PROJETO E AJUSTE SIMULTÂNEO DE CONTROLADORES EM CASCATA E COMPENSADORES DE REALIMENTAÇÃO PELO MÉTODO DAS INEQUAÇÕES

Dissertação submetida à
UNIVERSIDADE FEDERAL DE ITAJUBÁ
como requisito parcial para obtenção do grau de
Mestre em Ciências em Engenharia Elétrica

por

CARLOS BENJAMIN VARAS CORDERO

Orientador: Prof. Dr. Carlos Alberto Dias Coelho (UNIFEI)
Co-Orientador: Prof. Dr. Ângelo José Junqueira Rezek (UNIFEI)

UNIVERSIDADE FEDERAL DE ITAJUBÁ
ITAJUBÁ - MG - BRASIL
Fevereiro de 2004

À minha esposa Flávia e aos meus filhos Gabriel, Júlia e Bruna.

AGRADECIMENTOS

À UNIVERSIDADE FEDERAL DE ITAJUBÁ - UNIFEI, pela oportunidade de cursar o mestrado.

À FUNDAÇÃO DE AMPARO À PESQUISA DO ESTADO DE MINAS GERAIS – FAPEMIG, pela concessão de bolsa de estudos durante parte do período de elaboração deste trabalho.

Aos professores Carlos Alberto Dias Coelho e Ângelo José Junqueira Rezek por sua competência e inestimável orientação no desenvolvimento deste trabalho.

Ao professor Carlos Augusto Ayres, Coordenador dos Cursos de Pós-Graduação em Engenharia Elétrica, pela cooperação desinteressada.

Aos professores dos Cursos de Pós-Graduação em Engenharia Elétrica, pelos conhecimentos ministrados.

Aos amigos e colegas Ramiro Ramírez, Ricardo Carrasco e Manuel Rendón pela ajuda e incentivo.

Aos meus pais Luis Enrique e Teresa, pelo constante incentivo, e principalmente pelo exemplo de vida.

E acima de tudo a Deus.

PROJETO E AJUSTE SIMULTÂNEO DE CONTROLADORES EM CASCATA E COMPENSADORES DE REALIMENTAÇÃO PELO MÉTODO DAS INEQUAÇÕES

RESUMO

Neste trabalho é apresentada uma nova estrutura de projeto de sistemas de controle univariáveis pelo Método das Inequações (*MI*), baseada em uma configuração padrão mais completa para os sistemas e com mais flexibilidade para definição dos parâmetros a serem ajustados. O MI é um procedimento de projeto assistido por computador, onde através de algoritmos iterativos de busca tenta-se encontrar uma combinação de valores dos parâmetros ajustáveis do sistema, para a qual consegue-se cumprir todas as restrições e especificações de projeto, as quais são estabelecidas através de um conjunto de inequações.

O MI é aplicado ao projeto e ajuste simultâneo de controladores em cascata e compensadores de realimentação para sistemas de controle lineares invariantes de uma entrada e uma saída. A estratégia de projeto inclui a opção de otimização do sistema através da minimização da integral do erro absoluto multiplicado pelo tempo (*IEAT*).

A implementação computacional do MI desenvolvida neste trabalho possui uma interface gráfica com o usuário que facilita a entrada de dados e a visualização de resultados, que torna o projeto muito mais rápido. A sua utilização e a eficácia do MI são ilustradas através de vários exemplos de aplicação.

SIMULTANEOUS DESIGN AND TUNING OF CASCADE CONTROLLERS AND FEEDBACK COMPENSATORS BY THE METHOD OF INEQUALITIES

ABSTRACT

In this work a new framework for design of single input–single output (*SISO*) control systems by the Method of Inequalities (*MI*) is presented, based on a more complete standard system configuration and with more flexibility for definition of the parameters to be set. The MI is a computer-aided design (*CAD*) procedure which, by means of iterative search algorithms, tries to find a values combination of the adjustable parameters of the system so as to satisfy all the design constraints and specifications, which are expressed by a set of inequalities.

The MI is applied to the simultaneous design and tuning of cascade controllers and feedback compensators for single-input single-output linear time invariant control systems. The design strategy includes the option of system optimization trough minimization of the integral of absolute error multiplied by time (*IAET*).

The computational implementation of the MI developed in this work includes a graphical user interface that eases data input and output visualization, thus making the design a lot faster. Its use and the MI effectiveness are illustrated by some examples of application.

Ministério da Educação
UNIVERSIDADE FEDERAL DE ITAJUBÁ
Criada pela Lei nº 10.435, de 24 de abril de 2002

A N E X O I

PRONUNCIAMENTO DA BANCA EXAMINADORA

A Banca Examinadora, abaixo assinada, nomeada pela Portaria nº 033 de 13 de fevereiro de 2004, considerando o resultado do Julgamento da Prova de Defesa Pública da Dissertação de Mestrado intitulada: **"Projeto e Ajuste Simultâneo de Controladores em Cascata e Compensadores de Realimentação pelo Método das Inequações"** apresenta pronunciamento no sentido de que o Coordenador dos Cursos de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Itajubá solicite ao DRA (Departamento de Registro Acadêmico) a expedição do título de **Mestre em Ciências em Engenharia Elétrica**, na **Área de Concentração: Automação e Sistemas Elétricos Industriais,** satisfeitas as demais exigências regimentais, a **Carlos Benjamin Varas Cordero.**

Itajubá, 16 de fevereiro de 2004.

