République **A**lgérienne **D**émocratique et **P**opulaire
Ministère de l'**E**nseignement **S**upérieur et de la **R**echerche **S**cientifique

UNIVERSITE **M**OULOUD **M**AMMERI DE **T**IZI-**O**UZOU

FACULTE DU **G**ENIE **E**LECTRIQUE ET D'**I**NFORMATIQUE
DEPARTEMENT D'AUTOMATIQUE

Mémoire de Fin d'Etudes de MASTER ACADEMIQUE

Spécialité : **Commande des systèmes**
Filière : **Génie électrique**

Présenté par
Sadek SMAIL
Amokrane SI HADJ MOHAND

Mémoire dirigé par **Ghania IDIRI**

Thème

Commande prédictive non linéaire basée sur l'optimisation globale déterministe

Mémoire soutenu publiquement le 08 / 09/2016 devant le jury composé de :

Nom Prénom
MAA, UMMTO, Président

Nom Prénom
MAA, UMMTO, Encadreur

Nom Prénom
MAA, UMMTO, Examinateur

Nom Prénom
MAA, UMMTO, Examinateur

SOMMAIRE

Introduction générale 01

Chapitre 1 : Commande prédictive :

1.1. Introduction 04

1.2. Principe de la commande prédictive 05

1.3. Eléments de la commande prédictive 06

1.3.1. Modèle du système 07

1.3.2. Prédiction 09

1.3.3. Le critère de performance 10

1.3.4. Contraintes 11

1.3.5. La loi de commande 11

1.4. Le réglage des paramètres de la commande prédictive 12

1.5. Algorithme DMC 13

1.5.1. Modèle de prédiction 13

1.5.2. Synthèse d'un correcteur DMC 16

1.5.3. L'algorithme de commande DMC 17

1.6. Conclusion 18

Chapitre 2 :Généralités sur l'optimisation

2.1. Introduction 20

2.2. Formulation mathématique d'un problème d'optimisation 20

2.3. Classification des problèmes d'optimisation 21

2.4. Condition d'optimalité pour un problème d'optimisation 23

2.5. Méthodes de résolution des problèmes d'optimisation 25

2.5.1. Optimisation numérique 27

2.5.1.1. La structure générale d'un algorithme d'une méthode numérique 28

2.5.1.2. La méthode de gradient 28

2.5.1.3. Méthode du gradient conjugué 29

2.5.1.4. Méthode de Newton 30

2.5.1.5. Méthode de Newton modifiée 31

2.5.2. Optimisation globale 33

2.5.2.1. Méthode d'Aliénor 33

2.6. Optimisation globale et commande prédictive 38

2.7. Conclusion 39

Chapitre 3 : Commande prédictive Non linéaire Basée sur l'Optimisation Globbale

3.1. Introduction 41

3.2. Commande prédictive non linéaire 41

3.3. La méthode Aliénor appliquée sur la commande prédictive non linéaire 43

3.4. Exemple d'application 45

3.5 Conclusion 50

Conclusion générale 55

Liste des figures

Figure 1.1 : Principe de la commande prédictive

Figure 1.2 : Stratégie de la commande prédictive

Figure 3.1 : Structure générale de la commande prédictive non linéaire

Figure 3.2 : Méthode d'Aliénor : cas non linéaire sans contrainte sur la variable de commande u(k)

Figure 3.3 : Méthode d'Aliénor : cas non linéaire avec contrainte sur la variable de commande u(k)

Figure 3.4 : Méthode d'Aliénor : cas linéaire sans contrainte sur la variable de commande u(k)

Figure 3.5 : Méthode d'Aliénor : cas linéaire avec contrainte sur la variable de commande u(k)

Liste des tableaux

Tableau2.1 : Classification des problèmes d’optimisation

Tableau2.2 : Nature du point critique

Liste des indices

K	temps discret
x_i	variable de décision
N_r	l'instant de début de prédiction
N_p	l'horizon de prédiction
N_u	l'horizon de commande
N_s	la longueur de la réponse indicielle
$y(k)$	la sortie du système
$\hat{y}(k+1/k)$	la sortie prédite
y^d	Sortie désirée
$d(k)$	perturbation
$u(k)$	la commande appliquée au système
$u^*(k)$	la commande optimale
$f(i)$	les coefficients de la réponse impulsionnelle
$s(i)$	les coefficients de la réponse indicielle
$g_i(x)$	contraintes d'égalité
$h_j(x)$	contraintes d'inégalité
$f(x)$	fonction quelconque
T_e	période d'échantillonnage
t_r	temps de réponse
τ	retard de système
Z^{-1}	opérateur de retard
Q_j	matrice de pondération des états
R_j	matrice de pondération de la commande
U	vecteurregroupant la séquence des commandes
J	critèrede performance

∇J	gradient de J
$L(x)$	fonction lagrangienne
α_i	multiplicateur de Lagrange
β_j	paramètre de Kuhn-Tucker
α_k	pas de descente
r	le rayon de coordonnée polaire
θ	l'angle de coordonnée polaire
a	la spirale d'Archimède
$\frac{\partial f(x)}{\partial x_i}$	dérivée partielle première de f par rapport à x_i
$\frac{\partial^2 f(x)}{\partial {x^2}_i}$	dérivée partielle secondaire de f par rapport à x_i
$\nabla_x f(x)$	gradient de f par rapport à x
$\nabla_x{}^2 f(x)$	Hessienne de f par rapport à x
Q	matrice constante symétrique de dimension $n \times n$
A	matrice constante de dimension $p \times n$
b	vecteur constant de dimension $p \times 1$
C	vecteur constant de dimension $n \times 1$

Introduction générale

La commande prédictive à base de modèle (MPC : Model Predictive Control) est une technique de commande avancée dans l'automatique. C'est la technique la plus indiquée pour la commande des systèmes complexes dans les milieux industriels, tels que la pétrochimie, le génie des procédés et le domaine de l'automobile où elle a démontré ses preuves. Sa faculté à traiter conjointement les spécifications de fonctionnement et les contraintes d'exploitation dans l'élaboration de la loi de commande ont fait son succès. L'évolution de l'informatique n'est pas non plus étrangère à son succès, du fait que les puissants ordinateurs développés ont permis aux automaticiens d'atteindre leurs objectifs avec une grande précision dans un lapse de temps record. Tous ces facteurs ont propulsé la commande prédictive pour occuper une place prépondérante dans les milieux académiques et industriels.

Dans une stratégie de commande prédictive, les objectifs sont spécifiés par un critère à minimiser et des contraintes à imposer sur les variables d'état, de commande et de sortie [1,2,3]. Le principe de la commande prédictive impose qu'à chaque instant d'échantillonnage, la commande à appliquer est obtenue en résolvant un problème d'optimisation avec contraintes en un temps inférieur à la période d'échantillonnage [2,3].

La commande prédictive est une commande basée sur le modèle. Etant donné que le critère à minimiser est généralement quadratique et qu'il exprime des objectifs de poursuite et de minimisation d'énergie, la nature du problème d'optimisation dépend généralement du type de modèle utilisé pour la prédiction des sorties futures [1,2]. La précision et la complexité du modèle influencent directement sur les performances à obtenir. Lorsque le modèle est linéaire, la solution analytique du problème d'optimisation existe et simple à calculer et les performances seront meilleures puisque l'optimum global est atteint.

Cependant, comme la quasi-totalité des systèmes sont de nature non linéaire, pour lesquels l'hypothèse de linéarité ne présente aucun intérêt, alors l'utilisation d'un modèle non linéaire pour la prédiction des sorties futures s'impose. Résoudre, à chaque instant d'échantillonnage, un problème d'optimisation de nature non linéaire est une tâche difficile et ne permet pas l'application de la commande prédictive à des systèmes non linéaires [2].

Afin d'aboutir aux meilleures performances possibles pour notre système, il est impératif d'atteindre l'optimum global du critère avec une très grande précision et une convergence assurée en un temps inférieur à la période d'échantillonnage.

Dans ce mémoire, on s'intéressera à étudier le rôle des méthodes d'optimisation globale dans une stratégie de commande prédictive d'un système non linéaire. Notre objectif consiste à utiliser la méthode d'Aliénor afin de démontrer son efficacité.

Notre mémoire sera organisé comme suit :

Le premier chapitre est consacré à des généralités sur la commande prédictive.

Le deuxième chapitre est consacré à des notions sur l'optimisation globale.

Le troisième chapitre aborde la commande prédictive non linéaire basée sur la méthode d'optimisation globale d'Aliénor. A la fin du chapitre, on étudiera un exemple d'application avec l'interprétation des résultats de simulation obtenus.

Chapitre 1

1.1 Introduction :

La commande prédictive à base de modèle est une approche de commande avancée [2,3]. Elle est utilisée principalement dans les raffineries de pétrole, l'industrie chimique et agro-alimentaire, l'industrie de l'automobile, la métallurgie, l'aérospatiale…etc.

Le terme commande prédictive ne désigne pas une stratégie de commande spécifique mais un ensemble d'algorithmes qui utilisent explicitement le modèle du système dans un problème d'optimisation, à résoudre, pour déterminer la commande à appliquer. Le modèle du système est utilisé pour la prédiction du comportement dynamique future du système.

La commande prédictive basée sur le modèle, présente des avantages qui ont fait d'elle une approche de commande attractive. Parmi ces avantages, on peut citer [1,2]:

- Elle permet de respecter les différentes contraintes sur les états, les commandes, et les sorties.
- Elle évite les variations excessives pour les variables de commandes.
- Elle peut être utilisée pour commander une grande variété de processus ; ceux qui sont avec des dynamiques simples et ceux plus complexe. Par exemple : les systèmes avec ou sans retard, les systèmes stables ou instables en boucle ouverte, les systèmes à phase minimale…etc.
- Son réglage est aisé et son principe est intuitif.
- Elle est assez robuste aux erreurs de modèle.
- Elle est très performante lorsque les consignes ou trajectoires à suivre sont connues à l'avance.
- Le réglage de ses paramètres relativement facile la rend accessible aux personnes avec des connaissances limitées en automatique.

Ce chapitre est consacré à la commande prédictive. Il sera présenté le principe de la commande prédictive, l'algorithme de la commande prédictive, les éléments et les paramètres de la commande prédictive et, à la fin, on illustrera notre travail par un exemple sur la commande prédictive.

1.2 Principe de la commande prédictive :

La commande prédictive est une commande basée sur le modèle dont le principe consiste à minimiser l'écart entre les sorties prédites du système et la consigne désirée à chaque instant d'échantillonnage. Le but est de créer, pour le système à commander, un effet anticipatif par rapport à une trajectoire à suivre connue à l'avance. Ceci est réalisé en se basant sur la prédiction du comportement futur du système et en minimisant l'écart de ces prédictions par rapport à la trajectoire, au sens d'une certaine fonction coût, tout en respectant les contraintes de fonctionnement [2].

La détermination de la loi de commande prédictive se fait par résolution d'un problème d'optimisation à horizon fini, à partir d'une trajectoire de référence à suivre connue à l'avance selon l'algorithme suivant :

1- Calculer les sorties prédites $\hat{y}$ sur un horizon de prédiction N_p, à chaque instant d'échantillonnage k. les prédictions $\hat{y}(k+j/k), pour\ j = 1, \dots, N_p$, sont calculées en utilisant le modèle du système et dépendent des sorties et commandes passées et des commandes futures $(u(k+i), pour\ i = 1, \dots, N_p - 1)$. Ainsi les prédictions ne vont pas dépendre uniquement des sorties précédentes mais aussi de l'évolution future de la variable de commande.

2- Calculer le critère qui est une fonction quadratique qui consiste à minimiser l'erreur entre les sorties prédites et la sortie désirée. La séquence de commande à calculer est incluse dans ce critère à minimiser. Dans le cas où le critère est quadratique et le modèle est linéaire une solution analytique peut être obtenue, sinon une méthode numérique doit être utilisée pour la résolution du problème. Notons qu'après un certain nombre d'échantillonnage la commande reste constante.

3- La résolution du problème d'optimisation permet d'obtenir une séquence de commande $(u(k), u(k+1), \dots u(k+N_u))$. Appliquer au système réel uniquement la première valeur de cette séquence u(k) à l'instant k. A l'instant d'échantillonnage suivant k+1, le principe sera répété, ce qui permet d'obtenir une commande réactualisée.

La figure (1.1) présente le principe de la commande prédictive.

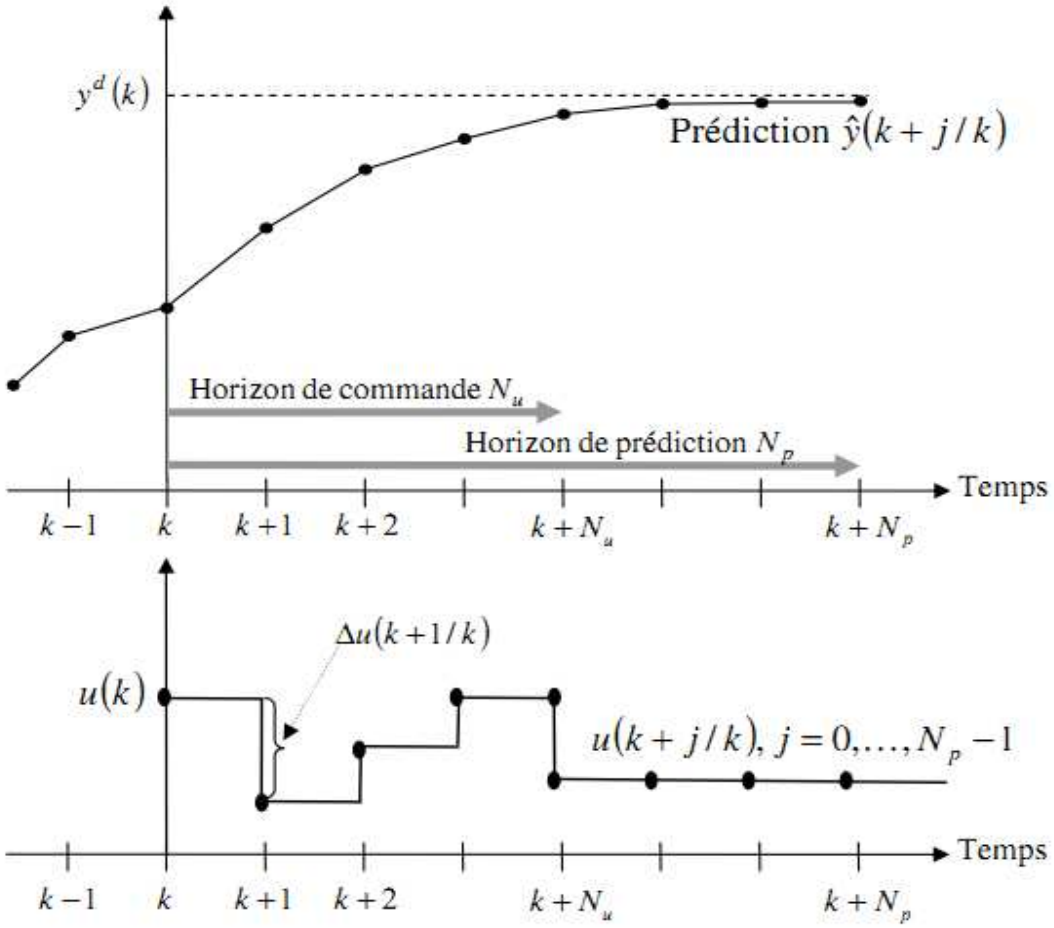


Figure 1.1 : principe de la commande prédictive.

