

Este exemplar corresponde à relação final do trabalho
defendida por Daniel Santos Monasterios
Morales e aprovada pela Comissão
Julgadora em 31 / 08 / 95.


Orientador

Universidade Estadual de Campinas
Faculdade de Engenharia Elétrica
Departamento de Computação e Automação
Industrial

Sintonia de Controladores DMC Utilizando Algoritmos Genéticos

Autor: Daniel Santos Monasterios Morales
Orientador: Prof. Dr. Wagner Caradori do Amaral

Dissertação apresentada à Faculdade de Engenharia Elétrica da
Universidade Estadual de Campinas - FEE - UNICAMP, como
parte dos requisitos exigidos para a obtenção do título de MES-
TRE EM ENGENHARIA ELÉTRICA.

Campinas, agosto de 1995.

UNIDADE	BC
N.º ORÇAMENTO	433/95
N.º UNICAMP	M742s
N.º F.	
N.º 1	26.024
PROD.	433/95
C	☐ 1 ☒ 2
PREÇO	R\$ 11,00
DATA	02/11/95
N.º CPD	

CM-00078581-8

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA - BAE - UNICAMP

M742s Monasterios Morales, Daniel Santos
Sintonia de controladores DMC utilizando algoritmos
genéticos / Daniel Santos Monasterios Morales.--
Campinas, SP: [s.n.], 1995.

Orientador: Wagner Caradori do Amaral.
Dissertação (mestrado) - Universidade Estadual de
Campinas, Faculdade de Engenharia Elétrica.

1. Algoritmos. 2. Controle de processo. I. Amaral,
Wagner Caradori do. II. Universidade Estadual de
Campinas. Faculdade de Engenharia Elétrica. III. Título.

A minha querida Bahia.

"Acho que não devemos começar a discussão das questões físicas
sob o prisma das passagens das Escrituras, mas sim das
experiências e demonstrações. Deus não é menos
extraordinariamente revelado nas ações da natureza do que nas
sagradas palavras da Bíblia"

Galileu Galilei (1564-1642)

Agradecimentos

Muitas foram as pessoas, que ao longo destes dois anos de trabalho e convívio, me auxiliaram e encorajaram na elaboração desta dissertação de mestrado. Cabe aqui porém, neste pequeno espaço, uma homenagem sincera àqueles, que de forma muito especial, contribuíram direta e decisivamente para a consecução deste trabalho.

Agradeço inicialmente ao CNPq pela bolsa concedida para a realização do mestrado, sem a qual este presente trabalho não poderia ter sido concebido.

Agradeço também à Universidade Estadual de Campinas, e em especial à Faculdade de Engenharia Elétrica, pela excelente infra-estrutura oferecida, permitindo não somente a realização deste trabalho, mas também o meu crescimento e aperfeiçoamento profissional.

Um agradecimento especial ao Prof. Dr. Paulo Cesar Bezerra que me aceitou nesta universidade e que muito me ajudou para que eu encontrasse o meu caminho dentro do mestrado, e ao Prof. Dr. Hermano Tavares pela oportunidade de poder ter sido seu aluno durante um ano e pelo profissional brilhante que é.

Toda a minha admiração, respeito e amizade ao Prof. Dr. Wagner Caradori do Amaral pelos conselhos e orientação precisa durante a elaboração desta tese, além da liberdade que me deu para o desenvolvimento do tema proposto.

Agradeço de coração aos muitos amigos que fiz aqui em Campinas. Dentre eles, gostaria de mencionar Marcos (potiguar), Marcos (paulista), Herbert, Gustavo, Rubén, Ivan (baiano), Ivan (paulista), Jussara, Cristiana, Gilmar, Geraldo, Paulo, Alberto, Edmilson, Flávia, Welfani, Gonzaga, Ulisses, Farley, Miguel, Osmar e Isamara.

Gostaria de deixar registrada a minha consideração especial e sincera amizade por Silmar, Paulo Henrique, Jês, Ricardo, Rivelino e Leila. Pessoas que aprendi a admirar e a me orgulhar por ter como verdadeiros amigos.

Por último, não poderia deixar de reverenciar os meus pais Jaime e Ivone, e os meus irmãos Dario e Magali. A eles, todo o meu amor e eterna gratidão pela certeza de que estes dois anos de imensa saudade valeram a pena.

Conteúdo

Resumo	1
Abstract	2
1 Introdução ao Controle Preditivo	3
1.1 Introdução	3
1.2 Uma Visão Geral do Controle Preditivo	4
1.3 O Controle Preditivo nos Sistemas de Controle Atuais	5
1.4 Sumário	7
2 O Algoritmo de Controle DMC	8
2.1 Introdução	8
2.2 O Algoritmo DMC sem Restrições	9
2.2.1 Critério de Otimização	9
2.2.2 Modelo do Processo	10
2.2.3 Equação de Previsão	11
2.2.4 Cálculo das Ações de Controle	13
2.2.5 Resumo do Algoritmo DMC sem Restrições . .	14
2.3 O Algoritmo DMC com Restrições	15
2.4 Considerações sobre a Identificação do Modelo do Pro- cesso	19
2.5 Considerações sobre a Sintonia dos Controladores DMC	20
2.6 Sumário	21
3 Algoritmos Genéticos	22
3.1 Introdução	22
3.2 O Algoritmo Genético Básico	24

3.2.1	A Função de Seleção	24
3.2.2	Os Operadores Genéticos	25
3.2.3	Os Parâmetros do Algoritmo Genético	29
3.2.4	Resumo do Algoritmo	30
3.3	Teorema Fundamental dos Algoritmos Genéticos . . .	31
3.3.1	Definições	32
3.3.2	Enunciado do Teorema	32
3.4	Mapeamento de Funções Objetivo	33
3.5	Escala Linear para a Função de Seleção	34
3.6	O Código Multiparamétrico Concatenado de Ponto Fixo	37
3.7	Sumário	39
4	Sintonia DMC Utilizando Algoritmos Genéticos	40
4.1	Introdução	40
4.2	O Algoritmo de Sintonia	41
4.3	Exemplo: Aplicação no Controle de uma Coluna de Destilação de Alta Pureza	45
4.4	Sumário	52
5	Aplicação no Controle do Processo Shell	54
5.1	Introdução	54
5.2	O Processo Shell	54
5.2.1	Modelo do Processo	55
5.2.2	Objetivos de Controle	56
5.2.3	Restrições de Controle	56
5.2.4	Perturbação no Processo	57
5.3	Controle do Processo Shell com Ganhos Fixos	58
5.4	Controle do Processo Shell com Ganhos Variáveis . .	63
5.5	Sumário	67
	Conclusões e Perspectivas	68
	Anexo I: Teorema Fundamental dos Algoritmos Genéticos	70
	Bibliografia	74

Lista de Figuras

1.1	<i>A característica de horizonte móvel do controle preditivo. . . .</i>	5
3.1	<i>A operação de recombinação.</i>	28
3.2	<i>A operação de mutação.</i>	29
3.3	<i>Escala linear para a função de seleção.</i>	35
3.4	<i>O problema da escala linear inconsistente.</i>	36
4.1	<i>Estrutura codificada dos horizontes de previsão e controle. . . .</i>	43
4.2	<i>Esquema da coluna de destilação de alta pureza.</i>	46
4.3	<i>Respostas das variáveis controladas para $p = 22$, $M = 1$, ponderações ECE e MSF unitárias.</i>	48
4.4	<i>Sinais de controle para $p = 22$, $M = 1$, ponderações ECE e MSF unitárias.</i>	49
4.5	<i>Respostas das variáveis controladas para $p = 22$, $M = 1$, ponderações ECE unitárias e ponderações MSF iguais a 2000. . .</i>	49
4.6	<i>Sinais de controle para $p = 22$, $M = 1$, ponderações ECE unitárias e ponderações MSF iguais a 2000.</i>	50
4.7	<i>Respostas das variáveis controladas para $p = 60$, $M = 11$, ponderações ECE unitárias e ponderações MSF iguais a 2000. .</i>	51
4.8	<i>Sinais de controle para $p = 60$, $M = 11$, ponderações ECE unitárias e ponderações MSF iguais a 2000.</i>	52
5.1	<i>Esquema da coluna fracionadora de óleo pesado.</i>	55
5.2	<i>Respostas das variáveis controladas para $p = 23$, $M = 1$, ponderações ECE unitárias e ponderações MSF iguais a 90. . . .</i>	59
5.3	<i>Sinais de controle para $p = 23$, $M = 1$, ponderações ECE unitárias e ponderações MSF iguais a 90.</i>	60

5.4	<i>Respostas das variáveis controladas para $p = 23$, $M = 2$, ponderações ECE unitárias e ponderações MSF iguais a 90. . . .</i>	61
5.5	<i>Sinais de controle para $p = 23$, $M = 2$, ponderações ECE unitárias e ponderações MSF iguais a 90.</i>	61
5.6	<i>Respostas das variáveis controladas para $p = 27$, $M = 2$, ponderações ECE unitárias e ponderações MSF iguais a 90. . . .</i>	62
5.7	<i>Sinais de controle para $p = 27$, $M = 2$, ponderações ECE unitárias e ponderações MSF iguais a 90.</i>	62
5.8	<i>Variação de $\Delta \varepsilon_3$ durante as simulações.</i>	63
5.9	<i>Respostas das variáveis controladas para $p = 20$, $M = 1$, ponderações ECE unitárias e ponderações MSF iguais a 90. . . .</i>	65
5.10	<i>Sinais de controle para $p = 20$, $M = 1$, ponderações ECE unitárias e ponderações MSF iguais a 90.</i>	66
5.11	<i>Respostas das variáveis controladas para $p = 23$, $M = 2$, ponderações ECE unitárias e ponderações MSF iguais a 90. . . .</i>	66
5.12	<i>Sinais de controle para $p = 23$, $M = 2$, ponderações ECE unitárias e ponderações MSF iguais a 90.</i>	67
0.1	<i>O efeito do operador recombinação em classes de cadeias. . . .</i>	72

Lista de Tabelas

3.1	<i>Passos iniciais da implementação do operador reprodução.</i>	27
3.2	<i>O Esquema de Amostragem Estocástica sem Reposição.</i>	27
4.1	<i>Duas simulações iniciais do algoritmo genético para a coluna.</i>	47
4.2	<i>Duas simulações adicionais do algoritmo genético para a coluna.</i>	51
5.1	<i>Coeficientes do modelo discreto do Processo Shell.</i>	56
5.2	<i>Incertezas nos ganhos em regime do Processo Shell.</i>	57
5.3	<i>Duas simulações do algoritmo genético para o Processo Shell com ganhos fixos.</i>	59
5.4	<i>Duas simulações do algoritmo genético para o Processo Shell com ganhos variáveis.</i>	65

Resumo

Este trabalho descreve a aplicação de um algoritmo genético para a sintonia de controladores DMC (*Dynamic Matrix Control*).

Sintonizar um controlador DMC consiste no ajuste de certos parâmetros (horizontes de previsão e controle, ponderações nas ações de controle e nos erros das saídas em relação às respectivas referências). Este ajuste é realizado normalmente por um especialista, a partir de sua experiência anterior na utilização do algoritmo.

Em vista disto, é proposta neste trabalho, uma metodologia que possa dar ao usuário bons valores para os parâmetros do controlador DMC. Esta metodologia de sintonia utiliza um algoritmo genético que seleciona um conjunto restrito de valores para os horizontes de previsão e controle do algoritmo DMC. O ajuste das ponderações é feito de forma iterativa, a partir da observação da resposta do sistema em malha fechada. Este algoritmo é completamente independente da natureza do processo a ser controlado pelo algoritmo DMC, sendo igualmente válido para a simulação de processos com modelos lineares e não lineares.

A metodologia proposta foi utilizada na sintonia de dois processos: uma coluna de destilação de alta pureza e uma coluna fracionadora de óleo pesado. Os resultados obtidos permitiram avaliar o desempenho do algoritmo genético utilizado na sintonia dos controladores DMC.

Abstract

This work describes the application of a genetic algorithm to the tuning of DMC (*Dynamic Matrix Control*) controllers.

Tuning a DMC controller consists of adjusting some parameters (prediction and control horizons, weights for the control actions and for the errors between the outputs and corresponding reference signals). This adjustment is generally achieved by an expert, based on his past experience using the algorithm.

In this work is proposed a methodology to give the user good values for the DMC controller parameters. This methodology uses a genetic algorithm to select a small set of values to the prediction and control horizons of the DMC algorithm. The weights' adjustment is interactive, based on the observation of the response for the closed loop system. This algorithm is completely independent from the nature of the process to be controlled by the DMC algorithm, being equally valid for the simulation of processes with linear and nonlinear models.

The proposed methodology was applied to the tuning of two processes: a high purity distillation column and a heavy oil fractionator column. The obtained results allowed an performance analysis of the genetic algorithm used to tune DMC controllers.

Capítulo 1

Introdução ao Controle Preditivo

1.1 Introdução

Este capítulo apresenta uma discussão geral sobre a técnica de *controle preditivo*, enfatizando a sua posição dentro dos sistemas de controle atuais, onde a integração de diversos níveis hierárquicos de automação se faz necessária para o alcance de objetivos econômicos e a satisfação de critérios de desempenho pré-estabelecidos [García 88]. Uma vez que o algoritmo DMC (*Dynamic Matrix Control*) é uma técnica de controle preditivo, o estudo deste capítulo visa dar uma visão global deste algoritmo antes do estudo de sua formulação matemática.

Os algoritmos de *controle preditivo* pertencem a uma classe ou família de algoritmos conhecida como MPC (*Model Predictive Control*) [García 89]. Estas estratégias de controle utilizam, de forma explícita e direta, um modelo identificável do processo para se fazer previsões das variáveis de saída. Estas previsões são incorporadas a uma função custo pré-estabelecida, que deve ser otimizada.

Em 1978, Richalet desenvolveu o algoritmo MPHC (*Model Predictive Heuristic Control*), cuja implementação computacional resultou no *software* IDCOM (*Identification Command*) [Richalet 78], que é referenciado como o próprio algoritmo MPHC [Martin 81]. Este algoritmo utiliza um modelo baseado na resposta ao impulso do processo para prever as saídas futuras e calcular a ação de controle. Um ano mais tarde, em 1979, Cutler e Ramaker

apresentaram o algoritmo DMC (*Dynamic Matrix Control*), muito empregado em indústrias petroquímicas [Cutler 79] [Cutler 82]. Este algoritmo obtém um modelo a partir da resposta ao degrau do processo e calcula a ação de controle utilizando um modelo inverso da planta. Tanto o IDCOM como o DMC calculam a lei de controle a partir de previsões da saída em um horizonte longo. Em 1987, Clarke apresentou o algoritmo GPC (*Generalized Predictive Control*) dentro da filosofia dos controladores com horizontes longos de previsão [Clarke 87] [Mohtadi 87] [Clarke 89]. Este algoritmo utiliza um modelo CARIMA (*Controlled Auto-Regressive Integrated Moving-Average*) para prever as saídas do processo e calcula a lei de controle minimizando uma função de custo semelhante à utilizada pelo algoritmo DMC.

A segunda seção deste capítulo refere-se ao *controle preditivo*. A terceira seção aborda a posição dos algoritmos de controle preditivo, em especial o DMC, dentro dos sistemas de controle atuais. O capítulo é concluído com um resumo dos principais pontos abordados no decorrer do texto.

1.2 Uma Visão Geral do Controle Preditivo

As estratégias de *controle preditivo* presumem, inicialmente, a existência de um modelo que represente a dinâmica do processo a ser controlado. Geralmente, são utilizados modelos de convolução baseados nas respostas ao impulso ou degrau [Clarke 87] [García 88] [García 89]. A *figura 1.1* apresenta a filosofia básica do *controle preditivo*. Nesta figura, observa-se o comportamento de um determinado processo, ao longo de um *horizonte de previsão* definido por p . O instante presente é definido por k . Supondo a existência de um modelo para o processo considerado, pode-se realizar previsões das variáveis de saída $y(k)$, ao longo do horizonte especificado, e calcular, a partir destas previsões, as melhores ações de controle $m(k)$ que satisfazem a determinados critérios de otimização. Geralmente, estas ações são calculadas para somente uma parte do *horizonte de previsão*, definindo-se assim um *horizonte de controle* representado por M ($M \leq p$). Das ações calculadas, somente a primeira, relativa ao instante presente, é implementada no processo. Em seguida, o instante k é incrementado de um intervalo de tempo e o procedimento é repetido. Este incremento equivale a mover o *horizonte de previsão* de um intervalo de tempo. Esta característica, inerente ao *controle preditivo*, é denominada de *horizonte móvel* (*moving horizon*) [Clarke 87] [García 88].

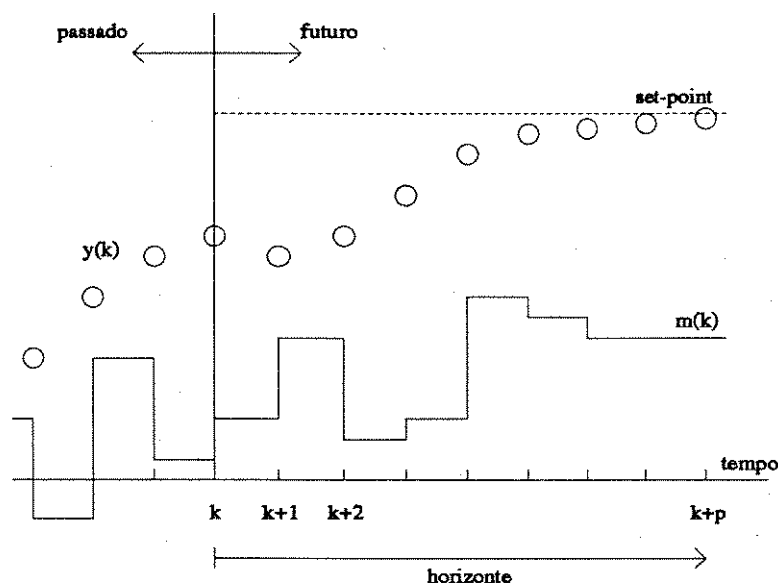


Figura 1.1: A característica de horizonte móvel do controle preditivo.

Os esquemas de *controle preditivo* permitem o projeto de sistemas de controle de alto desempenho, capazes de operar por um longo período de tempo sem a interferência de especialistas [García 89]. Exemplos de algoritmos de *controle preditivo* são: DMC (*Dynamic Matrix Control*), GPC (*Generalized Predictive Control*), MAC (*Model Algorithmic Control*) [Mehra 81] e ID-COM (*Identification Command*). Estes algoritmos apresentam uma estrutura interna de controle denominada IMC (*Internal Model Control*) [Levien 87] [Morari 89].

1.3 O Controle Preditivo nos Sistemas de Controle Atuais

O interesse pelas técnicas de *controle preditivo* se tornou maior para a indústria, quando se percebeu que os sistemas de controle necessitavam garantir mais do que simplesmente a manutenção de pontos de operação. Tornou-se claro que, devido a condições de mercado muito dinâmicas e imprevisíveis, o máximo desempenho de um conjunto de plantas somente poderia se tornar efetivo através de uma integração de diversos níveis de automação, intima-

mente vinculados ao processo de tomada de decisões. Desta forma, seria possível responder rapidamente às variações do mercado com o mínimo de investimento de capital [García 88] [García 89]. Os *níveis hierárquicos de automação* que devem ser integrados dentro do processo de tomada de decisões, são os seguintes [García 89] [Richalet 78]:

Medições e Controle Auxiliar: representa a obtenção e o monitoramento das medições efetuadas em um processo, via instrumentação. Inclui o acesso a valores de variáveis de processo importantes como temperatura, pressão, fluxo, nível, composição, etc. O controle de sistemas auxiliares (válvulas servo, por exemplo) também está incluído neste *nível zero* de automação. Para este tipo de controle em particular, os controladores PID são bastante eficientes.

