un ensemble plus restreint.

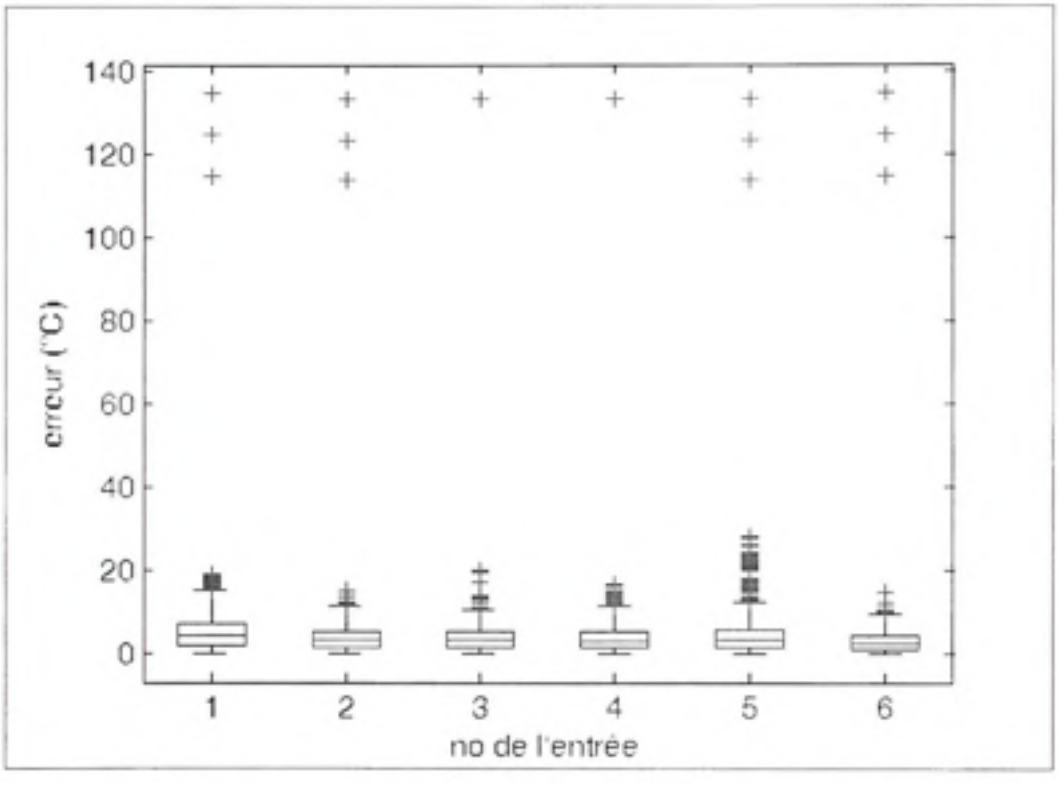

Figure 4.21 Distribution des erreurs de découplage de la configuration 12x12 symétrique.

Le second ensemble de test est constitué de toutes les combinaisons de 420°K, 425°K et 430°K sur chacune des six entrées correspondant aux éléments du dessus. La présente la distribution des erreurs pour cet ensemble.

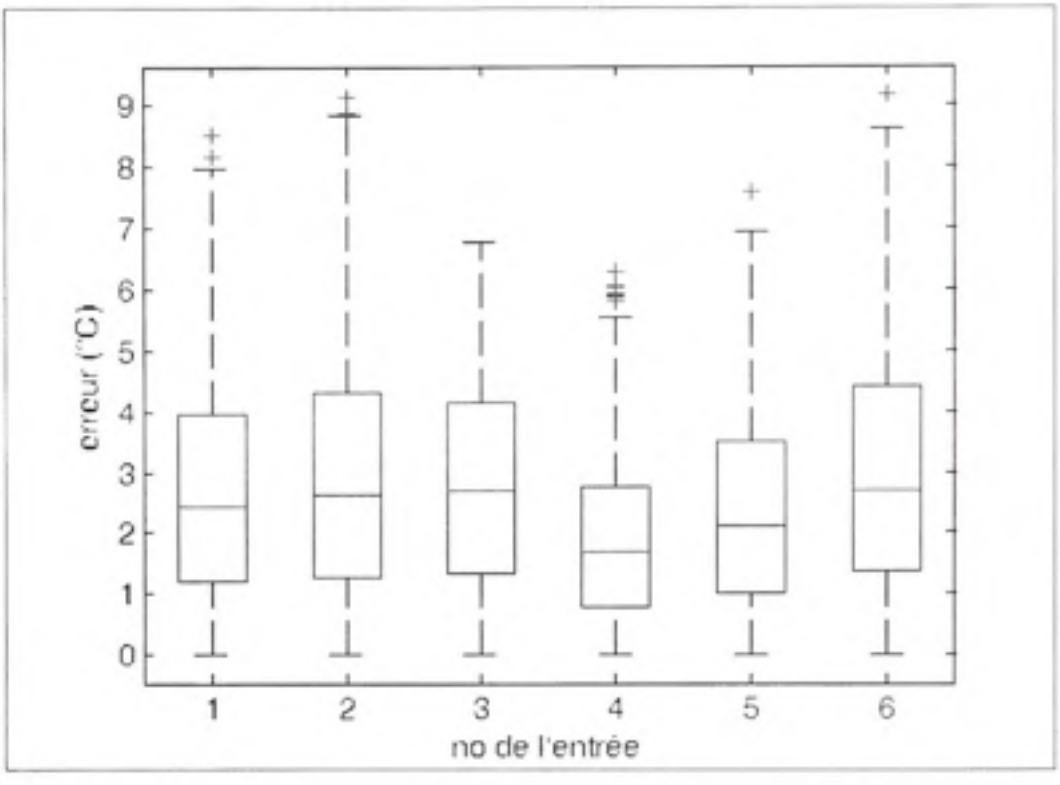

Figure 4.22 Distribution des erreurs de découplage de la configuration 12x12 symétrique (ensemble de test restreint).

Le tableau 4.6 fournit la taille de la base de données nécessaire à l'apprentissage du système d'inférence flou ainsi que le temps de calcul requis pour le développement de la solution.

Tableau 4.6 Temps de calcul pour le développement du module de découplage

Version	Temps de développement (s)	Nombre de points
Symétrique	994.403	4096

Comme vous le voyez, les performances sont meilleures pour cet ensemble : 50% des cas testés ont une erreur inférieure à 3°C. Nous comptons toutefois sur le module de commande pour réduire l'erreur des pires cas.

4.3.2 Asservissement

Pour éliminer les effets des perturbations et réduire les erreurs, nous proposons d'utiliser une commande TILC. Comme pour la configuration à quatre capteurs et à quatre groupes d'éléments chauffant, nous proposons tout d'abord d'utiliser le même contrôleur d'ordre un. Par la suite, nous augmenterons l'ordre pour améliorer la robustesse du module de commande.

TILC d'ordre un

La première tentative de commande utilise une commande TILC d'ordre un présenté à la section se rapportant à la configuration à deux capteurs. Puisque le nombre de sorties est élevé, nous utiliserons une nouvelle mesure pour représenter l'évolution de l'erreur. Nous avons constitué un vecteur contenant les erreurs de toutes les sorties pour toutes les itérations. Puis, nous prenons la norme infinie de ces vecteurs pour tracer le graphique de la figure 4.23.

Lors de ce test, les températures désirées sont toutes posées à 425°K et la température initiale est de 280°K. Malheureusement, pour ce cas, l'erreur n'est pas bornée à l'intérieur d'une bande de 5°C. Tentons malgré tout le même test avec la version asymétrique du four. La figure 4.24 illustre le comportement de l'erreur dans ces conditions. La figure 4.25 illustre la commande appliquée en amont du module de découplage.

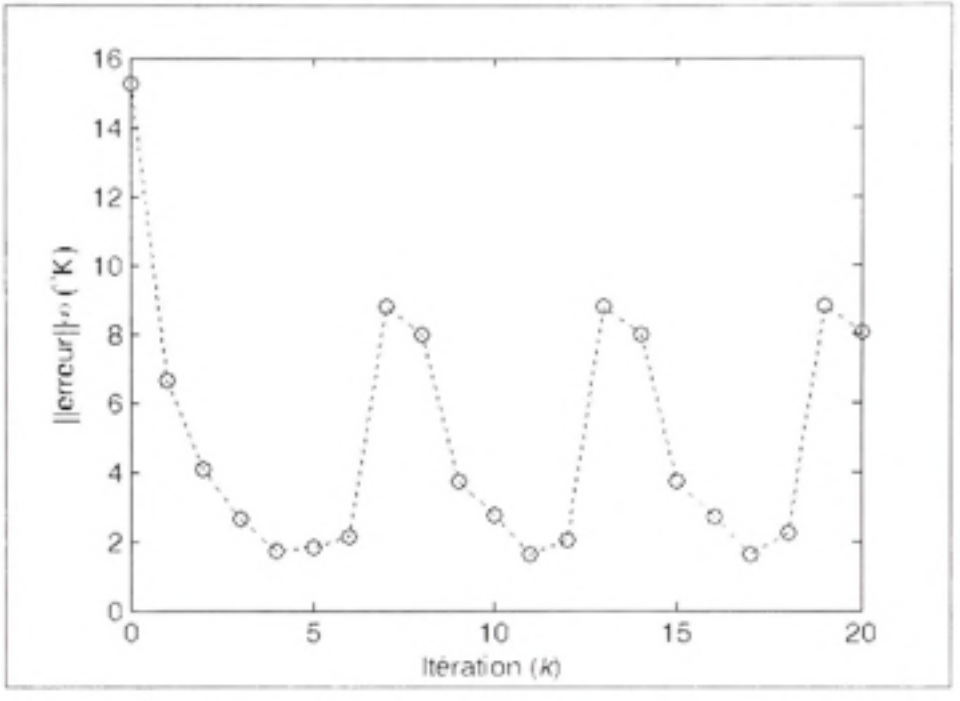

Figure 4.23 Évolution de l'erreur maximale du TILC d'ordre un (cas 12x12).

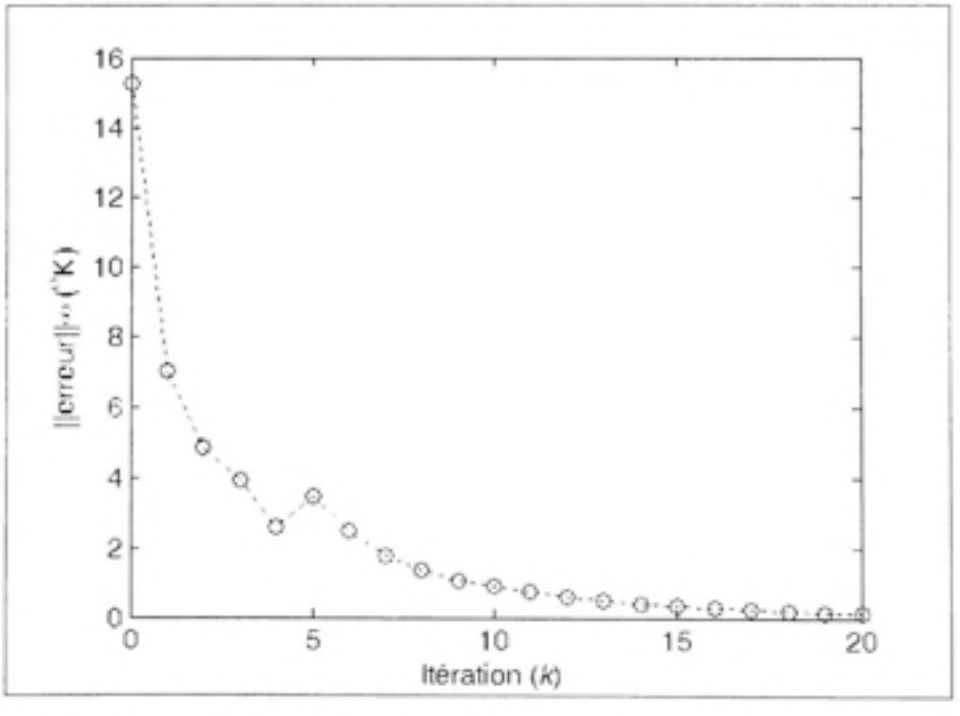

Figure 4.24 Évolution de l'erreur maximale du TILC d'ordre un (cas 12x12 asymétrique).

