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

Domaine : **Sciences et Technologies**
Filière : **Génie électrique**
Spécialité : **Commande des systèmes**

Présenté par
Kheloudja Imerzoukene
Tassadit Haddad

Thème

Synthèse d'un correcteur PID robuste basée sur l'approche des scénarios

Mémoire soutenu publiquement le 10/07/ 2017 devant le jury composé de :

M Mohand Achour Touat
MCB, UMMTO, Président

M Ahmed Maidi
Professeur, UMMTO, Encadreur

M Mohamed Ali Bey
MCB, UMMTO, Examinateur

M Abdrahmane Akkouche
MCB, UMMTO, Examinateur

Remerciements

Nous remercions avant tout le bon dieu de nous avoir donné la santé, le courage et la volonté pour finir ce travail.

Nous tenant à remercier notre promoteur Monsieur ***A.MAIDI*** *pour son aide précieuse et de nous avoir fait profiter de son expérience et de nous avoir encouragés tout au long de ce travail.*

Nous exprimons également notre gratitude à tous les enseignants qui ont contribue à notre formation.

Nos remerciements s'adressent aussi aux membres se Jury qui nous ont fait l'honneur de juger ce travail

Enfin, nos remerciement les plus sincères à tous ceux qui ont contribué de prés ou de loi à la concrétisation de ce mémoire.

Dédicaces

Je dédie ce mémoire :

- A mes chers parents que j'aime beaucoup.
- A mes frères et leurs femmes : Mohand et Mazouza, Mehrez et Lila, Nordine et Kamelia ,Yazid.
- A ma très chère et unique sœur Dadiche et son fiancé Lyes.
- A mes adorables nièces et neveux : Maram, Numidia, amina, Akli, Idir.
- A mes amis (es) :Ramdane, Moloud.
- A mes tantes Chabha, Henia et son mari Moussa.
- A mes cousins et cousines spécialement : Nora et Houcine.
- A mon binôme Tassadit, et toute sa famille.
- A toute personne qui m'a aidé de prés ou de loin .

Kheloudja

Dédicaces

Je dédie ce modeste travail avec amour, sincérité et fierté :

- A mes chers parents pour leur contribution à chaque travail que j'ai pu accomplir tout au long de ma vie.
- A mes chers frères : Malik, Idir.
- A mes très chères sœurs : Kahina, karima et sa fille Melissa
- A mes chères tantes et mes adorables cousines, spécialement Ghania.
- A mon binôme Kheloudja, et toute sa famille.
- A mes chers amis (es).
- Et a tout qui compulse ce modeste travail.

Tassadit

NOTATION

u(t) : Signal d'entrée

y(t) : Signal d sortie

p(t) : Perturbation

G(p) : Fonction de transfert

M(t) : La matrice de fonction de transfert d'interconnexion

Δ: Bloc de perturbation.

R : une fonction a plusieurs variables.

p(i) : La probabilité de sélection d'un individu.

$g_i(x)$: Les de type égalité.

$h_j(x)$: Les contraintes de type inégalité.

$f(x)s$: Le cout de la solution *x* sur le scénarios.

θ : Est le vecteur des paramètres incertain.

x_0 : Est appelé le germe (modulo).

m : Appelé le terme de congruence

u_n : Suite de variable aléatoire.

n : Degré de récurrence.

x_i : Variable aléatoire.

N : Nombre de variable aléatoire généré.

ϵ : Paramètre de risque.

β : Paramètre de confiance.

a_1 : Multiplicateur.

a_2 : Incrément.

ACRONYMES

EDO : équations différentielles ordinaires.

EDP : équations aux dérivées partielles.

AG : algorithme génétique.

i.i.d : indépendant identiquement distribué.

LCG : générateur congruentiel linéaire.

Liste des tableaux

Tableau 2.1 Conditions de Kuhn-Tucker……24

Tableau 2.2 Création de la population initiale…… 31

Tableau 2.3 Sélection des individus…… 31

Tableau 2.4 Reproduction d'une nouvelle génération…….32

Tableau4.1 tableau du dénominateur……51

Tableau4.2 tableau du numérateur……51

Tableau 4.3 Dénominateur……54

Tableau 4.4 Numérateur……54

Tableau 4.5 tableau de dénominateur de E(p) ……..……58

Tableau 4.6 tableau de numérateur de E(p)……..……..58

Liste des figures

Figure 1.1 Schéma fonctionnel……..... 4

Figure 1.2 Structure de commande…….……4

Figure 1.3 Configuration de perturbation additive…….…….9

Figure 1.4 Configuration de perturbation additive inverse…….....9

Figure 1.5 Configuration d'une perturbation multiplicative d'entrée…….. .10

Figure 1.6 Configuration d'une perturbation multiplicative de sortie…….10

Figure 1.7 Configuration de perturbation multiplicative d'entré inversée…….....10

Figure 1.8 Configuration d'une perturbation multiplicative de sortie inverse…….…...11

Figure 1.9 Configuration des perturbations du facteur coprime gauche……....11

Figure 1.10 Configuration correcte des perturbations du facteur de coprime droite…....12

Figure 1.11 Schéma de blocs de l'exemple (1.2)……14

Figure 1.12 Schéma bloc des incertitudes structurées de l'exemple (1.2)……...…..14

Figure 1.13 Configuration standard de l'exemple (1.2)…….…...15

Figure 1.14 Configuration standard…….....16

Figure 2.1 Optimum local et global…….……..21

Figure 2.2 Organigramme des algorithmes génétiques…….…….26

Figure 2.3. Représentation schématique du croisement en un point……29

Figure 4.1 Optimisation indicielle…….…….50

Figure 4.2 Commande d'un double intégrateur…….…….....53

Figure 4.3 Réponse du système en boucle fermée pour différentes incertitudes…….61

Figure A.1 Schéma fondamental d'un système à rétroaction……63

Table des matières

Introduction générale 1

Chapitre 1 : Généralités sur les systèmes dynamique 3

1.1Introduction 3

1.2 Généralités sur les systèmes 3

1.2.1 Définition d'un système 3

1.2.2 Modélisation et représentation des systèmes 3

1.2.3 Entrées - sorties-perturbations 3

1.2.4 Structures de commande 4

1.2.5 Quelques classes de systèmes 4

1.2.5.1 Systèmes mono entrée - mono sortie 4

1.2.5.2 Systèmes multi entrées -multi sorties 5

1.2.5.3 Systèmes stationnaires 5

1.2.5.4 Systèmes continus - systèmes discrets 5

1.2.5.5 Systèmes statiques - systèmes dynamiques 5

1.2.5.6 Systèmes déterministes - systèmes stochastiques 5

1.2.5.7 Systèmes linéaires-non linéaires 6

1.2.5.8 Système causal 6

1.2.5.9 Systèmes localisés - systèmes distribués 6

1.3 Robustesse 6

1.3.1 Définitions 6

1.3.1.1 Robuste en stabilité 6

1.3.1.2 Robuste en performance 7

1.3.2 Stabilité robuste dans le contexte H∞ 7

1.3.2.1 Définition de la norme H∞ d'une matrice de transfert 7

1.3.3 Avantages de la commande H∞ 7

1.3.3.1 Propriétés de performance qualitatives garanties par la norme H∞ 7

1.4 Systèmes incertains 8

1.4.1 Définition d'incertitudes 8

1.4.2 Incertitudes non structurées (dynamique) 8

1.4.2.1 Modélisation des systèmes incertains 9

1.3.3 Incertitude paramétrique 13

1.4 Transformations fractionnaires linéaires 16

1.4.1 Perturbation additive 17
1.4.2 Perturbation additive inverse 17
1.4.3 Perturbation multiplicative d'entrée 17
1.4.4 Perturbation multiplicative de sortie 17
1.4.5 Perturbation multiplicative d'entrée inverse 17
1.4.6 Perturbation multiplicative de sortie inverse 18
1.4.7 Perturbations du facteur coprime gauche 18
1.4.8 Perturbations du facteur coprime droite 18
1.5 Conclusion 19
Chapitre 2 : Généralités sur l'optimisation robuste 20
2.1 Introduction 20
2.2 Généralités sur l'optimisation 20
2.2.1 Définition de l'optimisation 20
2.2.2 Méthodes déterministes 21
2.2.3 Optimisation sans contraintes 21
2.2.3.1 Formulation mathématique 21
2.2.4 Optimisation avec contraintes 22
2.2.5 Méthodes stochastiques 24
2.3 Algorithmes génétiques 25
2.3.1 Introduction 25
2.3.2 Définition 25
2.3.2.1 Principaux points qui différencient les AG des autres approches d'optimisation : 25
2.3.2.2 Présentation des algorithmes génétiques (AG) 26
2.4 Principe d'algorithme génétique 26
2.5 Fonctionnement d'un algorithme génétique 27
2.5.1 Codage des données 27
2.5.2 Population initiale 27
2.5.3 Evaluation 27
2.5.4 Opérateurs des algorithmes génétiques 28
2.5.4.1 Opérateur de sélection 28
2.5.4.2 Opérateur de croisement 28
2.5.4.3 Opérateur de mutation 29
2.5.5 Critère de terminaison 30

2.6 Optimisation robuste 35
2.6.1 Historique 35
2.6.2 Définition de l'optimisation robuste 36
2.6.3 Classification de l'optimisation robuste 36
2.6.4 Méthodes d'optimisation robuste 36
2.7 Quelques problèmes d'optimisation en commande robuste 37
2.7.1 Modélisation incertaine 37
2.7.2 Synthèse robuste 37
2.7.3 Analyse robuste 38
2.8 Conclusion 38
Chapitre 3 : Optimisation robuste pa la méthode des scénarios 39
3.1 Introduction 39
3.2 Optimisation par l'approche des scénarios 39
3.3 Principe des scénarios 40
3.3.1 Définition 40
3.3.2 Description 40
3.4 Génération des nombres aléatoires 41
3.4.1 Source aléatoire 41
3.5 Méthode de Monte Carlo 42
3.5.1 Principe de la méthode de Monte Carlo 42
3.5.2 Avantage de la méthode de Monte Carlo 42
3.6 Génération d'une suite i.i.d. uniforme sur [0,1] 43
3.6.1Générateur de nombre pseudo-aléatoire 43
3.6.1.1Générateur congruentiels linéaires 43
3.6.1.2 Problème rencontré par les générateurs congruentiels linéaires 45
3.7 Conclusion 47
Chapitre 4 : Shyntèse d'un correcteur PID robuste par l'approche des scénarios 48
4.1 Introduction 48
4.2 Régulation optimale 48
4.2.1 Représentation 48
4.2.2 Critères de performance 48
4.3Traitement du critère IES 50
4.3.1 L'approche transfert (approche polynomiale) 50

4.3.2 L'approche d'état 52
4.4 Exemple d'utilisation 53
4.5 Optimisation du critère 55
4.6 Utilisation de l'algorithme génétique dans la synthèse des correcteur PID 57
4.7 Formulation du problème d'optimisation 59
4.8 Conclusion 61
Conclusion générale 62
Annexe 63
Références bibliographique 64

En général la conception d'un correcteur pour un système dynamique repose sur l'utilisation d'un modèle nominal. Ce dernier est obtenu soit par modélisation ou identification. Néanmoins, en pratique la plupart des systèmes présentent des variations paramétriques ou les paramètres ne sont pas connus de manière précise mais appartenant à des intervalles supposés connus.

Ainsi, la conception d'un correcteur en se basant sur le modèle nominal conduit généralement à des correcteurs avec des performances médiocres et peu robuste. Ainsi, il est toujours adéquat de considérer un modèle incertain pour la conception d'un correcteur. En effet, il existe deux approches pour la conception des correcteurs robustes pour les systèmes incertains. La première est déterministe et consiste à considérer le pire cas pour les incertitudes. Cette approche conduit généralement à des problèmes d'optimisation très difficiles à résoudre. La deuxième approche stochastique a pour principe de sélectionner un certain nombre de scénarios possibles pour les incertitudes pour concevoir un correcteur robuste vis-à-vis des variations paramétriques. En général, pour les deux approches, on doit résoudre à problème d'optimisation robuste.

Dans ce mémoire, nous allons présenter une approche permettant de concevoir un correcteur PID robuste pour un système incertain en utilisant la méthode des scénarios. L'idée consiste à minimiser un critère de performances défini comme la somme des critères correspondants pour les incertitudes extraites aléatoirement.

Le mémoire est organisé comme suit :

Le premier chapitre introduit des généralités sur les systèmes et les systèmes incertains et la notion sur la robustesse.

Le deuxième chapitre porte sur des généralités sur l'optimisation, et la méthode d'optimisation stochastique : les algorithmes génétiques. Ce chapitre présente aussi l'optimisation robuste.

Le troisième chapitre introduit une méthode d'optimisation robuste appelée approche des scénarios, et les générateurs des variables aléatoires.

Enfin, **le dernier chapitre** présente une approche de synthèse d'un correcteur PID robuste par l'approche des scénarios.

1.1 Introduction :

Dans ce chapitre, nous allons présenter des généralités sur les systèmes dynamiques en particulier les systèmes incertains.

1.2 Généralités sur les systèmes :

1.2.1 Définition d'un système:

Un système est un ensemble d'objets interconnectés et interagissant les uns avec les autres en fonction d'un objectif. Les divers composants (ou sous-systèmes) sont décrits physiquement ou par leurs loi de comportement.

1.2.2 Modélisation et représentation des systèmes:

Modéliser un système c'est déterminer une maquette virtuelle qui le représente par des symboles et des signes. On cherche un modèle d'une part parce qu'on ne peut pas répéter à l'infini une expérimentation.

1.2.3 Entrées - sorties-perturbations:

Le système peut communiquer avec le monde extérieur par l'intermédiaire de deux types de variables.

1. On appelle entrée (ou signale d'entrée) l'excitation appliquée au système à partir d'une source extérieure afin de provoquer une réponse spécifique.

2. On appelle sortie (ou signale de sortie) la réponse effective obtenue à partir du système.

3. perturbations qui sont en général imprévisibles.

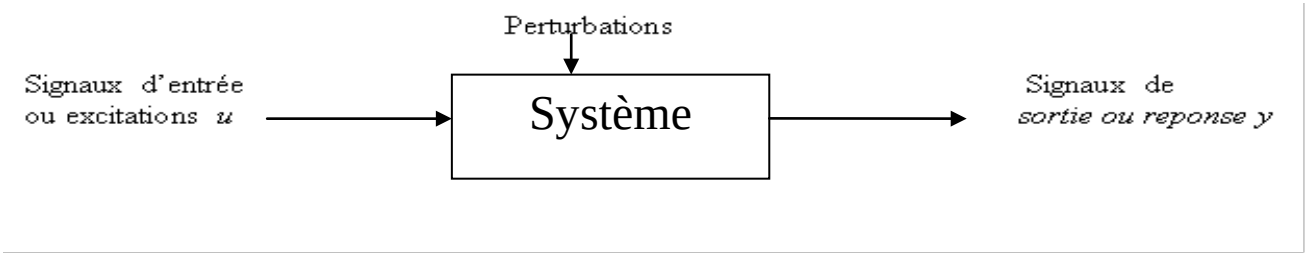


Figure 1.1 Schéma fonctionnel.

1.2.4 Structures de commande:

Il existe deux types de structures de commande. La commande en boucle ouverte (Open-loop) et la commande en boucle fermée ou en contre réaction (Feedback, closed-loop).

