N° d'ordre : ...

THESE

présentée à

L'UNIVERSITE DES SCIENCES ET TECHNIQUES DE LILLE
FLANDRES ARTOIS

pour l'obtention du titre de

DOCTEUR
En Automatique et Informatique Industrielle

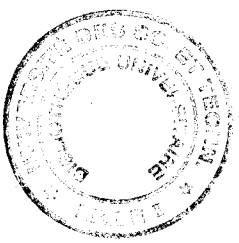
par

Mohamed Ali KOUBAA

CONTRIBUTION A LA COMMANDE PREDICTIVE : MISE EN OEUVRE POUR LE PILOTAGE D'UN AUTOCLAVE DE TEINTURE

soutenue publiquement le 29 janvier 1997 devant la commission d'examen :

MM.	P. VIDAL	Président
	A. RACHID	Rapporteur
	R. LAURENT	Rapporteur
	C. VASSEUR	Directeur de Recherche
	V. KONCAR	Codirecteur
	E. LESTOQUOY	Examinatrice



A mes parents,
Mon frère, Ma soeur
Et à ma chère femme

REMERCIEMENTS

Le travail présenté dans ce mémoire a été réalisé au sein du laboratoire GEMTEX (Génie et Matériaux Textiles) de l'ENSAIT (Ecole Nationale Supérieure des Arts et Industries Textiles).

Je tiens à remercier Monsieur le Professeur Pierre VIDAL, qui m'a fait l'honneur de présider le jury.

Je tiens à exprimer ma profonde reconnaissance à Monsieur Ahmed RACHID, professeur à l'Université de Picardie Jules Verne, et à Monsieur Robert LAURENT, professeur à l'Université d'Orléans, pour avoir accepté d'être les rapporteurs de mes travaux, malgré leurs très nombreuses activités.

Je remercie Monsieur le Professeur Christian VASSEUR, Directeur de l'ENSAIT, pour avoir accepté de diriger mes travaux de recherche.

Mes sincères remerciements vont également à Monsieur Vladan KONCAR, codirecteur de mes travaux de recherche, Maître de Conférences à l'ENSAIT et Responsable du département Productique du GEMTEX, pour ces précieux conseils et le soutien qu'il m'a apporté au cours de ces années.

Je remercie également Madame E. LESTOQUOY, Directeur du Département Teinture chez D.M.C fil à coudre à LOOS, pour avoir accepté de faire partie du jury.

Je remercie l'ensemble des membres du GEMTEX et de l'ENSAIT pour la gentillesse et la patience qu'ils ont témoignées à mon égard.

Finalement, j'adresse mes profonds remerciements à Monsieur Pascal BRUNIAUX, Maître de Conférences à l'ENSAIT et Monsieur Besoa RABENASOLO, Maître de Conférences à l'ENSAIT pour leurs collaboration et pour toutes les discussions fructueuses que nous avons pu avoir.

CHAPITRE 5 : AUTOCLAVE DE TEINTURE

INTRODUCTION

Introduction

Dans le domaine du textile, nous avons constaté à travers les manifestations internationales telles que l'ITMA'91 à Hanovre ou encore l'ITMA'95 à Milan, que l'automatisation des machines textiles est de plus en plus présente. Pour améliorer les performances de ces dernières, les constructeurs concentrent leurs travaux sur la mise en place d'une électronique poussée. Aujourd'hui, ils s'intéressent à l'interface homme/machine pour améliorer l'ergonomie du dialogue entre l'utilisateur et la supervision du procédé.

Notre principal axe de recherche vise à automatiser un procédé de teinture.

Les problèmes de régulation de position, de vitesse, d'accélération, de force et aussi de température sont rencontrés dans des domaines industriels très divers (textile, métallurgie, agro-alimentaire, chimie, etc...). Habituellement, l'utilisation de régulateurs classiques fondés essentiellement sur une structure PID permet la commande de ce type de procédé [CLARKE 1984]. Le point fort de ces méthodes de commande classiques provient du fait qu'elles sont capables de fournir des performances suffisantes en terme de rapidité, de dépassement et d'annulation d'erreurs entre la consigne et la sortie du système. De plus, leur réglage est relativement simple pour un utilisateur non familié des nouvelles stratégies de commande mises au point ces dernières années. En revanche, l'une des principales difficultés rencontrées dans la commande des procédés utilisant des régulateurs de type PID est la présence du retard dans la réponse du système.

Cependant, il est intéressant de constater que de nouvelles méthodes avancées, utilisant par exemple la logique floue sont mise en oeuvre pour le dosage et le contrôle du pH [ITB 1995], [RITT 1996]. D'autres techniques sont utilisées pour le contrôle et la régulation de la concentration d'eau oxygénée dans les bains de traitement, la mesure de la température de la matière dans le séchoir, la mesure des bains de colorants, etc...

L'objectif des constructeurs est de conserver en permanence la maîtrise du processus par l'intermédiaire de capteurs et d'actionneurs et par le stockage de l'ensemble des paramètres du processus.

Notre étude a été réalisée au sein du laboratoire GEMTEX (Génie et Matériaux Textiles) à l'ENSAIT (Ecole Nationale Supérieure des Arts et Industries Textiles). Son objet est d'apporter des éléments supplémentaires à l'élaboration de nouvelles méthodes de commandes, afin de participer au développement industriel que demande la filière textile.

Cette thèse vise à l'élargissement de la commande prédictive à des systèmes complexes linéaires et non linéaires, ce qui permet de maîtriser davantage la conduite des processus. Par ailleurs, notre contribution au monde industriel consiste en l'élaboration d'un logiciel de commande d'un autoclave de teinture, à partir d'un PC facile à gérer par l'utilisateur. Cette partie pratique répond au souhait des utilisateurs de machines industrielles et réalise en partie l'objectif fixé par les constructeurs.

En réalité, cette étude rentre dans le cadre d'un projet plus ambitieux du GEMTEX à savoir : mesurer l'épuisement du bain, en extraire un spectre ou une signature permettant une meilleure conduite en température. Ce mode de commande en temps réel est illustré par le schéma bloc suivant.

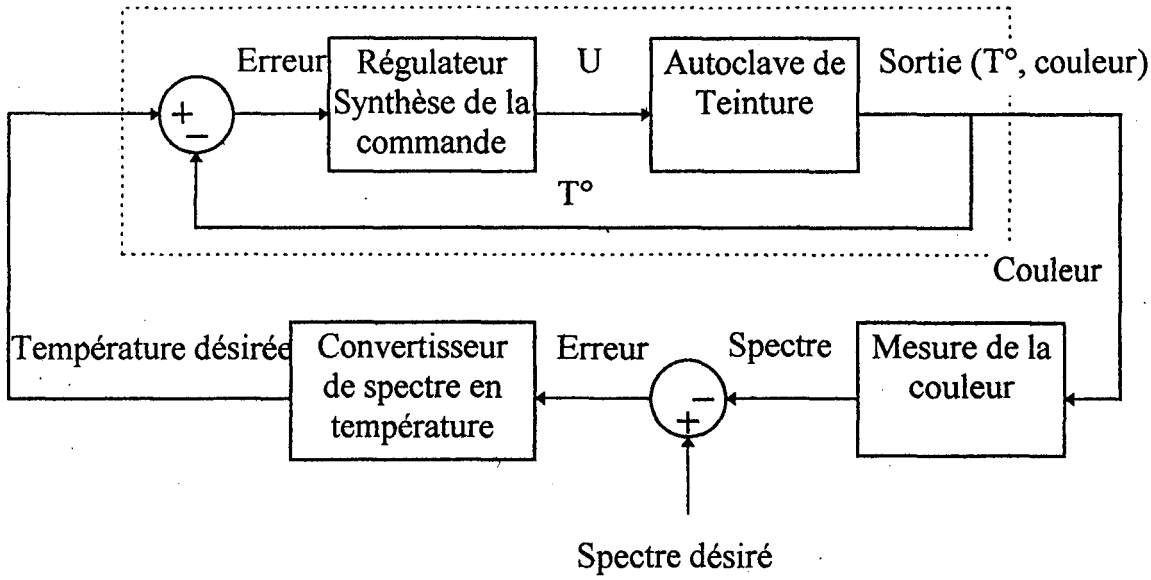


Schéma bloc général du projet

Ce schéma fait apparaître deux boucles : une boucle interne réalisant le suivi d'un profil de température. Cette boucle interne correspond au fonctionnement des systèmes actuels qui utilisent un profil de température élaboré en laboratoire (recette). Toutefois, la référence définie en laboratoire ne permet pas de garantir une reproductibilité parfaite au niveau industriel. En effet, les conditions du laboratoires sont généralement différentes de celles rencontrées sur les procédés industriels qui font apparaître des contraintes multiples et complexes (différence des machines et des supports textiles).

Le projet consiste à superposer au système existant, une boucle externe dont le rôle est de corriger en temps réel le profil de température appliqué à la boucle interne. Cette correction utilise des informations supplémentaires sur le procédé tels que : épuisement, pH, conductivité etc...

L'intérêt industriel et économique de cette démarche réside essentiellement dans le fait que les machines existantes peuvent intégrer cette technique sans modifications notables.

La commande prédictive nous a paru la plus appropriée pour commander l'autoclave de teinture, en raison de ses capacités à prendre en considération les effets de retard. Ces retards sont dus aux caractéristiques physiques du système à contrôler: constantes de temps d'éléments chauffants, phénomènes de transport de matières, acquisition et conversion des données, etc...

Ce type de commande est particulièrement bien adaptée aux problèmes d'asservissement en température (commande de fours des traitements thermiques, commande de colonnes à distiller etc...). L'idée principale est de transformer le problème classique de commande de la sortie à l'instant présent, par la commande d'une prédiction de la sortie à un ou à des instants futurs [SEBORG 1986]. Pour cela nous prenons en compte explicitement la connaissance d'une trajectoire à suivre dans le futur et nous faisons coïncider la sortie du processus avec cette consigne.

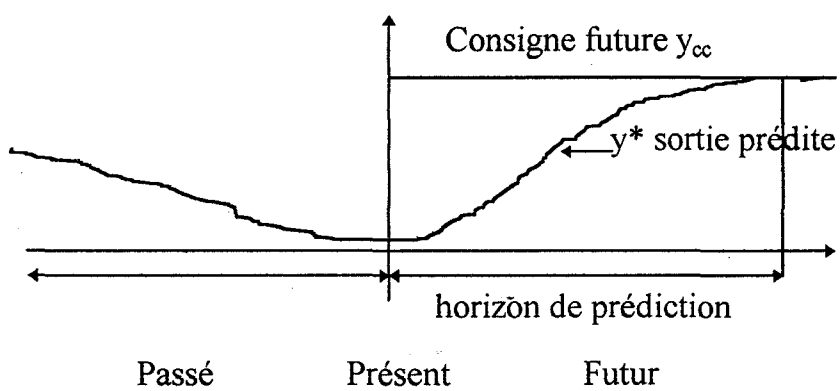


Schéma de prédiction

Cette notion de sortie prédite permet aussi de prendre en compte l'effet de retard, à condition que cette prédiction soit faite à un instant au moins égal à l'instant présent auquel on ajoute le retard. L'efficacité de la commande prédictive vient du fait qu'on connaît la trajectoire à suivre. Néanmoins, **quelques problèmes se posent lors de son application à un processus physique réel**, parmi lesquels :

- l'attribution d'une valeur fixe à certains paramètres pendant toute la durée de commande n'est pas valable étant donnée que le système change d'état au cours du temps (variations, capacité de chauffage et de circulation du bain de teinture, etc..).

- l'utilisateur est amené à fixer un comportement de référence à suivre par le système. Ce choix est relativement délicat et risque de dégrader les performances de la commande telles que la sensibilité et la robustesse.

- cette commande ne s'applique qu'aux systèmes linéaires. Or, le processus que nous commandons est non linéaire et est loin d'obéir aux hypothèses faites en théorie.

Ainsi, notre objectif a été l'élaboration d'une nouvelle commande prédictive intelligente applicable au secteur industriel. La propriété essentielle de notre commande est d'être capable de contrôler des processus à modèle linéaire ou non linéaire, tout en conservant la robustesse déjà acquise pour les systèmes linéaires.

Les éléments essentiels qui ont permis de réaliser notre objectif, se résument à :

- un contrôle du procédé plus fréquent grâce à l'utilisation du suréchantillonnage.

- un ajustement automatique du facteur de pondération de la commande au cours du fonctionnement du procédé.

- une mise au point de la méthode permettant le choix de la dynamique du système global de contrôle assurant une robustesse maximale de la commande.

- une application de la commande prédictive aux systèmes non linéaires, par l'utilisation de la programmation non linéaire, afin de minimiser le critère quadratique relatif à ce type de commande.

Tous ces éléments réunis améliorent considérablement la commande prédictive, en élargissant son domaine de validité aux systèmes non linéaires et lui confèrent une excellente robustesse vis à vis des perturbations extérieures.

PLAN ET CONTRIBUTION DE NOTRE ETUDE

Dans un premier temps, nous présentons l'état de l'art de la commande prédictive, ensuite nous introduisons la commande prédictive généralisée et nous finissons par la commande prédictive à modèles de références multiples.

Dans un second chapitre, nous proposons une méthode qui permet **l'ajustement automatique du facteur de pondération de la commande** intervenant dans le critère à minimiser. L'idée générale de notre approche est d'imposer un comportement de suivi exponentiel à l'erreur entre la consigne et la sortie du système. Cette technique d'ajustement automatique rend l'utilisation de la commande prédictive plus facile à gérer. Généralement l'efficacité d'une méthode de commande n'est pas limitée uniquement au bon choix des paramètres qui contribuent à son élaboration, ni à ses performances directes (qualité d'asservissement, temps de réponse, optimisation de la commande, ...). L'efficacité peut être améliorée si la commande prend en compte les perturbations extérieures voire même l'écart existant entre le modèle réel du processus et le modèle théorique. Ceci implique que la commande soit à la fois performante et robuste. En pratique, nous avons utilisé cette méthode pour contrôler l'asservissement en température d'un processus thermique.

Dans un troisième chapitre, nous proposons une méthode aidant l'utilisateur à **réaliser un choix optimal du comportement désiré**. L'introduction de cette notion de choix augmente la robustesse de la commande prédictive vis à vis des variations paramétriques du modèle du système. De plus, l'application de cette méthode permet l'évaluation de la robustesse du système global de contrôle. Nous finirons cette partie

par la présentation des résultats de simulations de cette méthode appliquée au processus thermique.

Dans un quatrième chapitre nous nous sommes intéressés à **l'application de la commande prédictive aux procédés non linéaires**. Actuellement les techniques utilisées couramment pour pallier les difficultés de la non linéarité des modèles sont basées essentiellement sur la linéarisation. Cette approche entraîne des erreurs entre les modèles théoriques et réels. Aussi, nous avons choisi de conserver le modèle non linéaire, et grâce à la programmation non linéaire utilisée pour la minimisation du critère quadratique, la commande prédictive peut être appliquée à ces modèles. Ceci contribue davantage à l'efficacité de la commande prédictive, car on s'approche beaucoup plus du modèle réel. La commande prédictive utilisant la programmation non linéaire a été utilisée dans le cadre d'une simulation, pour contrôler un processus thermique représenté par un modèle non linéaire.

La partie expérimentale de cette thèse consiste en l'application de la commande pratique à haut gain et de notre méthode de commande prédictive pour piloter un autoclave de teinture en temps réel. Cette méthode utilise l'ajustement automatique du facteur de pondération de la commande et un choix optimal du comportement désiré.

En conclusion, seront présentés des remarques sur les méthodes proposées et des travaux de recherches qui pourraient être réalisés dans l'avenir.

REFERENCES

[Clarke 1984]:
Clarke D. W.
« Self Tuning Control of Minimum Phase Systems », Automatica, Vol. 20 n°5, pp. 501-517, 1984.

[ITB 1995]:
International Textile Bulletin, Teinture/Impression/Finissage, N°4, pp 37-42, 1995.

[Ritt 1996]:
Ritt H. M, Krauss P, Rake H.
« Predictive Control of a pH-plant Using Gain Scheduling », Symposium on Control, Optimization and Supervision, Computational Engineering in Systems Applications, CESA'96 IMACS Multiconference, Lille, July 9-12, 1996, pp 473-478.

[Seborg 1986]:
Seborg D. E, Edgar T. F, Shah S. L.
« Adaptive Control Strategies for Process Control », AICHE Journal, Vol. 32, n°6, pp. 881-913, 1986.

CHAPITRE 1

COMMANDE PREDICTIVE

Chapitre 1

COMMANDE PREDICTIVE

I.1) INTRODUCTION

Dans ce chapitre nous allons analyser les algorithmes de commande prédictive généralisée que nous noterons dans toute la suite GPC (Generalized Predictive Control).

Dans un premier temps, nous présentons l'état de l'art de la commande prédictive et ensuite nous étudions de façon détaillée l'algorithme GPC développé par Clarke [CLARKE 1987], [CLARKE 1988], [CLARKE 1989]. Cet algorithme fait partie des lois de commande à horizon de prédiction étendu.

Dans un second temps, nous explicitons l'algorithme de la commande prédictive généralisée à modèles de références multiples (GPC/MRM) proposée par Irving [IRVING 1986]. Cet algorithme constitue une version plus élaborée de l'algorithme GPC. L'algorithme GPC/MRM a servi de base pour notre étude et a été un support pour toutes les améliorations proposées dans ce domaine.

I.2) ETAT DE L'ART DE LA COMMANDE PREDICTIVE

En ce qui concerne la commande prédictive, les premières études ont véritablement commencé au début des années soixante par Smith qui a proposé une commande basée sur l'utilisation d'un modèle de prédiction et d'une loi de commande classique [SMITH 1959]. Le prédicteur de Smith avait pour but d'apporter une solution aux difficultés rencontrées lors de la commande des procédés et de prendre en compte l'effet du retard dans la boucle de commande.

Le développement de ce type de commande a connu plusieurs étapes. Le plus célèbre des algorithmes est la commande à variance minimale, proposée par Aström et Wittermark [ASTRÖM 1973]. Cette commande est basée sur une prédiction à d pas en avant, où d est le nombre de pas d'échantillonnage représentant le retard du système. La commande est calculée par minimisation d'un critère quadratique, intégrant d'une part l'erreur de prédiction de la sortie par rapport à la consigne à l'instant t+d et d'autre part la valeur de la commande.

Malgré les avantages apparents de la commande à variance minimale, son domaine d'application demeure relativement restreint. En effet, on doit non seulement disposer d'un modèle de prédiction à phase minimale, c.à.d à inverse stable, mais aussi connaître exactement la valeur du retard (ce qui n'est pas toujours évident en pratique).

Or, il est bien connu que la discrétisation d'un système continu à inverse stable peut aboutir à un modèle discret à phase non minimale [WELLSTEAD 1979]. Une solution à ce problème a été proposée par Clarke. Elle consiste à incorporer dans le critère une pondération du terme de la commande [CLARKE 1975]. C'est la Commande à Minimum de Variance Généralisée (GMV).

Deux autres méthodes ont été présentées par la suite, pour surmonter les difficultés rencontrées: le placement des pôles [ASTRÖM 1980] et le concept de prédiction à horizon étendu.

La commande à placement de pôles ne nécessite ni une connaissance précise du retard, ni un modèle à phase minimale. En effet, il suffit d'augmenter l'ordre du modèle pour prendre en compte les incertitudes sur la valeur du retard. Le problème inhérent à cette méthode est le manque de robustesse numérique de la résolution de l'équation de Bezout (présence de facteurs communs).

Enfin, la notion de commande prédictive étendue est apparue comme une solution possible, car elle utilise des horizons supérieurs au retard pour le calcul de ses prédictions. Il est évident que le fait de prédire plus loin le comportement du système permet d'obtenir des commandes plus modulées et donc d'améliorer la régularité du fonctionnement du procédé considéré.

D'une façon générale, trois facteurs différencient les méthodes proposées. Ce sont l'expression du critère à minimiser, la manière dont la prédiction est effectuée et l'horizon sur lequel se fait cette prédiction.

Avant de passer à la présentation d'autres formes de la commande prédictive, il serait intéressant de préciser que le concept de prédiction étendue, en tant qu'outil de synthèse de commande, est dû à Richalet [RICHALET 1978]. Celui-ci a proposé une méthode basée sur une modélisation du procédé par réponse impulsionnelle, couplée à la synthèse d'une trajectoire de référence, représentant le comportement désiré en boucle fermée. Mais les performances de cette méthode reste limitées car elle ne permet pas de stabiliser des systèmes instables en boucle ouverte, ou à phase non minimale.

Ces méthodes restent malgré tout limitées par les types de modèles de prédiction utilisés, car ces derniers ne fournissent pas beaucoup d'informations sur le système, notamment en ce qui concerne les systèmes instables en boucle ouverte. Par conséquent, les études qui ont suivi se sont tournées vers l'utilisation de modèles de type fonction de transfert, généralement discrète.

Le premier modèle utilisé a été le modèle contrôlé « ARMA » (Auto Regressive Moving Average) représenté par l'équation :

$$A(q^{-1})y(t) = B(q^{-1})u(t) + w(t)$$

où

- y(t) : la sortie monovariable du système.
- u(t) : la commande du système.
- w(t) : une fonction aléatoire représentant le bruit subi par le système.
- q^{-1} : l'opérateur de retard.
- $A(q^{-1})$ et $B(q^{-1})$: deux polynômes fonction de q^{-1}, caractérisant le système à étudier.

En 1982, Martin-Sanchez a proposé un système de commande prédictive adaptative (APCS) [MARTIN-SANCHEZ 1982].

De plus, De Keyser a proposé sa propre approche dans ce domaine en calculant la sortie prédite non pas à l'instant présent, mais à un ensemble d'instants futurs [DE KEYSER 1982].

La commande adaptative à horizon étendue EHAC (Extended Horizon Adaptive Control) proposée par Ydstie en 1984 utilise le principe de commande à horizon glissant, le calcul de la commande peut se faire par minimisation d'un critère quadratique [YDSTIE 1984]. Cependant, l'inconvénient de ce type de commande réside dans le fait qu'une seule prédiction soit utilisée et qu'elle ne permet pas de stabiliser des systèmes naturellement instables.

La commande prédictive généralisée GPC [CLARKE 1987], est l'un des derniers membres de la famille des commandes à horizon étendu. Elle se veut une généralisation des méthodes présentées auparavant. Le modèle utilisé est du type CARIMA (Controlled Auto Regressive Integrated Moving Average) qui n'est autre

qu'une extension du modèle ARMA introduisant un effet intégral. Les formes explicites représentant ces modèles seront présentées dans les chapitres à venir avec plus de détails. Ainsi, grâce à ce modèle, la réponse indicielle du système est prédite sur plusieurs périodes d'échantillonnage. Une séquence d'entrée est ensuite calculée de manière à minimiser l'écart entre les sorties prédites et les références correspondantes au sens d'un critère quadratique, sous la contrainte que la commande ne varie pas au delà d'un certain horizon. Le principe de la commande à horizon glissant est conservé dans la mesure où seule la première commande est appliquée au système. Finalement, la notion de prédiction à horizon étendu de la sortie et d'horizon de commande, rend la GPC peu sensible au retard (variable ou inconnu) et au problème de déphasage non-minimal.

D'autres types de commandes basées essentiellement sur le principe de la GPC peuvent être retrouvés à partir du choix des paramètres de synthèse (horizon de prédiction, horizon de commande, etc...). Le développement le plus significatif de la GPC est certainement la commande à doubles modèles de références proposée par Irving en 1986 [IRVING 1986] et [M'SAAD 1987]. L'idée de base est de combiner une commande à placement de pôles avec la méthode de synthèse précitée. Ceci implique deux variables d'erreurs, l'une portant sur la sortie par rapport à sa trajectoire de référence, l'autre portant sur l'entrée par rapport à sa trajectoire de référence.

La minimisation de ces erreurs est alors faite en utilisant le principe de la commande prédictive généralisée. Cette méthode affine l'intérêt de pouvoir contrôler le comportement transitoire de la sortie par placement des pôles, ce qui n'est pas le cas avec la GPC. Le développement de cette commande s'étend jusqu'à la commande prédictive généralisée en cascade [DUMUR 1991] [DUMUR 1992a]. Cette commande consiste à décomposer le système en deux sous système notés « système interne » et « système externe », chacun de ces sous systèmes étant défini comme une boucle corrigée par un algorithme de commande prédictive. Le système interne est géré par une GPC/MRM selon la présentation donnée ci-dessus. L'avantage de cette

approche est qu'en présence de perturbations les rejections de celles-ci au niveau de la boucle interne sont rapides.

Un autre type de commande prédictive, basé sur l'utilisation d'un modèle de connaissance (modèle interne du procédé), est la commande prédictive fonctionnelle PFC (Predictive Functional Control) introduite par Richalet [RICHALET 1987].

Cette commande présente de nombreux points communs avec la GPC développée précédemment. Néanmoins, la prédiction se fait à l'aide d'un modèle de processus appelé modèle interne, représenté sous la forme CARIMA ou ARMA, où la consigne n'intervient pas directement lors de la minimisation du critère. Une trajectoire, appelée trajectoire de ralliement, spécifie la manière selon laquelle on souhaite rallier la consigne sur un horizon de prédiction. La PFC calcule la commande suivant une stratégie d'horizon glissant [DUMUR 1992b] [DUMUR 1993].

D'autres variétés de commande prédictive ont été élaborées pendant ces dernières années afin d'améliorer les performances de la commande de procédé. Parmi ces variétés, on note principalement la commande prédictive à contraintes sur un horizon lointain CRHPC (Constrained Receding Horizon Predictive Control) et la commande prédictive à horizon étendu EHPC (Extended Horizon Predictive Control) [KONG 1994].

Dans le cas de la CRHPC, le critère à minimiser de type quadratique fait intervenir des coefficients de pondération pour la sortie et pour la commande. Ces coefficients se présentent sous forme de fonctions exponentielles du temps. Ces coefficients variables à chaque itération amènent les valeurs propres de la matrice d'état représentant le modèle du procédé dans un cercle de rayon R [YOON 1993] [YOON 1994].

Quant à la commande prédictive à horizon étendu EHPC, un critère de choix de l'horizon de prédiction est élaboré de sorte que le système soit stable en boucle fermée.

Dans ce critère l'horizon est déterminé en fonction de la réponse impulsionnelle du système [SOH 1985]. Bien que ces méthodes paraissent toutes performantes, le problème majeur de la commande prédictive demeure dans le choix du modèle et des différents paramètres intervenant dans le critère quadratique, tels que l'horizon de prédiction de la sortie, l'horizon de prédiction de la commande et le coefficient de pondération de la commande.

I.3) COMMANDE PREDICTIVE GENERALISEE (GPC)

I.3.1) INTRODUCTION

La notion de commande prédictive à horizon de prédiction étendu a connu plusieurs approches. Néanmoins, l'idée de base demeure la même, à savoir calculer une séquence future de commande à appliquer à l'entrée du processus en vue de forcer la sortie à suivre la trajectoire désirée. Seul le premier élément de cette séquence est appliqué au système, ce calcul étant réinitialisé à chaque période d'échantillonnage.

De nombreux chercheurs se sont intéressés à ce sujet et ont proposé différents algorithmes de commande prédictive. Cependant on relève des points communs qui constituent le fondement de cette stratégie de commande :

- Une commande calculée en minimisant un critère quadratique sur un horizon futur, ce calcul étant effectué à chaque période d'échantillonnage.

- Un critère constitué en général de l'écart entre la consigne (ou la trajectoire issue de celle ci) et la sortie prédite du système qui peut également inclure une pondération sur la commande ou son incrément.

- Un modèle de représentation nécessaire à la prédiction de la sortie du système, ce modèle est linéaire et peut être issu d'un calcul analytique ou d'une identification.

Cette notion de commande prédictive a été introduite par Richalet [RICHALET 1978].

I.3.2) PRINCIPE DE LA COMMANDE PREDICTIVE GENERALISEE (GPC)

Nous allons commencer par rappeler brièvement les versions de la GPC développées dans les travaux de D. W. Clarke publiés dès 1985. La commande à appliquer au système est calculée de manière à minimiser un critère au sens quadratique, sous réserve que la commande ne varie pas au delà d'un certain horizon. La philosophie de la commande à horizon glissant est conservée, c'est à dire que seul la première commande est appliquée.

Un des intérêts de cet algorithme réside dans la simplicité de sa mise en oeuvre informatique. La combinaison des notions de prédiction à horizon étendu et d'horizon de commande rend la GPC peu sensible aux problèmes du retard variable ou inconnu et du déphasage non minimal.

Le principe de la commande prédictive généralisée obéit au schéma suivant :

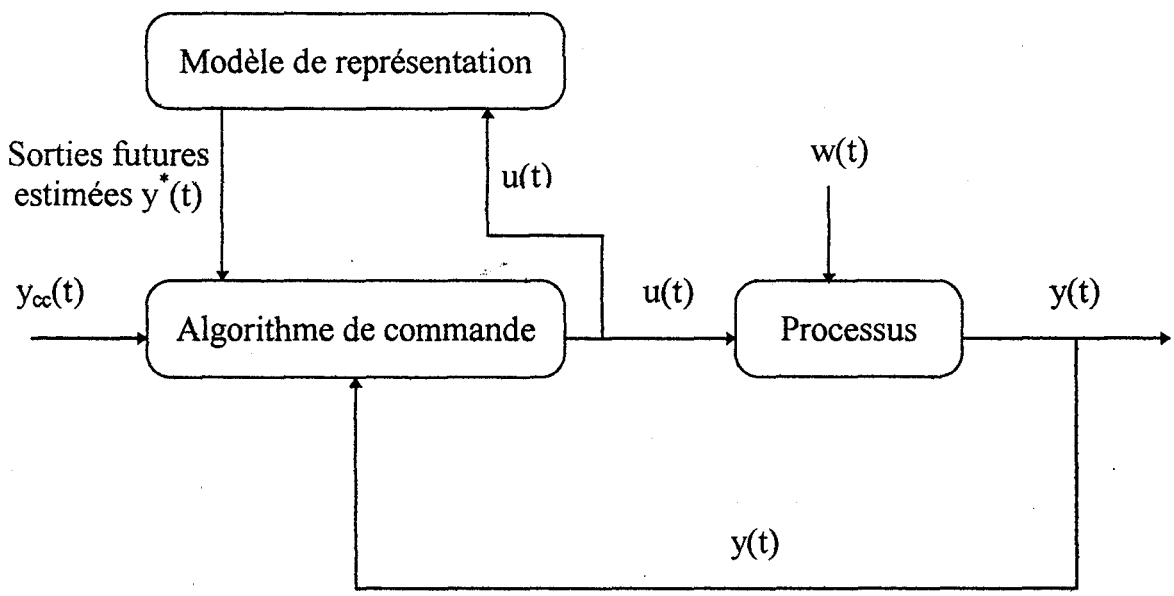


Fig I.1 Schéma structurel de la GPC

$y_{cc}(t)$: représente la consigne

$y^*(t)$: la sortie prédite obtenue grâce au modèle de représentation du processus

Ce modèle, obtenu soit par transformée en z de la fonction de transfert continue du processus ou encore par identification préalable du système, permet de calculer la sortie prédite sur un certain horizon en échantillonné ou en continu.

Ce calcul détermine alors la séquence de commande minimisant l'erreur entre la sortie prédite et la consigne, tout en assurant la convergence de cette sortie vers la consigne souhaitée. Le schéma de prédiction donné en introduction générale illustre ce principe.

I.3.3) FORMULATION MATHEMATIQUE DE LA METHODE

Soit $H(z) = \dfrac{B(z)}{A(z)}$ la fonction de transfert du processus. Cette fonction de transfert n'est pas le modèle de représentation.

Ce qui conduit à la relation suivante (modèle ARMA) :

$$A(q^{-1})y(t) = B(q^{-1})u(t) + w(t) \qquad (I.1)$$

où

y(t) : la sortie du système.

u(t) : la commande appliquée à l'entrée du système.

w(t) : les bruits ou perturbations.

q^{-1} : l'opérateur retard.

$A(q^{-1})$ et $B(q^{-1})$: deux polynômes fonction de q^{-1}, caractérisant le système à étudier.

Les polynômes $A(q^{-1})$ et $B(q^{-1})$ caractérisant le système sont définis par :

$$A(q^{-1}) = 1 + \sum_{i=1}^{n} a_i q^{-i} \qquad (I.2)$$

$$B(q^{-1})=B'(q^{-1})q^{-d} \quad (I.3)$$

avec,

$$B'(q^{-1}) = \sum_{i=1}^{m} b_i q^{-i} \quad (I.4)$$

d représente le retard et m<n

Remarque : Clarke a également introduit le modèle CARIMA [CLARKE 1987] défini par l'équation :

$$A(q^{-1})y(t) = B(q^{-1})u(t) + \frac{w(t)}{\Delta(q^{-1})} \quad (I.5)$$

soit,

$$A(q^{-1})\Delta y(t) = B(q^{-1})\Delta u(t) + w(t) \quad (I.6)$$

avec $\Delta(q^{-1})=1-q^{-1}$ appelé opérateur différence.

Cette forme permet de raisonner sur un modèle incrémental du système. En ce qui nous concerne, nous utilisons le modèle ARMA.

I.3.4) EXPRESSION DU CRITERE

La séquence de commandes futures est obtenue par minimisation du critère quadratique introduit par Clarke [CLARKE 1987], [CLARKE 1988], [CLARKE 1989] défini comme suit, dans le domaine échantillonné :

Soit $t=nT_e$, T_e est la période d'échantillonnage et n un entier.

$$J(t,N_1,N_2,N_u)=\sum_{j=N_1}^{N_2}\left[y_{cc}((n+j)T_e)-y^*((n+j)T_e)\right]^2+\lambda\sum_{j=0}^{N_u-1}\Delta u((n+j)T_e)^2 \qquad (I.7)$$

sous la contrainte pour $j\geq N_u$, $\Delta u((t+j)T_e)=0$

avec,

y_{cc} (.) : consigne

y* (.) : sortie prédite

Δu(.) : incrément de commande

N_1 : horizon minimal de prédiction de la sortie

N_2: horizon maximal de prédiction de la sortie

N_u : horizon de commande

λ : facteur de pondération sur la commande.

La synthèse de la commande prédictive généralisée s'effectue en deux étapes distinctes. La première étape comporte la synthèse d'un prédicteur optimal à j pas nécessaire pour remplacer $y((n+j)T_e)$ par $y^*((n+j)T_e)$. La seconde permet le calcul de la commande au sens d'un horizon glissant. Le calcul du prédicteur optimal se fait par la résolution des équations diophantines. Les détails de cette méthode seront présentés dans le paragraphe I.4 ci-après.

L'analyse du critère à minimiser conduit aux remarques suivantes :

* La connaissance de la consigne entre les horizons N_1 et N_2 aide à optimiser la convergence de la sortie prédite vers la consigne de façon optimale.

* La présence de Δu(.) dans le critère permet d'appréhender la commande de manière incrémentale. Ceci vise à minimiser les variations de commande.

* Le coefficient de pondération λ donne plus ou moins de poids à la commande par rapport à la sortie afin d'assurer la convergence du système lorsque celui-ci présente un risque d'instabilité.

I.3.5) CONCLUSION

Cette méthode offre la possibilité de commander différents types de processus notamment des systèmes à retard important, systèmes instables ou systèmes à non minimum de phase. L'exécution de l'algorithme nécessite un temps de calcul assez faible vu sa simplicité. En revanche, l'un des inconvénients de cette méthode est la méconnaissance totale de l'état de stabilité du système en boucle fermée et le choix du paramètre λ qui reste délicat.

I.4) COMMANDE PREDICTIVE GENERALISEE A MODELES DE REFERENCES MULTIPLES (GPC/MRM)

Comme nous l'avons vu précédemment, la GPC présente un avantage important par rapport aux méthodes classiques de commande dans la mesure où elle est capable de prendre en compte le futur dans le calcul de la commande et ceci à l'instant présent. La GPC/MRM proposée par Irving s'inspire à la fois de la GPC, de la technique des modèles de références multiples sur la commande et sur la sortie et de la méthode de placement des pôles.

La figure ci-dessous présente la schéma général de l'algorithme GPC/MRM.

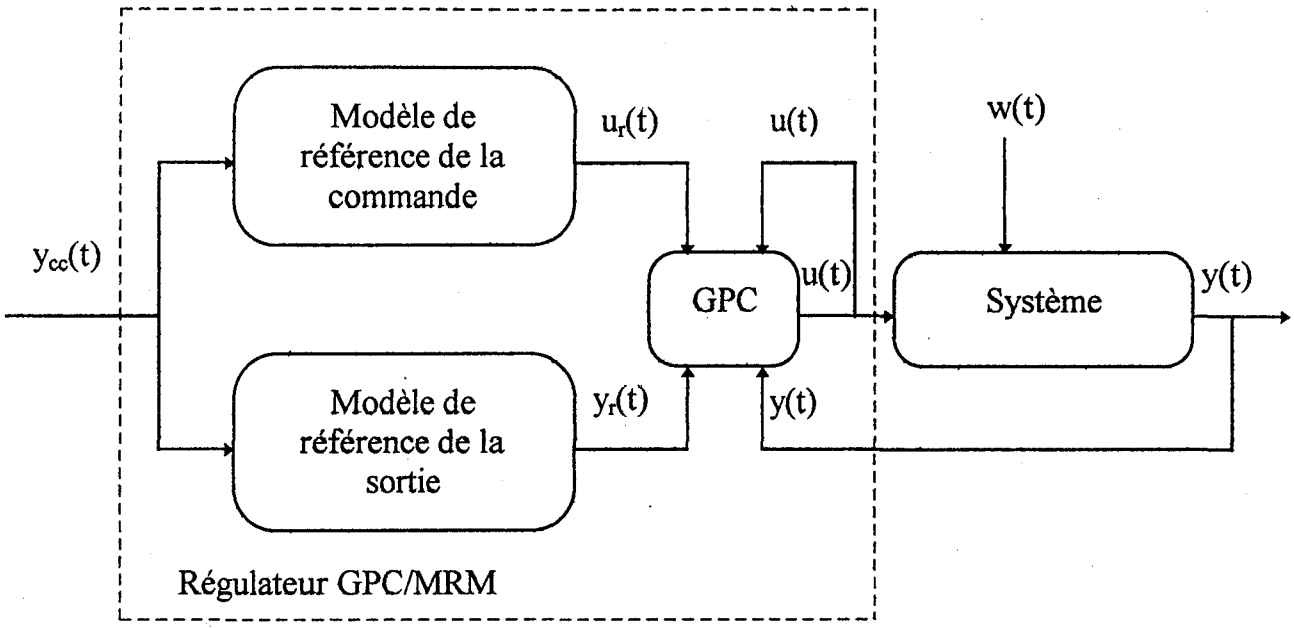


Fig I.2 Schéma de l'algorithme GPC/MRM

$y_{cc}(t)$: consigne

$y_r(t)$: sortie de référence

$u_r(t)$: commande de référence

$y(t)$: sortie du système

$u(t)$: commande du système

$w(t)$: perturbations

Ainsi, nous adoptons la démarche suivante. Nous imposons tout d'abord un modèle sur la commande et sur la sortie du système et nous forçons ensuite la commande et la sortie de ce système à suivre ces modèles de références par l'algorithme de GPC.

I.4.1) FORMULATION MATHEMATIQUE

Le système à commander est caractérisé par un modèle de type ARMA, tel que celui présenté dans l'équation (I.1).

La commande est calculée par la méthode de placement des pôles. L'originalité de cette méthode est de considérer une commande à placement de pôles, permettant la spécification de la dynamique en boucle fermée, sans simplifier les zéros du système [IRVING 1986], [M'SAAD 1987]. Le schéma bloc correspondant à cette méthode est présenté ci-dessous (Fig I.3).

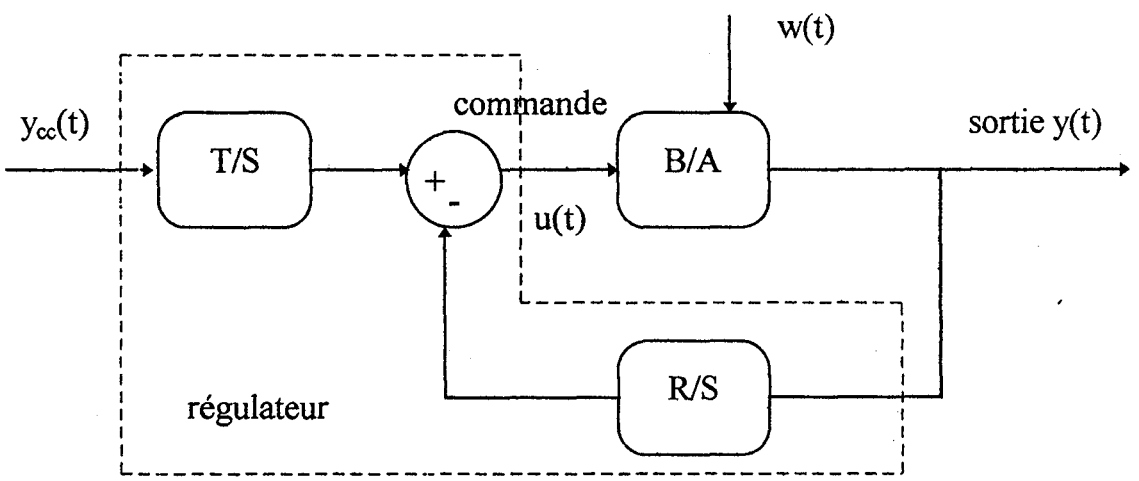


Fig I.3 Schéma structurel du système de commande à placement de pôles

Selon ce schéma, la loi de commande est donnée par :

$$S(q^{-1})\, u(t) = -\, R(q^{-1})\, y(t) + T(q^{-1})\, y_{cc}(t) \qquad (I.8)$$

avec

$$S(q^{-1}) = 1 + \sum_{i=1}^{n_s} s_i q^{-i} \qquad (I.9)$$

$$R(q^{-1}) = \sum_{i=0}^{n_r} r_i q^{-i} \qquad (I.10)$$

$$T(q^{-1}) = \sum_{i=0}^{n_t} t_i q^{-i} \qquad (I.11)$$

avec $n_t \leq n_s$ et $n_r \leq n_s$ ce qui rend impossible une dérivation pure des signaux.

Dans ces conditions, le régulateur est donc un système multivariable d'entrées $y_{cc}(t)$ et $y(t)$, et de sortie $u(t)$.

Nous allons montrer dans la suite du développement comment le schéma de la figure I.3 constitue une réalisation du schéma de la figure I.2. Dans un premier temps les équations permettent d'écrire :

$$(A(q^{-1})\, S(q^{-1}) + B(q^{-1})R(q^{-1}))*y(t) = B(q^{-1})\, T(q^{-1})\, y_{cc}(t) + S(q^{-1})\, w(t) \qquad (I.12)$$

et

$$(A(q^{-1})\, S(q^{-1}) + B(q^{-1})R(q^{-1}))*u(t) = A(q^{-1})\, T(q^{-1})\, y_{cc}(t) - R(q^{-1})\, w(t) \qquad (I.13)$$

Par ailleurs, la méthode de placement des pôles permet d'imposer la dynamique du système global de contrôle par une fonction de transfert notée :

$$H_r(z) = \frac{B_r(z)}{A_r(z)} = \frac{y(z)}{y_{cc}(z)} \qquad (I.14)$$

Dans ces conditions, les équations (I.13) et (I.14) suggèrent d'imposer :

$$A_r(q^{-1}) = A(q^{-1})\, S(q^{-1}) + B(q^{-1})R(q^{-1}) \qquad (I.15)$$

Finalement, et conformément au schéma de la figure I.2 nous adoptons les modèles de références suivants :

* Pour la sortie :

$$A_r(q^{-1})\, y_r(t) = B(q^{-1})\, T(q^{-1})\, y_{cc}(t) \qquad (I.16)$$

* Pour la commande :

$$A_r(q^{-1})\, u_r(t) = A(q^{-1})\, T(q^{-1})\, y_{cc}(t) \qquad (I.17)$$

En conclusion les équations (I.12), (I.13), (I.16) et (I.17) permettent de réorganiser le schéma de la figure I.3 selon le schéma ci-dessous (Fig I.4).

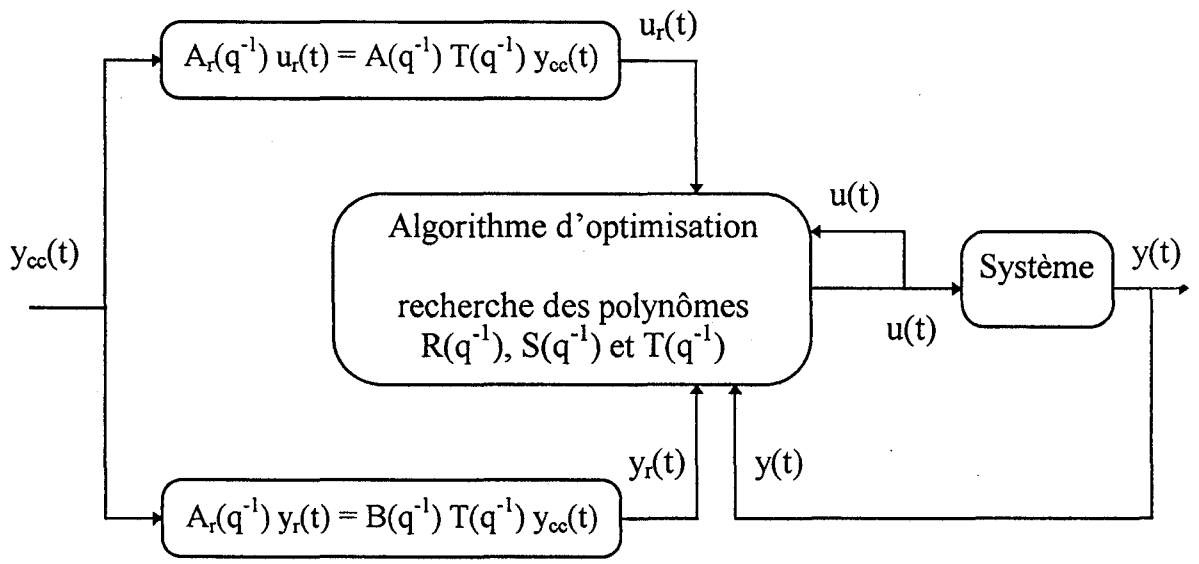


Fig I.4 Schéma structurel de commande

I.4.2) EXPRESSION DU CRITERE D'OPTIMISATION

Cette réorganisation qui est conforme au schéma de la figure I.2, laisse apparaître un élément qui reste à définir : l'algorithme d'optimisation permettant de définir les polynômes $R(q^{-1})$, $S(q^{-1})$ et $T(q^{-1})$. Cet algorithme est présenté dans la suite du développement.

Pour forcer le système à suivre ces modèles de références, il faut minimiser conjointement $y_r(t)-y(t)$ et $u_r(t)-u(t)$. Selon les équations (I.12), (I.13), (I.16) et (I.17) nous avons le système d'équations :

$$\begin{cases} A_r(q^{-1})\,(y_r(t) - y(t)) = - S(q^{-1})\,w(t) \\ A_r(q^{-1})\,(u_r(t) - u(t)) = R(q^{-1})\,w(t) \end{cases} \quad (I.18)$$

Minimiser $(y_r(t)-y(t))$ et $(u_r(t)-u(t))$ revient donc respectivement à minimiser $S(q^{-1})\,w(t)$ et $R(q^{-1})\,w(t)$ par les expressions :

$$\varepsilon_{sy}(nT_e)=A_r(q^{-1})y(nT_e)-B(q^{-1})T(q^{-1})y_{cc}(nT_e) \quad (I.19)$$

$$\varepsilon_{su}(nT_e)=A_r(q^{-1})u(nT_e)-A(q^{-1})T(q^{-1})y_{cc}(nT_e) \quad (I.20)$$

Dans ces conditions, nous adoptons un critère d'optimisation quadratique de la forme échantillonnée :

$$J(nT_e,N_y,N_u) = \sum_{i=0}^{N_y-1} \varepsilon_{sy}^2((n+i+d)T_e) + \lambda \sum_{i=0}^{N_u-1_e} \varepsilon_{su}^2((n+i)T_e) \quad (I.21)$$

Ce critère est un critère à horizon fuyant du type GPC. De plus, il tend à annuler l'effet du retard pur du système (dT_e).