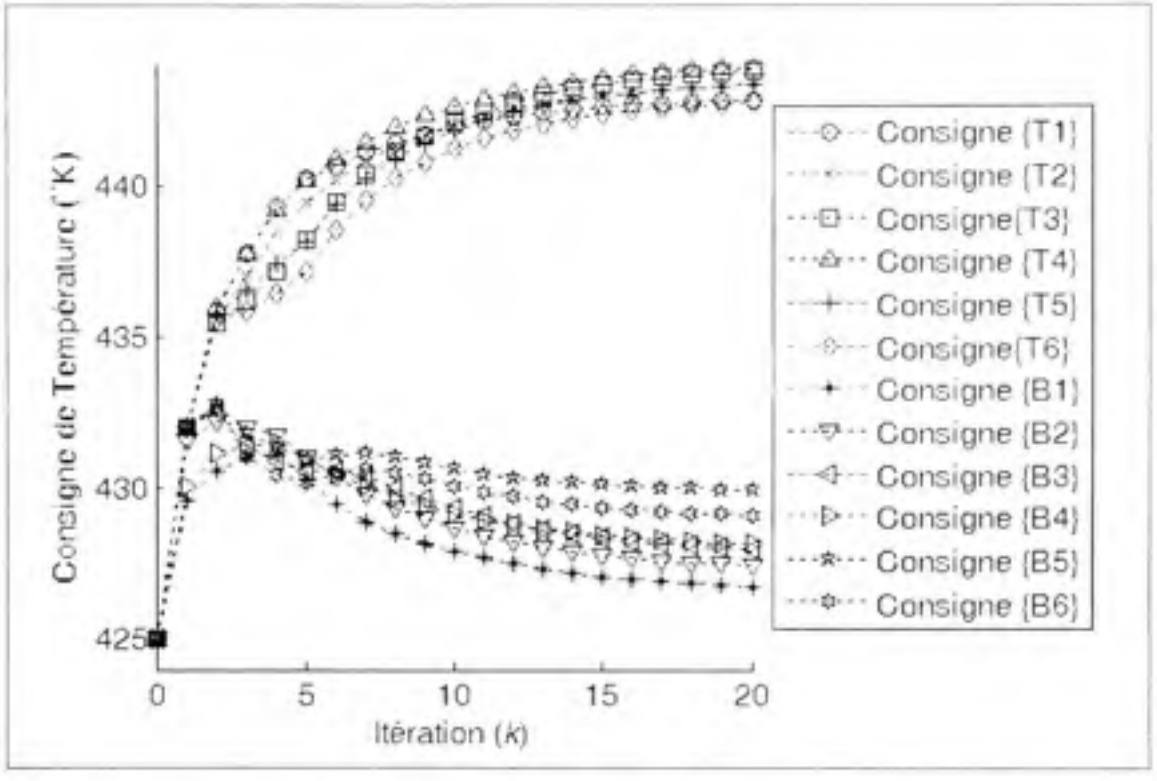

Figure 4.25 Consigne de température du TILC d'ordre un (cas 12x12 asymétrique).

Cette fois, la solution est stable. Nous avons tenté de stabiliser en modifiant la position des singletons du système d'inférence flou sans succès. Pour améliorer la robustesse, nous suggérons d'utiliser une commande TILC du second ordre.

TILC du second ordre

Nous avons utilisé un algorithme génétique pour déterminer la position des singletons de sortie du système d'inférence flou. L'ensemble d'apprentissage est constitué de différentes températures initiales (280°K, 290°K, 300°K et 310°K), mais d'un unique patron de chauffage. La température désirée est fixée à tous les points à 425°K. La figure 4.26 présente la surface de commande du système d'inférence obtenue. Nous utiliserons cette commande pour toutes les boucles SISO du système. La figure 4.27 montre l'évolution de la meilleure solution aux différentes itérations de l'algorithme d'apprentissage.

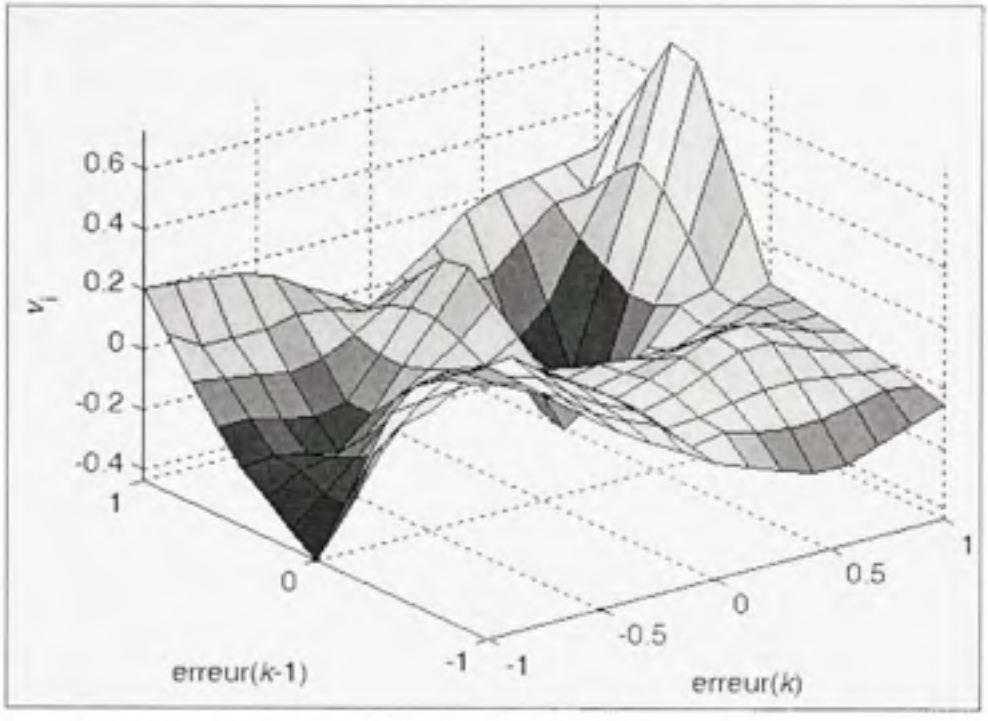

Figure 4.26 surface de commande du TILC d'ordre deux.

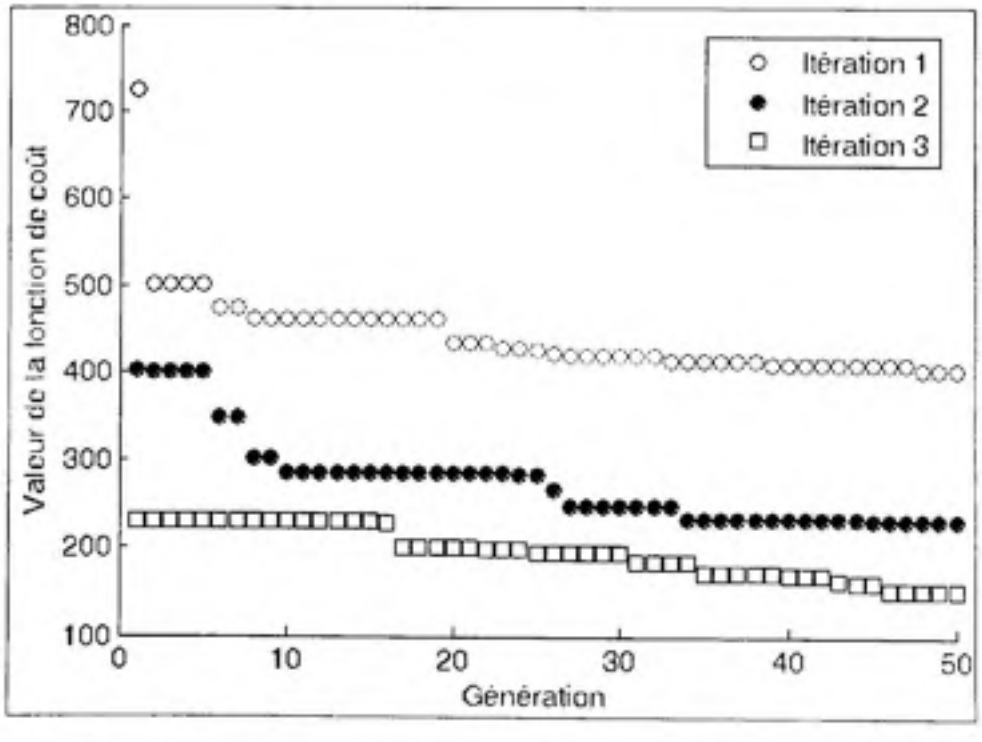

Figure 4.27 Évolution du meilleur individu de la population.

Nous avons répété les expériences de la section précédente. Nous avons fixé toutes les consignes à 425°K et défini la température initiale à 280°K. La figure 4.28 présente l'évolution de l'erreur pour le modèle symétrique du four et la figure 4.29 illustre les commandes appliquées en amont du module de découplage. Cette fois, l'erreur converge à l'intérieur d'une zone de 3°C en six cycles.

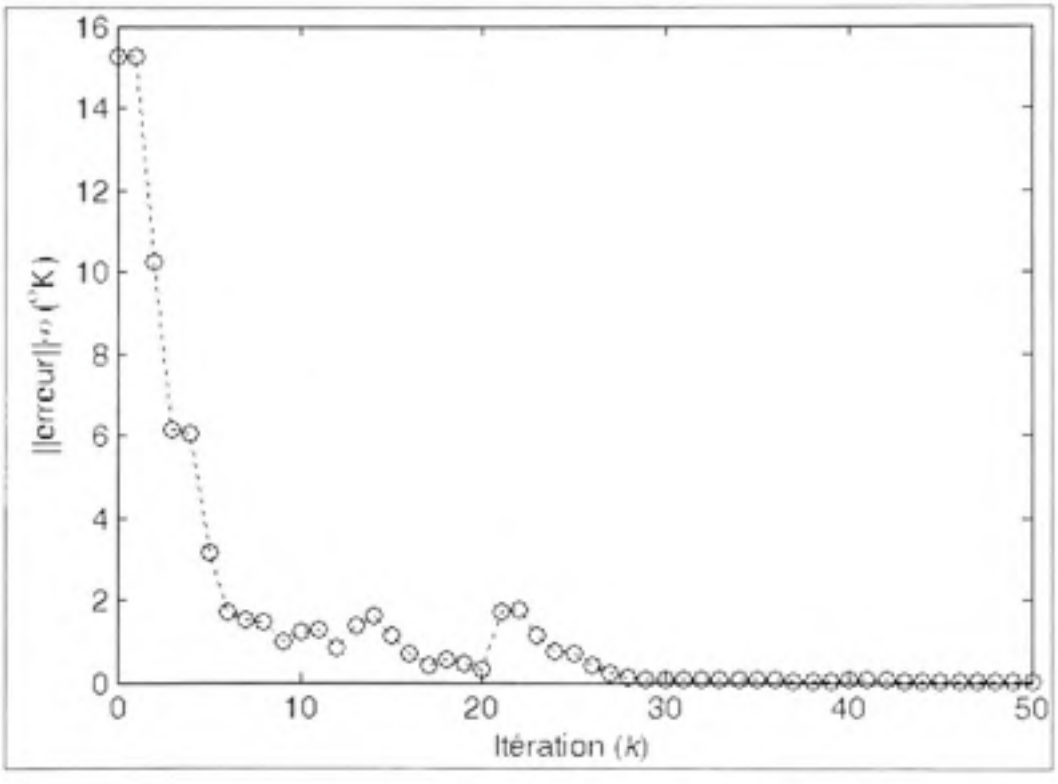

Figure 4.28 Évolution de l'erreur du TILC d'ordre deux (cas 12x12 symétrique).