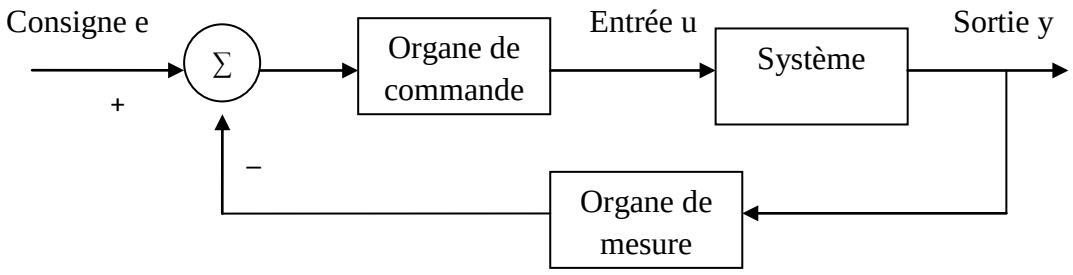


Figure 1.2 Structure de commande

Le système en boucle ouverte est un système où le signale de commande est indépendant du signal de sortie.

Le système en boucle fermée est un système où le signal de commande est dépendant du signal de sortie.

1.2.5 Quelques classes de systèmes:

1.2.5.1 Systèmes mono entrée - mono sortie:

Ce sont des systèmes ne comportant qu'un seul signal d'entrée et un seul signal de sortie.

1.2.5.2 Systèmes multi entrées -multi sorties:

Ce sont des systèmes comportant plusieurs signaux d'entrées et plusieurs signaux de sortie.

1.2.5.3 Systèmes stationnaires:

Ce sont des systèmes dont le modèle mathématique n'a aucun coefficient dépendant du temps. Autrement le système est non stationnaire ou dépendant du temps.

1.2.5.4 Systèmes continus - systèmes discrets:

On parle de système continu quand tous les signaux d'entrées et de sorties sont définis à tout instant t.

On parle de système discret quand tous les signaux d'entrée et de sortie sont définis périodiquement dans le temps.

1.2.5.5 Systèmes statiques - systèmes dynamiques:

Un système statique est un système dont la réponse à un signal d'entrée est instantanée. Sinon il est dit dynamique : la réponse dépend du signal d'entrée présent et des valeurs passées.

1.2.5.6 Systèmes déterministes - systèmes stochastiques:

Un système déterministe est un système dont le comportement est parfaitement prévisible. C'est à dire connaissant le signal d'entrée, la sortie peut être calculée exactement à partir du modèle du système. Un système stochastique est un système à comportement aléatoire.

Certains paramètres (dynamique, données initiales, perturbations) dépendent des variables aléatoires et font que le signal de sortie ne peut pas être déterminé exactement.

1.2.5.7 Systèmes linéaires-non linéaires:

On dit qu'un système est linéaire s'il est régi par un système d'équations différentielles linéaires, au sens mathématique, si le principe de superposition s'applique.

Un système est non linéaire si le signal de sortie est une fonction non linéaire par rapport au signal d'entrée.

Un système peut admettre une représentation linéaire et une autre représentation non linéaire. Par exemple, un système pourra être linéaire en utilisant des coordonnées cartésiennes, et deviendra non linéaire en coordonnées polaires.

1.2.5.8 Système causal:

La réponse temporelle du système ne peut précéder son entrée, tous les systèmes physiques sont causaux.

1.2.5.9 Systèmes localisés - systèmes distribués:

On parle de système localisé (lumped system) lorsqu'il s'agit d'un système dynamique ne faisant apparaitre que la variable temps.

Le système est dit distribué ou à paramètres répartis (distributed paramètre system) lorsqu'il évolue dans le temps et l'espace.

Les systèmes localisés sont généralement décrits par des équations différentielles ordinaires (EDO) alors que les systèmes distribués sont gouvernés par ses équations aux dérivées partielles (EDP).

1.3 Robustesse :

1.3.1 Définitions :

1.3.1.1 Robuste en stabilité :

Si le système demeure stable en présence d'incertitudes telles que les mesures de modélisation, les bruits de mesures ou les perturbations externes.

1.3.1.2 Robuste en performance :

Si les performances sont conservés en présence des perturbations (les performances pouvant se mesurer en terme : temps de réponse, temps de monté, d'amortissement des modes flexibles, découplage, perturbation/sortie, ……..).

1.3.2 Stabilité robuste dans le contexte H∞ :

1.3.2.1 Définition de la norme H∞ d'une matrice de transfert :

- **Cas mono variable :** Soit G(p) une fonction de transfert ||G||∞ est le gain maximal tel qu'il apparaît sur le diagramme de Bode de G(p) : sup| G (p)|

- **Cas multi variable :** Soit G(p) une matrice de transfert multi variable propre, on définit la norme H∞ de G(p) de la manière suivante : G supσ (G(p)ω) = σ désigne la plus grande valeur singulière.

1.3.3 Avantages de la commande H∞ :

• Contrairement aux méthodes modernes (LQ, LQG), en plus de la performance, la méthode H∞ permet en théorie d'assurer la robustesse en prenant explicitement en compte des incertitudes.

• Contrairement aux méthodes classiques (réglages de PI, PID, ...) qui demandent beaucoup de savoir-faire de la part de l'ingénieur, dans la méthode H∞ le problème de synthèse est formulé comme un problème de minimisation d'un critère H∞, qui se ramène à un problème d'optimisation convexe sous contraintes LMI et peut donc se résoudre efficacement par des algorithmes.

1.3.3.1 Propriétés de performance qualitatives garanties par la norme H∞ :

Outre ces propriétés remarquables, la norme H∞ permet de garantir le respect des propriétés qualitatives du cahier des charges : en effet, un système linéaire stationnaire de réalisation d'état minimale est asymptotiquement stable si sa norme H∞ est finie et cela entraîne les propriétés qualitatives :

- l'unicité de la convergence vers un régime permanent.
- une sortie périodique en réponse à une entrée périodique.
- une sortie constante en réponse à une entrée constante.

1.4 Systèmes incertains:

Les incertitudes sont à l'origine de très nombreux échecs survenant lors de l'exécution des programmes de manipulation. Il est donc essentiel de maitriser au maximum ces incertitudes, si l'on veut synthétiser automatiquement des programmes de la commande de la robustesse(les Algorithme génétiques, les méthodes des scénarios,….).

❖ **Vocabulaire :**

On confond souvent dans le langage scientifique (erreur) et incertitude même si on utilise plutôt le terme (erreur) lorsque un processus de mesure est mal maitrisé et (incertitude) lorsque l'évaluation de la fiabilité est immédiate et intuitive.

1.4.1 Définition d'incertitudes :

Il est bien entendu que les incertitudes sont inévitables dans un système de contrôle réel. L'incertitude peut être classée en deux catégories: signaux perturbateurs et perturbations dynamiques. Le premier inclut les perturbations d'entrée et de sortie (comme une rafale sur un aéronef), le capteur et le bruit de l'actionneur, etc. Ce dernier représente l'écart entre le modèle mathématique et la dynamique réelle du système en fonctionnement. Un modèle mathématique de tout système réel est toujours juste une approximation de la vraie réalité physique de la dynamique du système. Sources typiques de discordance comprennent la dynamique non modélisée (habituellement à haute fréquence), non linéarités négligées dans la modélisation, les effets de modèles délibérés à ordre réduit et les variations des paramètres du système en raison des changements environnementaux. Ces erreurs de modélisation peuvent nuire à la stabilité et aux performances d'un système de commande.

On peut distinguer deux types d'incertitudes :

1.4.2 Incertitudes non structurées (dynamique) :

De nombreuses perturbations dynamiques qui peuvent se produire dans différentes parties d'un système peuvent, cependant, être groupées dans un seul bloc de perturbation Δ, par exemple, dynamique non modélisée et à haute fréquence. Cette représentation d'incertitude est

comme une incertitude «non structurée». Dans le cas d'une loi linéaire invariante dans le temps, le bloc Δ peut être représenté par une matrice fonction de transfert inconnue. L'incertitude dynamique non structurée dans un système de contrôle peut être décrite de différentes manières, tel que listée dans la suite, où Gp (s) désigne la dynamique réelle du système perturbé et Go (s) un modèle nominal.

1.4.2.1. Modélisation des systèmes incertains :

- **Perturbation additive:**

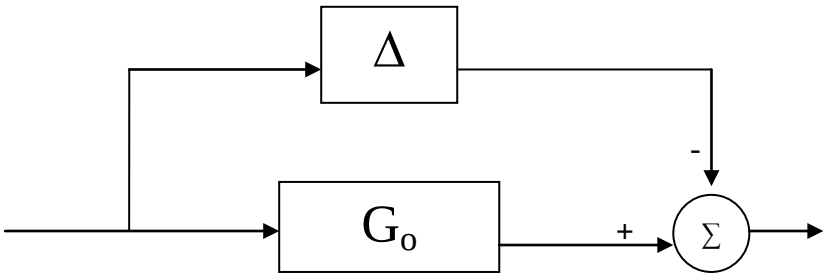


Figure 1.3 Configuration de perturbation additive.

$$Gp(s) = Go(s) + \Delta(s) \qquad (1.1)$$

- **Perturbation additive inverse:**

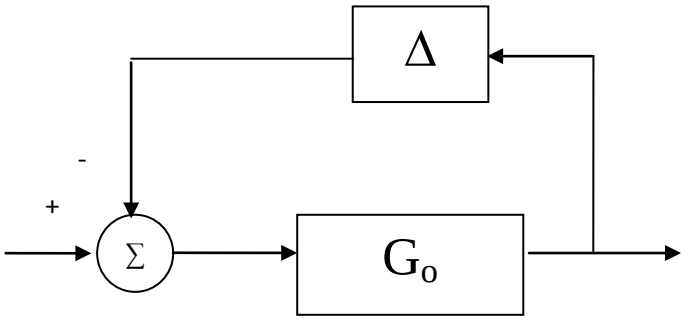


Figure 1.4 Configuration de perturbation additive inverse**.**

$$(Gp(s))^{-1} = (Go(s))^{-1} + \Delta(s) \qquad (1.2)$$

- **Perturbation multiplicative d'entrée:**

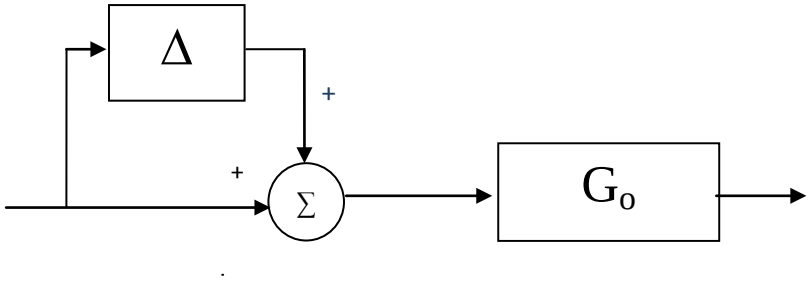


Figure 1.5 Configuration d'une perturbation multiplicative d'entrée.

$$Gp(s) = Go(s)\,[I + \Delta(s)] \qquad (1.3)$$

- **Perturbation multiplicative de sortie:**

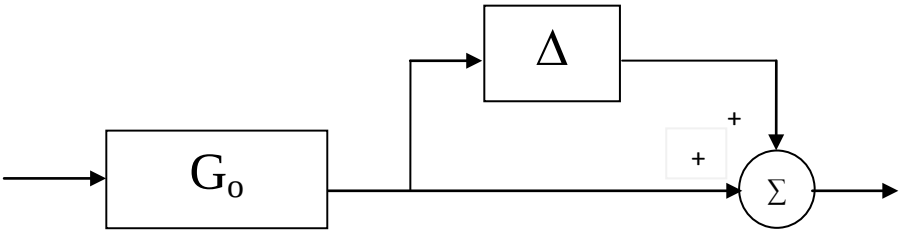


Figure 1.6 Configuration d'une perturbation multiplicative de sortie.

$$Gp(s) = [I + \Delta(s)](s) \qquad (1.4)$$

- **Perturbation multiplicative d'entrée inverse:**

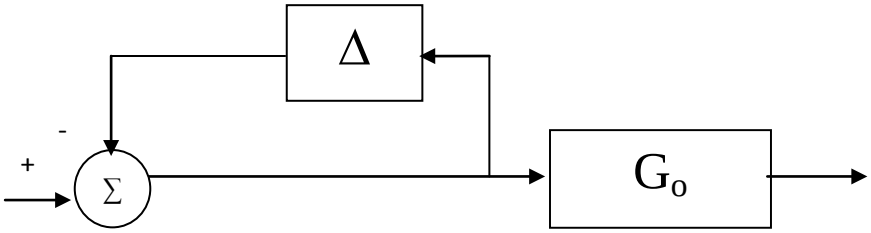


Figure 1.7 Configuration de perturbation multiplicative d'entré inversée.

$$(Gp(s))^{-1} = [I + \Delta(s)]\,(Gp(s))^{-1} \qquad (1.5)$$

- **Perturbation multiplicative de sortie inverse:**

$$(Gp(s))^{-1} = (Gp(s))^{-1}\,[I + \Delta(s)] \qquad (1.6)$$

- **Perturbations du facteur coprime gauche :**

$$Gp(s) = (\tilde{M}+\Delta_{\tilde{M}})^{-1}\,(\tilde{N}+\Delta_{\tilde{N}}) \qquad (1.7)$$

- **Perturbations du facteur coprime droite :**

$$Gp(s) = (N + \Delta_N)\,(M + \Delta_M)^{-1} \qquad (1.8)$$

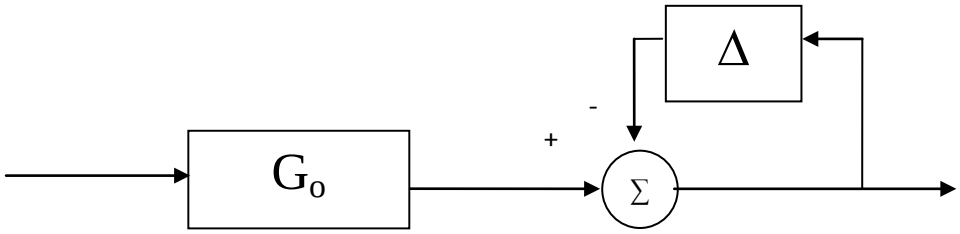


Figure 1.8 Configuration d'une perturbation multiplicative de sortie inverse.

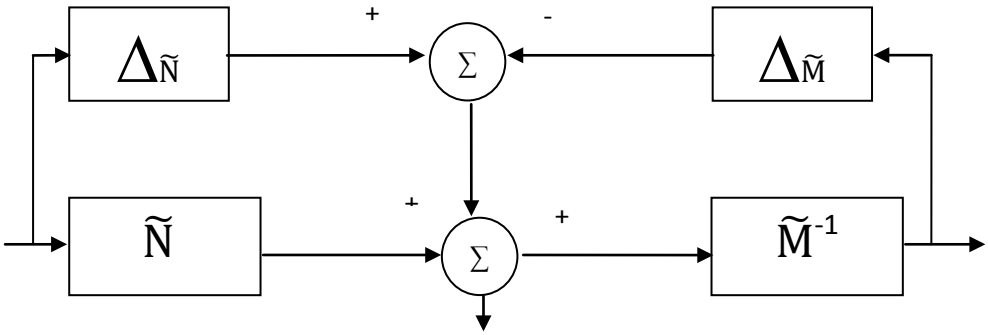


Figure 1.9 Configuration des perturbations du facteur coprime gauche.

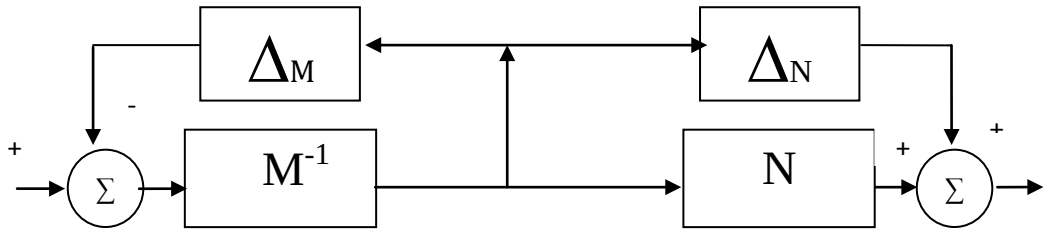


Figure 1.10 Configuration correcte des perturbations du facteur de coprime droite.