Prof. Dr. Oscar Armando Maldonado Astorga
1º Examinador - FEG / UNESP

Prof. Dr. Antonio Carlos Zambroni de Souza
2º Examinador - UNIFEI

Prof. Dr. Ângelo José Junqueira Rezek
3º Examinador – (Co-Orientador) - UNIFEI

Prof. Dr. Carlos Alberto Dias Coelho
4º Examinador – (Orientador) - UNIFEI

Pró-Diretoria de Pesquisa e Pós-Graduação - Av BPS, 1303 - Caixa Postal 50 - 37500-903 - ITAJUBÁ/MG - BRASIL
Tel.: (35) 3629-1118 - 3629-1121 - Fax (35) 3629-1120

Ministério da Educação
UNIVERSIDADE FEDERAL DE ITAJUBÁ
Criada pela Lei nº 10.435, de 24 de abril de 2002

A N E X O II

FOLHA DE JULGAMENTO DA BANCA EXAMINADORA

Título da Dissertação: **"Projeto e Ajuste Simultâneo de Controladores em Cascata e Compensadores de Realimentação pelo Método das Inequações "**

Autor: **Carlos Benjamin Varas Cordero**

JULGAMENTO

Examinadores	Conceito A = Aprovado R = Reprovado	Rubrica
1º	A	
2º	A	
3º	A	
4º	A	

Observações:

1) O Trabalho será considerado Aprovado se todos os Examinadores atribuírem conceito A.
2) O Trabalho será considerado Reprovado se forem atribuídos pelos menos 2 conceitos R.
3) O Trabalho será considerado Insuficiente (I) se for atribuído pelo menos um conceito R. Neste caso o candidato dever apresentar novo trabalho. A banca deve definir como avaliar a nova versão da Dissertação.

Resultado Final: Conceito: A, ou seja, aprovado

Observações: _____

Itajubá, 16 de fevereiro de 2004.

Prof. Dr. Oscar Armando Maldonado Astorga
1º Examinador - FEGV UNESP

Prof. Dr. Antonio Carlos Zambroni de Souza
2º Examinador - UNIFEI

Prof. Dr. Ângelo José Junqueira Rezek
3º Examinador – (Co-Orientador) - UNIFEI

Prof. Dr. Carlos Alberto Dias Coelho
4º Examinador – (Orientador) - UNIFEI

Pró-Diretoria de Pesquisa e Pós-Graduação - Av . BPS, 1303 - Caixa Postal 50 - 37500-903 - ITAJUBÁ/MG – BRASIL
Tel.: (35) 3629-1118 - 3629-1121 - Fax (35) 3629-1120

SUMÁRIO

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

LISTA DE SIGLAS

BIBO	Entrada Limitada–Saída Limitada (*Bounded Input–Bounded Output*)
CAD	Projeto Assistido por Computador (*Computer Aided Design*)
CRD	Características da Resposta ao Degrau
DB	Diagrama de Blocos
ED	Equação Diferencial
FC	Função Custo
FSC	Funcionais do Sistema de Controle
FT	Função de Transferência
GUI	Interface Gráfica com o Usuário (*Graphics User Interface*)
IAET	*Integral of Absolute Error Multiplied by Time*
ID	Índice de Desempenho
IEA	Integral do Erro Absoluto
IEAT	Integral do Erro Absoluto Multiplicado pelo Tempo
IEQ	Integral do Erro Quadrático
MBP	*Moving Boundaries Process*
MI	Método das Inequações (*Method of Inequalities*)
PCM	Processo dos Contornos Móveis
SISO	Univariável (*Single Input–Single Output*)
SLI	Sistema Linear Invariante
TL	Transformada de Laplace

CAPÍTULO 1

PROJETO DE CONTROLADORES PELO MÉTODO DAS INEQUAÇÕES

1.1 Introdução

O objetivo do presente trabalho é o de desenvolver uma ferramenta computacional útil para o projeto de sistemas de controle, onde os complexos cálculos matemáticos e as necessárias repetições de tentativa e erro sejam feitas pelo computador, liberando ao projetista para se concentrar em problemas que requerem a sua análise, tais como escolher o tipo apropriado de controlador para cada caso, e definir um conjunto de restrições de operação e especificações de desempenho que levem à consecução de um sistema de controle adequado às necessidades do projeto.

Utiliza-se o Método das Inequações (MI) para o projeto de sistemas de controle univariáveis (*SISO*) e invariantes no tempo (SLI), que possuam controladores em cascata e/ou compensadores de realimentação. O MI é uma técnica que propicia uma formulação realista do problema, permitindo ao projetista expressar diretamente, em forma de inequações, as restrições de operação e índices de desempenho do sistema.

No projeto de sistemas de controle utilizando técnicas de controle clássico, tais como a do lugar das raízes ou de resposta em freqüência, são estabelecidas inicialmente as especificações de desempenho que o sistema deve satisfazer e, partindo dessas especificações, são calculados os parâmetros dos controladores que deveriam satisfazê-las. Por causa de esses métodos utilizarem simplificações e aproximações, às vezes são necessárias várias seqüências de tentativa e erro até que finalmente sejam encontrados os valores de parâmetros que atendam às especificações.

O problema de projeto vai ficando mais complexo à medida que aumenta o número de especificações, o número de parâmetros variáveis de controladores, e com a utilização simultânea de controladores em cascata e em realimentação, até se chegar a um ponto

onde não é mais viável a utilização dos métodos clássicos de projeto. Por causa dos motivos expostos, é plenamente justificável a utilização de métodos computacionais para o projeto de sistemas de controle.

O MI teve a sua concepção e o seu desenvolvimento inicial motivados pelos avanços na teoria de sistemas dinâmicos e do aumento contínuo da capacidade de armazenamento e de processamento de dados nos computadores. A aplicabilidade do MI gira principalmente em torno do uso eficaz de recursos computacionais para a resolução das inequações, através de algoritmos numéricos iterativos.

O MI foi desenvolvido por Zakian e Al-Naib, e na sua primeira publicação [26], além de ser apresentada a sua filosofia e suas premissas básicas, também foi apresentado o Processo dos Contornos Móveis (PCM), um algoritmo extremamente simples e eficaz para a solução de sistemas de inequações. Naquela primeira publicação, a filosofia de projeto do MI foi aplicada ao projeto de sistemas de controle univariáveis e multivariáveis, considerando como índices de desempenho as características da resposta ao degrau.

Em função dos bons resultados obtidos com o MI, foi proposta uma Nova Formulação para o Método das Inequações–NFMI [27], onde é estabelecido que os índices de desempenho devem contemplar o conjunto de todas as entradas possíveis a que os sistemas estão sujeitos em sua operação, aí incluídos os sinais de referência e os distúrbios, e que, na formulação das inequações, também devem ser incluídas limitações nos sinais de controle, de modo a garantir a validade do modelo do sistema controlado e a limitar o esforço de controle.

Os comandos e rotinas computacionais desenvolvidos ao longo deste trabalho estão direcionados para o projeto de controladores em cascata e compensadores de realimentação de sistemas univariáveis, lineares e invariantes no tempo.