1.3 Eléments de la commande prédictive:

Les algorithmes de la commande prédictive sont basés sur les mêmes éléments (**Figure 1.2**), par contre, différentes options peuvent être considérées pour chaque élément, ce qui nous permet d'avoir une multitude d'algorithmes.

Ces éléments sont :

- Le modèle du système
- La prédiction
- Le critère de performance
- Les contraintes

- La loi de commande

1.3.1 Modèle du système :

Dans la commande prédictive, un modèle du système n'est pas utilisé pour la conception de la loi de commande, mais il est utilisé pour prédire l'évolution future des sorties du système. Par conséquent, le modèle du système fait partie de l'algorithme de commande. Ainsi la prédiction ne va pas dépendre uniquement des sorties précédentes, mais aussi de l'évolution envisagée dans le temps futur pour la variable de commande.
Le modèle du système comprend deux parties : le modèle utilisé pour décrire le système lui-même, c'est-à-dire les relations entrées/sorties du système. Et le modèle de perturbation qui est utilisé pour approximer les erreurs de modélisation.

On peut utiliser différentes formes de modèles dans une commande prédictive, à condition qu'elles soient de nature discrète puisque la commande prédictive elle-même est une commande numérique. Les formes les plus utilisées, généralement de type entrée/sortie, elles sont présentées ci-après.

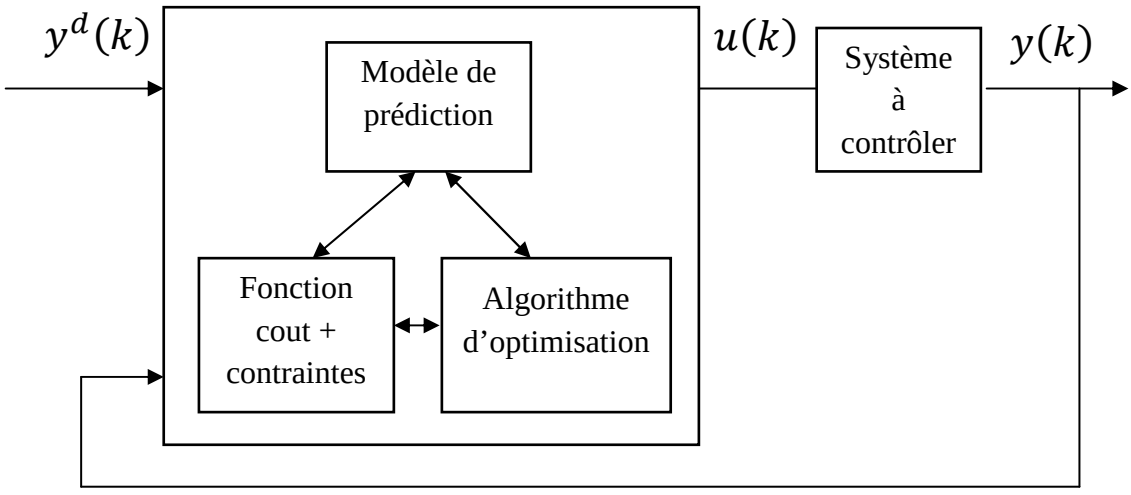


Figure 1.2 : stratégie de la commande prédictive

- **La réponse impulsionnelle :**

La réponse impulsionnelle est donnée comme suit :

$$y(k)=\sum_{i=1}^{+\infty} f(i)u(k-i) \qquad \textbf{(1.1)}$$

Où $f(i)$ représentent les coefficients de la réponse impulsionnelle (les valeurs de la sortie, aux instants d'échantillonnage, lorsque l'entrée est une impulsion de Dirac).
Souvent, cette somme est tronquée et les seule valeurs considérées sontN_s , Où N_s est le temps de stabilisation du processus, c'est-à-dire à partir de N_{s+1}, les réponses impulsionnelle sont nulles. $N_s \times T_e$ représente le temps de réponse du systeme.

- **La réponse indicielle :**

La réponse indicielle est donnée comme suit :

$$y(k) = \sum_{i=1}^{+\infty} s(i)\Delta u(k-i) \qquad \textbf{(1.2)}$$

Où $s(i)$ représentent les coefficients de la réponse indicielle pour une entrée échelon unité, cette entrée échelon unitaire est donnée comme suit :

$$u(k) = \begin{cases} 0 \ \forall\, k < 0 \\ 1 \ \forall\, k \geq 0 \end{cases} \qquad \textbf{(1.3)}$$

$\Delta u(k)$ est l'incrément de commande d'où : $\Delta u(k) = u(k) - u(k-1)$.
Pour un système stable, les coefficients $s(i)$ sont constants après l'instant N_s .
$N_s \times T_e$ représente le temps de réponse du système.
Comme les coefficients de la réponse impulsionnelle peuvent être considérés comme la différence entre deux coefficients successifs de la réponse indicielle, alors les relations suivantes sont vérifiées :

$$f(i) = s(i) - s(i-1) \quad \textbf{(1.4)}$$

$$s(i) = \sum_{j=1}^{+\infty} f(i) \quad \textbf{(1.5)}$$

- **L'équation aux différences :**

La sortie du système est donnée comme suit :

$$y(k) = F\left(y(k-1), \dots, y\left(k-n_y\right), u(k-1), \dots, u(k-n_u)\right) \quad \textbf{(1.6)}$$

Où n_y et n_u sont des entiers naturels. La fonction $F(.)$ peut être linéaire ou non linéaire selon la nature du système.

1.3.2 Prédiction :

Les prédictions sont faites en utilisant le modèle du système. Ainsi la prédiction ne va pas dépendre uniquement des sorties précédentes, mais aussi de l'évolution envisagée dans le temps futur pour la variable de commande. Ces prédictions seront utilisées par la suite pour la détermination de la variable de commande en résolvant un problème d'optimisation.

Comme exemple, considérons le modèle suivant :

$$y(k) = -a\, y(k-1) + b\, \Delta u(k-1) \quad \textbf{(1.7)}$$

Les deux premières prédictions futures de la sortie sont données comme suit :

$$\hat{y}(k+1/k) = -a\, y(k) + b\, \Delta u(k) \quad \textbf{(1.8)}$$

$$\hat{y}(k+2/k) = -a\, y(k+1) + b\, \Delta u(k+1)$$

$$\hat{y}(k+2/k) = -a\left(-a\, y(k) + b\, \Delta u(k)\right) + b\, \Delta u(k+1)$$

$$\hat{y}(k+2/k) = a^2 y(k) - ab\Delta u(k) + b\Delta u(k+1) \quad \textbf{(1.9)}$$

Les autres prédictions sont déterminées de la même manière en utilisant le modèle sous forme matricielle. Les deux premières prédictions sont données comment suit :

$$\begin{bmatrix}\hat{y}(k+1/k)\\ \hat{y}(k+2/k)\end{bmatrix} = \begin{bmatrix}-a\\ a^2\end{bmatrix} y(k) + \begin{bmatrix} b & 0\\ -ab & b\end{bmatrix}\begin{bmatrix}\Delta u(k)\\ \Delta u(k+1)\end{bmatrix} \quad \textbf{(1.10)}$$

Les prédictions sont composées de deux termes, le second terme du côté droit de l'équation dépend des entrées futures. Par contre le premier terme dépend seulement des sorties précédentes ou également des entrées précédentes selon le système.
Les prédictions peuvent être décomposées en général en deux parties :

$$\hat{y}(k+j/k) = \hat{y}_{libre}(k+j) + \hat{y}_{forcée}(k+j) \quad \textbf{(1.11)}$$

La réponse libre $\hat{y}_{libre}(k+j)$, correspond à la prédiction de la sortie lorsque la commande est maintenue constante à sa valeur $u(k)$ le long de l'horizon de prédiction N_p. Par contre la réponse forcée $\hat{y}_{forcée}(k+j)$ correspond à la prédiction de la sortie due aux actions futures $u(k+j), j = 1, \dots, N_p$

1.3.3 Le critère de performance :

La commande prédictive consiste à minimiser le critère de performance, appelé aussi fonction objectif. Ce critère dépend des commandes appliquées (les entrées) ainsi que des grandeurs de sortie du système.

Pour déterminer la loi de commande, les algorithmes de la commande prédictive utilisent plusieurs formes pour le critère, le but recherché est d'assurer la poursuite de la consigne désirée $y^d(k+j)$avec un minimum d'effort $u(k+j-1)$.
Le critère utilisé le plus souvent prend la forme quadratique suivante :

$$j = \sum_{j=N_r}^{N_p} \left[y^d(k+j) - \hat{y}\left(k+\frac{j}{k}\right)\right]^T Q_j \left[y^d(k+j) - \hat{y}\left(k+\frac{j}{k}\right)\right] + \sum_{j=0}^{N_u-1} u(k+j-1)^T R_j u(k+j-1) \quad \textbf{(1.12)}$$

Où les matrices de pondération Q_j et R_j sont symétriques et définies positives.

Les paramètres $N_r, N_p \; et \; N_u$, sont des paramètres de réglage du correcteur prédictif et représentent respectivement l'instant du début de la prédiction, l'horizon de prédiction maximal et l'horizon de commande.

- L'horizon de prédiction N_p représente la longueur de l'intervalle du temps futur sur lequel on cherche à minimiser le critère.
- L'horizon de commande N_u représente le nombre de variations que l'on l'autorise dans le futur.

Remarque :

- Pour les algorithmes qui utilisent un modèle de convolution tronqué, on utilise un autre paramètre N_s qui représente la longueur de la réponse indicielle ou impulsionnelle.
- Il faut choisir convenablement la période d'échantillonnage T_e puisque le correcteur prédictif est de type discret. Si on définit le temps de réponse t_r comme le temps à partir duquel la réponse du système atteint 99% de la valeur final, on doit vérifier :

$$N_s \times T_e \; = \; t_r \qquad \textbf{(1.13)}$$

1.3.4 Contraintes :

Les contraintes caractérisent en général les limitations physiques sur la commande, sur l'état ou sur la sortie du système. Elles sont introduites pour éviter des changements brusques de la commande. Pour certains systèmes, des contraintes sur les sorties (prédites) doivent être respectées pour des raisons économiques ou de sécurité. Généralement, les contraintes sont instantanées et s'expriment par des inégalités de la forme :

$$U_{min} \leq \; \mathrm{U}(k) \; \leq \; U_{max} \forall \; k \qquad \textbf{(1.14)}$$

$$y_{min} \leq \; \hat{y}(k+j/k) \leq \; y_{max} \, , N_r \; , \; \leq j \; \leq N_p, \; \forall \; k \qquad \textbf{(1.15)}$$

1.3.5 La loi de commande :

Le principe de la commande prédictive stipule qu'on commence d'abord par la détermination des prédictions en utilisant le modèle du système [2,4]. Ces dernières seront forcément fonction des actions de commande $u(k+i)$. Ensuite, en substituant ces prédictions dans le critère J à minimiser, on va résoudre le système d'équations algébriques obtenu en imposant le gradient ∇J par rapport aux actions de commande $u(k+i)$ égale à zéro, c'est-à-dire :

$$\nabla_{(u(k),u(k+1),\dots,u(k+N_u))} J\big(u(k),u(k+1),\dots,u(k+N_u)\big) = \begin{bmatrix} \frac{\partial J(u(k),\dots,u(k+N_u))}{\partial u(k)} \\ \frac{\partial J(u(k),\dots,u(k+N_u))}{\partial u(k+1)} \\ \vdots \\ \frac{\partial J(u(k),\dots,u(k+N_u))}{\partial u(k+N_u)} \end{bmatrix} = 0 \qquad \textbf{(1.16)}$$

Une fois le système d'équations algébriques résolu, on détermine les actions de commande $u^*(k+i)$ optimales.

Rappelons que, d'après le principe de la commande prédictive, seule l'action $u^*(k)$ sera appliquée au système et que la procédure sera répétée à l'instant d'échantillonnage suivant.

Remarque :

Dans le cas où on résout le système avec une méthode numérique (itérative), le temps de convergence ne doit pas dépasser une période d'échantillonnage pour que l'action optimale $u^*(k)$ soit disponible à l'instant k .

1.4 Le réglage des paramètres de la commande prédictive :

En plus de la période d'échantillonnage, une commande prédictive est caractérisée par les paramètres de réglage suivants [3,4]:

- L'instant du début de prédiction N_r : ce paramètre est pris égal au retard du système τ.
  Si le système ne présente pas de retard, on prend $N_r = 1$.
- Horizon du système N_s: il est déterminé selon la relation. $N_s \times T_e = t_s$. Des valeurs entre 20 et 70 sont des recommandations typiques

- Horizon de commande N_u : il représente le nombre d'actions futures calculées par optimisation du critère pour réduire les erreurs de prédictions. La valeur de ce paramètre sur le temps, calcule la solution optimale. L'augmentation de ce paramètre permet d'accélérer le système, mais en contrepartie, on va avoir des actions plus énergiques. Comme le principe de la commande prédictive consiste à appliquer seulement la première commande, alors il est inutile de prendre une valeur supérieure à 5.
- Horizon de prédiction N_p : il représente le nombre de prédictions utilisées dans le critère à optimiser. La valeur minimal doit êtreégale a $\tau + 1$ où τ désigne le temps de retard du système à commander. L'augmentation de N_p revient à considérer plus de prédictions, par conséquent, la sortie sera bien amortie et les commandes seront de faibles amplitudes, mais leur traitement nécessite un temps de calcul important.

1.5 Algorithme DMC

L'algorithme DMC (Dynamic Matrix Control ou Commande par Matrice Dynamique), est largement utilisé par les industriels, en particulier dans les secteurs des hydrocarbures et de la pétrochimie.

Dans notre cas, nous allons parler d'une version simplifiée de cet algorithme, où le modèle est fourni sous forme de réponse à un échelon. Cette formulation est généralement la plus choisie car elle permet la compréhension intuitive du fonctionnement de la commande prédictive. Néanmoins des développements équivalents peuvent être conduits pour le modèle de réponse impulsionnelle où fonction de transfert conduisant respectivement au *Model Algorithmic Control* (MAC) et au *Generalized Predictive Control* (GPC).

L'algorithme est utilisé uniquement dans le cas des systèmes linéaires [1,2].