I.4.3) CALCUL DU PREDICTEUR OPTIMAL

Pour calculer le critère de l'équation (I.21) il est nécessaire de prédire la sortie de l'instant $(n+d)T_e$ à l'instant $(n+d+N_y-1)T_e$. Selon l'équation (I.1) du système, cette prédiction peut prendre la forme :

$$y(t+kT_e)=F_k(q^{-1})y(t)+G_k(q^{-1})u(t+(k-d)T_e)+H_k(q^{-1})u(t-T_e)+E_k(q^{-1})w(t+k\,T_e) \quad (I.22)$$

avec $t=nT_e$ et $k=d, ..., d+N_y-1$.

De plus, par identification du modèle (I.1) et de l'équation (I.22), les polynômes $G_k(q^{-1})$, $F_k(q^{-1})$, $H_k(q^{-1})$ et $E_k(q^{-1})$ doivent vérifier les équations diophantines ci-dessous :

$$A(q^{-1})E_k(q^{-1})+q^{-k}F_k(q^{-1})=1 \quad (I.23)$$

$$G_k(q^{-1})+q^{-1-k+d}H_k(q^{-1})=B'(q^{-1})E_k(q^{-1}) \quad (I.24)$$

avec les contraintes de degré suivantes :

$d°G_k(q^{-1})=k-d$
$d°E_k(q^{-1})=k-1$
$d°F_k(q^{-1})= d°A(q^{-1})-1$ et $k\geq d$

Dans l'équation (I.22) le bruit $w(t+k\ T_e)$ n'est pas accessible, par conséquent nous ne pouvons calculer qu'une valeur estimée de la prédiction, selon l'équation :

$$y^*(t+kT_e)=F_k(q^{-1})\tilde{y}(t)+G_k(q^{-1})u(t+(k-d)\ T_e)+H_k(q^{-1})u(t-T_e) \quad (I.25)$$

En fait (I.25) est (I.22) sans tenir compte du bruit. De la même façon ε_{sy} est estimé par :

$$\varepsilon^*_{sy}(nT_e)=A_r(q^{-1})y^*(nT_e)-B(q^{-1})T(q^{-1})y_{cc}(nT_e) \quad (I.26)$$

A partir de (I.19), (I.23), (I.24), (I.25) et (I.26) on peut exprimer ε^*_{sy} en fonction de ε_{su} selon l'équation :

$$\varepsilon^*_{sy}(t+kT_e)=G_k(q^{-1})\varepsilon_{su}(t+(k-d)T_e)+\rho_k(t) \quad (I.27)$$

avec,

$$\rho_k(t)= A_r(q^{-1})[\, F_k(q^{-1})y(t)\, H_k(q^{-1})u(t-T_e)]+G_k(q^{-1})\, A(q^{-1})\, T(q^{-1})\, y_{cc}(t+(k-d)T_e) \\ -B(q^{-1})\, T(q^{-1})\, y_{cc}(t+kT_e) \qquad (I.28)$$

I.4.4) RESOLUTION DU CRITERE ET CALCUL DE LA COMMANDE

Considérons les vecteurs :

$$Y^T(t) = \left[\varepsilon^*_{sy}(t+d)T_e),..,\varepsilon^*_{sy}(t+(d+N_y-1)T_e)\right] \qquad \dim Y=N_y \qquad (I.29)$$

$$U^T(t) = \left[\varepsilon_{su}(t),..,\varepsilon_{su}(t+(N_u-1)T_e)\right] \qquad \dim U=N_u \qquad (I.30)$$

Le critère estimé J^* peut alors s'écrire sous la forme :

$$J^*(t,N_y,N_u) = Y^T(t)Y(t)+\lambda U^T(t)U(t) \qquad (I.31)$$

Par ailleurs, d'après l'équation (I.27) on a :

$$Y(t) = MU(t)+V(t) \qquad (I.32)$$

avec M matrice diagonale formée par les coefficients du polynôme $G_k(q^{-1})$ suivant l'expression :

$$M=\begin{bmatrix} g_{d,0} & 0 & . & . & 0 \\ . & . & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \\ . & . & . & . & g_{d+N_y-1,N_u-1} \end{bmatrix} \qquad \dim M=N_y \times N_u \qquad (I.33)$$

et V(t) défini selon l'équation (I.28) par :

$$V^T(t) = \left[\rho_d(t),..,\rho_{d+N_y-1}(t)\right] \qquad \dim V=N_y \qquad (I.34)$$

Pour trouver la commande optimale, il nous reste à minimiser le critère par la condition nécessaire :

$$\frac{\partial J^*(t,N_y,N_u)}{\partial U(t)} = 0 \qquad (I.35)$$

soit encore, $$2\left[\frac{\partial Y(t)}{\partial U(t)}\right]Y(t) + 2\lambda U(t) = 0 \qquad (I.36)$$

Finalement,

$$\left[\frac{\partial Y(t)}{\partial U(t)}\right] = M \Rightarrow U(t) = -\left[\lambda I + M^T M\right]^{-1} M^T V(t) \qquad (I.37)$$

La commande u est alors calculée en utilisant l'équation (I.20) au sens d'un horizon glissant.

I.4.5) CONCLUSION

Cette méthode présente les avantages suivants :

* Elle permet de gérer les systèmes à retard.
* Elle confère à la commande une bonne robustesse.
* La dynamique du système en boucle fermée est imposée par l'intermédiaire du polynôme $A_r(q^{-1})$. On trouvera dans le chapitre 3, des exemples réalisés avec la méthode GPC/MRM.

* L'obtention d'une commande stable est possible grâce au facteur de pondération λ.

Par contre, elle présente les inconvénients suivants :

* Elle n'inclut pas de stratégie de choix ni pour le coefficient λ, ni pour le polynôme A_r définissant la dynamique désirée du système global de contrôle.
* Son application est restreinte aux systèmes linéaires.
* La détermination des polynômes $E_k(q^{-1})$, $F_k(q^{-1})$, $G_k(q^{-1})$ et $H_k(q^{-1})$ à partir des équations correspondantes est assez compliquée. Ce calcul est généralement réalisé d'une manière récursive.
* Elle présente des difficultés dans l'inversion de la matrice $(M^TM+\lambda I)$ de dimension $N_u \times N_u$, le calcul étant très vite alourdi lorsque l'horizon de prédiction de la commande N_u devient important.

Sur la base des travaux théoriques présentés dans ce chapitre, qui résultent des études de Irving, nous proposons dans les chapitres suivants les améliorations permettant notamment :

* Un ajustement dynamique optimal de λ (chapitre 2).
* Une stratégie de choix du polynôme A_r, visant l'optimisation de la robustesse (chapitre 3).
* La mise en oeuvre sur les systèmes non linéaires (chapitre 4).

Le chapitre 5 illustre tous ces aspects sur un système industriel.

REFERENCES

[Aström 1973]:
Aström K. J , Wittermark B.
« On Self Tuning Regulators », Automatica 1973, Vol. 9, pp. 185-199.

[Aström 1980]:
Aström K. J, Wittermark .
« Self Tuning Conrtollers Based on Pole-zero Placement », IEEE. Proc 1980, Vol. 127, n°3, pp. 120-130.

[Clarke 1975]:
Clarke D. W, Gawthrop P. J.
« A Self Tuning Controller », IEEE Proc 1975, Vol. 122, n°9, pp. 929-934.

[Clarke 1987]:
Clarke D. W, Mohtadi C, P. Tuffs.
« Generalized Predictive Control. part I: The Basic Algorithm. Part II: Extension and Interpretation.», Automatica 1987, Vol. 23, n°2, pp. 137-160.

[Clarke 1988]:
Clarke D. W.
« Application of Generalized Predictive Control to Industrial Processes», IEEE Control Systems Magazine 1988, Apr. 1988, pp. 49-55.

[Clarke 1989]:
Clarke D. W. , Mohtadi C.
« Properties of Generalized Predictive Control», Automatica 1989, Vol 25, n°6, pp. 859-875.

[De Keyser 1982]:
De Keyser R. M. C, Van Cauwenberghe A. R.
« Simple Self Tuning Multistep Predictors », IFAC 1982, Symposium on Identification and System Parameter Estimation, Washington, 1982.

[Dumur 1991]:
Dumur D, Boucher P, Daumüller S.
« Predictive Cascade Control of Machine Tools Motor Drives », 4th European Conference on Power Electronics and Application, EPE 91, September 1991, Florence.

[Dumur 1992a]:
Dumur D, Boucher P, Daumüller S.
« Parameters Automatic Design of Predictive Cascaded Controllers », 4th IFAC, International Symposium on Adaptive Systems in Control and Signal Processing, Grenoble, Juillet, ACSP 1992.

[Dumur 1992b]:
Dumur D, Boucher P, Chêne A, Lafabrègue E.
« Polynomial Predictive Functional Controller (PPFC) for A.C Motors », Workshop on Motion Control for Intelligent Automation, MCIA 1992, Perugia, Octobre 1992.

[Dumur 1993]:
Dumur D.
« Commande Predictive et Machine Outil », Thése d'Etat, Ecole Supérieure d'électricité, Gif sur Yvette, 1993.

[Irving 1986]:
Irving E., Falinower C., Fonte C.
« Adaptive Generalized Predictive Control with Multiple Reference Model », Proc. of the 2nd IFAC Workshop on Adaptive Systems in Control and Signal Processing, Lund, 1986.

[Kong 1994]:
Kong K, De Keyser R.
« Criteria for Choosing The Horizon in Extended Horizon Predictive Control », IEEE Transaction on Automatic Control 1994, Vol. 39, July 1994, N°7, pp 1467-1470.

[Martin-Sanchez 1982]:
Martin-Sanchez J. M.
« A Global, Stable APC in The Presence of Bounded Unmeasured Noises and Disturbances », 21th IEEE. Proc. Conference on Decision and Control 1982, p 761.

[M'Saad 1987]:
M'Saad M.
« Sur l'applicabilité de la Commande Adaptative », Thèse de Doctorat d'Etat de l'Institut Polytechnique de Grenoble, 1987.

[Richalet 1978]:
Richalet J., Rault A., Testud, Papon J.
« Model Predictive Heuristic Control: Application to Industrial Processes », Automatica 1978, Vol. 14, pp. 413-428.

[Richalet 1987]:
Richalet J, Abu El Ata Doss S, Arber C.
« Predictive Functional Control , Application to Fast and Accurate Robots », Proc. of the 10th IFAC 1987, Munich.

[Smith 1959]:
O. J. M. Smith .
« A Controller to Overcome Dead-Time », Instrument Society of America Journal 1959, Vol. 6, n°2, pp. 28-33.

[Scattolini 1990a]:
Carini P, Micheli R, Scattolini R.
« Multirate Self-tuning Predictive Control with Application to a Binary Distillation Column », Int. J. System Sc. 1990, Vol. 21, pp. 51-64.

[Scattolini 1990b]:
Colaneri P, Scattolini R, Schiavoni N.
« Regulation of Multirate Sampled-Data Systems », C-TAT, Vol. 7, pp. 429-441, 1990.

[Scattolini 1990c]:
Colaneri P, Scattolini R, Schiavoni N.
« Stabilization, Regulation and Optimization of Multirate Sampled-data Systems », Control and Dynamic Systems (C.T. Leondes, Ed.) Academic Press, 1990.

[Scattolini 1990d]:
Colaneri P, Scattolini R, Schiavoni N.
« Stabilization of Multirate Sampled-Data Systems », Automatica 1990, Vol. 26, pp. 377-380.

[Soh 1985]:
Soh Y. C, Berger C. S, Dabke K. P.
« On The Stability Properties of Polynomials with Perturbed Coefficients », I.E.E.E Transaction on Automatic Control 1985, Vol. 30, pp. 1033-1036.

[Wellstead 1979]:
Wellstead P.E, Edmunds J. M, Prager D. , Zanger P.
« Self tuning Pole-Zero Assignment Regulators », Int. J. Control 1979, Vol. 30, n°1, pp. 1-26.

[Ydstie 1984]:
Ydstie B. E.
« Extended Horizon Adaptive Control », 9th IFAC World Congress 1984, Budapest.

[Yoon 1993]:
Yoon T. W, D. W. Clarke.
« Receding Horizon Predictive Control with Exponential Weighting », Int. J. System Science 1993, Vol. 24, pp. 1745-1757.

[Yoon 1994]:
Yoon T. W, D. W. Clarke.
« Adaptive Predictive Control of The Bebchmark Plant », Automatica 1994, Vol. 30, N°4, pp. 621-628.

CHAPITRE 2

COMMANDE PREDICTIVE A PARAMETRE λ AUTO-AJUSTABLE

Chapitre 2

COMMANDE PREDICTIVE A PARAMETRE λ AUTO-AJUSTABLE

II.1) INTRODUCTION

Dans ce chapitre nous allons présenter la commande prédictive à paramètre de pondération λ auto-ajustable et son application à un banc d'essai réalisé pour contrôler un asservissement en température.

Dans un premier temps, nous introduisons le principe de suréchantillonnage. Ce principe sera appliqué dans la suite, à la commande prédictive à modèles de références multiples.

Dans un second temps, nous présentons **la commande prédictive à paramètre auto-ajustable en vue d'une poursuite exponentielle**.

Enfin, nous terminons ce chapitre par la **présentation des résultats de la commande MPC/MRM avec paramètre auto-ajustable, appliquée à un processus thermique**.

II.2) LE SURECHANTILLONNAGE

II.2.1) INTRODUCTION

Les méthodes de commande échantillonnée sont très souvent utilisées pour contrôler des processus industriels. La simplicité de la mise en oeuvre et la flexibilité d'implantation de ces algorithmes encouragent l'utilisation de ce type de commande. Cependant, il est important de considérer les effets de la commande échantillonnée à l'intérieur des périodes d'échantillonnage afin de prendre en compte le phénomène d'oscillations cachées, qui peuvent apparaître entre deux périodes [RAGAZINI 1958].

Nous nous plaçons, dans ce chapitre dans l'hypothèse linéaire. L'utilisation d'une commande échantillonnée pose le problème du choix de la période d'échantillonnage T_e (T_e que nous désignerons par la période d'échantillonnage de référence par la suite). Ce choix est souvent limité par des contraintes à respecter telles que le théorème de Shannon [ASTROM 1984], [BÜHLER 1987].

En effet, une période d'échantillonnage très courte permet non seulement de stabiliser le système, mais aussi d'éliminer les oscillations cachées et les perturbations en diminuant le temps de fonctionnement en boucle ouverte. Par contre si la période est trop petite par rapport à la dynamique du système, la fonction de transfert échantillonnée obtenue, risque de ne plus représenter la réalité physique du système et pousse ce dernier à tendre vers une fonction de transfert d'intégrateur (des explications sont données au paragraphe II.2.4).

Un compromis peut être obtenu en utilisant le suréchantillonnage [KONCAR 1991]. **Le principe consiste en l'utilisation de plusieurs périodes d'échantillonnages différentes dans le système de contrôle.**

Depuis quelques années, plusieurs méthodes de commande avec suréchantillonnage ont été développées. Des régulateurs utilisant ce principe sont

étudiés depuis 1957 par plusieurs chercheurs. Citons à titre d'exemple Kranc 1957, Chammas et Leondes [CHAMAS 1978], [CHAMAS 1979], Araki et Hagiwara [HAGIWARA 1986], [HAGIWARA 1988]. L'intérêt de ces méthodes est l'amélioration de l'asservissement et de la stabilité du système de contrôle. Ainsi, il est souvent possible de stabiliser les systèmes non stables par l'utilisation de tels régulateurs.

Dans toute la suite de cette étude les algorithmes présentés utilisent le suréchantillonnage.

II.2.2) METHODES DE COMMANDE UTILISANT LE SURECHANTILLONNAGE

D'une façon générale, toute méthode de commande utilisant ce principe comporte plusieurs périodes d'échantillonnage différentes pour la commande et pour la mesure des valeurs de sortie du système. Une première possibilité est de mesurer les valeurs de la sortie du système à une cadence supérieure à la cadence d'envoi de la commande à l'entrée de ce système. Une seconde forme est d'envoyer la commande à une cadence supérieure à la cadence de mesure des valeurs de sortie.

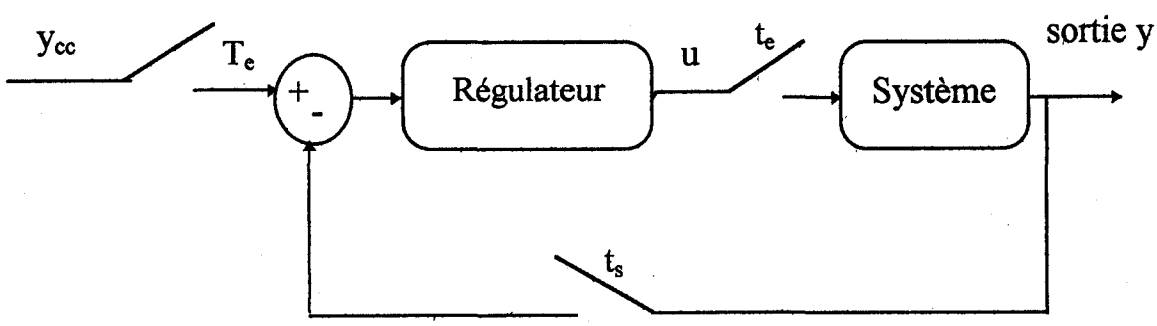


Fig II.1 Commande avec utilisation du principe de suréchantillonnage

Avant de passer à l'application de ce principe à la commande prédictive, il est important de donner la définition suivante :

Définition

On appelle multiplicités d'entrée ou de sortie, les nombres n et n_1 définis par :

$$n=T_e/t_e$$
$$n_1=T_e/t_s$$

t_e et t_s désignent les périodes d'échantillonnage respectivement d'entrée et de sortie et T_e étant la période d'échantillonnage de référence.

Les figures ci-dessous montrent l'évolution de la commande et de la sortie suivant leur multiplicité. Ce type de commande est appelé « Multirate Output Control ».

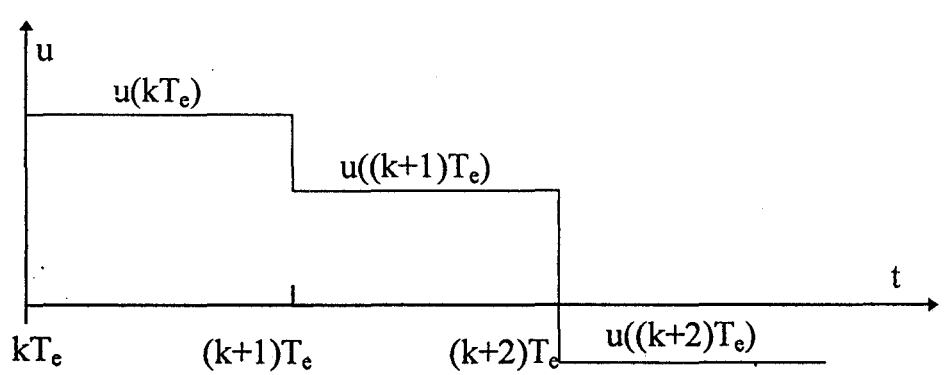


Fig II.2 Evolution de la commande avec n=1 (t_e=T_e)

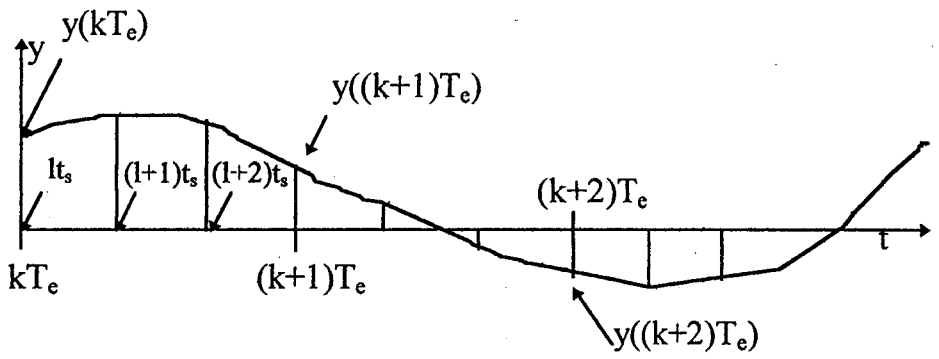


Fig II.3 Evolution de la sortie avec $n_1=3$ ($t_s=T_e/3$)

II.2.3) CHOIX DE LA PERIODE D'ECHANTILLONNAGE

Le choix de la période d'échantillonnage tel qu'il a été présenté en introduction s'avère très important. Généralement une période aussi petite que possible permet non seulement d'améliorer la précision et le temps de réponse mais aussi de remédier aux effets de perturbations lorsque le système suit une consigne variable dans le temps.

Toutefois il convient de tenir compte des limites du système, mais ceci n'empêche pas d'être confronté à divers problèmes tels que :

* le temps de calcul : le calcul de la commande doit pouvoir se faire à l'intérieur d'une même période d'échantillonnage (contrôle en temps réel).

* la tendance vers un intégrateur : en effet, si nous représentons par s_i les pôles du système continu, z_i les pôles du système échantillonné, ces pôles sont liés par la relation $z_i = \exp(s_i T_e)$. Par conséquent si T_e tend vers 0, alors z_i tend vers 1. Ce qui caractérise un intégrateur.

Le suréchantillonnage a pour but de remédier à ces problèmes tout en conservant les avantages apportés par une période d'échantillonnage assez faible.

II.2.4) COMMANDE PREDICTIVE AVEC SURECHANTILLONNAGE (MPC/MRM)

V. Koncar et C. Vasseur ont introduit la notion de suréchantillonnage dans la commande prédictive à modèles de références multiples [KONCAR 1994]. Nous noterons cette méthode MPC/MRM (Multirate Predictive Control with Multiple Reference Model). La particularité de leur approche est de calculer le critère quadratique modifié noté J_m, à chaque période t_e ($t_e<T_e$) appelé période de suréchantillonnage. La commande du système est ensuite déterminée par minimisation de ce critère. Les paramètres de la commande ainsi que les horizons de prédiction N_y et N_u utilisés dans ce critère, sont définis quant à eux sur la base de la période d'échantillonnage initiale T_e. Ainsi, le critère J_m est présenté sous la forme :

$$J_m(lt_e,N_y,N_u) = \sum_{i=0}^{N_y-1} \varepsilon_{sy}^2(lt_e+(i+d)T_e) + \lambda \sum_{i=0}^{N_u-1} \varepsilon_{su}^2(lt_e+iT_e) \qquad (II.1)$$

dans lequel :

$$\varepsilon_{sy}(lt_e)=A_r(q^{-1})y(lt_e)-B^*(q^{-1})T(q^{-1})y_{cc}(lt_e) \qquad (II.2)$$

$$\varepsilon_{su}(lt_e)=A_r(q^{-1})u(lt_e)-A^*(q^{-1})T(q^{-1})y_{cc}(lt_e) \qquad (II.3)$$

selon les notations du chapitre 1 où q^{-1} est l'opérateur de décalage de t_e.

Par ailleurs, les polynômes caractéristiques du système sont notés ici A* et B* ($A^*\neq A$, $B\neq B^*$) parcequ'ils ont été calculé avec t_e.

Ensuite, selon le schéma présentée ci-dessous par Fig. II.4

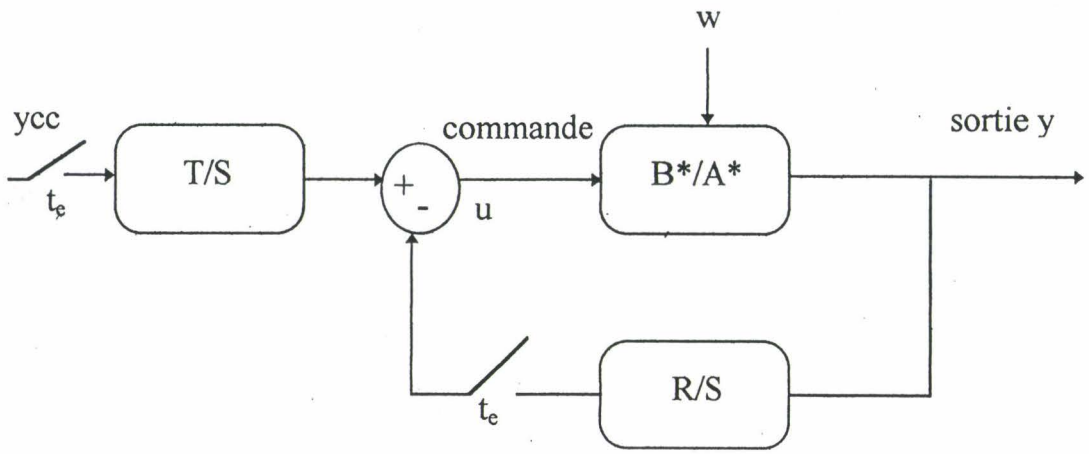


Fig II.4 Schéma de commande

nous pouvons réécrire la commande $u(lt_e)$:

$$u(lt_e) = \frac{T(q^{-1},\lambda)}{S(q^{-1},\lambda)} y_{\infty}(lt_e) - \frac{R(q^{-1},\lambda)}{S(q^{-1},\lambda)} y(lt_e) \qquad (II.4)$$

Expression dans laquelle les polynômes R, S et T dépendant de λ sont à calculer. Ainsi, la commande calculée est remise à jour et appliquée au système selon une période d'échantillonnage réelle notée t_e ($t_e<T_e$). Ceci permet de réduire le temps pendant lequel le système fonctionne en boucle ouverte et par conséquent d'améliorer les performances comme nous le montrerons dans la suite du chapitre.

II.3) COMMANDE PREDICTIVE A PARAMETRE λ AUTO-AJUSTABLE

II.3.1) INTRODUCTION

Comme nous l'avons vu précédemment la GPC est à la base de toutes les variétés existantes de commande prédictive. Néanmoins le problème majeur d'utilisation de ce type de commande est le réglage des paramètres (horizons de

prédiction, facteur de pondération, etc..). Parmi les nouvelles méthodes de contrôle aidant à trouver une stratégie de choix de ces éléments, nous trouvons :

* La commande CRHPC (Constrained Receding Horizon Predictive Control) introduite par Clarke et Scattolini [CLARKE 1991]. Cette méthode consiste à définir deux facteurs de pondération pris sous une forme exponentielle en fonction du nombre d'itérations « i » intervenant dans le critère quadratique. Le premier est associé à la somme des carrées de la différence entre la sortie et la consigne, le second remplace le facteur de pondération λ. En d'autres termes, le facteur λ varie dans le temps suivant une fonction exponentielle de la forme $\lambda(i) = \bar{\lambda}\,\alpha^{-2i}$, $\bar{\lambda} > 0$ et $\alpha \leq 1$ [CLARKE 1993].

Ces fonctions exponentielles sont utilisées pour conduire les valeurs propres du système en boucle fermée, à l'intérieur d'un cercle de rayon α [CLARKE 1994]. Cette méthode CRHPC garantit la stabilité du système.

* La commande EHPC (Extended Horizon Predictive Control) et le critère présenté par De Keyser permettant de stabiliser le système en boucle fermée. Dans ce critère l'horizon est choisi selon les caractéristiques de la réponse du système à une entrée en impulsion [DE KEYSER 1994].

* La commande UPC (Unified Predictive Control) où le facteur de pondération λ est obtenu suivant le principe de placement des pôles, avec un paramètre variant de 0 à l'infini et un horizon de prédiction variant de 1 à 10 [SOETEROBEK 1992].

II.3.2) PRINCIPE DE BASE

L'une des principales difficultés rencontrées lors de l'utilisation de la commande prédictive est le choix de ses paramètres. Le paramètre qui doit être fixé pour la commande prédictive et qui a une importance majeure sur la stabilité globale

du système, est le coefficient de pondération de la commande λ, utilisé dans le critère quadratique à minimiser.

Le choix judicieux de ce paramètre est très important car il influe directement sur le critère et par suite sur le suivi de la consigne. Cependant, un mauvais choix de ce facteur est relativement vite ressenti car il permet de donner plus ou moins d'importance à la commande vis à vis de la sortie. Habituellement, une série de mesures et d'essais, avant de lancer la commande du système est réalisée pour trouver la meilleure valeur du coefficient de pondération de commande. Néanmoins, la valeur ainsi obtenue n'est peut être pas optimale **durant tout le cycle de commande**.

Afin de rendre ce type de commande plus facile à utiliser, nous proposons dans cette partie, une méthode permettant l'ajustement automatique de ce coefficient de pondération, imposant le meilleur comportement du système suivant les contraintes imposées à chaque période d'échantillonnage.

L'idée de base consiste à imposer un suivi exponentiel à l'erreur entre la sortie réelle du système et la sortie correspondante au comportement désiré. Ceci entraîne un coefficient de pondération variant dans le temps.

Pour tester la fiabilité de notre méthode, nous l'avons appliquée à la MPC/MRM utilisée pour commander un processus thermique. Les résultats obtenus sont donnés à la fin de ce chapitre.

II.3.3) FORMULATION MATHEMATIQUE

Dans ce chapitre, le principe du suivi exponentiel présenté par L. T. Grujic [GRUJIC 1992], a été utilisé pour développer une méthode permettant l'ajustement automatique du facteur de pondération de la commande. Ce facteur est utilisé dans le critère à minimiser de la commande prédictive. Au cours de la régulation, ce paramètre

est adapté automatiquement de façon à satisfaire la condition de suivi exponentiel initialement imposée. Ceci implique une modification des paramètres du régulateur en temps réel. Le facteur λ est ajusté à chaque période d'échantillonnage T_e (sinon t_e si nous utilisons le principe du suréchantillonnage).

Avant de présenter cette méthode, il est indispensable de présenter le principe du suivi exponentiel utilisé pour la MPC/MRM. L'idée de base de ce concept, est née à partir des travaux développés dans le domaine de la commande à haut-gain pour les systèmes linéaires, par Kouvaritakis et MacFarlane [KOUVARITAKIS 1976], [KOUVARITAKIS 1978]. D'autres travaux significatifs dans le développement de la commande à haut-gain sont attribués à Porter et Bradshaw en 1979 [BRADSHAW 1979a], [BRADSHAW 1979b].

Ces derniers ont développé une méthode permettant la synthèse d'une commande à haut-gain garantissant un suivi asymptotique. Cette méthode consiste en la définition d'un algorithme de commande qui ne requiert pas de connaissance sur le système et/ou son modèle mathématique. Toutefois, le système à commander doit être contrôlable et observable.

Il a apparue intéressant de synthétiser une commande qui garantie une poursuite de haute qualité. En effet, ce sont les diverses propriétés de poursuites dans un temps fini et non fini qui sont étudiés dans le cadre des systèmes linéaires stationnaires, non stationnaires et continus avec une seule sortie. Il a été introduit et défini le principe de poursuivibilité naturelle. Ce principe ouvre le problème de la synthèse de la commande correspondante qui sera nommée commande naturelle. La commande naturelle à haut-gain est synthétisée pour garantir le suivi exponentiel dans un temps fini et non fini. Nous référant aux travaux de Grujic [GRUJIC 1992], nous acceptons le lemme suivant:

lemme :

Si la commande u(t) est définie de sorte que l'erreur du système e(t) obéit à l'équation suivante :

$$\forall\ t \in \Re \quad K_1 \frac{de(t)}{dt} + K_0 e(t) = 0 \qquad (II.5)$$

où,

$$K_1>0,\ K_0>0 \qquad (II.6)$$

alors le système présente une poursuite exponentielle, c'est à dire :

$$e(t) = e(0) \exp\left(-\frac{K_0}{K_1} t\right) \qquad (II.7)$$

Cependant, la poursuite et la stabilité sont deux propriétés généralement indépendantes. Le système peut être instable et présenter une capacité de poursuite exponentielle.

II.3.4) AJUSTEMENT AUTOMATIQUE

Dans l'optique d'un ajustement du facteur de pondération λ, nous réécrivons la commande (II.4) sous la forme :

$$u(lt_e) = \frac{T(q^{-1},\lambda(lt_e))}{S(q^{-1},\lambda(lt_e))} y_{cc}(lt_e) - \frac{R(q^{-1},\lambda(lt_e))}{S(q^{-1},\lambda(lt_e))} y(lt_e) \qquad (II.8)$$

où λ est fonction de lt_e. Définissons par ailleurs l'erreur de sortie comme étant :

$$e(lt_e)=y_{cc}(lt_e)-y_d(lt_e) \qquad (II.9)$$

où y_d est la trajectoire désirée, correspondant à l'erreur qui satisfait à la condition de poursuite exponentielle et u_d désigne la commande souhaitée générant la sortie y_d. La commande u_d est obtenue par la minimisation du critère correspondant J_d.

A partir de l'équation (II.7), nous obtenons l'expression de l'erreur de sortie définie par :

$$e(lt_e) = e(0)\exp(-\frac{K_0}{K_1}lt_e) \tag{II.10}$$

en combinant ces deux dernières équations, la sortie $y_d(lt_e)$ est donnée par l'expression :

$$y_d(lt_e) = y_{cc}(lt_e) - e(0)\exp(-\frac{K_0}{K_1}lt_e) \tag{II.11}$$

La sortie $y_d(lt_e)$ permet de respecter la condition de poursuite exponentielle. Cette sortie est obtenue à partir de la loi de commande $u_d((l-1)t_e)$, calculée par la minimisation du critère $J_d((l-1)t_e, N_y, N_u)$, ce critère étant lui même défini par le facteur de pondération $\lambda_d((l-1)t_e)$.

$$J_d(lt_e, N_y, N_u) = \sum_{i=0}^{N_y-1}\varepsilon_{dsy}^2(lt_e + (i+d)T_e) + \lambda\sum_{i=0}^{N_u-1}\varepsilon_{dsu}^2(lt_e + iT_e) \tag{II.12}$$

avec :

$$\varepsilon_{dsy}(lt_e)=A_r(q^{-1})y_d(lt_e)-B(q^{-1})T(q^{-1})y_{cc}(lt_e) \tag{II.13}$$

$$\varepsilon_{dsu}(lt_e)=A_r(q^{-1})u_d(lt_e)-A(q^{-1})T(q^{-1})y_{cc}(lt_e) \tag{II.14}$$

Pendant le contrôle du procédé, lorsque $t=lt_e$, la sortie réellement mesurée $y(lt_e)$ est obtenue à partir de la commande $u((l-1)t_e)$, calculée par la minimisation du critère $J((l-1)t_e, N_y, N_u)$, ce critère étant défini avec le coefficient de pondération $\lambda((l-1)t_e)$.

Cependant, nous calculons l'erreur entre la sortie souhaitée obéissant à la condition de poursuite exponentielle, et la sortie réelle mesurée. Cette erreur est donnée par l'expression :

$$ee(lt_e)=y_d(lt_e)-y(lt_e) \qquad (II.15)$$

A ce stade, le test d'optimisation est déclenché :

Si $ee(lt_e)=0$ Nous considérons que le suivi exponentiel est réalisé et la valeur optimale du facteur de pondération λ est atteinte. Donc :

$$\lambda(lt_e)=\lambda((l-1)t_e)$$

NB : Du fait que les instants $(l-1)t_e$ et lt_e sont suffisamment rapprochés, nous pouvons substituer la valeur $\lambda((l-1)t_e)$ à $\lambda(lt_e)$.

Sinon, $ee(lt_e)\neq 0$: Le suivi exponentiel n'est pas encore réalisé, la valeur du facteur λ n'est donc pas optimale.

Le paragraphe suivant donne une méthode de recherche de λ optimale.

II.3.5) OPTIMISATION DU FACTEUR DE PONDERATION λ

A partir de l'équation (I.1) représentant le modèle du système, le cas idéal est obtenue lorsque :

$$A(q^{-1})\, y(lt_e) = B(q^{-1})\, u(lt_e) \qquad (II.16)$$

c'est à dire, les perturbations seraient annulées, nous pouvons écrire :

$$(1+\sum_{i=1}^{n} a_i q^{-i}) y(lt_e) = \sum_{i=1}^{m} b_i q^{-i-d} u(lt_e) \qquad (II.17)$$

donc,

$$y(lt_e) = b_1 u((l-1-d)t_e) + \sum_{i=2}^{m} b_i u((l-i-d)t_e) - \sum_{i=1}^{n} a_i y((l-i)t_e) \qquad (II.18)$$

la valeur optimale de λ est obtenue lorsque :

$$y_d(lt_e) = y(lt_e) \qquad (II.19)$$

Par ailleurs, $y_d(lt_e)$ est donnée par l'expression suivante :

$$y_d(lt_e) = b_1 u_d((l-1-d)t_e) + \sum_{i=2}^{m} b_i u_d((l-i-d)t_e) - \sum_{i=1}^{n} a_i y((l-i)t_e) \qquad (II.20)$$

où les valeurs de $y((l-i)t_e)$, pour i=1, .., n sont mesurées et u_d est le signal de commande permettant l'obtention de la sortie y_d, donc nous avons :

$$u_d((l-1-d)t_e) = b_1^{-1}[y_d(lt_e) - \sum_{i=2}^{m} b_i u_d((l-i-d)t_e) - \sum_{i=1}^{n} a_i y((l-i)t_e)] \qquad (II.21)$$

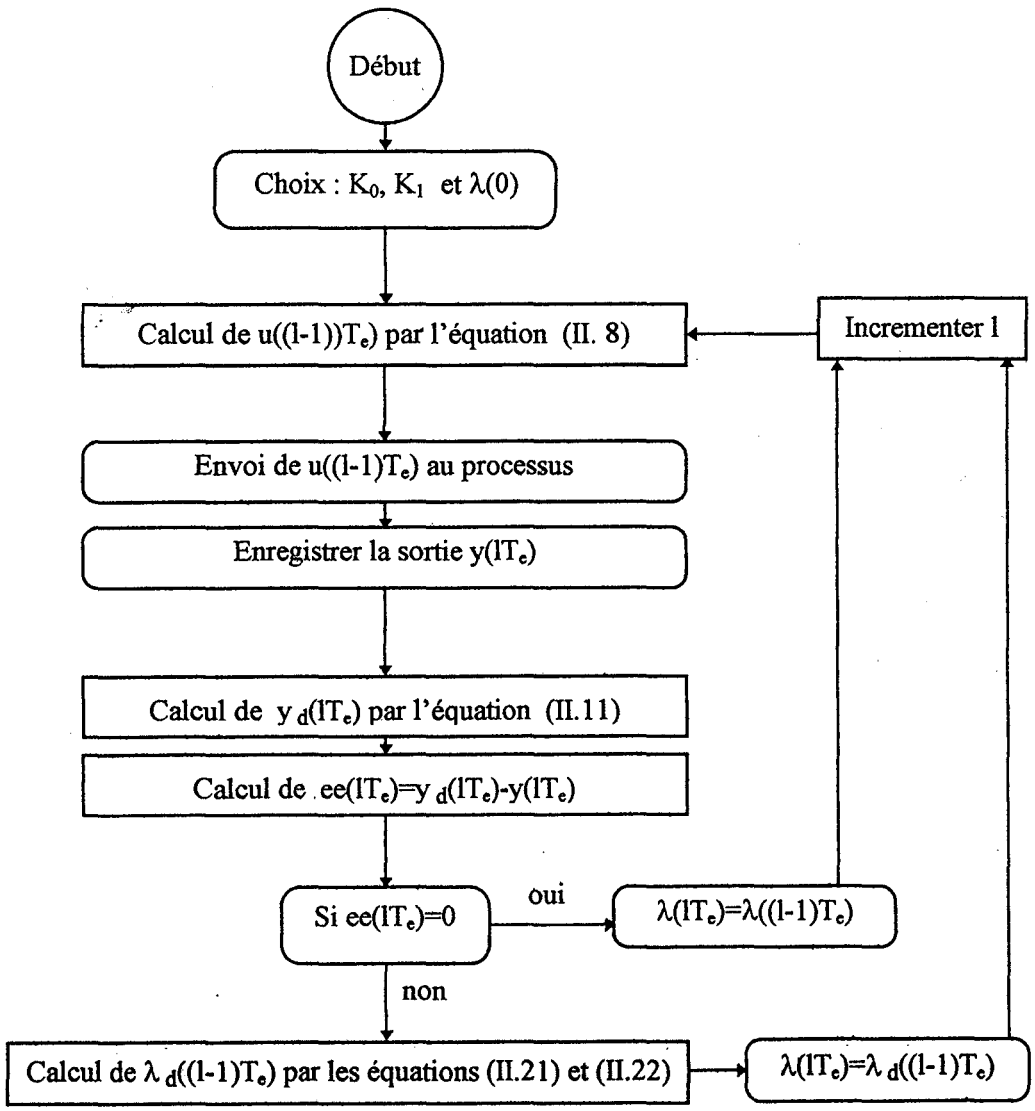
Or, la commande $u_d((l-1-)t_e)$ est calculée suivant le principe de la commande prédictive multiéchantillonnée à modèles de références multiples par l'expression suivante :

$$u_d((l-1-d)t_e) = \frac{T(q^{-1}, \lambda_d((l-1)t_e))}{S(q^{-1}, \lambda_d((l-1)t_e))} y_\infty((l-1)t_e) - \frac{R(q^{-1}, \lambda_d((l-1)t_e))}{S(q^{-1}, \lambda_d((-ll)t_e))} y((l-1)t_e) \qquad (II.22)$$

En résolvant les équations (II.21) et (II.22), nous déterminons la valeur optimale de λ_d ((l-1)t_e). Cette valeur de λ_d((l-1)t_e) permet d'avoir la commande $u_d(lt_e)$ qui à son tour génère la sortie $y_d(lt_e)$. Cette dernière assure le suivi exponentiel.

La MPC/MRM impose la minimisation du critère à chaque période d'échantillonnage t_e, choisie pour respecter la dynamique du système. De ce fait, nous pouvons supposer que la valeur du paramètre λ_d((l-1)t_e) vu par le processus à l'instant t=lt_e, peut être considérée comme valeur optimale. Donc, à t=lt_e, le critère est minimisé et la commande est calculée suivant $\lambda(lte)=\lambda_d((l-1)t_e)$.

II.3.6) ORGANIGRAMME DE LA MPC/MRM A PARAMETRE λ AUTO-AJUSTABLE



II.4) DESCRIPTION DU PROCESSUS THERMIQUE

Afin de valider notre étude, nous avons appliqué la MPC/MRM à paramètre de pondération λ auto-ajustable à un processus thermique afin d'en assurer l'asservissement en température.

Ce banc d'essai est composé des éléments suivants :

- Une cuve en Plexiglas contenant de l'eau.
- Une résistance chauffante de 1000 W plongée dans l'eau. Son temps de fonctionnement (chauffage) est entièrement déterminé par la commande.
- Une sonde thermocouple pour mesurer la température du bain qui envoie ses informations à l'ordinateur à travers une carte d'interface avec convertisseur analogique-numérique.
- Un agitateur pour mélanger l'eau et homogénéiser la température dans le bain.

La figure II.10 ci-dessous illustre ce banc d'éssai.

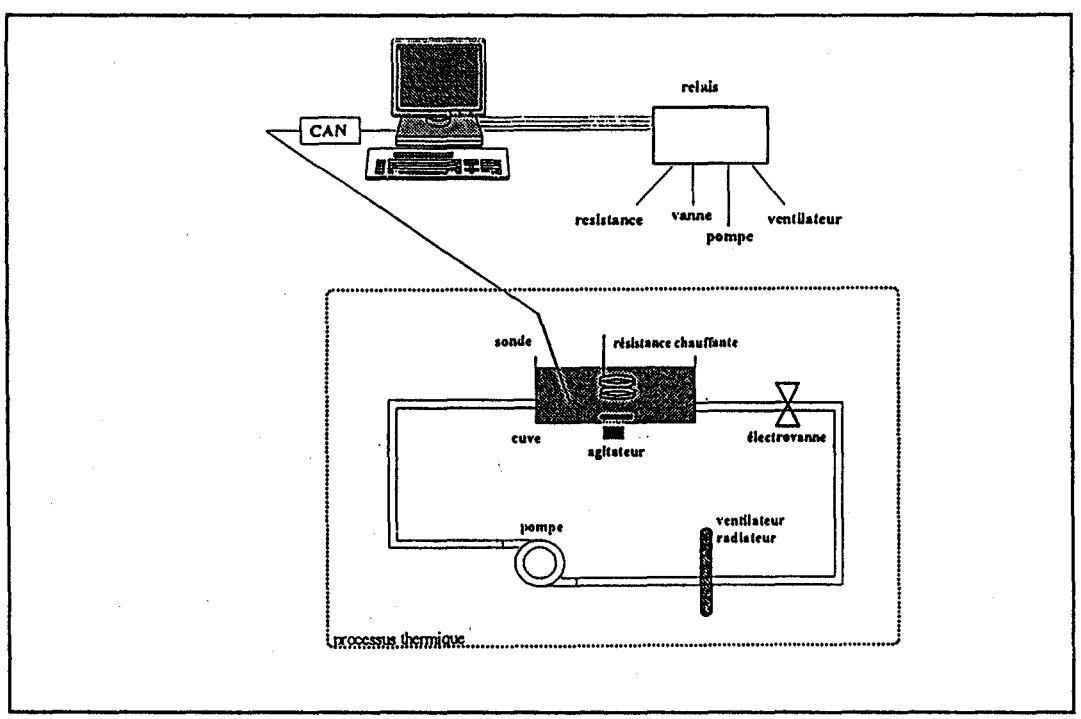


Fig II.10 Processus thermique.

Le refroidissement est assuré par la circulation du bain dans un circuit composé d'une pompe qui envoie l'eau dans un radiateur avec un ventilateur.

La commande ainsi que le temps de fonctionnement de la pompe, du ventilateur et de la résistance chauffante, sont gérés par ordinateur. En ce qui concerne la température, la commande calculée par le programme est transmise en modulation de largeur d'impulsion, c'est à dire :

* si $u>0$ on chauffe.
* si $u<0$ on refroidit.

Par exemple :

* si la commande vaut $u_{max}/2$, on chauffe pendant le temps $T_e/2$.
* si la commande vaut $-u_{max}/4$, on refroidit pendant le temps $T_e/4$.

Pour notre système d'échantillonnage, la commande u est bornée en fonction de T_e.

$$-u_{max} < u < u_{max}$$

Ce processus thermique, en réalité non linéaire, est ici modélisé par la simple fonction de transfert suivante :

$$H(q^{-1}) = \frac{B(q^{-1})}{A(q^{-1})} = \frac{\beta q^{-1}}{1+\alpha_1 q^{-1}+\alpha_2 q^{-2}} \qquad \text{(II.23)}$$

avec :

$\alpha_1=-1.6065$ (II.24)

$\alpha_2=0.6065$ (II.25)

$\beta=0.3935$ (II.26)

Le chapitre 3 présente une étude par rapport au choix de la dynamique désirée, ce qui nous a conduit à la définir par :

$A_r(q^{-1})=1+a_1q^{-1}+a_2q^{-2}$ (II.27)

avec :

$a_1=0.84$ (II.28)

$a_2=0.1864$ (II.29)

donc, les racines de A_r sont :

$\lambda_1=-0.42+0.1*j$ (II.30)

$\lambda_2=-0.42-0.1*j$ (II.31)

Les horizons de prédiction de la commande et de la sortie sont données par les expressions suivantes :

$N_u=2$ (II.32)

$N_y=3$ (II.33)

Avec les périodes d'échantillonnage $T_e=10s$ et $t_e=1s$ et après résolution des équations diophantines, nous obtenons :

$$u(lt_e)=\frac{-RQ}{(1+a_1q^{-1}+a_2q^{-2})}y_\infty(lt_e)-\frac{X+Yq^{-1}+Zq^{-2}+Wq^{-3}}{(1+a_1q^{-1}+a_2q^{-2})}y(lt_e) \quad \text{(II.34)}$$

avec,

$V=(\beta^2+\alpha)(\beta^2+\alpha_1{}^2\beta^2+\lambda)-\alpha_1{}^2\beta^4$ (II.35)

$$X=(\beta^2+\lambda)\beta\alpha_1\alpha_1\beta\lambda(\alpha_1^2-\alpha_2^2))/V \quad (II.36)$$

$$Y=((\beta^2+\lambda)\beta\alpha_2+\alpha_1(\beta^2+\lambda)\beta\alpha_1+\alpha_1^2\alpha_2\beta\lambda+\alpha_1 a_1\beta\lambda(\alpha_1^2-\alpha_2))/V \quad (II.37)$$

$$Z=(\alpha_1 a_2\beta(\beta^2+\lambda)-\alpha_2 a_1\beta(\beta^2+\lambda)+\alpha_1^2 a_2 a_1\beta\lambda+\alpha_1\beta\lambda a_2(\alpha_1^2-\alpha_2))/V \quad (II.38)$$

$$W=(\alpha_2 a_2\beta(\beta^2+\lambda)\alpha_1^2\alpha_2 a_2\beta\lambda)/V \quad (II.39)$$

$$Q=(\beta^2+\lambda)\beta^2(\alpha_1+\alpha_2)+\alpha_1\beta^2\lambda(\alpha_1+\alpha_2)-\alpha_1^2\beta\lambda(1+\alpha_1+\alpha_2)-V(1+\alpha_1+\alpha_2) \quad (II.40)$$

$$R=V(X+Y+Z+W)/Q \quad (II.41)$$

les polynômes $R(q^{-1})$, $S(q^{-1})$ et $T(q^{-1})$ sont définis selon les relations suivantes:

$$\frac{T(q^{-1},\lambda)}{S(q^{-1},\lambda)}=\frac{-RQ}{V(1+a_1q^{-1}+a_2q^{-2})} \quad (II.42)$$

et

$$\frac{R(q^{-1},\lambda)}{S(q^{-1},\lambda)}=\frac{X+Yq^{-1}+Zq^{-2}+Wq^{-3}}{(1+a_1q^{-1}+a_2q^{-2})} \quad (II.43)$$

II.5) PERFORMANCES

* Les essais ont été réalisés en temps réel, avec une période d'échantillonnage T_e=10s. Le facteur de pondération λ a été initialement fixé à 0.1. La consigne de température a été fixée à 32°C durant le contrôle du procédé.