Controle: relaciona-se com a manipulação dos graus de liberdade de um processo, com o intuito de satisfazer determinados critérios de operação. Neste *nível um* está o controle dinâmico de processos monovariáveis e multivariáveis sujeitos a perturbações. Os controladores PID também são encontrados neste nível de automação.

Otimização: refere-se à manipulação dos graus de liberdade de um processo, visando alcançar determinados objetivos econômicos da planta. Os *set-points* das variáveis de entrada são otimizados através da minimização de funções custo, assegurando a qualidade e a quantidade da produção. Fala-se em um *nível dois* de automação.

Logística: determina a alocação de matérias-primas e a definição do sequenciamento das operações envolvidas na planta, objetivando a maximização de benefícios e o cumprimento do programa estabelecido pela companhia. Aspectos de planejamento através de ferramentas de pesquisa operacional caracterizam este *nível três* de automação.

Os algoritmos MPC, e em particular o DMC, fazem parte do *nível um* de automação. A integração destes algoritmos com os demais níveis de automação se dá pelos seguintes motivos [García 89] [Richalet 78]:

1. O bom desempenho dos algoritmos MPC depende de uma instrumentação e de um controle auxiliar eficientes (*nível zero*).

2. A otimização dos *set-points* das variáveis de entrada (*nível dois*) somente é eficiente se tanto a instrumentação quanto os algoritmos MPC forem capazes de diminuir a variância das saídas relevantes do processo, fazendo-as convergir para os seus respectivos valores de referência.
3. O cumprimento de um planejamento da produção com metas pré-estabelecidas (*nível três*) depende da eficiência dos algoritmos MPC e dos demais níveis de automação.

1.4 Sumário

As técnicas de *controle preditivo* MPC são caracterizadas pela utilização direta de um modelo identificável do processo. Este modelo é empregado para se fazer previsões do comportamento futuro do processo em questão, ao longo de um *horizonte de previsão* definido. Ações de controle correspondentes a estas previsões são calculadas ao longo de um *horizonte de controle*, sendo que somente a primeira ação é implementada no processo. Exemplos de algoritmos pertencentes à classe MPC são: DMC, GPC, MAC e IDCOM. Estes algoritmos, e em particular o DMC, se enquadram dentro de uma estrutura de controle dividida em níveis hierárquicos de automação que buscam levar em consideração não somente aspectos técnicos, mas também econômicos os quais, em conjunto, contribuem para a determinação do ponto ótimo de operação de uma planta.

Capítulo 2

O Algoritmo de Controle DMC

2.1 Introdução

Neste capítulo apresenta-se o algoritmo de controle DMC (*Dynamic Matrix Control*) e discute-se aspectos relacionados com a identificação do modelo do processo a ser controlado e a sintonia destes controladores.

Em 1979, Cutler e Ramaker apresentaram o algoritmo DMC [Cutler 79] [Cutler 82] como uma metodologia capaz de manipular restrições operacionais nas variáveis de controle e saída de forma sistemática, durante o projeto e implementação do controlador. Este algoritmo de controle multivariável é adequado a processos com elevado grau de interação entre suas variáveis, ordens elevadas e atrasos dominantes [Hawkins 88] [Cernik 91] [Houk 91] [Finlayson 88]. Estas características, aliadas à capacidade de manipular restrições, garantem a utilização bem sucedida deste algoritmo na indústria química e petroquímica [Hawkins 88] [Cernik 91] [Houk 91] [O'Connor 91] [Cutler 88] [Finlayson 88].

O algoritmo DMC necessita de um modelo nominal obtido a partir da resposta ao degrau do processo considerado. Uma vez conhecido o modelo, o problema a ser resolvido pelo algoritmo DMC se resume à previsão da saída do processo, utilizando o modelo, e à otimização de uma função objetivo quadrática, sujeita a um conjunto de restrições, que considera os erros entre as previsões da saída e os sinais de referência mais os incrementos nas ações de controle [García 88] [Prett 88].

A segunda seção deste capítulo apresenta o algoritmo DMC sem restrições. Na terceira seção discute-se a inclusão de restrições nas variáveis

de saída e controle, e resolve-se o problema completo do algoritmo DMC. A quarta e quinta seções apresentam considerações sobre a identificação do modelo do processo e a sintonia de controladores DMC. O capítulo é concluído com um resumo dos principais pontos abordados.

2.2 O Algoritmo DMC sem Restrições

Neste algoritmo as variáveis de controle e saída não estão sujeitas a restrições operacionais, estando teoricamente livres para assumir quaisquer valores.

2.2.1 Critério de Otimização

O objetivo do algoritmo DMC multivariável é calcular as ações de controle, representadas por incrementos nas variáveis manipuladas, através da resolução do seguinte problema de otimização:

$$\min_{\Delta m(k), \dots, \Delta m(k+M-1)} \sum_{l=1}^p \|\hat{y}(k+l) - y_s(k+l)\|_{\Xi_l}^2 + \sum_{l=1}^M \|\Delta m(k+l-1)\|_{\Lambda_l}^2 \quad (2.1)$$

onde:

- $\hat{y}(k+l) \in \mathbb{R}^r$ é o vetor coluna das previsões das variáveis de saída.
- $y_s(k+l) \in \mathbb{R}^r$ é o vetor coluna das trajetórias dos sinais de referência.
- $\Delta m(k+l-1) \in \mathbb{R}^v$ é o vetor coluna dos incrementos nas ações de controle.
- p define o horizonte de previsão.
- M define o horizonte de controle ($\Delta m(k+j) = 0 \ \forall j > M; \ M \leq p$).
- $\|x\|_Q^2$ representa o termo quadrático $x^T Q x$.
- Ξ_l, Λ_l são matrizes diagonais de ponderações para os termos quadráticos.

A equação (2.1) pode ser escrita na forma vetorial como

$$\min_{\Delta \mathcal{M}(k)} \|\hat{\mathcal{Y}}(k+1) - \mathcal{Y}_s(k+1)\|_{\Xi}^2 + \|\Delta \mathcal{M}(k)\|_{\Lambda}^2 \quad (2.2)$$

onde:

$$\Xi = \text{diag} [\Xi_1 \dots \Xi_p] \quad (2.3)$$

$$\Lambda = \text{diag} [\Lambda_1 \dots \Lambda_M] \quad (2.4)$$

$$\hat{\mathcal{Y}}(k+1) = \begin{bmatrix} \hat{y}(k+1) \\ \vdots \\ \hat{y}(k+p) \end{bmatrix} \quad \mathcal{Y}_s(k+1) = \begin{bmatrix} y_s(k+1) \\ \vdots \\ y_s(k+p) \end{bmatrix} \quad (2.5)$$

$$\Delta \mathcal{M}(k) = \begin{bmatrix} \Delta m(k) \\ \vdots \\ \Delta m(k+M-1) \end{bmatrix} \quad (2.6)$$

Para que este problema de otimização possa ser resolvido, é necessário que se tenha um modelo nominal do processo, para se fazer as previsões das variáveis de saída.

2.2.2 Modelo do Processo

O modelo de convolução utilizado pelo algoritmo DMC é dado pela equação (2.7):

$$y(k+1) = \sum_{i=1}^{\infty} S_i \Delta u(k+1-i) + w(k) \quad (2.7)$$

onde:

- $y(k) \in \mathbb{R}^r$ é o vetor coluna de saída no instante k .
- $\Delta u(k) \in \mathbb{R}^s$ é o vetor coluna dos incrementos na entrada no instante k , incluindo os incrementos nas variáveis manipuladas e de perturbação.

- $w(k) \in \mathbb{R}^r$ é um vetor coluna que representa o efeito de perturbações aleatórias.
- $S_i \in \mathbb{R}^{r \times s}$ é a matriz resposta ao degrau no i -ésimo instante de tempo contado a partir da aplicação do degrau no processo.

Os coeficientes da matriz S_i são representados por $s_{pq}^{(i)}$, significando a resposta ao degrau no i -ésimo instante, relativa à p -ésima variável de saída e q -ésima variável de entrada.

Como o vetor de entrada $\Delta u(k)$ contém os incrementos nas variáveis manipuladas $\Delta m(k)$ e de perturbação $\Delta d(k)$, a equação (2.7) pode ser escrita na forma:

$$y(k+1) = \sum_{i=1}^{\infty} S_{mi} \Delta m(k+1-i) + \sum_{i=1}^{\infty} S_{di} \Delta d(k+1-i) + w(k) \quad (2.8)$$

A equação (2.8) é utilizada a seguir para o cálculo da previsão da saída.

2.2.3 Equação de Previsão

Supondo que as variáveis de perturbação não mudam de valor ao longo do horizonte de previsão ($\Delta d(k+l) = 0$; $l = 1, 2, \dots$), e podem ser mensuradas no instante k , pode-se escrever a equação (2.8), para um instante futuro $k+l$, da seguinte maneira:

$$y(k+l) = \sum_{i=1}^{\infty} S_{mi} \Delta m(k+l-i) + \sum_{i=1}^{\infty} S_{di} \Delta d(k+l-i) + w(k+l-1)$$

Separando os termos referentes às contribuições do passado e do futuro, obtém-se:

$$y(k+l) = \sum_{i=l+1}^{\infty} S_i \Delta u(k+l-i) + S_{dl} \Delta d(k) + S_{m1} \Delta m(k+l-1) + \dots + \\ + S_{ml-2} \Delta m(k+2) + S_{ml-1} \Delta m(k+1) + S_{ml} \Delta m(k) + w(k+l-1)$$

ou

$$y(k+l) = \hat{y}^*(k+l) + S_{m1} \Delta m(k+l-1) + \dots + \\ + S_{ml-2} \Delta m(k+2) + S_{ml-1} \Delta m(k+1) + S_{ml} \Delta m(k) + w(k+l-1) \quad (2.9)$$

onde

$$\hat{y}^*(k+l) = \sum_{j=l+1}^{\infty} S_j \Delta u(k+l-j) + S_{dl} \Delta d(k) \quad (2.10)$$

A equação (2.10) representa a contribuição para as variáveis de saída, dos incrementos passados nas variáveis de entrada até o instante $k-1$, acrescida da contribuição mais recente dos incrementos nas variáveis de perturbação. As demais parcelas da equação (2.9) representam as contribuições do futuro.

Para a estimativa das perturbações aleatórias $w(k+l-1)$ em (2.9), considera-se que *estas perturbações se mantêm constantes ao longo do horizonte de previsão*. Desta forma, sua estimativa é obtida pela seguinte equação:

$$\hat{w}(k+l-1) = \hat{w}(k) = y(k) - \hat{y}^*(k) \quad (2.11)$$

onde $y(k)$ é o vetor medido da saída e $\hat{y}^*(k)$ é dado pela equação

$$\hat{y}^*(k) = \sum_{i=1}^{\infty} S_i \Delta u(k-i) \quad (2.12)$$

A equação de previsão do algoritmo DMC é obtida pela substituição de $w(k+l-1)$ por sua estimativa em (2.9), obtendo-se:

$$\hat{y}(k+l) = \hat{y}^*(k+l) + S_{m1} \Delta m(k+l-1) + \dots + \\ + S_{ml-2} \Delta m(k+2) + S_{ml-1} \Delta m(k+1) + S_{ml} \Delta m(k) + \hat{w}(k) \quad (2.13)$$

Fazendo-se $l = 1, \dots, p$ em (2.13) e lembrando que $\Delta m(k+j) = 0 \quad \forall j > M$, pode-se escrever a equação de previsão, na forma vetorial, como

$$\hat{\mathcal{Y}}(k+1) = \mathcal{S} \Delta \mathcal{M}(k) + \hat{\mathcal{Y}}^*(k+1) + \hat{\mathcal{W}}(k) \quad (2.14)$$

onde

$$\hat{\mathcal{Y}}^*(k+1) = \begin{bmatrix} \hat{y}^*(k+1) \\ \vdots \\ \hat{y}^*(k+p) \end{bmatrix} \quad \hat{\mathcal{W}}(k) = \begin{bmatrix} \hat{w}(k) \\ \vdots \\ \hat{w}(k) \end{bmatrix} \quad (2.15)$$

e a *matriz dinâmica* do processo é definida como

$$\mathcal{S} = \begin{bmatrix} S_{m1} & 0 & \dots & 0 \\ S_{m2} & S_{m1} & \dots & 0 \\ \vdots & \vdots & \dots & S_{m1} \\ \vdots & \vdots & \ddots & \vdots \\ S_{mp} & S_{mp-1} & \dots & S_{mp-M+1} \end{bmatrix} \quad (2.16)$$

2.2.4 Cálculo das Ações de Controle

A lei de controle é obtida, através da resolução do problema de otimização descrito pela equação (2.2). Desenvolvendo-se esta equação, obtem-se:

$$\begin{aligned} \min_{\Delta \mathcal{M}(k)} [\hat{\mathcal{Y}}(k+1) - \mathcal{Y}_s(k+1)]^T \Xi [\hat{\mathcal{Y}}(k+1) - \mathcal{Y}_s(k+1)] \\ + \Delta \mathcal{M}(k)^T \Lambda \Delta \mathcal{M}(k) \end{aligned} \quad (2.17)$$

Definindo-se o erro que atua no controlador DMC como

$$\hat{\mathcal{E}}(k+1) = \mathcal{Y}_s(k+1) - \hat{\mathcal{W}}(k) - \hat{\mathcal{Y}}^*(k+1) \quad (2.18)$$

e substituindo-se (2.14) e (2.18) em (2.17):

$$\begin{aligned} \min_{\Delta \mathcal{M}(k)} [\mathcal{S} \Delta \mathcal{M}(k) - \hat{\mathcal{E}}(k+1)]^T \Xi [\mathcal{S} \Delta \mathcal{M}(k) - \hat{\mathcal{E}}(k+1)] \\ + \Delta \mathcal{M}(k)^T \Lambda \Delta \mathcal{M}(k) \end{aligned}$$

Desenvolvendo a última equação, tem-se que:

$$\min_{\Delta \mathcal{M}(k)} \Delta \mathcal{M}(k)^T (\mathcal{S}^T \Xi \mathcal{S} + \Lambda) \Delta \mathcal{M}(k) + \hat{\mathcal{E}}(k+1)^T \Xi \hat{\mathcal{E}}(k+1) \\ - \Delta \mathcal{M}(k)^T \mathcal{S}^T \Xi \hat{\mathcal{E}}(k+1) - \hat{\mathcal{E}}(k+1)^T \Xi \mathcal{S} \Delta \mathcal{M}(k)$$

ou, equivalentemente:

$$\min_{\Delta \mathcal{M}(k)} \Delta \mathcal{M}(k)^T (\mathcal{S}^T \Xi \mathcal{S} + \Lambda) \Delta \mathcal{M}(k) - 2 \Delta \mathcal{M}(k)^T \mathcal{S}^T \Xi \hat{\mathcal{E}}(k+1) \quad (2.19)$$

Resolvendo o problema dado pela equação (2.19), através do *Método dos Mínimos Quadrados*, tem-se que:

$$\Delta \mathcal{M}(k) = (\mathcal{S}^T \Xi \mathcal{S} + \Lambda)^{-1} \mathcal{S}^T \Xi \hat{\mathcal{E}}(k+1) \quad (2.20)$$

A equação (2.20) fornece os incrementos nas ações de controle dentro do horizonte de controle. Como somente a primeira ação é aplicada, tem-se que:

$$\Delta m(k) = \begin{bmatrix} I & 0 & \dots & 0 \end{bmatrix} \Delta \mathcal{M}(k) \quad (2.21)$$

A equação (2.21) fornece o incremento na ação de controle a ser aplicado no processo.

2.2.5 Resumo do Algoritmo DMC sem Restrições

O algoritmo de controle DMC multivariável sem restrições pode ser resumido como segue:

1. Obter o modelo do processo descrito pela equação (2.7).
2. Definir os horizontes de previsão e controle.
3. Definir as matrizes diagonais dadas pelas equações (2.3) e (2.4).
4. Construir a *matriz dinâmica* dada pela equação (2.16).
5. Definir o vetor com as trajetórias dos sinais de referência, descrito em (2.5).

6. Para cada instante k , faça:

- (a) Calcular o vetor erro, descrito em (2.18), utilizando as equações (2.10), (2.11) e (2.15).
- (b) Calcular e aplicar o incremento na ação de controle definida pelas equações (2.20) e (2.21).

Note-se que a cada iteração deste algoritmo, somente o vetor erro é atualizado.

2.3 O Algoritmo DMC com Restrições

O algoritmo DMC sem restrições define a estrutura básica de controle DMC. Quando existem restrições nas variáveis de controle e saída do processo considerado, obtém-se uma nova estratégia de controle: o algoritmo DMC com restrições, denominado na literatura especializada, como algoritmo QDMC (*Quadratic Dynamic Matrix Control*).

No algoritmo QDMC, são consideradas três tipos de restrições [Prett 88]:

Restrições nas Variáveis Manipuladas: representam limites rígidos nas variáveis de entrada $m(k)$, com o intuito de evitar problemas de saturação (em válvulas, por exemplo).

Restrições nos Incrementos das Variáveis Manipuladas: representam limites rígidos impostos aos incrementos $\Delta m(k)$, influenciando diretamente nas taxas de variação de $m(k)$.

Restrições nas Variáveis de Saída: representam limites rígidos ou não, nas variáveis de saída, com o intuito de evitar valores demasiadamente grandes. Estas restrições são de dois tipos:

Restrições nas Variáveis Controladas: definem limites para estas variáveis, apesar dos desvios de seus valores de referência serem minimizados na função objetivo descrita pela equação (2.1).

Restrições nas Variáveis Associadas: são variáveis não controladas que apesar de não possuírem valores de referência, devem ser mantidas dentro de limites pré-definidos ($\Xi_l = 0$ para estas variáveis, na equação (2.1)).

Estas restrições podem ser descritas matematicamente, como segue:

$$\mathcal{M}_{low}(k) \leq \mathcal{I}_L \Delta \mathcal{M}(k) + \mathcal{M}(k-1) \leq \mathcal{M}_{high}(k) \quad (2.22)$$

$$\Delta \mathcal{M}_{low}(k) \leq \Delta \mathcal{M}(k) \leq \Delta \mathcal{M}_{high}(k) \quad (2.23)$$

$$\mathcal{Y}_{low}(k+1) \leq \hat{\mathcal{Y}}(k+1) \leq \mathcal{Y}_{high}(k+1) \quad (2.24)$$

onde:

$$\mathcal{I}_L = \begin{bmatrix} I & 0 & \dots & 0 \\ I & I & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ I & I & \dots & I \end{bmatrix} \quad \mathcal{M}(k-1) = \begin{bmatrix} m(k-1) \\ \vdots \\ m(k-1) \end{bmatrix} \quad (2.25)$$

$$\mathcal{M}_{low}(k) = \begin{bmatrix} m_{low}(k) \\ \vdots \\ m_{low}(k+M-1) \end{bmatrix} \quad \mathcal{M}_{high}(k) = \begin{bmatrix} m_{high}(k) \\ \vdots \\ m_{high}(k+M-1) \end{bmatrix} \quad (2.26)$$

$$\Delta \mathcal{M}_{low}(k) = \begin{bmatrix} \Delta m_{low}(k) \\ \vdots \\ \Delta m_{low}(k+M-1) \end{bmatrix}$$

$$\Delta \mathcal{M}_{high}(k) = \begin{bmatrix} \Delta m_{high}(k) \\ \vdots \\ \Delta m_{high}(k+M-1) \end{bmatrix} \quad (2.27)$$

$$\mathcal{Y}_{low}(k+1) = \begin{bmatrix} y_{low}(k+1) \\ \vdots \\ y_{low}(k+p) \end{bmatrix} \quad \mathcal{Y}_{high}(k+1) = \begin{bmatrix} y_{high}(k+1) \\ \vdots \\ y_{high}(k+p) \end{bmatrix} \quad (2.28)$$

A equação (2.26) representa os limites inferiores e superiores impostos às variáveis manipuladas. As equações (2.27) e (2.28) representam limites

análogos para os incrementos nas variáveis manipuladas e para as variáveis de saída, respectivamente.