Para a execução de um projeto pelo MI, devem-se definir as especificações de desempenho, as restrições de operação, a estrutura de controle, o que define um conjunto de parâmetros a serem ajustados. O ajuste desses parâmetros é feito

automaticamente através do PCM, que determina conjuntos de valores dos parâmetros para os quais o sistema de inequações é satisfeito.

Esta nova implementação computacional do MI se vale das ferramentas de programação do aplicativo MATLAB, o qual permite criar programas complexos (*toolboxes*), através da utilização de sub-rotinas aplicadas aos sistemas de controle, desenvolvidas por múltiplos grupos de desenvolvimento ao redor do mundo. A *toolbox* desenvolvida utiliza uma interface gráfica com o usuário, que converte o trabalho de projetar controladores em uma tarefa agradável e estimulante.

1.2 Descrição do Texto

No capítulo dois são apresentados os fundamentos teóricos dos sistemas de controle; no capítulo três é revista a idéia central do MI e do PCM; no capítulo quatro é explicada a aplicação do MI ao projeto de sistemas de controle; no capítulo cinco é apresentada a implementação computacional do método utilizando o MATLAB; e no capítulo seis são apresentados vários exemplos de aplicação do MI ao projeto de sistemas de controle, com controladores em cascata e compensadores de realimentação. As conclusões são apresentadas no capítulo sete.

CAPÍTULO 2

CONCEITOS DE ANÁLISE E PROJETO DE SISTEMAS DE CONTROLE

Neste capítulo são revisados os conceitos de controle utilizados para a execução do projeto, com base em trabalhos de Coelho [6], Ogata [20] e D'Azzo e Houpis [13].

2.1 Sistemas Dinâmicos

Um *sistema dinâmico* é qualquer conjunto de elementos interagentes, nos quais existam relações de causa e efeito que dependem do tempo. As grandezas que caracterizam o estado do sistema são denominadas *variáveis dinâmicas*. A análise de tais sistemas consiste em determinar a resposta do sistema a uma *excitação*. Entende-se por excitação qualquer variável vinda de uma fonte externa, a qual, sendo aplicada ao sistema, provoca variações em uma ou mais de suas variáveis dinâmicas. Um sistema ou subsistema pode ser representado através de um bloco como mostra a Figura 2.1. A excitação é representada pela seta que aponta para dentro do bloco e a resposta pela seta que aponta para fora do bloco.

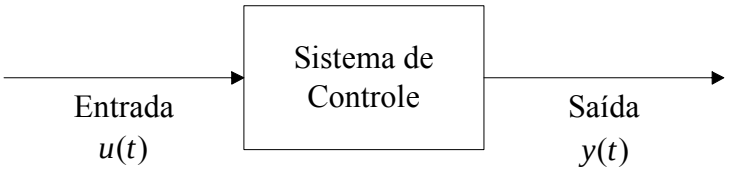


Figura 2.1: Representação de bloco de sistema de controle

Entre os muitos tipos de sistemas dinâmicos que podem ser encontrados, neste trabalho são considerados sistemas com as seguintes características:

- parâmetros concentrados,
- lineares,
- contínuos,
- invariantes,

- univariáveis (SISO),
- excitados,
- com condições iniciais nulas.

Para permitir a determinação das características dinâmicas e a saída do sistema, é preciso modelá-lo matematicamente.

Os modelos matemáticos utilizados neste trabalho são as funções de transferência (FT's) e os diagramas de blocos (DB's).

As funções de transferência são utilizadas para representar sistemas ou subsistemas lineares invariantes, relacionando diretamente a saída com a entrada do sistema. É um tipo de modelo bastante importante para análise de estabilidade e de outras características dinâmicas, além de ser de fundamental importância para representar elementos e subsistemas simples, na construção de modelos mais complexos.

Os diagramas de blocos tem a finalidade de ilustrar a interconexão de elementos e de subsistemas para a formação do sistema; cada elemento é representado por um bloco com apenas uma entrada e uma saída. A saída de um bloco pode ser entrada para outro ou outros. Assim, os blocos são interligados através de linhas orientadas, mostrando claramente as relações de causa e efeito entre os vários componentes do sistema.

Na Figura 2.2. mostra-se um exemplo da utilização conjunta de FT's e DB's.

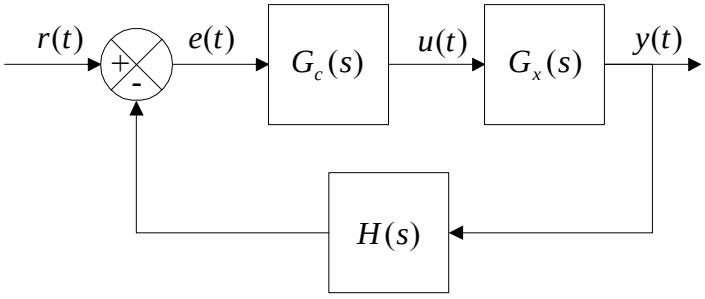


Figura 2.2: Diagrama de blocos com funções de transferência

2.2 Transformada de Laplace

A transformada de Laplace (TL) é uma ferramenta matemática das mais eficazes na análise, ajuste e controle de sistemas lineares.

As transformadas de Laplace são definidas no domínio de uma variável complexa *s*, dada por:

$$s = \sigma + j\omega \tag{2.1}$$

A transformada de Laplace de uma função $f(t)$ é definida como segue:

$$F(s) = \mathcal{L}[f(t)] = \int_{0-}^{\infty} f(t)e^{-st}dt \tag{2.2}$$

A transformada de Laplace possui a propriedade da linearidade a qual abrange as propriedades de homogeneidade e de superposição. Esta propriedade é expressa por:

$$\mathcal{L}[a\ f_1(t) + b\ f_2(t)] = a\,F_1(s) + b\,F_2(s) \tag{2.3}$$

O degrau unitário é a função básica da família das funções singulares. Sua definição é a seguinte:

$$u_{-1}(t) = \begin{cases} 0, \text{ para } t < 0 \\ 1, \text{ para } t > 0 \end{cases} \tag{2.4}$$

A transformada de Laplace do degrau é:

$$\mathcal{L}[u_{-1}(t)] = \frac{1}{s} \tag{2.5}$$

Um degrau com amplitude R_0 vai ser utilizado neste trabalho como sinal de excitação, a fim de analisar o desempenho dos sistemas dinâmicos. Integrais sucessivas do degrau produzem as correspondentes funções singulares, tais como rampa, parábola e assim por diante. As funções singulares dadas pelas integrais sucessivas do degrau são utilizadas

neste trabalho unicamente para o cálculo do erro de regime permanente de sistemas tipo um e superiores, tal como é explicado na seção 2.5.3.