Toutefois, le module de découplage est moins efficace que pour les autres configurations. En effet, sur la figure 4.29, on remarque beaucoup plus d'interaction entre les consignes. Maintenant, répétons l'expérience en introduisant l'asymétrie dans les lectures de température. Les figures 4.30 et 4.31 montrent respectivement l'évolution de l'erreur et les commandes appliquées.

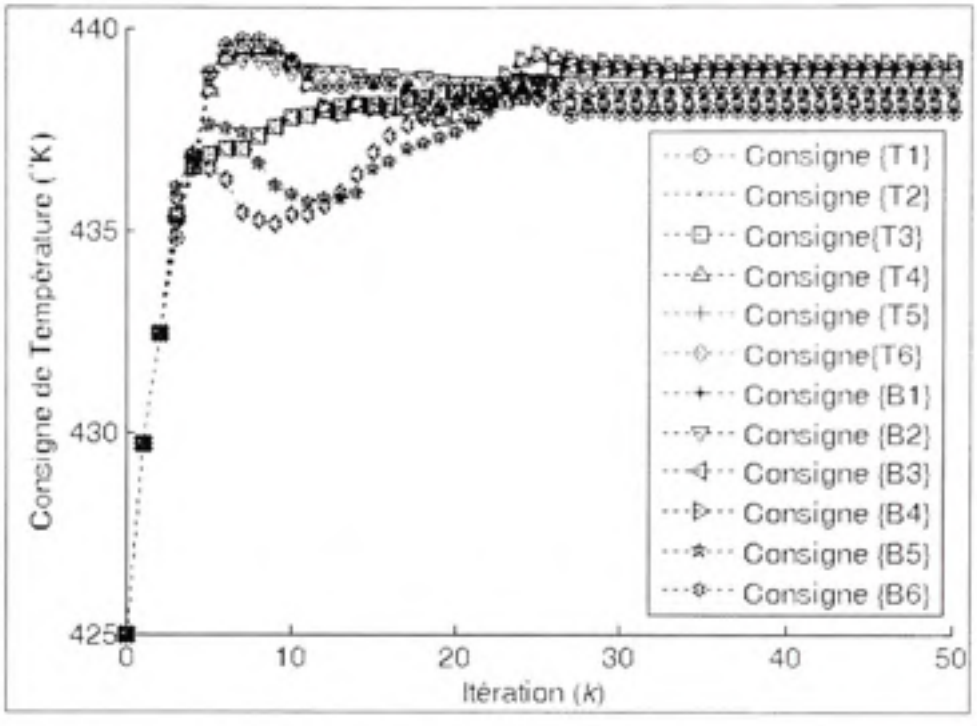

Figure 4.29 Consigne de température du TILC d'ordre deux (cas 12x12 symétrique).

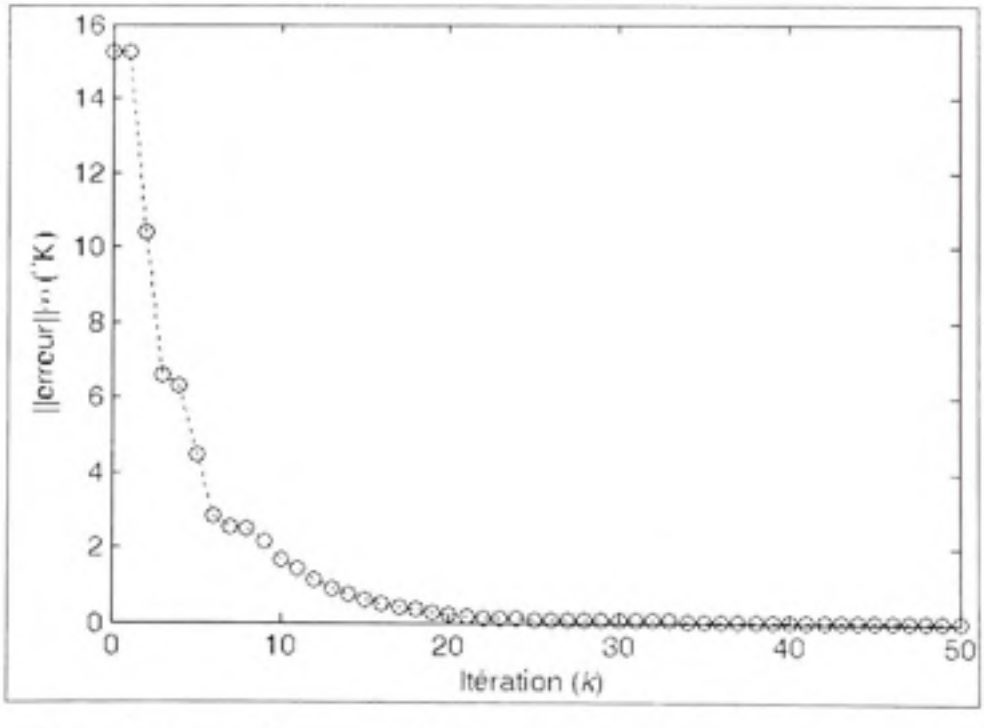

Figure 4.30 Évolution de l'erreur du TILC d'ordre deux (cas 12x12 asymétrique).

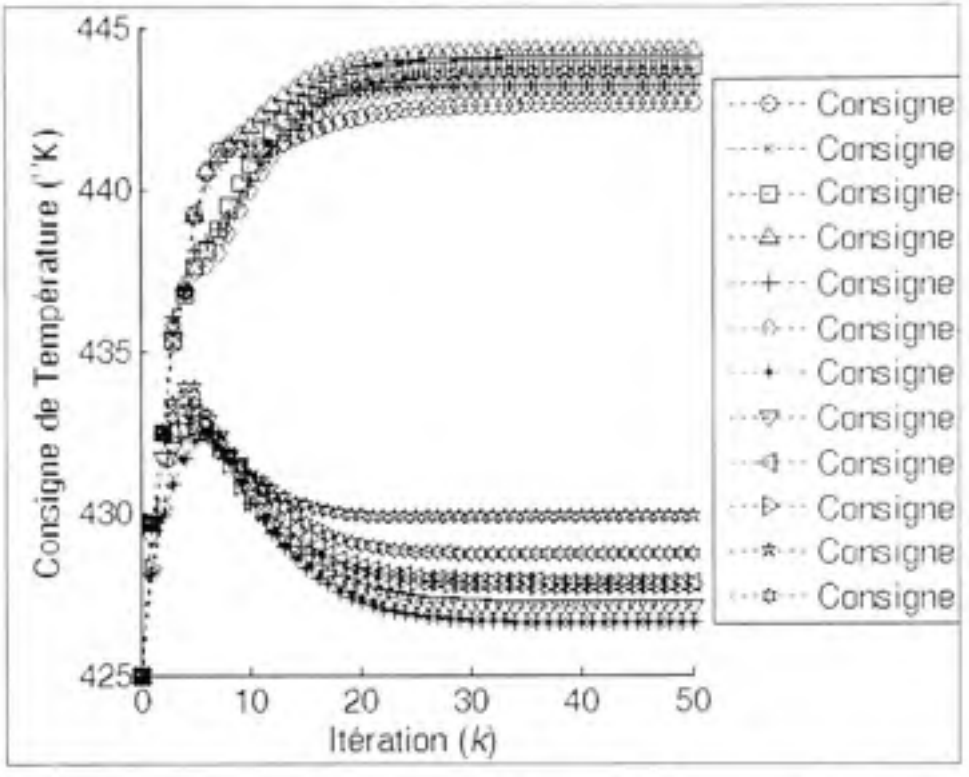

Figure 4.31 Consigne de température du TILC d'ordre deux (cas 12x12 asymétrique).

Pour nous rassurer quant à la stabilité de la commande, nous avons fait d'autres tests. La figure 4.32 présente l'évolution de l'erreur pour différentes températures initiales en conservant toujours les températures désirées à 425°K et en utilisant le modèle original. En d'autres termes, nous utilisons les mêmes conditions qu'à l'apprentissage du module. À la figure 4.33, nous avons repris le test précédent, mais cette fois, nous avons inclus l'asymétrie pour valider la robustesse de la commande. Veuillez noter que pour une température initiale de 310°K, la solution ne converge pas. La figure 4.34 montre les résultats du dernier test. Les températures initiales sont 285°K, 295°K et 300°K. De plus, nous avons demandé une température de 425°K sauf sur les zones mesurées par IR_{T7} et IR_{B7}. Nous avons fixé, pour ces deux localisations, la température désirée à 430°K.

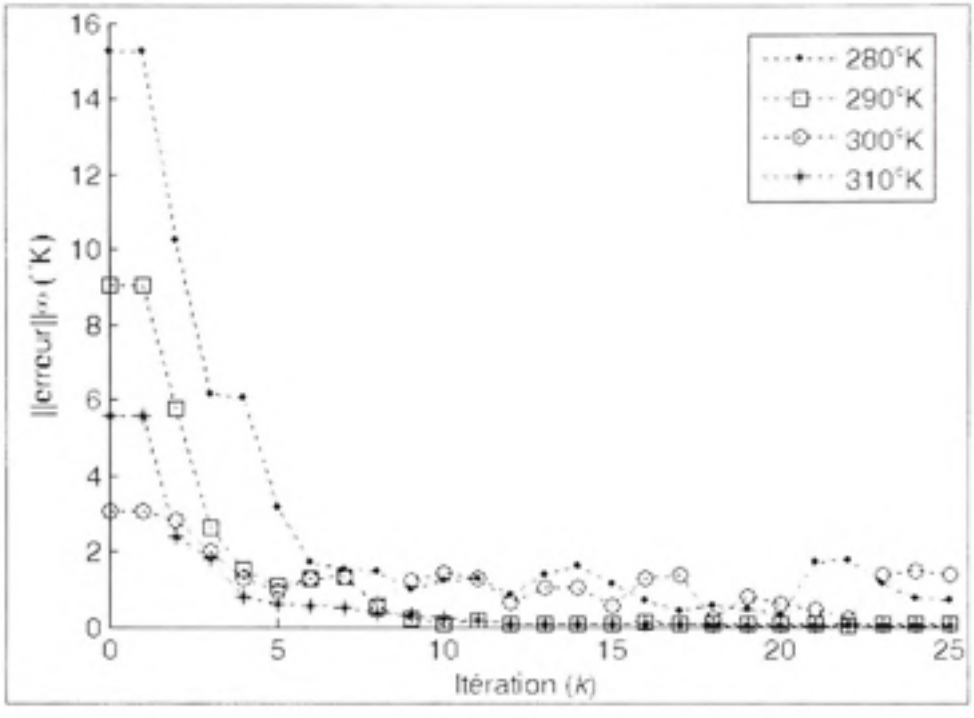

Figure 4.32 Erreur pour différentes températures initiales (cas 12x12).