1.5.1 Modèle de prédiction :

Le modèle utilisé pour la prédiction dans l'algorithme de commande prédictive DMC est le modèle réponse indicielle. Considérons la réponse indicielle du système

$$y(k) = \sum_{i=1}^{+\infty} s_i \Delta u(k-i) \qquad \textbf{(1.17)}$$

On peut la mettre sous la forme suivante :

$$y(k) = \sum_{i=1}^{N_s} s_i \Delta u(k-i) + \sum_{i=N_s+1}^{+\infty} S_i \Delta u(k-i) \quad \textbf{(1.18)}$$

$$y(k) = \underbrace{\sum_{i=1}^{N_s} s_i \Delta u(k-i)}_{\text{Réponse indicielle finie}} + Z(k) \quad \textbf{(1.19)}$$

Ainsi, on remarque bien que la réponse indicielle infinie du système fait intervenir une réponse indicielle finie qu'on peut exploiter pour l'étape de prédiction, c'est-à-dire :

$$\hat{y}(k+j/k) = \sum_{i=1}^{N_s} s_i \Delta u(k+j-i) + Z(k+j) \quad \textbf{(1.20)}$$

Comme nous venons de le voir, la réponse du système prédite par le modèle peut être divisée en deux parties : la réponse libre et la réponse forcée. Ainsi, l'équation (1.20) peut-être décomposée en deux termes : le premier terme contient les actions de commande futures $(\Delta u(k), (\Delta u(k+1), \dots)$, et le second terme contient les actions de commande passées $(\Delta u(k-1), \Delta u(k-2), \dots)$, ce qui donne :

$$\hat{y}(k+j/k) = \sum_{i=1}^{j} s_i \Delta u(k+j-i) + \sum_{i=j+1}^{N_s} s_i \Delta u(k+j-i) + Z(k+j) \quad \textbf{(1.21)}$$

Cette dernière équation peut être arrangée pour faire apparaître les deux termes de la sortie (forcée et libre) comme suit :

$$\hat{y}(k+j/k) = \sum_{i=1}^{j} s_i \Delta u(k+j-i) + y^*(k+j) \quad \textbf{(1.22)}$$

Où le terme :

$$y^*(k+j) = \sum_{i=j+1}^{N_s} s_i \Delta u(k+j-i) + Z(k+j) \quad \textbf{(1.23)}$$

Représente le terme libre :

$$\hat{y}_{libre}(k+j) = y^*(k+i) \quad \textbf{(1.24)}$$

Le terme restant représente le terme forcé qui est l'évolution de la réponse du système à partir de l'instant k dû à l'application des actions de commande futures. Ainsi,

$$\hat{y}_{forcée}(k+j) = \sum_{i=1}^{j} s_i \Delta u(k+j-i) \quad \textbf{(1.25)}$$

Le critère considéré dans un algorithme DMC consiste à minimiser la différence entre la consigne désirée et la sortie prédite par le modèle le long de l'horizon de prédiction N_p. Les sorties prédites sont données comme suit :

$$\begin{aligned}
&\hat{y}(k+1/k) = y^*(k+1) + s_1 \Delta u(k) \\
&\hat{y}(k+2/k) = y^*(k+2) + s_2 \Delta u(k) + s_1 \Delta u(k+1) \\
&\hat{y}(k+3/k) = y^*(k+1) + s_3 \Delta u(k) + s_2 \Delta u(k+1) + s_1 \Delta u(k+2) \\
&\cdots \\
&\hat{y}(k+N_u/k) = y^*(k+N_u) + s_{N_u} \Delta u(k) + s_{N_{u-1}} \Delta u(k+1) + \cdots + s_1 \Delta u(k+N_u-1) \\
&\vdots \\
&\hat{y}\big(k+N_p/k\big) = y^*\big(k+N_p\big) + s_{N_p} \Delta u(k) + s_{N_{p-1}} \Delta u(k+1) + \cdots + s_1 \Delta u\big(k+N_p-1\big)
\end{aligned} \quad \textbf{(1.26)}$$

Notons que :

$$\Delta u(k+N_u) = \Delta u(k+N_u+1) = \cdots = \Delta u\big(k+N_p-1\big) = 0 \quad \textbf{(1.27)}$$

car l'action de commande est maintenue constante à partir de l'instant d'échantillonnage $k = N_u$.

Les prédictions **(1.26)** peuvent être écrites sous la forme matricielle comme suit :

$$\hat{y} = y^* + M\ \Delta u \quad \textbf{(1.28)}$$

Avec :

$$\hat{y} = \begin{bmatrix} \hat{y}(k+1/k) \\ \hat{y}(k+2/k) \\ \vdots \\ \hat{y}(k+N_p/k) \end{bmatrix},\ y^* = \begin{bmatrix} y^*(k+1) \\ y^*(k+2) \\ \vdots \\ y^*(k+N_p) \end{bmatrix}, \Delta u = \begin{bmatrix} \Delta u(k) \\ \Delta u(k+1) \\ \vdots \\ \Delta u(k+N_u-1) \end{bmatrix} \quad \textbf{(1.29)}$$

Et la matrice

$$M = \begin{bmatrix} s_1 & 0 & 0 & \cdots & 0 \\ s_2 & s_1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ s_{N_u} & s_{N_{u-1}} & s_{N_{u-2}} & \cdots & s_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ s_{N_p} & s_{N_{p-1}} & s_{N_{p-2}} & \cdots & s_{N_p-N_u+1} \end{bmatrix} \quad \textbf{(1.30)}$$

est appelée matrice dynamique (Dynamic Matrix) .

1.5.2 Synthèse d'un correcteur DMC :

Rappelons que l'objectif consiste à poursuivre une trajectoire, le long de l'horizon de prédiction, donnée comme suit :

$$y^d = \begin{bmatrix} y^d(k+1) \\ y^d(k+2) \\ \vdots \\ y^d(k+N_p) \end{bmatrix} \quad \textbf{(1.31)}$$

Ainsi, pour déterminer la séquence de commande à appliquer, il suffit de résoudre, à chaque instant k, le problème d'optimisation suivant :

$$\min_{\Delta u} J(\Delta u) = (y^d - \hat{y})^T Q (y^d - \hat{y})$$

Sujet à : **(1.32)**

$$\hat{y} = y^* + M\ \Delta u$$

En substituant $\hat{y}$ dans le critère J, il vient :

$$\min_{\Delta u} J(\Delta u) = (y^d - y^* - M\,\Delta u)^T Q (y^d - y^* - M\,\Delta u) \qquad \textbf{(1.33)}$$

La solution optimale est obtenue comme suit :

$$\nabla_{\Delta u} J(\Delta u) = -2M^T Q^T (y^d - y^* - M\,\Delta u) = 0 \qquad \textbf{(1.34)}$$

Ce qui donne :

$$\Delta u = (M^T Q^T M)^{-1} M^T Q^T (y^d - y^*) \qquad \textbf{(1.35)}$$

1.5.3 L'algorithme de commande DMC :

L'objectif de la régulation est d'amener le système à suivre une trajectoire de référence y^d: $(y^d(k+1), y^d(k+2), \dots, y^d(k+N_p))$. Nous cherchons donc à minimiser un critère du type :

$$J(t) = \sum_{j=1}^{N_p} \left(y^d(k+j) - \hat{y}(k+j) \right)^2 \qquad \textbf{(1.36)}$$

De façon à obtenir une séquence d'entré variant de façon douce, nous ajoutons un terme de pénalisation des variations d'entrée :

$$J(t) = \sum_{j=1}^{N_p} \left(y^d(k+j) - \hat{y}(k+j) \right)^2 + \lambda \sum_{j=1}^{N_u} \left(\Delta u(k+i) \right)^2 \qquad \textbf{(1.37)}$$

Où λ est le facteur de pénalisation (pondération).

Le vecteur Δu qui minimise le critère $J(t)$ s'obtient par la méthode des moindres carrés :

$$\Delta \mu(k) = (M^T M + \lambda I)^{-1} A^T \left[y^d(k) - y_{passé}(k) \right] \qquad \textbf{(1.38)}$$

Seul le premier élément de cette séquence est utilisé :

$$u(k) = \mu(k-1) + \Delta u(k) \quad \textbf{(1.39)}$$

En résumé, les étapes d'un algorithme prédictif sont les suivantes :

1- Calcul des effets des actions passées, compte tenu de la valeur de la sortie du système à l'instant courant.
2- Calcul de séquence d'entrée à appliquer pour atteindre une trajectoire de référence, en minimisant un critère pondérant l'écart prédiction – référence et la variation de l'action.
3- Application effective de la première valeur.
4- Recommencer l'opération.

1.6 Conclusion :

Ce chapitre a été consacré à des généralités sur la commande prédictive. Après avoir présenté le principe de la commande prédictive et les étapes de son algorithme, nous avons détaillé les différents éléments d'une commande prédictive. A la fin, nous avons parlé des différents paramètres d'une commande prédictive, en occurrence l'instant du début de prédiction, l'horizon de commande, l'horizon de prédiction, et l'horizon du système. Et, pour illustrer, nous avons donné l'exemple de l'algorithme DMC utilisé pour le cas de la commande prédictive linéaire.

Dans une commande prédictive, l'obtention de la solution optimale passe par la résolution d'un problème d'optimisation souvent non linéaire, par conséquent cette étape d'optimisation joue un rôle du premier plan dans une stratégie de commande prédictive. En effet, le succès d'une stratégie de commande prédictive est lié directement à l'algorithme d'optimisation utilisé. Ainsi, l'examen de la littérature dédié à la commande prédictive, en particulier la commande prédictive non linéaire montre que les efforts de recherche sont focalisés, ces dernières années, au développement d'algorithmes d'optimisation rapides qui convergent vers l'optimum globale en un temps inférieur à une période d'échantillonnage, qui constituent actuellement un vrai challenge pour les mathématiciens appliqués et en particulier aux automaticiens.

Dans ce travail, un algorithme d'optimisation globale sera adopté dans la stratégie de commande prédictive non linéaire. Ainsi, le chapitre suivant est consacré à l'optimisation globale des fonctions mathématiques où les différentes notions utilisées le long du travail seront exposées.

Chapitre 2

2.1 Introduction :

Pour réussir de bonnes performances dans une stratégie d'une commande prédictive, l'étape de l'optimisation est primordiale, car les performances dépendent de la solution du problème d'optimisation [5,2]. Elles sont meilleures si l'optimum global du critère est atteint.
En plus, la contrainte relative au temps de résolution du problème d'optimisation, qui doit être inférieur à une période d'échantillonnage, doit être respectée.

L'objectif de l'optimisation, c'est d'atteindre le minimum global du critère en présence ou en absence de contraintes [2,3,5]. Mais étant donné que la plupart des problèmes d'optimisation sont non linéaires, la localisation de l'optimum globale est très difficile. C'est pourquoi, avec l'évolution de l'informatique ces dernières années, les recherches se sont focalisées sur le développement d'algorithmes sophistiqués et rapides permettant de localiser l'optimum global d'une fonction mathématique et d'échapper ainsi au problème rencontré avec les méthodes de l'optimisation locale qui sont souvent piégées dans un minimum local [2].

Ce chapitre est consacré à des généralités sur l'optimisation globale. L'objectif consiste à présenter les différentes notions d'optimisation. Ainsi, après la formulation mathématique du problème d'optimisation, on donne une classification des problèmes d'optimisation. Puis, on présentera les méthodes de résolution des problèmes d'optimisation illustrées par un exemple de ces méthodes.

2.2 Formulation mathématique d'un problème d'optimisation :

Soit $f(x)$ une fonction scalaire, appelée fonction objectif ou critère. Elle est en fonction de plusieurs variables $x_1, x_2, \dots, x_n$, appelées variables de décision. Le vecteur de variables de décision, noté $x = (x_1, x_2, \dots, x_n)^T$, doit appartenir à un domaine donné $D \in R^n$. Ce dernier est défini par des relations de contraintes de types égalité [2,6]

$$g_i(x) = 0 \, , i = 1, \dots, p \qquad \textbf{(2.1)}$$

Et / ou inégalité

$$h_j(x) \leq 0, j = 1, \dots, q \qquad \textbf{(2.2)}$$

L'objectif de l'optimisation globale est de rechercher parmi les $x \in D$, un x particulier qui est appelé le minimum absolu ou global, noté x^*, tel que :

$$f(x^*) \leq f(x), \forall\, x \in D \qquad \textbf{(2.3)}$$

Nous présentons un problème d'optimisation mathématiquement sous la forme suivante :

$$\begin{aligned} &\min_x f(x) \\ \text{Sujet à :}\quad & \\ &g_i(x) = 0\,, i = 1, \dots, p \\ &h_j(x) \leq 0, j = 1, \dots, q \end{aligned} \qquad \textbf{(2.4)}$$

Remarque :

On peut passer d'une formulation minimiser à une formulation maximiser et vice versa : minimiser $f(x)$ revient à maximiser $-f(x)$, c'est-à-dire :

$$\min\ f(x) = -\max[-f(x)] \qquad \textbf{(2.5)}$$

Dans notre étude, on considère les différents résultats et théories relatifs à la formulation $\ll minimiser \gg$

2.3 Classification des problèmes d'optimisation :

Les problèmes d'optimisations sont classés selon la nature de la fonction objectif (ou critère) et des contraintes qui définissent le domaine des solutions admissibles D. les cas les plus courants sont résumés dans le tableau **2.1.**

		Fonction objectif		
		Linéaire	**Quadratique (convexe)**	**Non linéaire**
Contraintes	**Linéaire**	Programmation Linéaire	Programmation Quadratique	Programmation non linéaire
	Quadratique (convexe)	Programmation non linéaire	Programmation convexe	Programmation non linéaire
	Non linéaire	Programmation non linéaire	Programmation non linéaire	Programmation non linéaire

Tableau **2.1 :** classification des problèmes d'optimisation

- Programmation linéaire :

Dans ce cas, la fonction objectif ou le critère est linéaire, c'est-à-dire :

$$f(x) = C^T x \qquad \textbf{(2.6)}$$

(C est un vecteur de coefficient) et l'ensemble des contraintes est un polyèdre convexe fermé (représenté sous la forme $Ax \leq b$, avec A et b sont respectivement une matrice et un vecteur constants).

- Programmation convexe :

Ici, la fonction objectif est convexe, et l'ensemble admissible est convexe.

Un domaine D est convexe si :

$$\forall (x_1, x_2) \in D \; et \; 0 \leq a \leq 1 \Longrightarrow x = ax_1 + (1-a)x_2 \in D \qquad \textbf{(2.7)}$$

Une fonction $f(x)$ est convexe dans un domaine convexe D si :

$$\forall (x_1, x_2) \in D \; et \; 0 \leq a \leq 1 \Longrightarrow f(ax_1 + (1-a)x_2) \leq af(x_1) + (1-a)f(x_2) \qquad \textbf{(2.8)}$$

- **Programmation quadratique :**

C'est un cas particulier de la programmation convexe, il s'agit d'un problème d'optimisation où la fonction objectif est quadratique convexe, c'est-à-dire :

$$f(x) = \frac{1}{2}x^T Qx + C^T x \qquad \textbf{(2.9)}$$

Avec A est une matrice symétrique semi-définie positive et l'ensemble de contraintes est toujours un polyèdre convexe fermé comme dans le cas de programmation linéaire.

- **Programmation non linéaire :**

Toutes les données du problème d'optimisation sont des fonctions différentiables (continues différentiables), c'est-à-dire la fonction objectif est différentiable et les contraintes sont supposées différentiables.

2.4 Condition d'optimalité pour un problème d'optimisation :

Pour calculer le minimum deux hypothèses rencontrées. La première hypothèse concerne le domaine admissible $D \epsilon R^n$ supposé convexe et compact. La deuxième est relative à la fonction objectif ou critère, à minimiser, supposée continue et possède des dérivées partielles premières et secondes, c'est-à-dire

$$\frac{\partial f(x)}{\partial x_i} \; et \; \frac{\partial^2 f(x)}{\partial {x^2}_i}, i = 1, \dots, n \qquad \textbf{(2.10)}$$

Qui sont continues pour tout x .