O problema de otimização a ser resolvido pelo algoritmo QDMC é então formulado como

$$\min_{\Delta \mathcal{M}(k)} \|\hat{\mathcal{Y}}(k+1) - \mathcal{Y}_s(k+1)\|_{\Xi}^2 + \|\Delta \mathcal{M}(k)\|_{\Lambda}^2 \quad (2.29)$$

s.a

$$\mathcal{M}_{low}(k) \leq \mathcal{I}_L \Delta \mathcal{M}(k) + \mathcal{M}(k-1) \leq \mathcal{M}_{high}(k)$$

$$\Delta \mathcal{M}_{low}(k) \leq \Delta \mathcal{M}(k) \leq \Delta \mathcal{M}_{high}(k)$$

$$\mathcal{Y}_{low}(k+1) \leq \hat{\mathcal{Y}}(k+1) \leq \mathcal{Y}_{high}(k+1)$$

Utilizando-se a equação (2.19), pode-se reescrever (2.29) no *formato padrão de um programa quadrático* [Prett 88]:

$$\min_{\Delta \mathcal{M}(k)} [\Delta \mathcal{M}(k)]^T \mathcal{H} [\Delta \mathcal{M}(k)] - h(k+1)^T \Delta \mathcal{M}(k)$$

s.a

$$\mathcal{G} \Delta \mathcal{M}(k) \geq \hat{g}(k+1) \quad (2.30)$$

onde a matriz *hessiana* é dada por

$$\mathcal{H} = \mathcal{S}^T \Xi \mathcal{S} + \Lambda \quad (2.31)$$

o *vetor gradiente* é definido por

$$h(k+1) = 2\mathcal{S}^T \Xi \hat{\mathcal{E}}(k+1) \quad (2.32)$$

e a matriz $\mathcal{G}$ e o vetor $\hat{g}(k+1)$ são dados por

$$\mathcal{G} = \begin{bmatrix} -\mathcal{I}_L \\ \mathcal{I}_L \\ -I \\ I \\ -\mathcal{S} \\ \mathcal{S} \end{bmatrix} \quad (2.33)$$

e

$$\hat{g}(k+1) = \begin{bmatrix} \mathcal{M}(k-1) - \mathcal{M}_{high}(k) \\ \mathcal{M}_{low}(k) - \mathcal{M}(k-1) \\ -\Delta\mathcal{M}_{high}(k) \\ \Delta\mathcal{M}_{low}(k) \\ \hat{W}(k) + \hat{Y}^*(k+1) - \mathcal{Y}_{high}(k+1) \\ \mathcal{Y}_{low}(k+1) - \hat{W}(k) - \hat{Y}^*(k+1) \end{bmatrix} \quad (2.34)$$

A solução do programa quadrático descrito pela equação (2.30) fornece os incrementos nas ações de controle dentro do horizonte de controle. Como somente a primeira ação é aplicada, substitui-se o resultado de (2.30) em (2.21), e obtém-se os incrementos para as variáveis manipuladas.

O algoritmo QDMC é resumido nos seguintes passos:

1. Obter o modelo do processo descrito pela equação (2.7).
2. Definir os horizontes de previsão e controle.
3. Definir as matrizes diagonais dadas pelas equações (2.3) e (2.4).
4. Construir a *matriz dinâmica* dada pela equação (2.16).
5. Definir o vetor com as trajetórias dos sinais de referência, descrito em (2.5).
6. Obter as restrições estabelecidas pelo processo, descritas pelas equações (2.22), (2.23) e (2.24).
7. Calcular a Hessiana definida em (2.31).
8. Para cada instante k , faça:
 - (a) Calcular o vetor gradiente e o vetor $\hat{g}(k+1)$ descritos em (2.32) e (2.34), respectivamente.
 - (b) Resolver o programa quadrático descrito em (2.30).
 - (c) Substituir o resultado do programa quadrático na equação (2.21) para obter o incremento na ação de controle.
 - (d) Aplicar o incremento, obtido no passo anterior, no processo.

Note-se que a cada iteração do algoritmo QDMC, somente os vetores $h(k + 1)$ e $\hat{g}(k + 1)$ são atualizados no programa quadrático. Um cuidado muito importante que se deve ter neste algoritmo, é a verificação da consistência das restrições, evitando assim problemas de infactibilidade [Prett 88].

2.4 Considerações sobre a Identificação do Modelo do Processo

A identificação do modelo de convolução da equação (2.7) se resume à determinação das matrizes resposta ao degrau S_i , com o auxílio de um pacote de *software* apropriado (por exemplo, o *Matlab*). O objetivo da identificação das matrizes resposta ao degrau das variáveis de perturbação mensuráveis, é realizar *compensações feedforward*, presentes no cálculo do vetor $\mathcal{Y}^*(k + 1)$.

No modelo de convolução, o somatório de termos é efetuado até um valor N (ordem do modelo), a princípio arbitrário. Na realidade, a partir do momento em que o processo alcança o estado de regime permanente, as matrizes resposta ao degrau de (2.7) se estabelecem em um determinado valor [Finlayson 88]. Assim, a ordem N é definida como sendo *maior ou igual ao tempo de acomodação do processo* [Prett 88]. Este procedimento, diminui o esforço computacional na identificação e posterior utilização do modelo dinâmico.

A ordem N é função direta da ordem das equações diferenciais lineares que descrevem a dinâmica do processo. Quanto maior a ordem das equações diferenciais, maior o valor de N . Desta maneira, o modelo descrito em (2.7), permite considerar qualquer ordem para as equações diferenciais do processo, mesmo sem conhecê-las, fazendo com que não se fique restrito a aproximações de primeira e segunda ordens, típicas de aplicações industriais baseadas no controle PID. Os pacotes de *software* de identificação utilizados nas aplicações do controle DMC, exploram esta característica [Cernik 91] [Cutler 88] [Finlayson 88].

2.5 Considerações sobre a Sintonia dos Controladores DMC

Os parâmetros que determinam a sintonia de um controlador DMC são os seguintes [García 88] [Prett 88] [Georgiou 88]:

As Ponderações ECE e MSF: as ponderações da matriz Ξ são conhecidas, na literatura especializada, pela sigla ECE (*Equal Concern Errors*), enquanto que as ponderações da matriz Λ são designadas pela sigla MSF (*Moving Suppression Factors*) [Cernik 91]. A importância relativa de cada um dos critérios especificados na função objetivo da equação (2.2), pode ser reforçada por um aumento nos valores dos parâmetros ECE e MSF correspondentes [García 88] porém, este reforço se torna difícil devido às interações entre as variáveis do processo [García 88]. Aumentos nas ponderações ECE tornam o controle mais rigoroso, diminuindo o erro entre a saída e sua referência [Cernik 91] [Cutler 88]. Aumentos nas ponderações MSF diminuem os incrementos nas variáveis manipuladas, tornando a resposta das variáveis de saída mais lenta e suave. As ponderações MSF também afetam diretamente a robustez do controlador DMC (MSF's maiores aumentam a robustez) [Georgiou 88].

Os Horizontes de Previsão e Controle: um valor elevado para o horizonte de controle (e consequentemente para o horizonte de previsão, pois $M \leq p$) geralmente melhora o desempenho do controlador DMC, porém aumenta o valor dos incrementos nas variáveis manipuladas, reduzindo a robustez do controlador [Georgiou 88]. A determinação adequada dos horizontes de previsão e controle é fundamental para o bom desempenho do algoritmo DMC. Observou-se que sintonizar um controlador DMC fixando os horizontes e manipulando as ponderações ECE e MSF, geralmente não produz bons resultados. Já o procedimento inverso torna o processo de sintonia mais factível. Isto demonstra que os horizontes de previsão e controle possuem uma importância relativa maior que aquela atribuída às ponderações ECE e MSF.

Além destes parâmetros, observa-se que as *restrições nos incrementos das variáveis manipuladas* influenciam diretamente na velocidade de resposta das

variáveis de saída do processo considerado, possuindo uma *ação superior aos parâmetros MSF*. Valores muito altos para os incrementos nas variáveis manipuladas podem tornar o programa quadrático em (2.30) infactível [Prett 88].

2.6 Sumário

O algoritmo DMC é uma estratégia de controle ótimo adequada a sistemas multivariáveis com interações, ordens elevadas e atrasos dominantes, que podem ser descritos ou aproximados por um conjunto de equações diferenciais lineares. A incorporação de restrições operacionais nas variáveis de saída e controle do processo e de *compensações feedforward* aumentam a eficiência do controle DMC.

A ordem do modelo de convolução deve ser no mínimo igual ao tempo de acomodação do processo. Quanto maior a ordem das equações diferenciais lineares que descrevem o processo, maior a ordem do modelo. Pode-se, portanto, considerar qualquer ordem para as equações diferenciais do processo, bastando aumentar ou diminuir a ordem do modelo de convolução.

Os parâmetros de sintonia de um controlador DMC são: as ponderações ECE e MSF e os horizontes de previsão e controle. O aumento das ponderações ECE e MSF reforça a importância dos critérios individuais da função objetivo do algoritmo DMC, contudo este reforço se torna difícil devido às interações entre as variáveis do processo. As ponderações MSF estão também diretamente relacionadas com a robustez do controlador DMC. Um aumento no valor do horizonte de controle geralmente melhora o desempenho do controlador DMC, porém reduz a sua robustez. Os horizontes de previsão e controle possuem uma importância relativa maior que aquela atribuída às ponderações ECE e MSF.

Além dos parâmetros de sintonia, observa-se que as restrições nos incrementos das variáveis manipuladas influenciam a velocidade de resposta do processo, sendo mais eficientes que os parâmetros MSF.

Capítulo 3

Algoritmos Genéticos

3.1 Introdução

Neste capítulo é apresentado o algoritmo genético básico, ressaltando sua utilidade como um algoritmo de busca aplicável a determinados problemas de otimização, onde não se tem informação sobre a continuidade e existência de derivadas da função objetivo.

Os algoritmos genéticos são esquemas de busca e otimização baseados em probabilidades [Michalewicz 90], que se mostram mais eficientes que as técnicas de busca e otimização locais, baseadas no gradiente da função objetivo do problema, e aquelas baseadas em buscas puramente aleatórias, quando estas técnicas são aplicadas a espaços de soluções complexos. Estes algoritmos são insensíveis à forma da função objetivo, sendo igualmente eficientes em funções descontínuas, ruidosas, multidimensionais e multimodais [De Jong 80] [Goldberg 89] [Grefenstette 86]. Os algoritmos genéticos são eficientes na busca de regiões de interesse dentro do espaço de soluções [Goldberg 89].

A idéia principal dos algoritmos genéticos é, a partir de uma população inicial de soluções para um dado problema de otimização, gerada aleatoriamente ou segundo alguma heurística [Grefenstette 86], gerar populações consecutivas de acordo com os seguintes passos [De Jong 80] [Goldberg 89] [Grefenstette 86] [Park 94]:

1. Avaliar a qualidade de cada uma das soluções da população corrente e, com base nesta informação, definir probabilidades correspondentes

para estas soluções. As melhores soluções terão maior probabilidade de serem selecionadas para a próxima população, enquanto que as soluções ruins terão menor probabilidade de seleção. Esta operação de seleção é realizada pelo operador genético *reprodução*.

2. Enquanto o tamanho fixo das populações não tiver sido alcançado, faça:
 - (a) Selecionar duas soluções da população corrente, de acordo com as probabilidades pré-definidas.
 - (b) Submeter estas soluções aos operadores genéticos *recombinação* (*crossover*) e *mutação* (*mutation*), gerando duas novas soluções que farão parte da nova população e darão ao algoritmo genético opções de novos caminhos dentro do espaço de soluções.

Este processo é então repetido para a mais recente população gerada, até que um determinado critério de parada seja satisfeito. Este critério pode ser [Grefenstette 86]:

1. Determinação de uma solução aproximada aceitável para o problema.
2. Fixação do número de populações a gerar.

Os algoritmos genéticos podem manipular, eficientemente, uma grande quantidade de informação acumulada sobre o problema, através de mecanismos relativamente simples (os operadores genéticos) [Grefenstette 86]. São capazes também de explorar várias soluções ao mesmo tempo (*paralelismo*), ao invés de uma única solução por vez, como nas técnicas tradicionais de busca e otimização. Esta característica permite aos algoritmos genéticos grande capacidade de adaptação a problemas diversos [De Jong 80] [Goldberg 89].

Os conceitos básicos dos algoritmos genéticos foram desenvolvidos por John Holland e colaboradores, sendo que o trabalho básico sobre o tema foi apresentado pelo próprio Holland em 1975, com o título: *Adaptation in Natural and Artificial Systems*. Os estudos de Holland deram início à teoria de busca e otimização genética [Goldberg 89] [Grefenstette 86]. Desde então, os algoritmos genéticos têm sido aplicados com sucesso em diversas áreas, tais como: Pesquisa Operacional [Goldberg 89] [Croce 93], Biologia, Processamento de Imagens, Ciências Sociais, Aprendizado de Máquinas [Goldberg 89], Lógica Nebulosa [Thrift 91], Identificação de Sistemas [Keane 93] [Kristinsson 88]

[Kristinsson 92] [Yang 94] e Controle [Park 94] [Fleming 93] [Michalewicz 90] [Hunt 92] [Goldberg 85] [Fonseca 93] [Hunt1 92] [Karr 92] [Kristinsson 92] [Linkens 92] [Oliveira 91] [Porter 92] [Renders 92] [Karr 91].

A segunda seção deste capítulo apresenta o algoritmo genético básico. A terceira seção introduz o *Teorema Fundamental dos Algoritmos Genéticos* que permite analisar o comportamento destes algoritmos. A quarta e quinta seções abordam soluções para dois problemas comuns que ocorrem na implementação de um algoritmo genético: funções objetivo que não obedecem a certos requisitos exigidos pelo algoritmo genético (quarta seção) e deficiências do processo de seleção (quinta seção). A sexta seção apresenta uma estrutura de codificação muito utilizada para as soluções candidatas de um determinado problema. O capítulo é concluído com um resumo dos principais pontos abordados em cada seção.

3.2 O Algoritmo Genético Básico

Os algoritmos genéticos representam uma classe de procedimentos de busca e otimização, baseados nos mecanismos da genética e seleção natural [Goldberg 89] [Grefenstette 86]. Estes algoritmos manipulam populações formadas por *cadeias de bits* (*strings of bits*), as quais representam codificações de soluções, dentro de um certo domínio de definição do problema de otimização. A manipulação destas soluções codificadas é feita pelos operadores genéticos a partir da informação fornecida pela *função de seleção* (*fitness function*) utilizada pelo algoritmo genético [Goldberg 89] [Kristinsson 92].

3.2.1 A Função de Seleção

A qualidade das soluções de uma determinada população é avaliada por uma *função de seleção*. Quanto maior o valor da *função de seleção* melhor será a solução associada a este valor. A *função de seleção* avalia as soluções dentro do domínio dos números reais. Contudo, como mencionado anteriormente, as soluções em um algoritmo genético estão codificadas em *cadeias de bits*. Desta forma, fica claro que antes da avaliação de uma determinada solução, é necessário que a mesma seja transformada para um número real. Transformações de *cadeias de bits* em números reais e vice-versa representam técnicas de codificação e decodificação de soluções inerentes aos algoritmos genéticos.

A *função de seleção* é a única fonte de informação de que dispõe o algoritmo genético sobre o problema de otimização, não havendo a necessidade de informações adicionais como a continuidade e a existência de derivadas da *função objetivo* do problema considerado. Para que uma função seja considerada *função de seleção*, os seguintes requisitos devem ser satisfeitos:

1. A função deve ser não negativa.
2. Obter a solução ótima para o problema de otimização implica em *maximizar* a função considerada.

Se a *função objetivo* do problema de otimização satisfizer os dois requisitos supracitados, nada impede a sua escolha como *função de seleção* do algoritmo genético. Quando a *função objetivo* não obedece aos requisitos supracitados, realiza-se um *mapeamento* que a transforme em uma *função de seleção*.

3.2.2 Os Operadores Genéticos

Uma vez avaliada a qualidade das soluções da população corrente, o algoritmo genético utiliza-se dos seguintes operadores para gerar a nova população: *reprodução*, *recombinação* e *mutação*. Estes operadores desempenham dois papéis importantes. Primeiro, preservando as soluções de boa qualidade em maior proporção que aquelas consideradas ruins (*reprodução*). Segundo, promovendo a geração de novas soluções (*recombinação e mutação*), a partir das já existentes, dando ao algoritmo genético as seguintes características de grande importância: a criação de novos caminhos dentro do espaço de soluções e a manutenção de um certo grau de diversidade entre as soluções geradas. Esta diversidade é importante, pois aumenta o número de novas soluções que podem ser geradas a partir de combinações das soluções já existentes.

O Operador Reprodução

Este operador tem como objetivo selecionar cópias da população corrente para aplicação dos operadores *recombinação e mutação*. Uma das implementações deste operador, amplamente utilizada em diversas aplicações, é o *Esquema de Amostragem Estocástica dos Restos sem Reposição* (*Remainder Stochastic Sampling without Replacement Scheme*) [Goldberg 89] [Kristinsson 92] que obedece aos seguintes passos:

1. Calcular os valores da *função de seleção* $f(x)$ para cada solução da população corrente.
2. Calcular o peso relativo de cada solução. Isto é, divide-se cada valor de $f(x)$ pelo somatório de todos os valores. Estes pesos relativos são então utilizados como probabilidades de seleção (p) de uma cópia das correspondentes soluções.
3. Calcular o *número esperado de cópias* $E(x)$ (esperança matemática) de cada solução multiplicando os respectivos valores de p pelo tamanho da população (n).
4. Gerar cópias normais de cada solução para a próxima população, em número igual à respectiva *parte inteira* de $E(x)$.
5. Considerar as *partes fracionárias* de $E(x)$ como probabilidades de seleção de mais uma *única* cópia de cada solução.
6. Enquanto o tamanho da próxima população for menor que n , faça:
 - (a) Para uma solução da população corrente, escolher um número aleatório entre zero e um.
 - (b) Se a parte fracionária de $E(x)$, referente a esta solução, for maior ou igual ao número aleatório escolhido, acrescentar uma cópia extra da respectiva solução na próxima população e retirar esta solução da população corrente.
7. Escolher, aleatoriamente, duas cópias de soluções entre as normais e as extras para a aplicação dos operadores *recombinação* e *mutação*.

A seguir apresenta-se um exemplo simples para ilustrar o funcionamento do operador *reprodução*. Este exemplo também será utilizado para ilustrar, mais adiante, o funcionamento dos operadores *recombinação* e *mutação*.

Exemplo 3.1 Deseja-se maximizar a função objetivo $g(x) = x^2$ para valores do parâmetro x dentro do intervalo inteiro $[0; 31]$. A função de seleção é a própria função objetivo. Utiliza-se uma população inicial, gerada aleatoriamente, com $n = 4$ soluções.