2.2.1 Transformadas de Laplace racionais

Uma transformada de Laplace racional é a relação de dois polinômios dada pela forma:

$$F(s) = \frac{P(s)}{Q(s)} = \frac{b_w s^w + b_{w-1} s^{w-1} + \ ... + b_2 s^2 + b_1 s + b_0}{a_n s^n + a_{n-1} s^{n-1} + ... + a_2 s^2 + a_1 s + a_0} \quad \textbf{(2.6)}$$

Uma característica importante das funções racionais é a diferença entre os graus dos polinômios do numerador e do denominador.

Sob este ponto de vista, as funções racionais podem ser classificadas da seguinte forma:

Se $w < n$, $F(s)$ é uma função estritamente própria.
Se $w \leq n$, $F(s)$ é uma função própria.
Se $w > n$, $F(s)$ é uma função imprópria.

Zeros de $F(s)$ são as raízes de $P(s) = 0$
Pólos de $F(s)$ são as raízes de $Q(s) = 0$

Se os zeros e pólos da função de transferência $F(s)$ forem conhecidos, então a função poderá ser escrita na forma fatorada:

$$F(s) = \frac{P(s)}{Q(s)} = \frac{b_w (s - z_1)(s - z_2) ... (s - z_w)}{a_n (s - s_1)(s - s_2) ... (s - s_n)} \quad \textbf{(2.7)}$$

O teorema do valor final para transformadas de Laplace racionais é enunciado da seguinte forma:

Seja $f(t)$ uma função do tempo, com uma transformada de Laplace racional $F(s)$ dada por:

$$\mathcal{L}[f(t)] = F(s) = \frac{P(s)}{Q(s)} = \frac{b_w s^w + b_{w-1} s^{w-1} + \ ... + b_2 s^2 + b_1 s + b_0}{a_n s^n + a_{n-1} s^{n-1} + ... + a_2 s^2 + a_1 s + a_0} = \\ = \frac{b_w (s - z_1)(s - z_2) ... (s - z_w)}{a_n (s - s_1)(s - s_2) ... (s - s_n)} \quad \textbf{(2.8)}$$

Então a existência e o cálculo do valor final podem ser estabelecidos como segue:

- Se todos os pólos de $F(s)$ tiverem parte real negativa, então o valor final de $f(t)$ é nulo.

- Se $F(s)$ tiver um único pólo nulo, e todos os demais pólos tiverem parte real negativa, então o valor final de $f(t)$ é constante e não nulo, sendo dado por:

$$f(\infty) = \lim_{t \to \infty} f(t) = \lim_{s \to 0} s F(s) = \frac{b_0}{a_1} \quad \textbf{(2.9)}$$

- Nos demais casos, $f(t)$ não possui valor final constante.

2.2.2 Teorema do valor final

O teorema do valor final é importante no cálculo do erro de regime permanente dos sistemas, e também no cálculo das características de resposta ao degrau, pois serve para calcular o valor em regime permanente (quando t tende a ∞) de uma variável dinâmica a partir de sua transformada de Laplace.

O teorema diz o seguinte:

Se uma função $f(t)$ tiver valor final $f(\infty)$, e se sua transformada de Laplace $F(s)$ for conhecida, então esse valor final pode ser calculado através da seguinte expressão:

$$f(\infty) = \lim_{t \to \infty} f(t) = \lim_{s \to 0} s F(s) \quad \textbf{(2.10)}$$

Deve-se tomar cuidado de, antes da aplicação do teorema do valor final, verificar se a função tem valor final finito; no caso de funções racionais isto é verificável na própria transformada, como já visto na seção 2.2.1.

2.3 Funções de Transferência

Os diagramas de blocos (DB's) tem a finalidade de ilustrar a interligação de componentes e de subsistemas para a formação de um sistema. Quando a informação dada pelo diagrama de blocos é apenas qualitativa, diz-se que o DB é *esquemático*. Quando cada bloco contém informação quantitativa sobre a função que relaciona suas entradas e saídas, permitindo obter os valores das diversas variáveis do sistema, diz-se que o DB é *funcional*.

2.3.1 Sistemas lineares invariantes

Dentro do conjunto dos sistemas dinâmicos existe um importante subconjunto que é o dos sistemas lineares. Para um sistema dinâmico ser linear, deve obedecer os princípios da superposição e da homogeneidade. O princípio da superposição estabelece que a resposta produzida pela aplicação da soma de dois sinais de excitação diferentes é igual à soma das respostas individuais produzidas pelo sistema a cada uma das excitações aplicadas separadamente; o princípio da homogeneidade estabelece que quando a excitação é multiplicada por uma constante, a saída é multiplicada por essa mesma constante. Esses dois princípios podem ser unificados pela seguinte equação:

$$y(a\,u_1 + b\,u_2, t) = a\,y(u_1, t) + b\,y(u_2, t) \qquad \textbf{(2.11)}$$

Os sistemas lineares ainda são classificados em duas sub-categorias: os invariantes no tempo e os variantes no tempo. Invariantes no tempo são aqueles cujos parâmetros são mantidos constantes durante a operação. Isto leva a sistemas onde todas as variáveis, bem como suas derivadas e integrais, só podem ser multiplicadas por constantes. Qualquer sistema cujas equações tenham um ou mais termos com coeficientes variáveis, isto é, multiplicados por funções do tempo, é variante.