Les représentations d'incertitude additive donne l'erreur absolue entre la dynamique réelle et le modèle nominal, alors que la multiplicative représentent les erreurs relatives.

Exemple 1.1

La dynamique de nombreux systèmes de contrôle peut inclure une partie "lente" et une "rapide" partie, par exemple dans un moteur DC. La dynamique réelle peut être de la forme :

$$G_p(s) = G_{gain}G_{lent}(s)G_{rapide}(s)$$

Où, G_{gain} est constant, et

$$G_{lent}(s) = \frac{1}{1+sT}; \quad G_{rapide}(s) = \frac{1}{1+\alpha sT}; \qquad (1.9)$$

Dans la conception, il peut être raisonnable de se concentrer sur la partie lente tout en traitant la dynamique de réponse rapide comme une perturbation. Désignons par Δ_a et Δ_m respectivement les perturbations additives et multiplicatives. Ces dernières peuvent être écrites comme suit :

$$\Delta_a(s) = G_p - G_{gain}G_{lent} = G_{gain}G_{lent}(G_{rapide} - 1)$$

$$= G_{gain}\frac{-\alpha sT}{(1+sT)(1+\alpha sT)}$$

$$\Delta_m(s) = \frac{G_p - G_{gain}G_{lent}}{G_{gain}G_{lent}} = G_{rapide} - 1 = \frac{-\alpha sT}{1+\alpha sT}.$$

1.3.3 Incertitude paramétrique :

Les représentations d'incertitude non structurées discutées à la section 1.3.1 sont utiles pour décrire la dynamique du système non modifiée ou négligée. Ces incertitudes complexes se produisent habituellement dans la gamme des hautes fréquences et peuvent inclure des défauts couplage parasite, hystérésis et autres non linéarités.

Cependant, des perturbations dynamiques dans de nombreux systèmes peuvent également être causées par une description inexacte des caractéristiques des composants. Par exemple, l'usure des composants ou le déplacement des points de fonctionnement, etc. De telles perturbations peuvent être représentées par des variations de certains paramètres du système sur certaines plages de valeurs possibles .Ces incertitudes affectent les performances en basse fréquence et s'appellent "incertitudes paramétriques".

Exemple 1.2 :

Un système masse-amortisseur-ressort peut être décrit par l'équation différentielle ordinaire du second ordre suivante :

$$m\frac{d^2x(t)}{dt^2} + c\frac{dx(t)}{dt} + kx(t) = f(t) \qquad (1.10)$$

Où m est la masse, c la constante d'amortissement, k la rigidité du ressort, x(t) le déplacement et f (t) la force externe. Lorsque les paramètres du système sont incertains ou variant, le système peut s'écrire comme suit :

$$(m_{0+} \quad \delta_m)\frac{d^2x(t)}{dt^2} + \quad (c_{0+}\delta_c)\frac{dx(t)}{dt} + (k_{0+} \quad \delta_k)x(t) = f(t) \qquad (1.11)$$

Où m_0 ,c_0 et k_0 désignent les valeurs nominales des paramètres et δ_m, δ_c et δ_k sont les variations possibles sur certaines gammes.

En définissant les variables d'état x_1 et x_2 comme variable de déplacement et sa dérivée de premier ordre (vitesse), l'équation différentielle de $2^{ème}$ ordre (1.2). Peut être réécrit sous forme d'espace d'état standard comme suit :

$$\dot{x}_1 = x_2$$

$$\dot{x}_2 = \frac{1}{m_0+\delta_m}\left[-(k_0 + \delta_k)\, x_1 - (c_o + \delta_c)\, x_2 + f\right]$$

$$y = x_1 \qquad (1.11)$$

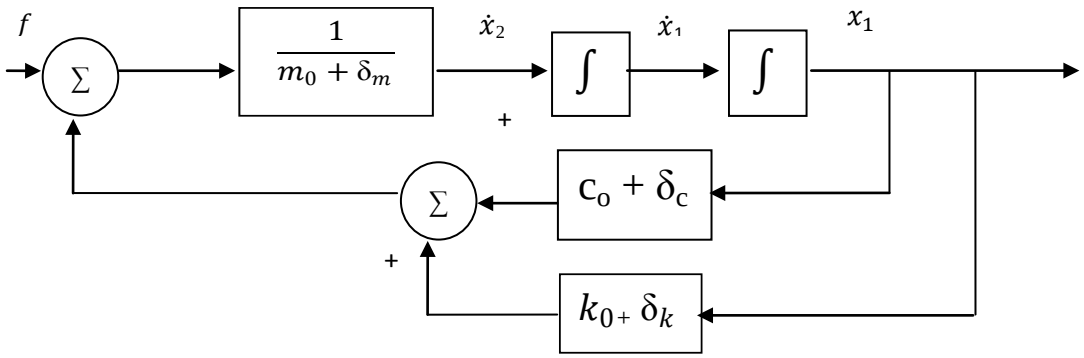


Figure 1.11 Schéma de blocs de l'exemple 1.2.

Le système peut être représenté par un schéma blocs de la figure 1.10 .Notez que $\frac{1}{m_0+\delta_m}$ peut être réarrangé en tant que rétroaction (boucle fermé) en termes de $\frac{1}{m_0}$ et δ_m. La figure (1.11) peut être représentée en séparant les paramètres et les incertitudes comme indiquéparlaFigure1.12.

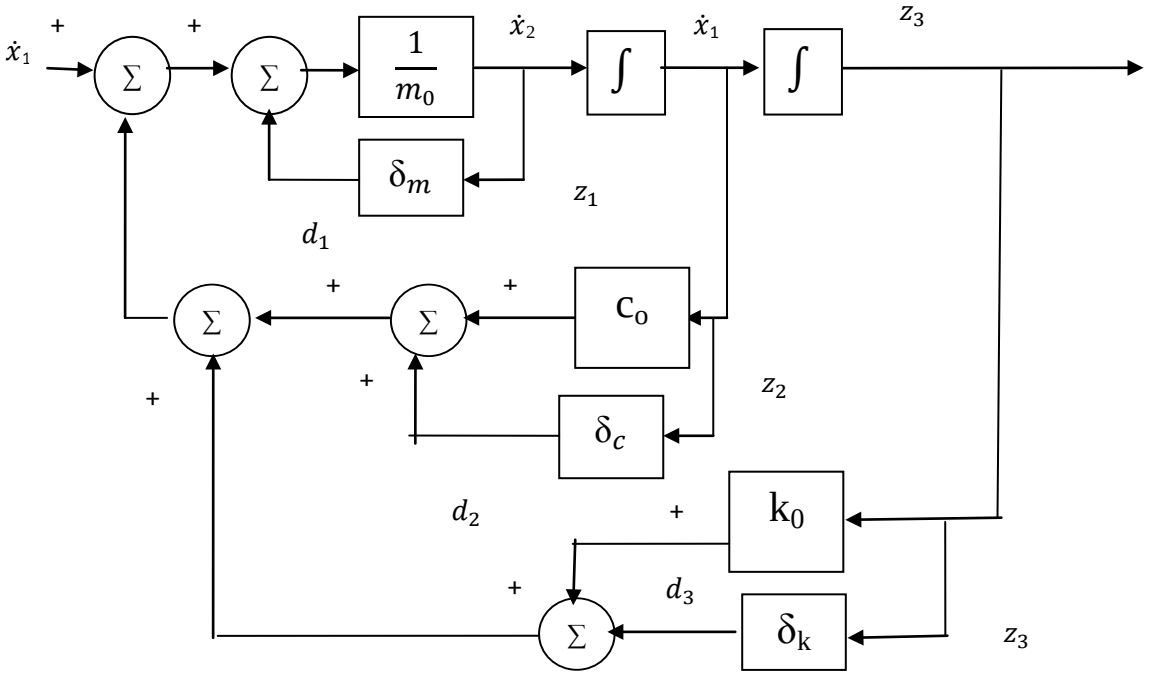


Figure 1.12. Schéma bloc des incertitudes structurées de l'exemple (1.2)

Soient z_1, z_2 et z_3, $\dot{x}_2$,x_2 et x_1, respectivement considérés comme autres vecteurs de sortie fictifs, et d_1, d_2 et d_3; les signaux sortant de blocs de perturbationδ_m, δ_c et δ_k, comme le montre la figure 1.12.

Le système perturbé peut être représenté par le modèle d'espace d'état suivant :

$$\begin{bmatrix}\dot{x_1}\\ \dot{x_2}\end{bmatrix}=\begin{bmatrix}0 & 1\\ -\frac{k_0}{m_0} & -\frac{c_0}{m_0}\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix}+\begin{bmatrix}0 & 0 & 0\\ -1 & -1 & -1\end{bmatrix}\begin{bmatrix}d_1\\ d_2\\ d_3\end{bmatrix}+\begin{bmatrix}0\\ \frac{1}{m_0}\end{bmatrix}\mathrm{f}$$

$$\begin{bmatrix}z_1\\ z_2\\ z_3\end{bmatrix}=\begin{bmatrix}-\frac{k_0}{m_0} & -\frac{c_0}{m_0}\\ 0 & 1\\ 1 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix}+\begin{bmatrix}-1 & -1 & -1\\ 0 & 0 & 0\\ 0 & 0 & 0\end{bmatrix}\begin{bmatrix}d_1\\ d_2\\ d_3\end{bmatrix}+\begin{bmatrix}\frac{1}{m_0}\\ 0\\ 0\end{bmatrix}\mathrm{f}$$

$$y=\begin{bmatrix}1 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix}$$

Et représenté comme suit :

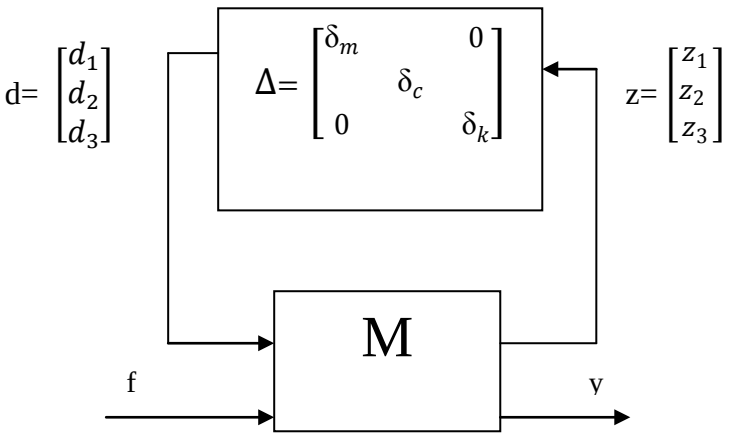


Figure 1.13 Configuration standard de l'exemple (1.2)

Le bloc de perturbation Δ de la figure 1.13 correspond aux variations de paramètres et s'appelle «incertitude paramétrique». Le bloc Δ n'est pas une matrice complète mais une diagonale. Il a une certaine structure d'où la terminologie de «l'incertitude structurée».

1.4 Transformations fractionnaires linéaires :

Le schéma de principe de la figure 1.13 peut être généralisé comme une configuration standard pour représenter la façon dont l'incertitude affecte la relation d'entrée / sortie du système a été adoptée pour la modélisation de l'incertitude et la synthèse de commandes robustes. Le cas général est représenté à la figure 1.14.

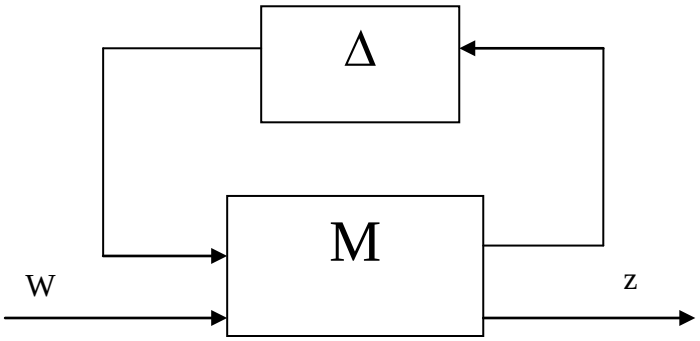


Figure 1.14 Configuration standard M-Δ

La matrice de fonction de transfert d'interconnexion M de la figure 2.13 est partitionnée comme suit :

$$M=\begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix}$$

Où les dimensions de M_{11} et de Δ sont identiques. Par des manipulations mathématiques élémentaires, on peut démontrer :

$$z= [M_{22}+M_{21}\Delta(\text{I}-M_{11})^{-1}M_{12}]\text{w}$$

Si $(\text{I}-M_{11})$ est inversible, nous pouvons définir :

$$\text{F (M ; }\Delta)=M_{22} + M_{21}\Delta\ (\text{I}-M_{11})^{-1}$$

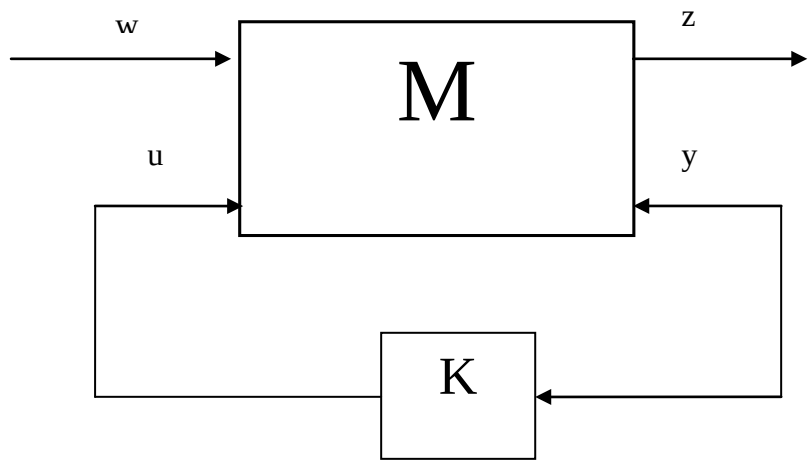
F (M ; Δ) s'appelle une transformation fractionnaire linéaire (LFT) de M et Δ.

Parce que la boucle "supérieure" M est fermée par le bloc Δ, ce type de transformation fractionnaire linéaire est également appelée transformation fractionnaire linéaire supérieure (ULFT), et noté avec un indice u, c'est-à-dire Fu (M; Δ), pour montrer le mode de connexion.

De même, il existe également des transformations fractionnaires linéaires inférieures (LLFT) qui sont habituellement utilisées pour indiquer l'incorporation d'un contrôleur dans un système.