* En ce qui concerne le suivi exponentiel, nous avons choisi K_0/K_1=0.02. Cette valeur a été choisie suite à une série de mesures, et correspond à la meilleure réponse du système par rapport à sa dynamique.

* Les perturbations sont dues à la non homogénéité du liquide, surtout dans la zone proche du capteur de température, ce qui a engendré de grandes variations du signal. Cependant ces imperfections ont pu être maîtrisées grâce à l'ajustement automatique de λ et à la rapidité de la réponse de l'ordinateur.

Les courbes présentées ci-dessous, ont été réalisées sans suréchantillonnage.

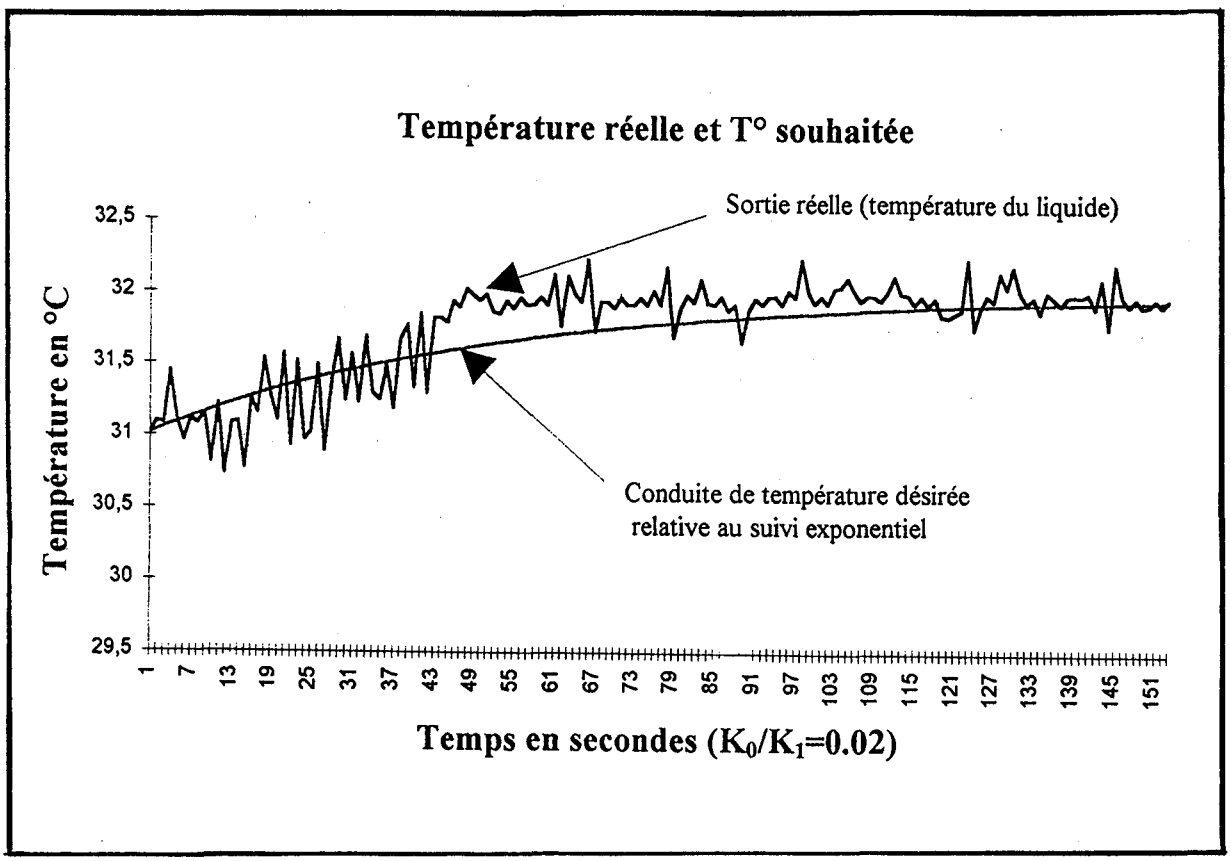


Fig II.11 Evolution de la température sans suréchantillonnage.

La Fig II.12 illustre la variation du facteur de pondération de la commande λ.

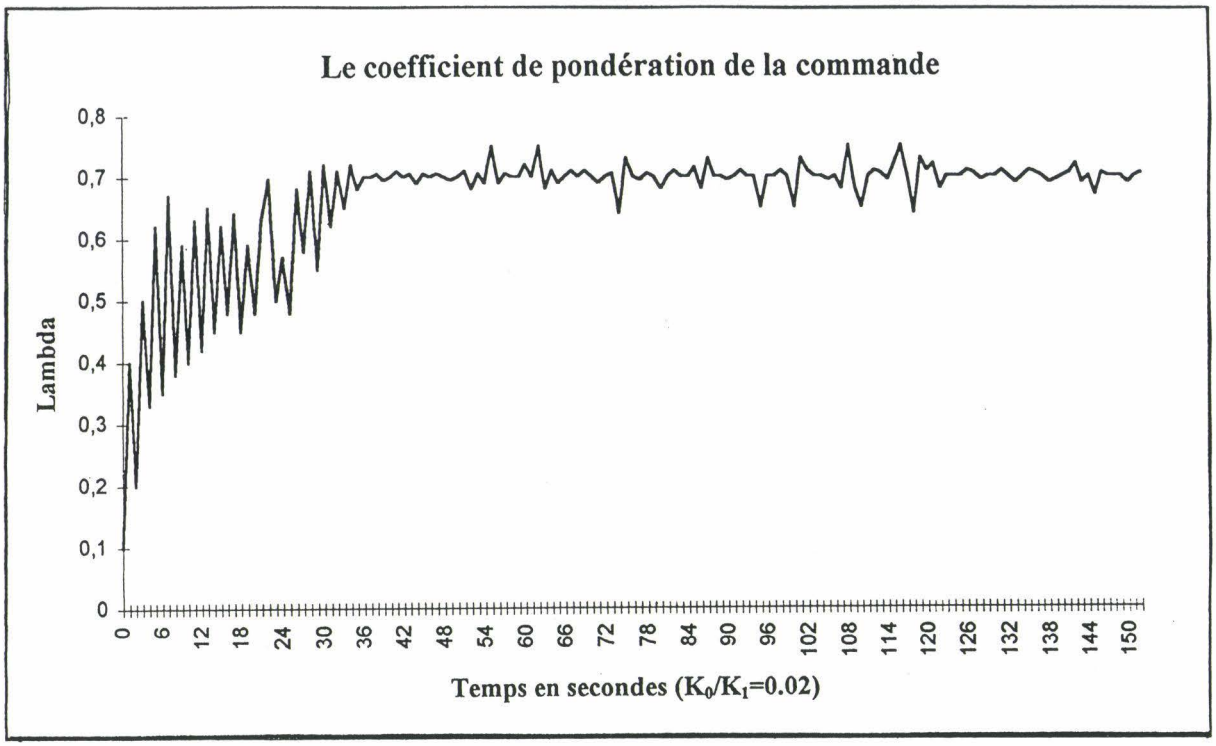


Fig II.12 Variation du facteur de pondération λ, sans suréchantillonnage.

COURBES AVEC SURECHANTILLONNAGE

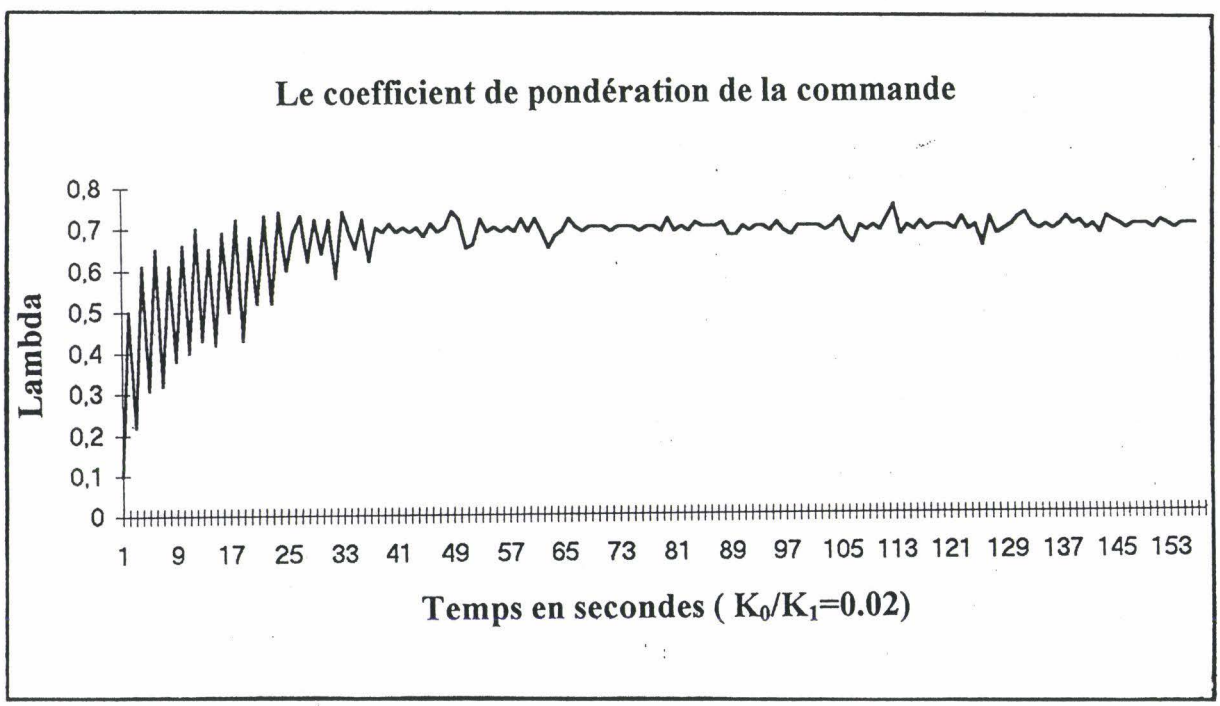


Fig II.13 Variation du facteur de pondération λ, avec suréchantillonnage.

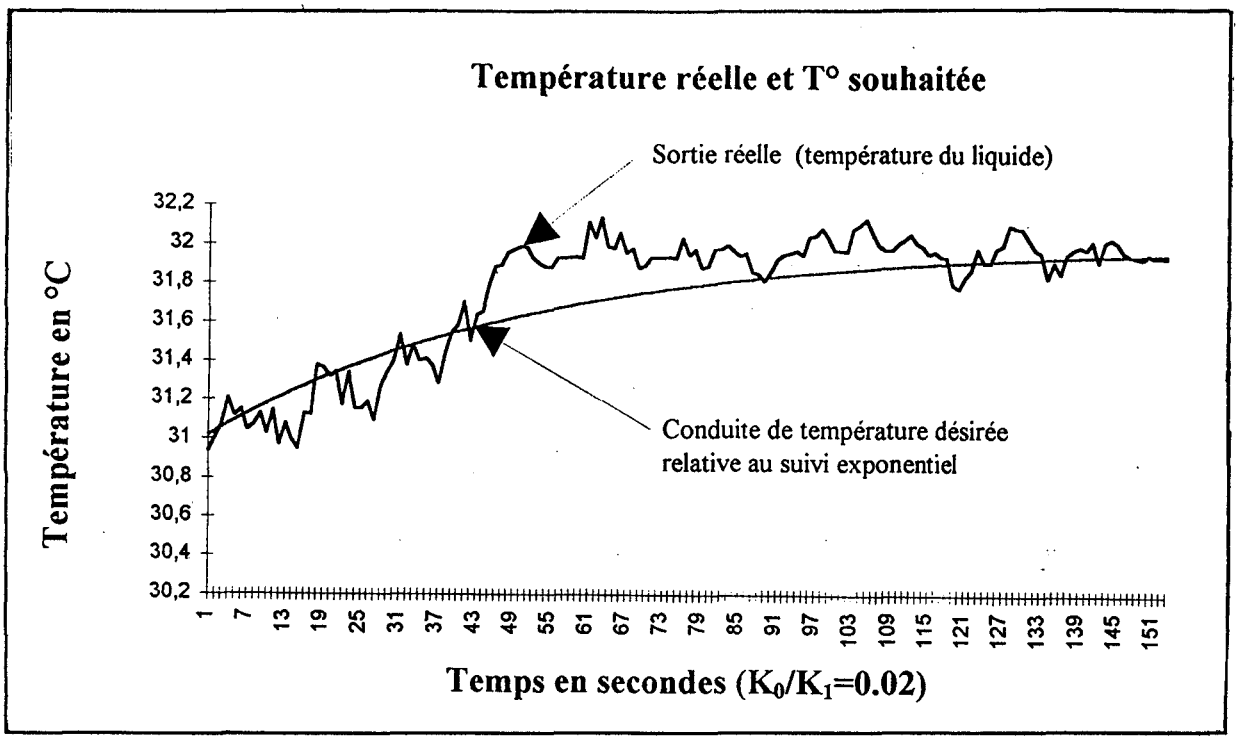


Fig II.14 Evolution de la température avec suréchantillonnage.

La figure II.15 montre l'évolution de la commande au cours du procédé de contrôle. L'essai est réalisé avec suréchantillonnage.

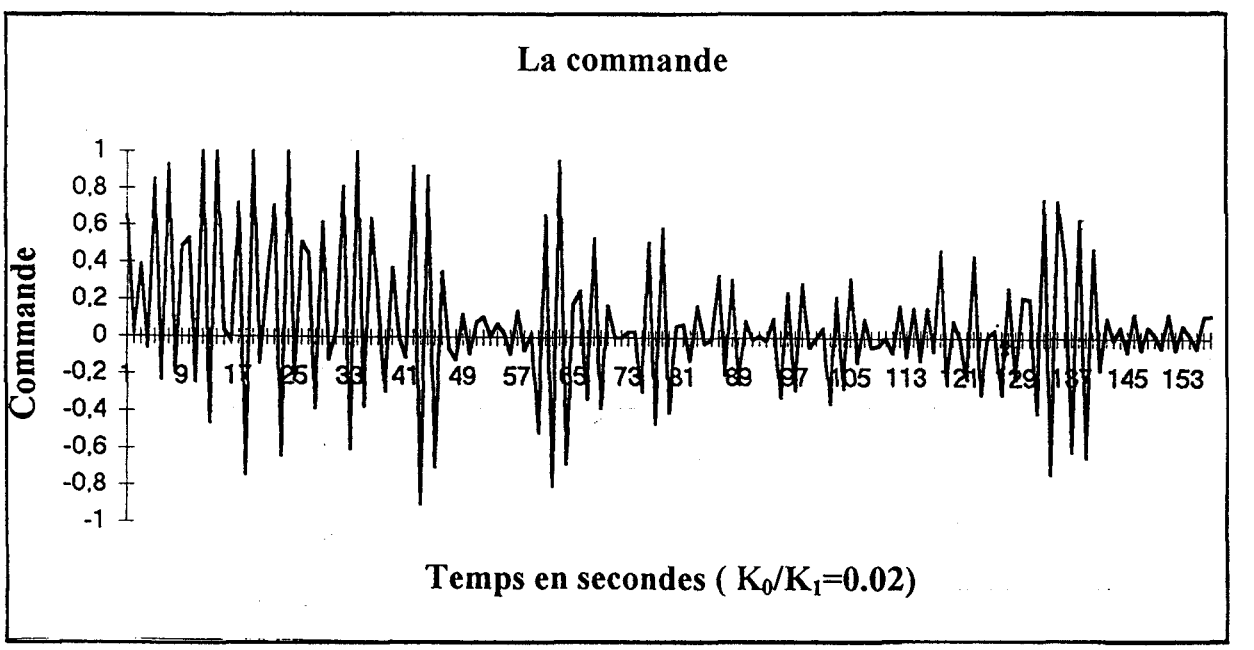


Fig II.15 Evolution de la commande u, avec suréchantillonnage.

Afin de mieux comprendre l'importance de la variation du facteur de pondération λ, au cours de la commande du procédé thermique, nous avons trouvé qu'il est intéressant de présenter les courbes d'asservissement en température réalisées avec une valeur de λ fixe. La consigne a été fixée à 45°C.

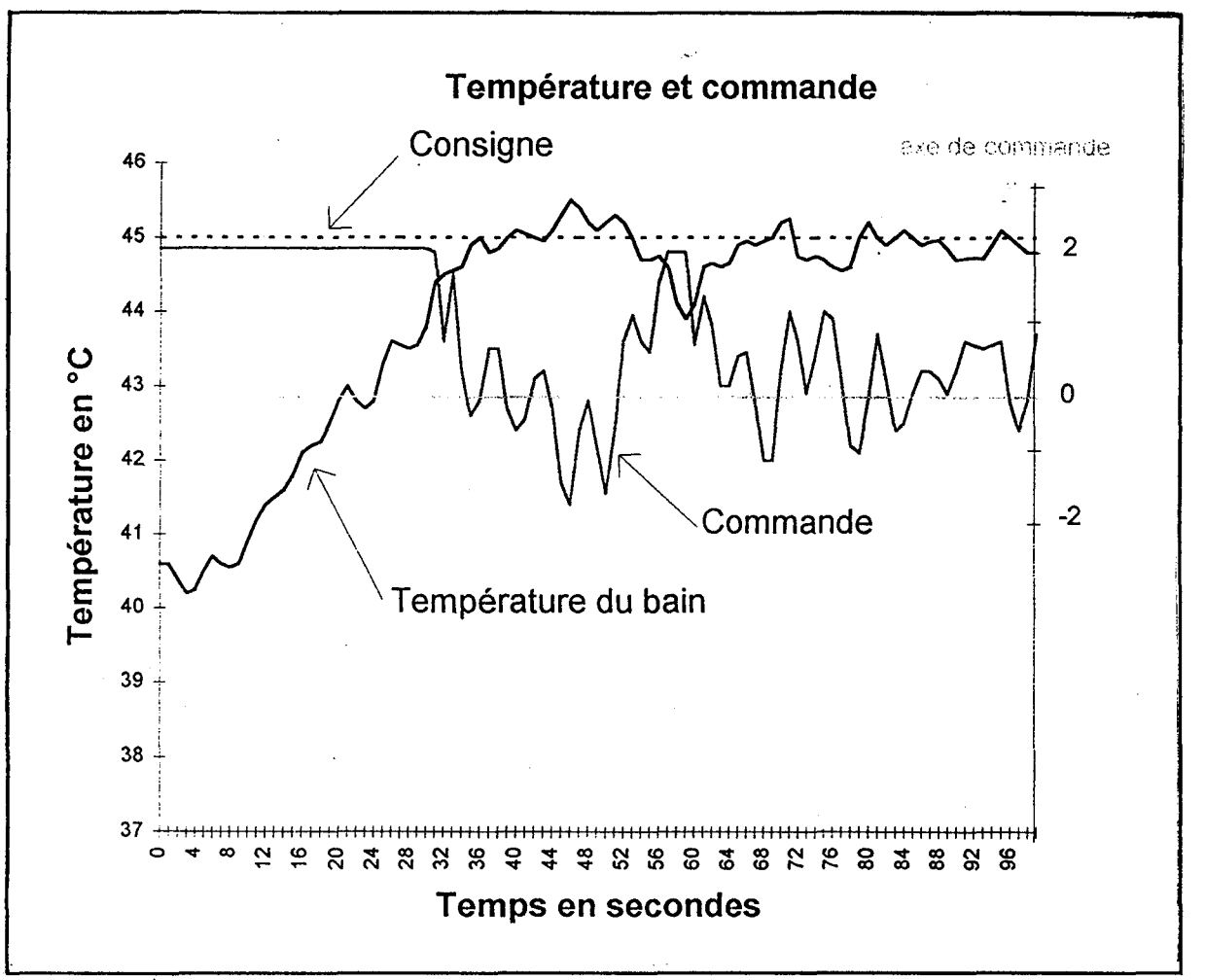


Fig II.16 Evolution de la température avec λ **fixe** au cours du procédé.
Sans suréchantillonnage.

Nous avons réalisé les essais de commande du processus thermique avec suréchantillonnage et λ fixe au cours du procédé. La figure II.17 montre l'évolution de la sortie du système. La consigne étant fixée à 39.5°C.

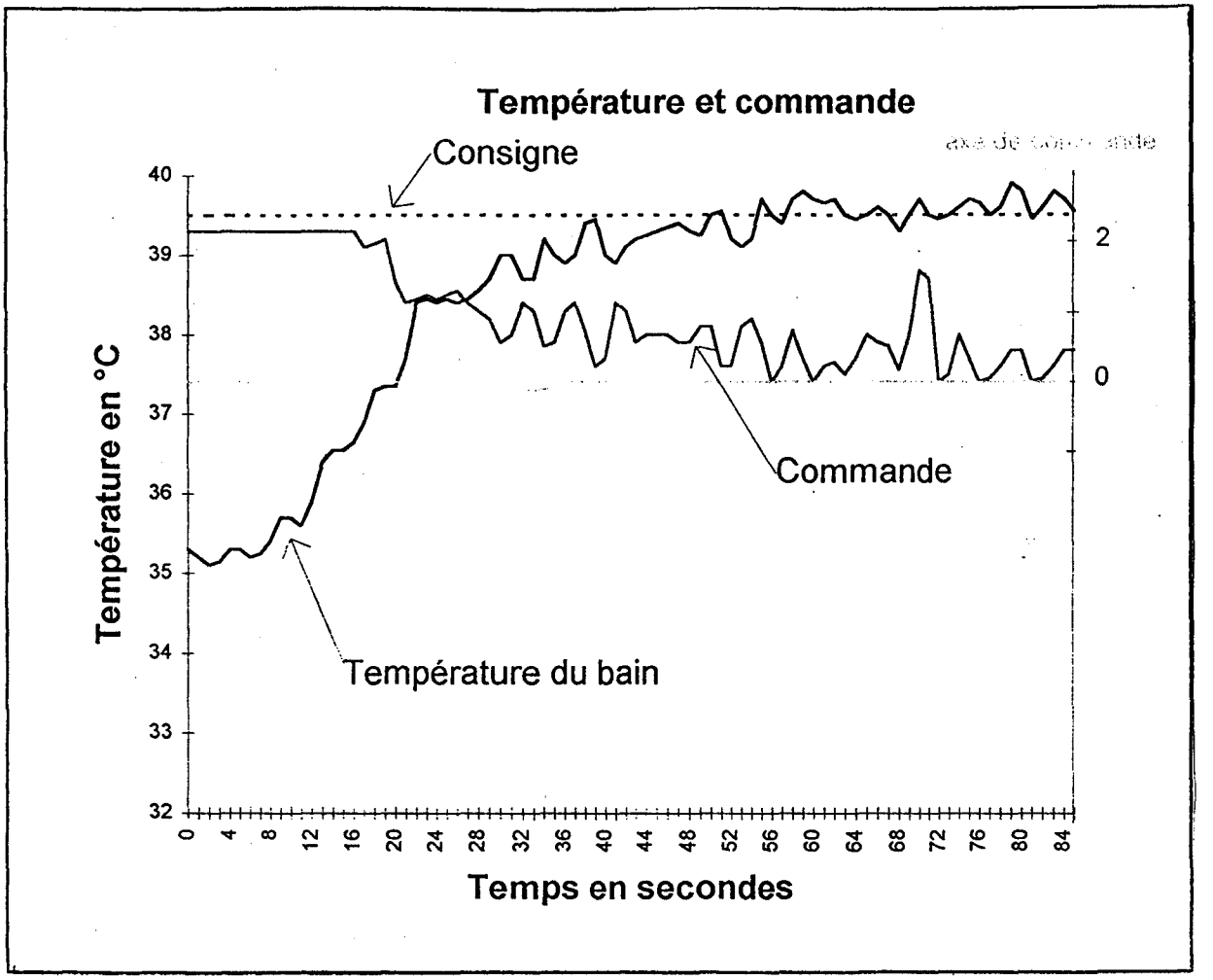


Fig II.17 Evolution de la température avec λ **fixe** au cours du procédé.

Avec suréchantillonnage.

II.9) INTERPRETATIONS DES RESULTATS

Les courbes présentées ci-dessus permettent d'établir les remarques suivantes :

* Commençons tout d'abord par mettre en évidence l'importance du suréchantillonnage dans le contrôle du procédé. Il est clair qu'en augmentant la fréquence d'échantillonnage, nous arrivons mieux à gérer l'évolution du système (voir les courbes aux Fig II.11 et Fig II.14).

La courbe de la figure II.11, montre des fluctuations de la sortie réelle assez importantes autour de la conduite de température désirée. L'amplitude maximale de ces variations est de l'ordre de 0.5°C. Tandis que la courbe de la figure II.14 présente des fluctuations d'une amplitude maximale de 0.3°C. Ainsi, la sortie du système a tendance à mieux suivre la courbe fixée par le suivi exponentiel. Nous remarquons que lorsque le système détecte l'arrivée de la sortie réelle à la consigne, il refroidit, ensuite il réchauffe à nouveau. La durée séparant les deux pics est de l'ordre de 5 secondes, le processus est donc bien contrôlé par le régulateur. La montée de température (qui parait relativement brusque vers les 50 secondes) est due à la prédiction.

Le suréchantillonnage confère au système une robustesse plus importante vis à vis des bruits (des fluctuations moins importantes) et lui permet d'être contrôlé plus souvent ce qui entraîne une meilleure maîtrise du système (voir les courbes des figures II.16 et II.17 réalisées avec λ fixe, il n'y a pas de chute de température vers les 57s).

Les courbes des figures II.14 et II.17, montrent des **variations** de la température de sortie réelle qui sont de l'ordre de **0.4°C** pour les essais réalisés avec λ **auto-ajusté** et de **1.5°C pour λ fixe**. Ainsi, la sortie enregistrée avec λ adapté automatiquement, suit mieux la consigne que celle réalisée avec le facteur λ fixe durant le déroulement du procédé.

* Remarquons que la commande prend une valeur plus élevée lorsque la sortie commence à s'écarter de la courbe consigne (vers les 60s). De la même façon, λ prend aussi une valeur plus grande, ce qui permet de donner plus d'importance à la commande dans le critère quadratique. Ces deux grandeurs (λ et la commande) vont gérer l'erreur entre la sortie réelle et la référence. Le résultat est visible vers les 70s où la sortie s'approche plus de la consigne, la commande et le coefficient λ reprennent des valeurs moins élevés.

II.7) CONCLUSION

La méthode présentée ci-dessus, permet non seulement une utilisation plus simple de la commande prédictive, mais aussi un meilleur suivi de la consigne souhaitée. De plus, l'ajustement automatique du facteur de pondération permet de palier les problèmes de perturbations éventuels durant l'évolution, en donnant plus de poids à la commande. Ainsi, l'association du suréchantillonnage avec l'ajustement automatique de λ permet d'élaborer une commande prédictive dotée d'une robustesse supérieure à une commande prédictive classique.

Dans le domaine de la teinture, la maîtrise de l'asservissement en température est très importante. En effet, le non respect du profil de température imposé est susceptible d'entraîner des pertes de matière très importantes. Par exemple, pour l'acrylique, un dépassement de la température préconisée peut entraîner un changement de phase (passage de l'état de fil à l'état de plastique, chose déjà vue en entreprise). La méthode présentée dans ce chapitre, ne permet pas à l'erreur entre la sortie réelle et la consigne s'amplifier, car le régulateur réagit automatiquement en augmentant la valeur de λ ce qui permet à la commande d'agir de façon plus efficace.

REFERENCES

[Aström 1984]:
Aström K. J, Wittermark B.
« Computer Controled Systems, Theory and Design », Prentice Hall Englenwood Cliffs, New York 1984.

[Bradshaw 1979a]:
Bradshaw A, Porter B.
« Asymptotic Properties of Linear Multivariable Continuous-time Tracking Systems Incorporated High-Gain Error-actuated Controllers », Int. J. Systems Sci 1979, Vol. 10, pp 1433-1444.

[Bradshaw 1979b]:
Bradshaw A, Porter B.
« High-Gain Error-actuated Controllers for a Class of Linear Multivariable Plants », Proc. 18th IEEE Conference on Decision and Control, 12-14 December 1979, Fort Lauderdale, FL, pp 322-325.

[Bühler 1987]:
Bühler H.
« Réglage Echantillonné, Traitement dans le domaine d'état », Vol. 2, Presse Polytechnique Romandes, Lausanne 1987.

[Chamas 1978]:
Chamas A. B, Leondes C. T.
« On the Design of Linear Time Invariant Systems by Periodic Output Feedback », Part I and Part II, « Discret Pole Assignement », Int. J. Control, Vol. 27, pp. 885-894, 895-903, 1978.

[Chamas 1979]:
Chamas A. B, Leondes C. T.
« On the Finite Time Invariant Systems by Picewise Constant Output Feedback », Int. J. Control, Vol.30, pp.227-234, 1979.

[Clarke 1991]:
Clarke D. W , Scattolini R. 1991.
« Constrained Receding Horizon Predictive Control », Proc, IEE 1991, Part. D, 138, pp 347-354.

[Clarke 1993]:
Clarke D. W , Yoon T. W. 1993.
« Receding Horizon Predictive Control with Exponential Weighting», Int. Journal. Syst. Sci 1993, N° 24, pp 1745-1757.

[Clarke 1994]:
Clarke D. W , Yoon T. W.
« Adaptive Predictive Control of the Benchmark Plant », Automatica, Vol. 30. N°4, pp 621-628, 1994.

[De Keyser 1994]:
De Keyser R. Kong F.
« Criteria for Choosing The Horizon in Extended Horizon Predictive Control », IEEE Transaction on Automatic Control, Vol. 39, N°. 7, July 1994.

[Grujic 1992]:
Grujic L. T, Mounfield W. P.
« High-gain PI Natural Control for Exponential Tracking of Linear Single-output Systems with State-space Description », APII, 1992, Vol. 26, pp 125-146.

[Hagiwara 1986]:
Hagiwara T, Araki M.
« Design of Stable State Feedback Controller Based on the Multirate Sampling of the Plant Output », IEEE Transc. On Automatic Control, Vol. 33, N°9, pp 812-819, September 1986.

[Hagiwara 1988]:
Hagiwara T, ARAKI M.
« Pole Assignment by Multirate Sampled Data Output Feedback », Int. J. Control, Vol. 44, N°. 6, pp. 1661-1673, 1988.

[Koncar 1991]:
Koncar V.
«Commande Numérique par Suréchantillonnage et Coordination en Série de sous-systèmes », Thèse de Doctorat, Université des Sciences et techniques Flandres Artois, Lille 1991.

[Koncar 1994]:
Koncar V, Vasseur C.
« Multirate Predictive Control with Multiple Reference Model, Application on DC Motor », 3rd Int. IEEE Workshop on Advanced Motion Control, Berkeley, Cal, USA, March 1994, pp. 20-23.

[Koncar 1995]:
Koncar V, Vasseur C, Bruniaux P, Pinchon D.
« Multirate Predictive Control with Multiple Reference Model Application on Thermic Process », IFAC Conference, System Structure and Control, Nantes, France, 5-7 July 1995.

[Koncar 1996]:
Koncar V, Koubaa M.A, Bruniaux P, Vasseur C.
« Predictive Control with Auto-adjustable Parameters», CESA'96 IMACS Multiconference, Symposium on Control Optimization and Supervision, Lille, France July 9-12 1996, pp 344-349.

[Kouvaritakis 1976]:
Kouvaritakis B, MacFarlane A. G. J.
Int. J. Control 1976, Vol. 23, N°. 2, pp 149-166.

[Kouvaritakis 1978]:
Kouvaritakis B.
Int. J. Control 1978, Vol. 27, N°. 5, pp 705-724.

[Lenartson 1989]:
Lenartson B.
« Multirate Sampled Data Control of Two Time Scale Systems », IEEE Trans. Automatic Control., Vol. 34., pp. 624-644, June 1989.

[Ragazini 1958]:
Ragazini J. R, Franklin G. F.
« Sampled Data Control Systems », MC Graw-Hill, New York 1958.

[Scattolini 1992]:
Scattolini R.
« Multirate Self-tuning Controller for Multivariable Systems », Int. J. System Sc.,Vol. 23, pp. 1347-1359, 1992.

[Scattolini 1994]:
Scattolini R, Schiavoni N.
« A Multirate Model based Predictive Controller », Proceedings of the 33d Conference on Decision Control, Lake Buena Vista, FL, December 1994, pp. 243-248.

[Soeterobek 1992]:
Soeterboek R.
« Predictive Control a Unified Approach », Prentice Hall International, U. K, Limited, Hemel Hempstead, U. K. 1992.

CHAPITRE 3

CHOIX DU COMPORTEMENT DESIRE, LE POLYNOME A_R

Chapitre 3

CHOIX DU COMPORTEMENT DESIRE : LE POLYNOME A_R

III.1) INTRODUCTION

Le contrôle des procédés industriels en temps réel, par les méthodes modernes de commande, nécessite le plus souvent la connaissance précise du modèle du système. Néanmoins, ces méthodes peuvent être appliquées de façon satisfaisante, tout en assurant des **comportements robustes**, surtout lorsqu'elles prévoient l'imperfection dans la modélisation globale du système ou encore la variation des paramètres du modèle du système pendant toute la durée de commande.

Depuis une trentaine d'années, un important travail a été effectué dans le domaine de la commande adaptative et de son application. L'idée essentielle de cette méthode de commande est d'arriver à contrôler en temps réel un système tout en maintenant un certain niveau de performances lorsque les paramètres du procédé sont soit inconnus, soit variant dans le temps. En effet, avec des méthodes classiques on aboutit à des performances moyennes et souvent à une **détérioration de la robustesse** du régulateur.

Pour palier cette difficulté, deux types de commande adaptative sont généralement utilisés :

- la commande adaptative directe ne comporte qu'une seule étape à chaque période d'échantillonnage qui consiste à identifier directement de manière récursive les paramètres du régulateur. Dans ce cas, nous identifions implicitement le procédé reparamètré en termes de paramètres du régulateur [NGUYEN 1989].
- la commande adaptative indirecte comporte deux étapes à chaque période d'échantillonnage. Dans un premier temps, les paramètres du modèle du procédé sont identifiés de manière récursive, puis dans un second temps, les paramètres du régulateur sont calculés à partir des paramètres estimés du procédé [LANDAU 1983].

Les schémas de commande linéaire adaptative les plus courants dans la littérature sont les suivants :

- la méthode de placement des pôles qui garde les zéros du procédé inchangés, ce qui est très intéressant dans le cas des systèmes à non minimum de phase. En effet, la simplification des zéros instables entraîne une entrée de commande non bornée. Un exemple de placement des pôles pour les systèmes linéaires multivariables est présenté dans [ELLIOT 1984].
- la méthode de placement des pôles et de zéros où nous retrouvons le schéma de commande à modèles de références [LANDAU 1981], le régulateur stochastique à variance minimale [ASTRÖM 1983] etc... Ces schémas sont basés sur la poursuite parfaite du modèle et peuvent être obtenus en minimisant un critère quadratique sur un pas d'échantillonnage.
- la commande à minimum de variance généralisée, qui est la généralisation de la commande à variance minimale résout le compromis relatif à la minimisation de la variance de l'erreur de sortie et de la variance de la variable de commande. Cette approche a été développée dans [CLARKE 1985] pour le cas monovariable, [KEVICKY 1981] pour le cas multivariable stochastique des systèmes découplables par retour d'état.
- la commande à critère quadratique. Ce type de commande est obtenu en minimisant un critère quadratique à horizon infini faisant intervenir le carré de la commande et le

carré de la différence entre la sortie et la référence. Ce genre de commande a été étudié dans sa version adaptative dans [SAMON 1982] et peut-être appliqué aux systèmes décrits par leur représentation d'état. La commande générée à travers ce principe assure la stabilité du système en boucle fermée quand le modèle de représentation d'état est stabilisable. Ce type de commande peut être approché par une commande quadratique à horizon fini.

Le fait que la dynamique du système peut être imposée par l'opérateur présente l'un des avantages de la commande prédictive. Néanmoins, le choix optimum n'est pas toujours évident et pose le problème du **compromis entre les performances et la robustesse**. Dans ce chapitre, nous présentons une méthode de choix du comportement désiré permettant d'obtenir ce compromis.

Dans un premier temps, **nous introduisons la méthode de choix du comportement désiré à la MPC/MRM**. Cette méthode permet à la commande prédictive appliquée aux systèmes discrets, **d'être plus robuste**.

Dans un second temps, nous abordons le problème **d'évaluation de la robustesse**. Enfin, nous terminons par la présentation des résultats de simulation de cette méthode appliquée au processus thermique.

III.2) ETUDE THEORIQUE DE LA METHODE

Nous nous proposons dans ce chapitre d'analyser la stabilité et d'améliorer la robustesse des systèmes discrets perturbés commandés par la MPC/MRM.

La robustesse de stabilité pour les systèmes continus a été largement étudiée dans la littérature, ce qui n'est pas le cas pour les systèmes discrets où peu de travaux ont été réalisés à ce sujet [SOH 1985], [RACHID 1989] et [RACHID 1990].

III.2.1) CHOIX DU POLYNOME A_R ET EVALUATION DE LA ROBUSTESSE

Notation :

Pour une matrice carrée M, on note :

$\lambda_i(M)$: Valeurs propres de la matrice M = Max $|\lambda_i(M)|$.

$\rho(|M|)$: Rayon spectral de la matrice M =

$|M|$: Matrice formée par le module des coefficients de la matrice M.

De plus, on note :

C(P, r): Cercle de centre P et de rayon r.

Bien que la MPC/MRM a été largement étudiée au cours des chapitres précédents, nous rappelons quelques équations nécessaires à la compréhension de la méthode. Le modèle de base du procédé est donné par l'équation (III.1) suivante :

$$A(q^{-1})y(t) = B(q^{-1})u(t) + w(t) \qquad \text{(III.1)}$$

Le principe de la méthode est de choisir dans un premier temps les polynômes $A_r(q^{-1})$ et $B_r(q^{-1})$ et de minimiser ensuite à chaque période d'échantillonnage le critère J_m. Ainsi, la dynamique du système se traduit par :

$$A_r(q^{-1})y(t) = B_r(q^{-1})y_{cc}(t) \qquad \text{(III.2)}$$

Ensuite, à partir du nouveau schéma de commande relatif à la commande prédictive (Fig II.4), nous pouvons écrire le vecteur de commande $u(lt_e)$ sous la forme :

$$u(lt_e) = \frac{T(q^{-1},\lambda)}{S(q^{-1},\lambda)} y_{cc}(lt_e) - \frac{R(q^{-1},\lambda)}{S(q^{-1},\lambda)} y(lt_e) \qquad \text{(III.3)}$$

les polynômes $R(q^{-1},\lambda)$, $S(q^{-1},\lambda)$ et $T(q^{-1},\lambda)$ sont des polynômes à calculer.

Enfin, à partir des équations (III.1), (III.2) et (III.3), nous pouvons écrire :

$$\frac{B_r(q^{-1})}{A_r(q^{-1})} = \frac{B(q^{-1}) * T(q^{-1})}{A(q^{-1}) * S(q^{-1}) + B(q^{-1}) * R(q^{-1})} \qquad \text{(III.4)}$$

Afin **d'assurer la stabilité du système malgré les perturbations** w(t) et les variations des paramètres, il est nécessaire de définir une stratégie sur le choix du polynôme $A_r(q^{-1})$.

Le système global de contrôle $\frac{y(t)}{y_\infty(t)}$ peut être représenté dans l'espace d'état (équation III.5) par la matrice F_r dont les valeurs propres sont les racines de $A_r(q^{-1})$.

$$X_{k+1}=F_rX_k \qquad \text{(III.5)}$$
$$Y_k=C^T{}_rX_k$$

Les valeurs propres $\lambda_i(F_r)$ i=1,..,n (où n est l'ordre de F_r) sont inscrites dans le cercle C(O,r) de rayon $r=\rho(|F_r|)$.

Dans le domaine des systèmes linéaires, de nombreuses méthodes de commande ont été proposées par différents auteurs, assurant une bonne stabilité et utilisant le principe de placement des pôles. Pour les systèmes discrets, A. Rachid propose un critère permettant d'améliorer la robustesse de stabilité suivant un corollaire, ramené en quelques termes, à la définition suivante :

Définition

Soit le système linéaire discret perturbé définit par :

$$X_{k+1}=(F_r+\Delta F_r)X_k \qquad \text{(III.6)}$$

Supposons la perturbation ΔF_r définie sous la forme :

$$|\Delta F_r| \leq \varepsilon |F_r| \qquad \text{(III.7)}$$

$\forall \varepsilon$, la stabilité du système est assurée par l'inéquation suivante :

$$\varepsilon < \varepsilon_r \tag{III.8}$$

avec :

$$\varepsilon_r = \frac{1-\rho(|F_r|)}{\rho(|F_r|)} \tag{III.9}$$

Suivant la condition de stabilité, le système perturbé est représenté par la matrice d'état $F_r+\Delta F_r$ (ΔF_r défini ci-dessus) avec $\Delta F_r = \varepsilon F_r$.

Notons $A'_r(q^{-1})$ le polynôme associé à cette nouvelle matrice d'état, avec le modèle $\frac{B_r(q^{-1})}{A'_r(q^{-1})}$ représentant la fonction de transfert pour le même système.

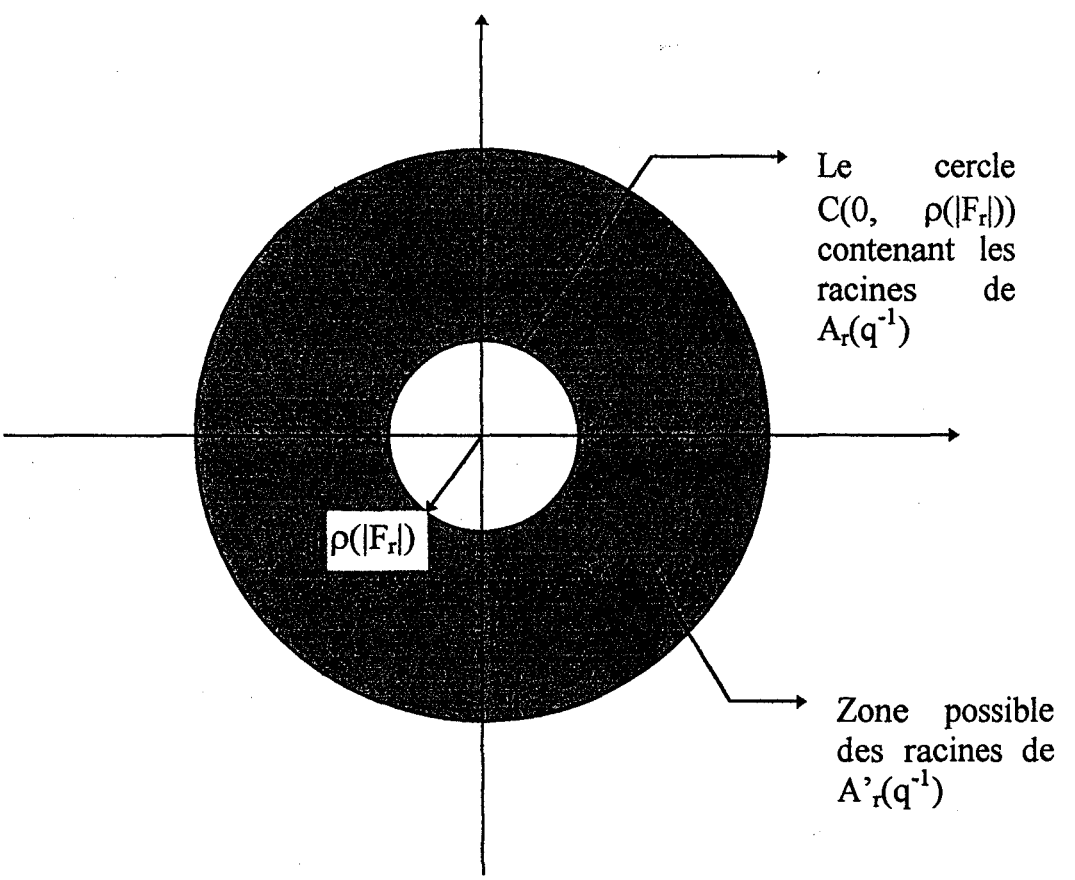


Fig III.1 Lieu des racines

D'après l'équation (III.4), au polynôme $A'_r(q^{-1})$ correspond un polynôme $A'(q^{-1})$ qui satisfait à l'équation caractéristique suivante :

$$\frac{B_r(q^{-1})}{A'_r(q^{-1})} = \frac{B(q^{-1}) * T(q^{-1})}{A'(q^{-1}) * S(q^{-1}) + B(q^{-1}) * R(q^{-1})} \qquad \text{(III.10)}$$

où, le polynôme $A'(q^{-1})$ caractérise le système perturbé comparativement au polynôme $A(q^{-1})$, lequel caractérise le système non perturbé. Ainsi, d'après les équations (III.4) et (III.10) nous obtenons :

$$A'(q^{-1}) = \frac{A'_r(q^{-1})}{A_r(q^{-1})} * (A(q^{-1}) + B(q^{-1}) * \frac{R(q^{-1})}{S(q^{-1})}) - B(q^{-1}) * \frac{R(q^{-1})}{S(q^{-1})} \qquad \text{(III.11)}$$

Donc, le système perturbé est maintenant décrit par $A'(q^{-1})=A(q^{-1})\pm\Delta A(q^{-1})$.