La fonction $f(x)$ peut être développée en série de Taylor comme suit :

$$f(x) = f(x_0) + (x - x_0)^T \nabla_x f(x_0) + (x - x_0)^T \nabla^2{}_x f(x_0)(x - x_0) + o(3) \qquad \textbf{(2.11)}$$

Avec :

$$x_0 = (x_{1O}, x_{2O}, \dots, x_{n0})^T \qquad \textbf{(2.12)}$$

Sous ces deux hypothèses, les conditions nécessaires et suffisantes pour que x^* soit un minimum local ou global de $f(x)$ sont :

- Condition de premier ordre (condition de stationnarité) relative au gradient de la fonction objectif

$$\nabla_x f(x^*) = 0 \qquad \textbf{(2.13)}$$

La résolution de **(2 .13)** permet de déterminer les points critiques ou stationnaires qui peuvent être la solution du problème.

- Condition de second ordre (condition pour un minimum) relative au HESSIEN de la fonction objectif.

$$\nabla^2{}_x f(x^*) = \left[\left.\frac{\partial^2 f(x)}{\partial x_i \partial x_j}\right|_{x=x^*} ; i,j = 1,\dots,n\right] > 0 \qquad \textbf{(2.14)}$$

La matrice HESSIENNE **(2 .14)** permet de préciser la nature du point critique (min ou max).

Remarque :

- Dans le cas de la recherche d'un minimum, la matrice HESSIENNE **(2 .14)** doit être définie positive, c'est-à-dire soit :
  - Les valeurs propres de la matrice Hessienne sont toutes positives.
  - Les déterminants de la matrice Hessienne pris en chaine sont tous positifs.
- Les deux conditions **(2 .13)** et **(2 .14)** sont nécessaires et suffisantes pour l'optimalité globale.
- La matrice Hessienne nous permet d'étudier la nature des points critiques, le tableau suivant :

Matrice	Commentaire	Nature du point critique
$\nabla^2 f(x^*) > 0$	Définie positive	Minimum
$\nabla^2 f(x^*) < 0$	Définie négative	Maximum
$\nabla^2 f(x^*) \geq 0$	Semi-définie positive	Point singulier (on ne peut pas préciser sa nature)
$\nabla^2 f(x^*) \leq 0$	Semi-définie négative	Point singulier
$\nabla^2 f(x^*) = 0$	Possède des valeurs propres négatives et positives	Point selle

Tableau**2.2 :** Nature de point critique.

2.5 Méthodes de résolution des problèmes d'optimisation :

Afin d'obtenir la solution d'un problème d'optimisation, on doit résoudre un système d'équations algébriques. De ce fait, la difficulté est directement liée à la nature de la fonction objectif $f(x)$. Ainsi, dans le cas où la résolution du système d'équations est facile, on peut obtenir la solution analytiquement. Dans le cas contraire, on procède par des méthodes numériques [2, 6,7].

Pratiquement, les fonctions objectifs à minimiser sont fortement non linéaires. Il résulte que la résolution du système algébrique avec une méthode analytique est quasiment impossible et l'utilisation des méthodes numériques est nécessaire. Ce besoin a donné naissance à un domaine de recherche, faisant partie des mathématiques appliquées, appelé optimisation numérique dont les efforts sont centralisés sur le développement des algorithmes numériques qui permettent de localiser l'optimum d'une fonction en un nombre d'itérations faible [6,7].

La programmation linéaire constitue un des rares cas particuliers pour lequel un algorithme itératif, appelé algorithme de simplexe, a été développé. Cet algorithme converge en un nombre d'itérations fini, et n'impliquent que des calculs assez élémentaires.

Dans le cas d'une programmation convexe, les algorithmes développés sont efficaces et convergent, en général, en une seule itération (par exemple la méthode de Newton).

Dans le cas de la programmation quadratique, la solution analytique est toujours disponible et facile à déterminer.

Pour l'optimisation continue, on sépare sommairement le cas linéaire (qui relève notamment de la programmation linéaire pour lequel la solution est facile à obtenir) du cas non linéaire. Pour certains problèmes qui vérifient la propriété de convexité, la solution peut être obtenue en utilisant une méthode locale qui exploite, ou non, le gradient de la fonction objectif [2]. Néanmoins, la plupart des problèmes d'optimisation sont classés difficiles, car le nombre de minima locaux est très élevés, alors le recours à une méthode globale s'impose.

Pour le traitement des contraintes, on a recours à la méthode des multiplicateurs de Lagrange dans le cas des contraintes du type égalités, et la méthode de Kuhn-Tucker ou la méthode de fonctions d'écarts dans le cas des contraintes inégalités. Notons que la méthode des fonctions de pénalisation constitue une approche élégante pour traiter les deux cas. Cette méthode permet de ramener un problème d'optimisation posé avec contraintes à un problème sans contraintes, puis d'appliquer les méthodes d'optimisation numériques.

En utilisant la méthode de fonction de pénalisation, pour le problème d'optimisation (**2.4**), posé avec contraintes égalités et inégalités, on obtient :

$$\min_x L(x) = f(x) + \sum_{i=1}^{p} \alpha_i [g_i(x)]^2 + \beta_j \left[h_j(x)\right]^2 \qquad \textbf{(2.15)}$$

Où les α_i et $\beta_j$représentent les paramètres de pénalisation, positifs, associés respectivement aux contraintes $g_i(x)$ et $h_j(x)$. Les valeurs de ces paramètres sont en général considérés constantes durant la résolution du problème d'optimisation, c'est-à-dire :

$$\alpha_i = \alpha, i = 1, \dots, p \; et \; \beta_j = \beta, j = 1, \dots, q \qquad \textbf{(2.16)}$$

Dans la relation (**2.15**), la fonction $h_j(x)$ est définie comme suit :

$$h_j(x) = \begin{cases} h_j(x) \; si \; h_j(x) > 0 \\ 0 \qquad si \; h_j(x) \leq 0 \end{cases} \qquad \textbf{(2.17)}$$

2.5.1 Optimisation numérique :

Pour déterminer l'optimum d'une fonction, les méthodes numériques se basent sur la connaissance d'une direction de la recherche donnée souvent par le gradient de la fonction objectif par rapport aux paramètres [8,9]. Elles génèrent une suite $(x_k, k\epsilon N)$ qui converge vers un minimum local x^* de la fonction f suivant le traitement des contraintes utilisées.

L'inconvénient principal des méthodes à base de gradient est que la dérivée de la fonction f n'est pas toujours connue.

On va donc utiliser des méthodes itératives, dont le principe général consiste à calculer à partir d'un estimé de départ de la solution $x_{(0)}$, la suite des valeurs $x^{(1)}, x^{(2)}, \ldots, x^{(k)}, \ldots$d'une façon à avoir :

$$f\left(x^{(0)}\right) > f\left(x^{(1)}\right) > \cdots > f\left(x^{(k)}\right) \qquad \textbf{(2.18)}$$

Comme la suite numérique générée est bornée inférieurement par un minimum $f(x^*)$ alors elle est convergente. Donc on partant de $x(0)$ et on faisant des itérations, on essaye d'approcher l'optimum avec une précision désirée contrôlée par une tolérance ε. Si le minimum global x^*est unique et s'il n'y a pas d'autres minimums local, la méthode converge alors vers le point x^*cherché. Dans le cas contraire seul l'un des minimums locaux est atteint.

Ce dernier cas justifie aussi, le besoin à développer des algorithmes d'optimisation globale.

Puisqu'un algorithme numérique ne peut pas fournir mieux qu'une réponse approchée (ne donne pas une solution exacte), dans le cas de contraintes boîtes, un élément x d'un des ensembles suivants (on fait des itérations pour approcher l'optimum globale jusqu'à ce que la condition d'arrêt soit satisfaite) sera considéré comme solution du problème, x^* est un minimum global et ε est un nombre positif quelconque :

$$A_x(\varepsilon) = \{\, x \in D \;;\; \|x - x^*\| \leq \varepsilon \,\} \quad \textbf{(2.19)}$$

$$A_f(\varepsilon) = \{\, x \in D \;;\; \|f(x) - f(x^*)\| \leq \varepsilon \,\} \quad \textbf{(2.20)}$$

2.5.1.1 La structure générale d'un algorithme d'une méthode numérique :

Le calcul du minimum de la fonction f par une méthode numérique se résume dans l'algorithme suivant [6] :

1- Choisir la valeur initiale $x^{(0)}$ et le seuil de tolérance ε.

2- $x^{(k+1)} = x^{(k)} + \Delta x^{(k)}$.

Avec : $\Delta x^{(k)}$ dépend de la méthode utiliser

3- Test : si $\left\|\nabla_x f\left(x^{(k+1)}\right)\right\| < \varepsilon$, arrêt : $x^{(k+1)}$ est la solution

Si non aller à l'étape (2)

Parmi les méthodes numériques les plus utilisées on trouve :

- La méthode de gradient.
- La méthode de gradient conjugué.
- La méthode de newton.
- La méthode de newton modifiée.

2.5.1.2 La méthode du gradient:

L'algorithme du gradient désigne un algorithme d'optimisation différentiable. Il est par conséquent destiné à minimiser une fonction réelle différentiable définie sur un espace euclidien R^n.

L'algorithme est itératif et procède donc par améliorations successives. Au point courant, un déplacement est effectué dans la direction opposée au gradient, de manière à faire décroitre la fonction. Le déplacement le long de cette direction est déterminé par technique numérique connue sous le nom de recherche linéaire. Cette description montre que l'algorithme fait partie de la famille des algorithmes à directions de descente. Le principe de cette méthode peut être explicité par l'algorithme suivant :

- Choisir la valeur initiale $x^{(0)}$ et le seuil de tolérance $\varepsilon \geq O$. L'algorithme du gradient definit une suite d'itération :

$$\mathrm{x}^{(1)}, \mathrm{x}^{(2)}, \ldots, x^{(k)}, \ldots \quad \textbf{(2.21)}$$

- Jusqu'à ce que le test d'arrêt soit satisfait, il passe de $x^{(k)}$ à $x^{(k+1)}$ par les étapes suivantes :

  1- Calcul de $f(x^{(k)})$ et son gradient $\nabla_x f(x^{(k)})$.

  2- vérification de la convergence en utilisant critère approprié c'est-à-dire :

$$\left\|\nabla f\left(x^{(k)}\right)\right\| \leq \varepsilon \quad \textbf{(2.22)}$$

- Si la convergence est assurée, arrêt et on passe à l'étape 4, sinon on passe à l'étape 3.

  3- Calcul le nouvel itéré par la relation suivante:

$$x^{(k+1)} = x^{(k)} - \alpha_k \nabla f\left(x^{(k)}\right) \quad \textbf{(2.23)}$$

Avec α_k : pas de descente .

puis aller à l'étape 1.

4- L'optimum est : $x^* = x^{(k)}$.

2.5.1.3 Méthode du gradient conjugué :

La méthode de gradient conjugué est une variante améliorée de la méthode de la plus grande pente, qui consiste à suivre la direction opposée au gradient. L'inconvénient de cette méthode est de créer des directions orthogonales, ce qui ralentit la convergence de l'algorithme. Le principe de cette méthode peut être explicité par l'algorithme suivant :

- Choisir la valeur initiale $x^{(0)}$ et le seuil de tolérance $\varepsilon \geq O$. L'algorithme du gradient conjugué définit une suite itérative :

$$\mathrm{x}^{(1)}, \mathrm{x}^{(2)}, \ldots, x^{(k)}, \ldots \quad \textbf{(2.24)}$$

- Jusqu'à ce que le test d'arrêt soit satisfait, il passe de $x^{(k)}$ à $x^{(k+1)}$ par les étapes suivantes :

  1- Calcul de $f(x^{(k)})$ et son gradient noté : $\nabla_x f(x^{(k)})$.

2- vérification de la convergence en utilisant un critère approprié c'est-à-dire :

$$\left\|\nabla f\left(x^{(k)}\right)\right\| \leq \varepsilon \qquad \textbf{(2.25)}$$

- Si la convergence est assurée, arrêt et on passe à l'étape 6, sinon on passe à l'étape 3.

3- Calcul de $\beta^{(k)}$ par la relation suivante:

$$\beta^{(k)} = \begin{cases} 0 \quad si\ k=0 \\ \dfrac{\left\|\nabla_x f\left(x^{(k)}\right)\right\|^2}{\left\|\nabla_x f\left(x^{(k-1)}\right)\right\|^2} \quad si\ k \geq 1 \end{cases} \qquad \textbf{(2.26)}$$

4- Calcul de $p\left(x^{(k)}\right)$ par la relation suivante:

$$p\left(x^{(k)}\right) = -\nabla_x f\left(x^{(k)}\right) + \beta^{(k)} p\left(x^{(k-1)}\right) \qquad \textbf{(2.27)}$$

5- Calcul le nouvel itéré par l'algorithme:

$$x^{(k+1)} = x^{(k)} + \alpha_k p\left(x^{(k)}\right) \qquad \textbf{(2.28)}$$

Avec α_k : pas de descente.

Ensuite aller à l'étape 1.

6- L'optimum est :

$$x^* = x^{(k)} \qquad \textbf{(2.29)}$$

Calcul de pas de descente :

On se déplace toujours dans la direction opposée au gradient $\nabla f\left(x^{(k)}\right)$ mais on va cette fois choisir le pas α_k de façon à maximiser la décroissance de f dans cette direction. En fait le α_k choisi va être la valeur de α qui minimise la fonction :

$$g(a) = f\left(x_k - a\nabla f\left(x^{(k)}\right)\right), a \geq 0 \qquad \textbf{(2.30)}$$

2.5.1.4 Méthode de Newton :

En optimisation numérique, la méthode de newton est, dans son application la plus simple, un algorithme efficace pour trouver numériquement une approximation précise un minimum d'une fonction réelle à n variable. Le principe de cette méthode peut être explicité par l'algorithme suivant :

- Choisir la valeur initiale $x^{(0)}$. L'algorithme de Newton définit une suite itéré :

$$x^{(1)}, x^{(2)}, \dots, x^{(k)}, \dots \quad \textbf{(2.31)}$$

- Jusqu'à ce que le test d'arrêt soit satisfait, il passe de $x^{(k)}$ à $x^{(k+1)}$ par les étapes suivantes :

  1- Calcul de $f(x^{(k)})$ et son gradient : $\nabla_x f(x^{(k)})$.

  2- Calcul la matrice Hessienne de $f(x^{(k)})$: $\nabla_x{}^2 f\left(x^{(k)}\right)$

  Vérifier si :

$$\nabla_x{}^2 f\left(x^{(k)}\right) = 0 \quad \textbf{(2.32)}$$

- Si la condition est vérifiée, arrêt et on passe à l'étape 5 sinon on passe à 3.

  3- Calcul de $p\left(x^{(k+1)}\right)$:

$$p\left(x^{(k+1)}\right) = -\left[\nabla_x{}^2 f\left(x^{(k)}\right)\right]^{-1} \nabla_x f(x^{(k)}) \quad \textbf{(2.33)}$$

  4- Calcul le nouvel itéré:

$$x^{(k+1)} = x^{(k)} + p\left(x^{(k)}\right) \quad \textbf{(2.34)}$$

  puis aller à l'étape 1.