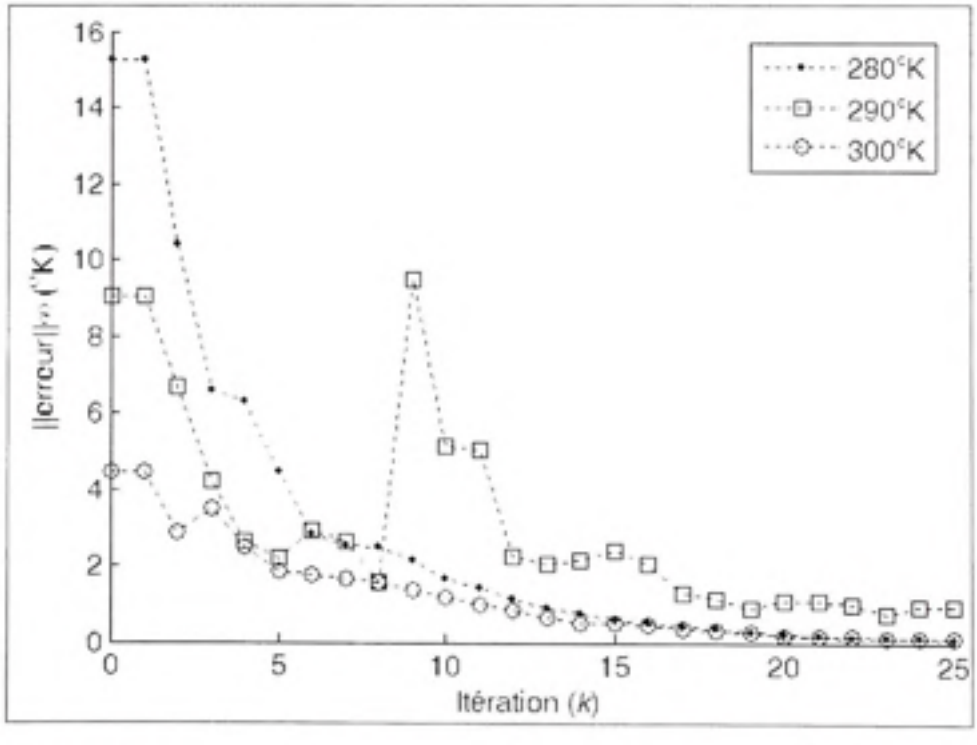

Figure 4.33 Erreur pour différentes températures initiales (cas 12x12 asymétrique).

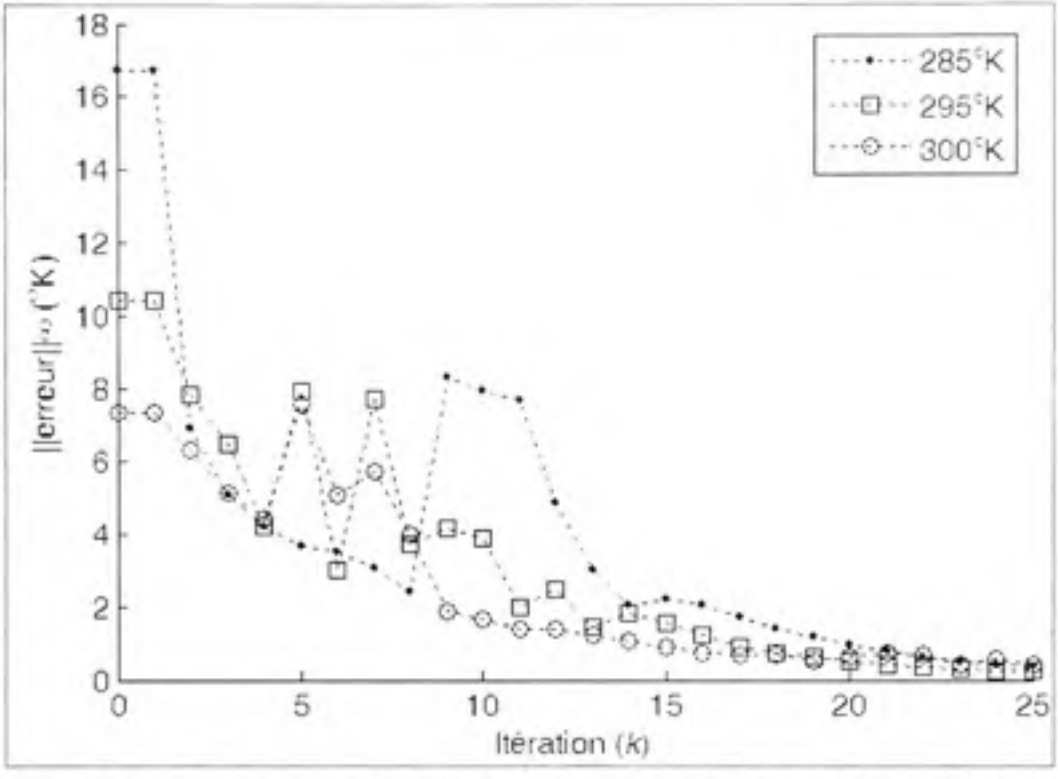

Figure 4.34 Erreur pour différentes températures initiales avec le profil modifié.

4.4 Analyse des résultats

Dans cette section, nous nous attarderons plus spécifiquement à la configuration à douze capteurs et à douze groupes d'éléments chauffant. En effet, avant cette configuration, tous les résultats obtenus étaient conformes à nos attentes. Ces configurations sont robustes à de fortes perturbations sur la température initiale de la feuille de plastique. De plus, elles sont également capables de corriger l'erreur due à la variation du modèle en utilisant une commande TILC d'ordre un.

Toutefois, ces constatations changent lors de la tentative de contrôler la configuration à douze capteurs. En effet, la commande TILC d'ordre un ne permettait pas de stabiliser l'erreur pour une température initiale de 280°K. Pour pallier ce problème, nous nous sommes lancés dans le développement d'une commande du second ordre. À l'exception de l'ensemble d'apprentissage, la stabilisation demeure incertaine pour certaines conditions

d'opération. Ce dernier fait suggère un problème important. En effet, le module de contrôle est sur-spécialisé sur l'ensemble d'apprentissage.

Le problème est cependant plus fondamental. En effet, en étudiant le module de découplage, nous observons une diminution de la qualité de l'approximation. En d'autres termes, l'interaction entre les variables demeure relativement importante après le découplage. Ce phénomène entraîne une interdépendance dans les boucles supposées SISO de la commande TILC. Ainsi, « un combat » s'engage entre les contrôleurs pour minimiser l'erreur. Les dynamiques des interactions changent avec les conditions d'opérations. C'est pour cette raison que le contrôleur est sur-spécialisé sur l'ensemble d'apprentissage.

Maintenant, nous devons expliquer la raison de l'inefficacité du module de découplage. Pour ce faire, étudions d'un peu plus près la figure 4.35. La configuration proposée associe par exemple le capteur IR_{T7} à l'élément chauffant T_1. Puisque le module de commande fonctionne en SISO, seulement T_1 sera modifié pour atteindre la température à IR_{T7}. Le module de découplage étant moins efficace, ces modifications provoqueront des changements sur les autres sorties entraînant ainsi des corrections perturbant IR_{T7}. Comme vous le remarquez, ce choix d'association est arbitraire, car les points de mesure sont situés entre les éléments chauffant. En effet, nous pourrions également associer IR_{T7} à T_4. Cette ambigüité n'existe pas dans les précédentes configurations. Leurs points de mesure étaient situés à l'intérieur de la zone définie par la projection du regroupement sur la feuille de plastique. Par exemple, dans le cas 4x4, la zone de lecture du capteur IR_{T5} était située à l'intérieur de la projection de T_1, T_2 et T_3. Donc, le regroupement associé doit expliquer la plus grande partie des variations observées au point de mesure.

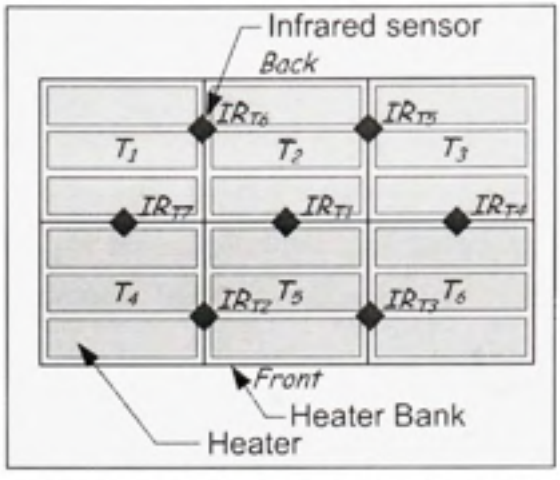

Figure 4.35 Disposition des capteurs de température.
Tirée de Gauthier (2008)

Cette dernière constatation définie une limite à la méthode proposée. En effet, une sortie doit être associée à l'entrée ayant l'interaction la plus élevée. Pour pallier à ce problème, nous pourrions simplement déplacer les points de mesure au milieu. Malheureusement, cette solution n'est pas techniquement envisageable pour les capteurs situés à l'intérieur du four. Éventuellement, l'installation d'un capteur à la sortie du four permettrait la mise en œuvre de cette solution.

CONCLUSION

L'objectif principal de ces travaux était l'application des systèmes d'inférence floue à la commande en fin de cycle par apprentissages successifs à un procédé précis. Le chauffage d'une feuille de plastique du processus de thermoformage est le procédé étudié. Bien entendu, ces travaux permettront également d'améliorer les performances actuelles et d'automatiser l'asservissement. En effet, les consignes de température des éléments chauffant sont déterminées par les opérateurs. Ainsi, les performances dépendent essentiellement de la compétence de l'homme.

Le chapitre un présente le four de thermoformage à partir duquel le modèle de simulation est basé. Puisque nous n'avons besoin en aucun temps du modèle mathématique, nous n'avons pas entrepris une présentation exhaustive de ce dernier. Nous nous sommes contentés de mettre en évidence la symétrie existante dans le modèle de simulation. Nous avons identifié l'origine de cette symétrie en analysant le facteur géométrique. Cette observation nous a permis d'utiliser une importante simplification réduisant le temps de développement d'une solution.

Le chapitre deux est le cœur de nos travaux. En effet, il représente plus de la moitié des efforts. Le module de découplage flou est unique pour deux raisons. La première raison est l'utilisation des données pour effectuer l'apprentissage du système d'inférence floue Takagi-Sugeno. Dans la littérature, les modules de découplage sont obtenus par analyse mathématique. Donc, cette approche impose l'obtention d'un modèle mathématique. Le module ressemblant le plus au nôtre utilise également un système d'inférence. Toutefois, son obtention dépend des connaissances d'un expert. Concernant la modélisation floue, il existe deux approches pour apprendre le modèle réciproque. La première méthode est de permuter les variables indépendante et dépendante. Cependant, cette méthode ne permet pas un bon contrôle de l'apprentissage et utilise beaucoup de données. La seconde méthode est l'utilisation des données directes et, par la suite, inverser le modèle. Dans le cadre de ces travaux, nous avons privilégié cette dernière. Donc, la seconde raison est la méthode

d'inversion analytique proposée pour obtenir le modèle réciproque du système. En effet, dans la littérature, l'inversion de système d'inférence floue se limite à un ordre zéro. Dans notre cas, nous avons démontré qu'il est possible d'inverser un ordre un en utilisant une décomposition et une adaptation des ensembles flous d'entrés. Cependant, nous proposons une approximation permettant une application plus rapide et plus simple. De plus, la flexibilité de la méthode offre la possibilité de l'appliquer à des contextes différents. En effet, elle est applicable à toutes les situations de modélisation en adaptant la base de données destinées à l'apprentissage du module. Par exemple, nous pourrions utiliser les états d'un système et les sorties correspondantes pour approximer le comportement réciproque d'un système linéaire invariant dans le temps.