No. da cadeia	Cadeia de bits	Parâmetro x	Função de Seleção $f(x) = x^2$	Probabilidade $p = f_i / \sum_{j=1}^n f_j$
1	01101	13	169	0.1444
2	11000	24	576	0.4923
3	01000	8	64	0.0547
4	10011	19	361	0.3085

Tabela 3.1: *Passos iniciais da implementação do operador reprodução.*

Esperança $E(x) = np$	No. de cópias normais	No. de cópias extras	Nova População
0.5776	0	1	01101
1.9692	1	1	11000
0.2188	0	0	11000
1.2340	1	0	10011

Tabela 3.2: *O Esquema de Amostragem Estocástica sem Reposição.*

A *tabela 3.1* mostra os passos iniciais envolvidos na implementação do operador *reprodução*. Na segunda coluna tem-se a população inicial de quatro soluções aleatórias. Na terceira coluna tem-se os valores inteiros do parâmetro x , correspondentes às respectivas soluções codificadas. O processo de codificação e decodificação empregado neste exemplo é a simples conversão da base decimal para a base binária e vice-versa. As duas colunas restantes indicam o cálculo da *função de seleção* e das respectivas probabilidades de seleção.

A *tabela 3.2* mostra a complementação da *tabela 3.1*. Na *tabela 3.2*, a parte inteira dos valores da primeira coluna representa o número de cópias normais. Já a parte fracionária representa a probabilidade de seleção de uma única cópia extra. A última coluna da *tabela 3.2* apresenta a nova população após o operador *reprodução*. Note-se que a soma das cópias normais com as cópias extras perfaz quatro elementos, mantendo portanto o mesmo número de indivíduos para a próxima população.

O Operador Recombinação

O operador *recombinação* gera novas alternativas de soluções, a partir daquelas já existentes. Sua implementação é feita de acordo com os seguintes passos:

1. Selecionar, segundo o operador *reprodução*, duas *cadeias de bits* de comprimento k .
2. Selecionar, aleatoriamente, uma posição nas *cadeias* dentro do intervalo inteiro $[1; k - 1]$. Esta posição define o ponto onde haverá a troca de informação, sendo denominado de *ponto de cruzamento* (*crossing site*).
3. Preservar as partes das *cadeias* à esquerda do *ponto de cruzamento* e trocar as partes à direita, dando origem a duas novas *cadeias de bits*.

Exemplo 3.2 *Considera-se o mesmo exemplo utilizado anteriormente para ilustrar a implementação do operador reprodução, supondo-se que as cadeias de número um e quatro na tabela 3.1 foram escolhidas para sofrer a operação de recombinação. Define-se o ponto de cruzamento como sendo igual à segunda posição das cadeias, contada a partir da esquerda. A figura 3.1 ilustra a operação de recombinação.*

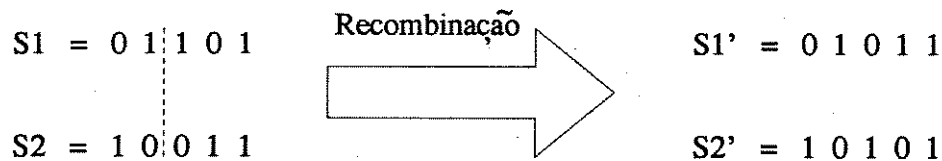


Figura 3.1: A operação de recombinação.

O Operador Mutação

As operações de *reprodução* e *recombinação* garantem a seleção das melhores soluções, através da cópia de *cadeias* e da troca de informação entre elas. Contudo, dentro deste processo, pode-se perder alguma informação importante, devido à ruptura de partes de *cadeias* potencialmente úteis, em operações de *reprodução* e posterior *recombinação*. Com o intuito de recuperar esta informação relevante, define-se o operador *mutação* [Goldberg 89]. A implementação deste operador é resumida nos seguintes passos:

1. Selecionar, segundo o operador *reprodução*, uma *cadeia de bits*.
2. Escolher, aleatoriamente, um *bit* da *cadeia* selecionada e complementar o seu valor, dando origem a uma nova *cadeia de bits*.

Exemplo 3.3 Considera-se que a cadeia de número um, do exemplo considerado para os dois operadores anteriores, foi selecionada para sofrer mutação e que o bit a ser mudado de valor é o segundo, da esquerda para a direita. A figura 3.2 ilustra a operação de mutação.

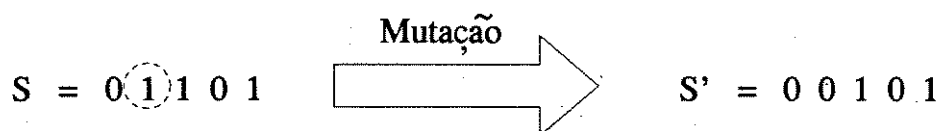


Figura 3.2: A operação de mutação.

3.2.3 Os Parâmetros do Algoritmo Genético

Em um algoritmo genético, há a necessidade de se especificar probabilidades de ocorrência para as operações de *recombinação* e *mutação*. Estas probabilidades, juntamente com o tamanho fixo das populações geradas, são *parâmetros do algoritmo genético*. A influência de cada um deles é analisada como segue [Goldberg 89] [Grefenstette 86]:

Tamanho da População (n): populações muito pequenas não representam amostras significativas do espaço de soluções para o problema de otimização, implicando em taxas de convergência rápidas para soluções pobres. Por outro lado, populações grandes, apesar de representarem amostras significativas do espaço de soluções e tornarem o processo de busca do algoritmo genético mais informado, requerem um tempo de processamento maior, redundando em taxas de convergência maiores para o algoritmo. Desta forma, o tamanho das populações representa um compromisso entre a quantidade de informação a ser manipulada pelo algoritmo genético e o tempo computacional necessário para processar as informações [Potts 94].

Probabilidade de Recombinação (p_r): quanto maior a probabilidade de recombinação, mais rapidamente novas soluções são introduzidas nas populações seguintes. Contudo, se a probabilidade de recombinação for muito alta, soluções de alta qualidade podem ser modificadas pela ação intensa do operador recombinação, impedindo a sua seleção para populações futuras e a consequente melhora destas soluções. Por outro lado, uma probabilidade de recombinação muito baixa, pode levar o processo de busca genética a um estado de estagnação, devido à pouca exploração do espaço de soluções.

Probabilidade de Mutação (p_m): a mutação é um operador secundário que aumenta a diversidade das populações geradas. Devido ao seu caráter intrinsecamente aleatório, a operação de mutação deve ter uma frequência restrita, em analogia com o que ocorre na natureza, onde a probabilidade de mutação é menor que 0.1% [De Jong 80]. Desta forma, uma probabilidade alta de mutação resulta, essencialmente, em uma busca puramente aleatória [Kristinsson 92]. Uma probabilidade baixa permite uma criação restrita e desejável de novas soluções capazes de aumentar a diversidade das populações geradas.

Schaffer [Potts 94] determinou as seguintes faixas de valores ótimos para os parâmetros de um algoritmo genético:

- **Tamanho da População:** entre 20 e 30 soluções.
- **Probabilidade de Recombinação:** entre 75% e 95%.
- **Probabilidade de Mutação:** entre 0.5% e 1%.

3.2.4 Resumo do Algoritmo

O algoritmo genético pode ser resumido pelos seguintes passos [De Jong 80] [Goldberg 89] [Grefenstette 86] [Park 94]:

1. Definir uma *função objetivo* para o problema de otimização.
2. Redefinir, caso necessário, a função objetivo para a forma de uma *função de seleção (fitness function)*.

3. Definir os seguintes parâmetros para o algoritmo genético: o tamanho das populações n , a probabilidade de recombinação p_r e a probabilidade de mutação p_m .
4. Definir um critério C de parada para o algoritmo genético.
5. Inicializar o contador de populações geradas: $t = 0$.
6. Gerar uma população inicial $\mathcal{P}(t)$ contendo soluções codificadas (cadeias de bits) para o problema considerado.
7. Enquanto C não for satisfeito, faça:
 - (a) Decodificar cada uma das soluções de $\mathcal{P}(t)$, calculando o valor da *função de seleção* para cada uma delas.
 - (b) Inicializar o contador do número de soluções da população: $i = 0$.
 - (c) Enquanto $i < n$ faça:
 - i. Selecionar duas soluções de $\mathcal{P}(t)$, segundo o *Esquema de Amostragem Estocástica sem Reposição*.
 - ii. Aplicar simultaneamente os operadores *recombinação* e *mutação* a estas duas soluções, dando origem a duas novas soluções (a cada *bit* copiado ou trocado durante a *recombinação* é aplicado o operador *mutação*).
 - iii. Incluir estas duas novas soluções em $\mathcal{P}(t + 1)$ e fazer $i \leftarrow i + 2$.
 - (d) Faça $t \leftarrow t + 1$.
8. Fim do algoritmo.

3.3 Teorema Fundamental dos Algoritmos Genéticos

A influência dos operadores *reprodução*, *recombinação* e *mutação* nos algoritmos genéticos é expressa matematicamente pelo *Teorema Fundamental dos Algoritmos Genéticos* [Goldberg 89]. A demonstração deste teorema é apresentada no **Anexo I**.

3.3.1 Definições

O enunciado deste teorema exige a introdução prévia de algumas definições. São elas:

Classe de Cadeias (Schema): conjunto de todas as *cadeias de bits* que possuem uma ou mais posições fixas comuns. Por exemplo, a classe $H = *11*0*1$ engloba todas as *cadeias* que possuam os correspondentes dígitos *um* e *zero* nas posições fixas indicadas em H e qualquer dígito nas posições preenchidas pelos asteriscos. O número de *cadeias* pertencentes à t -ésima população e que fazem parte da classe H , é denotado por $m(H, t)$.

Ordem de Classe: refere-se ao *número de posições fixas em uma classe* H , sendo denotada por $o(H)$. Assim, a classe $H = *11*0*1$ tem $o(H) = 4$.

Comprimento de Classe: refere-se à *distância entre a primeira e a última posições fixas de uma classe* H , sendo denotada por $\delta(H)$. Para a classe $H = *11*0*1$ tem-se $\delta(H) = 6$.

3.3.2 Enunciado do Teorema

Teorema 3.1 *O número de cadeias da classe H contidas na $(t + 1)$ -ésima população é estimado pela equação*

$$\hat{m}(H, t + 1) = m(H, t) \frac{f(H)}{\bar{f}} \left[1 - p_r \frac{\delta(H)}{k - 1} - o(H) p_m \right] \quad (3.1)$$

onde:

- $\hat{m}(H, t + 1)$: valor estimado do número de cadeias da classe H contidas na $(t + 1)$ -ésima população.
- $m(H, t)$: número de cadeias da classe H contidas na t -ésima população.
- $f(H)$: média dos valores da função de seleção de todas as cadeias da classe H contidas na t -ésima população.
- $\bar{f}$: média da função de seleção para todas as cadeias da t -ésima população.

- p_r : probabilidade de recombinação.
- p_m : probabilidade de mutação (de cada bit).
- k : comprimento das cadeias da classe H .

A equação (3.1) expressa matematicamente a influência dos operadores *reprodução*, *recombinação* e *mutação* no algoritmo genético. A contribuição do operador *reprodução* é obtida fazendo-se $p_r = p_m = 0$. As contribuições dos operadores *recombinação* e *mutação* são obtidas fazendo-se $p_m = 0$ e $p_r = 0$, respectivamente. Deve-se notar que como não há *recombinação* e *mutação* sem *reprodução*, a contribuição destes dois primeiros operadores leva necessariamente em conta a contribuição do operador *reprodução*.

Pela equação (3.1) conclui-se que classes de *cadeias* com valores $f(H)$ acima da média $\bar{f}$, comprimentos $\delta(H)$ e ordens $o(H)$ pequenos, recebem um número maior de descendentes na próxima geração. Estas classes são denominadas de *blocos fundamentais* (*building blocks*). Desta forma, pode-se concluir que os algoritmos genéticos têm como objetivo principal selecionar e recombinar os *blocos fundamentais* para dar origem a *cadeias* de qualidade potencialmente maior.

3.4 Mapeamento de Funções Objetivo

Como dito anteriormente, a *função de seleção* de um algoritmo genético obedece a duas restrições:

1. Achar a solução ótima para o problema de otimização implica em *maximizar* a função considerada.
2. A função deve ser não negativa.

Admite-se, de início, uma violação da primeira restrição. Quando deseja-se minimizar a *função objetivo*, trocar o sinal desta função não resolve a questão porque nada garante a não negatividade da *função objetivo*. Utiliza-se então o seguinte mapeamento:

$$f(x) = \begin{cases} C_{max} - g(x) & \text{se } g(x) < C_{max} \\ 0 & \text{caso contrário} \end{cases} \quad (3.2)$$

Neste mapeamento, tem-se

- $f(x)$: função de seleção.
- $g(x)$: função objetivo.
- C_{max} : coeficiente que pode ser o maior valor de $g(x)$ observado até aquele momento, ou o maior valor das últimas k gerações ou então o maior valor da *função objetivo* na população corrente.

Quando a segunda restrição é violada, utiliza-se o seguinte mapeamento:

$$f(x) = \begin{cases} C_{min} + g(x) & \text{se } g(x) + C_{min} > 0 \\ 0 & \text{caso contrário} \end{cases} \quad (3.3)$$

Neste caso, C_{min} representa o maior valor absoluto de $g(x)$ observado, por exemplo, nas últimas k gerações ou na população corrente.

3.5 Escala Linear para a Função de Seleção

O uso de escalas melhora o processo de seleção (operador *reprodução*) [Potts 94], sendo utilizada para [Goldberg 89]:

1. Controlar o número de cópias das melhores soluções para manter a diversidade das populações geradas, evitando problemas de convergência prematura [Potts 94].
2. Corrigir problemas de falta de sensibilidade da *função de seleção* (a função varia pouco para soluções boas e ruins).

Uma das escalas utilizadas é a linear, dada por

$$f' = a f + b \quad (3.4)$$

Na equação (3.4), tem-se

- f : função de seleção.
- f' : função de seleção em escala.
- a e b : coeficientes que definem a escala linear.

O cálculo destes dois coeficientes é feito com base nas seguintes condições:

$$\bar{f}' = \bar{f} \quad (3.5)$$

$$f'_{\max} = C_{\text{mult}} \bar{f} \quad (3.6)$$

A primeira condição, dada pela equação (3.5), faz com que a média dos valores da *função de seleção* permaneça inalterada após a aplicação da escala. A segunda condição, dada pela equação (3.6), define o melhor valor da *função de seleção em escala* como um múltiplo da média $\bar{f}$. Deste modo, ficam definidos dois pontos da equação (3.4), permitindo portanto o cálculo dos coeficientes a e b . A *figura 3.3* ilustra a escala linear.

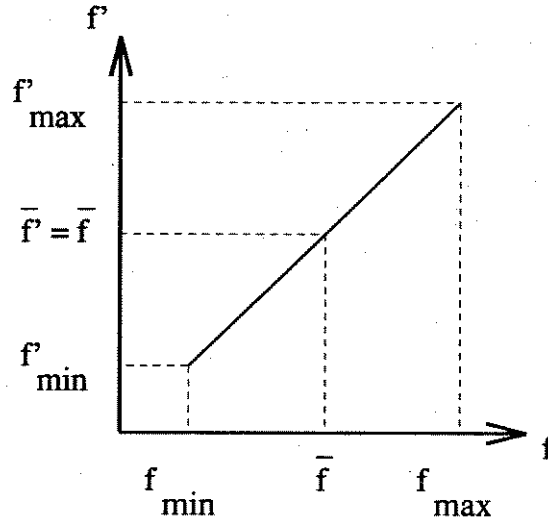


Figura 3.3: Escala linear para a função de seleção.

A constante C_{mult} é definida como o *número esperado de cópias do melhor indivíduo da população corrente, na próxima geração*, como pode ser notado pela substituição da equação (3.5) em (3.6). Tem-se então, a seguinte equação para uma população de n indivíduos:

$$C_{\text{mult}} = \frac{f'_{\max}}{\bar{f}'} = n \frac{f'_{\max}}{\sum_{j=1}^n f'_j} \quad (3.7)$$

A constante C_{mult} restringe, portanto, o número de cópias do melhor indivíduo da população corrente. Valores comumente usados para esta constante variam dentro do intervalo $[1.2; 2]$.

Como visto anteriormente, a definição da escala linear implica na fixação de dois pontos e posterior cálculo dos coeficientes a e b . Contudo, a escolha destes pontos pode acarretar um pequeno problema de inconsistência, como ilustra a *figura 3.4*. Nela, observa-se o segmento de reta, traçado com a linha mais fina, definido de acordo com os procedimentos normais da escala linear. Percebe-se, claramente, que o pior valor da *função de seleção* (f_{min}) foi mapeado em um valor negativo, violando assim a restrição de não negatividade exigida pelo algoritmo genético.

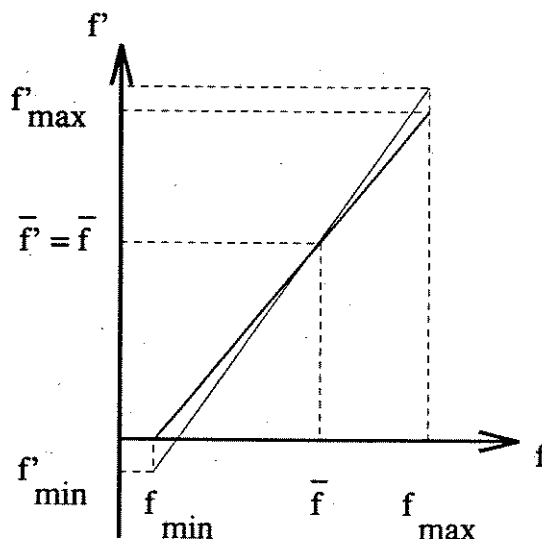


Figura 3.4: O problema da escala linear inconsistente.

Para contornar esta situação, redefine-se os pontos fixos, necessários ao cálculo dos coeficientes a e b , da seguinte maneira:

$$\bar{f}' = \bar{f} \quad (3.8)$$

$$f'_{min} = 0 \quad (f = f_{min}) \quad (3.9)$$

Assim, a única diferença destes dois novos pontos para os anteriores, está na equação (3.9) que mapeia o pior valor da *função de seleção* em zero.

Isto faz com que o valor de C_{mult} diminua (vide o segmento de reta traçado com a linha mais cheia na *figura 3.4*), restringindo um pouco mais o número esperado de cópias do melhor indivíduo da população corrente, na próxima geração.

3.6 O Código Multiparamétrico Concatenado de Ponto Fixo

Uma das características dos algoritmos genéticos é a manipulação de estruturas codificadas (*cadeias de bits*), referentes a possíveis soluções para um determinado problema. A codificação destas soluções permite ao algoritmo genético separá-las em partes, as quais são combinadas para dar origem a novas soluções. O *Código Multiparamétrico Concatenado de Ponto Fixo* (*Multiparameter Concatenated Fixed-Point Coding*) representa uma alternativa de codificação composta dos seguintes passos:

1. Definir um intervalo real $\mathcal{R} = [\mathcal{U}_{min}; \mathcal{U}_{max}]$ de possíveis valores assumidos por um determinado parâmetro $\mathcal{U}$.
2. Definir o intervalo inteiro $\mathcal{I} = [0; 2^l - 1]$, onde l é o número de bits utilizados para representar os valores assumidos por $\mathcal{U}$.
3. Definir o mapeamento $\mathcal{M} : \mathcal{I} \rightarrow \mathcal{R}$, de acordo com a seguinte expressão:

$$r = \frac{i}{2^l - 1} (\mathcal{U}_{max} - \mathcal{U}_{min}) + \mathcal{U}_{min} \quad (3.10)$$

Na equação (3.10), $r \in \mathcal{R}$ e $i \in \mathcal{I}$. Esta equação executa um *mapeamento linear* de $\mathcal{I}$ em $\mathcal{R}$. Note-se que para $i = 0$ e $i = 2^l - 1$ tem-se, respectivamente, $r = \mathcal{U}_{min}$ e $r = \mathcal{U}_{max}$.