  5- L'optimum est :

$$x^* = x^{(k)} \quad \textbf{(2.35)}$$

2.5.1.5 Méthode de Newton modifiée:

Le principe de cette méthode peut être explicité par l'algorithme suivant :

- Choisir la valeur initiale $x^{(0)}$. L'algorithme de Newton modifiée définit une suite itéré :

$$x^{(1)}, x^{(2)}, \dots, x^{(k)}, \dots \quad \textbf{(2.36)}$$

- Jusqu'à ce que le test d'arrêt soit satisfait, il passe de $x^{(k)}$ à $x^{(k+1)}$ par les étapes suivantes :

1- Calcul de $f(x^{(k)})$ et son gradient noté : $\nabla_x f(x^{(k)})$.

2- Calcul de Hussein de $f(x^{(k)})$ noté : $\nabla_x{}^2 f\left(x^{(k)}\right)$

Test : si

$$\nabla_x{}^2 f\left(x^{(k)}\right) = 0 \quad \textbf{(2.37)}$$

- Si la condition est vérifiée, arrêt et on passe à l'étape 5.

3- Calcul de $p\left(x^{(k+1)}\right)$ par la relation suivante :

$$p\left(x^{(k+1)}\right) = -\left[\nabla_x{}^2 f\left(x^{(k)}\right)\right]^{-1} \nabla_x f(x^{(k)}) \quad \textbf{(2.38)}$$

4- Calcul le nouvel itéré par l'algorithme:

$$x^{(k+1)} = x^{(k)} + \alpha_k p\left(x^{(k)}\right) \quad \textbf{(2.39)}$$

Puis aller à l'étape 1.

5- La solution (l'optimum) est :

$$x^* = x^{(k)} \quad \textbf{(2.40)}$$

- **Détermination de α_k :**

Avant d'entamer l'algorithme de Newton modifié, il faut d'abord calculer le paramètreα_k.

Il existe deux méthodes pour le déterminer :

- **Méthode 1 : Méthode du pas de descente**

On génère des vecteurs $x^{(1)}, x^{(2)}, \dots, x^{(k)}, \dots$ tels que la valeur de la fonction f décroit à chaque itération :

$$f\left(x^{(k+1)}\right) < f\left(x^{(k)}\right) \quad \textbf{(2.41)}$$

On remplace $x^{(k+1)}$ avec sa valeur et on aura :

$$f\left(x^{(k)} + \alpha_k p\left(x^{(k)}\right)\right) < f\left(x^{(k)}\right) \quad \textbf{(2.42)}$$

α_k : Solution de l'équation **(2 .44)**, par conséquent plusieurs valeurs peuvent être considérées Pour α_k.

- **Méthode 2 : Méthode du pas de descente optimal**

Dans ce cas α_k est déterminé en résolvant le problème d'optimisation suivant :

$$\min_{\alpha_k} f\left(x^{(k+1)}\right) = \min_{\alpha_k} f\left(x^{(k)} + \alpha_k p\left(x^{(k)}\right)\right) \quad \textbf{(2.43)}$$

2.5.2 Optimisation globale :

Les méthodes numériques se basent sur les valeurs de la fonction objectif $f\left(x^{(k)}\right)$et des contraintes, ainsi que sur la valeur de leurs dérivées premières et parfois leurs dérivées secondes, ce sont des méthodes itératives qui convergent le plus souvent vers un minimum local. La convergence vers un minimum globale n'est pas toujours assurée. Les méthodes déterministes sont généralement efficaces quand l'évaluation de la fonction objectif est très rapide [8].

Parmi ces méthodes, on trouve la méthode d'Aliénor, qui est une méthode exacte, et les méthodes heuristiques comme la méthode d'essaims de particules et les algorithmes génétiques. Dans notre mémoire, nous allons présenter le principe de la méthode d'Aliénor [2,7].

2.5.2.1 Méthode d'Aliénor :

La méthode Aliénor, proposée par Cherruault (2005), consiste à ramener une fonction multi variable à une fonction d'une seule variable à l'aide d'une transformation réductrice [2,7].

Son principe est le suivant :

Soit $(x, y) \in \mathcal{R}^2$. En coordonnées polaires ce point s'écrit :

$$\begin{matrix} x=r\cos\theta \\ y=\mathrm{r}\sin\theta \end{matrix} \quad \textbf{(2.44)}$$

On peut alors relier r et θ grâce à la spirale d'Archimède d'équation :

$$r = a\theta \quad \textbf{(2.45)}$$

Où a est un paramètre fixe, destiné à tendre vers 0, les relations **(2 .44)** deviennent :

$$\begin{matrix} x=a\theta\cos\theta=h_1(\theta) \\ y=\mathrm{a}\theta\sin\theta=h_2(\theta) \end{matrix} \quad \textbf{(2.46)}$$

Par conséquent, elles permettent d'exprimer x et y en fonction d'une seule variable $a \geq 0$.

Pour trois variables x, y, z, on relie d'abord les deux variables x et y à l'aide d'une spirale d'angle θ_1 ce qui donne :

$$\begin{matrix} x=a\theta_1\cos\theta_1=h_1(\theta) \\ y=a\theta_1\sin\theta_1=h_2(\theta) \end{matrix} \quad \textbf{(2.47)}$$

Puis on relie θ_1 et z à l'aide d'une spirale d'angle θ en posant :

$$\begin{matrix} \theta_1=a\theta\cos\theta \\ z=a\theta\sin\theta=h_3(\theta) \end{matrix} \quad \textbf{(2.48)}$$

Ainsi, on obtient la courbe paramétrée $h(\theta) = \big(h_1(\theta), h_2(\theta), h_3(\theta)\big)$ définie par :

$$\begin{cases} h_1(\theta) = a^2\theta\cos\theta\cos(a\theta\cos\theta) \\ h_2(\theta) = a^2\theta\cos\theta\sin(a\theta\cos\theta) \\ h_3(\theta) = a\theta\sin\theta \end{cases} \quad \textbf{(2.49)}$$

On peut généraliser ce procédé à n variables $x_1, x_2, \dots, x_n$ en les reliant deux à deux par des spirales d'angle θ_i . A la fin de processus on obtient la variable θ qui permet d'exprimer tous les x_i :

$$x_i = h_i(\theta)\,, i = 1, \dots, n \quad ; \quad \theta \geq 0 \quad \textbf{(2.50)}$$

Nous avons approché l'espace $\mathcal{R}^n$ par $\mathcal{R}$ grâce à une transformation réductrice utilisant la spirale d'Archimède. La précision de cette approximation dépend de coefficient a : plus a est petite et la précision est meilleure.

La méthode d'Aliénor peut être exploitée pour la résolution des problèmes d'optimisations de la manière suivante :

$$\min_{x_1,\dots,x_n} f(x_1, \dots, x_n) \quad \textbf{(2.51)}$$

Où f une fonction non linéaire continue.

La transformation **(2.50)** permet de remplacer la fonction $f(x_1, \dots, x_n)$ par la nouvelle fonction $f^*(a)$ donnée comme suit :

$$f^*(a) = f\big(h_1(\theta), \dots, h_n(\theta)\big) \quad \textbf{(2.52)}$$

Qui est une fonction d'une seule variable.
Le problème d'optimisation **(2.51)** est alors remplacé par le problème suivant :

$$\min_{\theta \geq 0} f^*(\theta) \quad \textbf{(2.53)}$$

Qui est un problème d'optimisation à une seule variable que l'on peut résoudre simplement à condition de savoir définir l'intervalle $[0, \theta_{max}]$ sur lequel on va rechercher le minimum globale.

Le succès de la méthode d'Aliénor de base présentée a poussé les chercheurs à développer d'autres transformations réductrices pour simplifier les calculs. Ainsi, plutôt que de réduire deux à deux les variables par la méthode d'Aliénor de base, des transformations réductrices ont été proposées dont l'idée de base consiste à exprimer toutes les variables en fonction de la variable α en une seule étape. Ceci impose à choisir correctement les fonctions h_i de la transformation réductrice **(2.50).** Par conséquent, les efforts se sont focalisés sur un point dont l'objectif est de développer des transformations réductrices simples qui demandent moins de calcul, et surtout de bien approximer le problème d'optimisation à n variables.

Parmi les transformations réductrices proposées dans la littérature, on retrouve les transformations suivantes :

- **Transformation 1 :**

$$x_i = \beta\, \alpha \cos(\rho_i\, \alpha) \quad \textbf{(2.54)}$$

Où les paramètres ρ_i forment une suite croissante et vérifient la condition suivante :

$$\frac{\rho_i}{\rho_{i+1}} < 1 \; et \; \beta > 0 \qquad \textbf{(2.55)}$$

- **Transformation 2 :**

$$x_i = \beta \cos\left(m^i \pi \alpha\right) \qquad \textbf{(2.56)}$$

Avec $m > 1$.

- **Transformation 3 :**

$$x_i = \beta \cos(\rho_i \alpha) \qquad \textbf{(2.57)}$$

Les paramètres $\rho_1, \dots, \rho_{n-1}$ sont choisis proches les uns des autres, tout en constituant une suite lentement croissante, par exemple

$$\rho_{i+1} = \rho_i + \varepsilon \qquad \textbf{(2.58)}$$

Avec $\varepsilon > 0$ et choisi très petit. Le paramètre ρ_n est choisi comme suit :

$$\rho_n = \pi\sqrt{n-1}\frac{\rho_{n-1}}{\beta} \qquad \textbf{(2.59)}$$

- **Transformation 4 :**

$$x_i = \beta \cos(\omega_i \alpha + \varphi_i) \qquad \textbf{(2.60)}$$

Où les $\omega_i > 0$ forment une suite croissante. φ_i est une suite lentement croissante dont les termes sont proches l'un des autres, par exemple

$$\varphi_{i+1} = \varphi_i + \varepsilon \qquad \textbf{(2.61)}$$

Avec $\varepsilon > 0$ et choisi très petit.

Notons que la dernière transformation réductrice $(\mathbf{2}.\mathbf{60})$, permet une exploration globale du plan paramètre (la spirale d'Archimède) par α en assurant une très bonne précision par rapport aux autres de converger rapidement vers l'optimum global.

Aussi d'autres variantes ont été proposées pour lier r et α afin d'améliorer la précision de l'approximation. Parmi ces relations, on peut citer les suivantes (Bendiab et Cherruault,1995) :

$$\begin{matrix} r= e^{-\beta\,\alpha} \\ r=e^{-\beta\,\alpha^2+b\,\alpha}, b>0 \end{matrix} \qquad \textbf{(2.62)}$$

Pour illustrer la transformation de la fonction à plusieurs variables à une fonction à une seule variable $f^*(\alpha)$, considérons le problème d'optimisation à deux variables de décision, suivant :

$$\min_x \cos(x_1)\sin({x_1}^2 - x_1) + \cos(x_2)\sin({x_2}^2 - x_2)$$

Sujet à : **(2.63)**

$$\begin{matrix} -10 \le x_1 \\ x_2 \le 10 \end{matrix}$$

Pour ce faire, on propose d'utiliser la transformation 4 donnée par la relation $(\mathbf{2}.\mathbf{60})$**.** Pour respecter les contraintes, il suffit de prendre $\beta = 10, \omega_1 = 100, 2 = 101, \varphi_1 = 1,$ et $\varepsilon = 0.0001$. Ce qui donne :

$$\begin{matrix} x_1=10\,\cos(100\,\alpha+1) \\ x_1=10\,\cos(101\,\alpha+1.00001) \end{matrix} \qquad \textbf{(2.64)}$$

Ce qui donne la fonction suivante :

$$f^*(\alpha) = \cos(10\,\cos(100\,\alpha+1))\sin((10\,\cos(100\,\alpha+1))^2 - 10\,\cos(100\,\alpha+1)) + \cos(10\,\cos(101\,\alpha+1.00001))\sin((10\,\cos(101\,\alpha+1.00001))^2 - 10\,\cos(100\,\alpha+1.00001)) \qquad \textbf{(2.65)}$$

En résolvant l'équation algébrique suivante :

$$\nabla_\alpha f^*(\alpha) = 0 \qquad \textbf{(2.66)}$$

On détermine facilement le minimum global α^* . Ensuite à partir de la transformation$(\mathbf{2}.\mathbf{64})$, on determine la solution globale comme suit :

$$x_1{}^*=10\ \cos(100\,\alpha^*+1)$$
$$x_1{}^*=10\ \cos(101\,\alpha^*+1.00001) \qquad \textbf{(2.67)}$$

2.6 Optimisation globale et commande prédictive :

Dans un problème de commande prédictive, l'étape de l'optimisation est importante pour réussir de bonnes performances. La nature du problème d'optimisation dépend de trois paramètres : la nature du système, la nature du critère et les contraintes considérées. Pour représenter avec précision les phénomènes et les grandeurs physiques de notre système, on utilise souvent des modèles ou des fonctions de nature non linéaire [2,3]. Ces modèles non linéaires constituent un moyen très efficace pour approcher le mieux le système à étudier. Néanmoins, la manipulation des non linéarités pose problème. En effet, leur présence demande plus d'efforts de calcul. La complexité s'accentue d'autant plus avec la présence de contraintes. D'autant plus que le nombre d'extremums devient plus important avec l'augmentation du nombre des variables de décision.

Dans le cas d'un modèle linéaire, comme l'algorithme Dynamic Matrix Control présenté précédemment, une solution analytique du problème d'optimisation peut être obtenue. L'implémentation ne pose pas de problème puisque la solution est calculée rapidement [2,4]. Cependant, dans la majorité des cas, bien que le critère de performance soit quadratique, ce qui représente une forme de fonction objectif intéressante pour localiser la solution globale, néanmoins, la non linéarité du problème d'optimisation vient de la nature du modèle utilisé pour la prédiction. Par conséquent, les prédictions seront non linéaires. De ce fait, le critère prend une forme non linéaire dont la complexité dépend directement des non linéarités du système. Ainsi, le problème d'optimisation est non linéaire, donc l'obtention de la solution à chaque instant d'échantillonnage est une tâche très délicate, et nécessite des méthodes numériques qui convergent en un nombre fini d'itérations et rapidement vers la solution optimale.

L'utilisation d'un modèle non linéaire pour la prédiction des sorties du système à commander est impératif du fait que la quasi-totalité des systèmes sont de nature non

linéaires. De plus, l'utilisation d'un modèle non linéaire permet d'accroître nettement la précision de la prédiction ce qui implique une amélioration des performances du système en boucle fermée. Néanmoins, la prédiction non linéaire complexifie l'application de la commande prédictive à des systèmes non linéaires puisque le problème d'optimisation à résoudre est de nature fortement non linéaire dont la difficulté de localiser l'optimum globale, en un nombre fini d'itérations ou en un temps inférieur à la période d'échantillonnage, se présente avec acuité [2,3,8]. Il n'est pas complètement faux d'appliquer une solution locale dans une stratégie de commande prédictive, cependant l'application d'une solution globale donne de meilleures performances puisque minimiser mieux le critère implique une bonne poursuite de la consigne, un bon rejet de perturbations et un minimum d'énergie (objectifs souvent visés par l'utilisation d'un critère quadratique). Par conséquent, l'utilisation des méthodes d'optimisation globale dans une commande prédictive non linéaire constitue une solution intéressante qui permettra d'élargir son champ d'application.