La LFT inférieure représentée sur la figure 1.15 et définie par :

$$F_l(M; K) = M_{11} + M_{12}K(I - M_{22}K)^{-1} M_{21}$$



1.4.1 Perturbation additive:

$$M=\begin{bmatrix}0 & I\\ I & G_0\end{bmatrix} \qquad (1.12)$$

1.4.2 Perturbation additive inverse:

$$M=\begin{bmatrix}-G_0 & G_0\\ -G_0 & G_0\end{bmatrix} \qquad (1.13)$$

1.4.3 Perturbation multiplicative d'entrée:

$$M=\begin{bmatrix}0 & I\\ G_0 & G_0\end{bmatrix} \qquad (1.14)$$

1.4.4 Perturbation multiplicative de sortie:

$$M=\begin{bmatrix}0 & G_0\\ I & G_0\end{bmatrix} \qquad (1.15)$$

1.4.5 Perturbation multiplicative d'entrée inverse:

$$M=\begin{bmatrix} -I & I \\ -G_0 & G_0 \end{bmatrix} \qquad (1.16)$$

1.4.6 Perturbation multiplicative de sortie inverse:

$$M=\begin{bmatrix} -I & G_0 \\ -I & G_0 \end{bmatrix} \qquad (1.17)$$

1.4.7 Perturbations du facteur coprime gauche:

$$M=\begin{bmatrix} \begin{bmatrix} -\widetilde{M_G}^{-1} \end{bmatrix} & \begin{bmatrix} -G_0 \\ I \end{bmatrix} \\ -\widetilde{M_G}^{-1} & G_0 \end{bmatrix} \qquad (1.18)$$

Où : $G_0 = \widetilde{M_G}^{-1}\widetilde{N_G}$;

Une factorisation coprime gauche de la structure nominale, et la structure perturbée est

$$G_p = (\widetilde{M_G} + \Delta_{\widetilde{M}})^{-1}(\widetilde{N_G} + \Delta_{\widetilde{N}}).$$

1.4.8 Perturbations du facteur coprime droite:

$$M=\begin{bmatrix} \begin{bmatrix} -M_G^{-1} & 0 \end{bmatrix} & M_G^{-1} \\ \begin{bmatrix} -G_0 & I \end{bmatrix} & G_0 \end{bmatrix} \qquad (1.19)$$

Dans ce qui précède, on suppose que [I -$M_{11}\Delta$] est inversible. Le système perturbé est :

$$G_p(s) = F_0\,(M;\Delta)$$

Exemple 1.3 :

(SISO) incertitude sur un coefficient du polynôme caractéristique:

$G(s) = \frac{1}{s^2+as+1}$ Pour $0.4 \leq a \leq 0.8$ $a = 0.6 + 0.2\Delta$ $\Delta \leq 1$

$\tilde{G}$(s)= $\frac{G(s)}{1+\Delta w_i(s)G(s)}$ $G(s) = \frac{1}{s^2+0.6s+1}$ $w_i(s)$=0.2s

1.5 Conclusion:

Nous avons introduit dans ce premier chapitre des généralités sur les systèmes, leurs classes et leurs différentes représentations. Puis nous avons introduit les systèmes incertains et leurs représentations.

La synthèse d'un correcteur pour un système incertain, c'est-à-dire un correcteur robuste, passe nécessairement par une étape d'optimisation. Le chapitre suivant est consacré à l'optimisation de fonctions.

2.1 Introduction:

La résolution d'un problème d'optimisation consiste à explorer un espace de recherche afin de maximiser (ou minimiser) une fonction donnée. Les complexités (en taille ou en structure) relatives de l'espace de recherche et de la fonction à maximiser conduisent à utiliser des méthodes de résolutions stochastique (Exemple : Algorithme génétique)

Dans ce chapitre, on introduit principalement la notion d'optimisation d'une manière générale. La première section est une introduction à l'optimisation .Dans la seconde section, on présente les algorithmes génétiques. Nous examinerons leurs terminologies de base, leur principe, leur fonctionnement, et les méthodes les plus utilisées pour chaque opérateur génétique. Par la suite on se focalise sur l'optimisation robuste, ainsi nous allons introduire la notion de robustesse et la stabilité des systèmes robuste.

2.2 Généralités sur l'optimisation:

2.2.1. Définition de l'optimisation:

En mathématiques, l'optimisation traite de la recherche d'un extremum d'une fonction, dont les entrées peuvent être soumises à des contraintes.

L'optimisation est une branche des mathématiques cherchant, à analyser et à résoudre analytiquement ou numériquement les problèmes qui consistent à minimiser ou maximiser une fonction sur un ensemble.

Aujourd'hui, tous les systèmes susceptibles d'être décrits par un modèle mathématique sont optimisés. La qualité des résultats et des prédictions dépend de la pertinence du modèle, de l'efficacité de l'algorithme et des moyens pour le traitement numérique.

- Optimum global et optimum local :

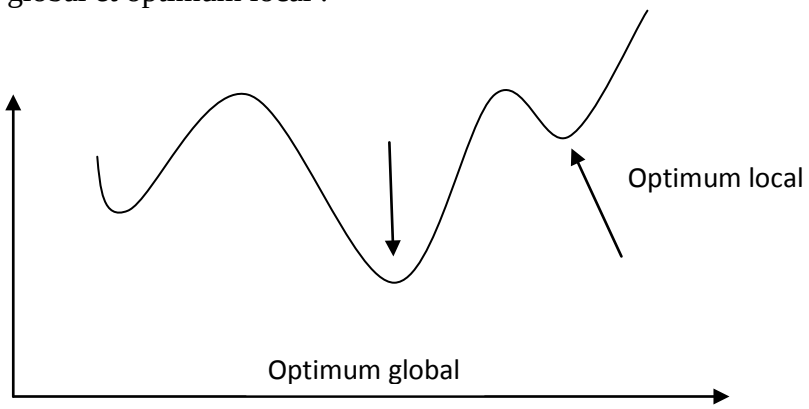


Figure 2.1 Optimum local et global

Les méthodes d'optimisation sont très nombreuses. On peut toutefois les classer en deux grandes familles : les méthodes déterministes et les méthodes stochastiques**.[1]**

2.2.2. Méthodes déterministes :

Les méthodes déterministes conduisent toujours à la même solution finale, elles nécessitent relativement peu d'évaluation de la fonction objectif, mais elles peuvent se bloquer sur un optimum local.

2.2.3. Optimisation sans contraintes :

2.2.3.1. Formulation mathématique :

Un problème d'optimisation est formulé mathématiquement comme suit :

$$\textbf{(p)} \qquad \begin{cases} \min J(x) \\ h(x) \leq 0 \\ g(x) = 0 \\ x \epsilon\, R^n \end{cases}$$

J: $R^n \rightarrow R$ est une fonction de plusieurs variables ($x = (x_1, \dots, x_n)$) à valeurs réelles. Cette fonction (que l'on minimise) est appelée indifféremment fonction coût, objectif ou critère.

g: $R^n \rightarrow R^p$ Est une fonction de plusieurs variables $x \epsilon\ R^n$ à valeurs dans

R^p : Elle a p composantes et on peut écrire

$$g(x) = \big(g_1(x), \dots \dots . . g_p(x)\big))$$

h: $R^n \rightarrow R^q$ Est une fonction de plusieurs variables $x \epsilon R^n$ à valeurs dans
R^q : Elle a q composantes et on peut écrire

$$h(x) = \big(h_1(x), \dots \dots . . h_q(x)\big))$$

Dans cette section, nous allons présenter quelques algorithmes permettant de calculer (de manière approchée) la ou les solutions du problème (**P)** de départ.

Nous présentons les méthodes les plus classiques. Toutefois, la plupart de ces algorithmes exploitent les conditions d'optimalité permettant (au mieux) de déterminer des minima locaux.

Parmi, les méthodes d'optimisation ont peut citer :

- **Méthode du Gradient :** La méthode (ou algorithme) du Gradient fait partie d'une classe plus grande de méthodes numérique appelées méthodes de descente.

- **Méthode de Newton :** La méthode de Newton n'est pas une méthode d'optimisation à proprement parler. C'est en réalité une méthode utilisée pour résoudre des équations non linéaires de la forme $f(x) = 0$ ou f est une fonction de R^n dans R^n.

- **Méthode du Gradient conjuguée :** (cas général) : On peut maintenant (indépendamment de la philosophie initiale de la méthode du gradient conjuguée) étendre cette méthode au cas de la minimisation d'une fonction J quelconque.

2.2.4 Optimisation avec contraintes :

On rappelle qu'on se donne f$\subset R^n$ où U est un ensemble fermé des contraintes, $U \neq \emptyset$ et $U \neq R^n$.

On cherche à résoudre le problème de minimisation

$$\min_{x \in U} f(x)$$

C'est-à-dire, on cherche $x^* \in U$ tel que

$$f(x^*) \leq f(x), \ \ \forall u \in U)$$

✓ **Remarque :**

Dans la pratique on peut rencontrer des problèmes de minimisation avec contraintes égalités, c'est-à-dire, des contraintes de la forme

$$\widetilde{U} = \{x \in R^n, g_i(x) = 0, i = 1,2, \dots, p \ \}$$

Ou des problèmes avec des contraintes mixtes (égalités et inégalités). Dans ce cas, on peut toujours se ramener à des problèmes avec contraintes inégalités, ceci par deux méthodes :

- soit en éliminant des contraintes à l'aide des égalités.
- soit en écrivant une égalité $g_i(x) = 0$ comme une double inégalité : $g_i(x) \leq 0$ et $-g_i(x) \leq 0$.

- **Multiplicateur de Lagrange :**

La méthode des multiplicateurs de Lagrange permet de trouver les points stationnaires (maximum, minimum…) d'une fonction dérivable d'une ou plusieurs variables, sous contraintes. Visuellement, cette méthode permet de trouver un optimum.

Formellement, on note comme suit l'écriture du lagrangien :

$$L(x, \lambda) = f(x) + \lambda g(x) \qquad (2.1)$$

Avec x est le vecteur de variables de décision (d'optimisation) figurant dans la fonction à maximiser ou à minimiser, $f(x)$ est la fonction à optimiser, λ le multiplicateur de Lagrange et $g(x)$ la contrainte du problème d'optimisation.

- **Méthode de Kuhn Tucker :**

Les conditions de Kuhn Tucker permettent de résoudre des problèmes d'optimisation sous contraintes non linéaires du type inégalités.

Les conditions d'optimalité d'une fonction soumise à des contraintes inégalité sont résumées dans le Tableau suivant **:[3]**

Contraintes	Problème	
	Minimum	Maximum
$h_{j(x)\leq 0}$	$\nabla_j L(x,\lambda) = 0$ $\nabla_\lambda L(x,\lambda) \leq 0$ $u_j h_{j=0}$ $u_j \geq 0$	$\nabla_j L(x,\lambda) = 0$ $\nabla_\lambda L(x,\lambda) \leq 0$ $u_j h_{j=0}$ $u_j \leq 0$
$h_{j(x)\geq 0}$	$\nabla_j L(x,\lambda) = 0$ $\nabla_\lambda L(x,\lambda) \geq 0$ $u_j h_j = 0$ $u_j \leq 0$	$\nabla_j L(x,\lambda) = 0$ $\nabla_\lambda L(x,\lambda) \geq 0$ $u_j h_j = 0$ $u_j \geq 0$

Tableau 2.1 Conditions de Kuhn-Tucker

- ✓ **Remarque :**

Lorsque le sens des inégalités des contraintes change, le signe de u_j correspondant change.

Il en est de même si au lieu d'un minimum on cherche un maximum.

2.2.5. Méthodes stochastiques :

Les méthodes d'optimisations stochastiques s'appuient sur des mécanismes de transition probabilistes et aléatoires. Ces méthodes ont une grande capacité de trouver l'optimum global du problème. Contrairement à la plupart des méthodes déterministes, elles ne nécessitent ni de

point de départ, ni la connaissance du gradient de la fonction objectif pour atteindre la solution optimale. Parmi ces méthodes, on retrouve les algorithmes génétiques.

2.3 Algorithmes génétiques :

2.3.1 Introduction :

Les algorithmes génétiques, initiés dans les années 1970 par John Holland, sont des algorithmes d'optimisation s'appuyant sur des techniques dérivées de la génétique et des mécanismes d'évolution de la nature : croisement, mutation, sélection. Leur but est d'obtenir une solution approchée à un problème d'optimisation, lorsqu'il n'existe pas de méthode exacte (ou que la solution est inconnu) pour le résoudre en un temps raisonnable.

2.3.2 Définition :

L'algorithme génétique (AG) est un algorithme de recherche basé sur les mécanismes de la sélection naturelle et de la génétique. Il combine une stratégie de « survie des plus forts » avec un échange d'information aléatoire mais structuré. Pour un problème pour le quel une solution est inconnue, un ensemble de solutions est crée aléatoirement. On appelle cet ensemble la population. Les caractéristiques (ou variables à déterminer) sont alors utilisées dans des séquences de gènes qui seront combinées avec d'autre gènes pour former des chromosomes et après des individus. Chaque solution est associée à un individu, et cet individu est évalué et classifié selon sa ressemblance avec la meilleure, mais encore inconnue, solution au problème.

2.3.2.1 Principaux points qui différencient les AG des autres approches d'optimisation :

- Les AG utilisent un codage des paramètres et non les paramètres eux mêmes.
- Les AG travaillent sur une population de points et non sur un point particulier.
- Les AG ne posent aucune condition sur la nature des fonctions à optimiser.
- Les AG utilisent des règles de transition probabilistes, et non déterministes.

2.3.2.2. Présentation des algorithmes génétiques (AG) :

- ✓ **Un algorithme génétique est définit par** :
- Individu/chromosome/séquence : une solution potentille du problème.
- Population : un ensemble de chromosome ou de points de l'espace de recherche.
- Environnement : l'espace de recherche.
- Fonction de fitness : la fonction- positive-que nous cherchons à maximiser.

2.4. Principe d'algorithme génétique :

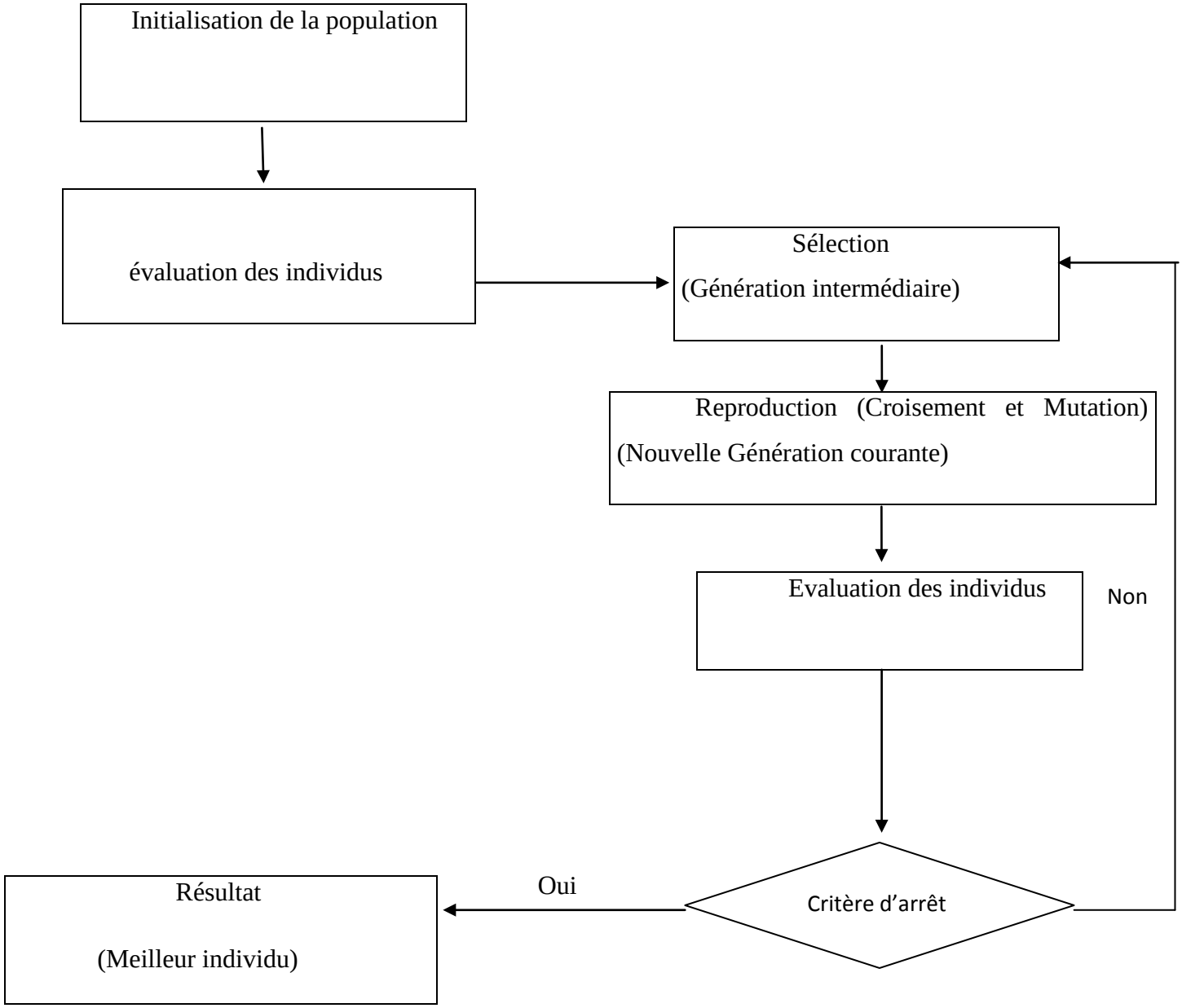


Figure 2.2 Organigramme des algorithmes génétiques

2.5. Fonctionnement d'un algorithme génétique :

2.5.1. Codage des données :

La première étape est de définir et de coder convenablement le problème. Le codage de chaque individu en séquence est essentielle dans l'élaboration d'un algorithme génétique dont dépend notamment l'implémentation des opérateurs de transformation. Le codage doit donc être adapté au problème traité.