Dans le chapitre trois, nous avons proposé une commande TILC basée sur l'utilisation d'un système d'inférence floue. En effet, nous suggérons de remplacer le calcul de variation courant par une évaluation floue. Nous limitons nos efforts aux TILC d'ordre un et deux. Pour développer ces contrôleurs, nous proposons deux méthodes. Puisque le TILC d'ordre un est relativement simple à définir, nous proposons la méthode intuitive basée sur les connaissances d'un expert. Toutefois, ces dernières ne sont pas suffisantes pour définir efficacement toutes les interactions existantes dans le cas du second ordre. Nous proposons donc une méthode d'apprentissage par algorithme génétique développée au cours d'une étude préliminaire. Puisque le temps de développement est très long, nous avons toujours privilégié la commande d'ordre un pour nos tests. Donc, ce paramétre est un facteur limitatif à l'utilisation des algorithmes génétiques.

Le chapitre quatre est dédié à la présentation ainsi qu'à l'analyse des résultats de simulation. Nous présentons trois configurations différentes. Pour chacune d'elle, nous caractérisons les performances du module de découplage et du module de commande. Les résultats sont conforment à nos attentes pour les deux premiéres configurations. En effet, un écart aussi important que 20°C sur la température initiale est compensé en moins de trois itérations. De plus, la commande est capable d'atténuer les variations dans le modèle. Toutefois, les résultats se dégradent dans le cas à douze capteurs et à douze éléments chauffant. Nous

remarquons une diminution importante de la robustesse due à la disposition des capteurs par rapport aux éléments chauffant. En effet, nous avons conclu qu'il serait préférable de placer les capteurs au milieu des zones de chauffage. Cependant, ce n'est pas techniquement possible pour les capteurs à l'intérieur du four. Donc, la motivation d'un capteur externe est justifiée. Malgré tout, ce fait constitue une limitation quant au module de découplage. Le lien existant entre la variable d'entrée et la variable de sortie choisie doit être plus fort que tous les autres.

Pour les travaux futurs, nous suggérons de mettre en œuvre les solutions proposées. Effectivement, nous devons poursuivre l'analyse de performance sur le système réel afin de valider les résultats obtenus en simulation. Le principal avantage de la structure de commande proposée est sa modularité. En effet, nous pouvons travailler indépendamment les deux modules pour améliorer les performances. Alors, nous suggérons de poursuivre les travaux sur le module de découplage. Nous n'avons pas encore exploité toutes les symétries et les simplifications possibles. Nous avons expliqué le rôle joué par le facteur de vue dans le second chapitre. En utilisant ce terme, il est possible de caractériser le degré d'interaction. Donc, nous pourrions négliger les interactions très faibles et ainsi diminuer la complexité du module de découplage. De plus, la méthode proposée pour l'élaboration du module de découplage s'applique plus facilement à des petits systèmes. Alors, il serait intéressant de la tester sur d'autres procédés comportant moins de six entrées.

Finalement, ces travaux nous ont permis d'étudier la mise en œuvre des systèmes d'inférence floue pour la commande en fin de cycle par apprentissages successifs. L'utilisation d'un modèle de simulation n'est pas nécessairement requis pour le développement de solutions, mais il facilite grandement le processus. De plus, pour les configurations simples, les solutions rencontrent facilement les objectifs de performances et permettra certainement une diminution des pertes liées à un mauvais chauffage de la feuille de plastique à thermoformer.

ANNEXE I

CRÉATION DU FIS À DEUX ENTRÉES ET UNE SORTIE

Dans cette annexe, nous présentons, à titre d'exemple, le script Matlab pour la création du FIS pour un TILC d'ordre deux. Nous utilisons le «Fuzzy Toolbox » de Mathworks pour créer une structure interprétable par les fonctions de Matlab.

```
% controleur_flou_MISO      (création de la structure FIS)
% Cette fonction est utilisée pour générer la structure FIS du
% TILC d'ordre deux.
% Syntaxe:  a = controleur_flou_MISO(individu)
% Paramètres d'entrées:
%   individu     - Matrice représentant la position du
%                              singleton de sortie
% Paramètres de sortie:
%   a            - Cette variable contient la structure du FIS
% Auteur:       Mathieu B.Turcotte
% Historique:   15.02.2009     fichier créé

function a=controleur_flou_MISO(individu)

%Création de la structure en spécifiant le type de moteur
%d'inférence et les opérations.

     a=newfis('controleur','sugeno','min','max','prod','sum','wtaver');

%Ajout d'un univers de discours d'entrée

     a=addvar(a,'input','erreur k',[-1 1]);

%Ajout des fonction d'appartenance

     a=addmf(a,'input',1,'grande négative','trimf',[-1 -1 -1/2]);
     a=addmf(a,'input',1,'-1/2','trimf',[-1 -1/2 0]);
     a=addmf(a,'input',1,'zero','trimf',[-1/2 0 1/2]);
     a=addmf(a,'input',1,'1/2','trimf',[0 1/2 1]);
     a=addmf(a,'input',1,'grande positive','trimf',[1/2 1 1]);
```

```
%Ajout d'un univers de discours d'entrée

     a=addvar(a,'input','erreur k-1',[-1 1]);

%Ajout des fonction d'appartenance

     a=addmf(a,'input',2,'grande négative','trimf',[-1 -1 -1/2]);
     a=addmf(a,'input',2,'-1/2','trimf',[-1 -1/2 0]);
     a=addmf(a,'input',2,'zero','trimf',[-1/2 0 1/2]);
     a=addmf(a,'input',2,'1/2','trimf',[0 1/2 1]);
     a=addmf(a,'input',2,'grande positive','trimf',[1/2 1 1]);

%Ajout d'un univers de discours de sortie

     a=addvar(a,'output','valve',[-1 1]);

%Ajout des singletons selon la position définit dans individu
   for i=1:25

     a=addmf(a,'output',1,['p' num2str(i)],'constant',individu(i));

   end

%Créer la liste de règle

     ind=1;

     for i=1:5

          for j=1:5

           rule=[i j ind 1 1]; a=addrule(a,rule);ind=ind+1;

             end

     end
```

ANNEXE II

FONCTION DE KRIGEAGE

Dans cette annexe, vous trouverez le script Matlab élaboré pour mettre en œuvre le krigeage. Dans nos travaux, nous avons utilisé ce code pour déterminer les fonctions des conséquents des règles d'inférence floues.

```
% krigeage.m      (Évaluation des paramètres d'un plan par
krigeage)
% Cette fonction est utilisée déterminer les paramètres d'un
% plan en utilisant la méthode statistique du krigeage avec
effet pépite.
% Syntaxe:  ObjVal = krigeage_reg(mat,Version)
% Paramètres d'entrées:
%    mat     - Matrice representant les coordonnées des points
pour
%              l'approximation du plan ([X1,X2,...,Xn,H]).
%    Version - 0:= avec valeur constante;
%              1:= sans valeur constante.
% Paramètres de sortie:
%    result    - Cette variable contient un vecteur
correspondant au
%                résultat ([alpha1,...,alphaN,A,B,C,...]T).
% Auteur:     Mathieu B.Turcotte

function result = krigeage_reg(mat,Version)

[nb_point,nb_dim]=size(mat);

if Version==1

  A=[eye(nb_point),zeros(nb_point,nb_dim-1);zeros(nb_dim-1,nb_point+nb_dim-1)];

Else

   A=[eye(nb_point),ones(nb_point,1),zeros(nb_point,nb_dim-
      1);ones(1,nb_point) zeros(1,nb_dim);zeros(nb_dim-
      1,nb_point+nb_dim)];
end
```

```
for i=1:nb_dim-1

    if Version==1
        A(nb_point+i,:)=[mat(:,i)' zeros(1,nb_dim-1)];
        A(:,i+nb_point)=[mat(:,i);zeros(nb_dim-1,1)];
    else
        A(nb_point+1+i,:)=[mat(:,i)' zeros(1,nb_dim)];
        A(:,i+1+nb_point)=[mat(:,i);zeros(nb_dim,1)];
    end

end

if det(A)==0
    result=zeros(1,nb_dim); disp('singular matrix');result=A;
else
    if Version==1
      Y=[mat(:,nb_dim);zeros(nb_dim-1,1)];
    else
      Y=[mat(:,nb_dim);zeros(nb_dim,1)];
    end
    result=A^-1*Y;
end
```