Nesta codificação, o intervalo real fica discretizado, significando que apenas um número limitado de valores reais pode ser representado. Com uma quantidade de l bits, representa-se 2^l números inteiros binários de 00...0 a 11...1, e obtem-se $2^l - 1$ intervalos de discretização. Para se calcular o valor de cada intervalo de discretização, basta dividir $\mathcal{R}$ pelo número de intervalos. Desta forma, tem-se que:

$$\pi = \frac{U_{max} - U_{min}}{2^l - 1} \quad (3.11)$$

Na equação (3.11), π é chamada de *precisão do código de ponto fixo* e $1/\pi$ é o *nível de resolução*.

Substituindo (3.11) em (3.10), conclui-se que:

$$r = i\pi + U_{min} \quad (3.12)$$

O *Código Multiparamétrico Concatenado de Ponto Fixo*, dentro do contexto dos algoritmos genéticos, é utilizado como segue:

1. Converter cada uma das *cadeias de bits* da população corrente, da base binária para a decimal, obtendo uma série de números inteiros, cada um correspondendo a uma *cadeia*.
2. Fazer i igual a cada um destes números inteiros e calcular os respectivos valores de r através da equação (3.12). Estes valores de r representam as soluções no campo dos números reais.
3. Utilizar os valores de r para calcular os valores da *função de seleção* para as respectivas soluções da população corrente.
4. Prosseguir com o algoritmo genético, aplicando os operadores *reprodução*, *recombinação* e *mutação*, para dar origem a uma nova população.

Até o presente momento, a codificação de ponto fixo está restrita a apenas um parâmetro real. Contudo, é possível criar-se uma estrutura de código que envolva, ao mesmo tempo, mais de um parâmetro real. Toda a discussão anterior continua válida. A única diferença está nas *cadeias de bits*. Elas agora são o resultado da concatenação de várias *cadeias* que representam, individualmente, um único parâmetro real. Desta forma, cada parâmetro pode variar dentro de um intervalo de valores reais diferente e ter uma *precisão de código* também diferente. Nesta situação, quando se for efetuar a conversão de uma *cadeia* resultante de uma concatenação, deve-se separar as suas diversas partes (correspondentes a cada parâmetro real) e utilizar a equação (3.12) para converter cada uma das partes, isoladamente. De forma esquemática, tem-se [De Jong 80]:

1. Sejam $\mathcal{U}_1, \mathcal{U}_2, \dots, \mathcal{U}_n$ parâmetros pertencentes aos intervalos reais $\mathcal{R}_1, \mathcal{R}_2, \dots, \mathcal{R}_n$, respectivamente.
2. Valores nestes intervalos reais são representados por *cadeias* de $l_1, l_2, \dots, l_n$ bits, com *precisões* $\pi_1, \pi_2, \dots, \pi_n$ dadas pela equação (3.11).
3. Desta forma, pontos em um *espaço n -dimensional* são representados por *cadeias* de $L = \sum_{i=1}^n l_i$ bits, com cada segmento representando um valor dos n parâmetros $\mathcal{U}_1, \mathcal{U}_2, \dots, \mathcal{U}_n$.

3.7 Sumário

Neste capítulo apresentou-se o algoritmo genético básico. Foram detalhadas e discutidas as ações dos operadores *reprodução*, *recombinação* e *mutação*. A conclusão desta discussão resultou no *Teorema Fundamental dos Algoritmos Genéticos* que ressalta a capacidade destes algoritmos de combinar soluções de qualidade superior para gerar novas soluções potencialmente melhores.

Algumas técnicas utilizadas na implementação computacional dos algoritmos genéticos também foram abordadas. Inicialmente, enfatizou-se as transformações matemáticas aplicáveis a *funções objetivo* para que estas sejam consideradas *funções de seleção*. Em seguida, tratou-se dos problemas de convergência prematura e de soluções boas com valores da *função de seleção* próximos da média $\bar{f}$. Para solucioná-los, sugeriu-se a utilização de uma escala linear para a *função de seleção*.

O capítulo foi concluído com a apresentação do *Código Multiparamétrico Concatenado de Ponto Fixo*, dando ênfase à sua aplicação dentro do contexto dos algoritmos genéticos.

Capítulo 4

Sintonia DMC Utilizando Algoritmos Genéticos

4.1 Introdução

Neste capítulo apresenta-se a aplicação de um algoritmo genético para a determinação dos horizontes de previsão e controle de um controlador DMC. Um exemplo ilustra o desempenho do algoritmo proposto.

A sintonia de um controlador DMC é feita por dois grupos de parâmetros: as ponderações ECE, MSF e os horizontes de previsão e controle [García 88] [Prett 88] [Georgiou 88]. Utiliza-se o algoritmo genético para determinar os horizontes de previsão e controle, permitindo-se um ajuste iterativo das ponderações ECE e MSF pelo usuário. Este algoritmo é completamente independente da natureza do processo a ser controlado pelo algoritmo DMC, sendo igualmente válido para a simulação de processos com modelos lineares e não lineares. Todos os programas desenvolvidos para as simulações foram escritos na linguagem de programação do **Matlab**.

Inicialmente, tentou-se desenvolver o algoritmo de sintonia para determinar os valores das ponderações ECE e MSF, fixando os horizontes de previsão e controle com valores arbitrários. Verificou-se então a ineficiência desta solução, devido à grande sensibilidade que o algoritmo DMC apresentava às variações nos valores dos horizontes. Concluiu-se portanto, que como os horizontes tinham uma importância relativa maior, eles deveriam ser o alvo principal do algoritmo de sintonia, ficando as ponderações ECE e MSF fixas em valores unitários. Nesta segunda fase, constatou-se que nem sempre era

possível uma sintonia adequada considerando apenas as variações nos horizontes e mantendo fixas as ponderações. Concluiu-se então, que o algoritmo de sintonia tinha que priorizar os horizontes e permitir também um ajuste iterativo das ponderações.

Como descrito no **capítulo 3**, os algoritmos genéticos são eficientes na determinação de regiões de interesse dentro do espaço de soluções [Goldberg 89]. O objetivo, portanto, do algoritmo genético aqui apresentado é determinar uma região restrita de soluções que representem os horizontes de previsão e controle, dentro da qual pode-se escolher a *solução associada ao menor valor da função objetivo*. Pequenas variações desta solução também podem ser consideradas.

A segunda seção deste capítulo apresenta o algoritmo genético para a determinação dos horizontes de previsão e controle. A terceira seção apresenta a aplicação deste algoritmo em uma coluna de destilação de alta pureza. O capítulo é concluído com um resumo dos principais pontos discutidos.

4.2 O Algoritmo de Sintonia

O algoritmo genético para a sintonia de um controlador DMC, determinando os seus horizontes de previsão e controle, apresenta as seguintes etapas:

1. Definir os intervalos inteiros de variação dos horizontes de previsão e controle.
2. Definir os parâmetros do algoritmo genético: tamanho da população (n), probabilidade de recombinação (p_r) e mutação (p_m).
3. Definir o número de populações a serem geradas, excluindo a inicial ($maxpop$): critério de parada do algoritmo genético.
4. Definir o tipo de controle DMC: com restrições (QDMC) ou sem restrições.
5. Obter as restrições estabelecidas pelo processo, no caso do controle QDMC.
6. Considerar as ponderações ECE e MSF inicialmente unitárias, definir o horizonte de simulação (H) e o intervalo de amostragem (T).

7. Inicializar o contador de populações geradas: $c = 0$.
8. Gerar uma população inicial aleatória, contendo pares de horizontes de previsão e controle (cada elemento da população é uma *cadeia de bits* concatenada, representando os pares de horizontes segundo o *Código Multiparamétrico Concatenado de Ponto Fixo*).
9. Enquanto $c \leq \text{maxpop}$ faça:
 - (a) Para cada par de horizontes da população corrente faça:
 - i. Simular o controle DMC no processo desejado ao longo do horizonte H .
 - ii. Armazenar as respostas das variáveis de saída do processo.
 - iii. Calcular a seguinte função objetivo:

$$g = \frac{1}{rN} \sum_{j=1}^r \sum_{t=1}^N |y_j(t) - y_{sj}(t)| + \alpha K \quad (4.1)$$

onde:

- $y_j(t)$: resposta da j -ésima variável de saída no t -ésimo instante de simulação.
 - $y_{sj}(t)$: valor de referência da j -ésima variável de saída no t -ésimo instante de simulação.
 - r : número de variáveis de saída.
 - N : número de instantes de tempo simulados.
 - K : número de instantes de tempo não simulados ($H = N + K$).
 - α : *fator de diversidade da função objetivo* ($\alpha > 0$) - explicado mais adiante.
- (b) Transformar os valores calculados da função objetivo em valores de função de seleção, de acordo com o seguinte mapeamento:

$$f_j = \begin{cases} C_{\max} - g_j & \text{se } g_j < C_{\max} \\ 0 & \text{caso contrário} \end{cases} \quad j = 1, \dots, \text{tampop} \quad (4.2)$$

onde:

- g_j : j -ésimo valor da função objetivo.

- f_j : j -ésimo valor da função de seleção.
- $C_{max} = \max g_j$

(c) Aplicar a seguinte escala linear aos valores de função de seleção calculados:

$$f'_j = a f_j + b \quad (4.3)$$

onde:

- f'_j : j -ésimo valor da função de seleção após a escala.
- a, b : coeficientes a calcular segundo apresentado no **capítulo 3**, considerando $C_{mult} = 2$.

(d) Aplicar os operadores genéticos à população corrente, gerando uma nova população, e fazer $c \leftarrow c + 1$.

10. As melhores soluções (aquelas com menor valor da função objetivo) das populações geradas, representam uma região de interesse para os horizontes de previsão e controle, dentro dos intervalos de variação especificados. A melhor solução desta região é a candidata mais direta a ser a resposta final para a sintonia do controlador DMC.

Como os intervalos de variação dos horizontes de previsão e controle são inteiros, a *precisão do código multiparamétrico* fica automaticamente definida, pois os intervalos são discretos por natureza.

As *cadeias de bits* que representam os pares de horizontes apresentam a estrutura ilustrada na *figura 4.1*.

Cadeia de bits

1 1 0 1 0 0 0 1 1 1 0 1 0

Horizonte de Horizonte de
Previsão (p) Controle (M)

Figura 4.1: Estrutura codificada dos horizontes de previsão e controle.

Nesta figura, nota-se que os *bits* mais à esquerda representam o horizonte de previsão, enquanto que os *bits* mais à direita representam o horizonte de controle. Como o algoritmo DMC exige que $M \leq p$, caso o valor do horizonte de previsão seja menor que o valor do horizonte de controle, os valores são trocados entre si após a decodificação da *cadeia de bits*.

Para sistemas monovariáveis ($r = 1$), a função objetivo definida em (4.1) representa o *erro absoluto médio observado na saída do processo durante N intervalos de simulação* mais uma parcela de ponderação (αK). Para sistemas multivariáveis ($r > 1$), a função em (4.1) representa a *média de todos os erros absolutos observados nas saídas do processo durante N intervalos de simulação* mais uma parcela de ponderação (αK). Esta parcela permite que pares de horizontes de previsão e controle que não apresentam os melhores resultados, sejam aproveitados pelo algoritmo genético, por poderem se transformar em boas soluções pela simples mudança de alguns poucos *bits* em suas estruturas codificadas. Para estes pares o controle do processo se torna instável após alguns intervalos de simulação, até o instante onde ocorre a violação das restrições operacionais nas variáveis de saída, tornando o programa quadrático do algoritmo DMC ineficaz. Neste instante, a simulação é interrompida. Portanto, ao invés de descartar estas soluções, o algoritmo genético as mantém, penalizando os seus valores da função objetivo pela adição de um valor proporcional ao número de intervalos que deixaram de ser simulados. Quanto maior este número de intervalos, pior é a solução. Assim, em populações futuras, esta solução ou é descartada definitivamente ou recombinada para gerar outra solução melhor.

A constante de proporcionalidade envolvida (α) é definida como o *fator de diversidade da função objetivo*, pois o seu valor influencia diretamente na diversidade das populações geradas. Esta constante pode ser estimada da seguinte maneira: o pior valor para a função objetivo em (4.1) é $g = \alpha H$. Um valor muito bom para esta função, com base nas simulações feitas, pode ser estimado na ordem de $g = 0.1$. Assim, supondo que o pior valor tem 99% de chance de não ser selecionado para a próxima população, e que o valor considerado muito bom tem 1% de chance de não ser selecionado, tem-se, a partir de uma regra de três simples, que:

$$\begin{array}{ll} \alpha H & \longrightarrow 99\% \\ 0.1 & \longrightarrow 1\% \end{array}$$

$$\alpha = \frac{9.9}{H} \quad (4.4)$$

Durante a sintonia, dois problemas podem ocorrer: os pares de horizontes explorados pelo algoritmo genético geram muitas infactibilidades durante a simulação do controle DMC ao longo do horizonte H (caso se esteja utilizando o algoritmo QDMC), ou os valores da função objetivo para as melhores soluções das populações geradas se encontram altos. Nestas duas situações, recomenda-se parar a sintonia e reiniciá-la com valores maiores para as ponderações MSF. Desta forma, o controle DMC gera mudanças mais graduais para os sinais de controle, melhorando a convergência das variáveis controladas para os seus respectivos sinais de referência.

Ao final da sintonia, a melhor solução da região de interesse encontrada pelo algoritmo genético, pode fornecer respostas não muito suaves para as variáveis controladas. Neste caso, recomenda-se, novamente, um aumento nos valores das ponderações MSF. Desta forma, obtem-se respostas mais suaves e bem comportadas, porém mais lentas. Outras soluções da região de interesse ou pequenas variações destas, também podem ser tentadas. Uma vez ajustadas as ponderações MSF, pode-se repetir a sintonia com este novo ajuste, buscando horizontes de previsão e controle que minimizem ainda mais a função objetivo.

4.3 Exemplo: Aplicação no Controle de uma Coluna de Destilação de Alta Pureza

Nesta seção são apresentados os resultados da sintonia do algoritmo DMC no controle de uma coluna de destilação de alta pureza, ilustrando a metodologia proposta. A *figura 4.2* mostra o esquema da coluna.

O modelo linear simplificado da coluna é dado pelas equações [Morari 89]:

$$\begin{bmatrix} dY_D \\ dX_B \end{bmatrix} = \begin{bmatrix} \frac{87.8}{1+194s} & \frac{-87.8}{1+194s} + \frac{1.4}{1+15s} \\ \frac{108.2}{1+194s} & \frac{-108.2}{1+194s} - \frac{1.4}{1+15s} \end{bmatrix} \begin{bmatrix} dL \\ dV \end{bmatrix} \quad (4.5)$$

$$Y_D = \ln \frac{1}{1 - y_D} \quad (4.6)$$

$$X_B = \ln \frac{1}{x_B} \quad (4.7)$$

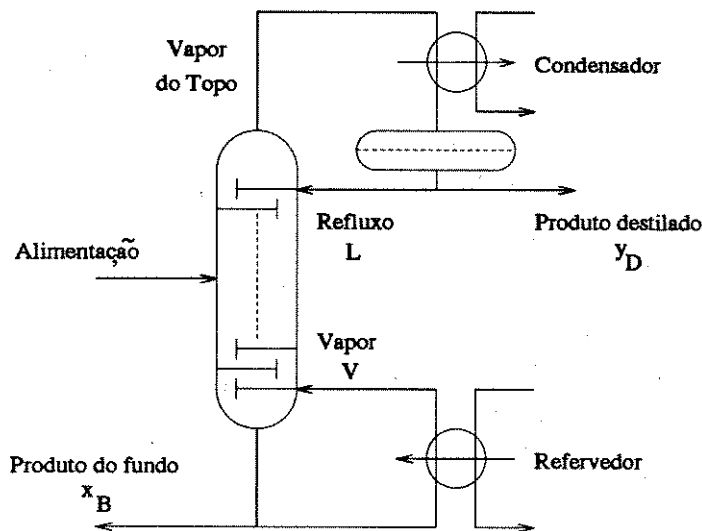


Figura 4.2: *Esquema da coluna de destilação de alta pureza.*

onde:

- Y_D : composição logarítmica do topo.
- X_B : composição logarítmica do fundo.
- y_D : composição do topo (variável controlada).
- x_B : composição do fundo (variável controlada).
- L : refluxo (variável manipulada).
- V : fluxo de vapor do refrervador (variável manipulada).

O controle DMC utilizado neste exemplo é sem restrições. Inicialmente, foram feitas duas simulações do algoritmo genético. Na primeira simulação, considerou-se as ponderações ECE e MSF unitárias, e os seguintes intervalos inteiros de variação para os horizontes: $p \in [1; 50]$ e $M \in [1; 25]$. Na segunda simulação, restringiu-se os intervalos de variação dos horizontes para: $p \in [1; 30]$ e $M \in [1; 2]$. Esta restrição foi realizada com base nos resultados obtidos na primeira simulação.

Os dados utilizados para o algoritmo genético foram os seguintes:

No. da populacao	1ª Simulação			2ª Simulação		
	Melhor p	Melhor M	Função objetivo	Melhor p	Melhor M	Função objetivo
0	20	1	2.119	23	1	2.113
1	20	1	2.119	23	1	2.113
2	20	1	2.119	21	1	2.099
3	20	1	2.119	22	1	2.098
4	20	1	2.119	22	1	2.098
5	23	1	2.113	22	1	2.098
6	23	1	2.113	21	1	2.099
7	23	1	2.113	21	1	2.099
8	23	1	2.113	22	1	2.098
9	24	1	2.135	22	1	2.098
10	24	1	2.135	22	1	2.098

Tabela 4.1: Duas simulações iniciais do algoritmo genético para a coluna.

- Tamanho da população: $n = 20$.
- Probabilidade de recombinação: $p_r = 80\%$.
- Probabilidade de mutação: $p_m = 1\%$.
- Critério de parada: $maxpop = 10$.
- Ponto de Operação: $y_D^0 = 99\%$ e $x_B^0 = 1\%$.
- Horizonte de simulação: $H = 30$.
- Intervalo de amostragem: $T = 1$ minuto.
- Fator de diversidade: $\alpha = \frac{9.9}{30} = 0.33$.

A *tabela 4.1* mostra a evolução do algoritmo genético para as duas simulações.

Os pares de horizontes apresentados na *tabela 4.1* representam regiões restritas de pontos dentro dos intervalos de variação especificados. Nas *figuras*

4.3 e 4.4 tem-se o resultado da simulação do controlador DMC para o melhor ponto destas regiões: $p = 22$ e $M = 1$.

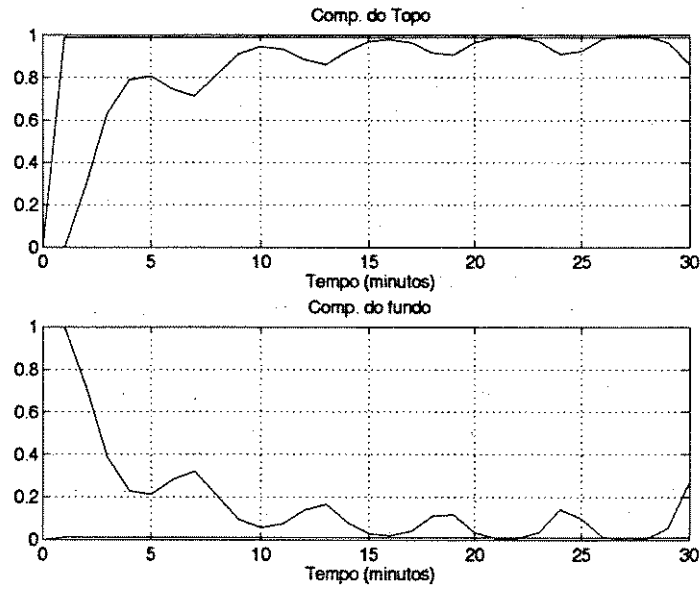


Figura 4.3: Respostas das variáveis controladas para $p = 22$, $M = 1$, ponderações ECE e MSF unitárias.