Plusieurs types de codages sont utilisés dans la littérature, les premiers résultats théoriques sur les algorithmes génétiques ont opté pour un codage par une séquence binaire. L'efficacité de l'algorithme génétique dépend donc du choix convenable du type de codage.

2.5.2 Population initiale :

L'AG démarre avec une population composée de N individus dans le codage retenu. Le choix des individus conditionne fortement la rapidité de l'algorithme. Si la position de l'optimum dans l'espace de recherche est totalement inconnu, il est intéressent que la population soit répartie sur tout l'espace de recherche.

2.5.3 Evaluation :

C'est la partie la plus complexe, avant de sélectionner les individus, il est nécessaire d'évaluer leur adaptation au problème considéré.

Pour cela, on définit pour chaque problème une fonction d'adaptation, qui mesure le niveau d'adaptation de chaque individu au problème d'optimisation. Cette fonction est appelée plusieurs fois, avant la sélection pour évaluer les individus de la population-parent, et après la mutation, pour évaluer la population enfant et mesurer son adaptation au problème, et ce, à chaque itération.

C'est cette évaluation qui donne la probabilité de sélection d'un individu $p(i)$, telle que :

$$p(i) = \frac{f(x_i)}{\sum_{i=1}^{n} f(x_i)} \qquad \forall\, i \in [1, n] \qquad (2.2)$$

2.5.4 Opérateurs des algorithmes génétiques :

Les opérateurs jouent un rôle prépondérant dans la possible réussite d'un AG. Nous en dénombrons trois principaux : l'opérateur de sélection, de croisement et de mutation. Si le principe de chacun de ces opérateurs est facilement compréhensible, il est toutefois difficile d'expliquer l'importance isolée de chacun de ces opérateurs dans la réussite de l'AG. Cela tient pour partie au fait que chacun de ces opérateurs agit selon divers critères qui lui sont propres.

2.5.4.1 Opérateur de sélection :

La sélection a pour objectif d'identifier les individus qui doivent se reproduire. Cet opérateur ne crée pas de nouveaux individus mais identifie les individus sur la base de leur fonction d'adaptation, la sélection doit favoriser les meilleurs éléments selon le critère à optimiser (minimiser ou maximiser).

2.5.4.2 Opérateur de croisement :

L'opérateur de croisement est un opérateur génétique qui permet à deux chromosomes parents de donner deux chromosomes enfants. Cet opérateur se produit selon une probabilité P_c fixée par l'utilisateur selon le problème à optimiser. L'opérateur de croisement permet de créer de nouvelles séquences de gènes pour les chromosomes enfants à partir d'une base de configuration des séquences héritées des chromosomes parents.

✓ **Exemple :**

1. Croisement en un point :

Pour chaque couple, on choisit au hasard un point de croisement (figure 2.3). Le croisement s'effectue au niveau binaire, et au niveau des gènes. Un croisement peut être coupé au milieu d'un gène.

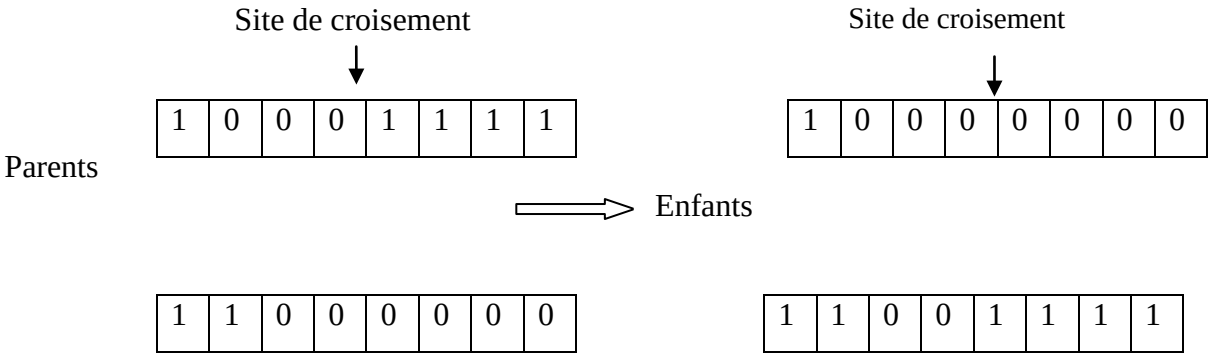


Figure 2.3 Représentation schématique du croisement en un point.

2.5.4.3 Opérateur de mutation :

La mutation est un changement aléatoire selon une certaine règle probabiliste qui doit se faire sur les génotypes, avec une faible probabilité P_m (fixée par l'utilisateur) de la valeur d'un ou plusieurs allèles d'un chromosome. En générale, la mutation ne permet pas l'obtention, de meilleurs solutions, mais elle permet de garder une diversité dans l'évolution des individus et d'éviter les optimum locaux, et se protège contre une perte dans les caractéristiques des individus.

La mutation consiste à transformer dans un chromosome binaire un 1 en 0 ou le contraire :

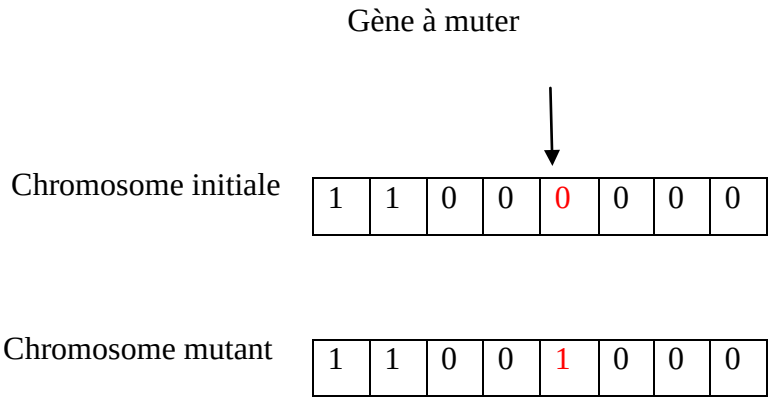


Figure 2.4 Mutation en codage binaire

2.5.5 Critère de terminaison :

Généralement, un algorithme génétique se termine après un certain nombre de générations, mais on peut également terminer l'exécution de l'algorithme lorsqu'une certaine condition soit atteinte, par exemple lorsque la qualité d'un individu dépasse un certain seuil.

- ✓ **Exemple d'application :**

L'exemple consiste à trouver le maximum de la fonction $f(x) = x$ sur l'intervalle [0,31] :

$$\max_{x} \quad f(x) = x$$

Sujet à :

$$0 \leq x \leq 31$$

Nous fixons la taille de la population N = 4. Nous tirons de façon aléatoire 4 chromosomes sachant qu'un chromosome est composé de 5 bits.

Tirage et évaluation de la population initiale :

Numéro	Séquence	Individu	$p(i)$ en %
1	01000	8	12,12
2	10001	17	25,75
3	01110	14	21,21
4	11011	27	40,09

Tableau 2.2 Création de la population initiale

- **Sélection de parents :**

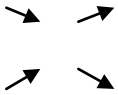
On tire au hasard 4 nouveaux individus parmi les existants en tenant compte de la valeur de répartition.

Numéro	Séquence	individu
1	01110	14
2	11011	27
3	11011	27
4	10001	17

Tableau 2.3 Sélection des individus.

- **Croisement** : on prend les individus 2 par 2. On coupe les chromosomes à une position aléatoire et on croise les parties coupées

$$\begin{cases} 14 : 0111||0 \\ 27 : 1101||1 \end{cases} \qquad \begin{cases} 15 : 0111||1 \\ 26 : 1101||0 \end{cases}$$



$$\begin{cases} 27: 11||011 \\ 17: 10||001 \end{cases} \qquad \begin{cases} 31: 11||111 \\ 21: 10||101 \end{cases}$$

- **Mutation :**

La mutation est la modification aléatoire de la valeur d'un bit (inversion d'un bit).

$15 : 01111 \rightarrow 01011 : 11$

La nouvelle génération est donc :

	Séquence	Individus	$p(i)$ en %
1	01011	11	12,35
2	11010	26	29,21
3	11000	31	34,83
4	10010	21	23,59

Tableau 2.4 Reproduction d'une nouvelle génération

Le maximum est maintenant de 31 (séquence 3). Nous somme donc passé de 27 à 31 après une seule génération.

- ✓ **Exemples sur l'optimisation déterministe :**

Contrainte égalité :

$$\min_x f(x) = 5x_1^2 + 6x_2^2 - x_1x_2$$

Sujet à :

$$g(x) = x_1 + 2x_2 - 24 = 0$$

Le multiplicateur de Lagrange :

$L(x,\lambda) = (5x_1^2 + 6x_2^2 - x_1x_2) + \lambda(x_1 + 2x_2 - 24)$

Points critiques :

$$\frac{\partial L(x,\lambda)}{\partial x_1} = 10x_1 - x_2 + \lambda = 0$$

$$\frac{\partial L(x,\lambda)}{\partial x_2} = 12x_2 - x_1 + 2\lambda = 0$$

$$\frac{\partial L(x,\lambda)}{\partial \lambda} = x_1 + 2x_2 - 24 = 0$$

On a un seul point critique :(x_1, x_2, λ) = (6,9,-51)

Nature du point critique :

$$\begin{vmatrix} WI_{2*2} - \nabla_x^2 L(x,\lambda) & |G(x)|^T \\ |G(x)| & 0 \end{vmatrix} = \begin{vmatrix} W\begin{vmatrix}1 & 0\\ 0 & 1\end{vmatrix} - \begin{bmatrix} \frac{\partial^2 L(x,\lambda)}{\partial x_1^2} & \frac{\partial^2 L(x,\lambda)}{\partial x_2\,\partial x_1} \\ \frac{\partial^2 L(x,\lambda)}{\partial x_1\,\partial x_2} & \frac{\partial^2 L(x,\lambda)}{\partial x_2^2} \end{bmatrix} & \begin{bmatrix} \frac{\partial g(x)}{\partial x_1} \\ \frac{\partial g(x)}{\partial x_2} \end{bmatrix} \\ \begin{bmatrix} \frac{\partial g(x)}{\partial x_1} & \frac{\partial g(x)}{\partial x_2} \end{bmatrix} & 0 \end{vmatrix} =$$

$$\begin{vmatrix} w-10 & 1 & 1 \\ 1 & W-12 & 2 \\ 1 & 2 & 0 \end{vmatrix} \Rightarrow W = \frac{56}{5} > 0$$

On constate que le point trouvé est un minimum

Contrainte inégalité (méthodes de Kuhn Tucker) :

$$\min_x f(x) = x_1^2 + x_2^2 - 8x_1 - 10x_2$$

Sujet à :

$$h_1(x) = 3x_1 + 2x_2 - 6 \leq 0$$

Lagrange :

$$L(x,u) = x_1^2 + x_2^2 - 8x_1 - 10x_2 + u_1(3x_1 + 2x_2 - 6) \qquad (2.3)$$

Les conditions de Kuhn Tucker :

$$\frac{\partial L(x,u)}{\partial x_1} = 2x_1 - 8 + 3u_1 = 0 \qquad (2.4)$$

$$\frac{\partial L(x,u)}{\partial x_2} = 2x_2 - 10 + 2u_1 = 0 \qquad (2.5)$$

$$\frac{\partial L(x,u)}{\partial u_1} = 3x_1 + 2x_2 - 6 \leq 0 \qquad (2.6)$$

$$u_1 h_1(x) = u_1(3x_1 + 2x_2 - 6) = 0 \qquad (2.7)$$

$$u_1 \geq 0 \qquad (2.8)$$

$\Rightarrow u_1 = 0$ ou $3x_1 + 2x_2 - 6=0$

$$x_1 = 4$$

$$x_2 = 5$$

12+10-6=16> 0 (la condition n'est pas vérifiée) donc solution rejetée.

$\Rightarrow u_1 \neq 0$

$$3x_1 + 2x_2 - 6 = 0 \qquad (2.9)$$

$u_1 = \frac{32}{13} > 0$, La condition est vérifiée

$$\boxed{f(x) = -\frac{227}{13}} \qquad (2.10)$$

La solution est : $x_1=\frac{4}{13}$, $x_2=\frac{33}{13}$

2.6 Optimisation robuste :

2.6.1 Historique:

Les origines de l'optimisation robuste remontent aux débuts de la théorie de la décision moderne dans les années 1950. Des « analyses des cas les plus défavorables » ont été réalisées pour faire face aux fortes incertitudes. L'optimisation robuste devient dans les années 1970 une discipline à elle-seule avec des applications dans des domaines tels que la recherche opérationnelle, la théorie du contrôle, les statistiques, l'économie…

2.6.2. Définition de l'optimisation robuste:

L'optimisation robuste est une branche de l'optimisation mathématique qui cherche à résoudre un problème d'optimisation en prenant en compte les différentes sources d'incertitude de celui-ci.

L'optimisation robuste concerne des situations où différents scénarios sont considérés et où l'objectif est de déterminer une solution qui reste "bonne" quelque soit les scénarios considérés.

2.6.3. Classification de l'optimisation robuste :

Il existe un certain nombre de critères de classification pour des problèmes / modèles d'optimisation robustes. En particulier, on peut distinguer les problèmes liés aux modèles locaux et globaux de robustesse. Et entre des modèles probabilistes et non probabilistes de robustesse**.**

2.6.4. Méthodes d'optimisation robuste :

Ils nécessitent un ensemble de scenarios représentants des réalisations possibles de paramètres aléatoires. Cependant, aucune probabilité n'est associée à ces scenarios. Ces scenarios peuvent être discret, ou encore continus, indiquant un intervalle dans lequel le paramètre aléatoire peut prendre valeur.
Les méthodes d'optimisation les plus courantes sont les modèles min max. Le but de cette mesure, introduite par Kouvelis et Yu (1997) est de minimiser le coût maximum parmi tous les scenarios.

- **Exemple explicatif**

Soit s un ensemble fini de scenarios et x un ensemble fini de solutions réalisables**.**

Soit :$f(x)s$ le cout de la solution x sur le scenario s et *s :$f(x)$ la solution optimale sur ce même scénario. Le modèle min max est alors :

$$Z_s = (\min (\max (f(x))))\ s$$

Cette mesure de robustesse est très conservative, se concentrant principalement sur le scenario de pire cas. La solution trouvée n'a aucune garantie de résultat sur les scenarios de plus faible coût. Cette mesure est donc adaptée aux problèmes d'optimisation avec un adversaire, telle que l'intelligence artificielle**.[10]**

2.7. Quelques problèmes d'optimisation en commande robuste

2.7.1 Modélisation incertaine :

La modélisation et l'obtention du modèle incertain nécessitent un travail complexe qui conditionne en majeure partie le choix de la méthode de résolution des problèmes d'analyse et de synthèse ainsi que son efficacité. Etant donné qu'il s'agit essentiellement d'optimiser le compromis entre la complexité du modèle produit et sa simplicité d'utilisation, cette étape implique dans la majorité des cas la résolution de problèmes d'optimisation.