ANNEXE III

EXEMPLE DE SIMULATION DU FOUR DE THERMOFORMAGE

Cette annexe présente, à titre d'exemple, la simulation du four de thermoformage en configuration à douze capteurs et à douze éléments chauffants. Cette simulation inclut le module TILC d'ordre deux ainsi que le module de découplage. Ce script en langage Matlab permet également le traçage de graphiques représentant l'évolution des erreurs et des commandes appliquées.

```
% eval_12x12_O2.M      (Simulation de la loi de commande d'ordre 2)
%
% Cette fonction est utilisée pour simuler l'asservissement du four de
% thermoformage en configuration 12x12.
%
% Syntaxe:   eval_12x12_O2(Singleton,gain,dec,Tini,Tamb,yd,eps)
%
% Paramètres d'entrees:
%    Singleton      - Vecteur ligne définissant la position des singletons
%                     du FIS calculant la variation de la commande.
%    gain           - Pondération de la plus ancienne commande.
%    dec            - Liste de decoupleurs.
%    yd             - Profil de température desire.
%    eps            - Valeur de décalage entre la température mesurée au
%                     dessus de la feuille et au dessous.
%
% Paramètres de sortie:
%
% Note d'utilisation:
%   Les variables Tini et Tamb doivent être déclarées dans le workspace
%   Les variables Yt et U doivent être déclarées comme variable globale
%
% Auteur:     Mathieu B.Turcotte

function eval_12x12_O2(Singleton,gain,dec,yd,eps)

global Yt U;
% Création du FIS calculant la variation de la commande
   c1=controleur_flou_MISO(Singleton);
%Initialisation des variables
   uk=[yd yd];
   delta_ukn=zeros(length(yd),2);
   error1=zeros(1,12);
   sim_ok=1;
```

```
%Simulation sur 50 itération
  for it=2:52
    %Calcul de la commande
     ukn=uk(:,1)*gain+uk(:,2)*(1-gain)+delta_ukn(:,it);
    %Mise à jour des historiques
     uk(:,1)=uk(:,2);
     uk(:,2)=ukn;
   %Découplage
     pu1=evalfis(uk(1:6),dec{1});
     pu2=evalfis(uk(1:6),dec{2});
     pu3=evalfis(uk(1:6),dec{3});
     pu4=evalfis(uk(1:6),dec{4});
     pu5=evalfis(uk(1:6),dec{5});
     pu6=evalfis(uk(1:6),dec{6});
     pu7=evalfis(uk(7:12),dec{1});
     pu8=evalfis(uk(7:12),dec{2});
     pu9=evalfis(uk(7:12),dec{3});
     pu10=evalfis(uk(7:12),dec{4});
     pu11=evalfis(uk(7:12),dec{5});
     pu12=evalfis(uk(7:12),dec{6});
    % Consigne de température demandé au four
     U=[pu1 pu2 pu3 pu4 pu5 pu6 pu7 pu8 pu9 pu10 pu11 pu12]';
    %Demarrage de l'évalution du profil
     try
      sim('heat_with_radiation',300);
     catch
      sim_ok=0;
     end
    %calcul d'erreur
     last_ech=length(Yt);
     if last_ech==0
       disp('Erreur de simulation');
       break
     else
      error1(it,:)=[Yt(last_ech,7)-yd(1),Yt(last_ech,6)-yd(2),
                    Yt(last_ech,5)-yd(3),Yt(last_ech,2)-yd(1),
                    Yt(last_ech,3)-yd(2),Yt(last_ech,4)-yd(3),
                    Yt(last_ech,14)+eps-yd(4),Yt(last_ech,13)+eps-yd(5),
                    Yt(last_ech,12)+eps-yd(6),Yt(last_ech,9)+eps-yd(4),
                    Yt(last_ech,10)+eps-yd(5),Yt(last_ech,11)+eps-yd(6)];
     end
     consigne(it,:)=ukn';
    %Normalisation des erreurs et calcul des variations de consignes
     for i=1:length(uk);
         errorn(1,1)=1/10*error1(it,i);
         if it>2
           errorn(2,1)=1/10*error1(it-1,i);
         else
           errorn(2,1)=errorn(1,1);
         end
         errorn=min(errorn,1);errorn=max(errorn,-1);
         delta_ukn(i,it+1)=evalfis(errorn,c1)*10;
     end

  end
```

```
% Création des graphiques des erreurs et des consignes
marque={':bo',':bx',':bs',':b^',':b+',':bd',':b*',':bv',':b<',':b>',':bp',':bh'};

      figure;
      %identification des erreurs maximum pour toutes les itérations
      maxerror=[];
      for i=2:52
          maxerror(i-1)=max(abs(error1(i,:)));
      end

      plot(0:1:50,maxerror,marque{1},'MarkerEdgeColor','k');
      xlabel('Itération');
      ylabel(['||erreur||\infty (^\oK)']);

      figure;
      hold on
      for i=1:length(uk)
          plot(0:1:50,consigne(2:52,i),marque{i},'MarkerEdgeColor','k');
      end
      xlabel('Itération');
      ylabel('Consigne de Température (^\oK)');
      h=legend('Consigne   {T1}','Consigne   {T2}','Consigne{T3}','Consigne
               {T4}','Consigne              {T5}','Consigne{T6}','Consigne
               {B1}','Consigne       {B2}','Consigne       {B3}','Consigne
               {B4}','Consigne {B5}','Consigne {B6}',12);
      set(h,'Interpreter','none');
```

ANNEXE IV

SCRIPT DE L'ALGORITHME GÉNÉTIQUE ITÉRATIF

Le script suivant utilise les fonctions d'un « Toolbox » développé par l'équipe de Chipperfield. Plus précisément, nous réalisons par ce script une optimisation des paramètres définissant le conséquent des règles d'inférence floues. Nous avons développé cet algorithme génétique itératif dans le cadre de travaux préliminaires à ce projet.

```
% GAITERATIF.M
%
% Ce script permet de mettre en œuvre un algorithme génetique ITERATIF
% La représentation entière est utilisée pour coder les individus.
%
% Author:     Mathieu B.Turcotte
% Historique:    28 février 2009     file created (copie du fichier SGA.M)

clc
clear
global a b c d e f c1 U error Yt Sim_ul Tini;

% Paramètre généraux
NIND = 20;            % Nombre d'individus dans la population
MAXGEN = 50;          % Nombre maximum de génération
NB_IT = 3;           % Nombre maximum d'itération
CHROMLEN = 26;       % Nombre de singleton de sortie (nombre de règles)

%Paramètres de simulation:
Tini=280; Tamb=398;
a=readfis('dec_12x12_a');
b=readfis('dec_12x12_b');
c=readfis('dec_12x12_c');
d=readfis('dec_12x12_d');
e=readfis('dec_12x12_e');
f=readfis('dec_12x12_f');

%Descripteur d'un individu
FieldDR = [rep([-1;1],[1,CHROMLEN])];

% Définition des opérateurs génétiques
SEL_F = 'rws';        % Nom de la méthode de sélection
XOV_F = 'recint';     % Nom de la méthode de recombinaison
OBJ_F = 'Fc_obj_dec_12x12_2_ordre_2';     % Nom de la fonction objectif
% Définition des probabilités utilisé par les opérateurs
CRS_P = 1;            % Probabilité de croisement (non nécessaire)
```

```
MUT_P =0.5;           % Probabilité de mutation
PENTE = -1/MAXGEN;    % Taux de variation de la fonction Shrink
GGAP = (NIND-1)/NIND; % Generation gap, nombre d'individu sélectionnée
%Paramètre des critères d'arrêt de l'algorithme
BORNE = 20;           % Borne d'arrêt pour une stagnation sur les
générations
BORNE2 = 2;           % Borne d'arrêt pour une stagnation sur les
itérations
FITBORNE = 70;        % Borne d'arrêt basée sur la valeur du meilleur coût
    %Création du fichier d'enregistrement des essaies
    name = strcat('solution_Param_',date,'.txt');
    logfile = fopen(name,'a');
    %Définition d'un format d'affichage d'un individu dans la population
    format = [rep('  %4.4f',[1,CHROMLEN]),'\n'];
    %Initialisation de la variable
    Optgen=MAXGEN*ones(NB_IT,1);
    %Creation des figures
    AllSol = figure;hold on;
    BestSol = figure;hold on;
    j=0;
    while j < NB_IT,
        j=j+1;
        fprintf(logfile,'Iteration %g\n',j);
        % Creation de la population
        % Pour la première exécution, nous créons aléatoirement les
        % individus
        if j==1
           Chrom = crtrp(NIND,FieldDR);
        else
           Chrom = crtrp(NIND-1,FieldDR);
           Chrom(NIND,:)= ChromP;
        end
        % Initialisation des variables d'historique
           gen = 0;
           Best = NaN*ones(MAXGEN,1);

        % Evaluation de la popultation initiale
           ObjV = feval(OBJ_F,Chrom);

        % Évolution de la population vers la génération suivante
           while gen < MAXGEN,

            % Assigne une valeur de fitness aux individus
              FitnV = ranking(ObjV);

            % Sélectionner des individus dans la population
              SelCh = select(SEL_F, Chrom, FitnV, GGAP);

            % Croisement des individus
              SelCh=recombin(XOV_F, SelCh,CRS_P);

            % Mutation des enfants
              Shrink = max(PENTE*gen+1,.5); % déterminer la plage de mutation
              SelCh=mutbga(SelCh,FieldDR,[MUT_P Shrink]);
```

```
        % Évaluation des enfants
          ObjVsel = feval(OBJ_F,SelCh);

        % Insertion des enfants dans la population en remplacant
        % certains parents
          [Chrom ObjV] = reins(Chrom, SelCh,1,1,ObjV,ObjVsel);
          gen=gen+1;disp(gen);

        % Identification du meilleur individu
          [Best(gen),indice] = min(ObjV);
          ChromP = Chrom(indice,:);
          disp(Best(gen));
        % Affichage des resultats
          set(0,'CurrentFigure',BestSol); plot(Best,'ro');
          set(0,'CurrentFigure',AllSol); plot(gen,ObjV,'bo');
          drawnow;

      % Évaluation du premier critere d'arrêt (Basée sur la stagnation
      % de l'évolution sur les génération)
          if gen > BORNE
            if Best(gen)==Best(gen-BORNE)
              fprintf(logfile,'ARRET2 à :%g génération\t Best:
                    ...%.3f\n',gen,Best(gen));
              Optgen(j)=gen;
              gen=MAXGEN;
            end
          end
    % Évaluation du second critere d'arrêt (Basée sur l'obtention d'une
    % solution satisfaisante)
          if Best(gen) < FITBORNE
              fprintf(logfile,'ARRET3 à :%g génération\t Best:
                      %.3f\n',gen,Best(gen));
              Optgen(j)=gen; gen=MAXGEN; j=NB_IT;
          end

        end
      % Déterminer la meilleur solution de l'itération
        BestIt(j)=min(Best);
      % Évaluation du premier critère d'arrêt (Basée sur la stagnation
      % de l'évolution sur les itérations)
        if j > BORNE2
            if BestIt(j)==BestIt(j-BORNE2)
                fprintf(logfile,'ARRET3 à :%g itération\t Best:
                        %.3f\n',j,BestIt(j));
                j=NB_IT;
            end
        end
end
```

```
    % Enregistrement des figures
    Namefile=(strcat('Best_(iteration).fig'));
    saveas(BestSol,Namefile); close(BestSol);
    Namefile=(strcat('All_(iteration).fig'));
    saveas(AllSol,Namefile); close(AllSol);

    % Sauvegarde des resultats
    fprintf(logfile,'Chromosome de la meilleur solution\n');
    fprintf(logfile,'%.3f\t',min(BestIt));fprintf(logfile,format,ChromP);
    fprintf(logfile,'FIN\n');
    fclose(logfile);

% End of script
```

ANNEXE V

ÉTUDE PRÉLIMINAIRE

Pour effectuer l'étude préliminaire, nous avons choisi un système facile à interpréter. Ainsi, nous pourrons mieux comprendre les réactions du système.

Le système à l'étude est un réservoir d'eau. Les deux variables à contrôler sont la température ainsi que le niveau du liquide. Le contrôle doit être réalisé en asservissant les débits d'entrée d'eau chaude et froide. Ce type de système est défini comme étant « multiples entrées - multiples sorties » (MIMO) non-linéaire. La figure suivante présente le procédé.

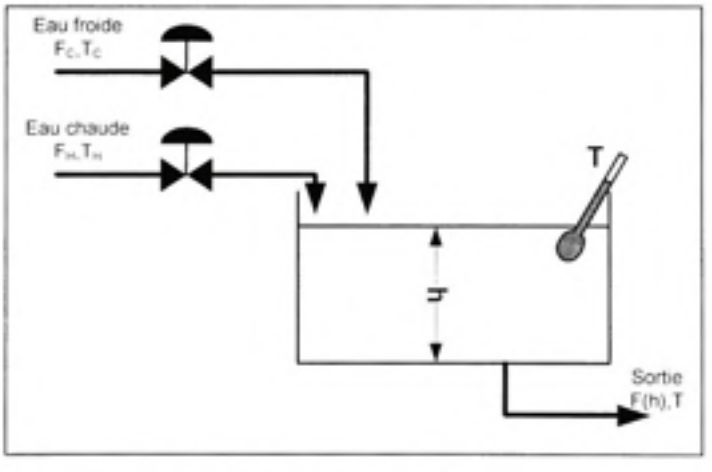

Figure-A V-1 Procédé à l'étude.

La liste suivante définit les variables qui caractérisent le procédé.

A_c := *Aire de la* section *du réservoir*
h := *Niveau de liquide dans le réservoir*
F_H := *Débit du liquide chaud*
F_c := *Débit du liquide froid*
K := Constante d'écoulement
T := *Température du liquide dans le réservoir*
T_H := *Température du liquide chaud*
T_c := *Température du liquide froid*
ρ := *Masse volumique du liquide*
C_p := *Chaleur spécifique du liquide*
à pression constante

Veuillez noter que ce système est traité dans le cours de modélisation et automatisation de procédés industriels (SYS 823). Nous utiliserons, dans ce travail, les éléments contenus dans la présentation (Gauthier, 2006).