Observa-se que as variáveis controladas tendem aos seus valores de referência, porém não de uma forma gradual e suave, levando o sistema em malha fechada a uma condição de instabilidade. Aumentou-se então os valores das ponderações MSF para 2000 e observou-se a resposta das variáveis controladas para os mesmos valores anteriores dos horizontes. As figuras 4.5 e 4.6 apresentam os resultados.

Observa-se que as composições do topo e do fundo convergem para $y_D = 98.79\%$ e $x_B = 0.88\%$ de forma suave e gradual. Uma vez obtido um ajuste das ponderações ECE e MSF, repetiu-se o processo de sintonia para estes novos valores das ponderações, objetivando a obtenção de horizontes com um valor menor para a função objetivo. Na primeira simulação do algoritmo genético, utilizou-se os seguintes intervalos inteiros de variação para os horizontes: $p \in [1; 50]$ e $M \in [1; 25]$. Na segunda simulação, restringiu-se os intervalos iniciais para: $p \in [47; 60]$ e $M \in [11; 16]$.

A tabela 4.2 mostra a evolução do algoritmo genético para as duas simulações. Nesta tabela observa-se, pela primeira simulação, que a função

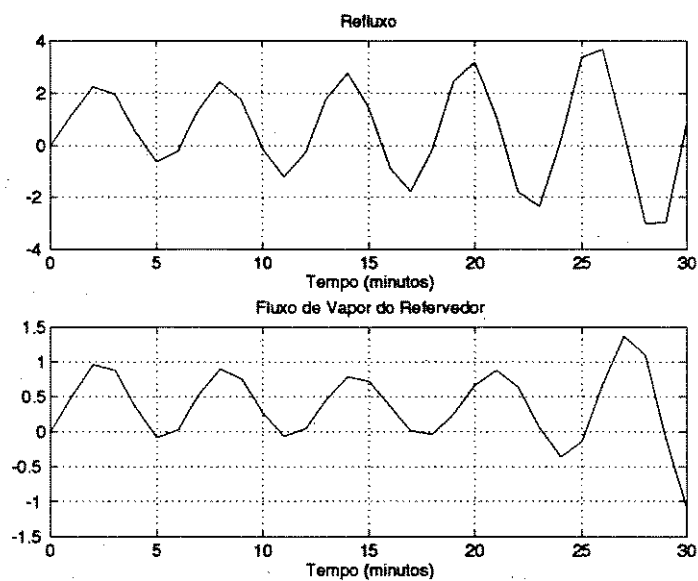


Figura 4.4: Sinais de controle para $p = 22$, $M = 1$, ponderações ECE e MSF unitárias.

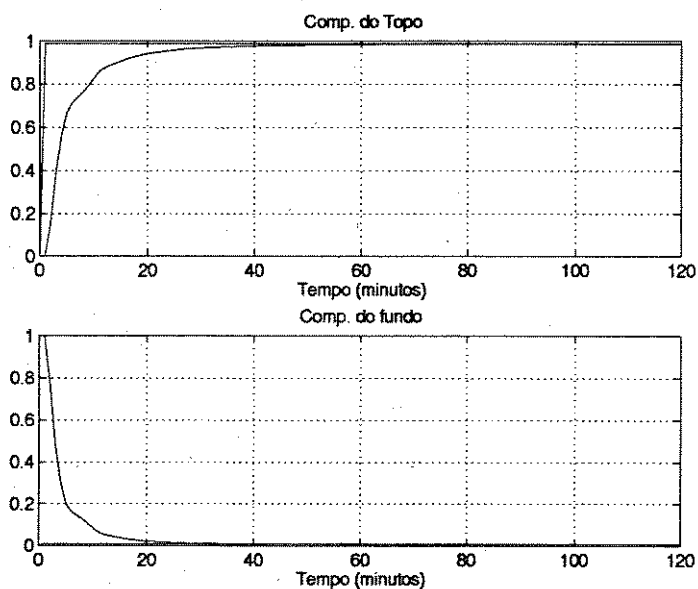


Figura 4.5: Respostas das variáveis controladas para $p = 22$, $M = 1$, ponderações ECE unitárias e ponderações MSF iguais a 2000.

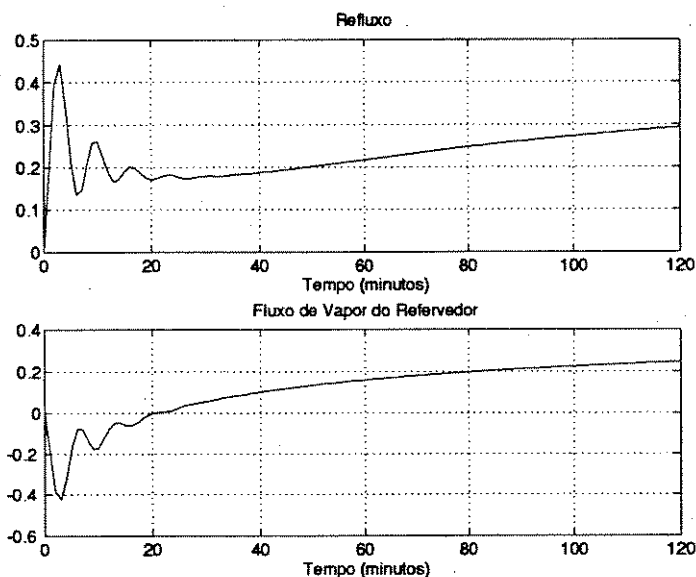


Figura 4.6: Sinais de controle para $p = 22$, $M = 1$, ponderações ECE unitárias e ponderações MSF iguais a 2000.

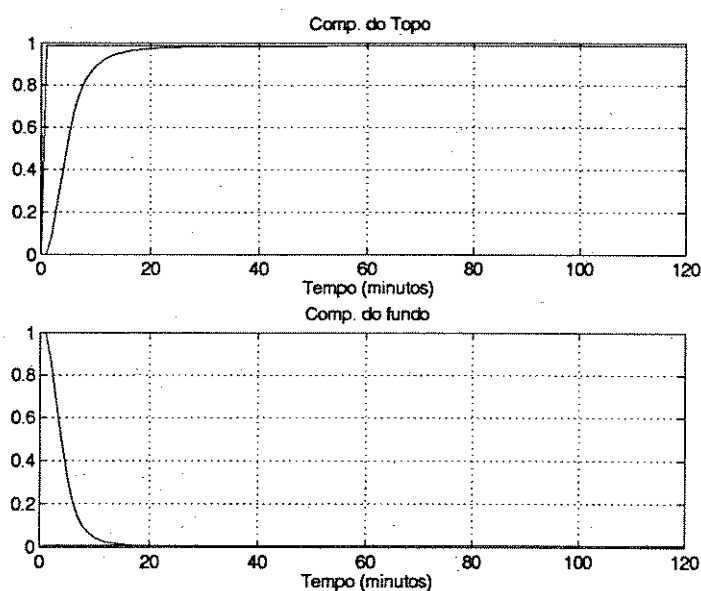
objetivo tende a diminuir mais, caso o intervalo de variação do horizonte de previsão tivesse um limite superior maior que 50. Por isto, restringiu-se o horizonte de previsão para o intervalo $[47; 60]$ na segunda simulação.

Observa-se que houve uma diminuição nos valores da função objetivo, em comparação com aqueles apresentados na *tabela* 4.1. Nas *figuras* 4.7 e 4.8 tem-se o resultado da simulação do controlador DMC para o melhor ponto das regiões apresentadas na *tabela* 4.2: $p = 60$ e $M = 11$.

Observa-se que as composições do topo e do fundo convergem para $y_D = 98.98\%$ e $x_B = 0.99\%$. Este resultado é melhor que o obtido anteriormente (*figuras* 4.5 e 4.6). Portanto, a sintonia escolhida para o controlador DMC é: $p = 60$, $M = 11$, ponderações ECE unitárias e ponderações MSF iguais a 2000.

No. da populacao	1ª Simulação			2ª Simulação		
	Melhor p	Melhor M	Função objetivo	Melhor p	Melhor M	Função objetivo
0	40	11	1.699	59	12	1.602
1	46	16	1.678	59	11	1.597
2	47	16	1.675	59	11	1.597
3	47	13	1.656	60	11	1.593
4	50	16	1.667	60	11	1.593
5	44	16	1.684	60	11	1.593
6	50	16	1.667	60	11	1.593
7	50	16	1.667	60	11	1.593
8	46	11	1.662	60	11	1.593
9	49	11	1.646	60	11	1.593
10	50	13	1.643	60	11	1.593

Tabela 4.2: Duas simulações adicionais do algoritmo genético para a coluna.

Figura 4.7: Respostas das variáveis controladas para $p = 60$, $M = 11$, ponderações ECE unitárias e ponderações MSF iguais a 2000.

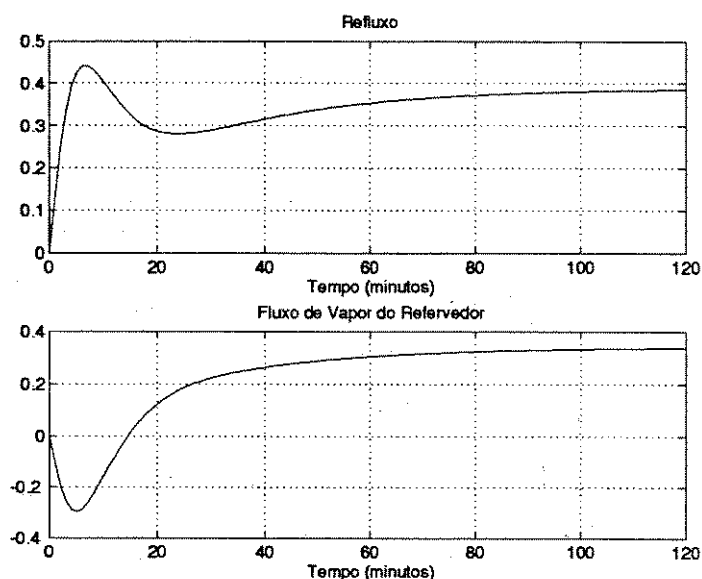


Figura 4.8: Sinais de controle para $p = 60$, $M = 11$, ponderações ECE unitárias e ponderações MSF iguais a 2000.

4.4 Sumário

Neste capítulo, apresentou-se a aplicação de um algoritmo genético para a determinação dos horizontes de previsão e controle de um controlador DMC. O objetivo do algoritmo é determinar regiões restritas de pontos que representam os horizontes de previsão e controle. Dentro destas regiões, pode-se então escolher a solução associada ao menor valor da função objetivo. Pequenas variações destas soluções também podem ser consideradas. O algoritmo permite ainda um ajuste iterativo das ponderações ECE e MSF, consideradas inicialmente unitárias.

A função objetivo escolhida para o algoritmo genético, baseia-se nos erros absolutos entre as variáveis de saída e os seus respectivos sinais de referência. Pares de horizontes ruins são aproveitados pelo algoritmo genético, através de uma penalização nos valores correspondentes da função objetivo. A manutenção destes pares, com baixas probabilidades de seleção, aumenta a diversidade das populações, permitindo um melhor desempenho do algoritmo genético.

A aplicação do algoritmo proposto a uma coluna de destilação de alta

pureza, visou demonstrar não somente a eficiência do método, bem como a maneira como este deve ser empregado.

Capítulo 5

Aplicação no Controle do Processo Shell

5.1 Introdução

Neste capítulo, apresenta-se o controle do Processo Shell, utilizando-se o algoritmo genético para a sintonia de controladores DMC. O Processo Shell, descrito pela *Shell Development Company*, é uma coluna fracionadora de óleo pesado com objetivos e restrições de controle bem definidas [Prett 90].

O objetivo desta aplicação é ilustrar a utilização da metodologia descrita no capítulo 4, a um processo com restrições operacionais em suas variáveis de entrada e saída.

A segunda seção deste capítulo apresenta a descrição do Processo Shell. Esta seção está dividida em quatro subseções: modelo do processo, objetivos de controle, restrições de controle e perturbação no processo. A terceira seção apresenta a aplicação do algoritmo genético no controle deste processo, supondo ganhos fixos. Na quarta seção, supõe-se ganhos variáveis. A quinta e última seção apresenta um sumário com as conclusões sobre os resultados obtidos.

5.2 O Processo Shell

A figura 5.1 apresenta o esquema da coluna fracionadora de óleo pesado. A descrição do funcionamento desta coluna pode ser vista em [Prett 90].

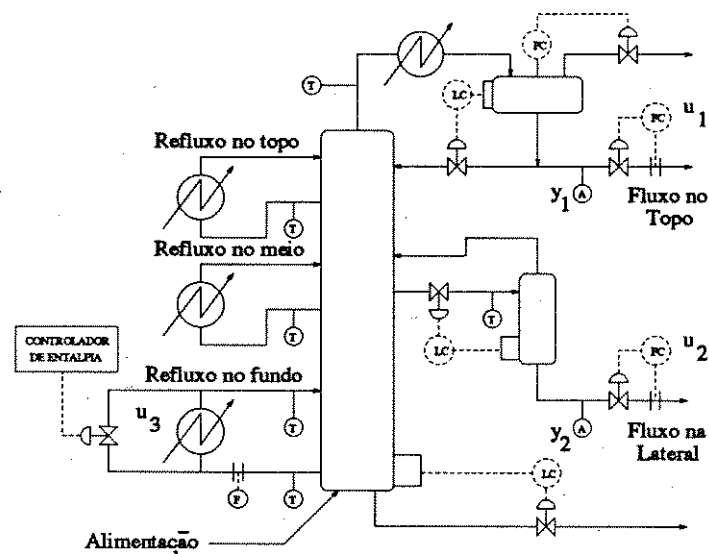


Figura 5.1: Esquema da coluna fracionadora de óleo pesado.

Para a aplicação apresentada neste capítulo, considerou-se as seguintes variáveis de entrada e saída:

- Variáveis de entrada:
 - Variáveis manipuladas: fluxo de produto no topo (u_1) e fluxo de produto na lateral (u_2).
 - Variável de perturbação: refluxo no fundo (u_3).
- Variáveis de saída: composição do produto no topo (y_1) e composição do produto na lateral (y_2).

5.2.1 Modelo do Processo

As funções de transferência discretas que relacionam as variáveis de entrada e saída acima, são do tipo:

$$G(z) = \frac{K z^{-d} (b_0 z + b_1)}{z + a_1} \quad (5.1)$$

	Fluxo no Topo (u_1)	Fluxo na Lateral (u_2)	Refluxo no Fundo (u_3)
Composição no Topo (y_1)	$K = 4.05$ $d = 7$ $a_1 = -0.9231$ $b_0 = 0.0198$ $b_1 = 0.0571$	$K = 1.77$ $d = 7$ $a_1 = -0.9355$ $b_0 = 0.0$ $b_1 = 0.0645$	$K = 5.88$ $d = 7$ $a_1 = -0.9231$ $b_0 = 0.0198$ $b_1 = 0.0571$
Composição na Lateral (y_2)	$K = 5.39$ $d = 5$ $a_1 = -0.9231$ $b_0 = 0.0392$ $b_1 = 0.0377$	$K = 5.72$ $d = 4$ $a_1 = -0.9355$ $b_0 = 0.0328$ $b_1 = 0.0317$	$K = 6.90$ $d = 4$ $a_1 = -0.9048$ $b_0 = 0.0247$ $b_1 = 0.0705$

Tabela 5.1: Coeficientes do modelo discreto do Processo Shell.

O intervalo de amostragem considerado para estas funções foi de 4 minutos. A *tabela* 5.1 apresenta os valores dos coeficientes da equação (5.1) para todos os pares de variáveis de entrada e saída.

Os ganhos das funções de transferência da *tabela* 5.1 são incertos em regime. Estas incertezas são apresentadas na *tabela* 5.2. Nesta tabela, $-1 \leq \Delta \varepsilon_i \leq 1$; $i = 1, 2, 3$.

5.2.2 Objetivos de Controle

Para a aplicação apresentada neste capítulo, os objetivos de controle são:

- Manter a composições do produto no topo e na lateral dentro do intervalo (0.0 ± 0.005) em regime permanente.
- Minimizar o valor do refluxo no fundo.

5.2.3 Restrições de Controle

As restrições sobre as variáveis de entrada e saída são as seguintes:

	Fluxo no Topo (u_1)	Fluxo na Lateral (u_2)	Refluxo no Fundo (u_3)
Composição no Topo (y_1)	$4.05 + 2.11 \Delta \varepsilon_1$	$1.77 + 0.39 \Delta \varepsilon_2$	$5.88 + 0.59 \Delta \varepsilon_3$
Composição na Lateral (y_2)	$5.39 + 3.29 \Delta \varepsilon_1$	$5.72 + 0.57 \Delta \varepsilon_2$	$6.90 + 0.89 \Delta \varepsilon_3$

Tabela 5.2: Incertezas nos ganhos em regime do Processo Shell.

- Os fluxos de produto no topo e na lateral estão restritos ao intervalo $[-0.5; 0.5]$.
- Os valores do refluxo no fundo estão restritos ao intervalo $[-0.5; 0.5]$.
- As mudanças nas variáveis manipuladas estão limitadas ao máximo de 0.05 por minuto (0.20 por 4 minutos).
- As composições do produto no topo e na lateral estão restritas ao intervalo $[-0.5; 0.5]$.

5.2.4 Perturbação no Processo

A variável de refluxo no fundo é considerada, nesta aplicação, como uma perturbação no processo. Contudo, um dos objetivos de controle é, justamente, a minimização do valor desta variável. Pelas restrições de controle, o valor ideal para o refluxo no fundo seria -0.5, porém para que não ocorresse violação das restrições de controle, otimizou-se o refluxo no fundo em -0.3. Desta forma, resolveu-se perturbar o processo com o seguinte sinal:

$$u_3 = -0.3(1 - e^{-0.23t}) \quad (5.2)$$

Pela equação (5.2), o refluxo no fundo converge para o valor -0.3 de forma gradual.

5.3 Controle do Processo Shell com Ganhos Fixos

O controle DMC utilizado neste exemplo é com restrições (algoritmo QDMC) e supõe $\Delta \varepsilon_i = 0$; $i = 1, 2, 3$. Inicialmente, considerou-se as ponderações ECE e MSF unitárias, porém observou-se que os pares de horizontes obtidos pelo algoritmo genético, geravam muitas infactibilidades durante a simulação do controle DMC ao longo do horizonte de simulação. Resolveu-se então, reiniciar o processo de sintonia com valores maiores para as ponderações MSF. Os valores escolhidos para estas ponderações foram iguais a 90.

Realizou-se duas simulações do algoritmo genético. Na primeira simulação, os intervalos inteiros de variação para os horizontes foram: $p \in [1; 50]$ e $M \in [1; 25]$. Na segunda simulação, restringiu-se os intervalos de variação dos horizontes para: $p \in [1; 30]$ e $M \in [1; 2]$. Esta restrição foi feita com base nos resultados obtidos na primeira simulação.

Os dados utilizados para o algoritmo genético foram os seguintes:

- Tamanho da população: $n = 20$.
- Probabilidade de recombinação: $p_r = 80\%$.
- Probabilidade de mutação: $p_m = 1\%$.
- Critério de parada: $maxpop = 10$.
- Ponto de Operação: $y_1^0 = 0$ e $y_2^0 = 0$.
- Horizonte de simulação: $H = 25$.
- Intervalo de amostragem: $T = 4$ minutos.
- Fator de diversidade: $\alpha = \frac{9.9}{25} = 0.396$.

A *tabela 5.3* mostra a evolução do algoritmo genético para as duas simulações.