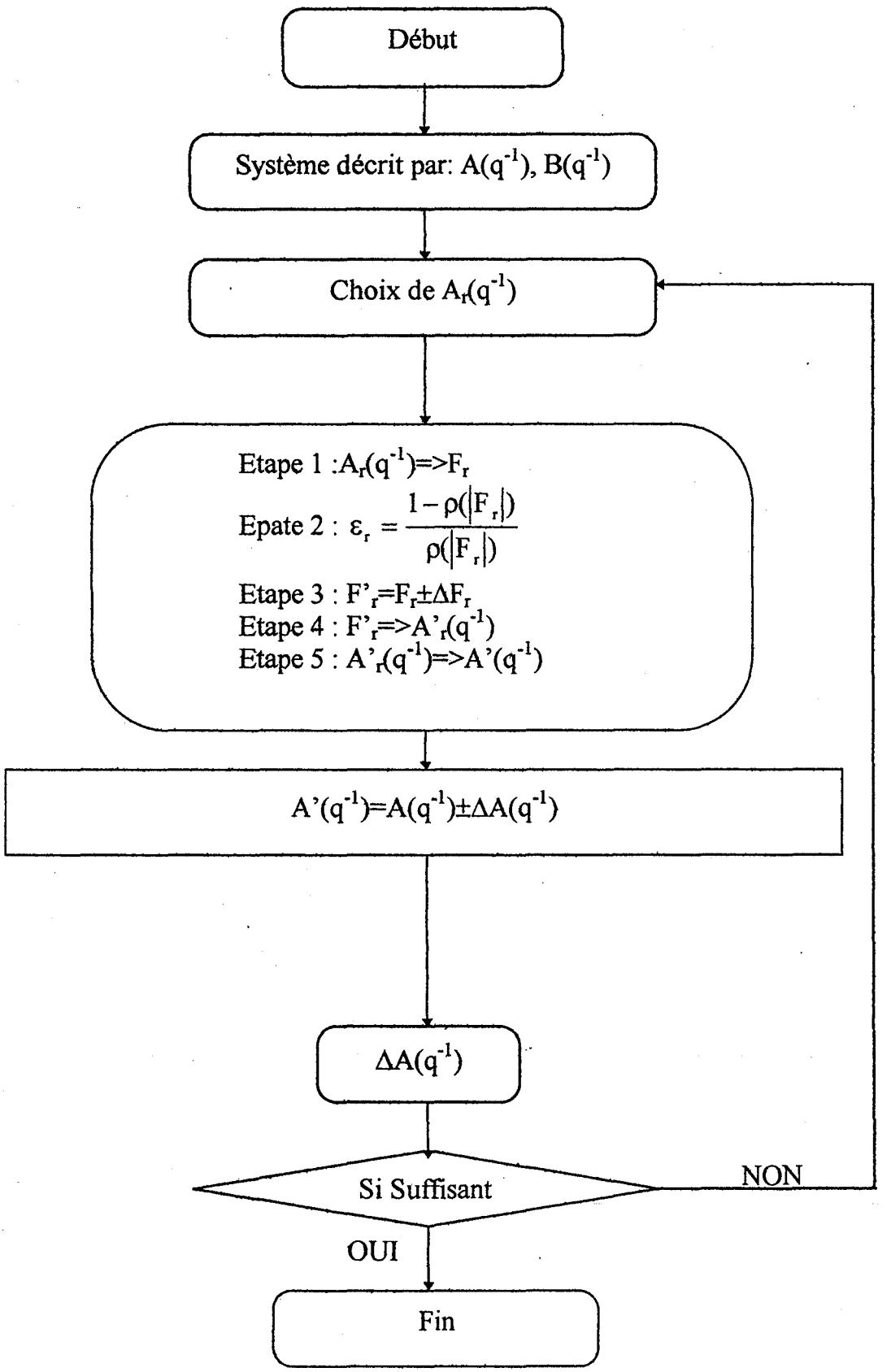
En résumé, la MPC/MRM dont le polynôme caractéristique définissant le comportement désiré est $A_r(q^{-1})$, appliquée au système modélisé par la fonction de transfert $B(q^{-1})/A(q^{-1})$, **assure** par l'intermédiaire de la définition précitée, **la robustesse de stabilité** et le bon fonctionnement **pour tous les systèmes perturbés** définis par la fonction de transfert $\mathbf{B(q^{-1})/(A(q^{-1})\pm\Delta A(q^{-1}))}$.

En plus, il nous est possible maintenant **d'évaluer cette robustesse par le terme $\Delta A(q^{-1})$** représentant la marge de stabilité du système de fonction de transfert $B(q^{-1})/A(q^{-1})$.

La Fig III. 1 montre que, plus les racines de $A_r(q^{-1})$ (valeurs propres de F_r) sont proches de l'origine, plus la marge de stabilité $\Delta A(q^{-1})$ est importante. Ainsi, **le choix optimal de $A_r(q^{-1})$ correspond à la marge maximale de stabilité $\Delta A(q^{-1})$.**

Il est important de préciser que $B_r(q^{-1})$, $B(q^{-1})$, $R(q^{-1})$, $S(q^{-1})$ et $T(q^{-1})$ restent inchangés et que trop rapprocher les racines de $A_r(q^{-1})$ de la valeur zéro dégrade les performances du système. Il faut donc trouver un compromis entre les performances et la robustesse du système.

III.3) ORGANIGRAMME DE LA COMMANDE PREDICTIVE AVEC EVALUATION DE LA ROBUSTESSE (Fig III. 2)



III.4) RESULTATS DE LA SIMULATION

La MPC/MRM a été utilisée en simulation, pour commander le processus thermique présenté dans le chapitre 2. La dynamique désirée est définie par :

$$A_r(q^{-1})=1+0.84\ q^{-1}+0.1864\ q^{-2} \qquad \text{(III.12)}$$

donc, les racines de $A_r(q^{-1})$ sont :

$$\lambda_1=-0.42+0.1*j \qquad \text{(III.13)}$$

$$\lambda_2=-0.42-0.1*j \qquad \text{(III.14)}$$

Ce qui entraîne une marge ΔA assez importante. Nous avons choisi en simulation comme hypothèse de départ, le retard d=0 et la consigne de température $y_{cc}(t)=1°C$.

Avec les périodes d'échantillonnage $T_e=10s$ et $t_e=1s$ et après résolution des équations diophantines, nous obtenons :

$$u(lt_e)=\frac{-RQ}{(1+a_1q^{-1}+a_2q^{-2})}y_{cc}(lt_e)-\frac{X+Yq^{-1}+Zq^{-2}+Wq^{-3}}{(1+a_1q^{-1}+a_2q^{-2})}y(lt_e) \qquad \text{(III.15)}$$

NB : Pour les coefficients, voir au chapitre 2 les équations de (II.35) à (II.41).

Notation :

On définit les polynômes $A(q^{-1})$, $B(q^{-1})$ et $A_r(q^{-1})$ par leurs coefficients suivant les puissances décroissantes :

$A=[0.6065\ -1.6065\ 1]$ désigne $A(q^{-1})=0.6065\ q^{-2}-1.6065\ q^{-1}+1$

$B=[0.3935\ 0]$ } idem pour $B(q^{-1})$ et $A_r(q^{-1})$

$A_r=[0.1864\ 0.84\ 1]$ }

L'analyse de stabilité donne ε limite, noté ε_r=1.31.

Cette valeur de ε_r signifie que nous pouvons accepter une variation des paramètres du système allant jusqu'à 131%. Cette variation peut être due à une mauvaise modélisation du procédé, à la non linéarité, à la présence d'un bruit extérieur etc..

Cependant, **la stabilité est assurée** pour toutes les perturbations **dans la limite où ($\varepsilon<\varepsilon_r$).**

Pour ε=0, le processus thermique décrit par $B(q^{-1})/A(q^{-1})$ peut être représenté par $B'(q^{-1})/A'(q^{-1})$ défini par :

A'=[0.0211 0.1341 0.1857 -0.4004 -1.0140 0.0735 1] , polynôme $A'(q^{-1})$
B'=[0.0137 0.1232 0.4243 0.6610 0.3935 0] , polynôme $B'(q^{-1})$

Le degré de $A'(q^{-1})$ est imposé par l'équation (III.11).

Remarquons que dans le cas où ε=0, nous avons bien $B'(q^{-1})/A'(q^{-1})=B(q^{-1})/A(q^{-1})$. Il s'agit simplement d'une notation différente.

Pour ε=0.8, nous avons A'=[0.0688 0.2967 0.0940 -0.0967 -1.1222 0.7455 1]
Pour ε=1.31, A'=[0.1141 0.4275 -0.0348 -1.6451 -1.0654 1.1791 1]
Pour ε=1.4, A'=[0.1225 0.4507 -0.0609 -1.7414 -1.0468 1.2495 1]

Le polynôme $\Delta A(q^{-1})$ peut être calculé par l'expression $|\Delta A(q^{-1})|=A'(q^{-1})-A(q^{-1})$.

Pour ε=**0.8**, nos avons ΔA=[**0.0477 0.1626** -0.0917 0.3037 **-0.1082 0.672** 0]
Pour ε=**1.31**, ΔA=[**0.093 0.2934** -0.2205 -1.2447 **-0.0514 1.1055** 0]
Pour ε=**1.4**, ΔA=[**0.1014 0.3166** -0.2466 -1.341 **-0.0328 1.176** 0]

On observe que plus la marge de sécurité augmente, plus les coefficients de haut degré de ΔA augmentent aussi. Ceci peut s'interpréter par le fait que plus on prend en compte l'effet de mémoire du système (termes de haut degré) plus on garantit une robustesse de stabilité.

III.5) COURBES DES SIMULATIONS ET COMMENTAIRES

La simulation a été réalisée sur une station SUN à l'aide du logiciel MATRIX. La sortie du système a été enregistrée pour chacune des valeurs de ε (Voir courbes). Les détails des simulations sont donnés par le tableau récapitulatif présenté à la page suivante.

Pour le système perturbé, lorsque $\varepsilon<\varepsilon_r$ (Fig III.5), non seulement le système est stable mais il demeure aussi robuste. C'était un résultat attendu car nous sommes encore dans la marge de robustesse donnée par l'équation de stabilité. L'utilisation du suréchantillonnage améliore considérablement le comportement du système.

Dans le cas où $\varepsilon=\varepsilon_r$ (Fig III.6), le système atteint sa limite de stabilité. Mais le suréchantillonnage permet au système d'être robuste vis à vis des variations paramétriques du modèle.

Enfin, dans le cas où $\varepsilon>\varepsilon_r$ (Fig III.7), le système ne peut plus suivre la consigne et diverge complètement malgré l'utilisation du suréchantillonnage, ce qui confirme la condition de stabilité.

Pour confirmer la fiabilité de cette méthode, nous avons introduit lors de la simulation du processus thermique une perturbation. Nous avons utilisé le modèle présenté dans le chapitre 2 (c'est-à-dire $H(q^{-1}) = \dfrac{0.3935q^{-1}}{1-1.6065q^{-1}+0.6065q^{-2}}$) pour déterminer la commande, puis nous avons appliqué cette commande au modèle du système auquel nous avons ajouté sin(u) (figures III.3 et III.4).

Méthode utilisée	Commande calculée à partir du modèle	Commande appliquée au modèle	Figure indiquant	Résultat de la simulation
MPC/MRM	$\frac{y(t)}{u(t)} = \frac{0.3935q^{-1}}{1-1.6065q^{-1}+0.6065q^{-2}} = \frac{B}{A} = \frac{B'}{A'}$ avec **A'** pour **ε=0** et **B'** (unique)	$(1-1.6065q^{-1}+0.6065q^{-2})*y(t) = (0.3935q^{-1})*u(t)+\sin(u(t))$	Fig III.3 **Commande**	La commande varie beaucoup jusqu'à ce que la sortie atteint la consigne, ensuite elle s'annule quasiment.
MPC/MRM	(même modèle) $y(t) = \frac{0.3935q^{-1}}{1-1.6065q^{-1}+0.6065q^{-2}}u(t)$	$(1-1.6065q^{-1}+0.6065q^{-2})*y(t) = (0.3935q^{-1})*u(t)+\sin(u(t))$	Fig III.4 **Sortie**	Confirmation du **bon choix de A_r,** qui offre une marge de robustesse de stabilité **ΔA suffisante,** capable de maîtriser **les perturbations causées par sin(u).**

L'analyse de stabilité du système montre que ε_r**=1.31**. Nous avons réalisé les simulations suivantes en fonction de la variation de ε.

Méthode utilisée	Commande calculée et appliquée à	Valeurs de A'	Résultat de la simulation
MPC/MRM	$\frac{y(t)}{u(t)} = \frac{B'}{A'}$ avec A' pour **ε=0.8**	A'=[0.0688 0.2967 0.0940 -0.0967 -1.1222 0.7455 1]	**Fig III.5** (Sortie). Le système **arrive à se stabiliser**. Le suréchantillonnage améliore les performances.
MPC/MRM	$\frac{y(t)}{u(t)} = \frac{B'}{A'}$ avec A' pour **ε=1.31**	A'=[0.1141 0.4275 -0.0348 -1.6451 -1.0654 1.1791 1]	**Fig III.6** (Sortie). Le système est à la **limite de sa stabilité.** Des fluctuations de grandes amplitudes **autour de la consigne.**
MPC/MRM	$\frac{y(t)}{u(t)} = \frac{B'}{A'}$ avec A' pour **ε=1.4**	A'=[0.1225 0.4507 -0.0609 -1.7414 -1.0468 1.2495 1]	**Fig III.7** (Sortie). Le système n'arrive plus à maîtriser les perturbations. **La marge de robustesse de stabilité est dépassée.**

Tableau récapitulatif des essais de simulation.

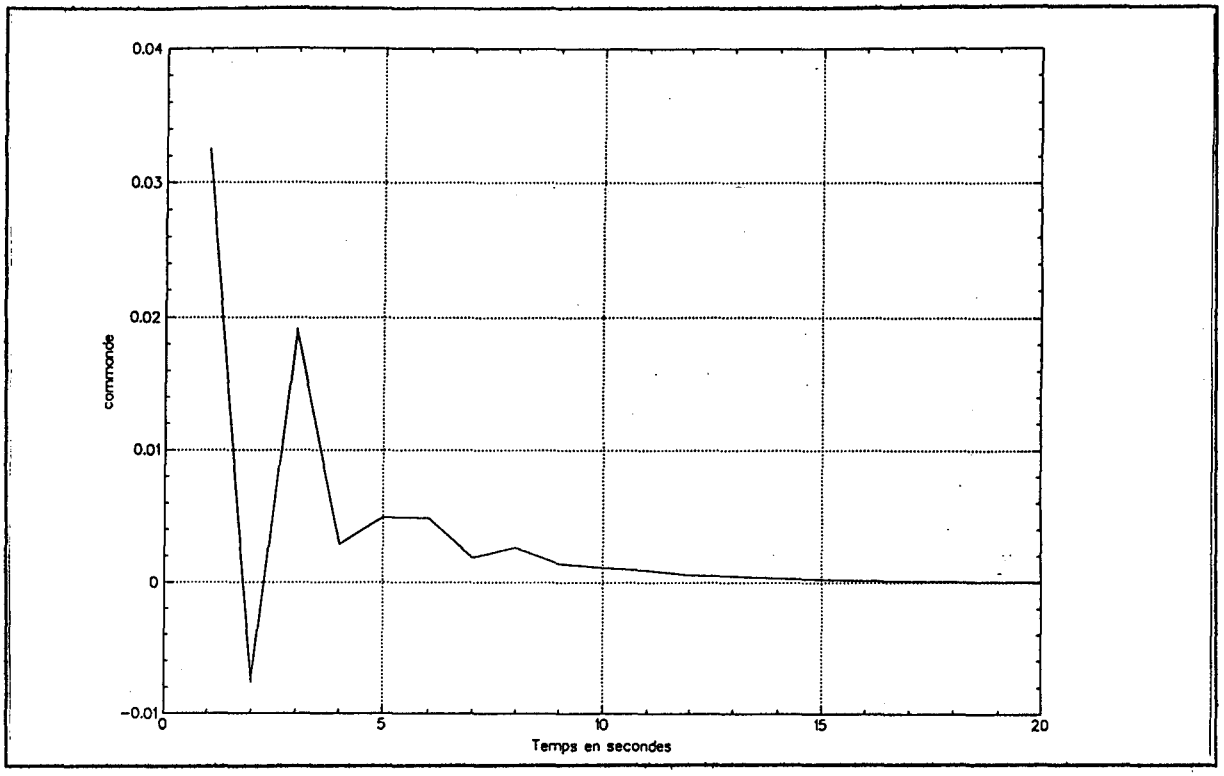


Fig III.3 Evolution de la commande (sans suréchantillonnage)

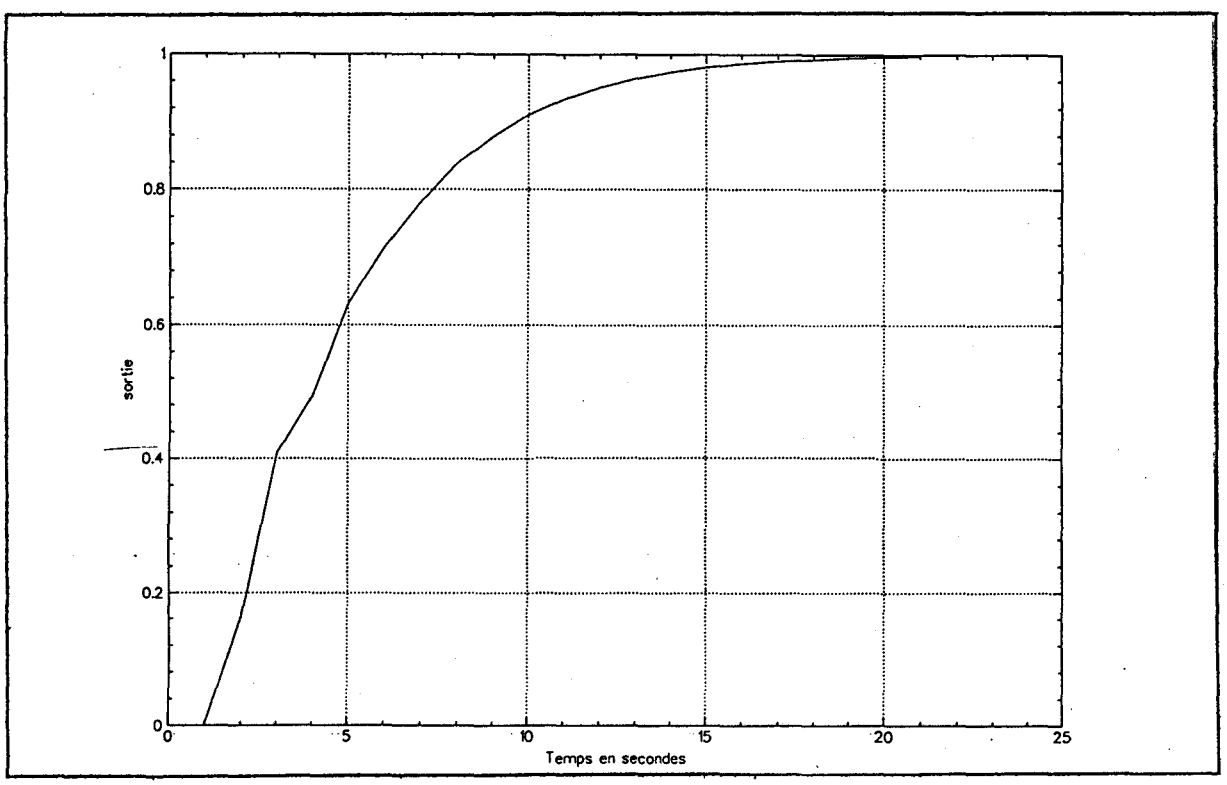


Fig III.4 Evolution de la sortie du système (sans suréchantillonnage)

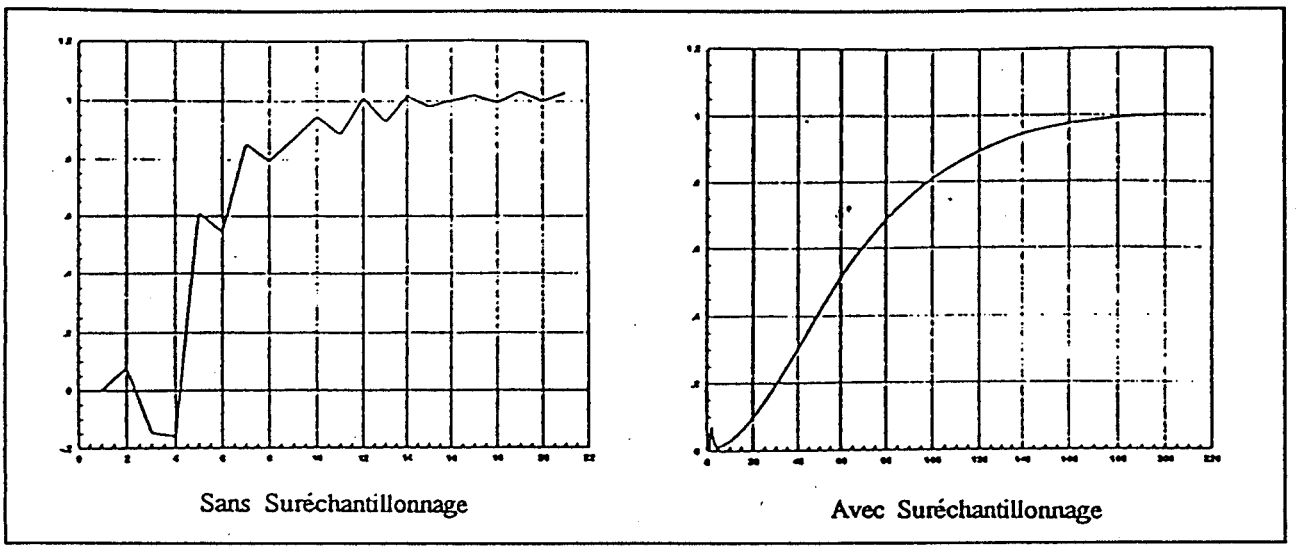


Fig III.5 Simulation ε=0.8

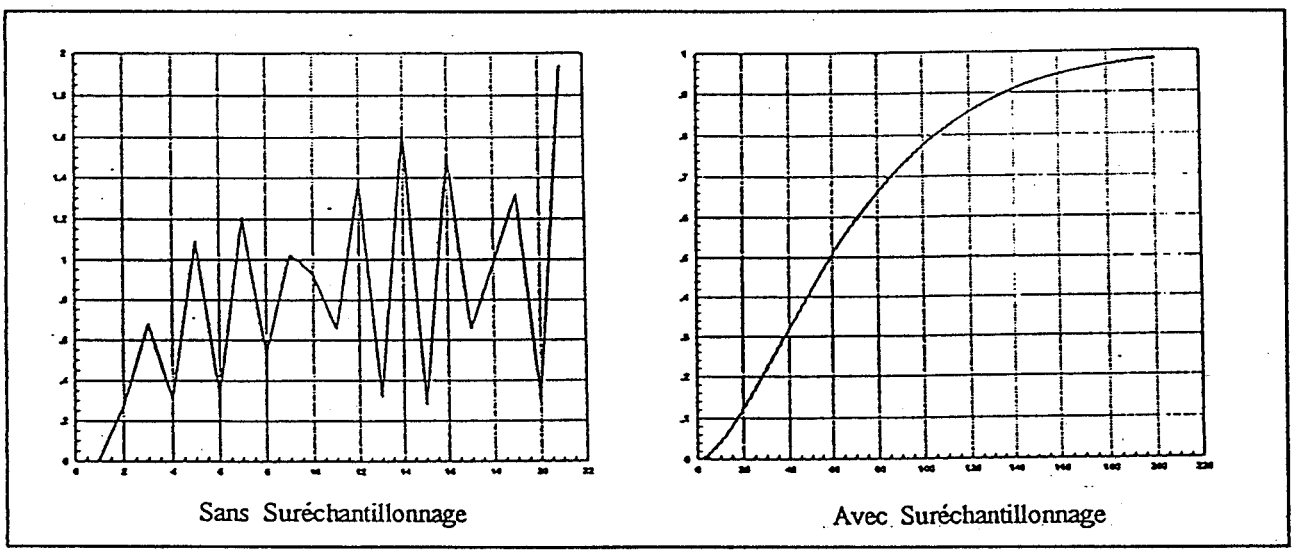


Fig III.6 Simulation ε=1.31

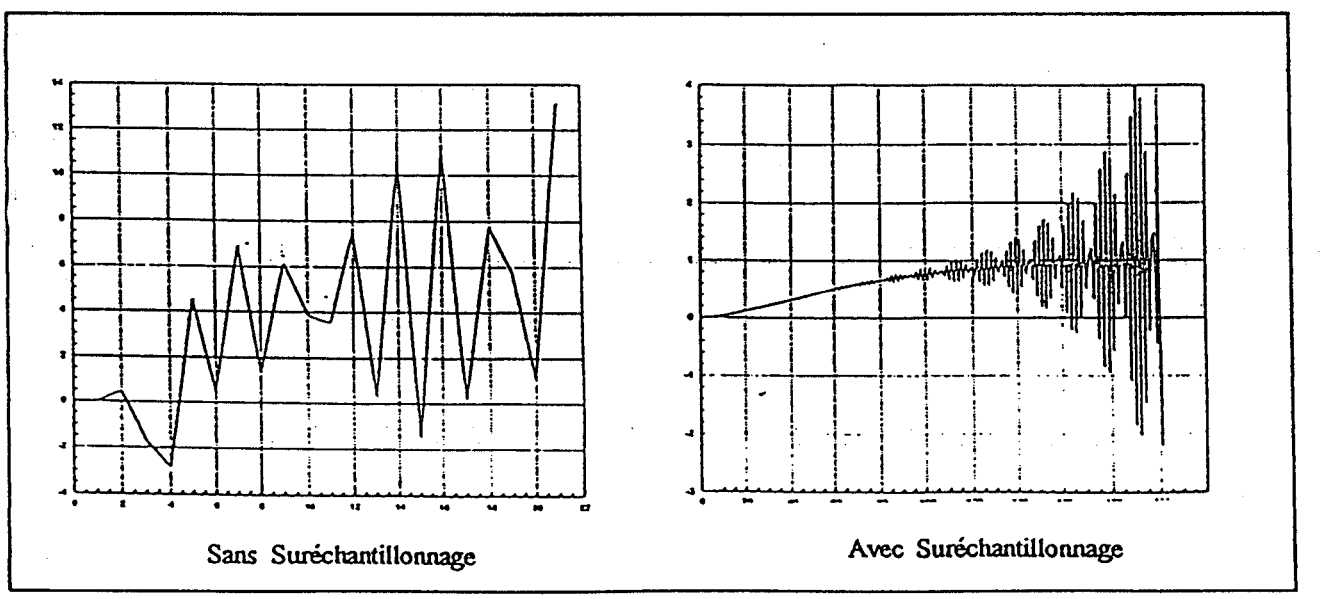


Fig III.7 Simulation ε=1.4

La courbe présentée dans la figure III.4 montre bien la capacité du système à prendre charge les perturbations occasionnées par l'ajout de sin(u). Le processus thermique a pu maîtriser ce bruit apporté grâce à la marge de stabilité engendrée par le choix des racines du polynômes A_r. Cette marge est limitée si nous rajoutons sin(10*u) le système devient instable. En effet, il y'a dépassement du seuil de stabilité soit par ε_r soit par la marge ΔA. Nous verrons dans le chapitre suivant comment la programmation non linéaire (PNL) permet d'améliorer la robustesse.

En pratique, lorsqu'on connaît l'expression de $A'(q^{-1})$, donc le polynôme représentant le système perturbé, nous pouvons définir le coefficient de proportionnalité ε qui a permis le passage de $A(q^{-1})$ à $A'(q^{-1})$. Nous pouvons donc déterminer le polynôme $A(q^{-1})$ représentant le système non perturbé. Par conséquent, il est possible de connaître (suivant la valeur de ε) si le système perturbé peut atteindre les performances souhaitées ou non. Dans ce cas, l'évaluation de la robustesse peut être déterminée assez facilement.

III.6) CONCLUSION

Ce chapitre a permis de présenter une méthode pour le choix optimal du comportement désiré. Ce choix est réalisé afin de palier les variations des paramètres du modèle du système causées par les perturbations (un système perturbé est équivalent à une variation des paramètres du système non bruité).

De même, une condition suffisante assurant la robustesse de stabilité est donnée. Cette condition prend en compte les racines du polynôme décrivant le système (ou aussi sur les valeurs propres de la matrice d'état représentant le système). Par conséquent, il nous est possible d'évaluer le robustesse du système, en calculant le coefficient de proportionnalité ε. Ce coefficient doit être comparé à ε_r donné par le rayon spectral de la matrice d'état définissant le comportement désiré du système à commander.

En pratique, à partir du modèle du système nous pouvons déduire la marge de stabilité maximale que peut offrir ce système. Cette marge ΔA est en réalité fonction du choix du comportement désiré que nous imposons au système. Nous pouvons donc évaluer le degré de robustesse de ce système par le biais de ΔA. Par ailleurs, lorsque nous connaissons la nature du bruit que va subir le système nous pouvons déterminer au travers du modèle du système bruité la valeur du coefficient de proportionnalité. Par une simple comparaison de ce dernier à ε_r (coefficient déduit de A_r), il est possible de prédire si le système sera stable ou non. S'il s'avère que le système ne peut pas résister à cette perturbation, nous pouvons remédier à cela en faisant un choix plus approprié du comportement désiré en choisissant des racines plus proches de l'origine. Cependant, il faut obtenir un compromis entre les performances et la stabilité.

Le chapitre suivant propose une solution par l'introduction des modèles non linéaires et par la réalisation d'une commande prédictive adapté à ces modèles.

REFERENCES

[Apkarian 1993]:
Apkarian P, Gahinet P.
« Synthèse H∞ des Systèmes Linéaires à Paramètres Variants», Ecole d'Eté d'Automatique de Grenoble, Robustesse et Synthèse de Commandes Robustes, 6-8 Septembre 1993, Chapitre 4.

[Aström 1983]:
Aström K. J, Wittermark B.
« One Self Tuning Regulators », Automatica 1983, Vol. 9, n°3, pp. 195-199.

[Bourlès 1993]:
Bourlès H, Aîoun F.
« Approche H∞ et μ-Synthèse », Ecole d'Eté d'Automatique de Grenoble, Robustesse et Synthèse de Commandes Robustes, 6-8 Septembre 1993, Chapitre 3.

[Bourret 1993]:
Bourret T, Lavigne G, Gauvrit M.
« Synthèse d'une Commande Robuste aux Incertitudes Paramétriques, Liée à la Qualité de l'Identification », Ecole d'Eté d'Automatique de Grenoble, Robustesse et Synthèse de Commandes Robustes, 6-8 Septembre 1993, Chapitre 8.

[Clarke 1985]:
Clarke D. W , Kanjilal P. P., Mohtadi C.
« A Generalized LQG Approach to Self Tuning Control », Int J. Control 1985, Vol 41, n°6, pp 1509-1544.

[Clarke 1987]:
Clarke D. W , Mohtadi C, Tuffs P.
« Generalized Predictive Control. Part I: The Basic Algorithm. Part II: Extension and Interpretation », Automatica 1987, Vol. 23, n°2, pp. 137-160.

[De Larminat 1993]:
De Larminat Ph.
« Méthodes Elémentaires pour la Commande Robuste », Ecole d'Eté d'Automatique de Grenoble, Robustesse et Synthèse de Commandes Robustes, 6-8 Septembre 1993, Chapitre 5.

[De Souza 1993]:
De Souza C. E.
« Robust State Estimation », Ecole d'Eté d'Automatique de Grenoble, Robustesse et Synthèse de Commandes Robustes, 6-8 Septembre 1993, Chapitre 7.

[Elliot 1984]:
Elliot H, Wolovich W. A. , Das M.
« Arbitrary Adaptive Pole Placement for Linear Multivariable Systems », IEEE Trans. Auto. Control 1984, AC-27, pp. 335-340.

[Garcia 1993]:
Garcia G, Bernussou M.
« Stabilité Quadratique et Paramétrisation Convexe », Ecole d'Eté d'Automatique de Grenoble, Robustesse et Synthèse de Commandes Robustes, 6-8 Septembre 1993, Chapitre 9.

[Kevicky 1981]:
Kevicky L, Kumar K.S.
« Multivariable Self Tuning Regulator with Generalized Cost Function », Int. J. Control 1981, Vol. 33, pp. 913-921.

[Koncar 1990]:
Koncar V, Vasseur C.
«Commande Numérique par Suréchantillonnage, Application à la réalisation d'une carte d'axe », ISMM, Lugano, Switzerland, June 1990.

[Koncar 1995]:
Koncar V, Koubaa M.A, Vasseur C.
«Multirate Predictive Control with Multiple Reference Model - Robustness Evaluation », Proceeding of th 12th International Conference Systems Engineering, ICSE'95, Vol. 1, pp. 444-454, Wroclaw, Poland 1995.

[Koncar 1995]:
Koncar V, Koubaa M.A, Bruniaux P, Vasseur C.
« Robust Setting of the Predictive Control Parameters», 2^{éme} Congrés Maghrébin de Génie Electrique, 16-17 Septembre 1995, Tunis, Tunisie.

[Kwakernaak 1993]:
Kwakernaak H.
« Robust Control - Mixed Sentivity Optimization », Ecole d'Eté d'Automatique de Grenoble, Robustesse et Synthèse de Commandes Robustes, 6-8 Septembre 1993, Chapitre 2.

[Landau 1981]:
Landau I. D, Lozano R.
« Unification of Discrete Time Explicit Model Reference Adaptive Control Designs », Automatica 1981, Vol. 17, pp. 593-611.

[Landau 1983]:
Landau I. D.
« Lectures notes on Adaptive Control : Theory and Applications », LAG Grenoble 1983.

[Landau 1993]:
Landau I. D, Rolland F, Cyrot C, Voda A.
« Régulation Numérique Robuste - Le Placement de Pôles avec Calibrage de la Fonction de Sensibilité », Ecole d'Eté d'Automatique de Grenoble, Robustesse et Synthèse de Commandes Robustes, 6-8 Septembre 1993, Chapitre 6.

[Magni 1993]:
Magni J. F.
« Robustesse et Performance par la Commande Modale », Ecole d'Eté d'Automatique de Grenoble, Robustesse et Synthèse de Commandes Robustes, 6-8 Septembre 1993, Chapitre10.

[Nguyen 1989]:
NGUYEN Minh Tri.
« Commande Adaptative Multivariable Avec Contraintes », Thèse de Doctorat, 19 Septembre 1989, Labo d'Automatique de Grenoble.

[Oustaloup 1993]:
Oustaloup A, Lanusse P, Mathieu B.
« Approche Fréquentielle et Non Entière de la Robustesse : la Commande CRONE », Ecole d'Eté d'Automatique de Grenoble, Robustesse et Synthèse de Commandes Robustes, 6-8 Septembre 1993, Chapitre 1.

[Rachid 1989]:
Rachid A.
« Robustness of Discrete Systems under Structured Uncertainties », Int. J. Control. 1989, Vol. 50, N°. 4, pp. 1563-1566.

[Rachid 1990]:
Rachid A.
« Robustness of Pole Assignment in a Specified Region for Perturbed Systems », Int. J. System. Sci. 1990, Vol. 21, N°. 3, pp. 579-585.

[Samon 1982]:
Samon C. 1982.
« An Adaptive LQ Controller for Non Minimum Phase Systems », Int. J. Control 1982, Vol 35. pp. 1-28.

[Soh 1985]:
Soh Y. C, Berger C. S, Dabke K. P.
« On The Stability Properties of Polynomials with Perturbed Coefficients », I.E.E.E Transaction on Automatic Control, Vol. 30, 1985, pp. 1033-1036.

[Trofino 1993]:
Trofino A.
« Robustesse et Performance des Systèmes Incertains », Ecole d'Eté d'Automatique de Grenoble, Robustesse et Synthèse de Commandes Robustes, 6-8 Septembre 1993, Chapitre 11.

CHAPITRE 4

COMMANDE PREDICTIVE PAR LA PROGRAMMATION NON LINEAIRE

Chapitre 4

COMMANDE PREDICTIVE PAR LA PROGRAMMATION NON LINEAIRE

IV.1) INTRODUCTION

Dans le domaine de la teinture, assurer une qualité optimale revient à garantir un bon unisson et de bonnes solidités. Or, la conduite de température se révèle être l'un des éléments les plus importants garantissant de telles exigences. Aussi, pour réaliser une commande de température de façon optimale, il faut faire appel à de nouvelles techniques, car les procédés industriels à commander sont le plus souvent complexes, non stationnaires, difficiles à modéliser et très sensibles aux perturbations extérieures, tout en recherchant sécurité, précision et fiabilité.

La plupart des procédés industriels sont, par nature, non linéaires et variant dans le temps. Ces variations d'origine paramétriques sont dues au changement du point de fonctionnement, à l'usure, au bruit extérieur, etc...

Cependant, si la commande prédictive semble un excellent choix pour la commande d'un processus thermique compte tenu de sa capacité à assimiler les phénomènes de retard, la non linéarité du modèle ne permet pas de l'appliquer

directement. Aussi, dans ce chapitre, nous introduisons la **Programmation Non Linéaire basée sur la théorie du Lagrangien augmenté en vue de faire l'interface entre modèle non linéaire et commande prédictive**.

Dans un premier temps, nous présentons quelques méthodes d'optimisation telles que, la programmation mathématique avec ou sans contraintes, la programmation dynamique et la programmation non linéaire.

Dans un second temps, nous exposons le principe de la méthode du Lagrangien augmenté et son application aux méthodes d'optimisation.

Ensuite nous présentons la commande prédictive, utilisant la programmation non linéaire en vue de la minimisation du critère quadratique.

Enfin nous terminons ce chapitre, par la présentation des résultats de la simulation de contrôle du processus thermique, représenté par son modèle **non linéaire** et la comparaison de cette méthode à la MPC/MRM « classique ».

IV.2) PRESENTATION DES METHODES D'OPTIMISATION

L'étude théorique des problèmes d'optimisation ainsi que la conception et la mise en oeuvre des algorithmes de résolution, nécessitent des outils mathématiques assez puissants. Les premières recherches et applications à ce sujet se sont développées dans le contexte de l'économie et de la recherche opérationnelle.

Le terme « algorithme de résolution » fait tout de suite appel au mot programmation. C'est ainsi que G. B. Dantzig propose en 1949 le terme de programmation linéaire pour l'étude des problèmes théoriques et algorithmes liés à l'optimisation de fonctions linéaires sous contraintes linéaires [DANTZIG 1949].

Dans le même sens, Kühn et Tücker proposent en 1951 le nom de programmation non linéaire pour l'étude des problèmes d'optimisation non linéaires avec ou sans contraintes [KUHN 1951]. La programmation dynamique est suggérée par R. Bellman pour une méthode générale d'optimisation des systèmes dynamiques c'est-à-dire évoluant au cours du temps [BELLMAN 1957].

Toutes ces méthodes, font partie d'une discipline plus vaste, dite programmation mathématique [MINOUX 1983]. Cette dernière, est aujourd'hui une branche particulièrement active des mathématiques appliquées, et il y a, à cela, de nombreuses raisons. La première est peut être le nombre, la variété et l'importance de ses applications que ce soit dans les sciences de l'ingénieur, ou dans d'autres domaines des mathématiques appliquées. Parmi ces domaines, nous pouvons citer :

* en analyse numérique : approximation, régression, résolution des systèmes linéaires et non linéaires, méthodes numériques liées à la mise en oeuvre des méthodes d'éléments finis, etc...

* en automatique : identification des systèmes, commande optimale des systèmes, filtrage, ordonnancement d'atelier, commande de robots, etc...

* en ingénierie : optimisation de structures, conception optimale de systèmes techniques complexes tels que systèmes informatiques, réseaux d'ordinateurs, réseaux de transport, de télécommunication, etc...

IV.2.1) PROGRAMMATION MATHEMATIQUE

D'une façon très générale, un programme mathématique est un problème d'optimisation sous contraintes dans l'ensemble des réels $\Re^n$ de la forme :

$$(P) \quad \begin{cases} \text{Minimiser } f(x) \\ \text{sous les contraintes :} \\ g_i(x) \leq 0 \quad (i = 1, 2, ..., m) \\ x \in S \subset \Re^n. \end{cases}$$

Le vecteur $x \in \Re^n$ a pour composantes $x_1, x_2, ..., x_n$ représentant les inconnues du problème.

La fonction f est appelée fonction objectif (ou aussi fonction économique) et l'ensemble des conditions : $g_i(x) \leq 0$ $(i = 1, 2, .., m)$ et $x \in S$ sont les contraintes du problème.

La distinction, dans (P), des deux types de contraintes se justifie par le tableau 1 présenté ci-dessous. Nous trouvons dans ce tableau une classification des différents types de problèmes d'optimisation que l'on rencontre en pratique, suivant les propriétés de la fonction f, celles des fonctions g_i et suivant la définition du sous-ensemble S de $\Re^n$ [GOMORY 1958].

Fonction f	Fonction g_i	Ensemble S	Terminologie employée
Continues, non linéaires quelconques		Continu, compact $\subset \Re^n$	Programmation mathématique continue
Non linéaires quelconques (pas nécessairement continues)		Discret (exemple : ensemble des points à coordonnées entières contenu dans un compact)	Programmation mathématique discrète (si $S \subset Z^n$, programmation non linéaire en nombres entiers)
Continue, non linéaire quelconque	m=0	$S=\Re^n$	Optimisation non linéaire continue sans contraintes
Non linéaire quelconque (pas nécessairement continue)	m=0	$S=Z^n$	Optimisation non linéaire en nombres entiers sans contraintes

TABLEAU 1 : Les principales classes en programmation mathématique

Remarque : nous nous sommes limités dans ce tableau, aux fonctions non linéaires uniquement, car c'est le cas qui nous intéresse principalement.

Le tableau ci-dessus illustre la large capacité de la programmation mathématique, à résoudre les problèmes d'optimisation (maximiser une fonction f se ramène à minimiser la fonction h=-f).

IV.2.2) PROGRAMMATION NON LINEAIRE

Parmi les ouvrages fondamentaux dans le domaine de la programmation non linéaire, on trouve le livre de Zangwill (Nonlinear programming : a unified approach, 1969) où apparaît pour la première fois une théorie générale de la convergence, ainsi que le livre de Luenberger (Introduction to linear and nonlinear programming, 1973) qui propose une analyse comparative systématique des algorithmes à partir de la notion de vitesse de convergence [LUENBERGER 1973], [ZANGWILL 1969].

IV.2.3) OPTIMISATION NON LINEAIRE AVEC CONTRAINTE

On s'intéresse ici au problème suivant :

$$(P') \quad \begin{cases} \text{Minimiser } f(x) \\ \text{sous les contraintes :} \\ g_i(x) \leq 0 \quad i \in I = \{1, 2, ..., m\} \\ x \in \Re^n. \end{cases}$$

qui n'est autre que le problème (P) du paragraphe IV.2.1 avec la condition $S=\Re^n$.

Toutes les fonctions f et g_i ($i \in I$) sont supposées continues et différentiables.

La plupart des méthodes existantes en programmation non linéaire sous contraintes peuvent se rattacher à deux grandes familles :

* méthodes directes (ou : primales)
* méthodes utilisant la notion de dualité

Les méthodes primales se caractérisent par le fait qu'elles opèrent directement sur le problème donné (appelé problème primal par opposition au problème dual). Elles engendrent une séquence de solutions (c'est-à-dire de points satisfaisant les contraintes) en assurant une décroissance monotone de la fonction à minimiser. Elles présentent donc un avantage important : si le processus itératif est interrompu, elles procurent une solution approchée satisfaisant les contraintes. Par contre, elles ont généralement l'inconvénient d'être de mise au point délicate, et la propriété de convergence globale est souvent difficile à obtenir. Par opposition, les méthodes duales sont plus robustes et la convergence globale est souvent plus facile à obtenir. En contrepartie, elles présentent l'inconvénient de ne fournir qu'une solution primale réalisable en fin de convergence.

Pour résoudre un problème primal, quelques techniques sont souvent utilisées telles que : les méthodes de directions réalisables, la méthode du gradient projeté, la méthode du gradient réduit, le gradient réduit généralisé et les méthodes de linéarisation.

Par contre pour résoudre un problème dual, nous utilisons les méthodes dites de pénalité ou les méthodes basées sur la notion de dualité. Leur principe commun consiste à ramener le problème initial à la résolution d'une suite de problèmes d'optimisation sans contraintes.

IV.2.4) METHODE DU LAGRANGIEN AUGMENTE

Pour résoudre un problème du type (P), présenté ci-dessus, nous avons vu jusqu'ici deux types de méthodes permettant de le remplacer par la résolution d'une séquence de problème d'optimisation sans contraintes.

* La méthode de pénalité [ABLOW 1955], [COURANT 1943], [FIACCO 1968].

* La méthode utilisant la dualité lagrangienne.

Associons à chaque contrainte un nombre réel $\lambda_i \geq 0$ appelé multiplicateur de Lagrange. La fonction de Lagrange associée au problème (P), est par définition la fonction :

$$L(x,\lambda) = f(x) + \sum_{i \in I} \lambda_i g_i(x)$$

Ainsi, le problème (P) est remplacé par le problème : minimiser $L(x,\lambda)$ sans contraintes.

$$\min_x \left\{ f(x) + \sum_{i=1}^{m} \lambda_i g_i(x) \right\}$$

Ces méthodes ne sont valables que lorsque les contraintes sont des inégalités. Dans le cas où les contraintes sont des égalités, il suffit de faire une combinaison des deux approches (pénalité+dualité) [HESTENES 1969] et [POWEL 1969]. Nous sommes donc amené à résoudre une séquence de problèmes sans contraintes de la forme :

$$\min_x L(x,\lambda,r) = \min_x \left\{ f(x) + \sum_{i=1}^{m} \lambda_i g_i(x) + r \sum_{i=1}^{m} \left[g_i(x) \right]^2 \right\}$$

λ_i de signe quelconque et r>0.

Mais comment faire lorsque les contraintes sont à la fois sous formes d'égalités et d'inégalités?. En effet, il suffit d'introduire des variables d'écart $s_i \geq 0$ [ROCKAFELLAR 1973]. Le problème (P') peut s'écrire sous la forme :

$$\begin{cases} \text{Minimiser } f(x) \\ \text{sous les contraintes :} \\ g_i(x) + s_i = 0 \\ s_i \geq 0 \\ x \in \Re^n. \end{cases}$$

Cet avantage est l'une des raisons qui nous ont poussés à utiliser la méthode du Lagrangien augmenté.

Nous avons alors à résoudre à chaque période d'itération le problème :

$$\underset{\substack{x\in\Re^n \\ s\geq 0}}{\text{Min}} \left\{ f(x) + \sum_{i=1}^{m} \lambda_i (g_i(x) + s_i) + r \sum_{i=1}^{m} \left[g_i(x) + s_i \right]^2 \right\}$$

Nous remarquons que dans cette expression, la minimisation suivant $s_i \geq 0$, pour x fixé, peut être faite explicitement pour chaque s_i séparément. En effet, les termes faisant apparaître s_i sont :

$$r.s_i^2 + (2.r.g_i(x) + \lambda_i).s_i$$

et, si l'on suppose r>0, le minimum sur $s_i \geq 0$ est atteint :

* soit pour $s_i = -g_i(x) - \lambda_i/2r$ (si $g_i(x) \leq -\lambda_i/2r$)

* soit pour $s_i = 0$ (si $g_i(x) \geq -\lambda_i/2r$)

Le problème revient alors à minimiser suivant x à chaque étape la fonction :

$$L(x,\lambda,r) = f(x) + \sum_{i=1}^{m} G(\lambda_i, g_i(x), r)$$

où,

$$G(g_i(x),\lambda_i r) = \begin{cases} -\dfrac{\lambda_i^2}{4r} & \text{si } r \succ 0 \text{ et } g_i(x) \le -\dfrac{\lambda_i}{2r} \\ \lambda_i g_i(x) + r[g_i(x)]^2 & \text{si } r \succ 0 \text{ et } g_i(x) \ge -\dfrac{\lambda_i}{2r} \\ \lambda_i g_i(x) & \text{si } r = 0 \text{ et } \lambda_i \ge 0 \\ -\infty & \text{si } r = 0 \text{ et } \lambda_i \prec 0 \end{cases}$$

appelée « Lagrangien augmenté ».

D'autres types de Lagrangiens augmentés, basés sur la même idée (combinaison du Lagrangien classique avec des fonctions de pénalités) ont pu être proposés [NAKAYAMA 1975].

L'utilisation d'un Lagrangien augmenté peut être considérée comme une amélioration des méthodes de pénalités qui évite d'avoir à utiliser des coefficients de pénalité trop grands. En plus, il permet d'étendre la théorie de la dualité à des problèmes non convexes.

Il est aujourd'hui largement reconnu, que les algorithmes d'optimisation basés sur l'utilisation des Lagrangiens augmentés, font partie des méthodes générales les plus

efficaces, pour résoudre les problèmes de programmation mathématique avec fonction objectif et contraintes fortement non linéaires.