La première hypothèse permettant de simplifier la modélisation concerne la température du liquide. Cette variable est considérée uniforme dans le réservoir. En réalité, la température n'est pas nécessairement uniformément distribuée, mais l'ajout d'un mélangeur corrige le problème. De plus, cet ajout n'affecte pas significativement la variable, car l'énergie transférée n'est pas importante. La seconde hypothèse concerne les caractéristiques du liquide. Pour ce projet, nous considérerons la masse volumique et la viscosité constante.

Ces deux hypothèses facilitent l'obtention du modèle mathématique. La première étape de la modélisation est d'identifier le volume de contrôle. Ici, nous choisissons le réservoir. Dans le cas présent, il y a deux dynamiques à modéliser. La première est celle du niveau et la seconde la température. Respectivement, nous devons faire un bilan de matière et un bilan thermique sur le volume de contrôle.

Puisque la masse volumique et la viscosité sont constantes, le bilan massique est réduit à un bilan volumique.

$$A_c \frac{dh}{dt} = F_H + F_c - K\sqrt{h} \tag{V-1}$$

Le bilan thermique est un peu plus complexe.

$$\rho C_p A_c \frac{d(hT)}{dt} = \rho C_p \left(F_H T_H + F_c T_c - K\sqrt{h}T \right) \tag{V-2}$$

La dernière équation doit être manipulée pour isoler la dérivée de la température.

$$T\frac{d(h)}{dt} + h\frac{d(T)}{dt} = {}^{1}\!/_{A_c} \left(F_H T_H + F_c T_c - K\sqrt{h}T \right) \tag{V-3}$$

En remplaçant la dérivée de la hauteur par l'équation 1 et en isolant, on obtient :

$$\frac{d(T)}{dt} = \frac{1}{A_c h}\left(F_H (T_H - T) + F_c (T_c - T) \right) \tag{V-4}$$

Une équation différentielle est dite linéaire lorsque les variables d'état sont multipliés par des constantes. Une variable d'état est une caractéristique représentative de l'état courant du procédé. Prenons, par exemple, un objet en mouvement; la position, la vitesse et l'accélération sont tous des descriptifs de l'état de l'objet. Dans notre cas, les variables d'états sont la température et le niveau dans le réservoir. À la lumière de ces explications, nous remarquons une non-linéarité sur le débit de sortie. De plus, nous remarquons une interaction entre les deux variables d'état. En effet, les débits d'entrées influencent le niveau et la température. Cette dernière conclusion est conséquente à notre interprétation physique du procédé.

Dans le cadre de ce travail, nous privilégions une approche par simulation. Nous utiliserons l'outil « Simulink » offert par Matwork. La figure suivante présente le modèle de simulation réalisé dans le logiciel. Nous avons utilisé les équations 1 et 3 pour réaliser ce schéma.

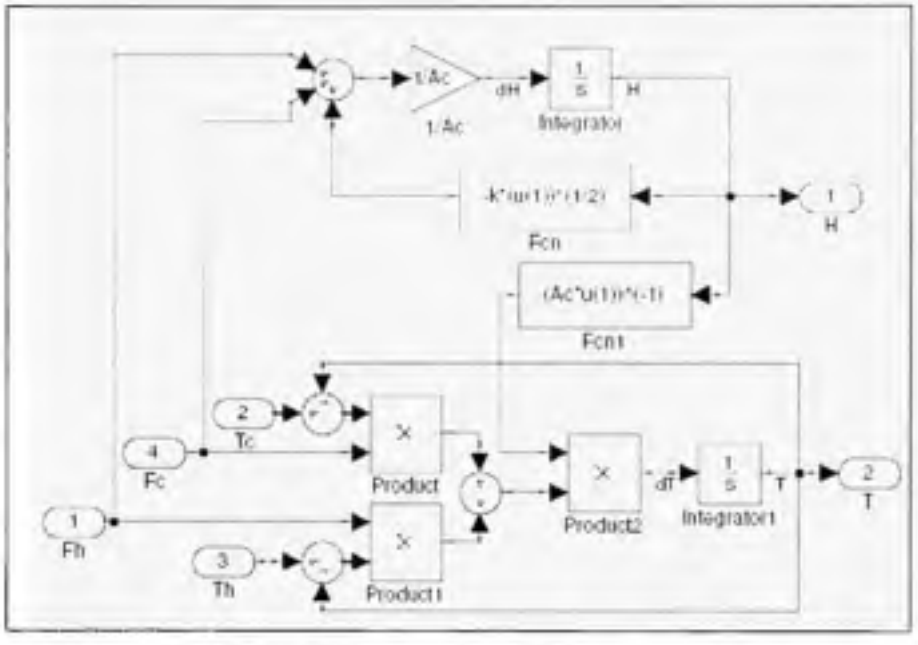

Figure-A V-2 Schéma de simulation du réservoir.

Connaissant le système, nous pouvons débuter la conception du contrôleur flou. L'utilisation de la logique floue est répandue dans les biens de consommations usuelles. Par exemple, les lave-linges « intelligents » utilisent la logique floue. Dans le milieu industriel, la commande floue est utilisée dans le traitement des eaux (Cordon *et al.*, 2001, p. 39). La principale motivation de cette approche est qu'elle ne nécessite pas de modèle mathématique du système à contrôler. En effet, nous pouvons déterminer un contrôleur en colligeant le savoir et l'expérience tacite d'un opérateur ou d'un expert du procédé. Toutefois, ce n'est pas la seule façon. L'utilisation des algorithmes évolutionnaires est une seconde voie possible. Dans le présent cas, la commande floue est une bonne alternative à l'approche classique, car la commande non-linéaire nous permettra de mieux gérer la dynamique du système. Maintenant, décrivons en détails le contrôleur pour l'application du réservoir.

Pour cette tentative de contrôle, nous avons adopté une approche naïve. En effet, nous avons simplement substitué chacun des contrôleurs PI classiques par des contrôleurs flous. Nous avons conservé la même relation entrée-sortie et nous avons retiré les fonctions de découplage. De plus, pour simplifier les tests et pour vérifier les capacités de notre algorithme génétique, nous avons imposé le même système flou aux deux contrôleurs. Toutefois, ce choix nous impose d'ajouter un gain négatif à la sortie de la commande du débit d'eau froide, car nous avons une réponse inverse sur la sortie de la température. La figure suivante présente la nouvelle structure.

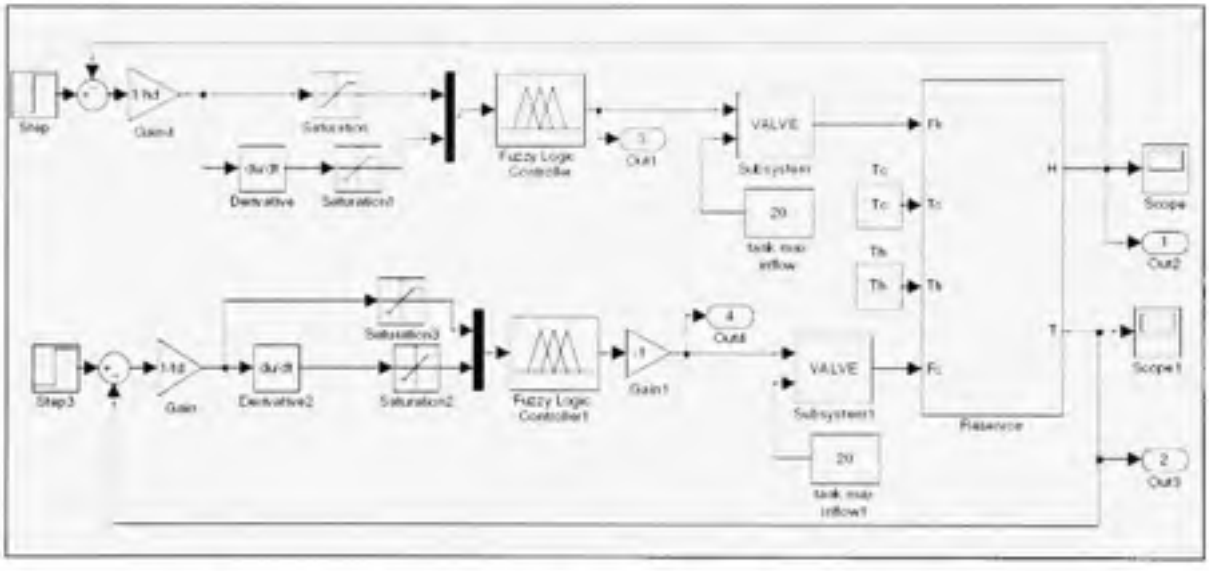

Figure-A V-3 Schéma de simulation de la commande floue.

L'ajout des sous-systèmes « valve » est pour adapter la commande donnée par le contrôleur. Ces sous-systèmes sont simplement des intégrateurs avec des saturations. La saturation est faite à la fin du processus d'intégration pour avoir une commande bornée.

En consultant le schéma précédent, il est étonnant de voir une dérivation du signal. Posons tout d'abord l'équation d'un PI dans le domaine temporel (Driankov, Hellendoorn et Reinfrank, 1996, p. 108).

$$u = K_p e + K_I \int e dt$$
$$K_p := \text{Gain proportionnel}$$
$$K_I := \text{Gain intégral}$$
$$e := \text{Erreur} \qquad \text{(V-5)}$$

Dans notre cas, nous voulons obtenir une variation de la commande. Donc, dérivons la dernière relation.

$$\Delta u = K_p \dot{e} + K_I e \qquad \text{(V-6)}$$

Dans le cadre de cette étude, nous avons décidé d'utiliser le type TSK d'ordre zéro. La continuité de la surface de commande est garantie lorsque l'on utilise le TSK. Dans notre cas, c'est un avantage indéniable puisque nous utiliserons un algorithme génétique. De plus, ce système d'inférence flou est beaucoup moins lourd que le Mamdani.

Nous avons décidé d'utiliser un ordre zéro pour faciliter l'interprétation de notre contrôleur. En effet, ce cas particulier peut être comparé à un système d'inférence flou de Mamdani avec des « singletons » comme fonction d'appartenance de sortie. Ceci signifie simplement que la fonction située dans le conséquent des règles d'inférence est une valeur constante. L'algorithme génétique permettra d'identifier ces valeurs.

La définition des ensembles flous se divise en deux grandes étapes. La première étape est de définir le nombre de fonction d'appartenance définissant l'univers de discours. La seconde étape est de définir les fonctions d'appartenance.

La complexité du moteur d'inférence croît rapidement avec l'ajout de fonctions d'appartenance en entrée. Donc, pour simplifier la conception de ce premier contrôleur flou, nous avons décidé de limiter le nombre de fonctions d'appartenance à trois. Nous avons normalisé l'univers de discours de -1 à 1. Pour ce projet, nous avons choisi des fonctions triangulaires. Pour définir la position des fonctions, il existe deux possibilités. La première

possibilité est d'interroger un expert et la seconde, est de placer uniformément les fonctions sur l'intervalle (Cordon *et al.*, 2001, p. 39). Nous avons choisi la seconde solution, car nous allons de toute façon adapter l'ensemble de sortie. Les deux entrées, l'erreur et la dérivée de l'erreur, utilisent le même ensemble flou. La figure suivante présente l'ensemble flou d'entrée utilisé pour ce projet.