Os pares de horizontes apresentados na *tabela 5.3* representam regiões restritas de pontos dentro dos intervalos de variação especificados. Nas *figuras 5.2* e *5.3* tem-se o resultado da simulação do controle DMC para o melhor ponto destas regiões: $p = 23$ e $M = 1$.

No. da populacao	1ª Simulação			2ª Simulação		
	Melhor p	Melhor M	Função objetivo	Melhor p	Melhor M	Função objetivo
0	30	1	0.0953	24	1	0.0929
1	42	1	0.1020	24	1	0.0929
2	30	1	0.0953	24	1	0.0929
3	27	1	0.0937	24	1	0.0929
4	30	1	0.0953	23	1	0.0928
5	27	1	0.0937	23	1	0.0928
6	28	1	0.0942	23	1	0.0928
7	28	1	0.0942	23	1	0.0928
8	28	1	0.0942	23	1	0.0928
9	27	1	0.0937	24	1	0.0929
10	27	1	0.0937	24	1	0.0929

Tabela 5.3: Duas simulações do algoritmo genético para o Processo Shell com ganhos fixos.

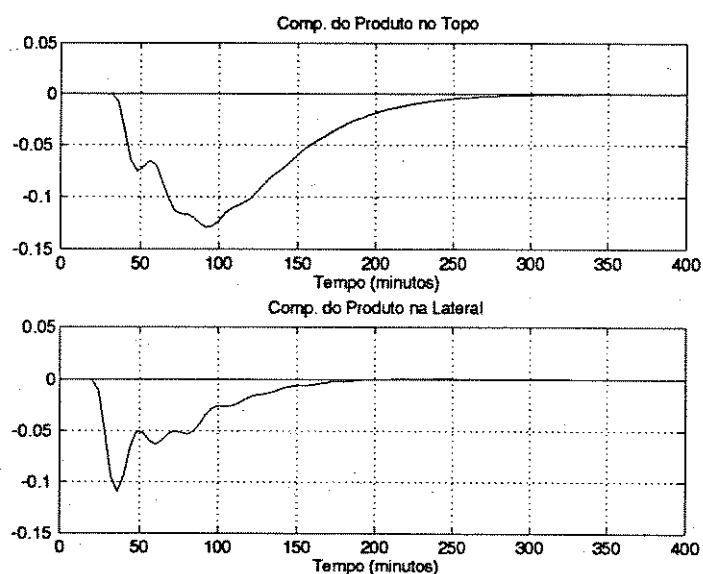


Figura 5.2: Respostas das variáveis controladas para $p = 23$, $M = 1$, ponderações ECE unitárias e ponderações MSF iguais a 90.

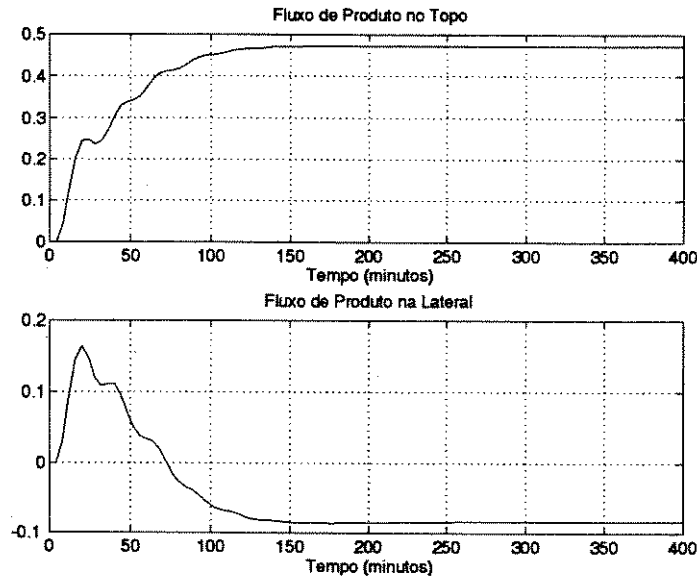


Figura 5.3: Sinais de controle para $p = 23$, $M = 1$, ponderações ECE unitárias e ponderações MSF iguais a 90.

Observa-se que as variáveis controladas convergem para $y_1 = 0.0001$ e $y_2 = 0.0002$, atendendo ao objetivo de controle; contudo, as curvas não se mostram suaves. As simulações demonstraram que aumentando-se os valores das ponderações MSF, não se consegue uma suavização adequada, além de tornar as respostas mais lentas. Optou-se então por pequenas variações nos pares de horizontes contidos na região de interesse encontrada pelo algoritmo genético. As figuras 5.4 e 5.5 mostram o resultado da simulação para $p = 23$ e $M = 2$.

Nota-se que as variáveis controladas convergem para $y_1 = 0.0000$ e $y_2 = 0.0001$. As curvas estão mais suaves, porém houve um aumento de aproximadamente 15% e 80% na variação máxima de amplitude das composições no topo e na lateral, respectivamente.

Como último teste, selecionou-se a melhor solução encontrada na primeira execução do algoritmo genético ($p = 27$ e $M = 1$) e, como no caso anterior, aumentou-se o horizonte de controle para 2. As figuras 5.6 e 5.7 apresentam o resultado da simulação.

Nota-se que as variáveis controladas convergem para $y_1 = 0.0000$ e $y_2 = 0.0002$. As curvas são muito parecidas com as obtidas anteriormente,

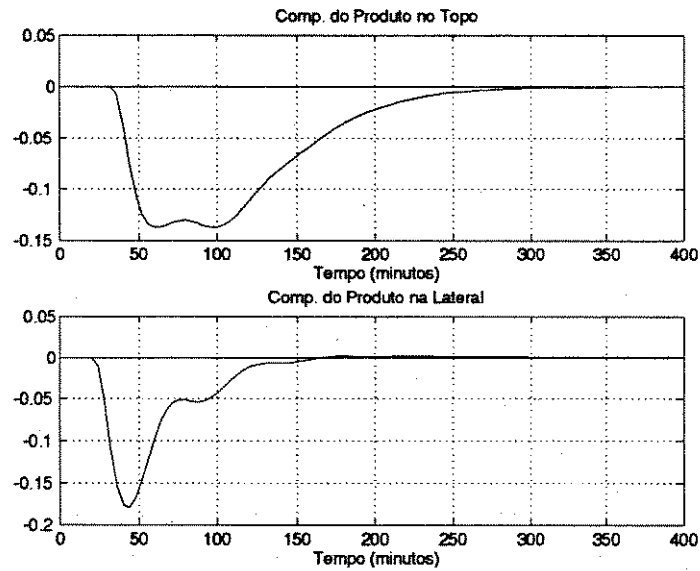


Figura 5.4: Respostas das variáveis controladas para $p = 23$, $M = 2$, ponderações ECE unitárias e ponderações MSF iguais a 90.

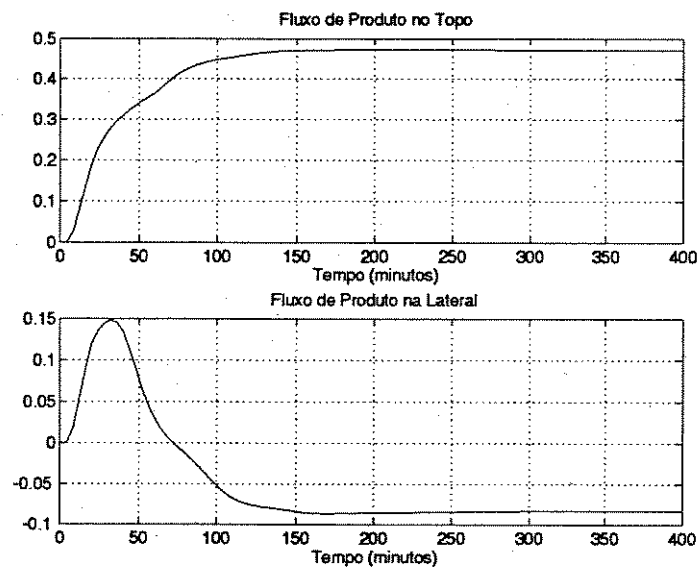


Figura 5.5: Sinais de controle para $p = 23$, $M = 2$, ponderações ECE unitárias e ponderações MSF iguais a 90.

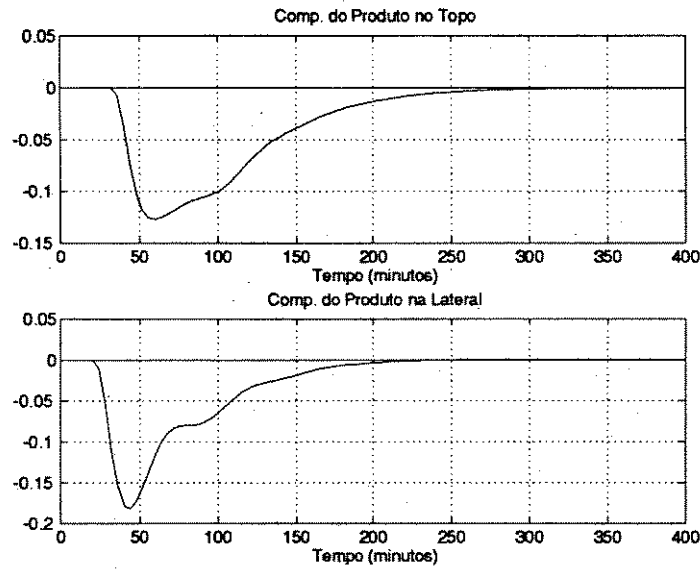


Figura 5.6: Respostas das variáveis controladas para $p = 27$, $M = 2$, ponderações ECE unitárias e ponderações MSF iguais a 90.

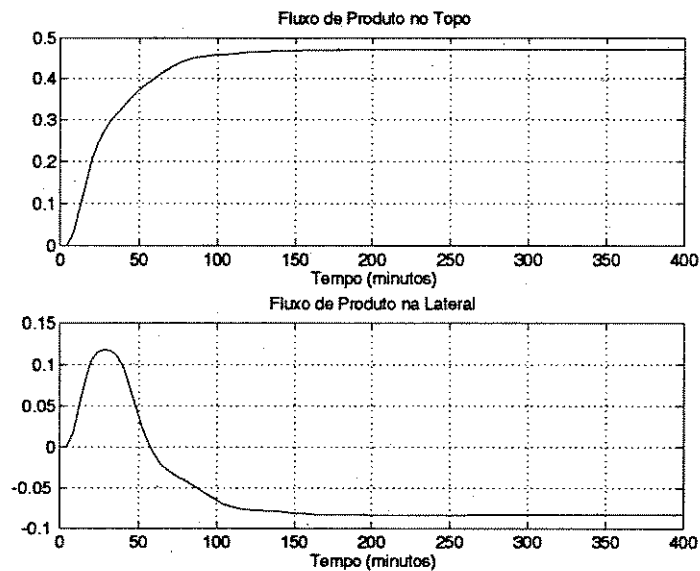


Figura 5.7: Sinais de controle para $p = 27$, $M = 2$, ponderações ECE unitárias e ponderações MSF iguais a 90.

porém observa-se uma diminuição em torno de 13% na variação máxima de amplitude da composição no topo, e um aumento, de aproximadamente 33%, no tempo de resposta da composição na lateral.

Pelo critério do tempo de resposta das variáveis de saída, a melhor solução encontrada é: $p = 23$, $M = 2$, ponderações ECE unitárias e ponderações MSF iguais a 90.

5.4 Controle do Processo Shell com Ganhos Variáveis

O controle DMC utilizado neste exemplo é com restrições (algoritmo QDMC) e supõe $\Delta \varepsilon_3$ variando como mostra a *figura 5.8*. Por esta figura, nota-se que $\Delta \varepsilon_3 = -1$ no intervalo [40; 200] minutos e zero nos demais instantes de tempo. As ponderações ECE continuaram unitárias e as ponderações MSF foram mantidas em 90.

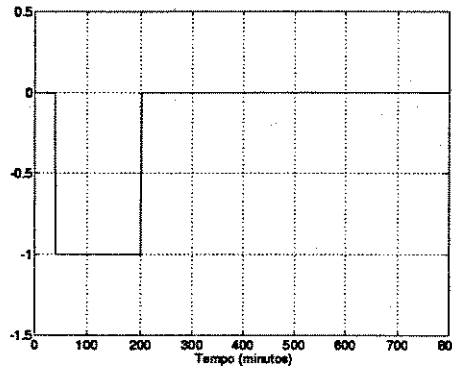


Figura 5.8: Variação de $\Delta \varepsilon_3$ durante as simulações.

Realizou-se duas simulações do algoritmo genético. Na primeira simulação, os intervalos inteiros de variação para os horizontes foram: $p \in [1; 50]$ e $M \in [1; 25]$. Na segunda simulação, restringiu-se os intervalos de variação dos horizontes para: $p \in [1; 30]$ e $M \in [1; 5]$.

Os dados utilizados para o algoritmo genético foram os seguintes:

- Tamanho da população: $n = 20$.

- Probabilidade de recombinação: $p_r = 80\%$.
- Probabilidade de mutação: $p_m = 1\%$.
- Critério de parada: $maxpop = 10$.
- Ponto de Operação: $y_1^0 = 0$ e $y_2^0 = 0$.
- Horizonte de simulação: $H = 40$.
- Intervalo de amostragem: $T = 4$ minutos.
- Fator de diversidade: $\alpha = \frac{9.9}{40} = 0.2475$.

A *tabela 5.4* mostra a evolução do algoritmo genético para as duas simulações. Nesta tabela observa-se, pela primeira simulação, que a função objetivo tende a diminuir mais para valores do horizonte de controle pertos de 1. Por isto, restringiu-se o horizonte de controle para o intervalo $[1; 5]$ na segunda simulação.

Os pares de horizontes apresentados na *tabela 5.4* representam regiões restritas de pontos dentro dos intervalos de variação especificados. Nas *figuras 5.9* e *5.10* tem-se o resultado da simulação do controle DMC para o melhor ponto destas regiões: $p = 20$ e $M = 1$.

Observa-se que as variáveis controladas convergem para $y_1 = 0.0000$ e $y_2 = 0.0002$, atendendo ao objetivo de controle; contudo, as curvas de resposta não se mostram muito suaves. Visando suavizá-las mais, optou-se pelo segundo melhor ponto encontrado pelo algoritmo genético, $p = 24$ e $M = 1$, porém os resultados foram muito parecidos com os anteriores. Optou-se então pelo ponto $p = 23$ e $M = 2$, presente na *tabela 5.4* e escolhido como o melhor ajuste para o controle do Processo Shell com ganhos fixos. As *figuras 5.11* e *5.12* mostram o resultado da simulação para $p = 23$ e $M = 2$.

Nota-se que as variáveis controladas convergem para $y_1 = 0.0000$ e $y_2 = 0.0002$. As curvas estão mais suaves, porém houve um aumento de aproximadamente 43% e 36% na variação máxima de amplitude das composições no topo e na lateral, respectivamente. Uma vez que estes aumentos de amplitude respeitam as restrições operacionais e considerando que a solução $p = 23$, $M = 2$, ponderações ECE unitárias e ponderações MSF iguais a 90, é válida também para o controle com ganhos fixos, resolveu-se estabelecer esta solução para o controle com ganhos variáveis apresentado nesta seção.

No. da populacao	1ª Simulação			2ª Simulação		
	Melhor p	Melhor M	Função objetivo	Melhor p	Melhor M	Função objetivo
0	23	19	0.05077	20	1	0.03850
1	23	11	0.05083	24	2	0.05596
2	24	1	0.04476	23	5	0.05405
3	24	1	0.04476	23	2	0.05335
4	26	1	0.05063	19	2	0.05243
5	23	18	0.05077	23	2	0.05335
6	24	20	0.05186	24	2	0.05596
7	22	18	0.05120	23	5	0.05405
8	22	18	0.05120	21	5	0.05302
9	22	16	0.05120	21	2	0.04937
10	22	10	0.05107	21	2	0.04937

Tabela 5.4: Duas simulações do algoritmo genético para o Processo Shell com ganhos variáveis.

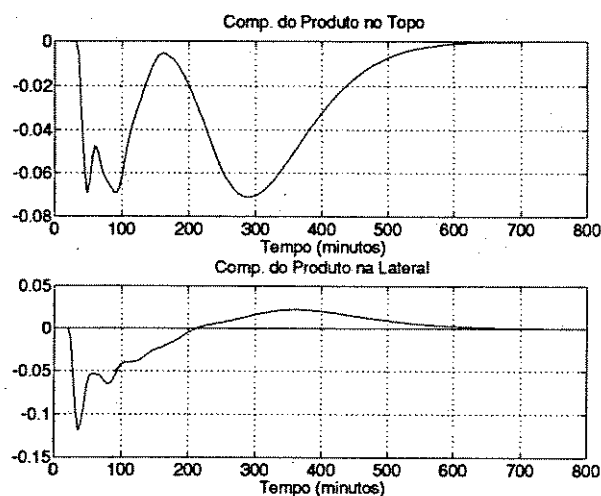


Figura 5.9: Respostas das variáveis controladas para $p = 20$, $M = 1$, ponderações ECE unitárias e ponderações MSF iguais a 90.

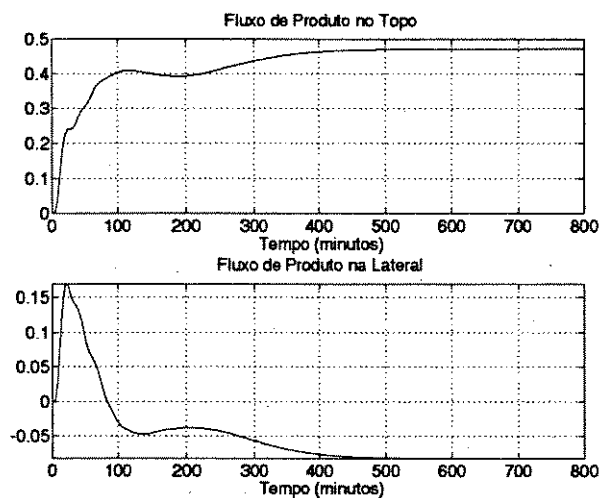


Figura 5.10: Sinais de controle para $p = 20$, $M = 1$, ponderações ECE unitárias e ponderações MSF iguais a 90.

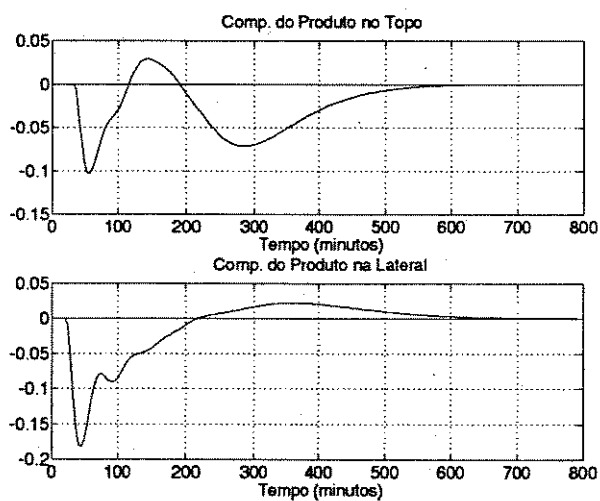


Figura 5.11: Respostas das variáveis controladas para $p = 23$, $M = 2$, ponderações ECE unitárias e ponderações MSF iguais a 90.