O comportamento de qualquer sistema dinâmico é dado pela sua equação diferencial (ED), tendo como variável independente o tempo *t*. A derivada de ordem *k* de uma variável *y* em relação a *t* pode ser denotada da seguinte forma:

$$\frac{d^k y}{dt^k} = \underbrace{p(\ldots(p(py)))}_{(k\ vezes)} = p^k y \tag{2.12}$$

Para sistemas lineares invariantes onde a entrada é $u(t)$ e a saída $y(t)$, a ED terá a seguinte forma geral:

$$\begin{aligned} &a_n p^n y + a_{n-1} p^{n-1} y + \ldots + a_1 py + a_0 y = \\ &= b_w p^w u + b_{w-1} p^{w-1} u + \ldots + b_1 pu + b_0 u \end{aligned} \tag{2.13}$$

A função de transferência (FT) é uma forma alternativa de se representar um sistema, que seja invariante no tempo, e que tenha apenas uma entrada, ou seja, uma única excitação. A função de transferência é o quociente da divisão da saída pela entrada, tratando-se o operador *p* como uma variável comum que pode ser colocada em evidência. A função de transferência obtida a partir da Eq. (2.13) é:

$$G(p) = \frac{y(t)}{u(t)} = \frac{b_w p^w + b_{w-1} p^{w-1} + \ldots + b_1 p + b_0}{a_n p^n + a_{n-1} p^{n-1} + \ldots + a_1 p + a_0} \tag{2.14}$$

2.3.2 Diagramas de blocos

Um sistema cuja FT seja $G(p)$ (Eq.(2.14)) é representado por um bloco funcional, na forma mostrada na Figura 2.3. Este bloco representa que a saída do sistema $y(t)$ é dada pelo produto da FT $G(p)$ pela entrada $u(t)$.

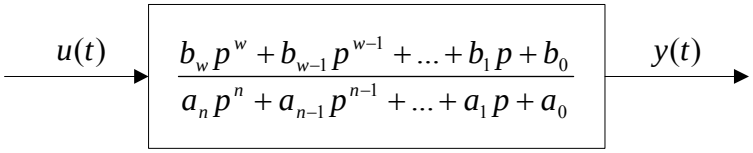


Figura 2.3: Função de transferência representada por bloco funcional

Em um sistema linear invariante (SLI) a resposta sempre pode ser expressa por uma soma de duas parcelas, como segue:

$$y(t) = y_L(t) + y_F(t) \tag{2.15}$$

$y_L(t)$ denota a resposta livre, e $y_F(t)$ a resposta forçada; $y(t)$ é a resposta total, ou, simplesmente, resposta. A resposta livre é a parcela da resposta causada pelas condições iniciais, e a resposta forçada é a parcela causada pela entrada (excitação) aplicada ao sistema.

Em um SLI, a obtenção e separação das respostas livre e forçada é feita com grande facilidade com o uso de transformadas de Laplace (TL's). A TL da Eq. (2.15) é

$$Y(s) = Y_L(s) + Y_F(s) \tag{2.16}$$

a resposta forçada é então dada por

$$Y_F(s) = G(s)U(s) \tag{2.17}$$

A Figura 2.4. ilustra os conceitos de resposta livre, forçada e total, e mais o de FT no domínio da transformada de Laplace.

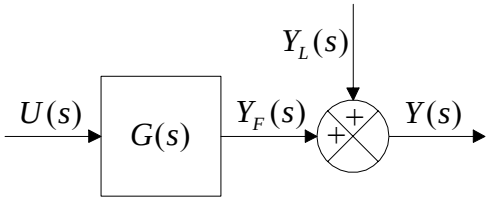


Figura 2.4: Decomposição de resposta total em resposta livre e forçada

Para sistemas racionais a FT no domínio da freqüência é obtida a partir de $G(p)$, com a simples substituição de p por s, e quando todas as condições iniciais dos sinais de entrada e saída do sistema são nulas, a resposta livre é igual a zero. Assim o sistema é representado pelo DB da Figura 2.5.

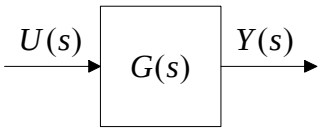


Figura 2.5: Representação em DB de FT no domínio *s*

Sendo $G(s)$ definido pela equação:

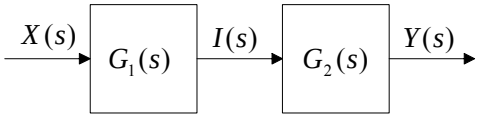
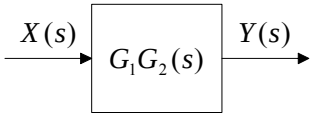
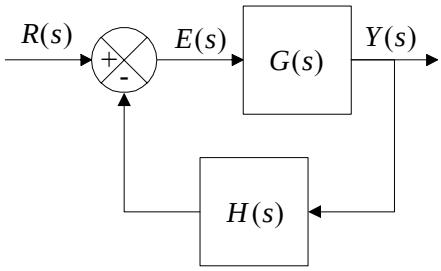
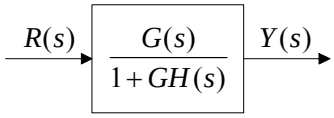
$$G(s) = \frac{Y(s)}{U(s)} = \frac{b_w s^w + b_{w-1} s^{w-1} + \ldots + b_1 s + b_0}{a_n s^n + a_{n-1} s^{n-1} + \ldots + a_1 s + a_0} \quad \textbf{(2.18)}$$

Para os sistemas lineares os DB's podem ser construídos com apenas três tipos de elementos, descritos a seguir:

- BLOCO: representa a operação de multiplicação da entrada pela FT; o produto resultante é a saída, tal como mostra a Figura 2.5.
- SOMADOR: o somador representa uma soma algébrica de variáveis. O símbolo que representa o somador é mostrado na Figura 2.4.
- PONTO DE IDENTIDADE: deve aparecer sempre que um sinal for entrada para mais de um somador e/ou bloco no diagrama. A Figura 2.2. mostra um ponto de identidade na linha do sinal *y*.

Todo DB pode ser modificado, e até reduzido a um único bloco equivalente, para o caso de sistemas univariáveis (*SISO*). O conjunto de regras para as modificações básicas dos DB's é conhecido como álgebra dos diagramas de blocos. Na Tabela 2.1. são apresentadas as regras básicas utilizadas neste trabalho.