2.7.2 Synthèse robuste :

Le modèle incertain du système étant défini, il s'agit de calculer une loi de commande compatible avec la structure en contre-réaction de la figure 1 et qui rend le système peu insensible possible aux incertitudes et perturbations pouvant l'affecter tout en permettant d'atteindre un niveau de performances satisfaisant. Cela implique le choix d'une structure de commande par la définition de spécifications fonctionnelles et de classe mais également le choix d'une méthode de synthèse**.[9]**

2.7.3 Analyse robuste :

La problématique générale de l'analyse robuste peut s'énoncer comme suit. Etant donnée une modélisation incertaine et une loi de commande associée, il s'agit d'´etablir si cette loi de commande garantit la stabilité et un certain niveau de performance de la boucle de contre réaction pour toute réalisation du modèle dans son ensemble d'incertitude. A de très rares exceptions, les problèmes d'analyse de la stabilité et de la performance robustes sont résolus complètement.

2.8 Conclusion :

Dans ce chapitre, nous avons présenté des généralités sur l'optimisation et la méthode de recherche stochastique qui est les algorithmes génétiques. L'avantage de cette dernière est l'énorme pouvoir d'être appliquée dans un grand nombre de domaines de recherche de solution (ou tout simplement si personne n'est capable de la trouver de manière théorique). Nous avons aussi introduire la notion d'optimisation robuste et la robustesse.

Dans le chapitre suivant, on introduit la méthode d'optimisation par scenarios.

3.1 Introduction :

La plupart des problèmes d'ingénierie caractérisés par la présence des incertitudes sont formulés, lorsqu'on cherche la meilleure solution, par des problèmes d'optimisation robuste. Deux approches sont utilisées pour la détermination de la solution. La première approche appelée « pire cas », elle cherche à prendre en considération tout le domaine d'incertitude pour inclure le pire cas. Mais cette approche conduit à des problèmes d'optimisation complexes. La deuxième approche probabiliste est basée sur le choix aléatoire des incertitudes à prendre en considération dont le but est de trouver une solution garantissant un bon fonctionnement avec une grande probabilité.

Dans ce chapitre, nous présentons la méthode d'optimisation robuste basée sur les scénarios.

3.2 Optimisation par l'approche des scénarios :

- **On a un problème d'optimisation avec des données incertaines**

Que faire ?

On pourrait optimiser la valeur moyenne de l'objectif par exemple mais on n'a pas forcément connaissance des lois de probabilité suivies par les données. De plus une solution qui optimise une valeur moyenne peut être mauvaise pour certaines réalisations des données.

L'approche robuste consiste à chercher une solution «bonne» quelque soit la réalisation des données.

$$\begin{matrix} \min_x f(x,\theta) \\ \text{sujet à : } x \in D \end{matrix} \quad (1)$$

Où x est le vecteur de variable de décision ou d'optimisation supposé appartenant au domaine admissible *D* défini par les relations de contraintes. θ est le vecteur des paramètres incertaines. On suppose que chaque paramètre est borné, c'est-à-dire

$$\theta_{min} \leq \theta \leq \theta_{max}$$

Ce problème peut être formulé par un problème min max comme suit :

$$\min_{x} \max_{\theta} f(x, \theta)$$

3.3 Principe des scénarios

3.3.1 Définition :

L'approche de scénario ou l'approche d'optimisation de scénario est une technique pour obtenir des solutions à une optimisation robuste , basés sur un échantillon de contraintes. La technique existe depuis des décennies comme une approche heuristique et a récemment reçu une base théorique systématique.

3.3.2 Description :

En optimisation, les fonctionnalités de robustesse se transforment en contraintes paramétrées dans les éléments incertains du problème. La méthode de scénario consiste simplement à extraire au hasard certaines instances de l'incertitude, puis à trouver la solution optimale d'un problème où se bornent les contraintes associées aux instances d'incertitude extraites. La théorie indique à l'utilisateur comment cette solution est "robuste", c'est-à-dire si et dans quelle mesure la solution trouvée satisfait les contraintes qui se produisent pour les

Pour résoudre le problème d'optimisation robuste (1), on suit les étapes suivantes :
On introduit une variable y et on sélectionne N scénarios pour les incertitudes, c'est-à-dire N valeurs pour θ. Le problème (1) prend la forme suivante :

$$sujet\ à : \begin{matrix} \min\limits_{x,y} y \\ f(x, \theta_i) \leq y, \qquad i = 1, N \end{matrix}$$

Le choix du nombre N et la connaissance de la loi de distribution des incertitudes sont très importants pour la détermination d'une solution fiable du problème.

Si la loi de distribution des incertitudes est connue, les N valeurs d'incertitudes sont extraits selon cette loi. Sinon, des générateurs de nombres aléatoires ou les systèmes chaotiques peuvent être utilisés.

3.4 Génération des nombres aléatoires :

Un des premiers problèmes auxquels on est confronté que l'on veut utiliser des méthodes probabilistes sur un ordinateur est celui de la générantion de nombres aléatoires. En effet, le fonctionnement d'un ordinateur est par nature déterministe. Il est cependant nécessaire de générer des réalisations de variables aléatoires selon des lois de probabilités variées et non déterminées à l' avance.

- **Que-ce qu'une variable aléatoire ?**

Une variable aléatoire est une variable dont la suite des valeurs n'est pas prédictible. Aussi longue que soit la série générée, il n'est pas possible de trouver une équation permettant de $x_{1,}$prédire le reste de la série à partir de celles déjà générées.

Par exemple $x_{,}$est une variable aléatoire. Et $x_1, x_2, x_3, \cdots, x_i$ est la suite des valeurs générées successivement. Toute variable aléatoire suit une loi de probabilité. Cette loi définit la répartition des valeurs au sein d'un intervalle. Par exemple, la loi « uniforme » définit une répartition uniforme des valeurs dans un intervalle, c'est-à-dire que toutes les valeurs de l'intervalle ont la même probabilité d'apparaître dans une longue série.

3.4.1 Source aléatoire :

Un problème centrale pour la simulation de Monte Carlo est celui de la génération de du l'aléatoire. Comment générer du hasard sur un ordinateur ?

Il ya plusieurs objectifs pour la génération de nombres aléatoire :

1) L'objectif prioritaire est que la suite générée suive les lois de probabilité souhaitées (de manière prouvée de préférence) ;

2) On peut vouloir que la suite générée ne soit pas reproductible. C'est le cas pour tous les problèmes de sécurité, ainsi que pour les lotries.On peut au contraire
3) vouloir pouvoir reproduire la suite, afin de la réutiliser pour faire des comparaisons de systèmes en simulation ;

La simulation de Monte Carlo recherchera plutôt la reproductibilité, et des preuves de bonnes propriétés**.[8]**

On souhaite que la génération soit Une simulation utilisant des nombres aléatoires est appelée simulation de Monte Carlo. Cette dernière peut donc être vue comme une méthode d'approximation statistique par opposition à l'analyse numérique classique, qui n'utilise pas le hasard.

3.5 Méthode de Monte Carlo :

On appelle méthode de Monte Carlo toute méthode visant à calculer une valeur numérique, et utilisant des procédés aléatoires, c'est-à-dire des techniques probabilistes. Le nom de ces méthodes fait allusion aux jeux de hasard pratiqués à Monte Carlo, a été inventé en 1947 par Nicholas Metropolis, et publié pour la première fois en 1949 dans un article coécrit avec Stanislaw Ulam.

3.5.1 Principe de la méthode de Monte Carlo :

La méthode de Monte Carlo est l'une des principales applications mathématiques et statistiques utilisant les nombres aléatoires. Cette méthode a pour objet de fournir une approximation numérique de la valeur d'une intégrale (quelque soit la dimension) d'une fonction donnée pourvu qu'elle soit intégrale.

3.5.2 Avantage de la méthode de Monte Carlo :

Les méthodes de Monte-Carlo permettent d'estimer numériquement des résultats grâce à l'usage de valeurs aléatoires ; elles servent à éviter les calculs exacts (elles ont l'avantage de réduire le temps de calculs ou encore d'approximer un résultat dont le calcul exact est impossible à réaliser). Plus précisément, la méthode de Monte-Carlo ordinaire utilise des variables indépendantes et identiquement distribuées.

3.6 Génération d'une suite i.i.d. uniforme sur [0,1] :

Nous décrivons dans cette section quelques exemples de générateurs pseudo-aléatoire : la famille historique des générateurs congruentiels linéaires, puis trois familles de générateurs connus comme les plus efficaces.

- **Variables indépendantes et identiquement distribuées (i.i.d) :**

Sont des variables aléatoires qui suivent toutes la même loi de probabilité et sont indépendantes. On dit que ce sont des variables aléatoires i.i.d ou plus simplement des variables i.i.d.

3.6.1 Générateur de nombre pseudo-aléatoire :

Un générateur de nombre pseudo aléatoire, est un algorithme qui génère une séquence de nombres présentant certaines propriétés du hasard. Ses propriétés statistiques correspondent à celles d'un échantillon de variables aléatoires.

Les méthodes pseudo-aléatoires sont souvent employées sur des ordinateurs, dans diverses tâches comme la méthode de Monte-Carlo, La plupart des algorithmes pseudo-aléatoires essaient de produire des sorties qui sont uniformément distribuées. Une classe très répandue de générateurs utilise une congruence linéaire.

3.6.1.1 Générateur congruentiels linéaires :

La famille la plus connue de générateurs, et peut être encore la plus utilisée, est celles des générateurs congruentiels linéaires plus communément appelée générateurs LCG en abrégé. L'idée de base est particulièrement simple. Il s'agit de générer une suite $(x_n)n \geq 0$ d'entiers sur $\{0, \cdots, m\}$, de manière récursive, par une valeur x_0 et $\forall n \geq 0$,

$$x_n = a_1 x_{n-1} + a_2 \; mod \; m \qquad (3.1)$$

Où $mod\ m$ est le reste de la division par m. x_0 est appelé le germe, m est appelé le terme de congruence, a_1 le multiplicateur et a_2 l'incrément.

La suite de nombre aléatoire uniformes est donné par la suite suivante :

$$u_n = \frac{x_n}{m} \qquad \forall n \geq 0 \qquad (3.2)$$

Les coefficients (a_1, a_2, m) définissent complètement un (LCG) et doivent obéir à certaines contraintes pour que le LCG produise des nombres aléatoires de qualité satisfaisantes.

Cette famille de générateur s de nombres aléatoires, a trois principales qualités :

1. Les LCG sont très simples à implémenter.
2. Les LCG s'appuient sur un processus reproductible via la graine.
3. Les LCG sont rapides par rapport à d'autres générateurs de nombres aléatoires.

   Ces trois qualités suffisent à expliquer la présence encore très importante des LCG dans les implémentations de langages.

✓ **Exemple 3.3.**

Choisir les valeurs a_1, a_2 et m <<au hasard>> n'est pas une bonne idée. Prenons un exemple, soit le générateur congruentiels linéaire employant les valeurs $a_1 = 25$, $a_2 = 16$, $m = 256$.

Pour $x_0 = 17$, les nombres produits sont : 17, 185, 33, 73, 49, 217, 65, 105, 81, 249. Ils sont tous impair !

Pour $x_0 = 120$, les nombres produits sont : 120, 200, 152, 232, 184, 8, 216, 40, 248.

Ils sont tous pair !

Pour $x_0 = 10$, les nombres produits sont : 10, 10, 10, 10, 10, 10, 10, 10, 10,10.

Cela produit toujours la même valeur.

Certes La formule est simple mais le choix des trois paramètres doit répondre à certains critères. Si on choisi a_2 non nul, il est toujours possible d'obtenir une période de longueur m (donc pas de risque de blocage puisqu'on va retrouver tous les nombres entre 0 et $m - 1$ dans la suite). D. Knuth fait la démonstration des critères que doivent remplir a_1, $a_2$et m pour cela :

- $a_2$et m doivent être premiers entre eux

- $a_1 - 1$ doit être un multiple de p, pour tout p nombre premier diviseur de c
- $a_2 - 1$ doit être un multiple de 4 si c est un multiple de 4.
- si m est une puissance de 2, le bit de poids faible des nombres produits vaut alternativement 0 et 1 (ce n'est d'ailleurs pas le seul cas où cela se produit).

3.6.1.2 Problème rencontré par les générateurs congruentiels linéaires :

Ces générateurs congruentiels linaires peuvent rencontrer des difficultés si l'on cherche a utiliser des vecteurs dont les coordonnées sont des points de la suite pseudo-aléatoire $(u_n)_{n\geq 0}$ .Un problème particulier est appelé le problème des anniversaire.

Il consiste à décomposer $[0,1]^2$ en k blocs (ordonnés) carrés de taille égale et à générer *n* vecteurs bidimensionnels $v_i=(u_{2i}, u_{2i+1})$ pour $0 \leq i<n$.Chaque v_i est dans un des blocs .On peut voir les blocs sélectionnés comme les dates d'anniversaires des *n* personnes, pour une année comprenant *k* jours.

Des générateurs ayant de meilleurs propriétés et considérés parmi les meilleurs sont les deux suivants

- **Générateurs MRG32k3a :**

Le premier des trois algorithmes que nous présentons a été proposé par Pierre L'Ecuyer en 1999.Le générateur est basé sur le MRG, et les combine de manière à être rapide en termes de temps d'exécution, mais procure également un gain en termes de période, tout en ayant des propriétés statistiques intéressantes. On considère deux MRG

$$x_n=a_1x_{n-1}+a_2x_{n-2}+a_3x_{n-3} \bmod m_1 \qquad (3.3)$$

$$y_n =b_1y_{n-1}+b_2y_{n-2}+b_3y_{n-3} \bmod m_2 \qquad (3.4)$$

Et la suite de nombres $(u_n)_{n\geq 0}$ dans [0,1] est donnée par la différence modulo 1 des deux suites générées par les deux MRG précédents

$$u_n=x_n|m_1\text{-}y_n|m_2 \bmod 1 . \qquad (3.5)$$

- **Générateur *Mersenne –Twister :***

Le générateur ***Mersenne –Twister*** a été propose 1998, notamment pour résoudre les problèmes de génération multidimensionnelles.

Soit $\boldsymbol{x_k}(\forall \boldsymbol{k} \geq \mathbf{0})$ et **a** des vecteurs lignes (ou mots) avec coordonnées dans {0,1} et de dimension (ou longueur) *w*, chaque élément correspondant à un bit du développement en base 2 d'un entier. Un mot forme donc un nombre entre 0 et 2^w-1 ,et en le divisant par 2^{w-1}On obtiendra un nombre entre 0 et 1.

$$\boldsymbol{X}_{k+n} = \boldsymbol{X}_{k+m} \oplus (\boldsymbol{X}_{\boldsymbol{k}}^{u} | \boldsymbol{X}_{\boldsymbol{k+1}}^{l}) A \qquad (3.6)$$

n est appelé le degré de la récurrence et $1 \leq m < n$;

$\oplus$ est l'addition bit-à-bit modulo 2 ;

A est une matrice $w \times w$ avec éléments dans {0,1} de la forme ;

$$A = \begin{bmatrix} & 1 & & & \\ & & 1 & & \\ & & & \ddots & \\ & & & & 1 \\ \boldsymbol{a}_{w-1} & \boldsymbol{a}_{w-1} & \boldsymbol{a}_{w-1} & \dots & \boldsymbol{a}_0 \end{bmatrix};$$

Le choix de la matrice A est tel que le produit **x**A utilise des opérations rapides sur les bits :**x**A décale mot a droite si $\boldsymbol{x}_0$=0,noté $\mathbf{x}' = (0 x_{w-1.} \dots . x_1)$,et effectue $\mathbf{x}' \oplus \mathbf{a}$ si $\boldsymbol{x}_0 = 1$ .De la même manière ,les opérations de la récurrence sont des opérations bit-à-bit très rapide.