IV.3) COMMANDE PREDICTIVE PAR LA PROGRAMMATION NON LINEAIRE

Pour appliquer la commande prédictive aux systèmes non linéaires, nous avons choisi une méthode d'optimisation du critère quadratique basée sur le principe du Lagrangien augmenté, vue son efficacité à optimiser des fonctions non linéaires.

De même, cette méthode présente peu de restriction vis à vis : de la convexité de la fonction objectif, de la formulation des contraintes (égalité et inégalité) et de la différentiabilité du critère à minimiser.

Considérons que le système à commander est régi par l'expression suivante :

$$y(lt_e)=f(Y, U) \qquad (IV.1)$$

avec,

$Y^T(lt_e)=[y(l-1)t_e, ..., y(l-n)t_e]$
$U^T(lt_e)=[u(l-1)t_e, ..., u(l-m-d)t_e]$

où, f est une fonction non linéaire, à laquelle correspond la « transmittance non linéaire $F(q^{-1})$ ».

Nous proposons d'utiliser la commande prédictive du type échantillonné à modèles de références multiples. Nous définissons le comportement désiré du système global linéaire :

$$\frac{y(lt_e)}{y_{cc}(lt_e)} = \frac{B_r(q^{-1})}{A_r(q^{-1})} \qquad (IV.2)$$

avec $A_r(q^{-1})$ imposé par l'utilisateur. Nous avons fixé le comportement désiré par le choix des polynômes $A_r(q^{-1})$ et $B_r(q^{-1})$, qui sont linéaires. Une idée à réaliser à l'avenir, est de fixer un comportement désiré non linéaire du type une fonction $F_r(q^{-1})$.

Utilisons le régulateur « classique » RST, nous pouvons écrire :

$$u(lt_e) = \frac{T(q^{-1})}{S(q^{-1})} y_{cc}(lt_e) - \frac{R(q^{-1})}{S(q^{-1})} y(lt_e) \qquad (IV.3)$$

qui correspond au schéma de commande suivant (Fig IV.1) :

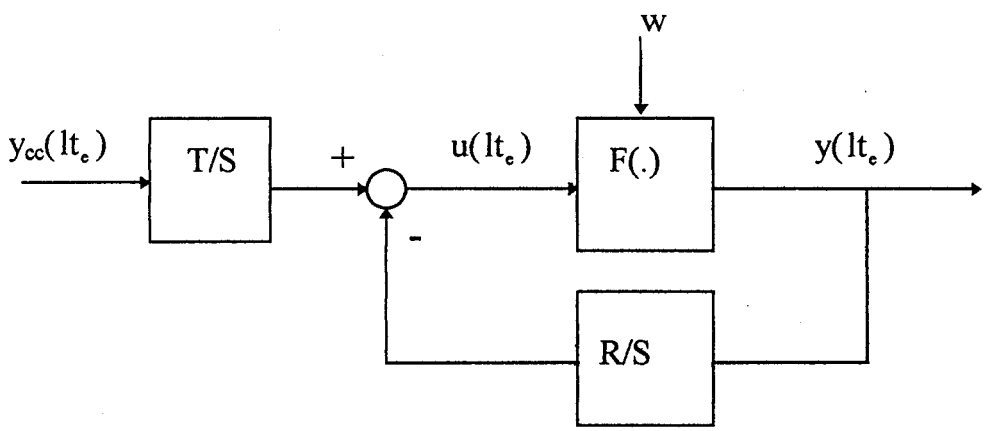


Fig IV.1 Schéma de commande

Nous allons introduire un critère quadratique inspiré du critère d'Irving à partir des erreurs suivantes :

$$\xi_{sy}(lt_e) = y(lt_e) - f(Y, U) \qquad (IV.4)$$

$$\xi_{su}(lt_e) = \frac{B_r(q^{-1})}{A_r(q^{-1})} y_{cc}(lt_e) - f(Y, U) \qquad (IV.5)$$

ξ_{sy} représente l'erreur entre la sortie effectivement mesurée (IV.1) et la sortie calculée. Cette erreur ne préjuge pas de la structure de contrôle par contre la minimisation de cette erreur par rapport à u permet de réduire l'effet des perturbations.

ξ_{Su} caractérise la différence entre la sortie désirée (c'est-à-dire la dynamique imposée) et la sortie calculée. Cette erreur vise à déterminer la commande u permettant de se rapprocher au mieux du comportement désirée défini par la structure bouclée.

Remarque : on peut facilement vérifier que si $F(q^{-1})$ est linéaire de la forme $\frac{B(q^{-1})}{A(q^{-1})}$ les expressions de ξ_{sy} et ξ_{su} ci-dessus sont équivalentes aux expressions ε_{sy} et ε_{su} définis au chapitre 1 (I.19) et (I.20).

Ainsi, le critère adopté est un critère quadratique prenant en compte ξ_{sy} et ξ_{su} :

$$J(lt_e, N_y, N_u) = \sum_{i=0}^{N_y-1} \varepsilon_{sy}^2((lte + (i+d)T_e) + \lambda \sum_{i=0}^{N_u-1} \varepsilon_{su}^2(lte + iT_e) \qquad (IV.6)$$

La commande optimale $u(lt_e)$ est alors obtenue par minimisation du critère non linéaire J par la programmation non linéaire (PNL).

Sous la contrainte :

$$u_{min} \leq u \leq u_{max}$$

C'est la programmation non linéaire qui se charge de résoudre ce problème en déterminant la commande $u(lt_e)$ optimale. Cette méthode est réalisée grâce à la fonction *optimize* du logiciel de simulation XMATH 4.1 décrite en annexe.

IV.4) SIMULATION DU PROCESSUS THERMIQUE

La commande utilisée pour la simulation du processus thermique, est du type MPC/MRM utilisant la PNL. Nous la notons MPC/PNL.

IV.4.1) MODELE LINEAIRE A RETARD DU PROCESSUS THERMIQUE

En réalité le modèle du processus thermique est représenté par l'équation suivante :

$$[\tau+(T_e-2\tau)q^{-1}-(T_e-\tau)q^{-2}]*y(t) = T_e q^{-1}\, u(t-a) \qquad (IV.7)$$

Pour le calcul, il convient donc de synchroniser le retard avec la période de suréchantillonnage. En effet, s'il n'y a pas de suréchantillonnage, nous ne pouvons pas détecter de retard $a<T_e$.

On choisit :

$$T_e = 10 \text{ s}, \qquad (IV.8)$$

$$t_e = 1 \text{ s}, \qquad (IV.9)$$

$$\tau = 20 \text{ s}, \qquad (IV.10)$$

$$a=6 \text{ s} \qquad (IV.11)$$

$$\lambda=50 \text{ (facteur de pondération du critère)} \qquad (IV.12)$$

La dynamique désirée est définie par:

$$A_r(q^{-1})=1+0.8\, q^{-1}+0.1864\, q^{-2} \qquad (IV.13)$$

Les horizons de prédiction de la commande et de la sortie, sont $N_u=2$ et $N_y=3$.

Le résultat de la simulation est donné par la figure IV.3 suivante :

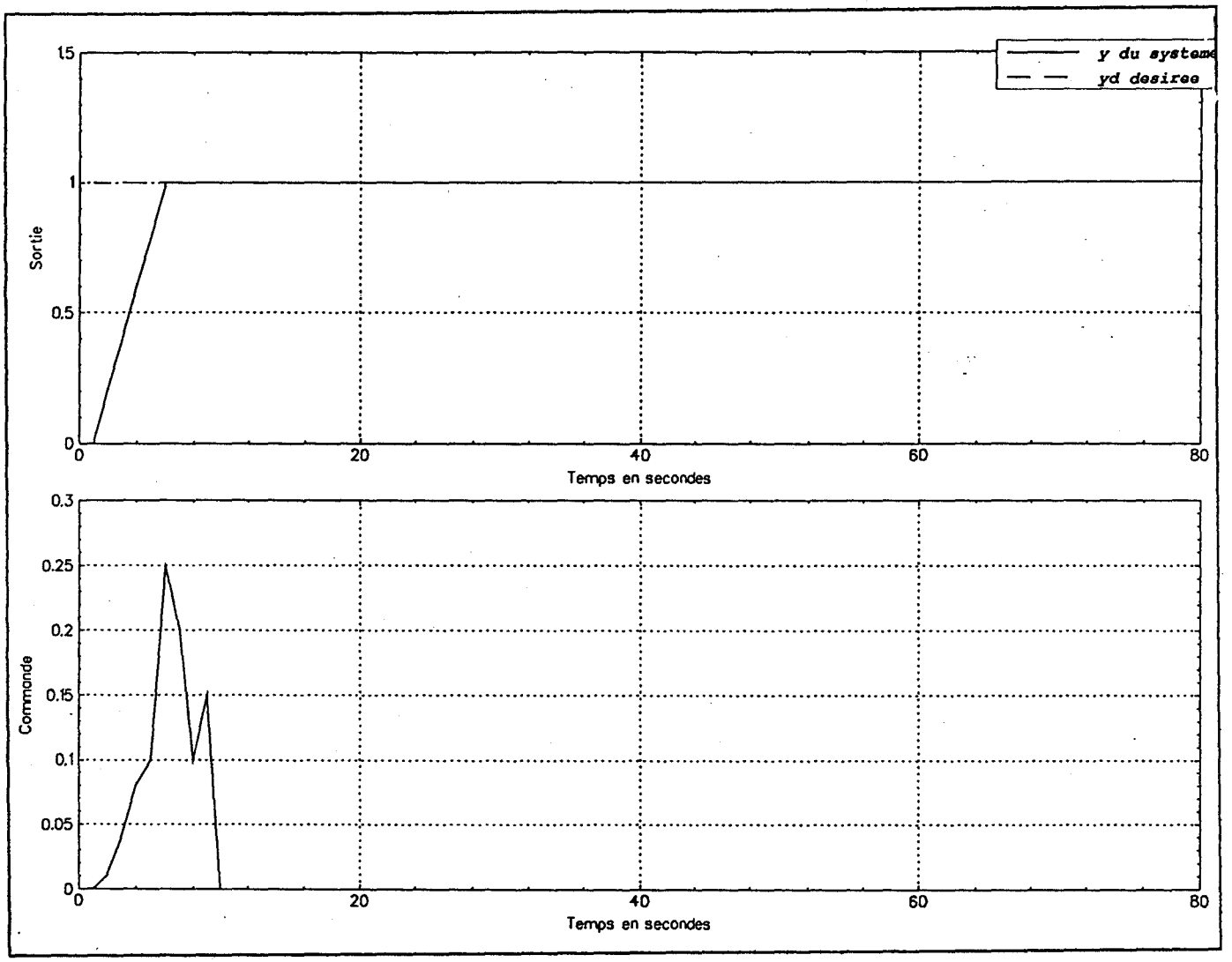


Fig IV.3 La MPC/PNL appliquée au Processus Thermique

Nous remarquons que le retard n'a pas posé de problème pour la MPC/PNL à commander le processus. Cependant, la commande prédictive atténue le retard pour réguler le procédé (Fig IV.3). La sortie du système est totalement confondue avec la consigne souhaitée, ce qui montre que la méthode utilisée est bien adaptée au modèle du système.

Afin de tester la robustesse de stabilité de cette méthode, nous avons ajouté une perturbation à la sortie du système. La figure IV.4 montre le résultat de cette simulation.

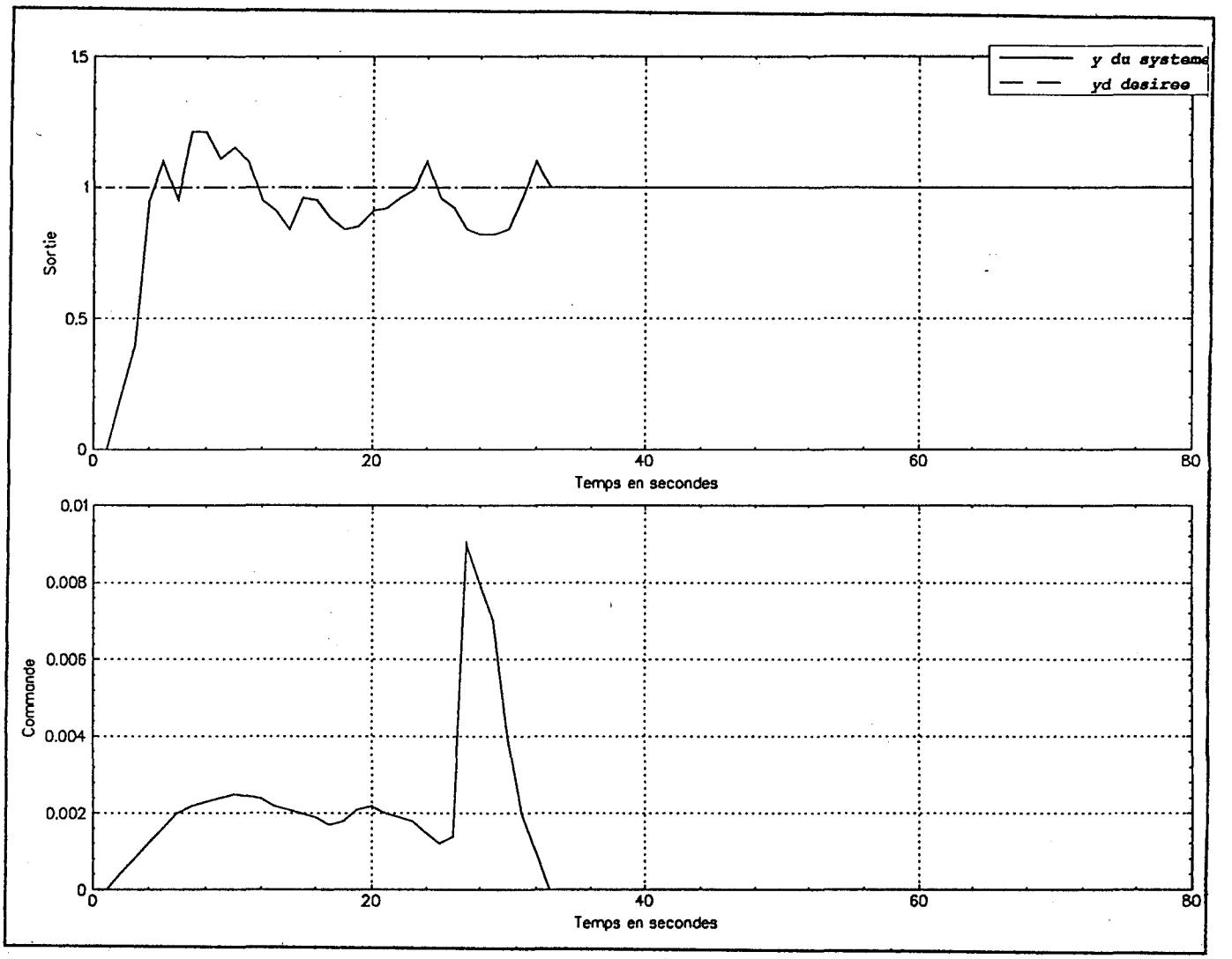


Fig IV.4 MPC/PNL appliquée au processus thermique bruité

La perturbation est définie comme une fonction aléatoire (random*(u(t))) multipliée par la commande (exemple : nous pouvons avoir une perturbation du type 0.201*u, 100*u, 123.456*u etc...). Pour réaliser ce test, la commande est calculée par la MPC/PNL en se basant sur le modèle du processus thermique décrit par l'équation (IV.12). Ensuite, cette commande est envoyée au système composé du modèle du

processus thermique auquel nous ajoutons la fonction de perturbation. Nous appelons ce dernier système, le processus thermique bruité (voir tableau à la page suivante).

Cette courbe (Fig IV.4) démontre la capacité de la commande prédictive à commander un tel processus malgré l'adjonction de bruit en sortie. La robustesse obtenue confirme entre autre le bon choix de la dynamique désirée imposée au système, pour garantir une bonne robustesse de stabilité.

IV.4.2) MODELE NON LINEAIRE

Etant donnée que la commande prédictive tient compte des effets du retard, nous avons introduit une non linéarité dans la fonction du modèle, pour mieux tester cette méthode de commande. Ainsi, l'expression du modèle est donnée par l'équation suivante pour $t=lt_e$:

$$[\tau+(T_e-2\tau)q^{-1}-(T_e-\tau)q^{-2}]*y(t) = T_e q^{-1}\, u(t) + \sin(10*u(t)) \qquad \text{(IV.14)}$$

Le tableau suivant montre les détails des simulations réalisées.

Méthode utilisée	Commande calculée à partir du modèle	Commande appliquée au modèle	Figure correspondante indiquant	But de la simulation
MPC/PNL	$A(q^{-1})y(t) = B(q^{-1})u(t-a)$	$A(q^{-1})y(t) = B(q^{-1})u(t-a)$	Fig IV.3 Commande et Sortie	La commande prédictive peut commander un système Linéaire à retard
MPC/PNL	$A(q^{-1})y(t) = B(q^{-1})u(t-a)$	$A(q^{-1})y(t) = B(q^{-1})u(t-a) + f_{aléa}(u(t))$	Fig IV.4 Commande et Sortie	Montrer la **robustesse de stabilité de la MPC/PNL**
MPC/PNL	$A(q^{-1})y(t) = B(q^{-1})u(t) + \sin(10*u(t))$	$A(q^{-1})y(t) = B(q^{-1})u(t) + \sin(10*u(t))$	Fig IV.5 Commande et Sortie	**La commande prédictive peut contrôler un système Non Linéaire**
MPC/PNL	$A(q^{-1})y(t) = B(q^{-1})u(t)$	$A(q^{-1})y(t) = B(q^{-1})u(t)$	Fig IV.6 Commande et Sortie	La MPC/PNL peut s'appliquer aussi à un modèle linéaire
MPC/MRM Classique	$A(q^{-1})y(t) = B(q^{-1})u(t)$	$A(q^{-1})y(t) = B(q^{-1})u(t)$	Fig IV.7 Sortie	Comparaison de la MPC classique à la MPC/PNL
MPC/MRM Classique	$A(q^{-1})y(t) = B(q^{-1})u(t)$	$A(q^{-1})y(t) = B(q^{-1})u(t) + \sin(10*u(t))$	Fig IV.8 Sortie	Montrer la **limite des performances de la MPC classique** à gérer des modèles non linéaires

Tableau récapitulatif des essais de simulation

Remarque : $A(q^{-1})=[\tau+(T_e-2\tau)q^{-1}-(T_e-\tau)q^{-2}]$ et $B(q^{-1})=T_e q^{-1}$

q^{-1} est l'opérateur de retard par rapportr à la période T_e dans la CP et t_e dans le cadre de la MPC (avec suréchantillonnage).

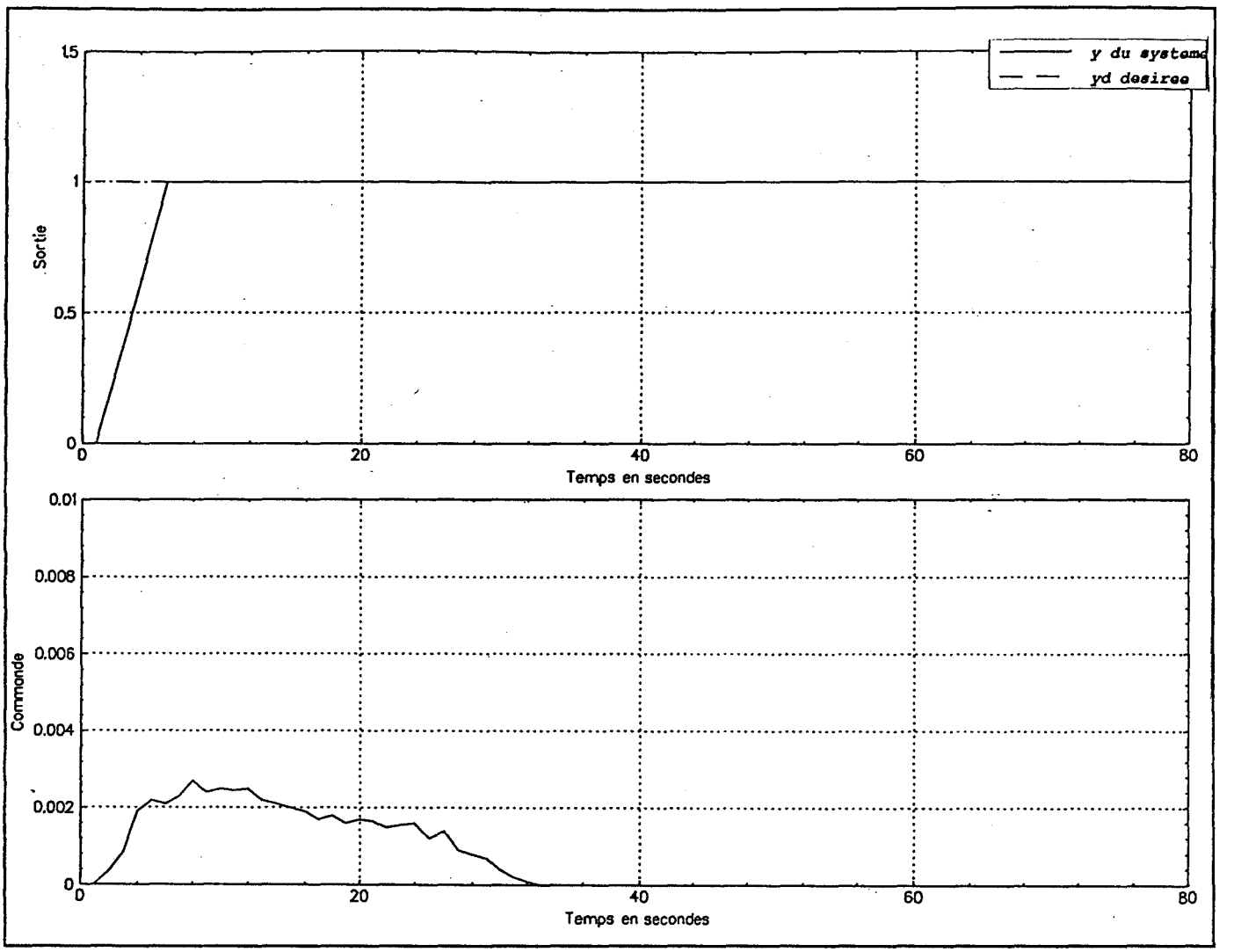


Fig IV.5 MPC/PNL appliquée au modèle Non Linéaire

La non linéarité dans le modèle du système a été bien gérée par le régulateur. A travers ces trois derniers exemples (Fig IV.3, Fig IV.4 et Fig IV.5), nous pouvons dire que la commande prédictive utilisant la PNL est apte à contrôler des systèmes non linéaires.

Afin de comparer les performances de la MPC/PNL à celles de la commande prédictive classique, nous avons réalisé les simulations suivantes :

* La MPC/PNL appliquée au modèle linéaire du processus thermique (Fig IV.6) représenté par : $[\tau+(T_e-2\tau)q^{-1}-(T_e-\tau)q^{-2}]*y(t) = T_e q^{-1} u(t)$.

* La MPC/MRM classique appliquée à ce même modèle (Fig IV.7).

* La MPC/MRM classique appliquée au modèle bruité par sin(10*u(t)) (Fig IV.8). Dans ce dernier cas, la commande est calculée pour $[\tau+(T_e-2\tau)q^{-1}-(T_e-\tau)q^{-2}]*y(t) = T_e q^{-1} u(t)$, mais envoyée à $[\tau+(T_e-2\tau)q^{-1}-(T_e-\tau)q^{-2}]*y(t) = T_e q^{-1} u(t) + \sin(10*u(t))$.

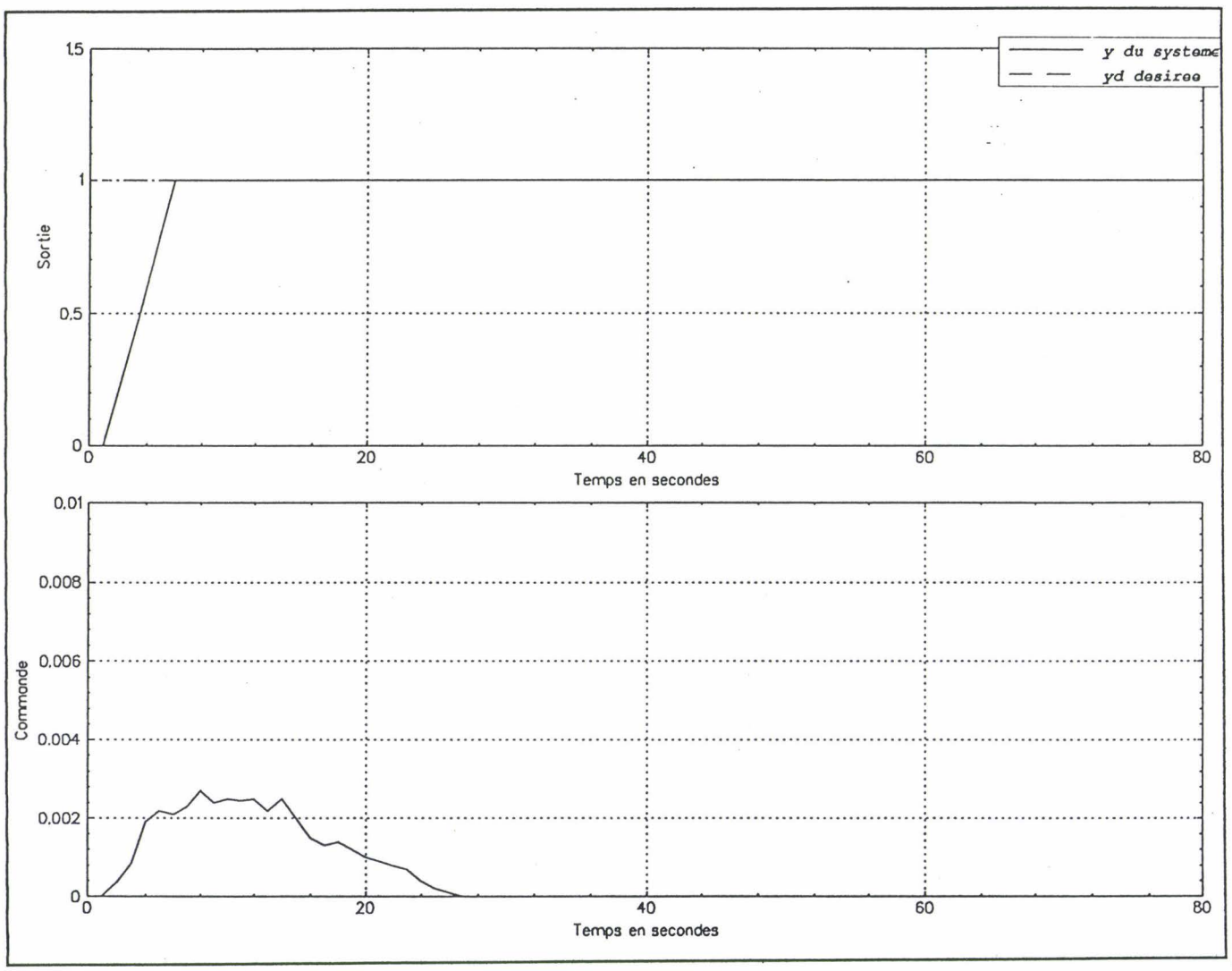


Fig IV.6 La MPC/PNL appliquée au modèle linéaire

Les courbes de la Fig IV.6 montrent les capacités de la MPC/PNL à réguler un système linéaire.

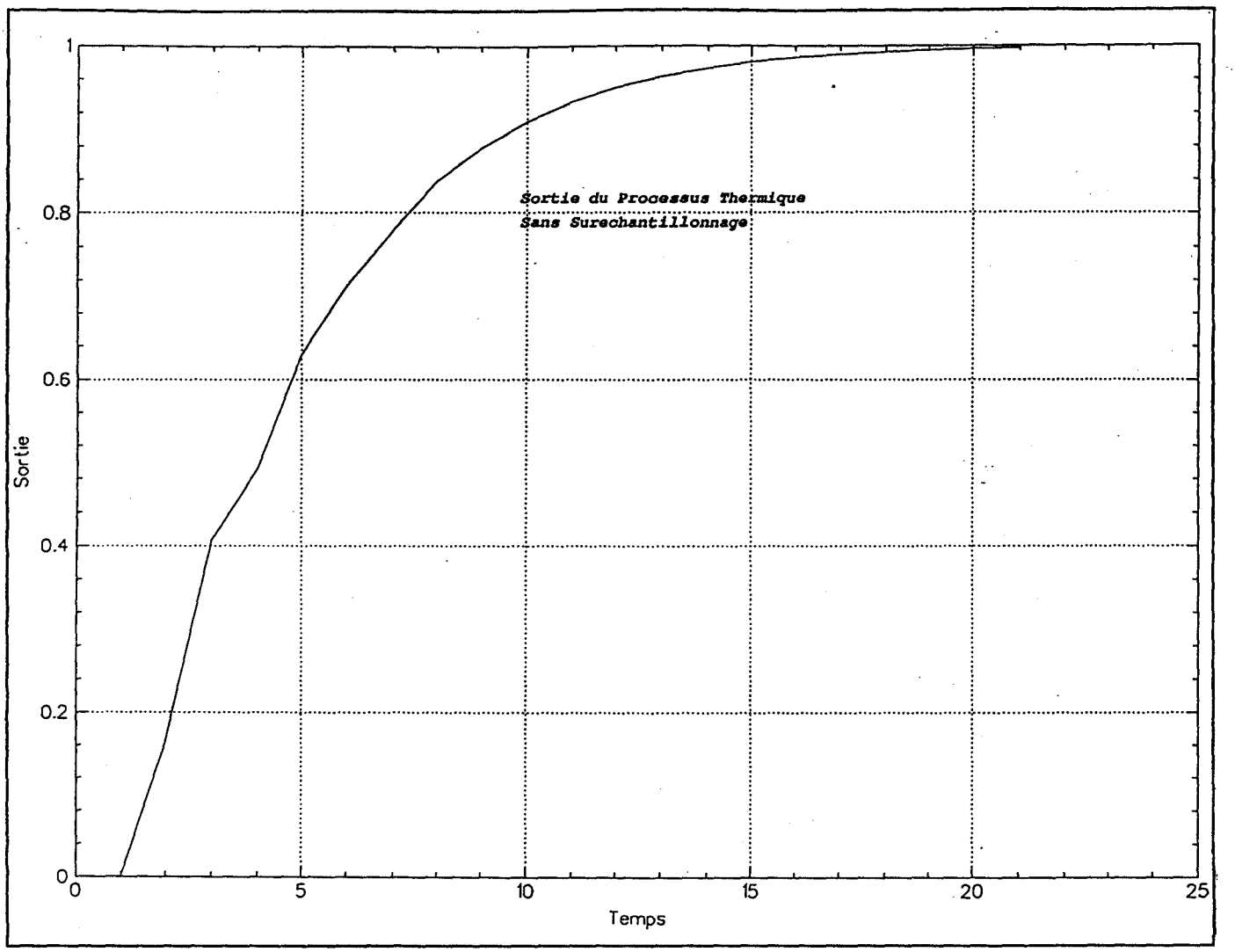


Fig IV.7 La MPC/MRM classique appliquée au processus thermique

La MPC/MRM est capable de contrôler le processus thermique. En comparant les courbes des Fig IV.6 et Fig IV.7, il s'avère que l'utilisation de la programmation non linéaire permet d'améliorer les résultats de la commande prédictive. Ce résultat était prévisible car la méthode MPC/PNL calcule la commande non seulement en minimisant le critère quadratique, mais en tenant compte aussi des contraintes qui font intervenir l'écart entre la consigne et la sortie. La MPC/MRM « classique » détermine la commande en minimisant le critère J. En fonction de l'erreur entre la sortie et la consigne, la commande de la méthode MPC/MRM est recalculée. C'est la raison pour laquelle la sortie du système atteint plus rapidement la consigne (Fig IV.6).

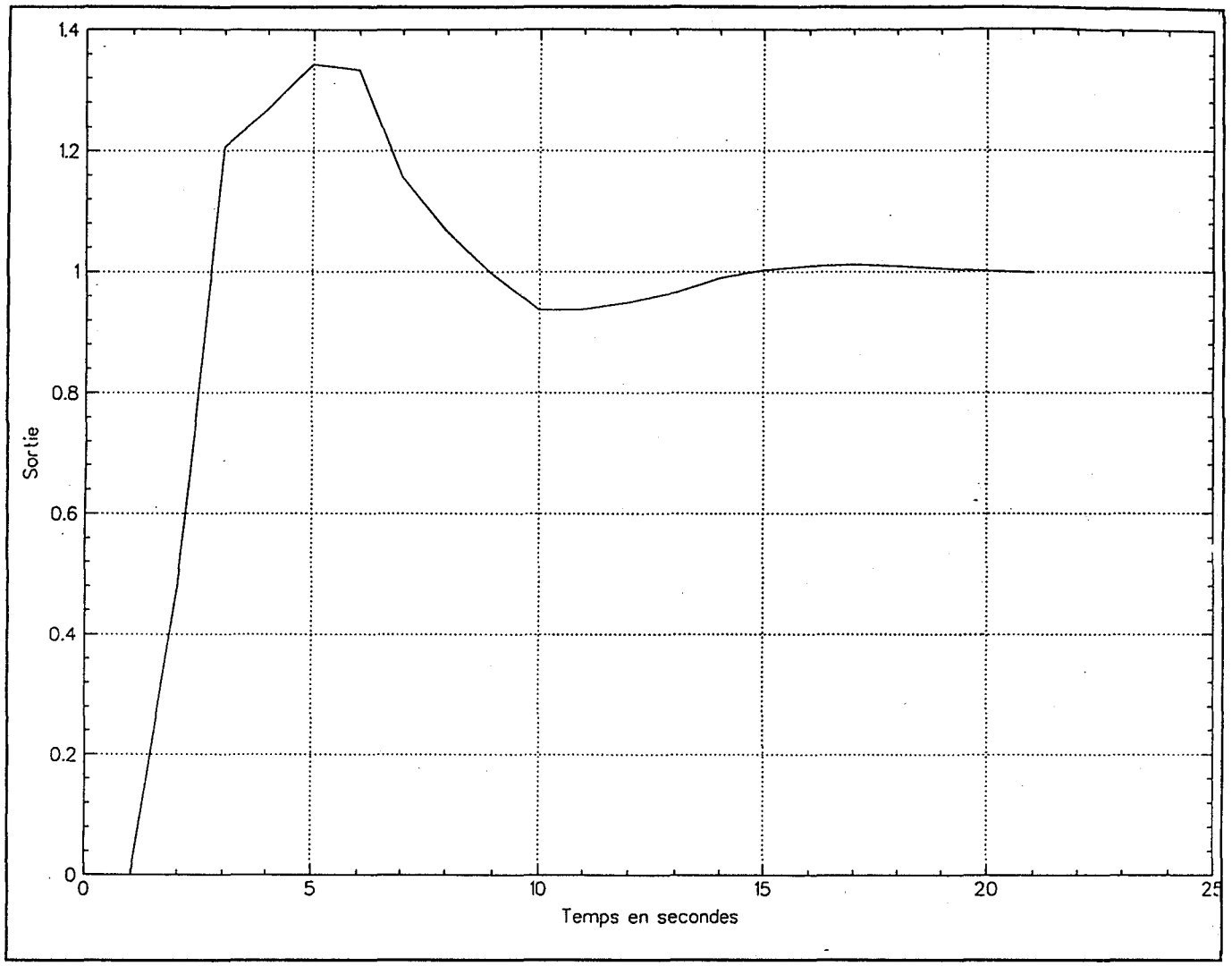


Fig IV.8 La MPC/MRM classique appliquée au système bruité

La courbe de la Fig IV.8 met en évidence la limite des capacités de la MPC/MRM, d'où le besoin à faire appel à de nouvelles techniques. Dans le chapitre 3, nous avons présenté les résultats du même test réalisé avec sin(u) et sans sin(10*u) (Fig III.4), la commande prédictive a pu maîtriser ce bruit.

IV.5) CONCLUSION

La Commande Prédictive, par sa capacité à réguler un processus industriel comportant un retard, autorise, par la programmation non linéaire, à mieux contrôler un processus non linéaire.

En effet, nous avons montré, au cours de cette étude, la remarquable aptitude de la commande prédictive par la programmation non linéaire, d'une part, à prendre en compte la non linéarité des processus à commander, et d'autre part, à anticiper l'évolution du processus afin d'atténuer l'effet du retard.

Aussi, la commande prédictive par la programmation non linéaire semble prometteuse en application industrielle. Cependant, elle nécessite des outils de calcul assez performant.

Notons que nous n'avons associé à cette commande aucune méthode supplémentaire lui permettant d'être plus performante, telle que l'ajustement automatique du facteur de pondération λ. Le comportement désiré que nous avons choisi, offre une marge de robustesse de stabilité suffisante pour supporter les bruits rajoutés. Les courbes correspondantes aux systèmes bruités le confirment.

Nous pouvons dire que les résultats de cette étude sont plutôt encourageants. Aussi, serait-il intéressant de poursuivre nos investigations sur la robustesse de stabilité (aussi bien du point de vue qualitatif que quantitatif) de la commande prédictive par la programmation non linéaire, en fonction du choix des valeurs des paramètres tels que N_u et N_y ou sur le choix des paramètres de la programmation non linéaire.

REFERENCES

[Ablow 1955]:
Ablow C.M, Brighman G.
« An Analog Solution of Programming Problems », Operation Research 3, 4, 1955, pp. 388-394.

[Bellman 1957]:
Bellman R.
« Dynamic Programming », Princeton University Press 1957.

[Courant 1943]:
Courant R.
« Variational Methods for Thee Solution of Problems of Equilibrium and Vibrations », Bull. Amer. Soc. 49, 1943, pp. 1-23.

[Dantzig 1949]:
Dantzig G.B.
« Programming in a Linear Structure », Econometrica 1949, Vol. 17, n°1.

[Fiacco 1968]:
Fiacco A. V, McCormick G. P.
« Nonlinear Programming : Sequential Unconstrained Minimization Techniques », John Willey, New York 1968.

[Gomory 1958]:
Gomory R.E.
« An Algorithm for Integer Solutions to Linear Programs », Princeton, IBM Mathematics Research Project 1958, Technical Report, n°1.

[Hestenes 1969]:
Hestenes M. R.
« Multiplier and Gradient Methods », Journal of Optimization Theory and Applications 1969, Vol. 4, pp. 303-320.

[Koncar 1996]:
Koncar V, Koubaa M.A, Legrand X, Bruniaux P, Vasseur C.
« Predictive Control with Non Linear Optimization: Application to a Thermal Process », 11th International Conference Systems Engineering, ICSE'96, 9-11 july 1996, Las Vegas USA. (acceptée).

[Kuhn 1951]:
Kuhn H.W, Tucker A.W.
« Nonlinear Programming », Econometrica 1951, Vol. 19, pp. 50-51.

[Luenberger 1973]:
Luenberger D.G.
« Introduction to Linear and Nonlinear Programming », Addison-wesley 1973.

[Minoux 1983]:
Minoux M.
« Programmation mathématique, théorie et algorithmes » (2 volumes), Tome 1, ouvrage paru dans la collection technique et scientifique des télécommunications, Dunod 1983, ISBN-2-04-015487-6.

[Nakayama 1975]:
Nakayama H, Sayama H, Sawaragi Y.
« A Generalized Lagrangian Function and Multiplier Method », Journal Optimization Theory and Appl 1975. Vol. 17, n° 3/4, pp. 211-227.

[Powell 1969]:
Powell M. J. D.
« A Method for Nonlinear Constraints in Minimization Problems, in Optimization », R. Fletcher ed., Academic Press, New York 1969, pp. 283-298.

[Rockafellar 1973]:
Rockafellar R. T.
« A Dual Approach to Solving Nonlinear Programming Problem by Unconstrained Optimization », Mathematical Programming 1973, Vol. 5, pp. 354-373.

[Zangwill 1969]:
Zangwill W.I.
« Nonlinear Programming : An Unified Approach », Prentice Hall 1969.

CHAPITRE 5
AUTOCLAVE DE TEINTURE

Chapitre 5

L'AUTOCLAVE DE TEINTURE

SEPROTEC

V.1) INTRODUCTION

L'autoclave de teinture de marque SEPROTEC a été choisi comme support industriel pour appliquer les méthodes de commandes développées dans cette étude. Certes, la conduite de la teinture par épuisement de bain a été, jusqu'à ce jour, conditionnée par l'évolution technologique du matériel, en vue de la recherche de meilleures performances. C'est la raison pour laquelle, nous avons équipé cet autoclave de capteurs plus performants que ceux qui étaient déjà montés afin d'avoir le maximum de précisions. L'ensemble est commandé par un ordinateur de type PC, équipé d'une carte CAN/CNA de type DCI Smartlab 8255 et d'une carte PIA nécessaire pour les entrées/sorties logiques.

Dans un premier temps, nous présentons les caractéristiques techniques de l'appareil, avant et après les transformations ainsi que les spécifications des nouveaux capteurs.

Dans un second temps, nous précisons les conditions opératoires des essais ainsi que le type de la méthode de commande utilisée et nous finirons par la présentation des courbes obtenues en temps réel pendant les essais de teinture.

Enfin, les problèmes rencontrés lors de cette application et les perpectives sont présentés.

V.2) CARACTERISTIQUES TECHNIQUES

Le schéma de base de l'appareillage proposé, est représenté par la figure V.1 Celui-ci comprend :

V.2.1) PARTIE MECANIQUE

1- Une cuve amovible (1), étudiée spécialement pour chaque cas particulier de dimensions et de poids de matière à teindre (capacité 1 bobine croisée, d'un poids maximum de 1Kg), est prévue pour procurer le rapport de bain minimum désiré (de 1 à 10). La variation ultérieure du rapport de bain s'effectue par l'addition de viroles amovibles entre la cuve et son support.

2- Un propulseur centrifuge (2), en acier inoxydable moulé AISI 316, permettant d'obtenir un débit de $6m^3/h$ pour une perte de charge à travers la matière de 2.5 bars environ. La puissance du moteur est de 1825 W.

3- Un échangeur de température tubulaire (3), d'une longueur d'environ 600 mm, de faible volume et de rendement élevé, permettant à la fois le réchauffage à une vitesse de 6°C/mn, et le refroidissement.

4- Un réponchonneur ouvert (4), d'une capacité d'environ 10 litres, permettant :

* soit la préparation du bain avant la teinture.
* soit sa récupération.

5- Une pompe volumétrique de pression statique (5), à débit réglable entre 0 et 60l/h.

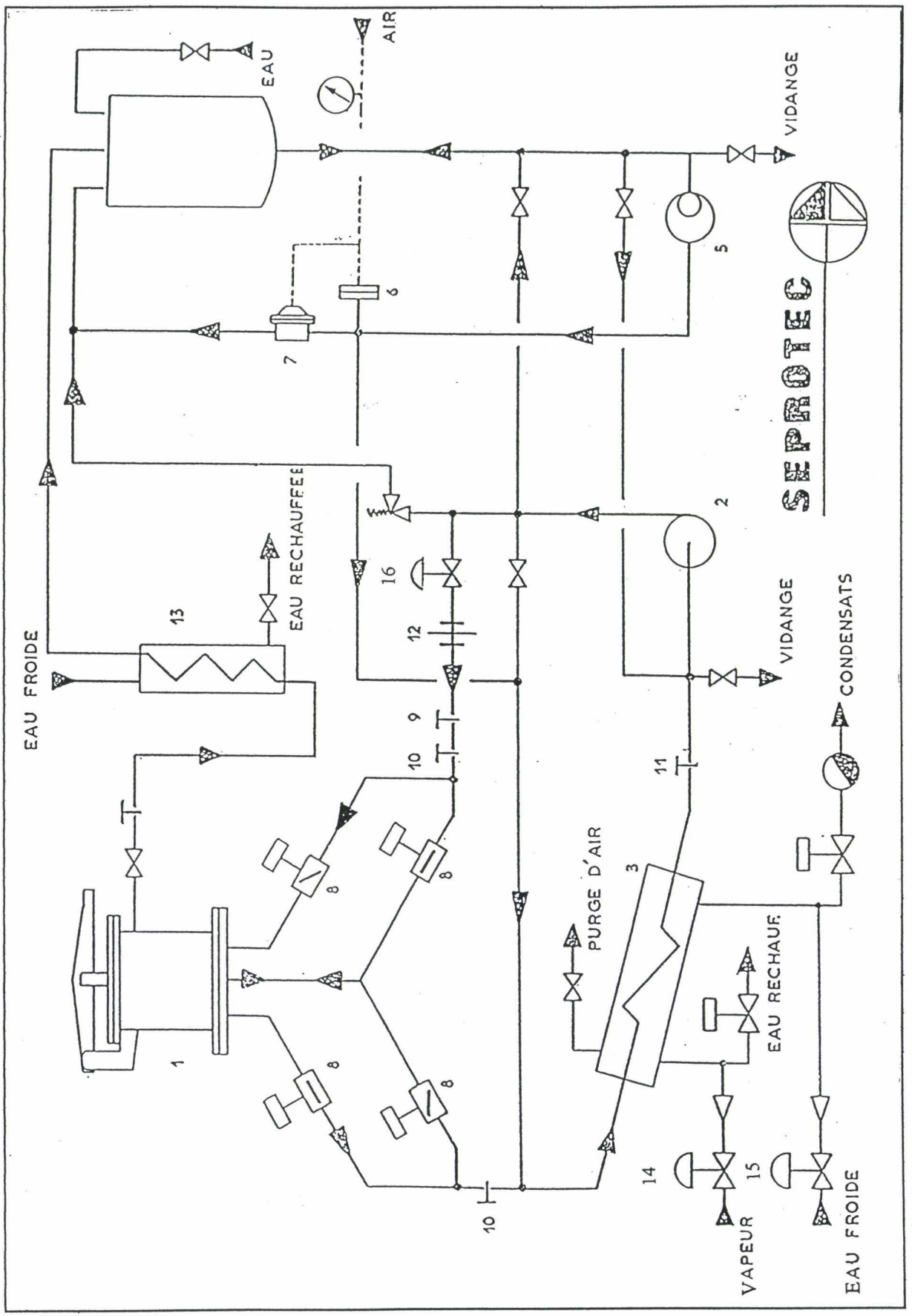


Fig V.1 Schéma de l'autoclave

Le circuit de sortie de cette pompe volumétrique est muni :

* d'un amortissement de pulsation (6).

* d'un régulateur de pression (7) permettant d'obtenir une pression statique constante, quelque soit le débit de recyclage extérieur (de 0 à 4 bars). La puissance du moteur est de l'ordre de 365 W.

6- Un ensemble de vannes d'inversion du sens de circulation (8). Ce dispositif, élaboré à cet effet, permet tout en assurant une parfaite séparation entre les circuits d'entrée et de sortie, de rendre possible une fréquence d'inversion très élevée, de l'ordre de 6 inversions à la minute. Il comprend l'ensemble des prises :

* de pression statique (9)
* de pression différentielle (10)
* de mesure de le température (11)
* de mesure du débit (12)

suivant une disposition telle que le contrôle de ces paramètres soit réalisé avec une grande précision, quel que soit le sens de circulation.

7- Un réfrigérant de retour de recyclage (13)

On trouvera en annexe une documentation détaillée.

V.2.2) INSTRUMENTATION DU PROCEDE ET ORGANISATION DES ENTREES/SORTIES

L'équipement comprend en entrée :

- un débitmètre (12) à affichage digital indiquant le débit en m^3/h. La valeur du débit est affiché à l'écran en temps réel.

- des capteurs de pression (9) et (10) sont installés aux mêmes emplacements. Les informations captées sont récupérées par l'ordinateur, la consigne étant fixée par l'utilisateur.
- un détecteur de niveau installé dans le réponchonneur (4), l'information vide/plein est récupérée par l'ordinateur.
- une sonde de température PT100 (11) est installée, elle est équipée d'un afficheur digital indiquant la température réelle du bain avant qu'elle ne soit affichée à l'écran.
- un dispositif de contrôle en continue du pH red/ox avec compensation de température est installée entre (12) et (9), la valeur du pH est affichée à l'écran. Ce dispositif comprend un afficheur digital indiquant la valeur du pH et la température du bain. Celle-ci est prise en un point différent de la sonde de température, elle permet une mise au point parfaite de la commande.
- un enregistrement des valeurs mesurées dans un fichier correspondant au procédé de teinture.