Tabela 2.1: Regras básicas do álgebra dos diagramas de blocos

	Diagrama Original	Diagrama Equivalente
Combinação de Blocos em Cascata	 $I = G_1X$; $Y=G_2I$	 $Y=G_1G_2X$
Redução de Malha de Realimentação Negativa	 $Y = GE$; $E = R\text{-}HY$	 $Y = \frac{G}{1+GH}R$

2.4 Estabilidade de Sistemas de Controle

2.4.1 Definição

Diz-se que um sistema de controle é estável se, e somente se, forem satisfeitas simultaneamente as duas condições seguintes:

1. Para toda entrada limitada, a saída também deverá ser limitada.
2. Para toda entrada que tenda a zero em regime permanente, a saída também deve tender a zero em regime permanente.

Sistemas que não satisfaçam qualquer uma destas duas condições são considerados instáveis.

A primeira condição é conhecida como de entrada limitada–saída limitada, ou então pela sigla BIBO, que vem da expressão correspondente em inglês (*bounded input–bounded output*). Matematicamente a condição BIBO é expressa da seguinte forma:

$$\forall u(t) \text{ tal que } \left|u(t)\right| \le U_{\max} \ , \quad \exists \text{ um } Y_{\max} \text{ tal que } \left|y(t)\right| \le Y_{\max} \tag{2.19}$$

onde $U_{\max}$ e $Y_{\max}$ são constantes positivas.

A segunda condição é conhecida como condição de estabilidade assintótica, e matematicamente é expressa como segue:

$$\forall u(t) \text{ tal que } \lim_{t\to\infty} u(t) = 0 \Rightarrow \lim_{t\to\infty} y(t) = 0 \tag{2.20}$$

2.4.2 Estabilidade de sistemas racionais

A estabilidade de um sistema linear invariante (SLI) pode ser verificada através de sua função de transferência global (FT) sem necessidade de ensaios ou simulações.

Considera-se que as condições iniciais são nulas, o que implica em resposta livre nula, sendo a resposta do sistema dada apenas pela resposta forçada, isto é

$$Y(s) = G(s)U(s) \tag{2.21}$$

Para verificar a estabilidade em sistemas racionais, é necessário e suficiente que a função de transferência global seja própria, e todos os seus pólos tenham parte real negativa e não nula.

Matematicamente, isto pode ser expresso da seguinte maneira:

$$G(s) = \frac{Y(s)}{U(s)} = \frac{b_w s^w + b_{w-1} s^{w-1} + \ldots + b_1 s + b_0}{a_n s^n + a_{n-1} s^{n-1} + \ldots + a_1 s + a_0} \tag{2.22}$$

1. Função de transferência global própria:

$$w \le n \tag{2.23}$$

2. Todos os pólos com parte real negativa e não nula:

$$\sigma_s < 0 \tag{2.24}$$

onde σ_s é a *abscissa de estabilidade* do sistema, definida como segue:

$$\sigma_s = \max\left\{ \mathrm{Re}(s) \,\middle|\, \frac{1}{G(s)} = 0 \right\} \tag{2.25}$$

onde

$$\left\{ s \,\middle|\, \frac{1}{G(s)} = 0 \right\} = \left\{ s \,\middle|\, D_G(s) = 0 \right\} = \left\{ s_1, s_2, \ldots, s_n \right\} \tag{2.26}$$

é o conjunto dos pólos da função de transferência do sistema. Assim, $D_G(s)$ é o denominador de $G(s)$, com a seguinte forma fatorada:

$$D_G(s) = a_n (s - s_1)(s - s_2) \ldots (s - s_n) \tag{2.27}$$

No plano complexo *s*, a condição dada pela Ineq. (2.24) é visualizada na Figura 2.6.

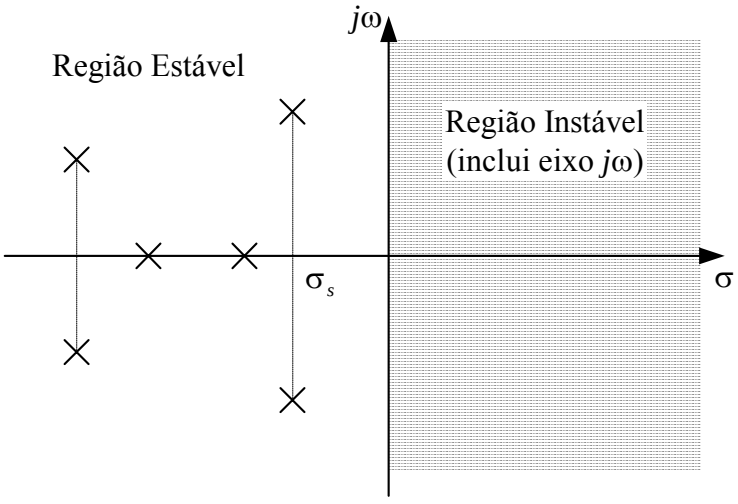


Figura 2.6: Visualização da região estável no plano complexo

2.5 Comportamento Dinâmico

Para analisar, ajustar e controlar sistemas dinâmicos é necessário haver uma base para sua identificação e especificação de desempenho. Isto é feito através da resposta do sistema a excitações padronizadas, sobre as quais são definidas as características de desempenho. O ajuste e o projeto de controladores tem por objetivo fazer com que essas características de desempenho cumpram as especificações.

As excitações padronizadas mais freqüentemente usadas na identificação e análise de sistemas dinâmicos são:

- a família das funções singulares (impulso, degrau, rampa, parábola);
- a excitação polinomial, que é uma combinação linear do degrau e de suas integrais;
- as funções senoidais.

Neste trabalho o degrau foi escolhido como excitação para os sistemas dinâmicos, por se tratar da excitação mais utilizada para análise e especificação de desempenho transitório.

2.5.1 Resposta a degrau de sistemas estáveis

Na análise de sistemas dinâmicos estáveis, é freqüente que a avaliação do desempenho seja feita com base na resposta $y(t)$ a uma entrada degrau, considerando que antes da aplicação do degrau o sistema está em repouso, isto é, com a saída e todas as suas derivadas nulas.