Afin de générer une suite avec une meilleure distribution multidimensionnelle, chaque mot généré est multiplié à droite par une matrice $w \times w$ inversible *T* pour améliorer la répartition des différents bits, notamment les moins significatifs. Il est conseillé d'utiliser finalement **z=x**T en prenant pour T la composition successive des opérations :

$y = \mathbf{x} \oplus (\mathbf{x} \gg u)$

$y = \mathbf{y} \oplus ((\mathbf{x} \ll u) \mathbf{AND}\ \boldsymbol{b})$

$y = \mathbf{y} \oplus ((\mathbf{x} \ll u) \mathbf{AND}\ \boldsymbol{c})$

$$y = \mathbf{x} \oplus ((\mathbf{x} \gg l), \gg$$

avec l,s,t,u entiers, $(\mathbf{x} \gg u)$ (respectivement $(\mathbf{x} \ll u)$) est le *shift* à droite (respectivement à gauche)de u bits.

3.7 Conclusion

Dans ce chapitre, nous avons présenté la méthode des scénarios utilisée pour la résolution des problèmes d'optimisation robuste. L'idée de cette méthode consiste à convertir du type minimax en un autre problème d'optimisation avec un ensemble fini de contraintes. Ces contraintes sont sélectionnées de manière aléatoire en considérant un certain nombre de scénarios pour les incertitudes.

Dans le chapitre qui suit, on s'intéresse à la synthèse d'un régulateur PID en utilisant l'approche des scénarios.

4.1 Introduction :

Le régulateur le plus utilisé dans l'industrie est le régulateur PID (proportionnel intégrale dérivé), car il permet de régler un grand nombre de grandeurs physiques. Son réglage consiste à déterminer les coefficients k_c, T_i etT_d afin d'obtenir une réponse adéquate du procédé et de la régulation.

Ce dernier chapitre consiste à trouver les paramètres du régulateur PID en minimisant le critère de performance de l'intégrale de l'erreur quadratique en satisfaisant les contraintes imposés. Premièrement on va présenter les critères de performances, et par la suite on passe à la formulation du problème de synthèse d'un correcteur PID.

4.2 Régulation optimale :

4.2.1 Représentation :

Le problème de la régulation optimale consiste à trouver la valeur optimale des paramètres d'un régulateur dont la structure est fixée à l'avance. L'intérêt majeur de cette pratique réside dans le fait d'imposer des contraintes sur la forme du régulateur et le nombre de coefficients de réglage, par exemple un PID série avec un rapport imposé entre les constantes de temps d'intégration et de dérivation.

En ce qui concerne les critères d'optimisation, ils portent généralement sur une variable d'écart $e(t)$ qui tend asymptotiquement vers 0.

4.2.2 Critères de performance :

En régulation automatique, on distingue les critères de performances suivants :

- Critère IAE (Intégrale de la valeur absolue de l'erreur pondérée)

Son expression est :

$$J = \int_0^{\infty} |e(t)| dt$$

Ce critère prend en compte tous les éléments de la réponse harmonique et donc J est important, si la réponse est oscillatoire.**[5]**

- Critère ITAE (Intégrale de la valeur absolue de l'erreur pondérée par le temps)

Son expression est :

$$J = \int_0^\infty t|e(t)|dt$$

Ce critère minimise les dépassements.

- Critère IES (Intégrale de l'erreur quadratique)

Son expression est :

$$J = \int_0^\infty e(t)^2$$

Ce critère minimise l'énergie de réglage.

- Critère ITES (Intégrale de l'erreur quadratique pondérée par le temps)

Son expression est :

$$J = \int_0^\infty te(t)^2 dt$$

- Critère EITES (Intégrale de l'extension de l'erreur quadratique pondérée par le temps)

Son expression est :

$$J = \int_0^\infty t^q \, e(t)^2 dt, q \geq 0$$

Sans être restrictif, on se place dorénavant dans la situation décrite par le schéma fonctionnel de la **Figure 4.1** où l'écart e est l'erreur en réponse à une consigne échelon.

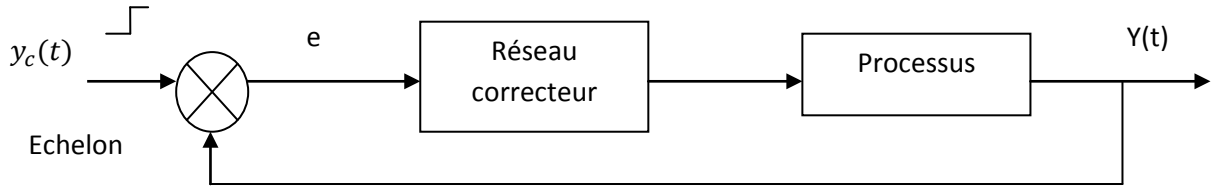


Figure 4.1 Optimisation indicielle

Les éléments de l'asservissement étant notés par les transferts suivants :

- Processus :$F(p)$;
- Réseau correcteur :$RC(p)$;
- Chaîne d'action : $G(p) = F(p)\ RC(p)$;
- Système en boucle fermée : $W(p) = \frac{F(p)RC(p)}{1+F(p)RC(p)}$

  On obtient le transfert entre la consigne et l'erreur :

$$H(p) = \frac{1}{1+F(p)RC(p)} \qquad (4.1)$$

Il est donc nécessaire, pour que cette erreur tende asymptotiquement vers 0, que ce transfert ait un gain statique nul, donc que la chaine d'action comporte au moins une intégration pure.

De plus et nous ne le montrons pas ici mais on le supposera toujours, le fait de considérer ce genre de critères impose un asservissement asymptotiquement stable, ce qui permet de leur assurer des valeurs finies. Pour cela, on suppose que le correcteur possède une action intégrale.

Remarque : 4.1

- IAE et ITSE sont moins faciles à utiliser.

4.3. Traitement du critère IES :

Ce critère peut être calculé à partir de deux approches : l'approche transfert ou l'approche d'état :

4.3.1. L'approche transfert (approche polynomiale) :

Si l'on suppose que $e(t)$ admet pour transformée de Laplace la fonction rationnelle :

$$E(p) = \frac{B(p)}{A(p)} = \frac{b_0+b_1p+\cdots+b_{n-1}p^{n-1}}{a_0+a_1p+\ldots+a_{n-1}p^{n-1}+a_np^n} \quad (4.2)$$

Alors la valeur du critère est donnée par :

$$J = \frac{1}{2\pi j}\int_{-j\infty}^{+j\infty} E(p)E(-p)\,dp \quad (4.3)$$

Le calcul de cette intégrale complexe peut être réalisé par la méthode d'Aströme qui utilise la construction de tableaux de Routh-jury.

A partir des polynômes $A(p)$ et $B(p)$ construisons les tableaux suivants :

Tableau du numérateur

$b_{n,n-1}$	$b_{n,n-2}$	$\cdots$	$b_{n,k-1}$	$\cdots$	$b_{n,0}$
	$b_{n-1,n-2}$	$\cdots$	$b_{n-1,k-1}$	$\cdots$	$b_{n-1,0}$
		$\ddots$	$\vdots$	$\cdots$	$\vdots$
			$b_{k,k-1}$	$\cdots$	$b_{k,0}$
				$\ddots$	$\vdots$
					$b_{k,0}$

Tableau4.2 tableau dénominateur

Tableau du dénominateur

$a_{n,n}$	$a_{n,n-1}$	$a_{n,n-2}$	$\cdots$	$a_{n,k-1}$	$\cdots$	$a_{n,0}$
	$a_{n-1,n-1}$	$a_{n-1,n-2}$	$\cdots$	$a_{n-1,k-1}$	$\cdots$	$a_{n-1,0}$
		$\ddots$		$\vdots$		
			$a_{k,k}$	$a_{k,k-1}$	$\cdots$	$a_{k,0}$
				$a_{k-1,k-1}$	$\cdots$	$a_{k-1,0}$
					$\ddots$	
						$a_{0,0}$

Tableau4.1 tableau du numérateur

A l'aide des récurrences suivantes :

- Initialisation : $k = 0, \cdots, n-1, a_{n,k} = a_k, b_{n,k} = b_k, a_{n,n} = a_n$;
- Récurrence :

$$i = k-1, \cdots, 0, a_{k-1,j} = \begin{cases} a_{k,i}\ si\ k-1-i\ est\ pair; \\ a_{k,i} - \dfrac{a_{k,k}}{a_{k,k-1}} a_{k,i-1}\ si\ k-1-i\ est\ impair; \end{cases}$$

$$i = k-2, \cdots, 0, b_{k-1,i} = \begin{cases} b_{k,i} sik-2-iestpair\ ; \\ b_{k,i} - \dfrac{b_{k,k-1}}{a_{k,k-1}} a_{k,isik-2-iestimpair}; \end{cases}$$

Alors on obtient :

$$J = \sum_{k=1}^{n} \frac{b_{k,k-1^2}}{2a_{k,k}a_{k,k-1}} \tag{4.4}$$

Notons que dans le cas où le système $F(p)RC(p)$ est connu par son transfert :

$$F(p)RC(p) = \frac{N(p)}{p^{\alpha}D^*(P)}, N(0)D^*(0) \neq 0 \tag{4.5}$$

Alors, d'après ce qui précède, on obtient :

$$E(p) = \frac{B(p)}{A(p)} = \frac{p^{\alpha-1}D^*(P)}{N(p)+p^{\alpha}D^*(P)} \tag{4.6}$$

4.3.2. L'approche d'état :

Si l'on suppose qu'il existe une réalisation de $e(t)$ sous la forme :

$$\dot{z}(t) = \Phi z(t), z(0) = z_0$$

$$e(t) = \Theta z(t)$$

Dans la quelle Φ et $\boldsymbol{\Theta}$représentent respectivement une matrice carrée et une matrice ligne constante, alors la valeur du critère IES est donnée par :

$$J = -z_0^T \Pi z_0$$

Où Π est solution symétrique définie négative de l'équation de Lyapunov :

$$\Phi^T \Pi + \Pi\Phi = \Theta^T\Theta$$

Ce résultat vient du fait que si $\varphi(t)$tend vers 0 lorsque t tend vers l'infini, alors :

$$\int_0^{\infty} \varphi(t)\mathrm{dt} = -\ \varphi(0)$$

Et il suffit de prendre $\varphi(t) = z(t)^T\ \Pi z(t)$ pour obtenir le résultat recherché.
Notons que, à partir de $A(p)$ et $B(p)$, une équation d'état qui réalise $E(p)$ peut être obtenue sous la forme :

$$\Phi = \begin{bmatrix} 0 & \cdots & 0 & -\frac{a_0}{a_n} \\ 1 & & & -\frac{a_1}{a_n} \\ \ddots & & & \vdots \\ & & 1 & -\frac{a_{n-1}}{a_n} \end{bmatrix}, \quad z_0 = \begin{bmatrix} -\frac{b_0}{a_n} \\ -\frac{b_1}{a_n} \\ \frac{-b_{n-1}}{a_n} \end{bmatrix} \quad \Theta = [0 \cdots 0\ 1]$$

4.4 Exemple d'utilisation :

Afin d'illustrer les deux méthodes approches, on se propose d'étudier l'exemple du double intégrateur $\frac{1}{p^2}$ commandé par le réseau correcteur de la forme $k\frac{1+aTp}{1+Tp}$ pour une entrée de type échelon unitaire.

Le schéma bloc de la rétroaction est comme suit :

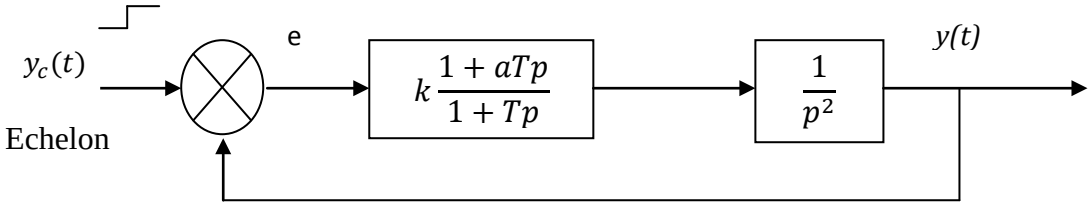


Figure 4.2 Commande d'un double intégrateur.

Le calcul de l'expression de l'erreur de E(p) donne :

$$H(p) = \frac{C(p)G(p)}{1+C(p)G(p)} = \frac{k(1+aTp)}{Tp^3+p^2+kaTp+k} \quad (4.8)$$

On déduit l'expression de l'erreur :

$$E(p) = \frac{p+Tp^2}{k+akTp+p^2+Tp^3} \quad (4.9)$$

Cela permet de construire les tableaux de Routh-Jury suivants en utilisant les relations de récurrences précédentes.

T	1	akT	k
	1	$(a-1)kT$	k
		$(a-1)kT$	k
			k

Tableau 4.3 Dénominateur

T	1	0
	1	$-kT$
		$-kT$

Tableau 4.4 Numérateur

Pour utiliser l'approche par espace d'état, on peut construire la réalisation de $e(t)$ suivante :

$$\dot{z}(t) = \begin{bmatrix} 0 & 0 & -\frac{k}{T} \\ 1 & 0 & -ak \\ 0 & 1 & -\frac{1}{T} \end{bmatrix} z(t)$$

$$z_0 = \begin{bmatrix} 0 \\ -\frac{1}{T} \\ -1 \end{bmatrix}$$

$$e(t) = [0 \quad 0 \quad 1]\, z(t)$$

- Cette méthode conduit à résoudre l'équation de Lyapunov :

$$\Pi\begin{bmatrix}0 & 0 & -\frac{k}{T}\\ 1 & 0 & -ak\\ 0 & 1 & -\frac{1}{T}\end{bmatrix}+\begin{bmatrix}0 & 0 & -\frac{k}{T}\\ 1 & 0 & -ak\\ 0 & 1 & -\frac{1}{T}\end{bmatrix}^{T}\Pi=\begin{bmatrix}0 & 0 & 0\\ 0 & 0 & 0\\ 0 & 0 & 1\end{bmatrix}$$

Dont la solution symétrique définie négative s'écrit :

$$\Pi=\frac{T}{2k(1-a)}\begin{bmatrix}\frac{1}{k} & 0 & -1\\ 0 & 1 & 0\\ -1 & 0 & ak\end{bmatrix}$$

Il en résulte la valeur du critère :

$$J=-z_0^T\Pi z_0=\frac{1+akT^2}{2(a-1)kT} \qquad (4.10)$$

La recherche d'un minimum de ce critère par rapport à T conduit, lorsque $0<ak<1$, à la valeur minimale :

$$J=\frac{a}{a-1}$$

4.5 Optimisation du critère :

La détermination des paramètres optimaux (a, k ,T) du correcteur revient à résoudre un problème d'optimisation qui consiste à minimiser le critère (4.9) soumis à la contrainte

$0{,}1<ak<0.9$, formulé comme suit :

$$min\frac{1+akT^2}{2(a-1)kT}$$

Le problème est un problème d'optimisation avec contraintes inégalités, pour cela, en propose de vérifier les conditions de Kuhn Tucker pour la recherche de la solution. Le Lagrangien de ce problème est donné comme suit :

$$L(a,k,T,u_1,u_2) = \frac{1+akT^2}{2(a-1)kT} + u_1(ak-0{,}9) + u_2(0{,}1-ak) \quad (4.11)$$