2.7 Conclusion :

Ce chapitre est consacré à des généralités sur les fonctions mathématiques, en particulier l'optimisation numérique et l'optimisation globale. Après avoir présenté comment formuler mathématiquement les problèmes d'optimisation et les avoir classifiés selon la fonction objectif et leurs contraintes, on a proposé les différentes méthodes numériques de résolution des problèmes d'optimisation. Par la suite on a parlé des inconvénients de ces méthodes numériques et l'intérêt des méthodes d'optimisation globale. Dans ce mémoire on s'est intéressé à la méthode d'Aliénor qui fait partie des méthodes d'optimisation exacte.

Chapitre 3

3.1 Introduction

La commande prédictive est un ensemble d'algorithmes qui utilisent le modèle du système afin de résoudre un problème d'optimisation qui consiste à minimiser un critère de performances [2,3]. La résolution du problème d'optimisation permet de calculer une séquence de valeurs pour la variable manipulée sur un horizon appelé horizon de commande où seule la première commande sera appliquée au système.

Pour la commande des systèmes linéaires, le problème revient à résoudre un problème de programmation quadratique, comme dans l'exemple de la méthode DMC (Dynamic Matrix Control) [2,4]. Par contre, pour les systèmes non linéaires, le problème est plus complexe étant donné que le problème d'optimisation est souvent non linéaire à cause de la prédiction. Par conséquent, l'optimum global est très difficile à obtenir.

Comme la résolution du problème d'optimisation doit être réalisée pendant une durée égale à une période d'échantillonnage, alors la commande prédictive des systèmes non linéaires nécessite des algorithmes d'optimisation globale qui doivent être convergeant et rapides[2,7,8].

Il existe une grande quantité d'algorithmes dans la théorie d'optimisation globale. Cependant, dans une stratégie de commande prédictive, la condition du temps de convergence qui ne doit pas dépasser une période d'échantillonnage doit être respectée. Par conséquent, leur application à des systèmes rapides et des systèmes fortement non linéaires est limitée.

Dans ce chapitre, nous allons utiliser une méthode d'optimisation globale, en l'occurrence la méthode d'Aliénor pour la commande prédictive d'un système non linéaire, en considérant des contraintes boîte pour la variable de commande.

3.2 Commande prédictive non linéaire :

La commande prédictive non-linéaire est appliquée aux systèmes non-linéaires, et est basée sur les prédictions du modèle du système non-linéaire. Son principe repose sur la résolution d'un problème d'optimisation à chaque pas d'échantillonnage [2,3]. Ce dernier consiste à minimiser un critère sur un horizon de prédiction fini. Généralement, ce critère prend la forme quadratique, mais comme le système est non-linéaire, alors on obtient un problème d'optimisation non linéaire.

Le principe de la commande prédictive non linéaire est résumé par la figure (**3. 1**)

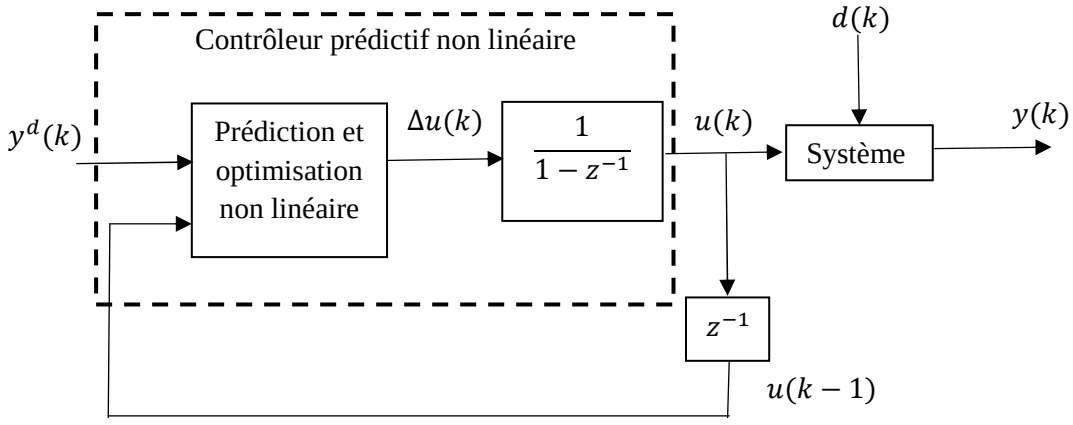


Figure 3.1 : Structure générale de la commande prédictive non linéaire.

Le modèle du système est utilisé pour la prédiction du comportement futur du système. Il existe plusieurs formes de modèle dans le cas des systèmes non-linéaires : modèle entrée-sortie (équation aux différences), modèle Volterra et modèle réseaux de neurones.
Dans notre étude, on considère le modèle entrée-sortie donnée par l'équation générale :

$$y(k) = \Phi\left(y(k-1), \dots, y\left(k-n_y\right), u(k-1), \dots, u(k-n_u), d(k-1), \dots, d(k-n_d)\right) \qquad \textbf{(3.1)}$$

Où y est la sortie du système, u est la variable de commande, d est la perturbation et Φ est une fonction non linéaire.
Le critère à minimiser englobe généralement deux objectifs :

- La poursuite.
- L'énergie minimale.

$$J(U) = \sum_{j=0}^{N_p} \left[y^d(k+j) - \hat{y}\left(k+\frac{j}{k}\right)\right]^T Q_j \left[y^d(k+j) - \hat{y}\left(k+\frac{j}{k}\right)\right] + \sum_{j=0}^{N_u-1} [\Delta u(k+j)]^T R_j [\Delta u(k+j)] \qquad \textbf{(3.2)}$$

Avec :

- N_p et N_u représentent respectivement les horizons de prédiction et de commande.
- y^d représente la consigne ou la trajectoire désirée.

- $\hat{y}$ représente la sortie prédite par le modèle **(3.1)**.
- Q et R sont des matrices de pondération qui peuvent être constantes ou variables.
- $k+j$ représente les instants de prédictions et k l'instant actuel.
- $\hat{y}(k+j/k)$ représente la sortie prédite à l'instant d'échantillonnage k pour le futur instant $k+j$.
- U représente le vecteur regroupant la séquence des commandes à appliquer aux instants d'échantillonnage dont le nombre est égale à N_u, c'est-à-dire :

$$U = [u(k), u(k+1), \dots, u(k+N_u-1)] \qquad \textbf{(3.3)}$$

Notons que seule la commande $u(k)$ sera appliquée au système à l'instant k.
Pour déterminer cette commande $u(k)$, on doit résoudre, en un temps égal à la période d'échantillonnage, le problème d'optimisation non linéaire suivant :

$$\min_u J(U) = \sum_{j=0}^{N_p} [y^d(k+j) - \hat{y}(k+j/k)]^T Q_j [y^d(k+j) - \hat{y}(k+j/k)] + \sum_{j=0}^{N_u-1} [\Delta u(k+j)]^T R_j [\Delta u(k+j)]$$

Sujet à :

$$y(k) = \Phi\left(y(k-1), \dots, y(k-n_y), u(k-1), \dots, u(k-n_u), d(k-1), \dots, d(k-n_d+1)\right) \quad \text{avec } |u(k)| \le M \qquad \textbf{(3.4)}$$

La minimisation du critère $J(U)$ permet de localiser la solution, c'est-à-dire l'optimum global. Cependant, comme le principe de la commande prédictive l'exige, ce calcul doit être effectué en un temps égal à la période d'échantillonnage, ce qui est pratiquement impossible étant donné que le problème d'optimisation est fortement non-linéaire.
Afin de remédier à cette difficulté, des algorithmes d'optimisation ont été développés.

3.3 La méthode Aliénor appliquée sur la commande prédictive non linéaire :

Dans cette section, nous allons appliquer la méthode Aliénor pour une stratégie de commande prédictive. Le vecteur de variables de décisions, donné par la relation **(3.3)**, est

de dimension N_u, il prend les valeurs de la commande à des instants d'échantillonnage allant de k jusqu'à $k + N_u - 1$ avec $N_u < N_p$. Pour les valeurs allant de $k + N_u$ jusqu'à $k + N_p$ on maintient la commande constante à une valeur égale à $u(k + N_u - 1)$. C'est-à-dire : $u(k + N_u - 1) = u(k + N_u) = \cdots = u\big(k + N_p\big)$

Nous avons expliqué précédemment qu'il était impératif que la résolution du problème d'optimisation s'effectue en un temps égal à une période d'échantillonnage, et nous avons vu qu'il était très difficile d'y parvenir vu la complexité des calculs, ajouter à ça : les deux problèmes suivants [2,7]:

- La convergence est lente lorsque l'horizon de commande est grand.
- La plupart des algorithmes d'optimisation, généralement numériques, peuvent être piégés dans un minimum local au lieu d'un minimum global ce qui n'assure pas de bonnes performances.

Grâce à son principe de réduire un problème d'optimisation à une seule variable, la méthode d'optimisation d'Aliénor permet de simplifier le problème **(3.4)** en réduisant les calculs. Le problème simplifié permettra d'avoir des prédictions sur les sorties $\hat{y}(k + j/k)$ en fonction des variables de commande $u(k), \dots, u(k + N_u)$

Par conséquent, le critère de performance **(3.2)** prend la forme suivante :

$$\hat{J}(U) = F\big(u(k), \dots, u(k + N_u - 1)\big) \qquad \textbf{(3.5)}$$

Et le problème revient à rechercher, à chaque instant d'échantillonnage k, la solution de problème d'optimisation équivalent suivant :

$$\min_{u(k),\dots,u(k+N_u)} \hat{J}\,(U) = F\big(u(k), \dots, u(k + N_u - 1)\big) \qquad \textbf{(3.6)}$$

C'est pour simplifier la résolution qu'on propose d'utiliser la méthode d'Aliénor.

Dans cette optique, chaque variable de décision $u(i)$ pour $i = k, \dots, k + N_u - 1$ sera définie par une transformation réductrice

$$u(i) = h_{i-k+1}(a) \qquad \textbf{(3.7)}$$

Et le problème d'optimisation (**3.6)** prend la forme suivante :

$$\min_a G(a) = F\left(h_1(a), \dots, h_{N_u}(a)\right) \quad \textbf{(3.8)}$$

On voit bien que le problème d'optimisation **(3.8)** obtenu, est à une seule variable. La suite du travail est facile, c'est-à-dire, l'obtention de l'optimum global a^* à chaque instant d'échantillonnage k.

D'après la relation **(3.7)**, la commande à appliquer à chaque instant d'échantillonnage k est donnée comme suit :

$$u(k) = h_1(a^*) \quad \textbf{(3.9)}$$

3.4 Exemple d'application:

On remarque bien que la méthode d'Aliénor facilite l'implémentation de la commande prédictive pour un système non linéaire.

- **Système non linéaire sans contrainte :**

L'approche proposée sera illustrée pour la commande du système non linéaire suivant :

$$y(k) = 0.9y(k-1) - 0.7y(k-2)u(k-1) + 0.5u(k-1) \quad \textbf{(3.10)}$$

L'objectif est de concevoir un correcteur prédictif pour assurer une poursuite de consigne y^d, supposée constante, ce que revient à minimiser le critère suivant (Q = 1 et $R = 0$ dans le critère **(3.2)**) :

$$J = \sum_{j=1}^{N_p}\left(y^d(k) - \hat{y}(k+j/k)\right)^2 \quad \textbf{(3.11)}$$

Ainsi, le problème d'optimisation à résoudre à chaque instant d'échantillonnage k est formulé comme suit :

$$\min_u J(U) = \sum_{j=1}^{N_p}\left(y^d(k) - \hat{y}(k+j/k)\right)^2 \quad \textbf{(3.12)}$$

Les prédictions $\hat{y}(k+j/k)$ sont données par le modèle **(3.10)**:

$$y(k+j/k) = 0.9\hat{y}(k+j-1/k) - 0.7\hat{y}(k+j-2/k)u(k+j-1/k) + 0.5u(k+j-1/k) \quad \textbf{(3.13)}$$

En considérant $N_p = 7$ et $N_u = 4$, le problème d'optimisation s'écrit :

$$\min_U \sum_{j=1}^{7} \Big(y^d(k) - 0.9\hat{y}(k+j-1/k) + 0.7\hat{y}(k+j-2/k)u(k+j-1/k) - 0.5u(k+j-1/k) \Big)^2 \quad \textbf{(3.14)}$$

Pour résoudre le problème d'optimisation **(3.14),** par la méthode d'Aliénor, on considère la transformation réductrice **(2.60)**. Ainsi :

$$u(i) = \beta \cos(w_{i-k+1}\alpha + \varphi_{i-k+1}), \quad i = 1, \dots, 4 \quad \textbf{(3.15)}$$

En substituant **(3.15)** dans les prédictions données par la relation de récurrence **(3.13),** puis dans le problème d'optimisation **(3.14),** on obtient le problème d'optimisation à une seule variable α suivant :

$$\min_a \sum_{j=1}^{7} \Big(y^d - \psi(\alpha) \Big)^2 \quad \textbf{(3.16)}$$

Où Ψ une fonction non linéaire à une seule variable α. Par conséquent, la commande à appliquer à chaque instant k est déduite de **(3.15)** comme suit :

$$u(k) = \beta \cos(w_i a^* + \varphi_i) \quad \textbf{(3.17)}$$

Où a^* est l'optimum global du problème d'optimisation **(3.16)** calculé à chaque instant d'échantillonnage k. La période considérée comme est de 20 s.

Les paramètres utilisés pour la transformation réductrice **(3.15)** sont :

$$\beta = 2, w_i = 100 + (i-1), \varphi_i = 1 + 0{,}00005(i-1); i = 1, \dots, 4 \qquad \textbf{(3.18)}$$

Rappelons que le but recherché par commande est d'assurer la poursuite d'une consigne constante $y^d = 1$.

Les résultats de simulations sont donnés par la figure **(3.2)**. D'après la figure **(Fig.3.2.a)** la sortie converge rapidement et parfaitement vers la consigne injectée. Sur la figure **(Fig.3.2.b)** on remarque de petites variations acceptables sur la commande.

- **Système non linéaire avec contrainte :**

On refait le même exemple précédent, on considère que la variable de commande $u(k)$ est soumise à la contrainte suivante :

$$|u(k)| \leq 1{,}25 \qquad \textbf{(3.19)}$$

Dans ce cas, il suffit de jouer sur le réglage des paramètres de la transformation réductrice $(\mathbf{3.15})$**.** Ainsi pour respecter la contrainte $(\mathbf{3.19})$, il faut prendre

$$\beta = 1{,}25 \qquad \textbf{(3.20)}$$

Les résultats de simulations sont donnés par la figure **(3.3)**. Sur la figure **(Fig.3.3.a)** On voit que la sortie suit la consigne imposée après un certain nombre de période d'échantillonnage tout en respectant les limites (contraintes) imposées sur la commande **(Fig.3.3.b)**.

La comparaison entre les deux figures **(Fig.3.2.b)** et **(Fig.3.3.b)** montre que pour une même consigne désirée $y^d = 1$, la commande générée dépasse la valeur 1,25 pour le cas sans contraintes, alors que lorsqu'on impose des contraintes, la commande générée n'atteint pas la valeur 1,25.