Par ailleurs, cet équipement comprend en sortie :

- l'ensemble des vannes de chauffage (14) et de refroidissement (15), leur commande aux instants voulus est gérée par l'ordinateur suivant le logiciel de commande.
- une vanne proportionnelle (16) permettant de réguler le débit souhaité. La commande de cette vanne est réalisée par le PC.
- l'ensemble de vannes d'inversion du sens de circulation (8) géré par le PC.

L'autoclave de teinture est entièrement automatisé, l'ordinateur gère à cet effet :

* toutes les commandes des électrovannes et des pompes.
* le temps d'acquisitions des valeurs de température, de pression, du débit et du pH.
* l'envoi et la durée de la commande de chauffage ou de refroidissement.
* l'affichage des valeurs récupérées à l'écran sous forme de courbe en temps réel.
* l'enregistrement de toutes les informations dans un fichier correspondant.

V.2.3) EQUIPEMENT OPTIONNEL

Comme il a été indiqué en introduction, cet autoclave sera équipé :

* d'un système de **mesure de nuance** par réflexion sur un échantillon de support textile. Cette mesure de nuance de consigne, servira de base pour la détermination de la conduite de température.

* d'un système de **mesure des saturations colorimétriques** par transmittance du bain de teinture dans l'autoclave.

* d'un **convertisseur de spectre en température** sous forme de logiciel **géré par le même PC** afin de comparer le spectre de la couleur du bain au spectre de consigne (déterminé par le système de mesure de nuance), et d'en déduire la nouvelle consigne de température.

V.3) TYPE DE COMMANDE

Deux types de commandes sont utilisées :

- la commande prédictive à multiples modèles de références à paramètre de pondération auto-ajustable, la dynamique globale du système étant choisie suivant l'étude présentée au chapitre 3.

- Afin de résoudre les problèmes liés, d'une part à la mauvaise identification des paramètres du modèle, et d'autre part à la difficulté de la modélisation de l'autoclave de teinture, nous avons utilisé **la commande pratique à haut gain** présentée ci-dessous. Le concept de cette commande est basé sur la représentation des systèmes discrets dans l'espace d'état.

Considérons que le système à commander est représenté par les équations (V.1) et (V.2) suivantes :

$$X(k+1)=AX(k)+BU(k) \quad (V.1)$$

$$Y(k)=CX(k) \quad (V.2)$$

où, dans le cas général

A : matrice d'état (n x n)

B : matrice (n x r) (n lignes , r colonnes)

C : matrice (p x n)

U : vecteur de commande (r x 1)

Y : vecteur de sortie (p x 1)

X : vecteur d'état (n x 1)

conformément au schéma de commande suivant :

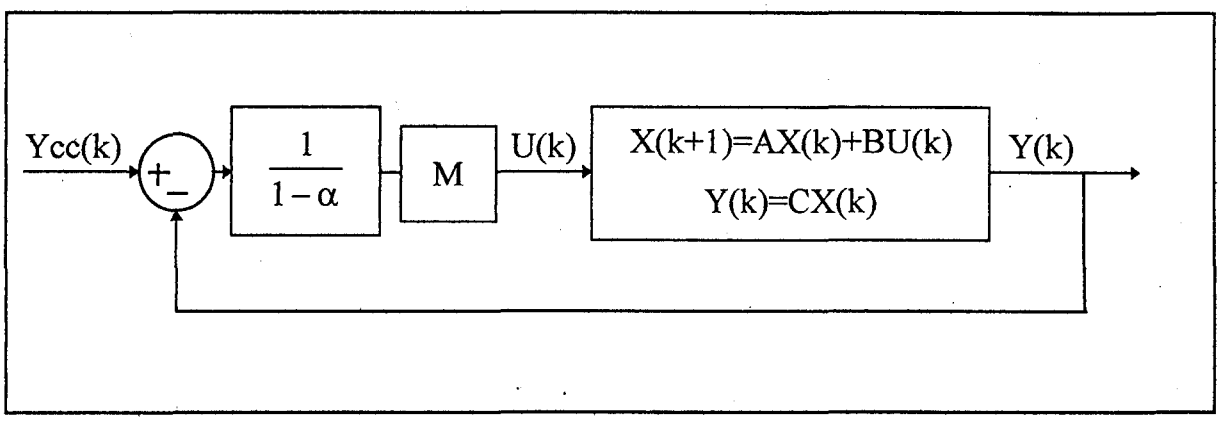


Fig V.2 Schéma de commande

dans lequel Ycc(k) désigne la consigne de sortie.

Soit $M=B^T * C^T$ (M est une matrice r x p), nous avons alors :

$$X(k+1) = AX(k) + \frac{1}{1-\alpha} B * M * (Ycc(k) - Y(k)) \quad (V.3)$$

$$X(k+1) = (A - \frac{1}{1-\alpha} B*M*C)*X(k) + \frac{1}{1-\alpha} B*M*Ycc(k) \quad (V.4)$$

Lorsque α tend vers 1 la matrice d'état $(A - \frac{1}{1-\alpha} B*M*C)$ est équivalente à $(-\frac{1}{1-\alpha} B*M*C)$. En effet, dans ce cas, les coefficients de la matrice $(-\frac{1}{1-\alpha} B*M*C)$ sont prépondérants vis à vis de ceux de la matrice A et la dynamique interne du système devient négligeable. Le système obéit alors au schéma de commande suivant (Fig V.3) :

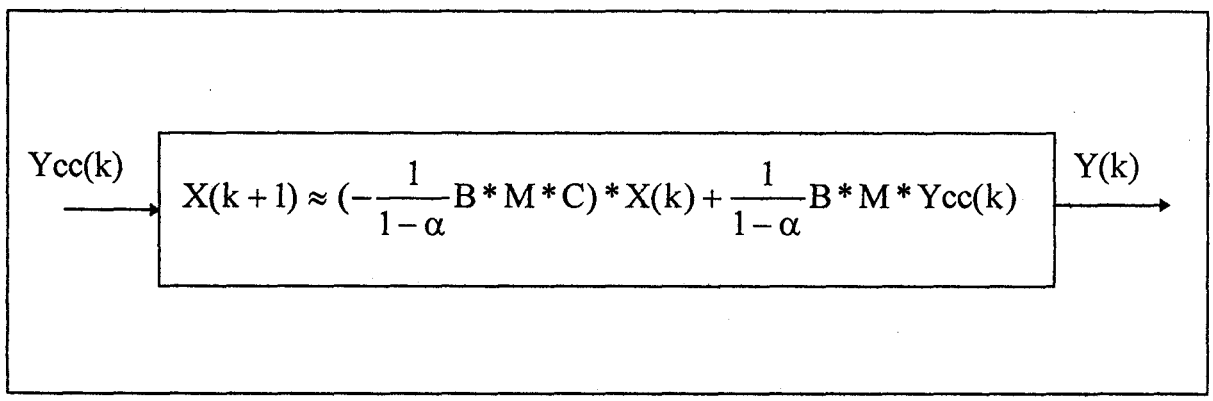


Fig V.3 Schéma de commande équivalent

soit :

$$X(k+1) = (-\frac{1}{1-\alpha} B*M*C)*X(k) + \frac{1}{1-\alpha} B*M*Ycc(k) \quad (V.5)$$

c'est-à-dire

$$(1-\alpha)*X(k+1) = -B*M*C*X(k) + B*M*Ycc(k) \quad (V.6)$$

soit en tenant compte de l'équation (V.2) :

$$(1-\alpha)*X(k+1) = -B*M*Y(k) + B*M*Ycc(k) \quad (V.7)$$

lorsque α tend vers 1, nous avons :

$$-B*M*Y(k)+B*M*Ycc(k)=0 \qquad (V.8)$$

ou encore :

$$B*M*(Ycc(k)- Y(k))=0 \qquad (V.9)$$

Dés lors, si $B*M\neq 0$ (qui est une matrice n x p), l'équation (V.6) n'est vérifiée $\forall\ k \in \mathbf{N}$ et Ycc(k) qu'à la condition:

$$Y(k)=Ycc(k) \qquad (V.10)$$

PRISE EN COMPTE DES CONTRAINTES DE COMMANDE

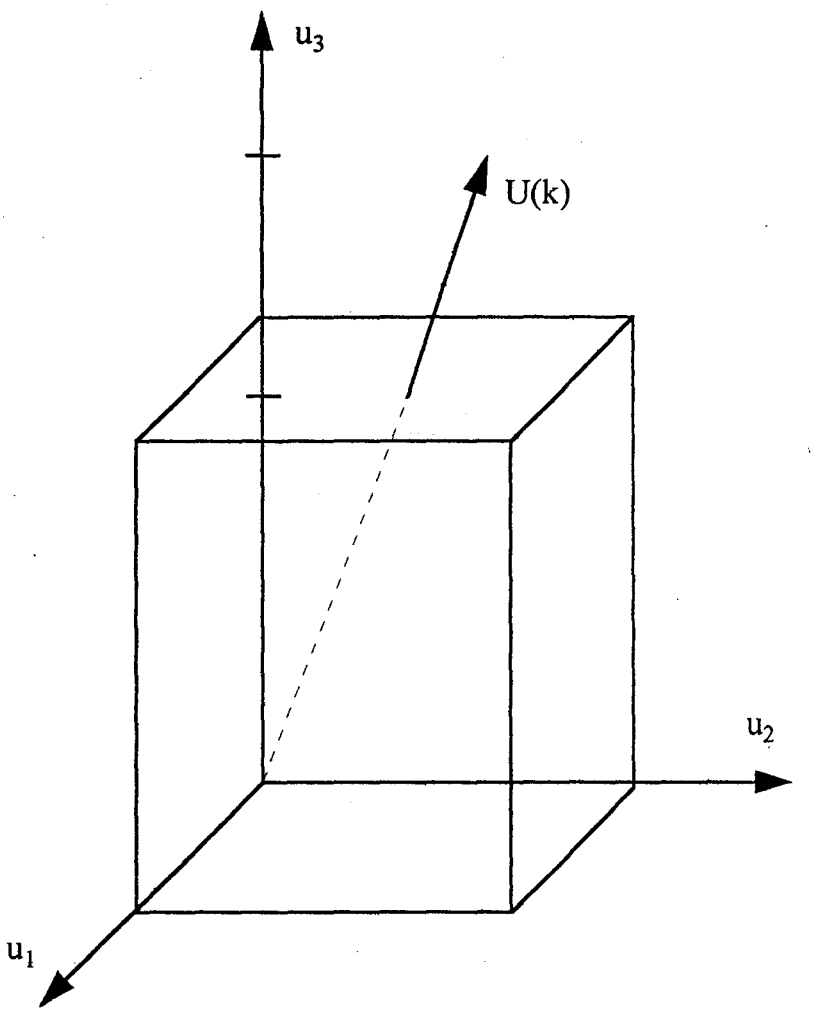
D'après le schéma de commande de la figure V.2 l'expression de la commande est donnée par :

$$U(k) = \frac{1}{1-\alpha} * M * (Ycc(k) - Y(k)) \qquad (V.11)$$

Si nous notons $U(k)=(u_1(k), u_2(k), ..., u_r(k))^T$, en pratique les $u_i(k)$ (i=1, .., r) ne prennent pas des valeurs infinies et vérifient des contraintes du type :

$$m_i \leq u_i(k) \leq M_i \qquad (V.12)$$

Ainsi, dans l'espace des commandes de dimensions r, U(k) doit être pris à l'intérieur du parallélépipède généralisé noté PG, défini par l'équation (V.12).



PG

Pratiquement la commande de saturation peut être obtenue selon l'algorithme suivant :

$$U(k) = \frac{1}{1-\alpha} * M * (Ycc(k) - Y(k))$$

si U(k) $\notin$ PG, alors

faire U(k)=α*U(k) (α<1) tel que (V.12) reste vérifiée

V.4) DESCRIPTIONS DES ESSAIS

Nous avons utilisé la consigne correspondante à la conduite de température représentée par la figure V.4 suivante :

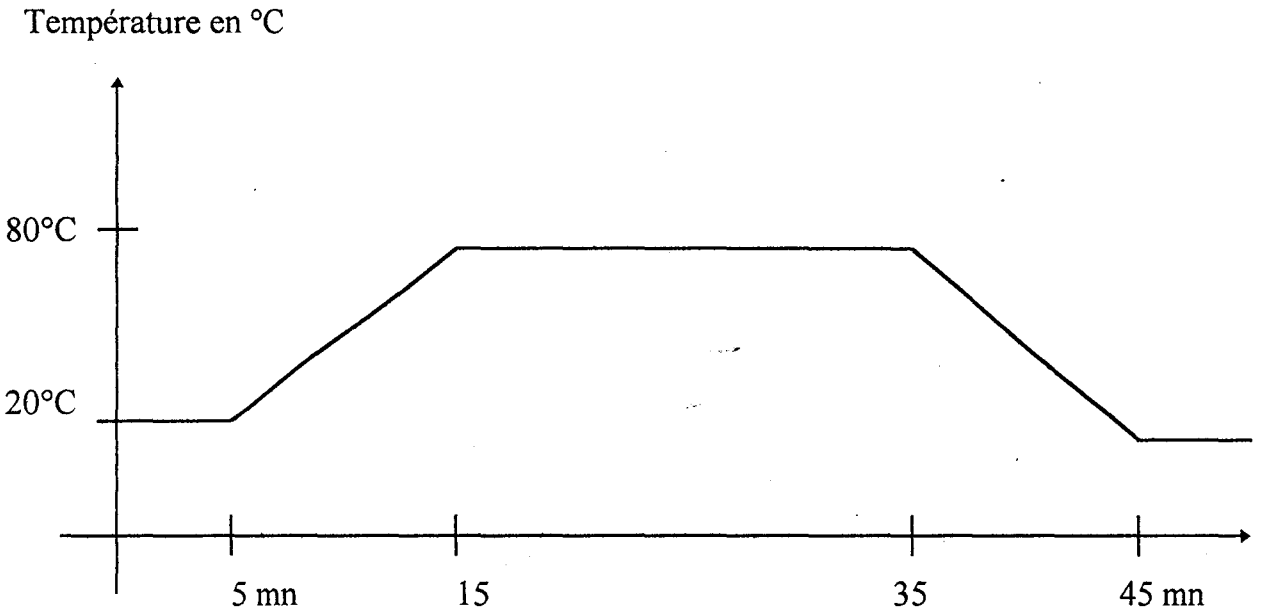


Fig. V.4 Conduite de température

Etant donnée que le chauffage et le refroidissement se font à travers l'échangeur thermique, nous avons utilisé le même modèle que pour le processus thermique avec une constante de temps τ' différente de τ, soit :

$$[\tau'+(T_e-2\tau')*q^{-1}-(T_e-\tau')*q^{-2}]*y(t) = T_e*q^{-1}\,u(t) \qquad (V.13)$$

avec T_e=15s et τ'=30

Certes, le modèle réel de l'autoclave de teinture est différent du modèle linéaire que nous avons adopté. Mais, comme nous avons utilisé la commande prédictive à paramètre λ auto-ajustable avec un choix judicieux de la dynamique globale du système, nous avons pu suivre la conduite de température souhaitée. Cette méthode de

commande a été développée dans le but de pouvoir contrôler des systèmes dont le modèle n'est pas parfaitement déterminé en donnant plus de poids à la commande et en faisant un choix judicieux de la dynamique globale assurant une certaine marge de robustesse. Le même modèle a aussi été utilisée lors de l'application de la commande pratique à haut gain.

V.5) RESULTATS DES ESSAIS

La commande prédictive

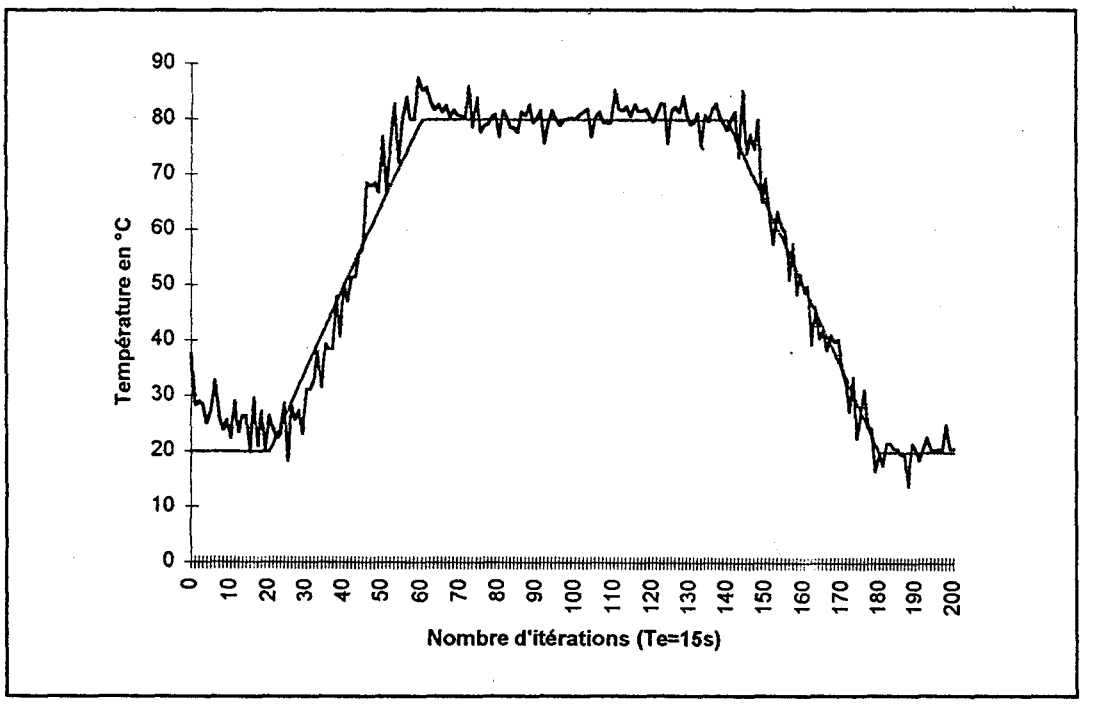


Fig V.5 Commande prédictive à facteur de pondération auto-ajustable

La température du bain au départ du cycle est de 38°C, alors que la consigne est de 20°C. Le système réagit de façon à atteindre la conduite de température imposée en refroidissant au maximum. C'est la raison pour laquelle la commande est en saturation jusqu'à la 15^{éme} itération.

A cause de la dynamique du système, à partir de la 21^{éme} période nous constatons que la température est inférieure à la consigne. La commande prédictive à paramètre de pondération auto-ajustable donne plus de poids à la commande afin de rattraper l'erreur entre la sortie et la consigne. Cette erreur est quasiment annulée au bout de 16 périodes d'échantillonnage c'est-à-dire 4 minutes.

Au niveau du palier, la présence des perturbations engendre une erreur de l'ordre de 7°C au maximum (à la 60^{éme} période), soit une variation de 8.75%. Il ne faut pas perdre de vue que le modèle du système choisi, ne correspond pas au modèle réel du système, c'est donc grâce à la capacité de la méthode de commande à atténuer les imperfections de la modélisation, que nous avons réussi à avoir une variation de 7°C sur 80°C. La dernière phase de contrôle ne pose pas de problème, grâce à l'association de l'ajustement automatique du facteur de pondération et de la prédiction.

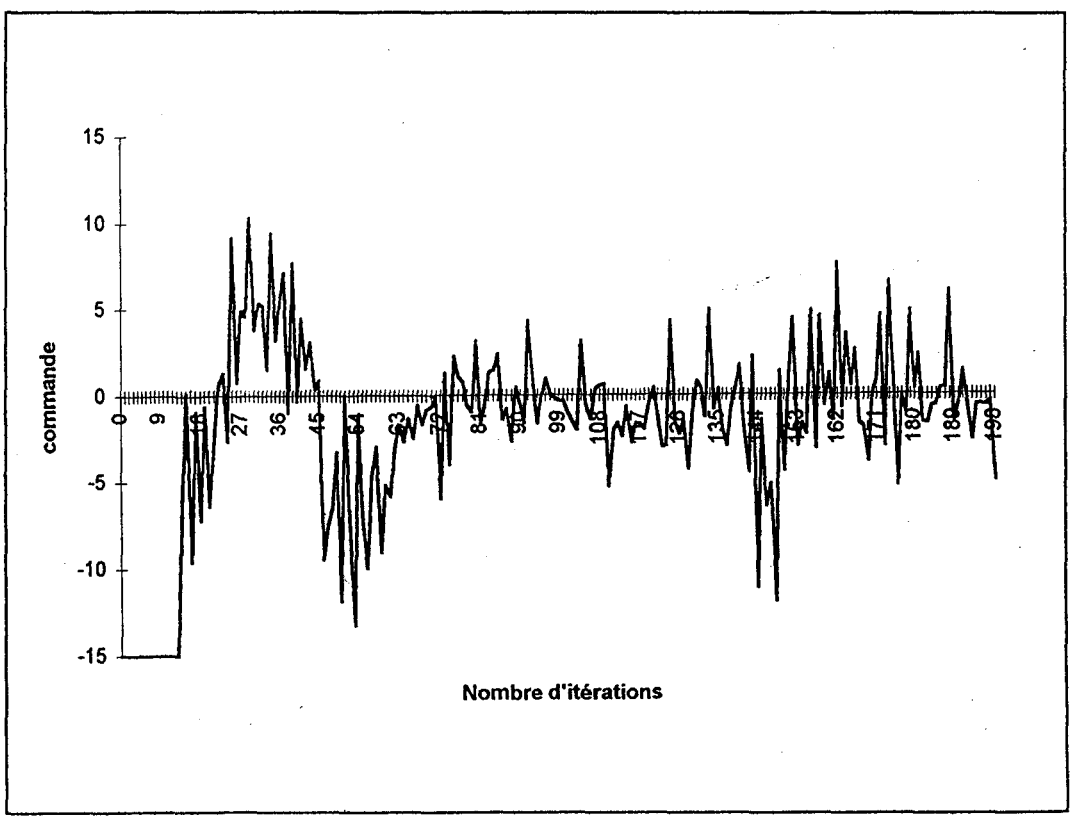


Fig V.6 Commande

La commande pratique à haut gain

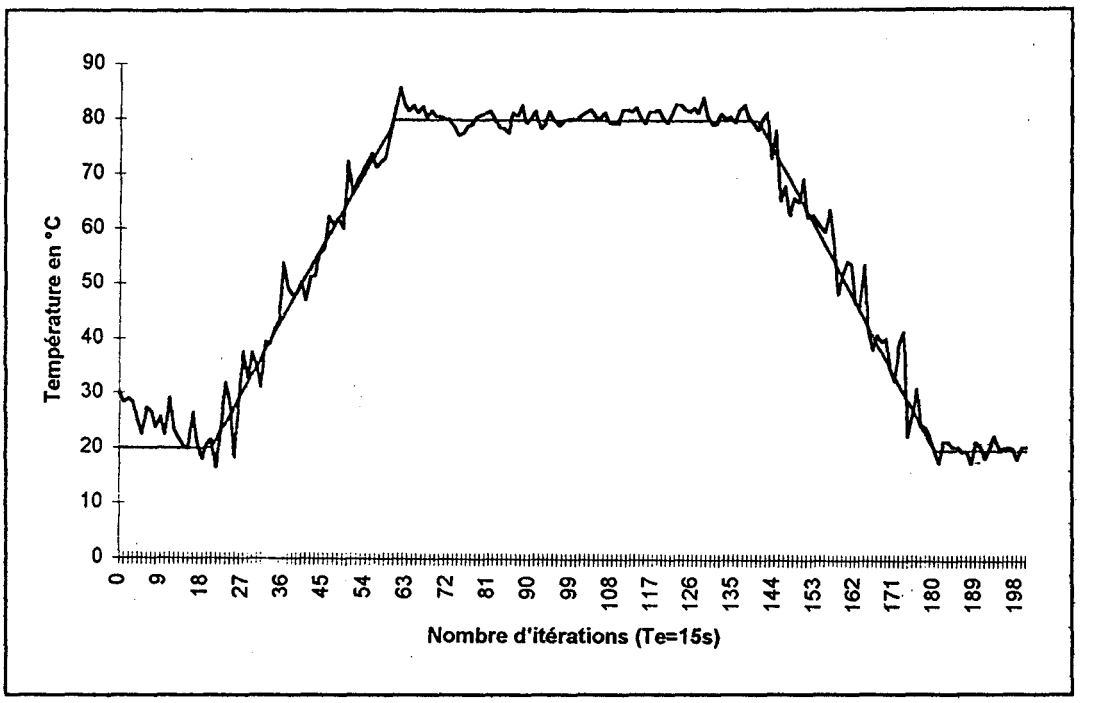


Fig V.7 Température du bain par la commande pratique à haut gain

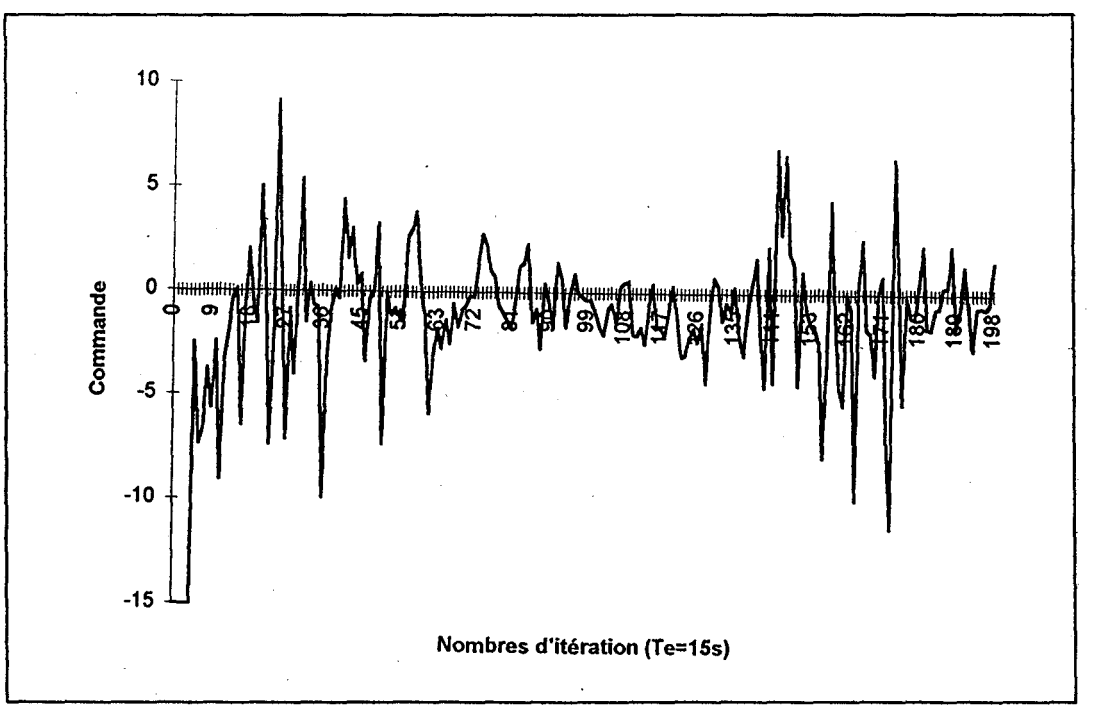


Fig V.8 Commande

La commande pratique à haut gain se révèle aussi capable que la commande prédictive à commander l'autoclave de teinture. En effet, le modèle du système importe peu dans cette commande, puisque nous faisons intervenir un haut-gain.

La sortie du système atteint plus rapidement la consigne avec la commande pratique à haut gain qu'avec la commande prédictive. En plus, quand la courbe accroche, la température du bain suit la conduite de température jusqu'à la fin du procédé.

Quant à la commande, elle reste en saturation moins longtemps que celle de la commande prédictive car au départ la température du bain était de l'ordre de 30°C (contrairement au cas précédent qui était de l'ordre de 38°C). L'erreur maximale est atteinte au niveau du changement de la valeur de la consigne (passage de la pente au palier), elle est de l'ordre de 6°C, soit 7.5%.

V.6) CONCLUSION

Suivant les résultats obtenus, la commande prédictive à paramètre de pondération auto-ajustable utilisée peut contrôler le processus de teinture. De même la commande pratique à haut gain permet le suivi de la conduite de température sans avoir un modèle mathématique exacte reflétant le comportement du système.

Nous avons appliquée la commande prédictive classique, mais les résultats obtenus sont loin d'être satisfaisants, la sortie ne suit pas la conduite de température surtout au départ. Il faut allonger davantage le premier palier de 20°C, afin de permettre à la commande de s'adapter au système. En effet, une valeur fixe du facteur de pondération garde un potentiel fixe de la commande et par suite il faut laisser du temps à la commande pour rattraper l'erreur entre la sortie et la consigne.

CONCLUSION GENERALE

Conclusion générale

Au terme de ce mémoire, nous sommes amenés à constater que le sujet traité offre un grand nombre d'applications.

A ce jour, l'application de la commande prédictive aux procédés industriels reste limitée, malgré tous les avantages qu'elle offre. L'élaboration de nouvelles versions plus performantes de cette commande et l'extension de son application aux systèmes non linéaires, représente de bonnes perpectives pour son utilisation dans le secteur industriel.

Les résultats présentés dans ce mémoire, font apparaître deux volets. Le premier volet traite des améliorations apportées à la commande prédictive. Le second volet traite de l'application de la commande pratique à haut gain à l'autoclave de teinture.

Dans un premier temps, nous avons développé une méthode permettant l'ajustement automatique du facteur de pondération de la commande. Cette méthode est basée sur le principe du suivi exponentiel imposée à l'erreur.

Dans un second temps, nous avons mis au point une stratégie aidant l'utilisateur à réaliser le choix de la dynamique désirée, à imposer au système. Cette stratégie permet à la fois de garantir et d'évaluer la robustesse de stabilité du système. Des

améliorations sont apportées à la commande prédictive à modèles de références multiples, appliquée aux systèmes linéaires.

Enfin, nous avons élargi le champ d'utilisation de la commande prédictive aux systèmes non linéaires en utilisant la programmation non linéaire.

En comparaison avec les différentes variétés de commande prédictive présentées dans cette étude, la commande pratique à haut gain parait plus facile à utiliser, du fait qu'elle n'exige pas la connaissance de la dynamique interne du système à commander.

Les régulateurs que nous proposons pour les systèmes linéaires ou non linéaires sont robustes, fiables et applicables à l'industrie. Leur couplage avec un procédé de mesure de nuance (ou de couleur), permet de réaliser une automatisation complète des cycles de teinture en temps réel, assurant de bons résultats.

Notre étude est, par conséquent, une étape vers la réalisation d'un système plus complet. Le fait de pouvoir utiliser ces méthodes de commande dans un environnement informatique, représente un progrès important pour l'avenir des machines de teintures.

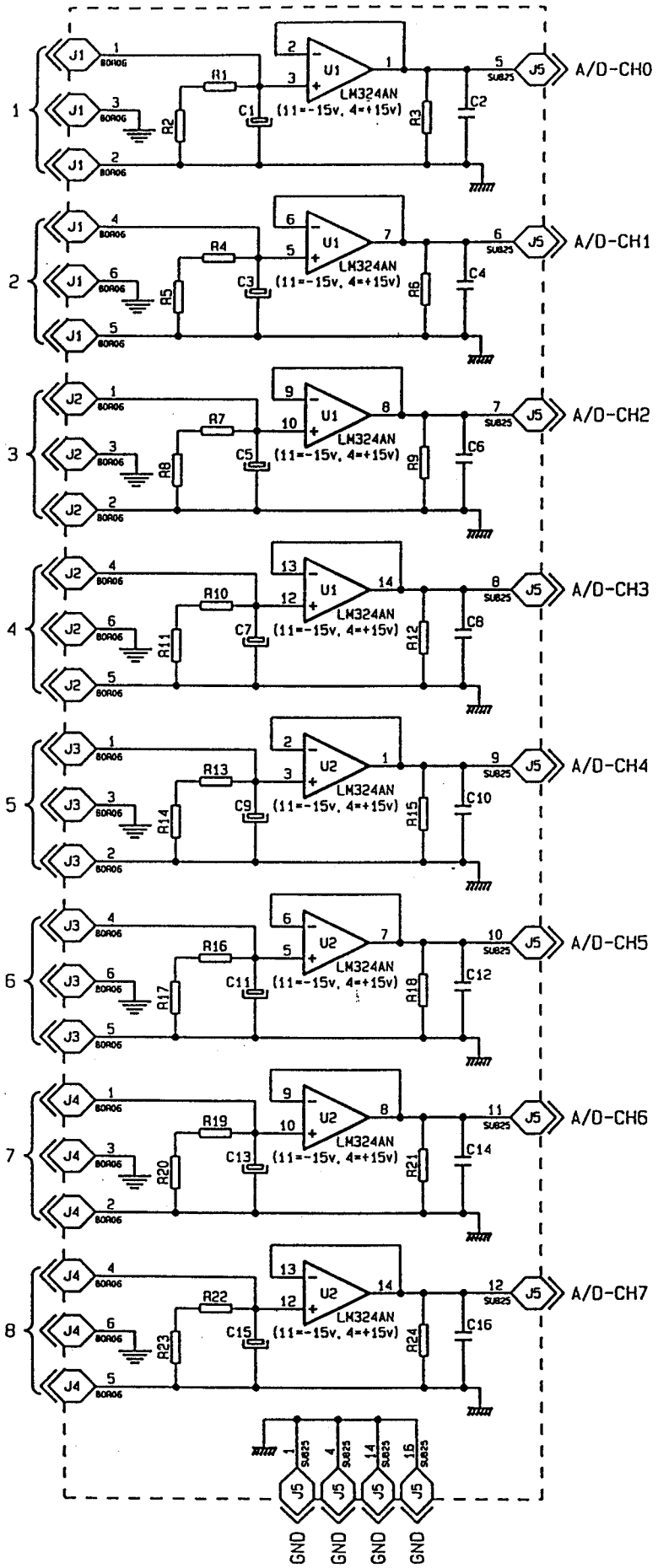
Les résultats de simulation montrent que les différentes commandes qui ont été présentées et testées, peuvent s'appliquer non seulement à l'industrie textile tel que nous l'avons envisagé au départ, mais également à d'autres industries (métallurgie, agro-alimentaire, ...).

Des travaux à réaliser dans l'avenir concernant le choix du comportement désiré non linéaire pour la commande prédictive utilisant la programmation non linéaire, peuvent être envisagés.

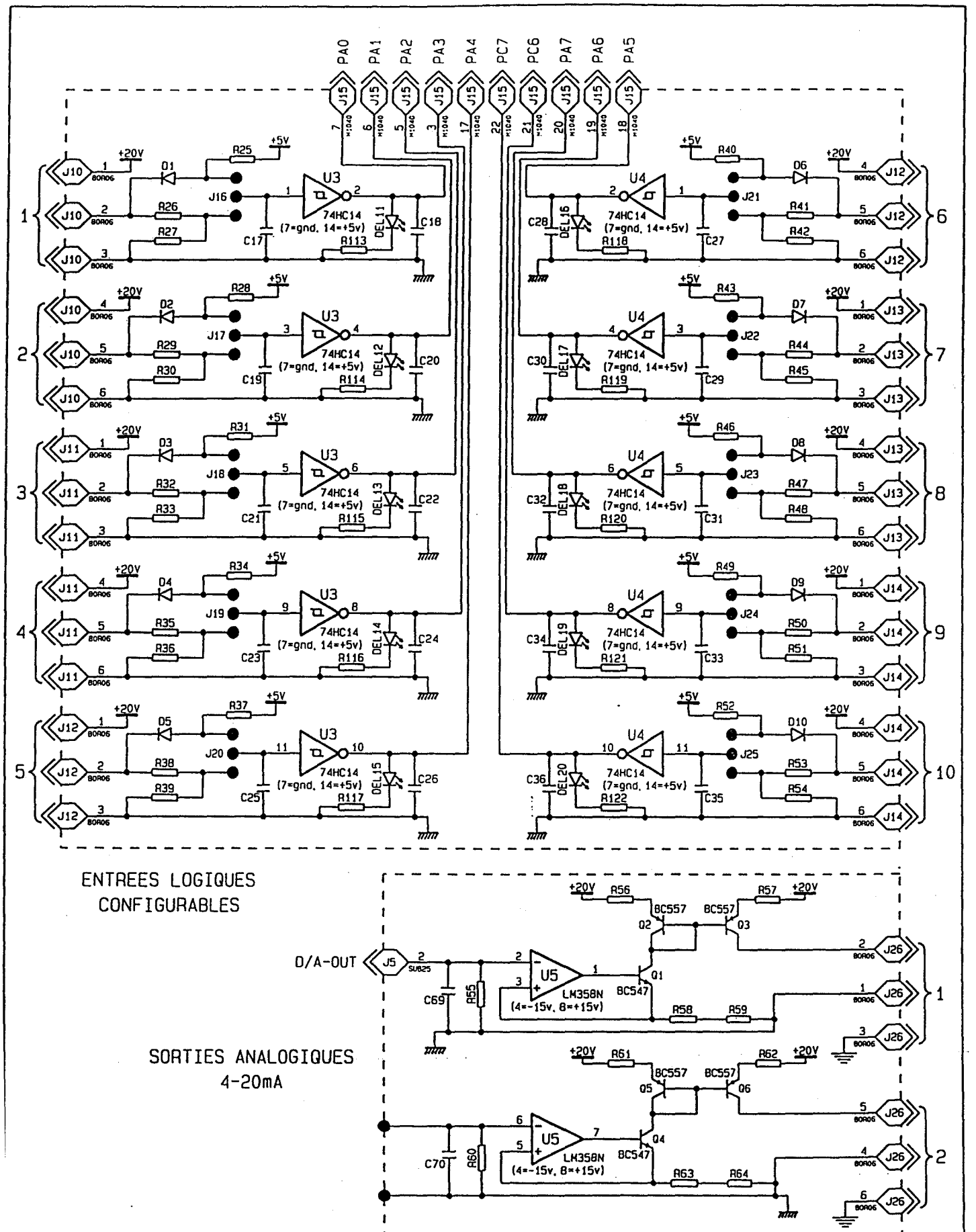
ANNEXE

CARTE ENTREE/SORTIE SLICE

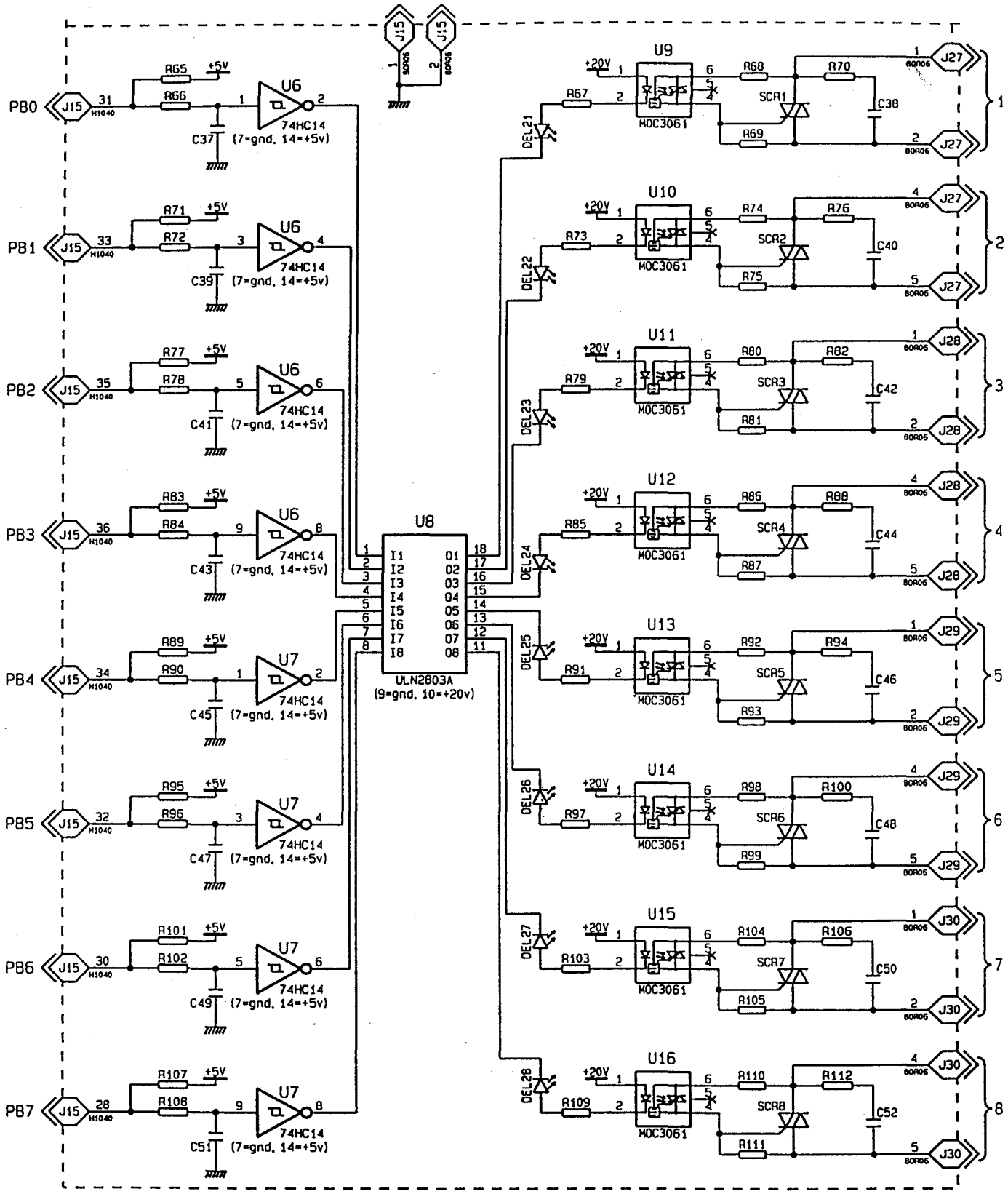
ENTREES ANALOGIQUES
4-20mA



SLICE		SCHEMA		Fait Par: PG	
ENS001				Le: 17/02/95	
				VER	DATE	PAR
				1		
A4	1/4	S0495	CARTE INTERFACE	2		

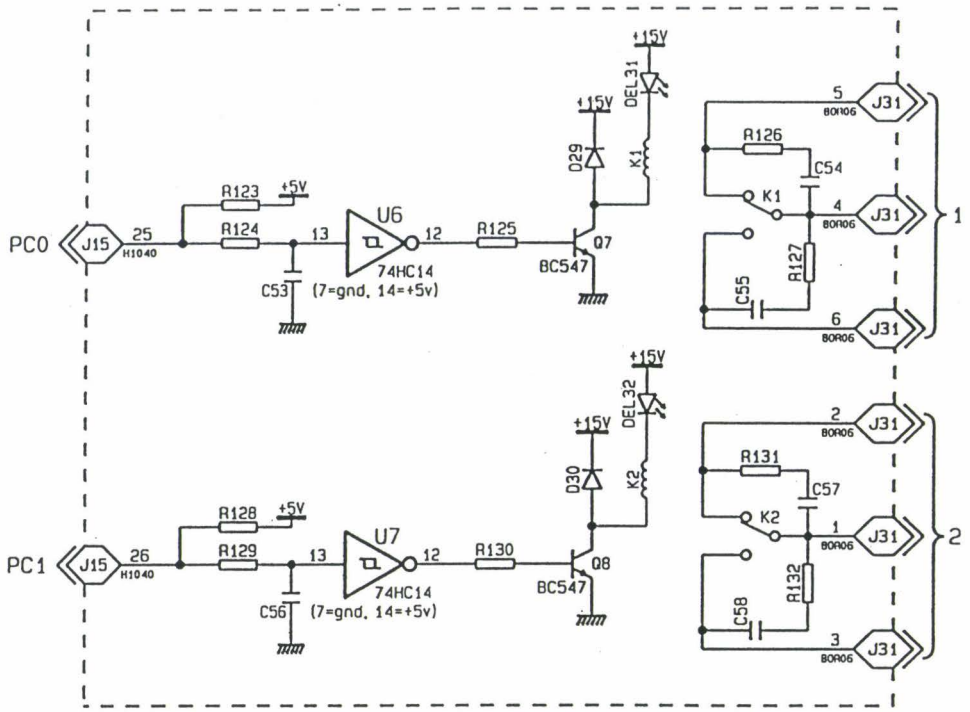


SLICE		SCHEMA		Fait Par: PG
ENS001				Le: 17/02/95
				VER / DATE / PAR
				1
A4	2/4	S0495	CARTE INTERFACE	2

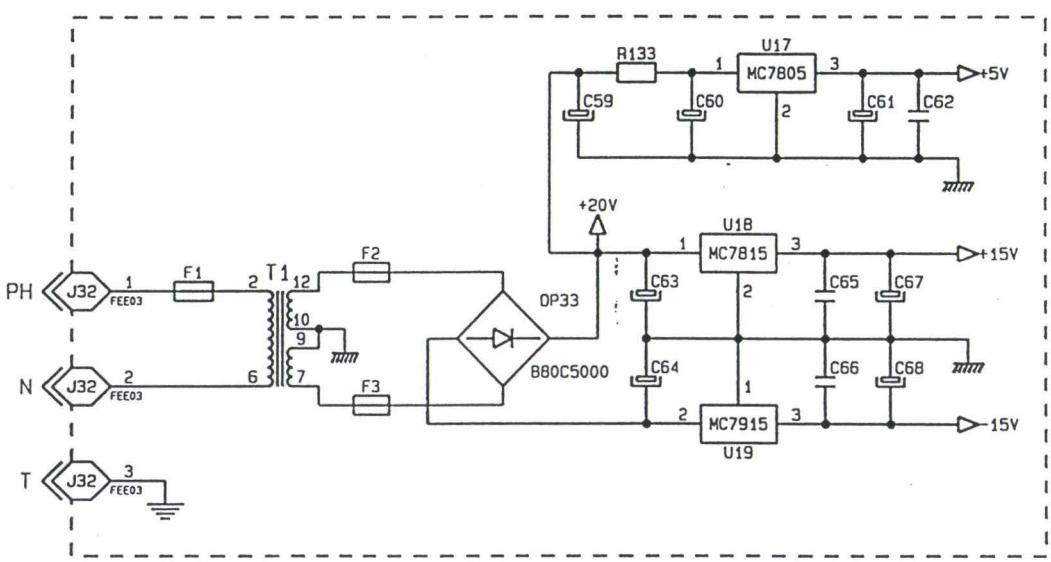


SORTIES LOGIQUES
POUR CONTACTEURS ET ELECTROVANNES 24VAC

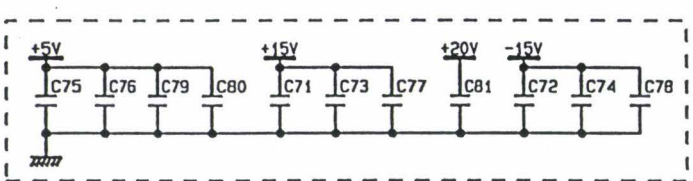
SLICE	SCHEMA			Fait Par: PG
ENS001				Le: 17/02/95
				VER / DATE / PAR
				1
A4 3/4	S0495	CARTE INTERFACE		2



SORTIES LOGIQUES SUPPLEMENTAIRE



ALIMENTATION



DECOUPLAGE

SLICE		SCHEMA		Fait Par: PG	
ENS001				Le: 17/02/95	
				VER	DATE	PAR
				1		
A4	4/4	S0495	CARTE INTERFACE	2		

SLICE	IMPLANTATION			Fait Par: PG	
ENS001				Le: 17/02/95	
Ech: 0.9				VER	DATE	PAR
A4	1/1	S0495	CARTE INTERFACE	1		
				2		

SCHEMA ELECTRIQUE DE L'AUTOCLAVE DE TEINTURE

TRANSMISSION DOCUMENTS D'ETUDES

ATEUR LE 23 NOV. 1979

PAGE USINE LE

MONTAGE EXTERIEUR LE

Ind.	Dates	Modifications

SEPROTEC

ENGINEERING

1_3, Place de la Gare_94-SUCY- EN
Tél.: 902-03-64

E.N.S.A.I.T.