Supondo que o degrau seja aplicado no instante t=0 e tenha amplitude R_0, então a entrada do sistema é:

$$r(t) = R_0 u_{-1}(t) \quad , \quad R_0 \neq 0 \qquad \textbf{(2.28)}$$

Se o sistema for estável, a resposta $y(t)$ será finita para qualquer t, e seu valor final será

$$y(\infty) = \lim_{t \to \infty} y(t) \tag{2.29}$$

Se o sistema for linear invariante e sua função de transferência for $W(s)$, de modo que

$$Y(s) = W(s)R(s) = \frac{W(s)R_0}{s} \tag{2.30}$$

esse valor final será dado por

$$y(\infty) = W(0)R_0 \tag{2.31}$$

$W(0)$ é o ganho estático do sistema.

A duração e a forma da resposta no período transitório dependem dos elementos dinâmicos que constituem o sistema. Em sistemas lineares, essas duração e forma dependem principalmente dos pólos da FT. Teoricamente, a resposta transitória

$$y_N(t) = y(t) - y(\infty) \tag{2.32}$$

só é anulada quando $t \to \infty$; na prática aceita-se que o transitório pode ser considerado nulo após decorrido um período igual a 5 vezes a maior constante de tempo do sistema, isto é,

$$y_N(t) = y(t) - y(\infty) \approx 0 \quad , \quad para\ t \geq 5T_{\max} \tag{2.33}$$

onde

$$T_{\max} = \frac{1}{|\sigma_s|} \tag{2.34}$$

$\sigma_s < 0$ é a abscissa de estabilidade do sistema.

2.5.2 Características da resposta a degrau

As especificações e avaliação de desempenho de sistemas dinâmicos neste trabalho vão se basear nas Características da Resposta a Degrau (CRD's), as quais são definidas com base em três considerações fundamentais:

(a) O sistema possui elementos armazenadores de energia, que o leva a ter uma certa inércia, e, por isto, a não responder instantaneamente a variações na entrada. Esta consideração implica resposta com valor inicial nulo:

$$y(0) = 0 \tag{2.35}$$

Em sistemas racionais, esta consideração implica FT $W(s)$ estritamente própria.

(b) O sistema é estável, portanto, a resposta a degrau é finita para todo t. Em um sistema linear com FT racional, esta consideração implica $W(s)$ com todos os pólos com parte real negativa.

(c) As CRD's são definidas em termos de relações entre determinados instantes de tempo e frações do valor final, que por isto deve ser diferente de zero:

$$y(\infty) = \lim_{t \to \infty} y(t) \neq 0 \tag{2.36}$$

Em sistemas lineares esta consideração implica $W(0) \neq 0$.

Esta última consideração significa que o valor final $y(\infty)$ tem o caráter de um objetivo a ser atingido, como acontece em sistemas de controle.

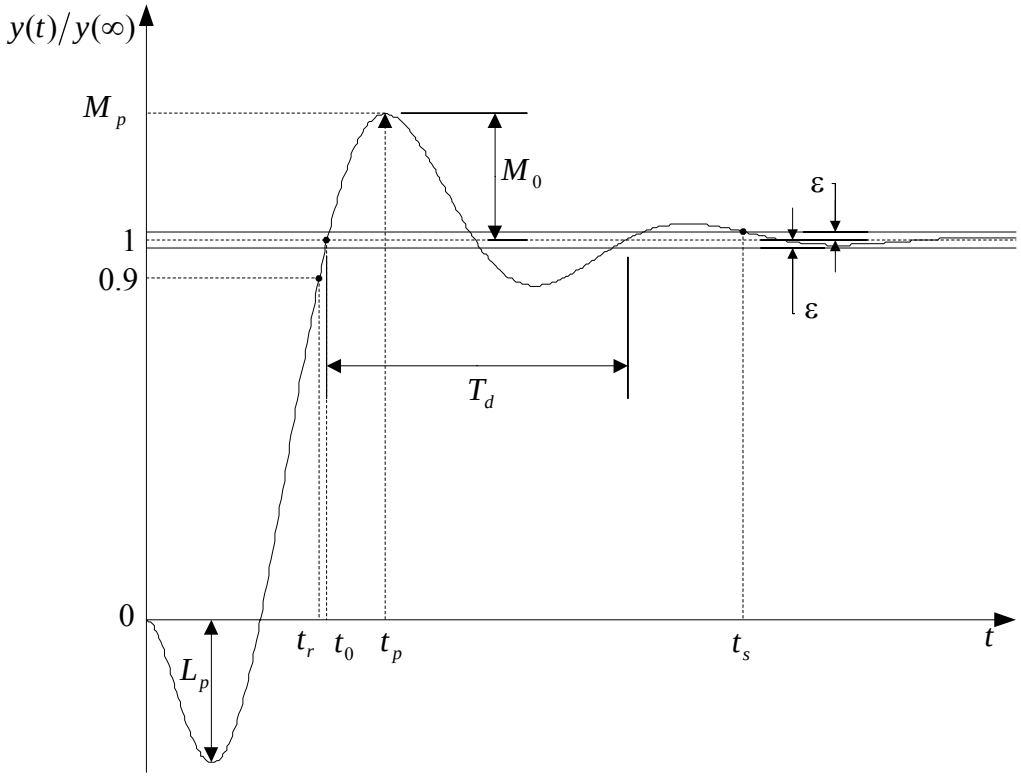


Figura 2.7: Resposta a degrau de um sistema oscilatório estável

É freqüente que a resposta de sistemas dinâmicos seja oscilatória. Assim, o conjunto completo das CRD's é definido para respostas oscilatórias. As definições estão ilustradas na Figura 2.7, que mostra a resposta a degrau típica de um sistema oscilatório estável. O gráfico apresentado é o da relação $y(t)/y(\infty)$, isto é, da resposta normalizada em relação ao valor final. Este procedimento tem a conveniência de padronizar o gráfico, deixando-o independente dos sinais algébricos do degrau e do ganho estático. Evidentemente o valor final da resposta normalizada é igual a um.