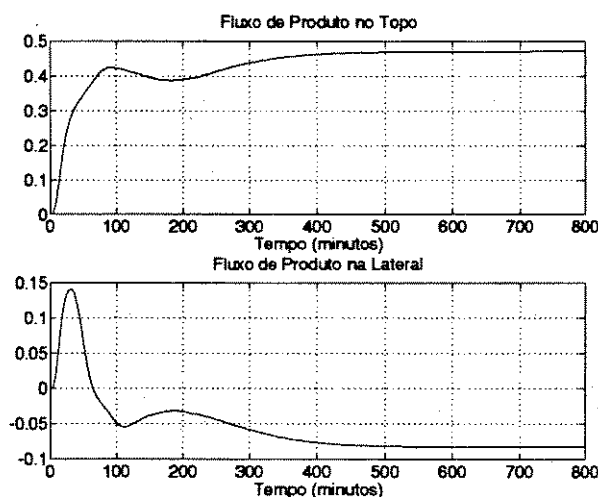


Figura 5.12: Sinais de controle para $p = 23$, $M = 2$, ponderações ECE unitárias e ponderações MSF iguais a 90.

5.5 Sumário

Apresentou-se a aplicação do algoritmo genético, para sintonia de controladores DMC, a uma coluna fracionadora de óleo pesado, conhecida como Processo Shell. Utilizou-se um modelo reduzido desta coluna em conjunto com objetivos e restrições de controle pré-estabelecidos.

Os resultados obtidos demonstram a rapidez com que o algoritmo genético encontra uma região de interesse dentro dos intervalos de variação dos horizontes de previsão e controle. Além disso, constatou-se como alguns dos pares de horizontes contidos nesta região, e pequenas variações destes, fornecem opções de sintonia para o controlador DMC.

Conclusões e Perspectivas

Este trabalho descreveu a aplicação de um algoritmo genético para a sintonia de controladores DMC. O resultado foi uma metodologia de sintonia que além de fornecer bons valores para os horizontes de previsão e controle, permite um ajuste iterativo das ponderações ECE e MSF com o objetivo de se obter uma sintonia final para o controlador DMC. Esta metodologia, válida para a simulação de processos com modelos lineares e não lineares, foi aplicada no controle de duas colunas com o objetivo de ilustrar a sua validade.

A aplicação do algoritmo genético para a sintonia de controladores DMC, permitiu constatar a rapidez com que este algoritmo seleciona regiões de interesse que procuram minimizar a função objetivo. O algoritmo genético é bastante eficiente em restringir drasticamente o espaço de soluções para um pequeno conjunto de pontos. Este conjunto dá ao usuário uma flexibilidade maior para conjugar a otimização da função objetivo do algoritmo genético com a satisfação de outros critérios (amplitude do sobresinal, tempos de subida e resposta, etc.).

Observou-se também que a maneira como a função objetivo foi definida, tornou o algoritmo insensível ao problema de suavidade da resposta da saída. Este problema está diretamente associado aos incrementos nas ações de controle. Quanto menores os incrementos nas ações de controle, mais suaves são as curvas de resposta. Desta forma, aumentando-se as ponderações MSF, e consequentemente diminuindo a magnitude dos incrementos nas ações de controle, consegue-se respostas mais suaves, porém mais lentas.

O algoritmo genético comprovou a maior sensibilidade às variações nos horizontes de previsão e controle em relação às ponderações ECE e MSF. Sobre o ajuste destas ponderações, concluiu-se que as ponderações ECE tiveram a sua ação incluída na função objetivo do algoritmo genético, ficando com valores fixos unitários. Já as ponderações MSF foram sempre ajustadas para mais, com o objetivo de suavizar as curvas de resposta e dar mais

estabilidade ao sistema de controle, caso fosse necessário.

Como perspectivas deste trabalho, sugere-se:

- Incorporar à função objetivo um termo envolvendo os incrementos nas ações de controle.
- Permitir a escolha da função objetivo segundo critérios diferentes do *erro absoluto*, tais como: *erro absoluto vezes o tempo*, *erro absoluto vezes o tempo ao quadrado*, *erro quadrático*, *erro quadrático vezes o tempo* e o *erro quadrático vezes o tempo ao quadrado* (a inclusão do tempo na função objetivo minimiza o tempo de resposta do sistema em malha fechada).
- Incorporar ao algoritmo genético mais um parâmetro a otimizar, representando o valor a ser assumido por todas as ponderações MSF.
- Procurar na literatura resultados que permitam uma delimitação adequada dos intervalos de variação dos horizontes de previsão, controle e ponderações MSF.
- Gerar um código executável para este algoritmo, objetivando não somente uma maior rapidez de processamento, mas também a aplicação deste algoritmo na indústria.

Anexo I: Teorema Fundamental dos Algoritmos Genéticos

A seguir determina-se o efeito de cada um dos operadores genéticos, combinando-os posteriormente para se chegar à forma final do teorema.

Efeito do Operador Reprodução

Para se determinar o efeito do operador *reprodução* quando aplicado a uma classe H , define-se:

- $f_j(H)$: valor da *função de seleção* para a j -ésima cadeia da classe H contida na t -ésima população.
- f_j : valor da *função de seleção* para a j -ésima cadeia da t -ésima população.
- n : tamanho da população.

Logo:

$$f(H) = \frac{\sum_{j=1}^{m(H,t)} f_j(H)}{m(H,t)} \quad (0.1)$$

$$\bar{f} = \frac{\sum_{j=1}^n f_j}{n} \quad (0.2)$$

Deseja-se relacionar as variáveis $m(H, t)$ com $\hat{m}(H, t + 1)$. Para tanto, considera-se a *esperança matemática* como a melhor estimativa de $\hat{m}(H, t + 1)$. Desta forma, a partir das equações (0.1) e (0.2), tem-se que:

$$\hat{m}(H, t + 1) = n \left[\frac{\sum_{j=1}^{m(H, t)} f_j(H)}{\sum_{j=1}^n f_j} \right] = m(H, t) \frac{f(H)}{\bar{f}} \quad (0.3)$$

A equação (0.3) representa a influência do operador *reprodução* no algoritmo genético. Por esta equação conclui-se que:

- Classes com valores $f(H)$ acima da média $\bar{f}$ ($f(H) > \bar{f}$), recebem um número maior de cópias na próxima geração.
- Classes com valores $f(H)$ abaixo da média $\bar{f}$ ($f(H) < \bar{f}$), recebem um número menor de cópias na próxima geração.

Isto é, o operador *reprodução* valoriza as soluções de melhor qualidade, atribuindo-lhes um valor esperado de cópias maior que aquele atribuído às soluções de qualidade inferior.

Efeito do Operador Recombinação

A operação de *recombinação* caracteriza-se pela ruptura de duas *cadeias* em um determinado ponto, com posterior troca de informação entre elas. Esta ruptura pode causar a destruição de uma das classes envolvidas na operação. As condições sob as quais este fato pode ocorrer são ilustradas com auxílio de um exemplo.

Exemplo 0.1 Na figura 0.1, observa-se uma operação de *recombinação* aplicada a duas classes de cadeias, com o ponto de cruzamento escolhido na terceira posição. Uma análise das duas novas classes resultantes da operação, permite concluir que a classe H_1 foi destruída, pelo fato de suas duas posições fixas estarem em lados opostos ao ponto de cruzamento. Já um subconjunto da classe H_2 sobreviveu, justamente pelo motivo inverso.

Este exemplo, apesar de particular, ilustra que classes com comprimentos $\delta(H)$ pequenos têm maiores chances de sobreviver, porque a probabilidade do ponto de cruzamento ser escolhido entre duas posições fixas muito próximas, é pequena. Seja:

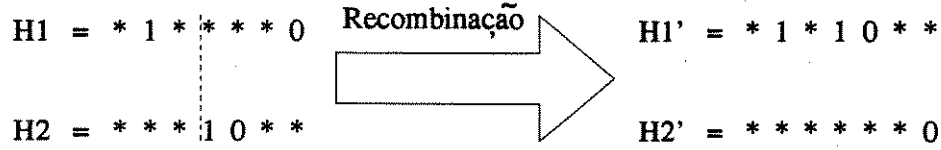


Figura 0.1: O efeito do operador recombinação em classes de cadeias.

- p_d : probabilidade da classe H ser destruída.
- p_s : probabilidade da classe H sobreviver.

A probabilidade de destruição de uma classe é igual ao produto das probabilidades de ocorrência de recombinação e de seleção do ponto de cruzamento entre as duas posições fixas da classe mais distantes entre si. Desta forma, tem-se:

$$p_d = p_r \frac{\delta(H)}{k-1} \quad (0.4)$$

$$p_s = 1 - p_d = 1 - p_r \frac{\delta(H)}{k-1} \quad (0.5)$$

Portanto, a influência do operador *recombinação* no algoritmo genético (considerando também a influência do operador *reprodução*) é dada por:

$$\hat{m}(H, t+1) = m(H, t) \frac{f(H)}{\bar{f}} \left[1 - p_r \frac{\delta(H)}{k-1} \right] \quad (0.6)$$

Efeito do Operador Mutação

As posições fixas de uma classe a caracterizam completamente. Deste modo, para que uma classe H sobreviva, é necessário que todas as suas posições fixas sobrevivam. A ocorrência de mutação em cada uma das posições fixas da classe são eventos independentes. Seja p_{sm} a probabilidade de sobrevivência da classe H sob efeito de *mutação*. Logo:

$$p_{sm} = (1 - p_m)^{o(H)} \quad (0.7)$$

Devido ao fato da probabilidade de *mutação* ser naturalmente muito pequena, pode-se reescrever a equação (0.7), de forma aproximada, como

$$p_{sm} = 1 - o(H) p_m \quad (0.8)$$

Portanto, a influência do operador *mutação* no algoritmo genético (considerando também a influência do operador *reprodução*) é dada por:

$$\hat{m}(H, t + 1) = m(H, t) \frac{f(H)}{\bar{f}} [1 - o(H) p_m] \quad (0.9)$$

Efeito dos Três Operadores Genéticos

A influência dos três operadores genéticos é dada pela combinação das equações (0.3), (0.6) e (0.9). Desta forma, tem-se que:

$$\hat{m}(H, t + 1) = m(H, t) \frac{f(H)}{\bar{f}} [1 - p_r \frac{\delta(H)}{k - 1}] [1 - o(H) p_m] \quad (0.10)$$

Desenvolvendo a equação (0.10) e desprezando o termo com o produto das probabilidades de *recombinação* e *mutação*, chega-se a

$$\hat{m}(H, t + 1) = m(H, t) \frac{f(H)}{\bar{f}} [1 - p_r \frac{\delta(H)}{k - 1} - o(H) p_m] \quad (0.11)$$

A equação (0.11) é a forma final do *Teorema Fundamental dos Algoritmos Genéticos*.

Bibliografia

- [Cernik 91] Cernik, R. J., D. L. Snowden, F. H. Yocum, S. R. Hendon (1991). *Multivariable DMC Control Applied on a Crude Acetone Column*. ISA International Conference, Anaheim, California.
- [Clarke 87] Clarke, D. W., C. Mohtadi, P. S. Tuffs (1987). *Generalized Predictive Control - Part I. The Basic Algorithm*. Automatica, vol. 23, no. 2, pp. 137-148.
- [Mohtadi 87] Clarke, D. W., C. Mohtadi, P. S. Tuffs (1987). *Generalized Predictive Control - Part II. Extensions and Interpretations*. Automatica, vol. 23, no. 2, pp. 149-160.
- [Clarke 89] Clarke, D. W., C. Mohtadi (1989). *Properties of Generalized Predictive Control*. Automatica, vol. 25, no. 6, pp. 859-875.
- [Croce 93] Croce, F. D., R. Tadei, G. Volta (1993). *A Genetic Algorithm for the Job Shop Problem*. Computers & Operations Research, Special Issue: Application of Genetic Algorithms in Business and Computer Science (to appear).
- [Cutler 79] Cutler, C. R., B. L. Ramaker (1979). *Dynamic Matrix Control - A Computer Control Algorithm*. AIChE National Meeting, Houston, Texas.
- [Cutler 82] Cutler, C. R. (1982). *Dynamic Matrix Control of Imbalanced Systems*. ISA Transactions, vol. 21, no. 1, pp. 1-6.

- [Cutler 83] Cutler, C. R., J. J. Haydel, A. M. Morshedi (1983). *An Industrial Perspective on Advanced Control*. AIChE National Meeting, Washington, D. C..
- [Cutler 88] Cutler, C. R., S. G. Finlayson (1988). *Multivariable Control of a C3C4 Splitter Column*. AIChE National Meeting, New Orleans, Louisiana.
- [Finlayson 88] Cutler, C. R., S. G. Finlayson (1988). *Design Considerations for a Hydrocracker Preflash Column Multivariable Constraint Controller*. IFAC Conference, Atlanta, Georgia.
- [Hawkins 88] Cutler, C. R., R. B. Hawkins (1988). *Application of a Large Predictive Multivariable Controller to a Hydrocracker Second Stage Reactor*. American Control Conference, Atlanta, Georgia.
- [De Jong 80] De Jong, K. (1980). *Adaptive System Design: A Genetic Approach*. IEEE Transactions on Systems, Man, and Cybernetics, vol. 10, no. 9, pp. 566-574.
- [Fleming 93] Fleming, P. J., C. M. Fonseca (1993). *Genetic Algorithms in Control Systems Engineering*. Proceedings of the IFAC 1993 World Congress, Sidney, Australia.
- [Fonseca 93] Fleming, P. J., C. M. Fonseca (1993). *Genetic Algorithms in Control Systems Engineering: A Brief Introduction*. IEE Colloquium on Genetic Algorithms for Control Systems Engineering, pp. 1/3-1/5.
- [García 88] García, C. E., D. M. Prett, B. L. Ramaker (1988). *Fundamental Process Control*. The Second Shell Process Control Workshop, pp. 401-440, Butterworths, Stoneham, MA.
- [García 89] García, C. E., D. M. Prett, M. Morari (1989). *Model Predictive Control: Theory and Practice - a Survey*. Automatica, vol. 25, no. 3, pp. 335-348.

- [Georgiou 88] **Georgiou, A., C. Georgakis, W. L. Luyben** (1988). *Nonlinear Dynamic Matrix Control for High-Purity Distillation Columns*. AIChE Journal, vol. 34, no. 8, pp. 1287-1298.
- [Goldberg 85] **Goldberg, D. E.** (1985). *Dynamic System Control Using Rule Learning and Genetic Algorithms*. Proceedings of the Ninth International Joint Conference on Artificial Intelligence, pp. 588-592.
- [Goldberg 89] **Goldberg, D. E.** (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley Publishing Company Inc.
- [Grefenstette 86] **Grefenstette, J. J.** (1986). *Optimization of Control Parameters for Genetic Algorithms*. IEEE Transactions on Systems, Man, and Cybernetics, vol. 16, no. 1, pp. 122-128.
- [Houk 91] **Houk, B. G., D. L. Snowden, T. E. Stevenson** (1991). *Improved Control of Ethylene Recovery Train using Dynamic Matrix Control*. ISA International Conference, Anaheim, California.
- [Hunt 92] **Hunt, K. J.** (1992). *Optimal Controller Synthesis: A Genetic Algorithm Solution*. IEE Colloquium on Genetic Algorithms for Control Systems Engineering, pp. 1/1-1/6.
- [Hunt1 92] **Hunt, K. J.** (1992). *Polynomial LQG and H1 Controller Synthesis: A Genetic Algorithm Solution*. Proceedings of the IEEE Conference on Decision and Control, Tucson, USA.
- [Karr 91] **Karr, C.** (1991). *GA's for Fuzzy Controllers*. AI Expert, pp. 26-33.
- [Karr 92] **Karr, C. L.** (1992). *An Adaptive System for Process Control Using Genetic Algorithms*. International Symposium on Artificial Intelligence in Real-Time Control, pp. 585-590.

- [Keane 93] **Keane, M. A., J. R. Koza, J. P. Rice** (1993). *Finding an Impulse Response Function using Genetic Programming*. Proceedings of The American Control Conference, pp. 2345-2350, San Francisco, California.
- [Kristinsson 88] **Kristinsson, K., G. A. Dumont** (1988). *Genetic Algorithms in System Identification*. Proceedings of the IEEE International Symposium on Intelligent Control, pp. 597-602.
- [Kristinsson 92] **Kristinsson, K., G. A. Dumont** (1992). *System Identification and Control Using Genetic Algorithms*. IEEE Transactions on Systems, Man, and Cybernetics, vol. 22, no. 5, pp. 1033-1046.
- [Levien 87] **Levien, K. L., M. Morari** (1987). *Internal Model Control of Coupled Distillation Columns*. AIChE Journal, vol. 33, no. 1, pp. 83-98.
- [Linkens 92] **Linkens, D. A., H. O. Nyongesa** (1992). *A Real-Time Genetic Algorithm for Fuzzy Control*. IEE Colloquium on Genetic Algorithms for Control Systems Engineering, pp. 9/1-9/4.
- [Martin 81] **Martin, G. D.** (1981). *Long-Range Predictive Control*. AIChE Journal, vol. 27, no. 5, pp. 748-753.
- [Mehra 81] **Mehra, R. K., R. Rouhani, J. Eterno, J. Richalet, A. Rault** (1981). *Model Algorithmic Control (MAC): Review and Recent Developments*. Proceedings of The Engineering Foundation Conference, Georgia.
- [Michalewicz 90] **Michalewicz, Z., J. B. Krawczyk, M. Kazemi, C. Z. Janikow** (1990). *Genetic Algorithms and Optimal Control Problems*. Proceedings of the 29th Conference on Decision and Control, Honolulu, Hawaii.
- [Morari 89] **Morari, M., E. Zafiriou** (1989). *Robust Process Control*. Prentice-Hall Inc.

- [O'Connor 91] O'Connor, D. L., K. Grimstad, J. McKay (1991). *Application of a Single Multivariable Controller to Two Hydrocracker Distillation Columns in Series*. ISA International Conference, Anaheim, California.
- [Oliveira 91] Oliveira, P., J. Sequeira, J. Sentieiro (1991). *Selection of Controller Parameters Using Genetic Algorithms*. Engineering Systems with Intelligence, pp. 431-438, Kluwer Academic, The Netherlands.
- [Park 94] Park, D., A. Kandel, G. Langholz (1994). *Genetic-Based New Fuzzy Reasoning Models with Application to Fuzzy Control*. IEEE Transactions on Systems, Man, and Cybernetics, vol. 24, no. 1, pp. 39-47.
- [Porter 92] Porter, B., A. H. Jones (1992). *Genetic Tuning of PID Controllers*. Electronics Letters, vol. 28, no. 9, pp. 843-844.
- [Potts 94] Potts, J. C., T. D. Giddens, S. B. Yadav (1994). *The Development and Evaluation of an Improved Genetic Algorithm based on Migration and Artificial Selection*. IEEE Transactions on Systems, Man, and Cybernetics, vol. 24, no. 1, pp. 73-86.
- [Prett 88] Prett, D. M., C. E. García (1988). *Fundamental Process Control*. Butterworths, Stoneham, MA.
- [Prett 90] Prett, D. M., C. E. García, B. L. Ramaker (1990). *The Second Shell Process Control Workshop*. Butterworths, Stoneham, MA.
- [Renders 92] Renders, J. M., J. P. Nordvik, H. Bersini (1992). *Genetic Algorithms for Process Control: A Survey*. International Symposium on Artificial Intelligence in Real-Time Control, pp. 579-584.
- [Richalet 78] Richalet, J., A. Rault, J. L. Testud, J. Papon (1978). *Model Predictive Heuristic Control: Applications to Industrial Processes*. Automatica, vol. 14, pp. 413-428.

- [Thrift 91] **Thrift, P.** (1991). *Fuzzy Logic Synthesis with Genetic Algorithms*. Genetic Algorithms: Proceedings of the Fourth International Conference, pp. 509-513.
- [Yang 94] **Yang, Z., T. Hachino, T. Tsuji, S. Sagara** (1994). *Identification of Parameters and Time Delays of Continuous Systems Using The Genetic Algorithm*. SYSID'94 Copenhagen Denmark, vol. 3, pp. 657-662.