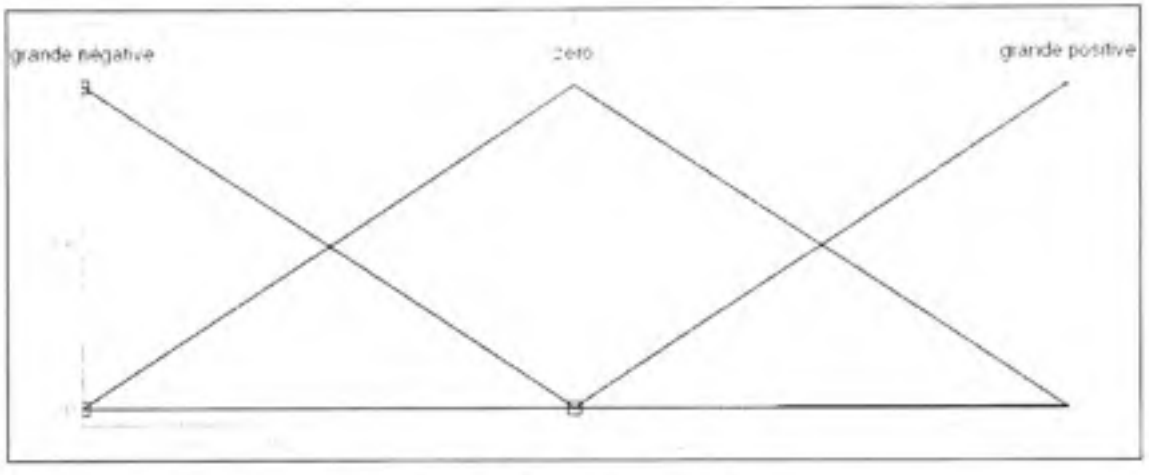

Figure-A V-4 Ensemble flou d'entrée.

Comme nous l'avons précisé plus tôt, nous avons choisi un moteur d'inférence flou de type TSK d'ordre zéro. Donc, l'ensemble de sortie est une série de singletons. Pour offrir plus de capacité d'adaptation, nous avons fixé le nombre de singletons égal au nombre de règles d'inférence. La position de ces singletons sera déterminée avec notre algorithme génétique itératif.

Le type de moteur d'inférence nous impose l'opérateur d'implication et d'agrégation. Respectivement, nous devons utiliser le produit et la somme pour effectuer ces traitements. Pour la fonction « ET » nous avons choisi d'utiliser l'opérateur minimal. Nous pourrions également utiliser le produit. La fonction « OU » n'est pas utilisée dans la définition des règles. Nous l'avons imposée arbitrairement à l'opérateur maximum. Pour la défuzzification, nous avons sélectionné la moyenne pondérée des sorties.

La définition des règles d'interférence est très simple. En optimisant l'ensemble de sortie, la topologie du moteur d'inférence demeure constante. Puisque nous avons seulement deux entrées, nous pouvons représenter sous forme matricielle les règles. Cette matrice de règle est présentée au tableau suivant. Notez que les variables Pi où i= {1,2,…, N} correspondent à la position du $i^{ème}$ singleton.

Tableau-A V-1 Matrice de décision

Règle d'inférence				
		erreur		
		Négative	Zéro	Positive
dérivée	Négative	P1	P4	P7
	Zéro	P2	P5	P8
	Positive	P3	P6	P9

L'opérateur liant les éléments dans la partie de l'antécédent de la règle est le « ET » logique.

LISTE DE RÉFÉRENCES BIBLIOGRAPHIQUES

Babuška, R., R. Jager et H. B. Verbruggen. 1994. « Interpolation issues in Sugeno-Takagi reasoning ». In *Proceedings of the Third IEEE Conference on Fuzzy Systems1994. IEEE World Congress on Computational Intelligence.* (26-29 Juin 1994). p. 859-863 vol.2. Orlando.

Babuška, R., et H. B. Verbruggen. 1996. « An overview of fuzzy modeling for control ». *Control Engineering Practice.* vol. 4, n° 11, p. 1593-1606.

Baranyi, P., I.M. Bavelaar, R. Bubuška, L.T. Kóczy, A. Titli et H. B Verbruggen. 1998. « A method to invert a linguistic fuzzy model ». *International Journal of Systems Science,* vol. 29, n° 7, p. 711-721.

Benahmed, Nadia. 2002. « Optimisation de reseaux de neurones pour la reconnaissance de chiffres manuscrits isoles: Selection et ponderation des primitives par algorithmes genetiques ». Mémoire de maîtrise, Montréal, Ecole de Technologie Superieure 124 p.

Boukezzoula, R., S. Galichet et L. Foulloy. 2003. « Nonlinear internal model control: application of inverse model based fuzzy control ». *Fuzzy Systems, IEEE Transactions on,* vol. 11, n° 6, p. 814-829.

Bristow, D.A., M. Tharayil et A.G. Alleyne. 2006. « A survey of iterative learning control ». *Control Systems Magazine,* vol. 26, n° 3 (Juin), p. 96-114.

Chen, YangQuan, Jian-Xin Xu et Changyun Wen. 1997. « A high-order terminal iterative learning control scheme ». In *36th IEEE conference on decision and control* (San Diego, 10-12 décembre 1997). Vol. 4, p. 3771-3772.

Chien, Chiang-Ju, Chun-Te Hsu et Chia-Yu Yao. 2004. « Fuzzy system-based adaptive iterative learning control for nonlinear plants with initial state errors ». *IEEE transaction on Fuzzy Systems,* vol. 12, n° 5, p. 724-732.

Chipperfield, A., P. Fleming, H. Pohlheim et C. Fonseca. N.d. *Genetic algorithm toolbox.* Sheffield: University of Sheffield, 94 p. <http://www.shef.ac.uk/content/1/c6/03/35/06/manual.pdf>. Consulté le 29 mars 2009.

Cordon, O., F. Herrera, F. Hoffmann et L. Magdalena. 2001. *Genetic fuzzy systems. Evolutionary tuning and learning of fuzzy knowledge bases*, 19. Coll. « Advances in fuzzy systems - applications and theory ». Danvers, USA: World Scientific Publishing, 462 p.

Driankov, D., H. Hellendoorn et M. Reinfrank. 1996. *An introduction to fuzzy control*, 2. USA: Springer, 316 p.

Driankov, Dimiter, et Palm Rainer. 1998. *Advances in fuzzy control*. Coll. « Studies in fuzziness and soft computing ». Heidelberg: Physica-Verlang, 421 p.

Ehlert, J.R., et T.F. Smith. 1993. « View Factors for perpendicular and parallel, rectangular plates ». *Journal of thermophysics and heat transfert*, vol. 7, n° 1, p. 173-174.

Eiben, A. E., R. Hinterding et Z. Michalewicz. 1999. « Parameter control in evolutionary algorithms ». *Evolutionary Computation, IEEE Transactions on*, vol. 3, n° 2, p. 124-141.

Galichet, Sylvie, Reda Boukezzoula et Laurent Foulloy. 2004. « Explicit analytical formulation and exact inversion of decomposable fuzzy systems with singleton consequents ». *Fuzzy Sets and Systems*, vol. 146, n° 3, p. 421-436.

Gauthier, G. 1997. « Coordination de robots pour la manipulation d'objets ». Mémoire de maîtrise, Montréal, Canada, École polytechnique de Montréal, 108 p.

Gauthier, G. 2006. *Système MIMO*. Présenté dans le cadre du cours Modélisation et automatisation de procédés industriels Montréal, ca: École de technologie supérieure.

Gauthier, G. 2008. *Modélisation d'un four de thermoformage*. Présenté dans le cadre du cours Modélisation et automatisation de procédés industriels Montréal (Québec): École de technologie supérieure.

Gauthier, G. 2009. « Terminal iterative learning for cycle-to-cycle control of industrial processes ». Thèse de doctorat en génie, Montréal, McGill university, 201 p.

Gratton, Y. 2002. « Le krigeage : la méthode optimale d'interpolation spatiale ». *Articles de l'Institut d'Analyse Géographique*, (juin), p. 1-4.

Hao, Ying, Ding Yongsheng, Li Shaokuan et Shao Shihuang. 1999. « Comparison of necessary conditions for typical Takagi-Sugeno and Mamdani fuzzy systems as universal approximators ». *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, vol. 29, n° 5, p. 508-514.

Herrera, F., M. Lozano et J.L. Verdegay. 1998. « Tackling real-coded genetic algorithms: Operators and tools for behavioural analysis ». *Artificial Intelligence Rev.*, vol. 12, n° 4, p. 265-319.

Holmann, J.P. 2002. *Heat Transfer*. 9. Coll. « Operations and decision sciences ». New York: McGraw Hill, 665 p.

K.F. Man, K.S. Tang, S. Kwong. 1999. *Genetic algorithms : concepts and designs* New York: Springer-Verlag. 344 p.

Mathworks. N.d. *Mathworks: Documentation on fuzzy logic toolbox*. En ligne. <www.mathworks.com/access/helpdesk/help/toolbox/fuzzy/fp49243.html>. Consulté le 18 janvier 2010.

Mingyu, Fu, Ding Zonghua, Xia Guoging et Fan Kunliang. 2008. « Research on the Fuzzy Decoupling in the Coordinated control system of nuclear power plant ». In *IEEE International Conference on Automation and Logistics 2008* (Quingdao, China, 1-3 Sept. 2008). p. 405-410. China.

Moore, K.L. 1998. « Iterative learning control: An expository overview ». In *Applied and computational controls, signal processing and circuits*. Vol. 1, p. 425-488. Dekalb (Illinois): Biswa Nath Datta.

Munataka, T., et Y. Jani. 1994. « Fuzzy systems: an overview ». *Communications of the ACM*, vol. 37, n° 3, p. 68-76.

Riid, A. , et E. Rüstern. 1998. « Extracting and Exploiting Linguistic Information from a Fuzzy Process Model for Fed-Batch Fermentation Control ». In *Intelligent Engineering Systems and Computational Cybernetics*. p. 95-105. Springer.

Salman, S. A , et G. S Anavatti. 2007. « A Novel Inversion Method for a Fuzzy Model Based Controller ». In *International Conference on Mechatronics and Automation*, sous la dir. de IEEE. p. 100-104. Harbin, China.

Trochu, F. . 1993. « A Contouring Program Based on Dual Kriging Interpolation ». *Engineering with Computers*, vol. 9, p. 160-177.

Verbruggen, H. B., et R. Babuška. 1999. *Fuzzy logic control : advances in applications*. Coll. « World Scientific series in robotics and intelligent systems », vol. 23 Singapore: River Edge, N.J. : World Scientific 329 p.

Wade, Harold L. 2004. *Basic and Advanced Regulatory Control : System Design and Application*, 2. États-Unis: ISA-The Instrumentation, Systems, and Automation Society, 372 p.

Wang, Quin-Guo. 2003. *Decoupling control*. Coll. « Lecture Notes in Control and Information Sciences », vol. 285. New-York: Springer-Verlang, 363 p.

Xu, C., et Y. C. Shin. 2007. « A Fuzzy Inverse Model Construction Method for a General MISO System with a Monotonic Input-output Relationship ». In *Fuzzy Information Processing Society, 2007. NAFIPS '07. Annual Meeting of the North American*. p. 1-6.

Xu, Jian-Xin, YangQuan Chen, Tong Heng Lee et Shigehiko Yamamoto. 1999. « Terminal iterative learning control with an application to RTPCVD thickness control ». *Automatica,* vol. 35, p. 1535-1542.