ROUBAIX

2 PLACE DES MARTYRS DE LA RESISTANCE

SCHEMA - AUTOMATISME -

Echelle: Dessiné le: 5

Par:

ETUDE N°

COMMANDE N° CS 1

PLAN N° 185

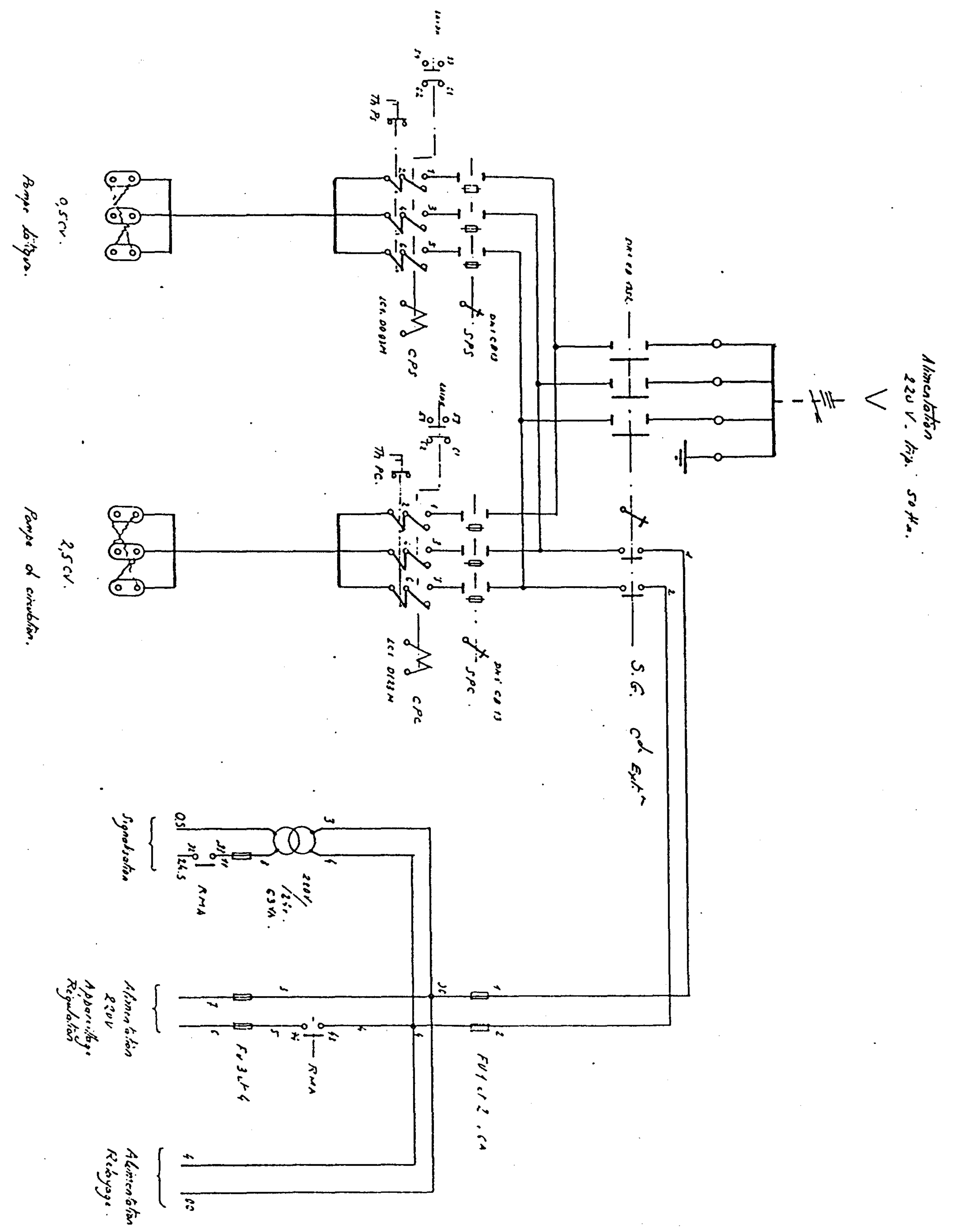
1 2 3 4 5 6 7 8 9 10 11 12 13 14

Suite de folios (N° de folio par dizaine par une x)

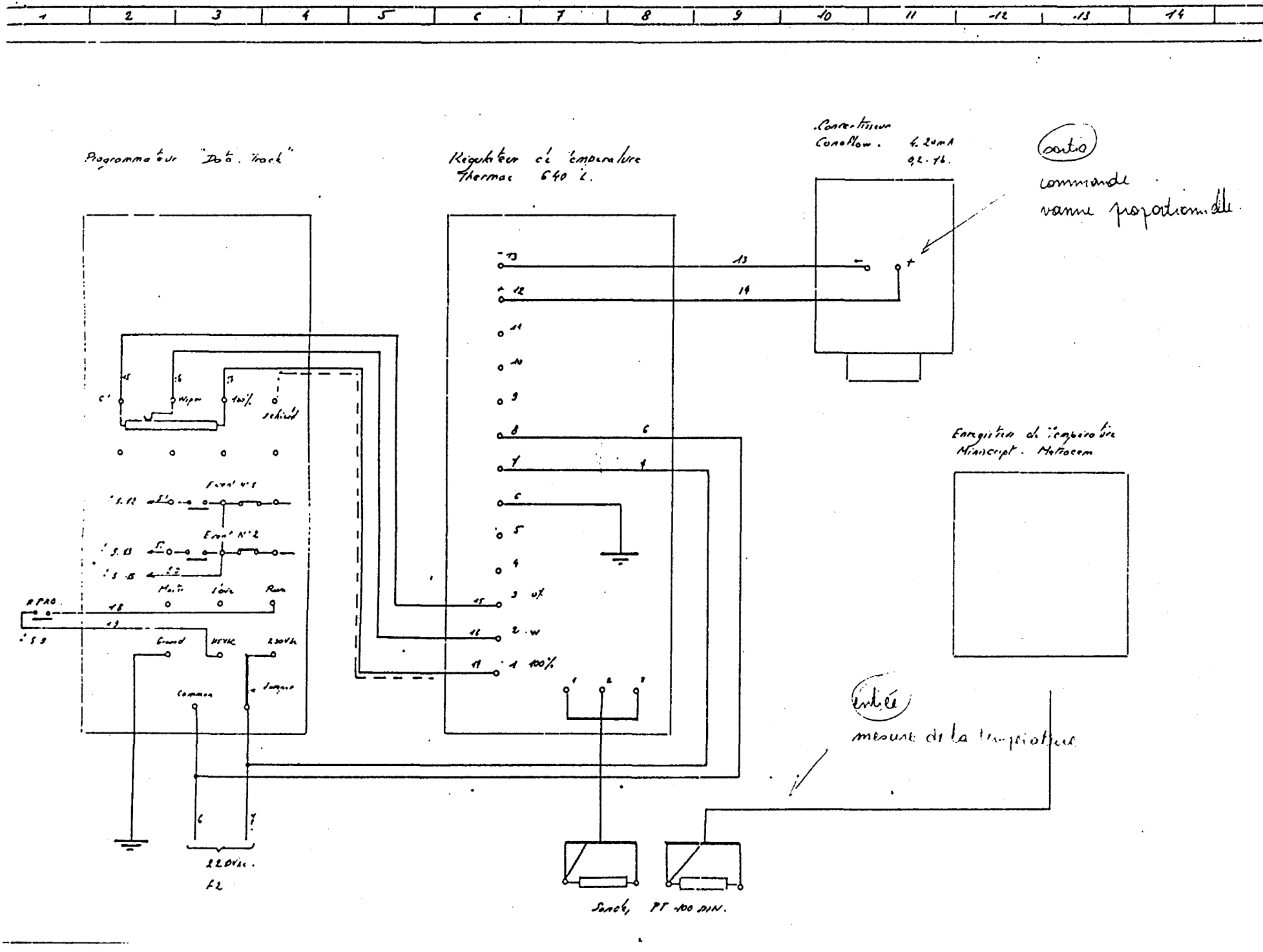
dizaine ↓ / Unités →

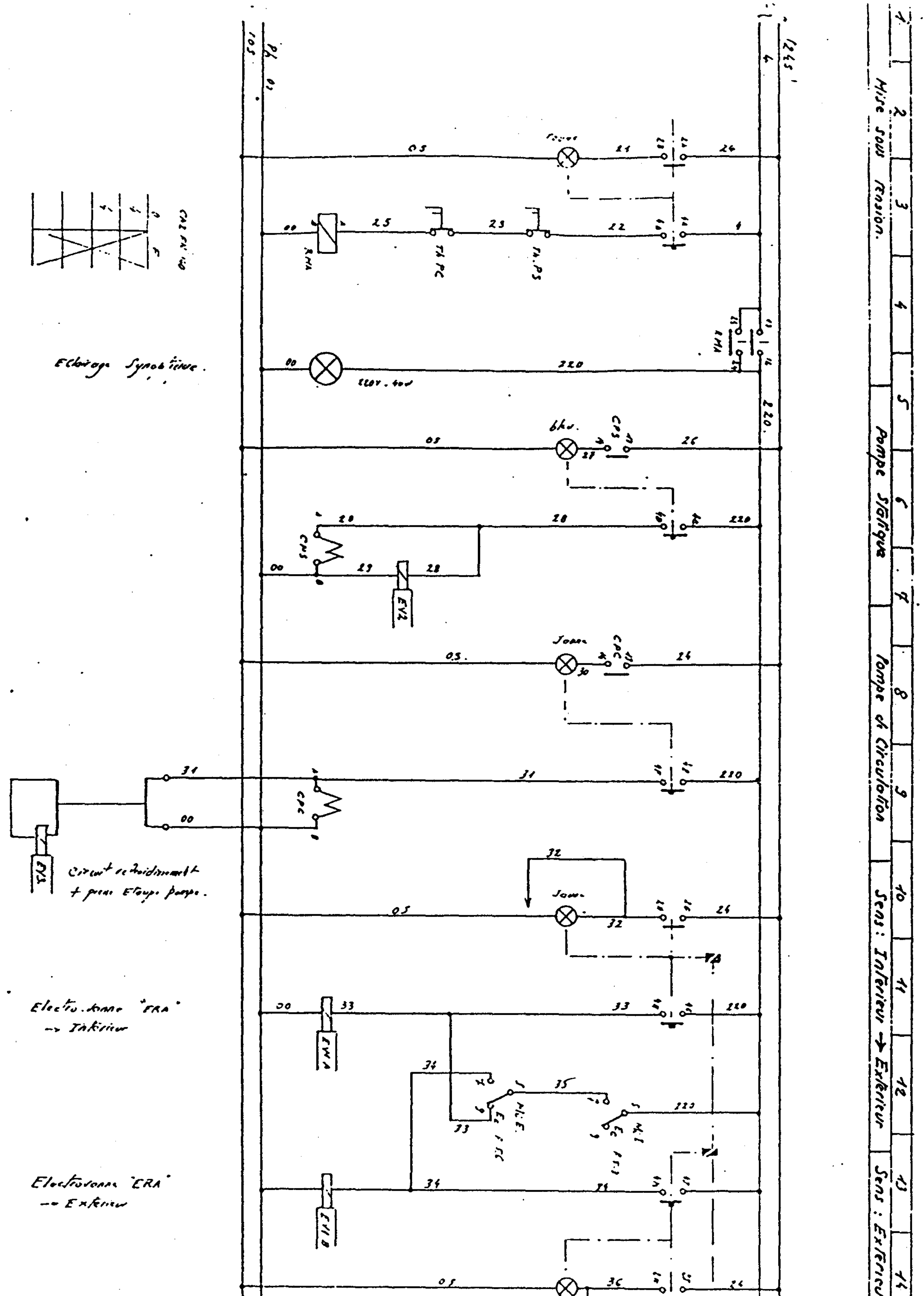
	0	1	2	3	4	5	6	7	8	9
0	x	x	x	x	x	x	x	x	x	
1		•		x	x	x	x	x	x	x
2		x	x	x	x	x	x	x	x	x
3	x	x	x	x	x	x	x	x	x	x
4	x	x	x	x	x	x	x	x	x	
5	–	:	x	x	x	x	x	x		
6										
7										
8										
9										

SEPROTEC Engineering	Client C.N.S.A.I.T.	Ensemble	Sous ensemble
N° 1 8 5 8 Folio 1	Fourniture Autoclave . Labo.	Schema d'automatisme	Sommaire
Ind.			



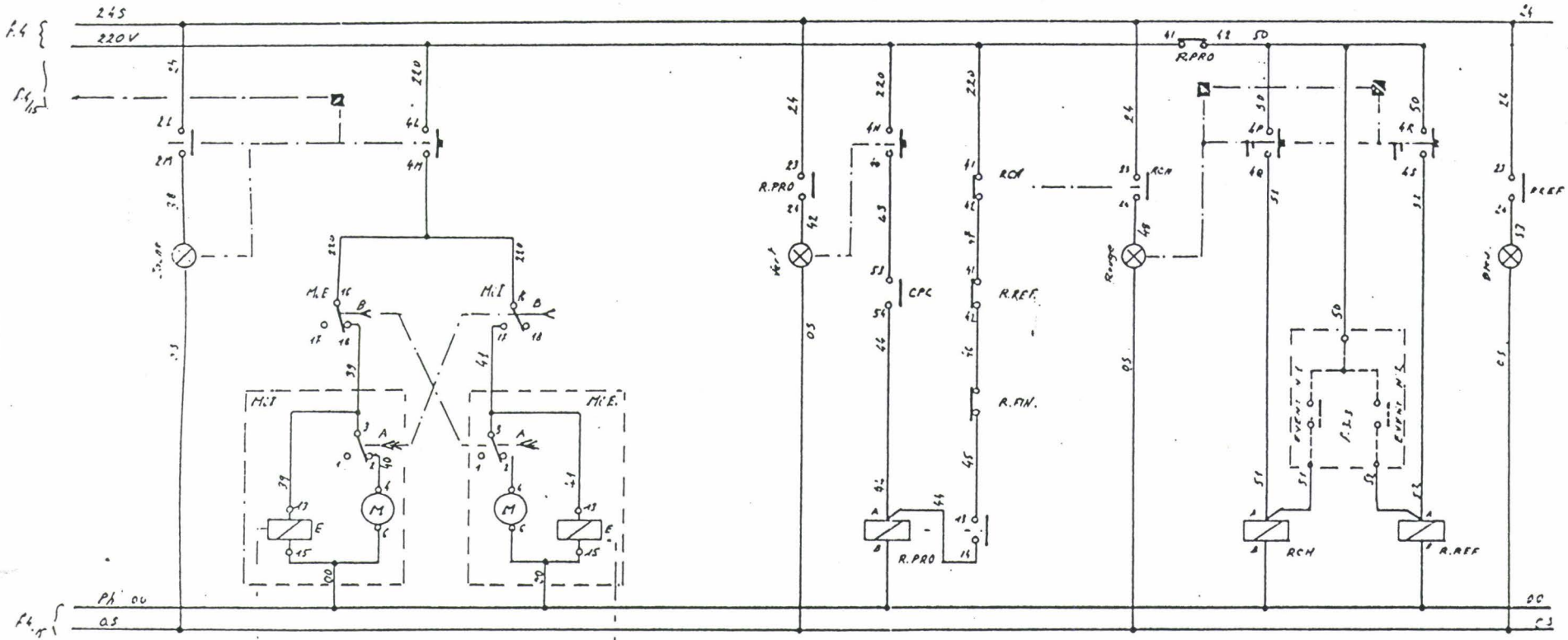
ALPROLEC Engineering	Client E.N.S.A.I.T.	Ensemble	Sous ensemble
1858 Folio 2	Fourniture Autoclave . Labo.	Schéma d'automatisme.	Distribution d'énergie . FL.

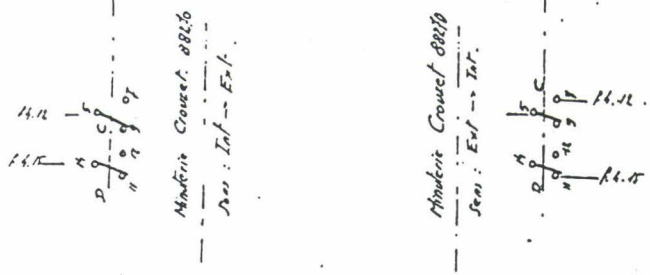

Programmateur "Data Track"
Regulateur de température Thermac 640 L.
Correction Conflow. 4-20mA 92.76.
sortie
commande vanne proportionnelle
Enregistreur de température Miniscript. Metrocem
entrée
mesure de la température
Sonde PT 100 DIN.
220Vac.
F2
Sous ensemble
Ensemble
Client E.N.S.A.I.T.
Fourniture Asservissement Labo.
Folio 3
Engineering

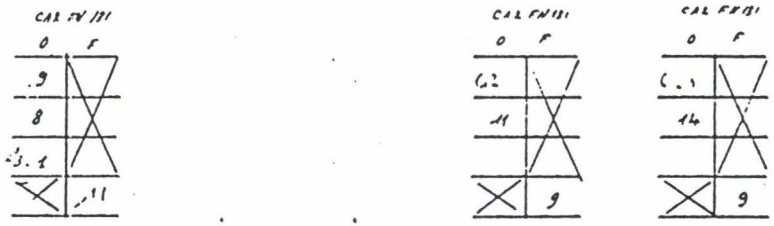


SEPROTEC Engineering	Client E.N.S.A.I.T.	Ensemble	Sous ensemble
N° 1856 Folio 4	Fourniture Ateliers Labo.	Schéma d'automatisme	Pompes. Sens de circulation

1	2	3	4	5	6	7	8	9	10	11	12	13	14
	Minuteries.					Marche Programmation				Chauffage		Refroidissement.	







SEPROTEC
Engineering

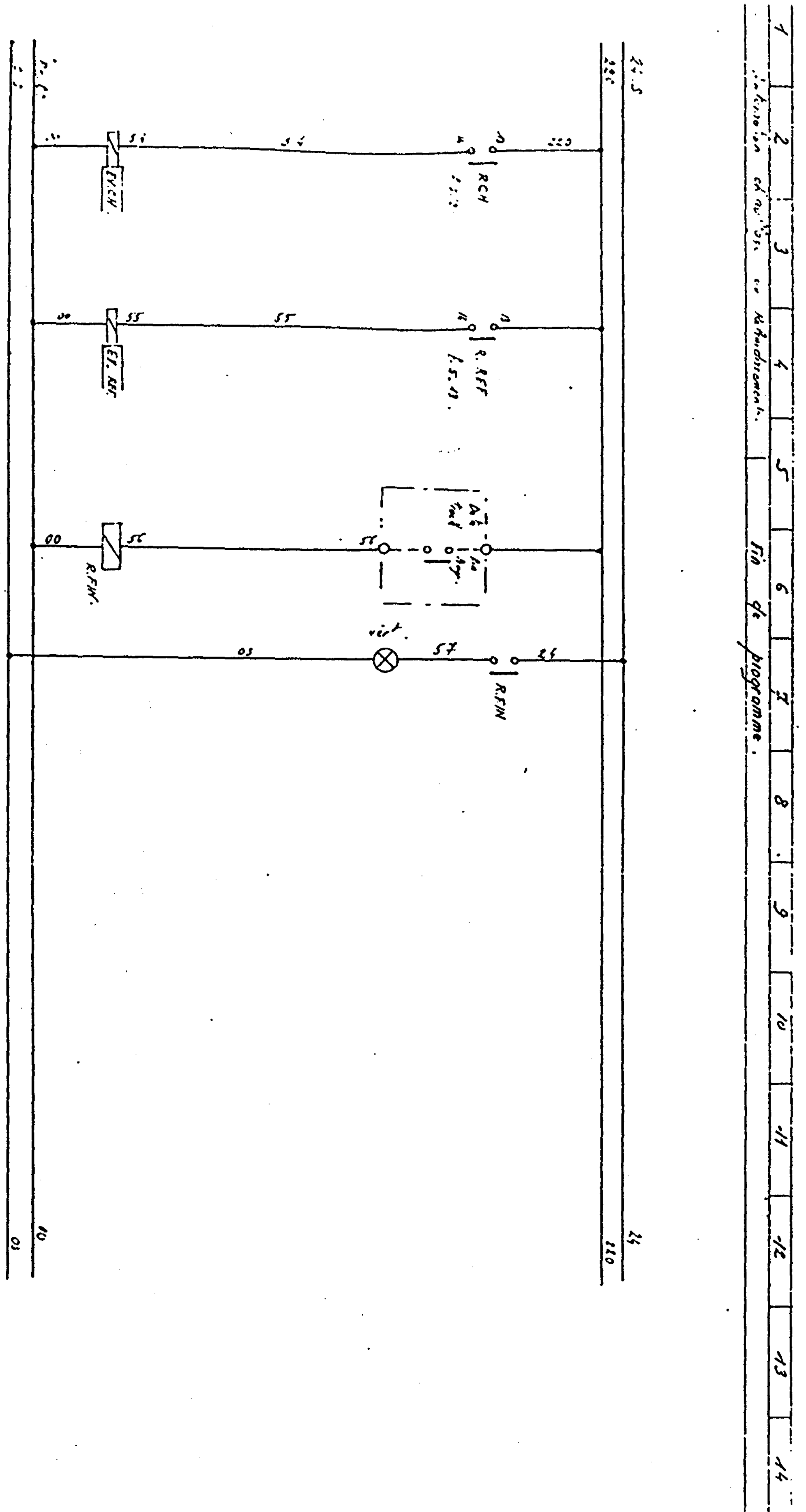
N°	1	8	5	8	folio 5

Ind.

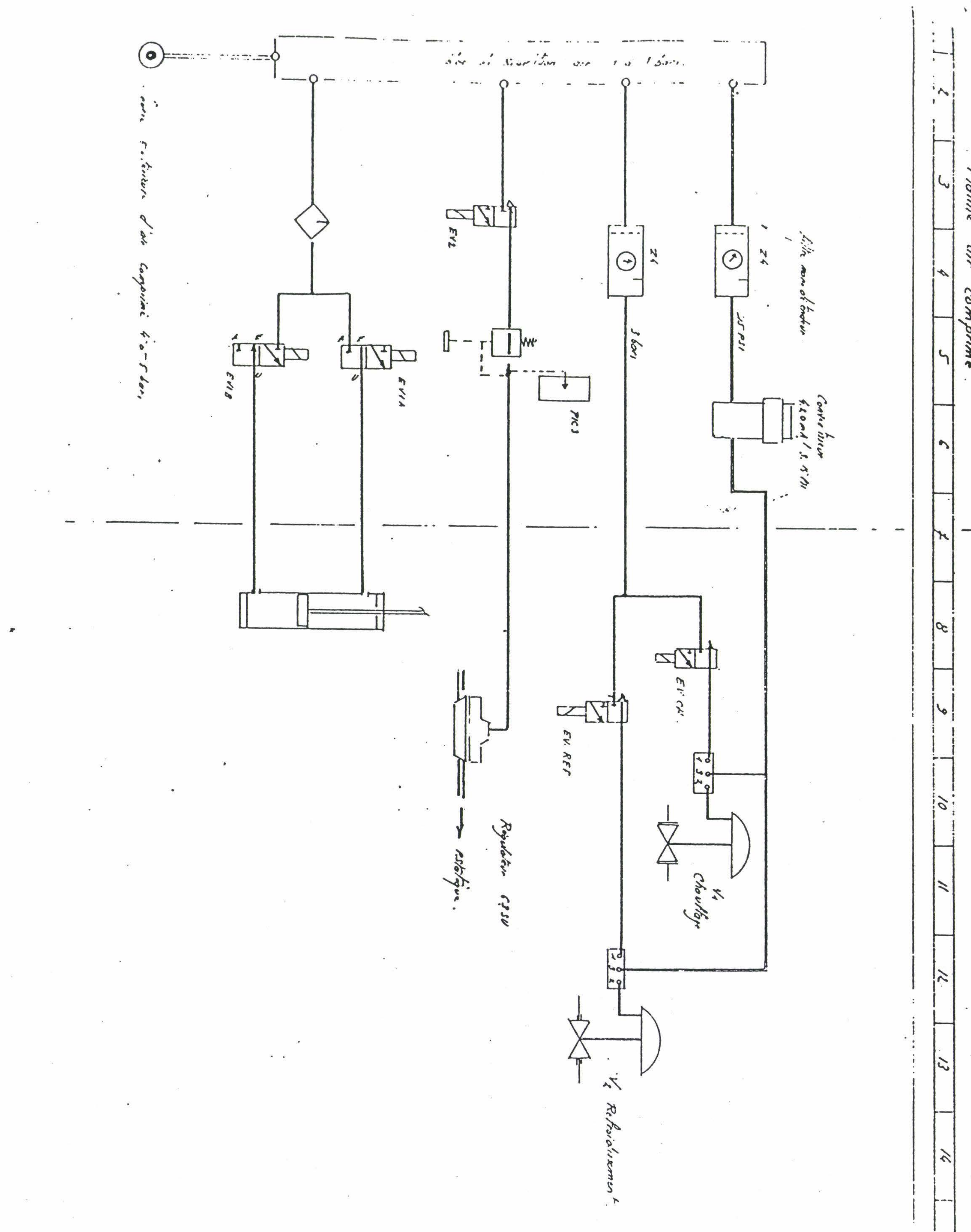
Client ENSAIT
Fourniture Armoire Locks.

Ensemble
Schéma d'automatisme.

Sous ensemble
Minuteries. Chauffage et Ventil.



SELPROTEC Engineering N° 1 8 5 0 — Folio 6	Client E.N.S.A.I.T Fourniture Autoclave Labo.	Ensemble Schéma d'automatisme.
		Sous ensemble

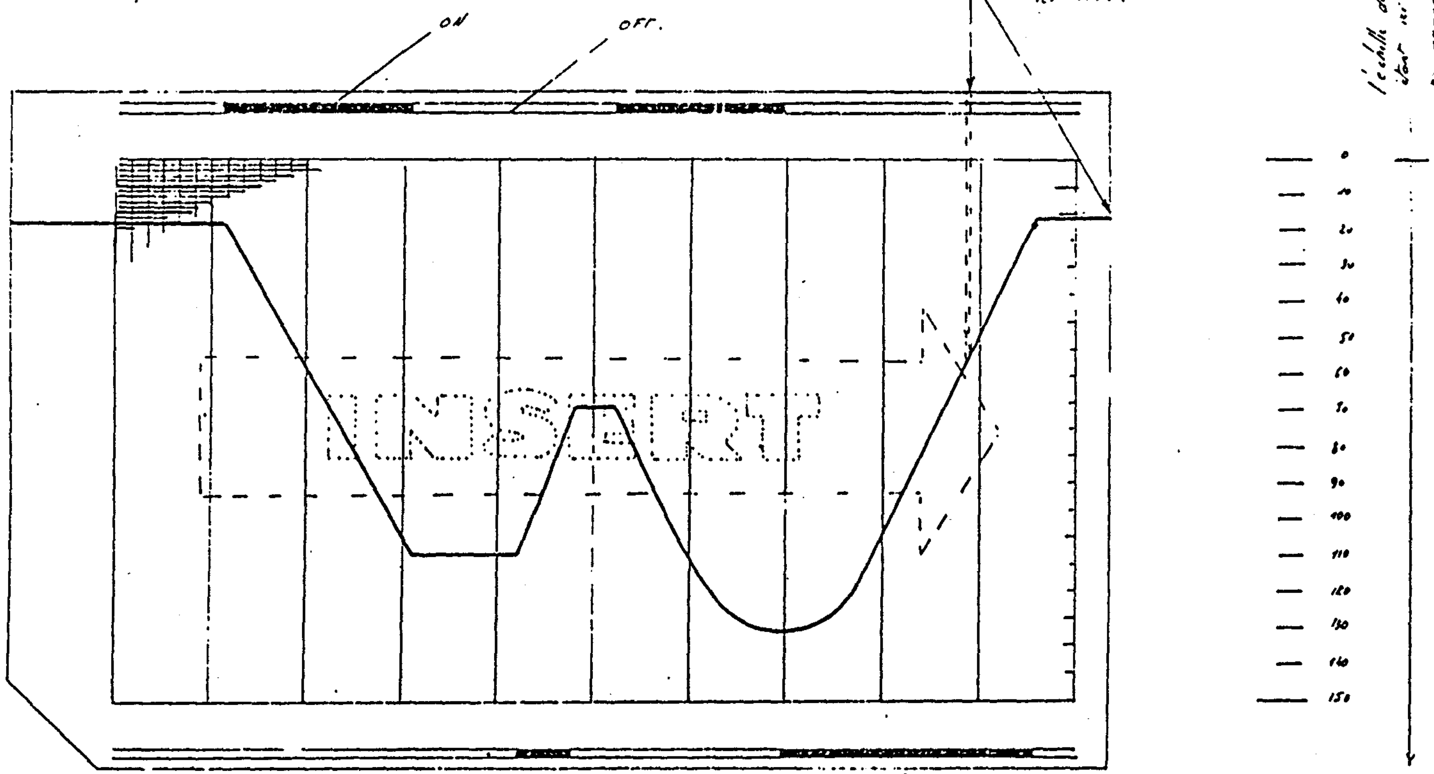


SEPROTEC Engineering	Client E.N.S.A.I.T	Ensemble	Sous ensemble
N° 1 8 5 8 — folio [illegible]	Fourniture Autoclave Labo.	Schéma d'automatisme	Câblage pneumatique
Ind.			

Pist. Event 1 → Utilisé pour activation du chauffage.

⚠ → Il faut absolument que la face de la carte n'ait pas été abîmée, séparée la surface en 2 zones électriquement distinctes.

Vérifier par transparence qu'il en soit bien ainsi



l'échelle de température utilisée étant ici de 150 °C, il est recommandé de se faire une règle graduée avec 150 divisions afin de tracer directement la courbe de température

Pist. Event 2 → Utilisé pour activation du refroidissement.

Pour plus ample information, consulter la Notice : DATA-TRAK model 5310.

SEPROTEC Engineering	Client E.N.S.A.I.T.	Ensemble	Sous ensemble
N° 1858 Folio 8	Fourniture Autoclave Labo.	Schéma d'automatisme	Carte programme
Ind.			

XMATH : LA FONCTION OPTIMIZE

2 Nonlinear Programming

The `optimize` function solves the general nonlinear programming problem:

$$\begin{aligned} \underset{p}{\text{minimize}} \quad & F(p) \\ \text{subject to} \quad & G(p) = 0 \\ & h_l \le H(p) \le h_u \\ & p_l \le p \le p_u \end{aligned}$$

where:

p	The $n \times 1$ vector of optimization parameters; must be real.
F(p)	The scalar valued cost (objective) function.
G(p)	The vector valued equality constraint function.
H(p)	The vector valued inequality constraint function.
h_l and h_u	The lower and upper limits for the inequality function.
p_l and p_u	The lower and upper limits for the optimization parameters.

In general, F(·), G(·), and H(·) are *any* nonlinear functions that may be specified and computed using Xmath and/or SystemBuild. The constraint equations and parameter bounds need not be specified. This flexibility results in a wide variety of problem solving capabilities, which are illustrated in the examples in this chapter.

2.1 Optimize

```
[P,Jh,L,H,IC,Ph]=optimize(P0,{Pmin,Pmax,ICmin,ICmax,L0,
                              H0,IC0,rho,majit,minit,delta,tol})
```

Conditions and parameters for operation of the function that invokes the `optimize` function are set up in two places:

1. The user furnishes the values of the cost and constraints via a MathScript function (MSF) named `COST`, which is executed by `optimize` whenever it needs the appropriate functions for a set of candidate parameters evaluated. `optimize` computations include the cost function F(*p*), an optional equality constraint function G(*p*), and an optional inequality constraint function H(*p*).

2. All other parameters are specified as parameters of the `optimize` function call, including:
   - Initial values and bounds for the optimization parameters.
   - Bounds for the inequality constraint equations.
   - A penalty parameter controlling the search for a feasible solution.
   - Maximum numbers of major and minor iterations.
   - Numerical gradient perturbation parameters.
   - A tolerance parameter on optimality and feasibility.

See page 60 or the Online Help for a detailed description of each `optimize` input, keyword, and output.

Each optional parameter is set via a keyword. Although some parameters must be set by the user for a problem to be solved efficiently; see the rest of this section for details.

2.1.1 Running Optimize

As shown in Figure 1, running `optimize` in Xmath is a straight-forward process:

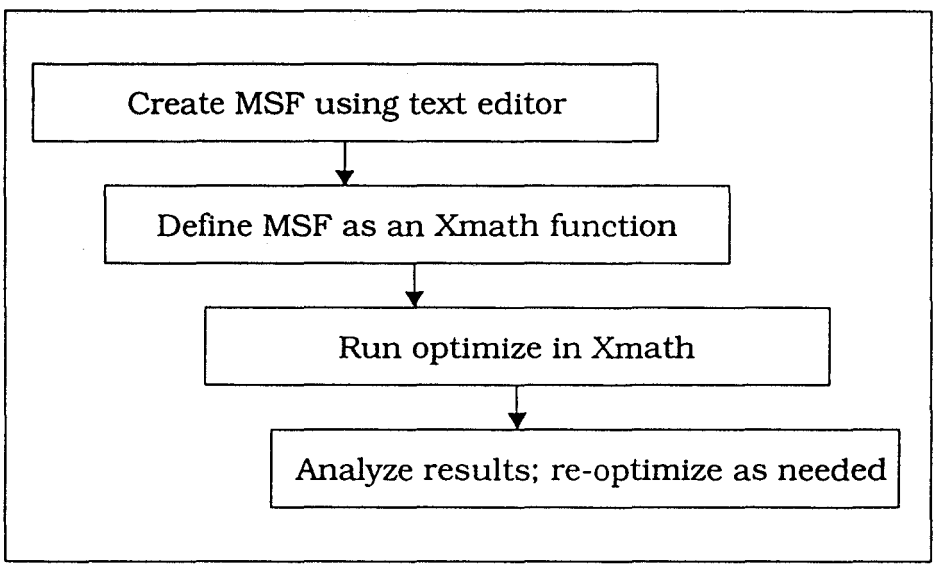


Figure 1 Running `optimize`

1. Create a MathScript function (MSF) named `cost.msf` using any text editor. Your function computes the cost that is to be minimized and also calculates the constraint functions.

2. Use the Xmath `DEFINE` command to make your MSF available to Xmath.

3. When the `cost.msf` is ready and debugged, specify input variables and execute the `optimize` function. `optimize` calls the `cost.msf` as many times as its operations require.

4. As the algorithm executes, you may monitor its progress by displaying intermediate values and/or creating plots in the cost function. Upon completion, all results are returned to Xmath's command level for quick access to the Xmath system analysis, post processing, and simulation capabilities. To save any of the variables computed by the cost function, use the Xmath SAVE command, or select File→Save from the commands window menu bar.

   The key to the Optimization Module's flexibility and breadth of problem solving ability lies in its close coupling MathScript and graphical modeling in SystemBuild. The objective function and any constraint functions are specified in `cost.msf`. This means that any dynamic or static system performance measure can be defined quickly and easily. Waveform math, matrix algebra, frequency response analysis and time simulations are just some of the many operations that can be used in defining problems – all in a few commands.

   The following is a general template for `cost.msf`.

```
# cost function syntax:

function [out]=cost(p,it)

#{ p is the list of candidate parameters cost.msf
   uses to evaluate the cost and constraint
   function values.
   it (iteration) informs the cost function that
   optimize() has completed a major or minor
   iteration:
      it<0  a minor iteration has completed
            a gradient evaluation (occurs
            once per minor cycle)
      it=0  a gradient or search evaluation
      it>0  major iteration completed
}#
# insert cost to be minimized (scalar):
```

```
mincost =

#{compute value(s) of equality function(s) and
  place them in the variable equal (a column
  vector). Skip this step if equality constraints
  are unnecessary:
}#

equal=

#{compute value(s) of inequality function(s) and
  place them in the variable inequal (a column
  vector). If inequality constraints are specified,
  the number of values must match the number of
  upper and lower inequality constraints specified
  in the optimize call. Skip this step if inequality
  constraints are unnecessary:
}#

inequal=

#{put results in out, omitting any variables that
  were not computed above}#

out=[mincost;equal;inequal];

#{display or store any intermediate results here
  using the variable it to determine points of
  major and minor iterations.}#

endfunction
```

---

cost accepts the column vector of optimization parameters p and the scalar variable it. It returns the argument out. In the body of cost, the user specifies how p will be used to compute the cost function and (optionally) any equality and/or inequality constraints. Any command or function that can be issued at the Xmath prompt can be used to compute these performance measures. optimize calculates the number of constraints by finding the size of out (the variable returned by the cost function) and determines the number of inequality

constraints by checking `ICO`. If *m* equality constraints and *n* inequality constraints are specified, the out vector will return (m + n + 1) elements ordered as follows:

Position	Description	Optional (y/n)
1	Objective (COST)	n
2: *m* + 1	Equality Constraints	y
m + 2: *m* + *n* + 1	Inequality Constraints	y

The `iter` input parameter informs `cost` of the completion of a major or minor iteration. `iter` will signal any *one* of three possible conditions in the optimization process after that event has occurred.

Condition	Situation
iter < 0	The gradient evaluation
iter = 0	Gradient or search evaluation
iter > 0	Major iteration completed

`iter` can be used to customize the run time information displayed on the screen (or in the log file, if the optimization job is run in batch mode). Data can be analyzed, displayed, plotted or saved to the disk at minor and/or major iterations. The `optimize` process is illustrated in the examples given in Section 2.4 on page 19. Both unconstrained and constrained nonlinear optimization problems are set up, implemented and solved. Section 2.3 on page 16 details how the `optimize` algorithm works and Section 2.2 on page 9 discusses some practical approaches that aid in using `optimize` effectively and efficiently. Detailed demo files supporting Section 2.4 are included with the software so you can follow along with the text and quickly come up to speed with optimization.

2.2 Practical Considerations

The `optimize` algorithm has been designed to handle a broad range of engineering problems with as few restrictions on problem definition as possible. The following subsections discuss several points to consider when using `optimize`. In most situations, following these guidelines will mean faster and more efficient convergence.

2.2.1 Finding a Feasible Solution

A feasible candidate solution should at least meet all the constraint criteria (even if it is not optimal or near optimal). If `optimize` is having difficulties converging to a feasible minimum for a constrained optimization problem, try altering the original problem to help `optimize` solve it more efficiently. One way to do this is to modify `cost` so that it keeps the cost constant whenever the error in the constraint equation (i.e., the difference between the current candidate parameter and the feasible range) is large. This means that the true cost will be computed only when `optimize` has found a feasible (or near feasible) solution. Forcing `optimize` to concentrate on finding a feasible region before searching for an optimal solution will get the best results from the algorithm.

2.2.2 Specifying the Penalty Parameter

The penalty parameter ρ is used to control the distance that the optimization search is allowed to travel outside the feasible range to seek an optimal solution. A value of this parameter greater than 1 forces the search to stay close to the known feasible range; a smaller value of ρ allows a broader range of searching. The default value, 1, allows an intermediate range of search.

Another way of understanding the role of the penalty parameter is as a weighting factor. The higher the value of ρ, the more emphasis will be put on bringing (or keeping) the parameters inside the feasible region. Be cautious when using large values of ρ, since a large ρ might make the prob-

lem ill-conditioned. We suggest not exceeding $\rho = 1000$. In most situations, the default value of ρ will result in good performance of the nonlinear optimization routine. In other cases, it may be necessary to specify a good initial guess for the penalty parameter.

For smooth problems, several guidelines may help you find a good initial guess. If a good solution to a similar problem is known, the value of the penalty parameter from the successful optimization will provide a good first guess for the current problem. If you have absolutely no idea about how to find an initial penalty parameter, try $\rho = 1$. In most cases, difficulties associated with a poor initial choice for the penalty parameter can be avoided by specifying reasonable bounds on the parameters.

For non-smooth problems, the difficulties associated with selecting a good initial penalty parameter are not as severe. In this case, a good strategy is to choose a large value for the initial parameter. This will help insure that the subproblem will have the desired local minimum. If the penalty parameter from a successful optimization of a similar problem is available, start the optimization with a slightly smaller value of the successful penalty parameter. If no help is available, try 10. As always, try to place reasonable bounds on the variables to reduce the problem's sensitivity to the penalty parameter.

In general, there is no systematic way of choosing the *best* value for the penalty parameter. Some problems can be extremely sensitive to this parameter, thus making subproblems very poorly conditioned whenever the penalty parameter is too small or too large. It is important to remember what effect this parameter can have, and that it may only be possible to find a good value through trial and error. For more details refer to Practical Optimization,† under "*Penalty Functions*" and "*Penalty Parameters*".

† Gill, P.E., Murray, W. and Wright, M.H.; Practical Optimization, Academic Press, 1981.

2.2.3 Evaluating Results

After the algorithm has completed the optimization, you should be careful to check that the optimal value meets any additional criteria implied by common sense. For example, the final result should be a true minimum; at least, it should be considerably smaller than the initial result.

2.2.4 Reducing Constraints to Improve Performance and Efficiency

Constraints add a great deal of complexity to the general optimization problem. It is always advantageous for the user to simplify, reduce or completely eliminate constraints. With general formulation of the `optimize` function, several types of constraints are possible. Equality constraints and inequality constraints are the most general classification. These constraints can be linear or nonlinear in the optimization parameters. Optimization with inequality constraints is the most difficult problem to solve, and nonlinear constraints pose greater difficulties than linear constraints. With careful specification, constraints can be simplified and the optimization will be performed more efficiently and with fewer problems.

Always check the constraint equation and first eliminate any redundant or unnecessary constraints. Next, try to replace or simplify any constraints, or replace the constraints with bounds on the parameters. In some cases, a transformation of variables will be helpful. The following problem,

$$\begin{aligned} &\text{minimize} && (x_1 + 2)^2 + (x_2 + 1)^2 \\ &\text{subject to} && x_1 \le 0 \end{aligned}$$

can be converted into an unconstrained minimization problem by letting:

$$(x_1 = z_1^2) \text{ and } (x_2 + 1)^2$$

The problem then becomes to minimize:

$$(z_1^2+2)^2+(z_2+1)^2$$

If this is not possible, then try to turn nonlinear constraints into linear constraints. Finally, try to combine and simplify constraint equations to keep the same limitation on the problem using a smaller number of simpler equations. The effort of simplifying the constraints beforehand is a wise investment towards making the optimization process much more efficient and robust, but often it's difficult to decide when or how to make the modifications.

Transformations can introduce very undesirable properties in the optimization problem (discontinuities, periodicity in the optimization parameters, singularities, poor scaling, a greater number of minima, etc.). For this reason, simplify the constraints only after full consideration of the pitfalls which may result. When in doubt, do not simplify!

Eliminating redundant parameters is as important as eliminating redundant constraints. The following transfer function is an example:

$$h(s) = \frac{as+b}{cs^2+ds+e}$$

This expression has five parameters: a, b, c, d, and e. However, the same transfer function can be expressed as

$$h(s) = \frac{k_1 s+k_2}{s^2+k_3 s+k_4}$$

where $k_1 = \frac{a}{c}$, $k_2 = \frac{b}{c}$, $k_3 = \frac{d}{c}$, and $k_4 = \frac{e}{c}$. This reduces the number of parameters from five to four.

2.2.5 Avoiding Discontinuities

Like most optimization algorithms, convergence of the `optimize` function can only be guaranteed when the objective function and any constraint functions are sufficiently smooth. Discontinuities should be avoided whenever possible. Discontinuities can arise from seemingly insignificant sources in problem formulation and definition. Be careful to avoid these common pitfalls:

Table Lookup Linear interpolation table lookup functions are a common, but avoidable, source of discontinuities in the formulation of an optimization problem. Table lookups are continuous in the function along all interior points, but the first derivative of the function will be piecewise constant (with linear interpolation). Since table lookups are often used to approximate continuous data, there is little or no loss of accuracy involved with providing a smooth approximation (e.g., by using the `spline` function in Xmath to smooth a linear interpolation).

Piecewise Constant Functions Piecewise constant functions such as non-interpolated table lookup, sampled data time histories, etc., can pose significant problems in optimization. If the resolution of a computation is coarse, numerical gradients will not be accurate. This results in slow convergence or poor algorithm performance. Two courses of corrective action may improve the performance of the algorithm:

- interpolation (preferably with a continuously differentiable algorithm) and/or,
- finer resolution (more samples) in the function.

Function Switching Avoid situations where the objective function or any constraint functions contain computations that change in structure (switching from one formula to another) as a function of the parameter values. When such a function must be included, make sure that the function outputs and first derivatives match at all switching conditions.

Iterative Algorithms If you use an iterative algorithm to compute the objective and/or constraint functions and you have specified tolerances considerably larger than the machine constants, nontrivial discontinuities can occur. A common example occurs when a variable step algorithm is used to evaluate an integral; successive iterations of the optimization algorithm may result in varying local accuracy along the output trajectory. The function and/or its gradients will not be smooth functions of the parameters. In certain cases, the error can be substantial. This error can be avoided by using fixed step methods or a very small local error tolerance for the variable step method. Another option is to increase the size of the perturbation used in gradient step-size (whenever this retains the integrity of the gradient computation). In most cases, it will be difficult to determine the proper balance between integration algorithm accuracy and the gradient perturbation, so try a fixed step integration algorithm.

We advise that you define step size for numerical derivatives so changes due to perturbing the parameters are greater than the errors in the integration, else you will end up with derivative of integration noise with respect to your parameters. For example, if the numerical integration is good to +/-.01 and you expected results to change by .001 when you perturb the parameter, you can't tell the difference. You can either make the integration error specification tighter (a precision of .0001) or boost the perturbation so the expected change is .1.

When using SystemBuild, avoid using integration blocks or saturation blocks as limiters (if saturation is reached, `optimize` does not get anything useful). Rather, make the signals sim outputs so that the optimization algorithm can use constraints to do the limiting.

These and other important considerations are discussed in greater detail in Practical Optimization.[†]

† Gill, P.E., Murray, W. and Wright, M.H.; Practical Optimization, Academic Press, 1981

2.2.6 Controlling the Numbers of Major and Minor Iterations

In an attempt to achieve convergence, the `optimize` function divides the general nonlinear programming problem into a series of linearly constrained nonlinear programming steps to approximate the actual nonlinear problem; these are the major iterations of the function. The major iteration linearizes the constraints and tries to find a feasible solution for the linearized problem. It then passes it to the minor iteration(s).

The minor iterations use quadratic methods to find a solution within the linear constraints. When the minor iterations are complete, or a solution is found `optimize` returns to the major iteration. These ideas are illustrated in Figure 2 on page 15.

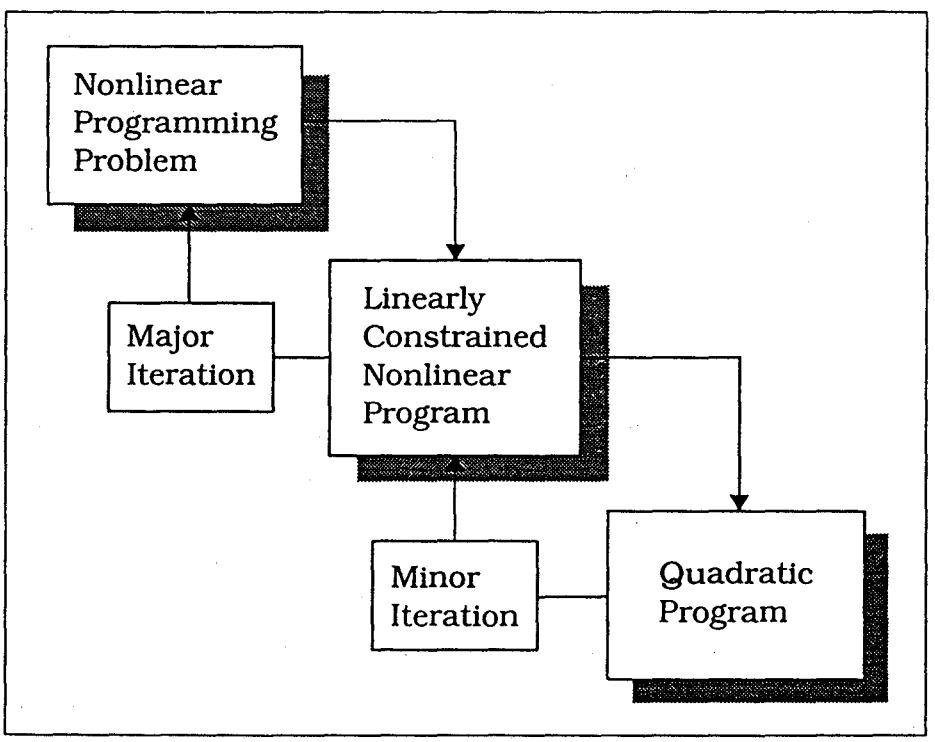


Figure 2 Major and Minor Iterations

Many optimization problems are sensitive to the number of either major or minor iterations, but it is difficult to give general rules for assigning these variables, because they are intensely problem-dependent. Ten is the default setting for

the number of both major and minor iterations. This is intentionally generous; often performance can be enhanced with no loss of accuracy by using smaller numbers for either or both parameters. It is easy to study this effect, because the calls to the COST MSF contain the indication that this call is for a major or a minor iteration, and one can see if, for example, more minor iterations give an evidently better local solution, or if they are just a waste of time.