Les conditions de Kuhn Tucker d'après le **tableau 2.1**

$$\nabla_{a,k,T}\ \mathrm{L}(\mathrm{a},\mathrm{k},\mathrm{T})=0 \quad (4.12)$$

$$\nabla_u \mathrm{L}(\mathrm{a},\mathrm{k},\mathrm{T},u_1,u_2) \leq 0 \quad (4.13)$$

$$u(ak-1)=0 \quad (4.14)$$

$$u \geq 0 \quad (4.15)$$

Avec :

$$\nabla_a \mathrm{L}(a,\mathrm{k},\mathrm{T},u_1,u_2) = \frac{-kT^2-1}{2kT(a-1)^2} + u_1\mathrm{k} - u_2\mathrm{k} = 0 \quad (4.16)$$

$$\nabla_k\ \mathrm{L}(a,\mathrm{k},\mathrm{T},u_1,u_2) = \frac{-1}{2(a-1)k^2T} + u_1 a - u_2 a = 0 \quad (4.17)$$

$$\nabla_T\ \mathrm{L}(a,\mathrm{k},\mathrm{T},u_1,u_2) = \frac{akT^2-1}{2(a-1)kT^2} = 0 \quad (4.18)$$

$$\nabla_{u_1}\ \mathrm{L}(a,\mathrm{k},\mathrm{T},u_1,u_2) = ak - 0.9 \leq 0 \quad (4.19)$$

$$\nabla_{u_2}\ \mathrm{L}(a,\mathrm{k},\mathrm{T},u_1,u_2) = 0.1 - ak \leq 0 \quad (4.20)$$

$$u_1\,(a\mathrm{k}-0.9)=0 \quad (4.21)$$

$$u_2\,(0.1-a\mathrm{k})=0 \quad (4.22)$$

$$u_1 \geq 0 \quad (4.23)$$

$$u_2 \geq 0 \quad (4.24)$$

La résolution de ce problème conduit à la solution suivante :

a_{opt}= 5.5227

k_{opt}= 0.1620

T_{opt}= 1.0733

Et la valeur minimale du critère : J= 1.2911

4.6 Utilisation de l'algorithme génétique dans la synthèse des correcteur PID :

Considérons le système du deuxième ordre décrit par la fonction de transfert :

$$G(s)=\frac{K}{(as+1)(bs+1)} \qquad (4.25)$$

Pour contrôler ce système, on propose d'utiliser un correcteur PID de structure mixte de fonction de transfert :

$$C(s)=k_c\left(1+\frac{1}{T_i S}+T_d s\right) \qquad (4.26)$$

La fonction de transfert du système en boucle fermé est :

$$H(s)=\frac{C(s)G(s)}{1+C(s)G(s)} \qquad (4.27)$$

L'expression de l'erreur est:

$$E(s)=\frac{T_i abs^2+T_i(a+b)s+T_i}{T_i abs^3+(T_i(a+b)+Kk_c} \qquad (4.28)$$

A partir des deux polynômes associes au numérateur et au dénominateur de $E(p)$, on construit les deux tableaux de Routh-Jury suivants :

a_3	a_2	a_1	a_0
a_2	0	a_0	0
	a_2	$a_1 - \frac{a_0 a_3}{a_2}$	a_0
	$a_1 - \frac{a_0 a_3}{a_2}$	0	0
		$a_1 - \frac{a_0 a_3}{a_2}$	a_0
		a_0	0
			a_0

Tableau 4.5. Tableau du dénominateur de $E(p)$

b_2	b_1	b_0
a_2	0	a_0
	$b_0 - \frac{a_0 b_2}{a_2}$	$b_0 - \frac{a_0 b_2}{a_2}$
	$b_0 - \frac{a_0 b_2}{a_2}$	0
		$b_0 - \frac{a_0 b_2}{a_2}$

Tableau 4.6. Tableau du numérateur de $E(p)$

$$E(s) = \frac{T_i abs^2 + T_i(a+b)s + T_i}{T_i abs^3 + T_i\left((a+b) + Kk_c T_d\right)s^2 + T_i(1+Kk_c)s + Kk_c} \qquad (4.29)$$

Afin de trouver le critère associé à l'erreur quadratique, on généralise l'expression de $E(s)$, puis on identifie chaque fonction de transfert au critère trouvé par l'approche polynomiale.

Ainsi, pour le problème de régulation considéré, l'erreur est donnée sous la forme suivante

$$E(s) = \frac{b_2 s^2 + b_1 s + b_0}{a_3 s^3 + a_2 s^2 + a_1 s + a_0} \tag{4.30}$$

et l'utilisation de la méthode Routh-Jury, on obtient la valeur du critère suivant :

$$J = \frac{b_0^2}{2a_0 a_1} + \frac{b_1^2}{2a_1 a_2} + \frac{b_2^2}{2a_1 a_2} \tag{4.31}$$

Et par identifications on a :

$$J = \frac{T_i(a+b+Kk_cT_d)+(a+b)^2Kk_c+ab(1+Kk_c)Kk_c}{2Kk_c(1+Kk_c)(a+b+Kk_cT_d)} \tag{4.32}$$

4.7 Formulation du problème d'optimisation :

On remarque que la le critère dépend des paramètres du système a, b et K. Ces paramètres sont supposés incertains, c'est-à-dire :

a = [0.5 1]
b = [1 2]
K = [0.5 1.5]

Donc on considère N scénarios pour ces paramètres. Pour le [i-ème] scénarios, la valeur du critère est donnée comme suit :

$$\boldsymbol{J_i = \frac{T_i(a_i + b_i + K_i k_c T_d) + (a_i + b_i)^2 K_i k_c + a_i b_i (1 + K_i k_c) K_i k_c}{2K_i k_c (1 + K_i k_c)(a_i + b_i + K_i k_c T_d)}}$$

Alors le problème d'optimisation à résoudre est donné comme suit

$$\min_{k_c,Ti,Td} J = \sum_{i=1}^{N} J_i = \sum_{i=1}^{N} \frac{\boldsymbol{T_i(a_i + b_i + K_i k_c T_d) + (a_i + b_i)^2 K_i k_c + a_i b_i (1 + K_i k_c) K_i k_c}}{\boldsymbol{2K_i k_c (1 + K_i k_c)(a_i + b_i + K_i k_c T_d)}}$$

Pour garantir une solution réalisable, on doit imposer les contraintes suivantes :

$$\begin{cases} -k_c \leq 0 \\ -T_i \leq 0 \\ -T_d \leq 0 \\ T_d \leq T_i \end{cases}$$

Les trois contraintes imposées permettent d'assurer que le gain et les constantes de temps sont positifs. La dernière contrainte assure le compromis rapidité.

En résumé le problème d'optimisation à résoudre est donné comme suit :

$$\min_{k_c,Ti,Td} J = \sum_{i=1}^{N} J_i = \sum_{i=1}^{N} \frac{\boldsymbol{T_i(a_i + b_i + K_i k_c T_d) + (a_i + b_i)^2 K_i k_c + a_i b_i (1 + K_i k_c) K_i k_c}}{\boldsymbol{2K_i k_c (1 + K_i k_c)(a_i + b_i + K_i k_c T_d)}}$$

$Sujet$ à :

$$\begin{aligned} -k_c &\leq 0 \\ -T_i &\leq 0 \\ -T_d &\leq 0 \\ T_d &\leq T_i \end{aligned}$$

Le nombre N utilisé est calculé par la formule suivante :

$$\mathrm{N} \geq \frac{2}{\epsilon}(\ln\frac{1}{\beta} + n)$$

Avec :

Paramètre de risque : $\epsilon = 0.01$

Paramètre de confiance : $\beta = 10^{-10}$

Nombre de variable de décision (k_1, k_2, k_3) : $n = 3$

Ce qui donne N= 5206 et les différents scénarios ont générés en utilisant la fonction randn de Matlab.

La résolution de ce problème est obtenue par les algorithmes génétiques qui conduisent aux paramètres optimaux suivants :

$k_{c_{opt}} = 0.0167$

$T_{i_{opt}} = 0.0353$

$T_{d_{opt}}$=1.8796 10^{-10}

Pour évaluer les performances du correcteur PID obtenu, nous avons réalisé des simulations pour ensemble d'incertitudes non pris en considération lors de la synthèse. Les résultats obtenus sont donnés par la figure 4.2. On constate que malgré les incertitudes, le système reste stable en boucle fermée. Ceci démontre la robustesse en stabilité du correcteur. De même le régime transitoire reste peu affecté.

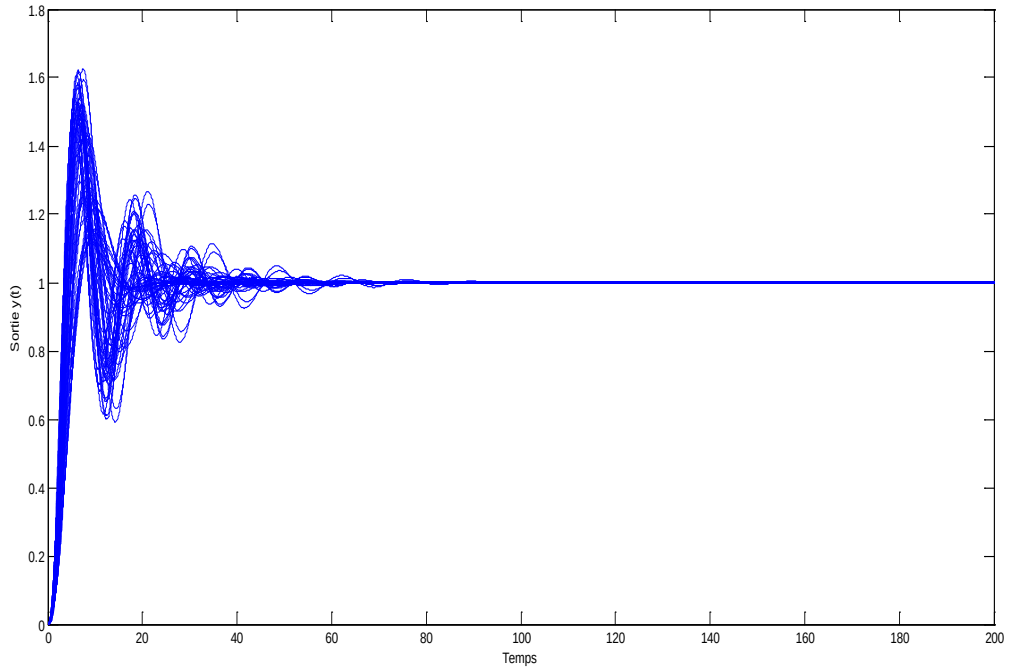


Figure 4.3 Réponse du système en boucle fermée pour différentes incertitudes.

4.8 Conclusion :

Dans ce chapitre, nous avons présenté une approche permettant de synthétiser un correcteur PID robuste pour un système incertain. L'idée consiste à minimiser un critère de performances défini comme la somme des critères de performances pour l'ensemble d'incertitudes. Cet ensemble est généré en utilisant la méthode des scénarios. La recherche de la solution du problème d'optimisation est déterminée en utilisant les algorithmes génétiques. Les performances du correcteur sont évaluées dans le cas d'un système de deuxième ordre à trois paramètres incertains. Les résultats obtenus démontrent la robustesse du correcteur PID.

Le travail présenté dans ce mémoire concerne la synthèse de correcteurs robuste pour les systèmes incertains. L'objectif du travail consiste à utiliser la méthode des scénarios, une méthode d'optimisation robuste, pour la détermination des paramètres optimaux du correcteur en dépit des incertitudes paramétriques.

Après avoir donné des généralités sur les systèmes incertains et de l'optimisation classique en particulier l'optimisation robuste, nous avons donné le principe général de la méthode des scénarios. Cette méthode a été exploitée par la suite pour la conception d'un correcteur PID optimale pour un système incertain.

Ainsi, un exemple illustratif est présenté. Il s'agit d'un système de second ordre caractérisé par trois paramètres incertains. L'objectif est de concevoir un correcteur PID robuste. Le problème est formulé comme un problème d'optimisation mutli-objectif. Chaque objectif consiste à optimiser un critère de performance pour un scénario donné.

Les résultats obtenus démontrent clairement que le correcteur PID obtenus est peu sensible aux variations paramétriques.

ANNEXE

Le schéma bloc de la rétroaction est comme suit :

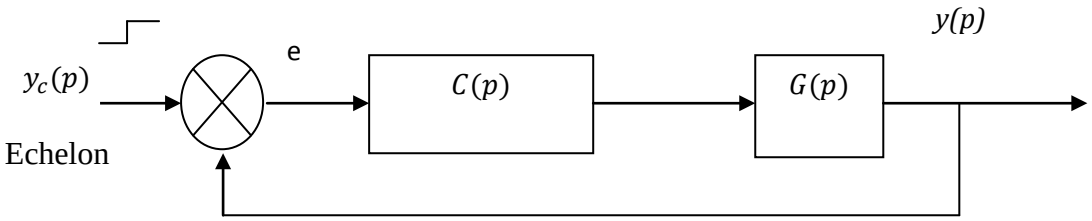


Figure A.1 schéma fondamental d'un système à rétroaction

on a:

$$\frac{y(p)}{y_{c(p)}} = \frac{c(p)G(p)}{1+c(p)G(p)} \quad \text{..... (A.1)}$$

et

$$\frac{E(p)}{y_c(p)} = \frac{1}{1+C(p)G(p)} \text{.....(A.2)}$$

et pour calculer l'erreur:

$$\frac{E(p)}{y_c(p)} = \frac{1}{1+C(P)G(p)} \text{.....(A.3)}$$

et dans notre exemple on a du chapitre 4

$C(p) = k\frac{1+aTp}{1+Tp}$ et $G(p) = \frac{1}{p^2}$

$$\frac{E(p)}{y_c(p)} = \frac{1}{1+C(P)G(p)}$$

$$= \frac{1}{1+k\frac{1+aTp}{1+Tp}\frac{1}{p^2}} \text{.....(A.4)}$$

$$\frac{E(p)}{y_c(p)} = \frac{TP^3+P^2}{TP^3+P^2+akTp+k} \text{..... (A.5)}$$

$$E(p) = \frac{TP^3+P^2}{TP^3+P^2+akTp+k}\frac{1}{p} \text{.....(A.6)}$$

$$E(p) = \frac{p+Tp^2}{k+akTp+p^2+Tp^3} \text{..... (A.7)}$$

Bibliographie

[1] L.Saludjiane. Optimisation en électrotechnique par algorithme génétique. Thèse de Doctorat 3éme cycle LMD, spécialité : génie électrique, université de Grenoble, 1997.

[2] V .Mignon. Cours de maximisation et minimisation sans contraintes. Université Paris Ouest Nantes.2012.

[3] A. Maidi. Cours d'optimisation et commande optimale. Département d'Automatique, Faculté Génie Electrique et d'Informatique, Université Mouloud Mammeri Tizi-Ouzo, 2016/2017

[4] T.Vallée et M.Yidizoglu. Présentation des algorithmes génétique et leurs applications en économie, Université de Nante, LEA-CIL, 2001.

[5] P. Borne et F.Rotella. Commande Optimale, Traité Informatique industrielle, R7427, page 10-12, Technique de l'Ingénieur, 1996.

[6] P.Siarry. Automatique de base, Berti, Alger, 1993

[7] D.Arzelier, J.Benssou, D.Henrion et D, peaculle.Optimisation et théorie de la théorie de la commande robuste. LAAS-CNRS .Toulouse .2012.

[8] B.Tuffin. La simulation de Monte Carlo, Hermés/Lavoisier, Paris, 2010.

[9] Huges Debedout.Supply chain DESIGN and Distribution planning Under Supply Uncertainty.Thése doctorat.Recherhce Operationnelle.Ecole Nationale Supérieure des Mines. Nantes.2013.

[10] A. Rachid. Systèmes de régulation. Masson. Paris, 1996.

[11] A.Dipanda. Cours algorithme génétique M2 BDIA/IIA.

[12] A.Eljai. Elément d'analyse et de contrôle des systèmes. Collection Etudes. Saint-Estève.2004.

Résumé :

Dans ce mémoire, nous allons présenter une approche permettant de concevoir un correcteur PID robuste pour un système incertain en utilisant la méthode des scénarios. Nous avons donné le principe général de la méthode des scénarios. Cette méthode a été exploitée par la suite pour la conception d'un correcteur PID optimale pour un système incertain. Nous avons aussi présenté des généralités sur l'optimisation et la méthode de recherche stochastique qui est les algorithmes génétiques.

Mots-clés : Systèmes dynamiques, systèmes incertains, robustesses, optimisation robuste, algorithmes génétiques, méthode des scénarios, nombre aléatoire, Monte Carlo, PID robuste.