- **Système linéaire sans contrainte :**

L'approche proposée sera illustrée pour la commande du système linéaire suivant :

$$y(k) = 0.5y(k-1) - u(k-1) \quad \textbf{(3.21)}$$

L'objectif est de concevoir un correcteur prédictif pour assurer une poursuite de consigne y^d, supposée constante, ce que revient à minimiser le critère suivant ($= 1$ et $R = 0$ dans le critère $(\mathbf{3.2})$) :

$$J = \sum_{j=1}^{N_p} \left(y^d(k) - \hat{y}(k+j/k) \right)^2 \quad \textbf{(3.22)}$$

Ainsi, le problème d'optimisation à résoudre à chaque instant d'échantillonnage k est formulé comme suit :

$$\min_u J(U) = \sum_{j=1}^{N_p} \left(y^d(k) - \hat{y}(k+j/k) \right)^2 \quad \textbf{(3.23)}$$

Les prédictions $\hat{y}(k+j/k)$ sont données par le modèle **(3.21)**:

$$y(k+j/k) = 0.5\hat{y}(k+j-1/k) - u(k+j-1/k) \quad \textbf{(3.24)}$$

En considérant $N_p = 4$ et $N_u = 2$, le problème d'optimisation s'écrit :

$$\min_U \sum_{j=1}^{4} \left(y^d(k) - 0.5\hat{y}(k+j-1/k) + 0.5u(k+j-1/k) \right)^2 \quad \textbf{(3.25)}$$

Pour résoudre le problème d'optimisation **(3.25)** par la méthode d'Aliénor en considère la transformation réductrice **(2.41)**. Ainsi :

$$u(i) = \beta \cos(w_{i-k+1}\alpha + \varphi_{i-k+1}), \;\; i = 1 \; et \; 2 \quad \textbf{(3.26)}$$

En substituant **(3.26)** dans les prédictions données par la relation de récurrence **(3.24)** puis dans le problème d'optimisation (**3.25),** on obtient le problème d'optimisation à une seule variable α suivant :

$$\min_{a} \sum_{j=1}^{4} \left(y^{d} - \psi(\alpha)\right)^{2} \qquad \textbf{(3.27)}$$

Où Ψ une fonction non linéaire à une seule variable α. Par conséquent, la commande à appliquer à chaque instant k est déduite de **3.26)** comme suit :

$$u(k) = \beta \cos(w_i a^{*} + \varphi_i) \qquad \textbf{(3.28)}$$

Où a^{*} est l'optimum global du problème d'optimisation (**3.27)** calculé à chaque instant d'échantillonnage k. La période considérée comme est de $20\ s$.
Les paramètres utilisés pour la transformation réductrice **(3.26)** sont :

$$\beta = 2, w_i = 100 + (i-1), \varphi_i = 1 + 0{,}00005(i-1); i = 1\ et\ 2 \qquad \textbf{(3.29)}$$

Les résultats de simulations obtenus sont donnés par la figure (**3.2**). D'après la figure (**Fig.3.4.a**) la sortie atteint sa consigne imposée. Sur la figure (**Fig.3.4.b**) on constate que l'évolution de la variable de commande est physiquement acceptable.

- **Système linéaire avec contrainte :**

On refait le même exemple précédent, on considère que la variable de commande $u(k)$ est soumise à la contrainte suivante :

$$|u(k)| \leq 1{,}25 \qquad \textbf{(3.30)}$$

Ensuite, on doit choisir convenablement le paramètre β de la transformation (**3.26)**, la valeur à choisir devra permettre de respecter la contrainte **(3.30),** on prendra :

$$\beta = 1{,}25 \qquad \textbf{(3.31)}$$

Les résultats de simulations sont donnés par la figure (**3.3**). Concernant l'évolution de la sortie, on remarque la même chose que pour le cas sans contraintes ; la sortie atteint la consigne imposée après un certain nombre de périodes d'échantillonnage sur la figure (**Fig.3.5.a**) et la commande reste dans son domaine admissible défini par la contrainte (**3.30**) (**Fig.3.5.b**).

On faisant la comparaison entre les deux figures (**Fig.3.4.b**) et (**Fig.3.5.b**) on note le même constat qu'avec la méthode non linéaire. Pour la même consigne désirée $y^d = 1$, la commande générée aura des valeurs supérieures à 1,25 pour le cas sans contraintes, alors que lorsqu'on impose des contraintes, les valeurs la commande générée sont inférieures à la contrainte imposée 1,25.

Remarque :

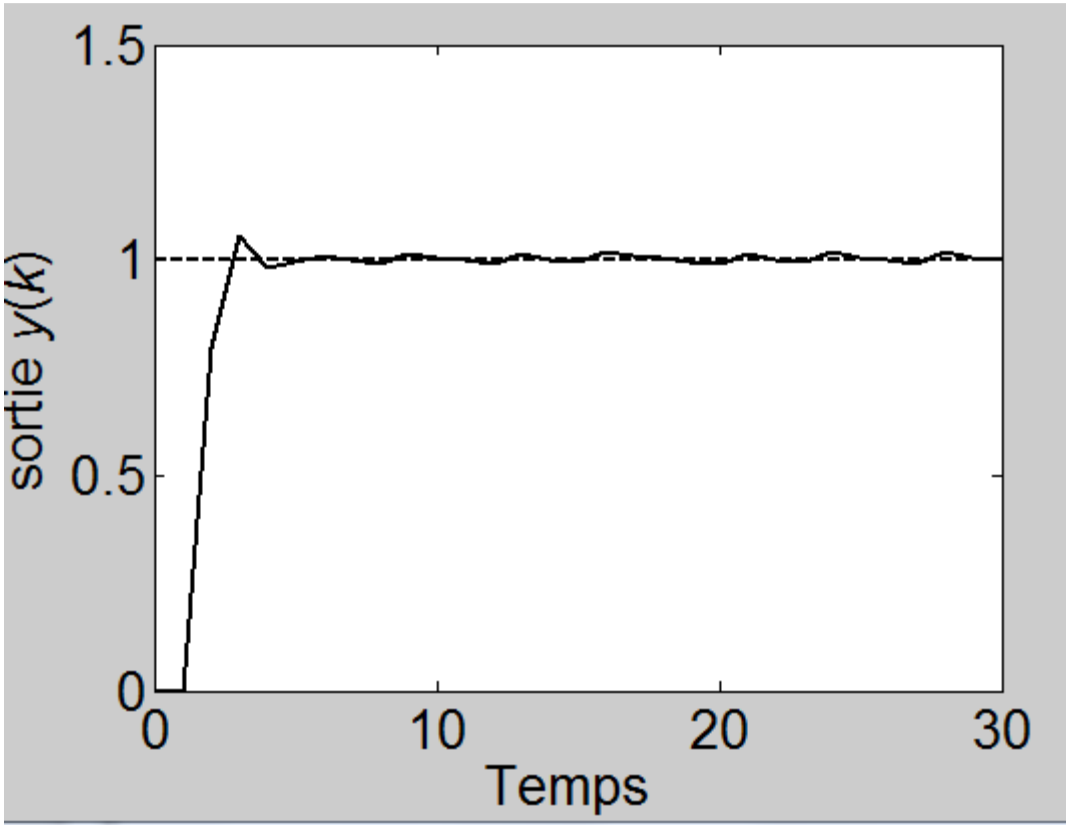
L'avantage de la méthode d'Aliénor permet de prendre en considération les contraintes sur la commande sans transformer le problème d'optimisation par introduction des variables d'écarts ou l'utilisation des conditions de Kuhn-Tucker.

3.5 Conclusion :

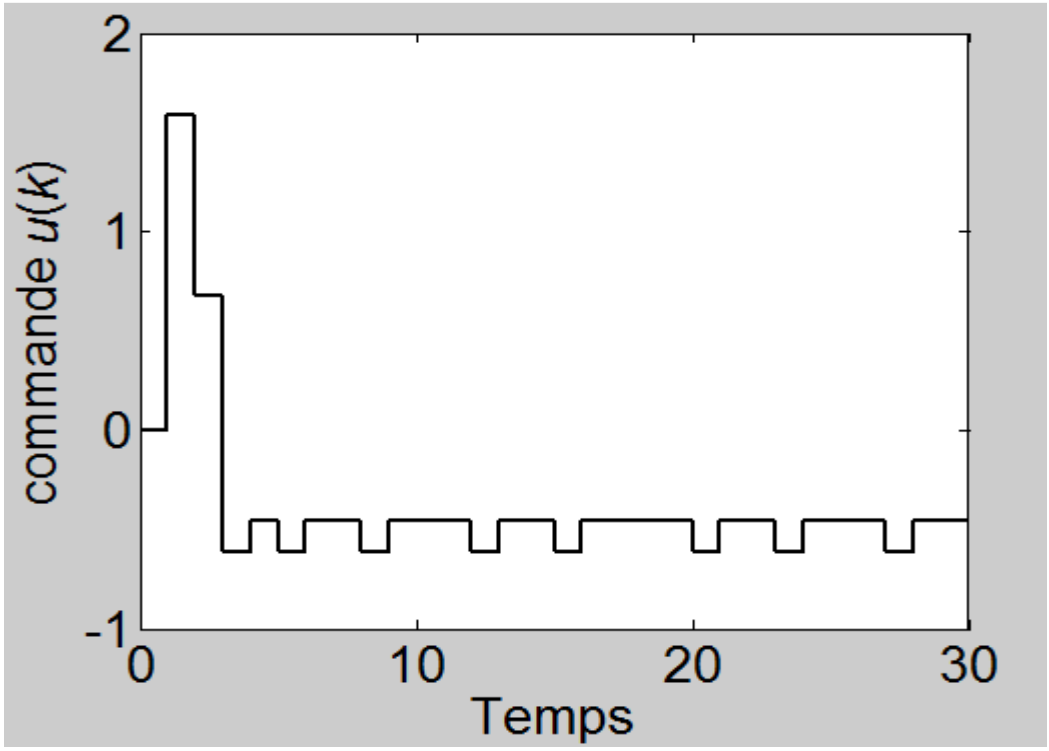
Dans ce chapitre, nous avons utilisé une méthode d'optimisation globale, en l'occurrence la méthode d'Aliénor, dans une commande prédictive d'un système non linéaire, pour la résolution d'un problème d'optimisation. L'étude s'est portée sur les deux cas avec et sans contraintes boîte sur la variable de commande. L'objectif qui consiste à assurer la poursuite d'une consigne constante, passe par la minimisation d'un critère.

Dans les deux cas d'étude, avec ou sans contrainte, la méthode d'Aliénor permet d'assurer la poursuite de la consigne avec un bon régime transitoire, ceci s'explique par le fait que la convergence est exacte avec la méthode d'Aliénor. Pour les commandes générées, on remarque de bons résultats avec la présence de petites fluctuations physiques admissibles.

En résumé, l'exemple étudié montre clairement que la commande prédictive non linéaire basée sur l'optimisation globale donne de bons résultats quand on utilise la méthode d'Aliénor qui assure une convergence exacte.

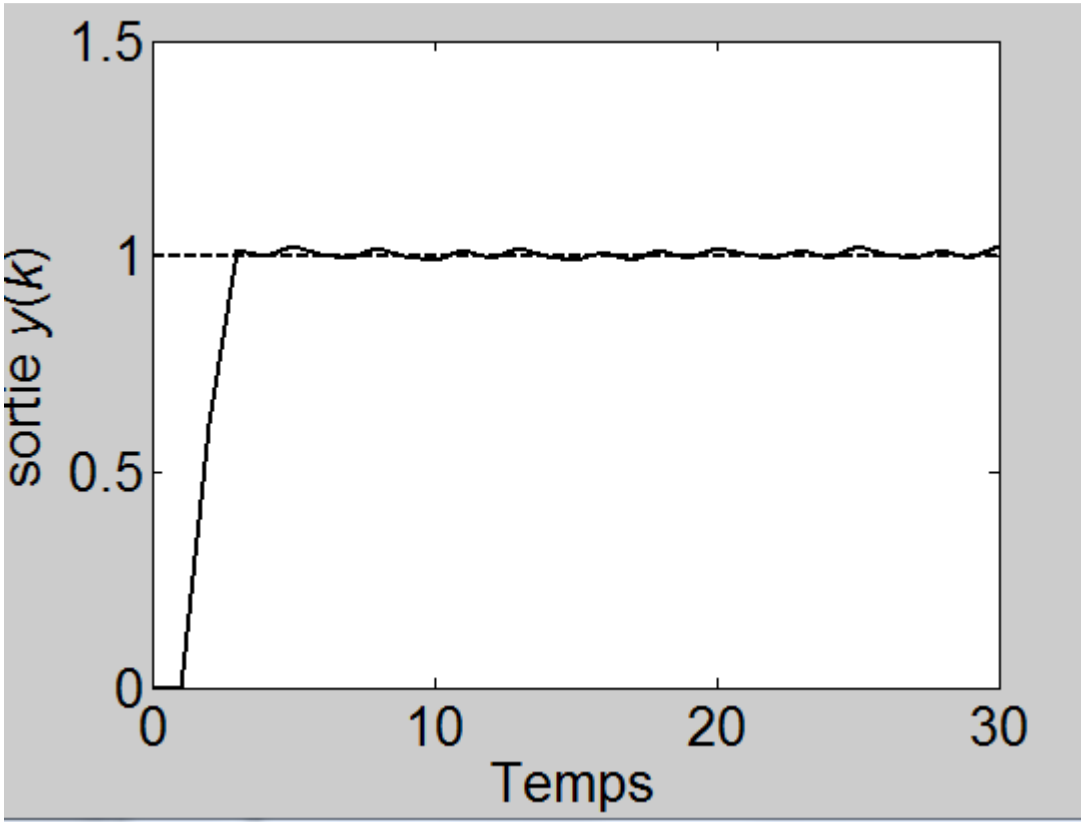


(a) Evolution de la sortie commandée $y(k)$.

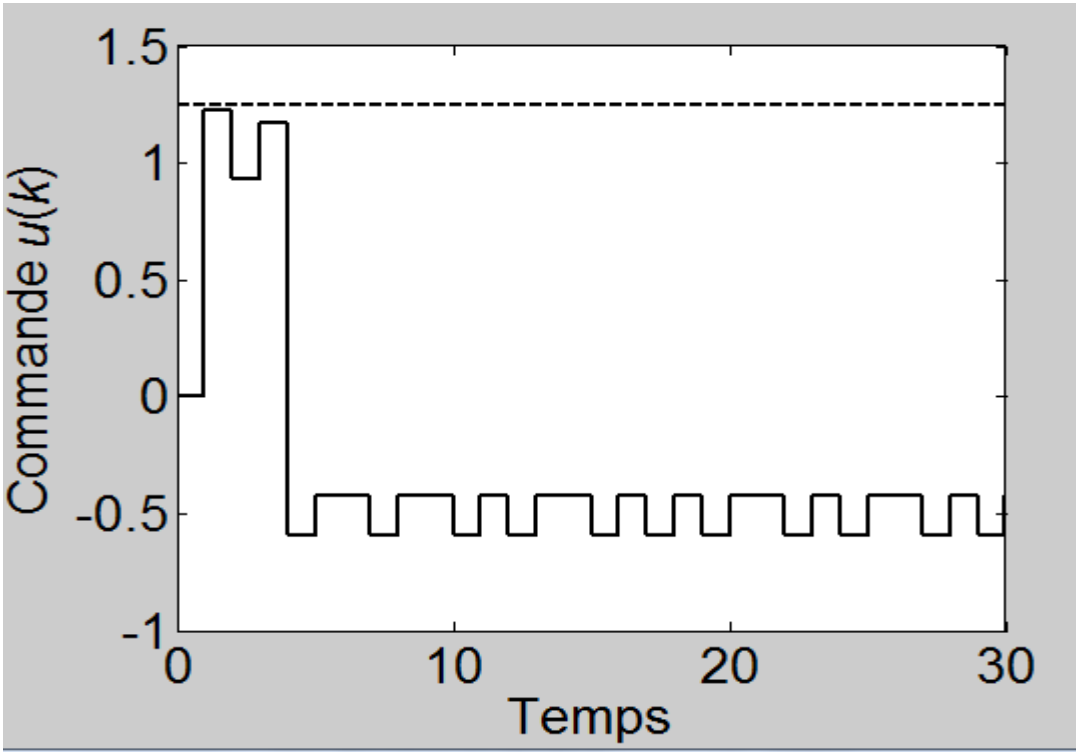


(b) Evolution de la commande $u(k)$.