For some problems, the constraints influence the iterations greatly. This is true if the solution from the minor iterations falls beyond the true nonlinear constraints. If the solution to the constraints is twice as bad as the solution from the previous major iteration, ρ will be increased. If it is safely within the tolerances ρ will be decreased.

The higher ρ is, the more important the constraints are. If ρ is small, it is more tolerant. If your problem is poorly behaved when the constraints are violated, you want to push the solution towards the constraints by increasing ρ and tightening the constraints.

The constraint used in the minor iteration quadratic program is only a linear approximation. This means some inaccuracy may result because of nonlinearities in the actual constraints. If your constraints are highly nonlinear, you may want to reduce the number of minor iterations so that the linear approximation will be updated more often.

2.3 The Optimize Algorithm

When given the following problem `optimize` converts

$$\begin{aligned} \underset{p}{\text{minimize}} \quad & F(p) \\ \text{subject to} \quad & G(p) = 0 \\ & h_l \le H(p) \le h_u \\ & p_l \le p \le p_u \end{aligned}$$

into

Equation 1

$$\begin{aligned} \underset{\hat{p}}{\text{minimize}} \quad & F(\hat{p}) \\ \text{subject to} \quad & \hat{G}(\hat{p}) = 0 \\ & \hat{p}_l \le \hat{p} \le \hat{p}_u \end{aligned}$$

where $\hat{G}$ is a combination of G and H:

$$\hat{G} = \begin{pmatrix} H(p) - s \\ G(p) \end{pmatrix} \qquad \text{Let} \qquad \hat{p} = \begin{pmatrix} p \\ s \end{pmatrix}$$

$$\hat{p}_u = \begin{pmatrix} p_u \\ h_u \end{pmatrix}$$

$$\hat{p}_l = \begin{pmatrix} p_l \\ h_l \end{pmatrix}$$

`optimize` solves the problem by first constructing a linearly-constrained optimization problem with an Augmented Lagrangian objective function (Equation 2 on page 17),

Equation 2

$$\begin{aligned} \underset{\hat{p}}{\text{minimize}} \quad & F(\hat{p}) - \hat{y}_k \hat{G}(\hat{p}) + \frac{\rho}{2} \hat{G}(\hat{p})^T \hat{G}(\hat{p}) \equiv \Phi(\hat{p}) \\ \text{subject to} \quad & J_k(\hat{p} - \hat{p}_k) = -\hat{G}(\hat{p}_k) \\ & \hat{p}_l \le \hat{p} \le \hat{p}_u \end{aligned}$$

where J is a numerical approximation to the Jacobian:

Equation 3

$$J_k = \left. \frac{\partial \hat{G}(\hat{p})}{\partial \hat{p}} \right| \hat{p} = \hat{p}_k$$

and $\hat{y}_k$ is the vector of Lagrange multipliers at step k $\hat{y}_0 = 0$ by default). This problem is solved with an iterative algorithm

for the variables $\hat{p}_{k+1}$ and $\hat{y}_{k+1}$. For the new problem, the first step involves checking to see if $\hat{p}_k$ satisfies the linear constraints.

$$J(\hat{p}-\hat{p}_k) = -\hat{G}(\hat{p}_k)$$
$$\hat{p}_l \le \hat{p} \le \hat{p}_u$$

If $\hat{p}_k$ is not a feasible solution, Equation 3 is solved to generate a valid initial condition for subsequent iterations.

Next, sequential quadratic programming (SQP) is implemented to solve Equation 2. We calculate the gradient vector $\hat{g}_k$ and update the Hessian matrix (Equation 4 on page 18) of the Augmented Lagrangian objective of Equation 2 to obtain the second-order approximation to the objective function $\Phi(\hat{p})$ in the quadratic programming (QP) problem in Equation 5 on page 18.

The Hessian matrix is updated as follows (Broyden-Fletcher-Goldfarb-Shanno):

Where,

$$\Delta p = \hat{p}_{k+1} - \hat{p}_k$$
$$\Delta g = \hat{g}_{k+1} - \hat{g}_k$$

then,

Equation 4

$$H = H - \frac{1}{\Delta p^T H \Delta p} H \Delta p \Delta p^T H + \frac{1}{\Delta g^T \Delta p} \Delta g \Delta g^T$$

Equation 5

$$\underset{\hat{p}}{\text{minimize}} \quad \frac{1}{2}(\hat{p}-\hat{p}_k)^T H_k (\hat{p}-\hat{p}_k) + \hat{g}_k(\hat{p}-\hat{p}_k)$$
$$\text{subject to } J_k(\hat{p}-\hat{p}_k) = -\hat{G}(\hat{p}_k)$$
$$\hat{p}_l \le \hat{p} \le \hat{p}_u$$

where H_k is a Hessian for $\Phi(\hat{p}_k)$ and $(\hat{g}_k)$ is a gradient for the Augmented Lagrangian objective function $\Phi(\hat{p}_k)$.

`optimize` uses the interior trust region method† to reach an approximate solution for QP (Equation 5). A line search technique provides a step size for Equation 2, and is used to generate the minimal solution $\hat{p}$.

If the optimal QP solution $\hat{p}$ is both feasible and optimal for Equation 2, `optimize` returns to the beginning; otherwise, it updates the Hessian matrix (Equation 4) for the subproblem (Equation 5) and solves the QP problem. This repeats until the optimal local solution is found or until the maximum number of minor iterations is computed.

The final solution $\hat{p}$ and multiplier $\hat{y}$ that result from the QP process are $\hat{p}_{k+1}$ and $\hat{y}_{k+1}$. These values are returned to solve for the next subproblem (Equation 2). When Equation 2 converges to a local minimum $\hat{p}$ for Equation 1, the algorithm is complete.

2.4 Application Examples

This section illustrates a variety of applications for the `optimize` function. The first example shows how to `optimize` the volume of a box; the second solves a trajectory optimization problem; and the last `optimizes` the response of a third order nonlinear system. This group of applications demonstrates the power and flexibility of the Optimization Module and demonstrates the approaches and procedures you will need to begin solving your own optimization problems. Each application has four steps:

Problem Definition Define the system and performance objectives or measures to be met.

† Y. Y. Ye, Ph.D. Dissertation, Dept. of Engineering-Economic Systems, Stanford University, (1987).]

Formulation The system cost and constraint equations are specified using Xmath and SystemBuild commands and functions in the `COST` MSF.

Optimization Use the `optimize` function to compute and solve optimization problems.

Analysis The preliminary solution is examined. If the algorithm does not converge in the specified number of iterations, the `optimize` function can be restarted. The algorithm can be run with different parameters or from another initial condition to test the assumptions about a local minimum.

To get the most out of these application examples you should be a proficient Xmath user able to create MSFs. Consult the online helps and the *Xmath Basics* if you do not understand MSFs.

2.4.1 Box Design

The objective of this example is to find the height, width and length that give a box a total volume of $100m^3$ and minimize the amount (surface area) of cardboard required. The situation is illustrated in Figure 3 on page 21. The only restrictions are that each box side must be greater than 1m and less than 10m in length.

Formulation

As shown in Figure 3, the dimensions of the box are p_1, p_2, and p_3. The top and bottom of the box have double flaps so that the surface area of the top and bottom is $4p_1p_2$. The total surface area of cardboard required is

$$J_{(p)} = 4p_1p_2 + 2p_2p_3 + 2p_1p_3$$

This is the quantity to be minimized and it will be returned as the first value from the `cost` MSF. The second argument to be

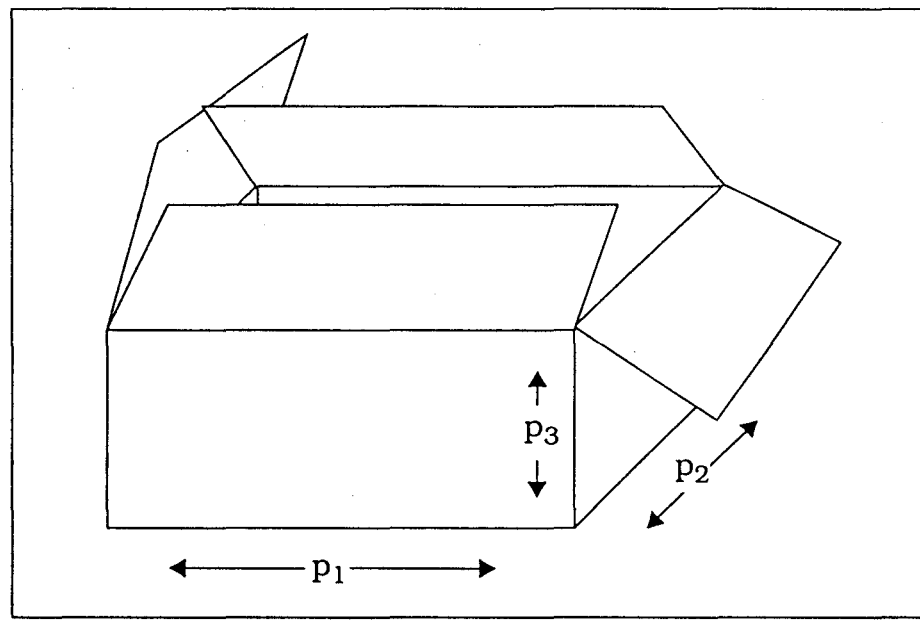


Figure 3 Box Optimization

returned is the difference between the actual and desired volume:

$$G_{(p)} = p_1 p_2 p_3 - 100$$

This argument defines the equality constraint that requires the volume of the box to be exactly $100m^3$. These equations are coded in a MathScript file named `cost.msf`. In the `cost` file, $J_{(p)}$ = `OUT(1)` and $G(p)$ = `OUT(2)`. In a directory of your choice, create `cost.msf` as shown below.

```
#Cost function for the box optimization problem.
function out=cost(p,iter)
   out=[
    4*p(1)*p(2)+2*p(2)*p(3)+2*p(1)*p(3)  #  surface
    p(1)*p(2)*p(3)-100];                 # vol -100
endFunction
```

Now you are ready to run the optimization in Xmath.

Optimization

Running optimize on this example is very simple.

1 Enter Xmath and define the function cost.msf:

```
define cost
```

2 Define the parameter bounds for each dimension of the box:

```
pmin=ones(3,1);pmax=10*pmin;
```

3 Make *P0* equal to the average of the upper and lower bounds, thus:

```
p0=(pmax+pmin)/2;
```

4 Execute the optimize function:

```
[p,jh,l]= opti(p0,{pmin=pmin,pmax=pmax,
   rho=1,majit=5,minit=10,delta=1e-6,tol=1e-6})
```

optimize computes successive estimates of the parameter dimensions and the error in the volume of the box. While running, optimize will display the current major iteration and the current minor iteration. The current value of the cost function, (J) will also be displayed. These are output to the screen with each iteration until the maximum number of major iterations is reached or optimize is completed.

In this example optimize returns the final parameter estimate p and the history of cost values (surface areas) jh after 5 major iterations are computed. l is the Lagrange multiplier.

```
Beginning minor iteration 1
Beginning minor iteration 2
Beginning minor iteration 3
Updated parameters:   3.51798   3.51798   7.26982
Beginning major iteration 2, J=151.805
Beginning minor iteration 1
Beginning minor iteration 2
Beginning minor iteration 3
Beginning minor iteration 4
Beginning minor iteration 5
Beginning minor iteration 6
Updated parameters:   3.75321   3.75319   7.1079
Beginning major iteration 3, J=163.055
Beginning minor iteration 1
```

```
Beginning minor iteration 2
Updated parameters:   3.74731   3.74728   7.12137
Beginning major iteration 4, J=162.912
Beginning minor iteration 1
Updated parameters:   3.74724   3.74721   7.12165
Beginning major iteration 5, J=162.912
Beginning minor iteration 1
Updated parameters:   3.74723   3.74721   7.12166
Completed in 5 iterations

p (a column vector) =

  3.74723
  3.74721
  7.12166

jh (a row vector) =   242    151.805    163.055...

l (a scalar) =   1.09409
```

Analysis

This example shows how `optimize` can be used to solve a simple problem with a minimum of knowledge concerning Xmath or the `optimize` options. The volume of the box is 100 to 10 significant digits, signifying that the final parameters give the box a volume of $100m^3$.

2.4.2 Trajectory Optimization (Zermelo's Problem)

Problem Definition

In this example, a ship must travel through water where the current varies with (x,y) location. Given any initial point (x_o, y_o), we want to find the trajectory of the ship that will place it at the origin (0,0) in minimum time. In this example, we assume that the ship travels at a constant speed V, that the current flows in the x direction, as shown in Figure 4 on page 24, and that the speed of the current varies linearly with y position.

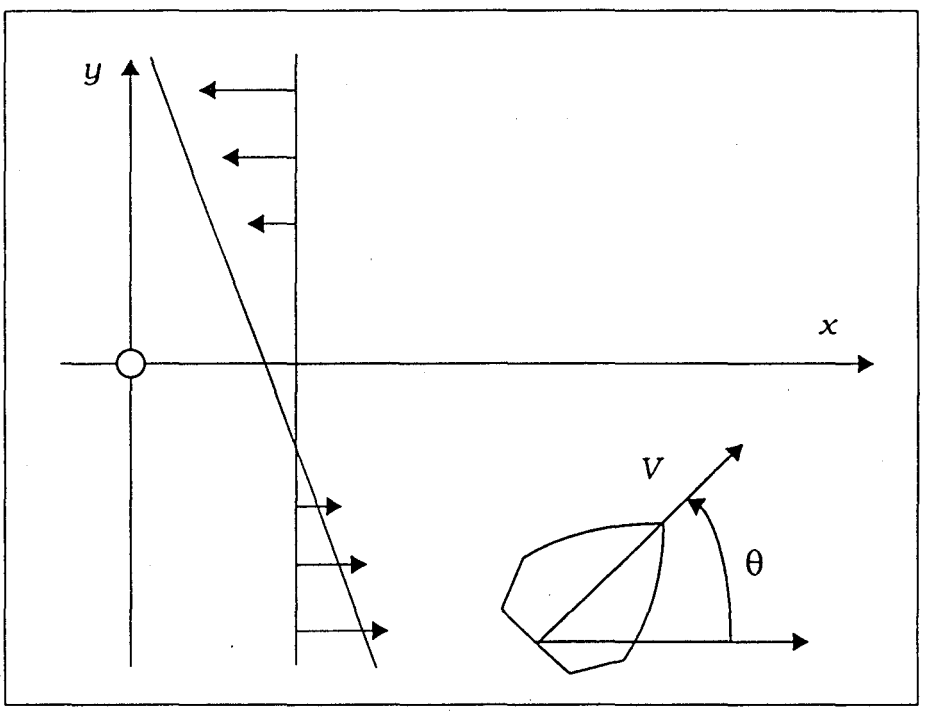


Figure 4 Ship Trajectory Optimization Problem

Formulation

The problem solving strategy is to constrain the final (x,y) position of the ship to (0,0) and minimize the time it takes the ship to reach this origin. The `cost` function parameters are the total time and the value of the steering angle at eleven evenly-spaced time points. The `cost` function takes the inputs and simulates the motion of the ship over the time interval. It computes the final position and returns the total time employed and the final distance from the origin.

The first step is to write the equations of motion for the ship in the coordinate system of Figure 4.

$$\dot{x} = V\cos\theta + ky$$
$$\dot{y} = V\sin\theta$$

For V=1, k=-1

$$\dot{y} = \sin\theta$$
$$\dot{x} = \cos\theta - y$$

Recall that the current acts only in the x direction and is a linear function of the ship's y position.

The next step is to implement these equations in a SystemBuild block diagram where we can simulate the (x,y) ship position as the input angle θ (t_k) varies.

The block diagram for the equations of motion is shown in Figure 5 on page 25. It takes θ, calculates the right hand side of the differential equation, and integrates the velocities to give x and y positions. Before proceeding with the example, start SystemBuild and load the data file for this model from the demo directory appropriate to your operating system:

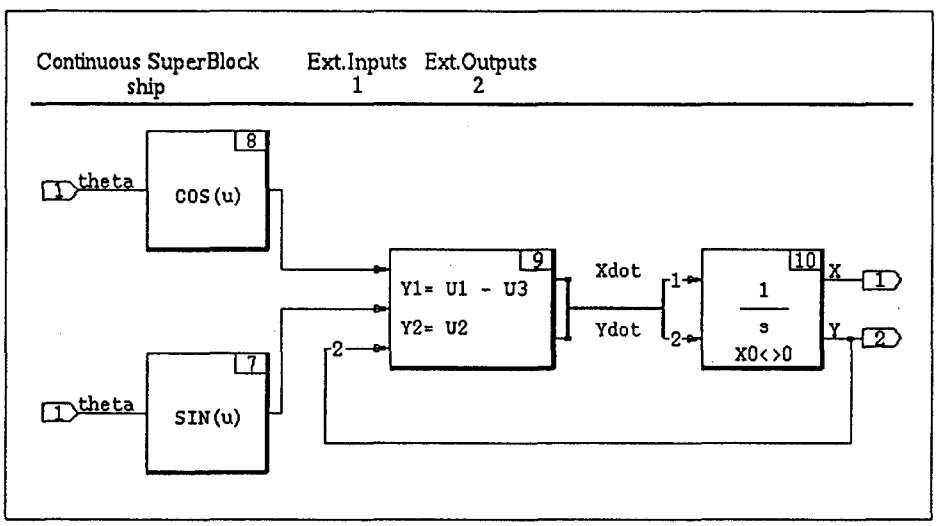


Figure 5 Ship Block Diagram

```
build
load file="$XMATH/demos/ship/ship.dat"   # UNIX
load file="$XMATH:[demos.ship]ship.dat" # VMS
```

Initial conditions for this example are set in the SystemBuild integrator block at:

$$(x_o, y_o) = (3.5, -1.8)$$

The cost function for this example is fairly simple:

```
#{For clarity we use unnecessary calculations
  and temporary values in the code below. This
  will slow down the optimization procedure.}#

function [out]=cost(p,iter)
theta=p(1:11);
tf=p(12);                                  # final time
time=[0:.1:1]'*tf;                         #time vector
xy=sim("ship",time,theta,{simclock=0,simmessage=0});
temp=makematrix(xy);
xf=temp(1,11);
yf=temp(2,11);              #final x and y positions
r=xf**2+yf**2;              # distance to the origin
out=[tf;r];                 #out=[objectv;eq_constr]
x=temp(1,:);y=temp(2,:);

if iter>0                   #plot at major iteration
   plot(x,y,{xmin=-1,xmax=5,ymin=-2,ymax=3,keep})?
endif
endFunction
```

You can copy the above cost function to your working directory with the command appropriate to your operating system:

```
% cp $XMATH/demos/ship/cost.msf .      # UNIX
$ copy $XMATH:[demos.ship]cost.msf [] # VMS
```

This MSF takes an input steering angle (as a function of time) and computes the trajectory and final (x,y) position of the ship via the sim function. The only tricky part of the cost function is the specification of the course heading as a function of time.

The set of steering angles (θ) is specified at 11 evenly-spaced time points (theta=p(1:11)).

We do not know the total time (the goal is to minimize the time to reach the origin), so we will recompute the simulation time vector from the normalized time column vector time before each simulation: time=[0:.1:1]'*tf.

This vector is the scaled time vector for the simulation. This gives the algorithm complete control over the total time needed to reach the origin. `tf(p(12))` is returned as the first element of the `out` vector; this parameter will be minimized by the `optimize` function. The sum squared of the terminal position (x,y coordinates) is returned as the equality constraint because we want the terminal position to be at the origin (0,0).†

The final line of the `cost` function checks to see if a major iteration has been completed. Upon completion of an iteration, the `cost` function plots the ship's x,y trajectory.

1 Once the cost function has been created in `cost.msf` (the text file) enter Xmath and define `cost` as an MSF. Note: if you have previously defined a cost file and now want to substitute a newer cost file, you must first undefine the old file:

```
undefine cost
define cost
```

2 Create the parameter bounds $-2p \le q(t) \le 2p$ (as arbitrarily large numbers; they are not expected to have much effect on the convergence process):‡

```
plower = [-2*pi*ones(11,1);0];
pupper = [2*pi*ones(11,1);20];
```

3 For an initial guess at the correct steering trajectory, assume that the ship maintains a constant heading in the general direction of the origin.

```
p0 = ones(11,1)*pi*2/3;
```

4 We guess that the ship will take 5 seconds to reach the origin:

```
p0 = [p0;5];
```

5 Run the cost function with the initial parameters. The graphics window displays a plot of trajectory from the initial parameter (Figure 6 on page 28).

```
cost(p0,1);
```

† Note that it is much more efficient to define the single constraints G1(p)=x(tf)2 + y(tf)2 = 0 than to have two constraints: G1(p)=x(tf) = 0, G2(p)=y(tf) = 0.

‡ It is tempting to constrain the steering angle θ to be between 0 and 2π, or between π and π, but this would not allow the ship to smoothly complete more than one circle.

Optimization

For the optimization phase, the penalty parameter `rho` is set to 1. The number of major and minor iterations are set empirically to 10 and 3. The tuning parameters have been reduced to cut the running time of the example.

```
[p,jh,l]= opti(p0,{pmin=plower,pmax=pupper,
   rho=1,majit=10,minit=3,delta=1e-4,tol=1e-2})
```

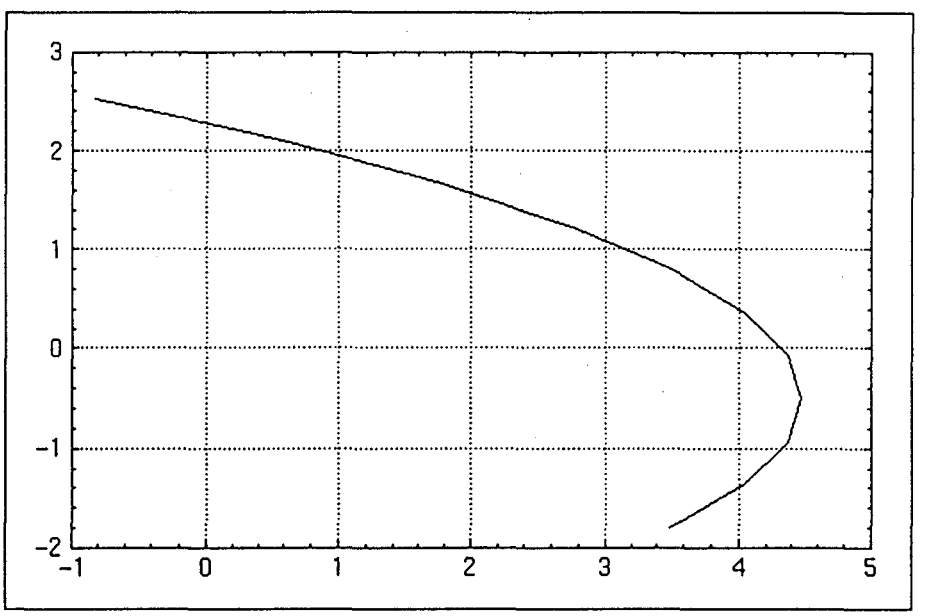


Figure 6

As the optimization proceeds, intermediate cost and constraint values are displayed, and updated parameters are output at each major iteration. A plot of the ship's trajectory is also created at each major iteration. Optimization is completed in 6 iterations.

```
Completed in 6 iterations

p (a column vector) =

1.54072
1.7487
2.31885
2.39624
```

```
1.68404
2.19942
2.91508
3.23675
3.5521
4.06592
3.71046
5.36374

jh (a row vector) = 5     4.41321     4.77413     ...

l (a scalar) = -16.5989
```

The final plot is shown in Figure 7. Note that the initial parameter plot is included in the graph; this is because `cost.msf` uses the `keep` keyword with `plot`. If you do not want the plots combined, type `erase` in the commands window command area before running `optimize`.

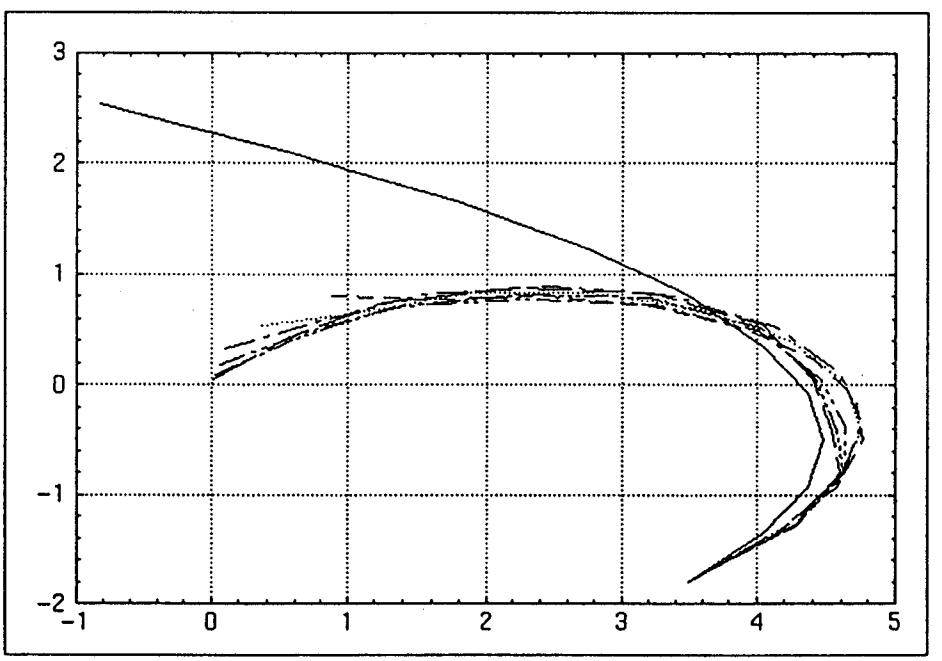


Figure 7 Ship Trajectory Plot

Analysis

The final time value is 5.36374. Figure 8 shows the initial response plotted with the final time value. To create this plot type:

```
erase
cost(p0,1)
cost([p;5.36374],1)
```

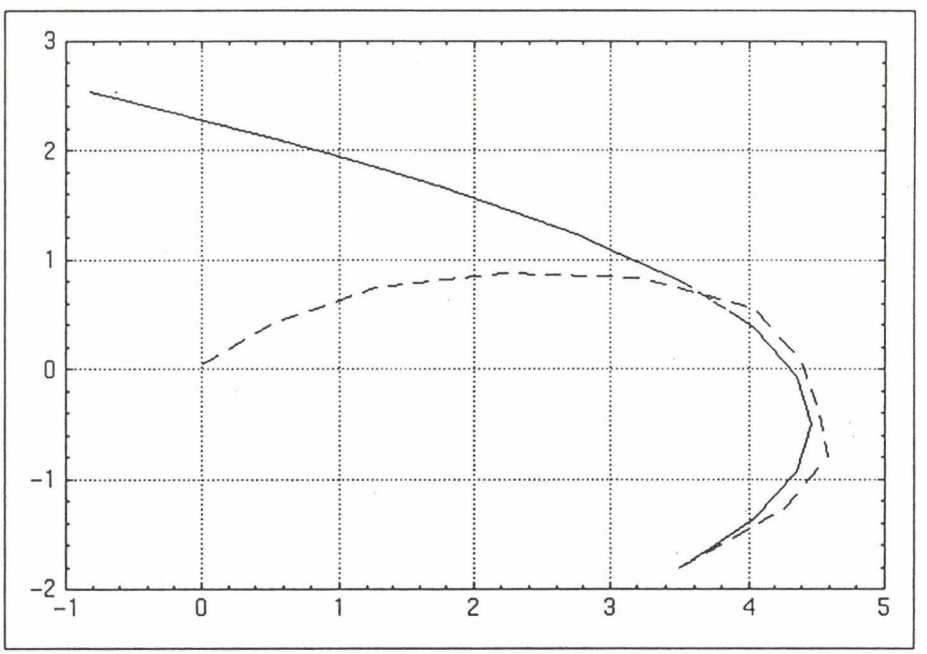


Figure 8 Initial vs. Optimal Trajectory

This was a concise, but very nonlinear trajectory optimization problem. Had we chosen a greater number of minor iterations we might have obtained a faster or better solution. If the tolerance had been reduced, or allowed to run at default the problem would have taken considerably longer to converge. You may want to experiment and fine tune the algorithm by adjusting some of the parameters.

As we have illustrated, one useful feature of `optimize` is the ability to change the iteration parameters and tolerances to help convergence at any intermediate point in the optimization process, without having to start over again or go through

extensive work to compile, re-define or re-submit the optimization job.

Note also that the value of the constraint has been reduced by the last iteration, and indeed the divergence from the origin of the last two iterations is scarcely visible in Figure 7.

2.4.3 Feedback Control Design

Problem Definition

In this example, we use `optimize` to specify feedback control logic for the closed-loop system shown in Figure 9.

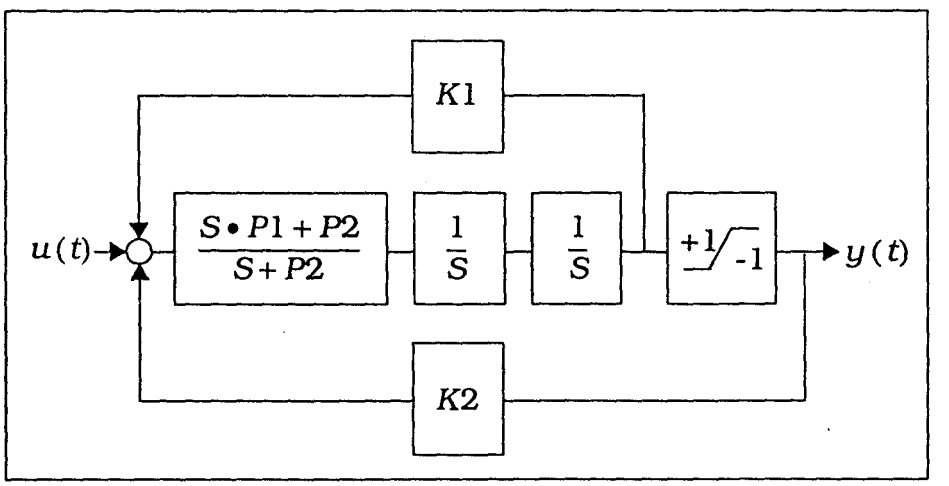


Figure 9 Closed-loop Control System Model

The objective is to find the values of the feedback gain parameters K_1 and K_2, along with the compensator coefficient parameters *p1*, and *p2* such that the response at *y* from a step input at *u* is as *good* as the saturation specified in the actuator will allow.

We want the output signal to follow the input step as closely as possible while keeping the maximum percentage overshoot less than5% (Figure 10). Our working strategy is to constrain the overshoot and hope that an acceptable settling time

results. If the overshoot criteria is met but the settling time is unacceptable, the problem can be reformulated.

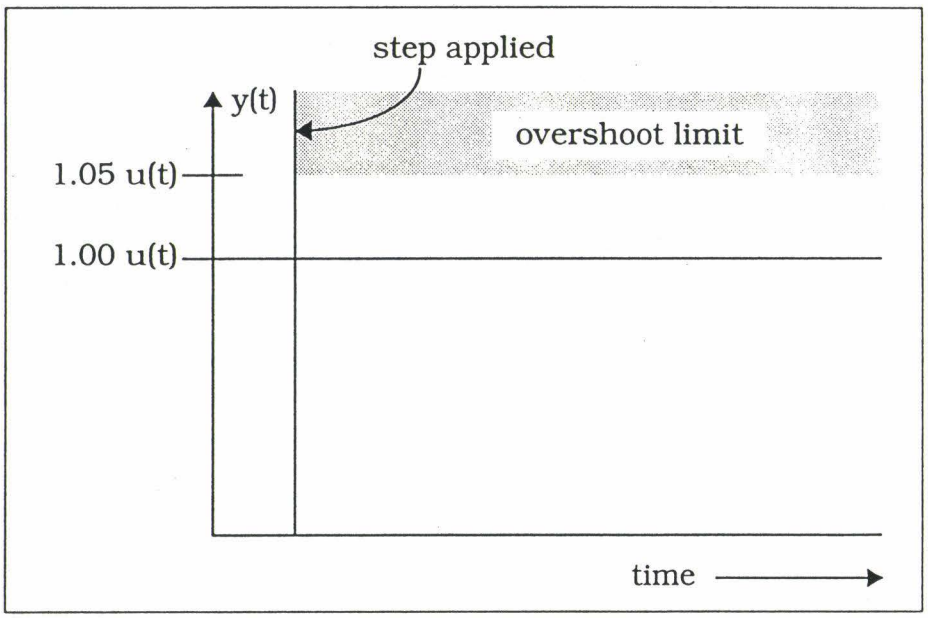


Figure 10 Desired Step Response

The first step is to create a SuperBlock diagram based on the model (Figure 9). The model system has a single input where the step input is applied, and a single output where the position response is measured.

Table 1 shows the parameters and their initial values and how they will be used in the SuperBlock.

Table 1 Parameters and Initial Values

Block	Parameter	Value	Parameter
compensator	NUM=[p1, p2)	[1,1]	vector of numerator coefficients
compensator	DEN=[1,p2]	[1,1]	vector of denominator coefficients

Table 1 Parameters and Initial Values (continued)

Block	Parameter	Value	Parameter
rate gain	K1	1	rate gain
position gain	K2	1	position gain

You may load the model shown in Figure 11 from the demo directory with the command appropriate to your operating system:

```
load file="$XMATH/demos/system/system.dat"  #UNIX
load file="$XMATH:[demos.system]system.dat" #VMS
```

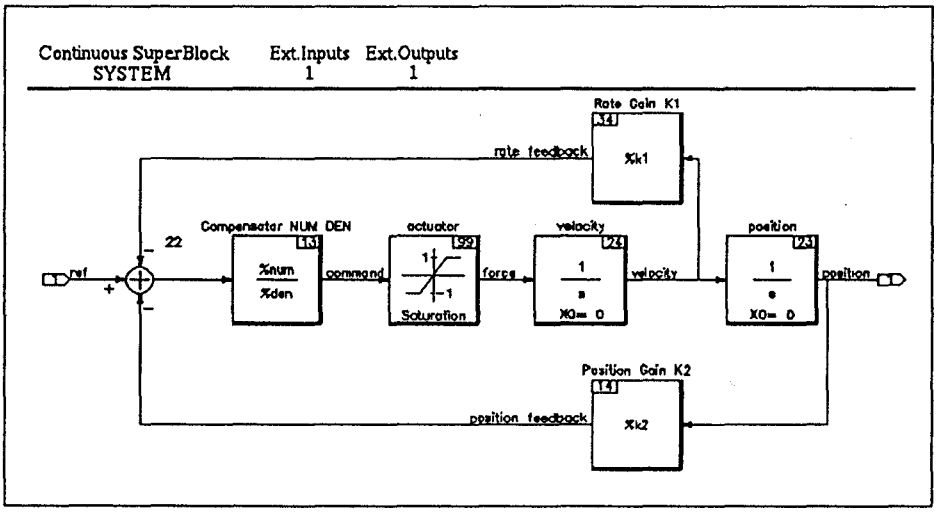


Figure 11 Closed-loop SuperBlock

Alternatively, you may define a continuous SuperBlock called SYSTEM. Build the model as shown in Figure 11 on page 33. When defining the block named Compensator NUM DEN use the values of the variables NUM and DEN. The form for the block compensator, should duplicate Figure 12. When defining the gain blocks named 'rate gain' and 'position gain', use the appropriate values of K_1 and K_2. Finally, parameterize NUM, DEN, K1 and K2 in the appropriate blocks.

Note that the SuperBlock SYSTEM contains a saturation block. We know in advance that saturation will not be

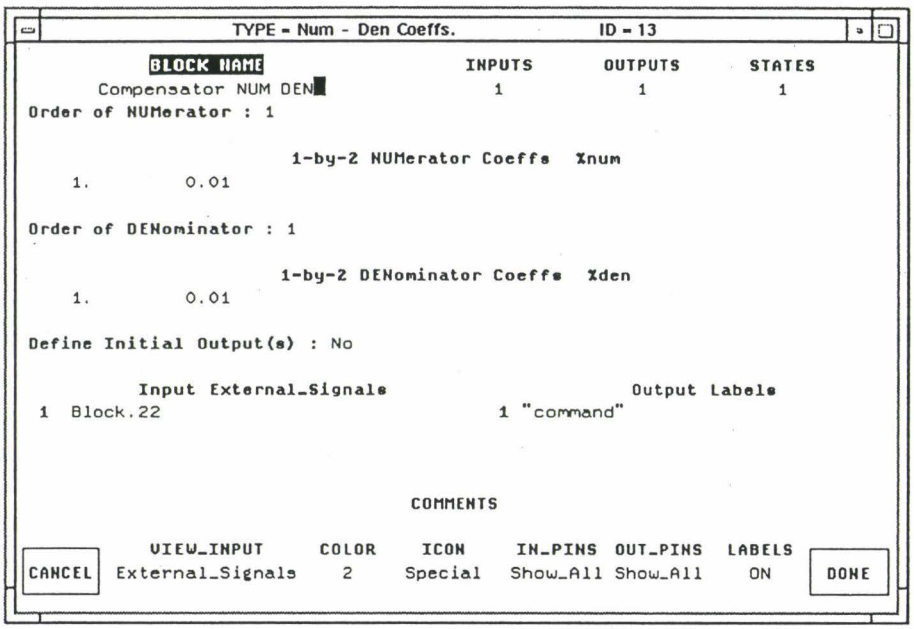


Figure 12 Designating Parameters

reached, so this is acceptable for this example. However, saturation blocks, or any blocks that might introduce linearities should be avoided. If saturation is reached, optimization will stop. Another way to handle the above problem would be to remove the saturation block and limit the signal with `icmin=-1, icmax=1`. See page 14.

After the `SYSTEM` model is built, construct `cost.msf`, which uses the model to compute the performance parameters. You can copy the cost function to your working directory with the command appropriate to your operating system:

```
cp $XMATH/demos/system/cost.msf .      # UNIX
copy $XMATH:[demos.system]cost.msf [] # VMS
```

Note that since the `SYSTEM` model uses parameterized values, `cost.msf` must specify the partition name along with the variable name, e.g., `main.num` so that SystemBuild can find and update the values.

```
Function [out]=cost(p,it)
main.num=[p(1), p(2)];
main.den=[1, p(2)];
main.k1=p(3);
main.k2=p(4);
t=[0:.1:10]';
u=ones(t);
[,y]=sim("SYSTEM",t,u,{simclock=0,simmessage=0});

# minimize dif  btwn the command and the output:

out(1)=norm(y-ones(y));

out(2)=100*(max(y)-1);   # percent overshoot

# plot at major iterations:

if it>0
        plot(t,y,{keep})?
endif

endFunction
```

`cost.msf` accepts model parameters in the *K* vector. First, the `p` vector is converted into the SystemBuild model parameters `NUM`, `DEN`, `K1` and `K2`. Next, the time and simulation vectors are created and used to simulate the step response. Once the output signal is returned, `cost` computes the rise time and overshoot. Whenever a major iteration is completed, `cost` displays the step response. The `plot` function keeps the plots for final comparison.

Optimization

1 **Define the `cost` MSF in Xmath.**

```
undefine cost      # undefine a previous cost file
define cost
```

2 **To view the initial step response, type the following values from Table 1 on page 32:**

```
p0=[1, 1, 1, 1]';
num=[p0(1), p0(2)];
den=[1, p0(2)];
k1=p0(3); k2=p0(4);
```

You may analyze SYSTEM to verify it:

```
analyze("SYSTEM")
```

Try cost with the initial parameters:

```
cost(p0,1)
```

The result in Figure 13 on page 36 shows that the step response is stable, but the overshoot is greater than the required 5%.

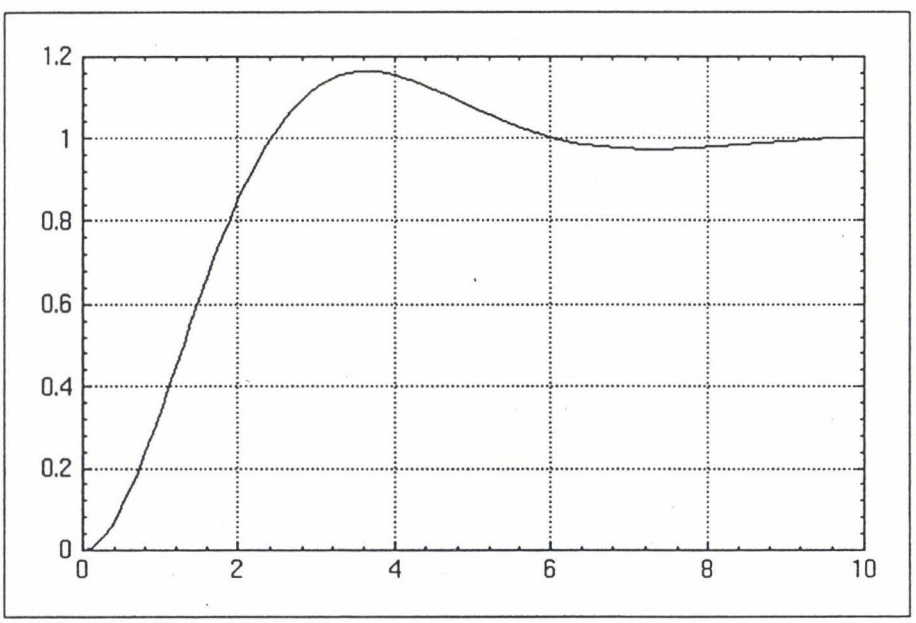


Figure 13 Step Response with Initial Parameters

3 **Start** `optimize`.

Use the default value of ρ, 1. We're arbitrarily guessing we might need a large number of major iterations (20), but reducing the number of minor iterations to 5 in hopes of

shortening running time. Set the peturbation parameter `delta` to 1×10^{-4} and the tolerance to 1×10^{-3}.

```
[P,JH,L] = opti(p0,{icmin=0,icmax=5,
   pmin=zeros(4,1),pmax=ones(4,1)*100,
   rho=1, majit=20, minit=5,delta=1e-4,tol=1e-3})
```

As the optimization proceeds you will see the following warning in the commands window message area:

```
MINOR OPTIMIZATION ROUTINE DID NOT CONVERGE IN THE
SPECIFIED NUMBER OF MINOR ITERATIONS. YOU MAY NEED
TO INCREASE THE NUMBER OF MINOR ITERATIONS.
```

This message merely indicates an intermediate point in the processing. It does not imply that the algorithm has failed or that the solution is invalid. Optimization completes in 7 iterations.

```
Beginning major iteration 7, J=2.93744
Beginning minor iteration 1
Updated parameters:  2.88236  0.476158  0.527184...
Completed in 7 iterations

P (a column vector) =

2.88236
0.476158
0.527184
1.03561

JH (a row vector) = 3.24032    3.14864    3.13624 ...

L (a scalar) = -0.0115143
```

Analysis

`optimize` computes a parameter set that drastically improves the step response of the closed-loop system. The overshoot is limited to 0.05%, (Figure 14 on page 38).

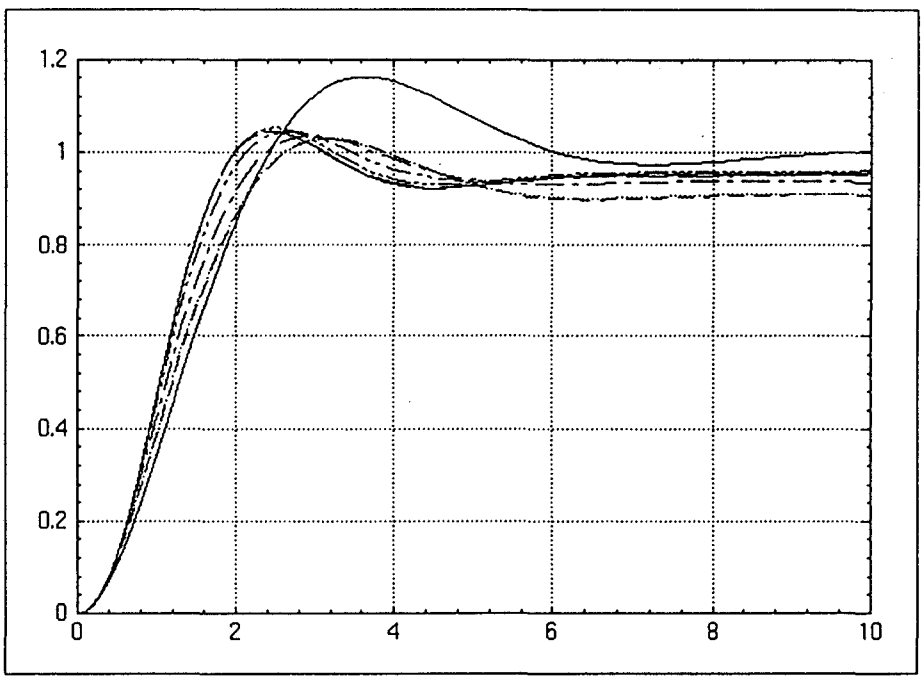


Figure 14 System Responses

Figure 15 on page 39 plots the initial step response against the final response.

```
erase
cost(p0,1)
cost([P;2.93744],1)
```

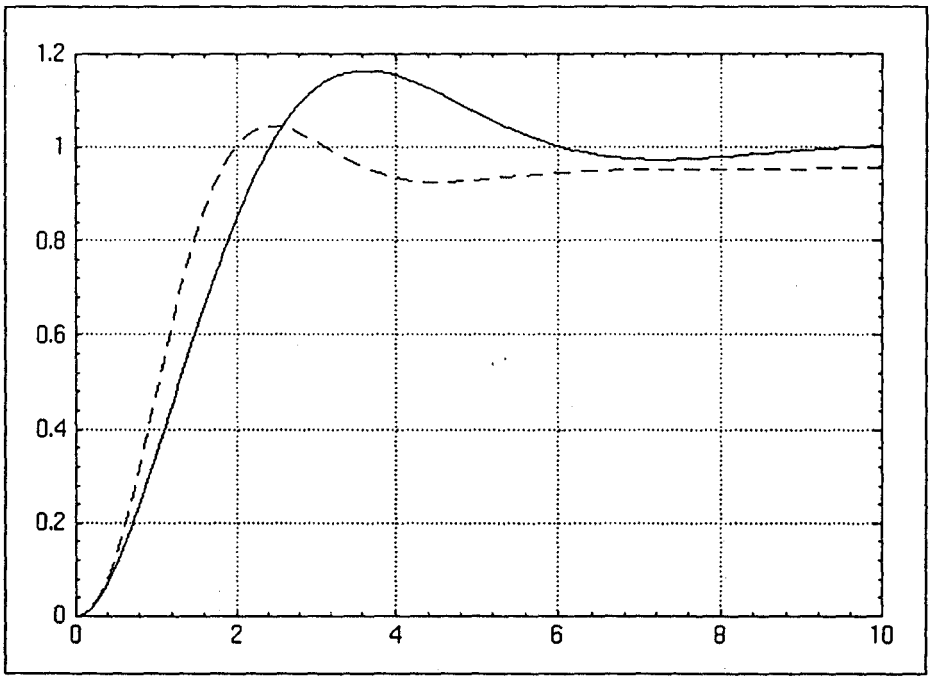


Figure 15 Initial Step Response vs. Optimal Response