Figure 3. 2 : méthode d'Aliénor : cas non linéaire sans contrainte sur la variable de commande $u(k)$.

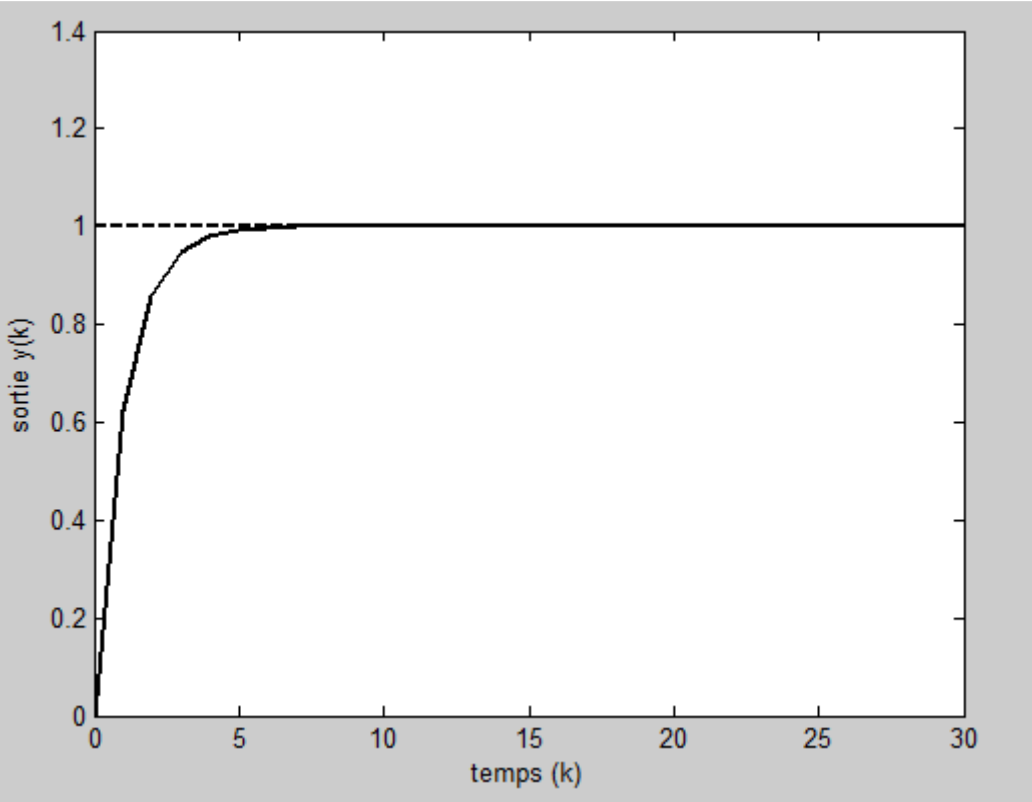


(a) Evolution de la sortie commandée $y(k)$.

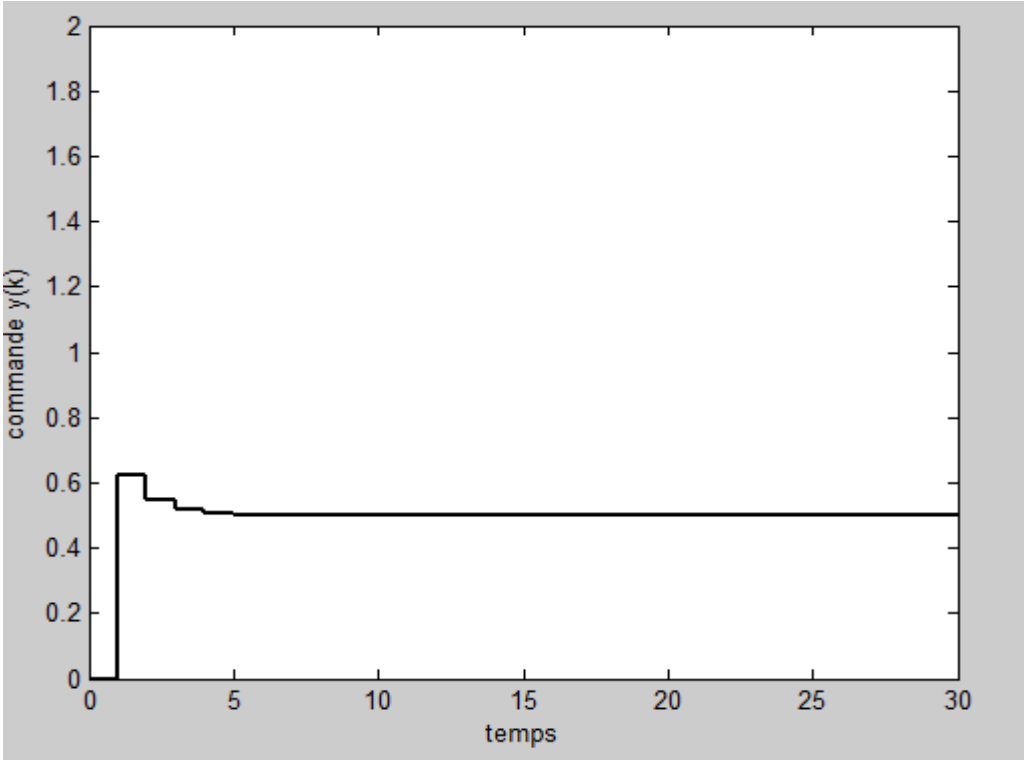


(b) Evolution de la commande $u(k)$.

Figure 3. 3 : méthode d'Aliénor : cas non linéaire avec contrainte sur la variable de commande $u(k)$.

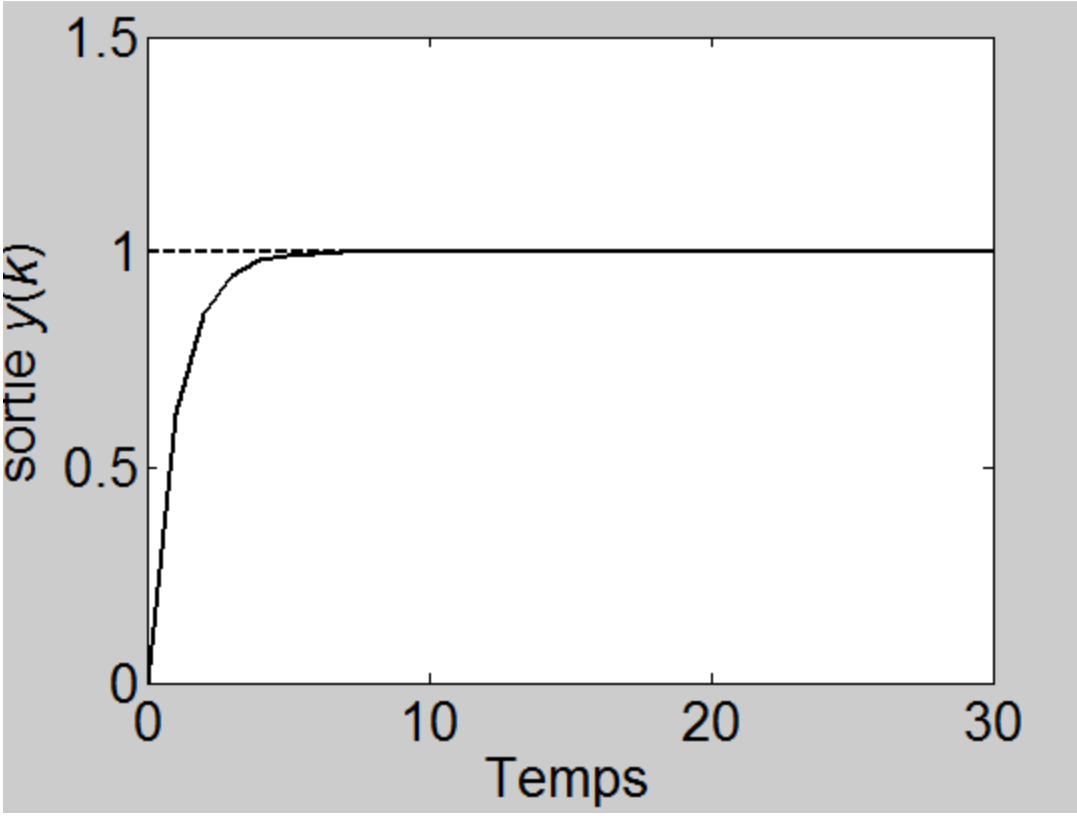


(a) Evolution de la sortie commandée $y(k)$.

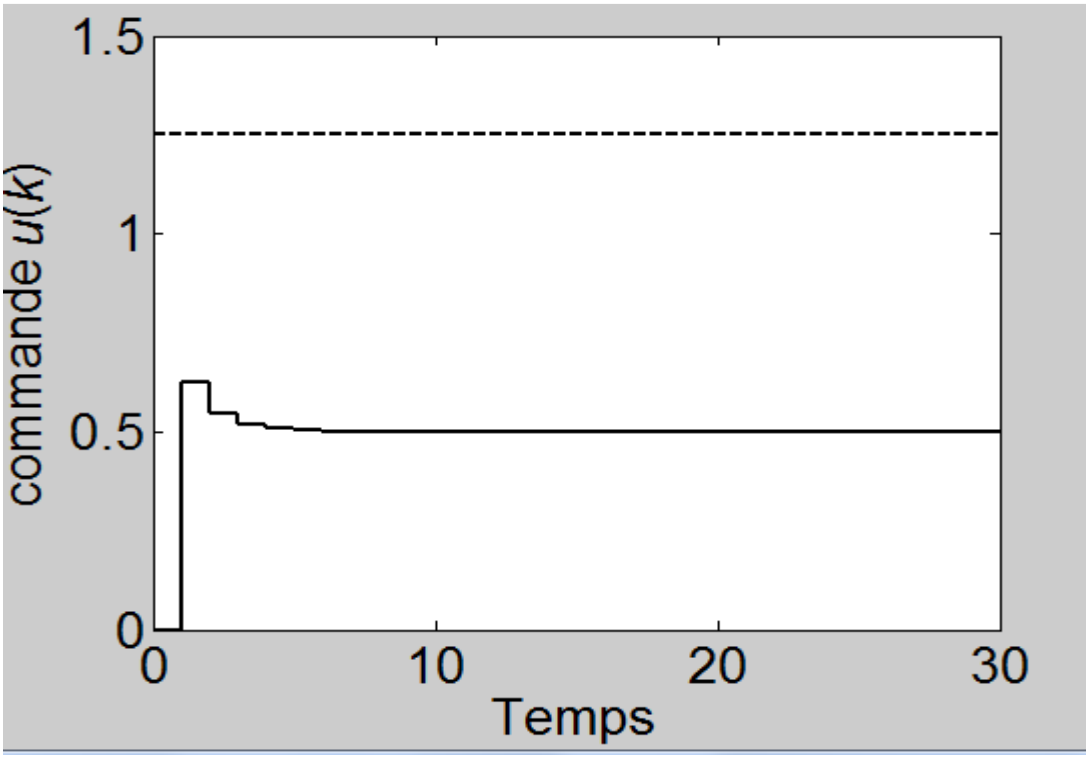


(b) Evolution de la commande $u(k)$.

Figure 3. 4 : méthode d'Aliénor : cas linéaire sans contrainte sur la variable de commande $u(k)$.



(a) Evolution de la sortie commandée $y(k)$.



(b) Evolution de la commande $u(k)$.

Figure 3.5 : méthode d'Aliénor : cas linéaire avec contrainte sur la variable de commande $u(k)$.

Conclusion Générale

L’étude sur la commande prédictive, réalisée dans ce mémoire, consiste à utiliser la méthode d’optimisation globale d’Aliénor dans une stratégie de commande prédictive d’un système non linéaire afin de trouver la commande optimale à appliquer au système.

On a entamé notre travail par une introduction sur la commande prédictive où on a présenté le principe de la commande prédictive, nous avons détaillé les éléments et les paramètres de réglage d’un algorithme d’une commande prédictive. Dans le cas linéaire, nous avons présenté l’algorithme DMC (Dynamic Matrix Control) qui permet de déterminer analytiquement la commande prédictive.

La suite du travail a été consacrée à l’optimisation d’une manière générale et en particulier à l’optimisation globale. Nous avons utilisé l’une des méthodes d’optimisation globale en l’occurrence la méthode d’Aliénor qui permet d’atteindre l’optimum globale, par conséquent d’avoir la meilleur commande à appliquer au système.

A la fin du mémoire, nous avons présenté les résultats de la commande prédictive, basée sur la méthode d’Aliénor, dans le cas d’un système non linéaire en considérant les cas sans et avec contrainte sur la variable de commande. Les résultats de simulation obtenus montrent l’intérêt et l’efficacité de l’utilisation de la méthode d’optimisation globale d’Aliénor dans une stratégie de commande prédictive.

Bibligraphie :

[1] Ghizlane HAFIDI, « Application de la commande prédictive non-linéaire à la commande de culture de bactéries Escherichia coli ». Thèse de Doctorat, Université Paris-Sud 11, 2008.

[2] Ghania IDIRI, « Commande prédictive des systèmes non linéaires dynamiques. ». Mémoire de Magister en Automatique, Université Mouloud Mammeri de Tizi-Ouzou, 2011

[3] Abdellah BEZZINI, « Commande Prédictive Non Linéaire en Utilisant Les Systèmes Neuro-Flous et les Algorithmes Génétiques. ». Mémoire de Magister en Automatique, Université Mohamed KHIDER de Biskra, 2013.

[4] Abdesslem DEBBEH, « Commande prédictive généralisée robuste Etude et application. ». Mémoire de Magister, Université Badji Mokhtar de Annaba, 2006.

[5] Amira GHERBOUDJ, « Méthodes de résolution de problèmes difficiles académiques. ». Thèse de Doctorat 3ème cycle LMD, Université de Constantine 2, 2013.

[6] A. MAIDI. Cours Optimisation et Commande Optimale. Département Automatique, Faculté de Génie Electrique et de l'Informatique, UMMTO.

[7] Balira Ousmane KONFE, « Nouvelles méthodes mathématiques Alienor et Adomian, pour la Biomédecine. ». Thèse de doctorat, Université de OUAGADOUGOU, 2005.

[8] Yves Cherruault, Optimisation : Méthodes locales et globales. Editions Presses Universitaires Françaises (PUF), Paris, 1999.

[9] Yves Cherruault, Modèles et méthodes mathématiques pour les sciences du vivant. Editions Presses Universitaires Françaises (PUF), 1998