As CRD's são definidas a seguir:

(a) Tempo de subida (rise time) (t_r)

Por tempo de subida será entendido o tempo necessário para que a resposta $y(t)$ atinja 90% do valor final pela primeira vez, isto é:

$$t_r = t_{0.9} = \min\left\{ t \,\middle|\, \frac{y(t)}{y(\infty)} = 0.9 \right\} \qquad \textbf{(2.37)}$$

(b) Tempo de passagem por transitório nulo (t_0)

É o instante de tempo em que a parcela transitória da resposta a degrau é nula pela primeira vez. O instante t_0 só poderá ter um valor finito se a resposta a degrau passar pelo valor final pelo menos uma vez, o que geralmente implica uma resposta com oscilações. A definição formal de t_0 então é

$$t_0 = t_1 = \min\left\{ t \left| \frac{y(t)}{y(\infty)} = 1 \right. \right\} \tag{2.38}$$

(c) Tempo de pico(t_p)

É o instante no qual a resposta atinge o seu valor máximo. Matematicamente é preciso definir primeiramente o valor máximo da resposta (M_p)

$$M_p = \sup_t \left[\frac{y(t)}{y(\infty)} \right] \tag{2.39}$$

onde "sup" denota supremo. Quando M_p>1, o supremo também é o valor máximo da resposta normalizada, e o tempo de pico é o instante de tempo tal que

$$\frac{y(t_p)}{y(\infty)} = \max_t \left[\frac{y(t)}{y(\infty)} \right] = M_p \ , \quad M_p > 1 \tag{2.40}$$

Quando $M_p \leq 1$, o tempo de pico não é definido.

(d) Ultrapassagem máxima (*overshoot*)(M_0)

É a diferença entre a maior amplitude da resposta, no mesmo sentido do valor final, e o próprio valor final, expressa como uma fração do valor final. Como o pico máximo já foi definido para a resposta normalizada, a ultrapassagem pode ser expressa, de forma bastante simples, da seguinte forma:

$$M_0 = M_p - 1 \tag{2.41}$$

(e) Tempo de acomodação(*settling time*)(t_s)

É o tempo necessário para a resposta entrar dentro de uma faixa de tolerância em torno do valor final, dada por uma fração $\pm\varepsilon$ do valor final, e permanecer dentro dessa faixa após esse tempo. Matematicamente esta definição é expressa como segue:

$$t_s = \min\left\{ t_b \left| \left| \frac{y(t)}{y(\infty)} - 1 \right| \le \varepsilon\,, \quad \forall\, t \ge t_b \right. \right\} \tag{2.42}$$

Normalmente faz-se $\varepsilon = 0.02$ (2%) ou $\varepsilon = 0.05$ (5%). No presente trabalho adota-se $\varepsilon = 0.02$.

(f) Retrocesso (*undershoot*)(L_p)

É o valor absoluto da maior amplitude negativa da resposta normalizada, isto é, em sentido oposto ao valor final. O retrocesso é expresso como uma fração do valor final, da seguinte forma:

$$L_p = \left| \inf_t \left[\frac{y(t)}{y(\infty)} \right] \right| \tag{2.43}$$

onde "inf" denota ínfimo.

(g) Período de oscilação aparente(T_d)

É a duração da primeira oscilação detectada a partir do tempo de passagem por transitório nulo. Matematicamente pode ser definido como segue:

$$T_d = \min\left\{ T \left| y(t_0 + T) = y(t_0)\,, \frac{y'(t_0 + T)}{y(\infty)} > 0 \right. \right\} \tag{2.44}$$

2.5.3 Erro estacionário

Qualquer sistema de controle físico sofre inerentemente de erro estacionário em resposta a certos tipos de entradas. Um sistema pode não apresentar erro estacionário para entradas degraus, mas o mesmo sistema pode apresentar erro estacionário não-nulo

para uma entrada rampa. O fato de um dado sistema apresentar erro estacionário para um dado tipo de entrada depende do tipo de função de transferência de malha aberta do sistema.

Os sistemas de controle podem ser classificados de acordo com a sua habilidade para seguir entradas degraus, entradas rampa, entradas parabólicas etc. Este é um esquema razoável de classificar porque as entradas reais podem ser freqüentemente consideradas como combinação de tais entradas. Os valores dos erros estacionários devidos a estas entradas individuais são indicativos da "qualidade" do sistema.

Considere a seguinte função de transferência de malha aberta $G(s)H(s)$:

$$G(s)H(s) = \frac{K(s-z_1)(s-z_2)\ldots(s-z_m)}{s^N(s-p_1)(s-p_2)\ldots(s-p_p)} \qquad \textbf{(2.45)}$$

Ela envolve o termo s^N no denominador, representando um pólo de multiplicidade N na origem. O presente esquema de classificação é baseado no número de integrações indicadas pela função de transferência de malha aberta. Um sistema é chamado de tipo 0, tipo 1, tipo 2, ..., se $N = 0$, $N = 1$, $N = 2$, ... respectivamente. Aumentando-se o tipo, melhora-se a precisão mas dificulta-se a estabilização do sistema. Um compromisso entre precisão em regime estacionário e estabilidade relativa sempre é necessário. Na prática, raramente tem-se um sistema de tipo 3 ou maior porque geralmente é difícil projetar sistemas estáveis com mais do que duas integrações no canal direto.

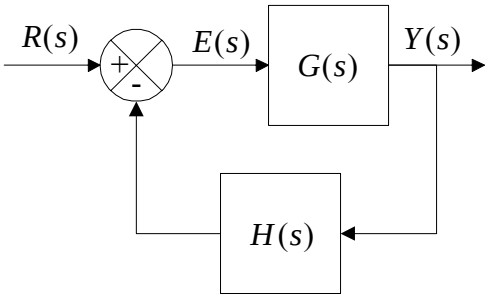


Figura 2.8: Sistema de controle com realimentação negativa

A função de transferência de malha fechada do sistema da Figura 2.8 é a seguinte:

$$\frac{Y(s)}{R(s)} = \frac{G(s)}{1+G(s)H(s)} \tag{2.46}$$

A função de transferência entre o sinal de erro atuante $e(t)$ e o sinal de entrada $r(t)$ é:

$$\frac{E(s)}{R(s)} = 1 - \frac{Y(s)H(s)}{R(s)} = \frac{1}{1+G(s)H(s)} \tag{2.47}$$

onde o erro atuante $e(t)$ é a diferença entre o sinal de entrada e o sinal de realimentação.

O teorema do valor final provê uma maneira conveniente de se determinar o desempenho estacionário de um sistema estável. Como $E(s)$ é:

$$E(s) = \frac{1}{1+G(s)H(s)} R(s) \tag{2.48}$$

o erro atuante estacionário é:

$$e_{ss} = \lim_{t\to\infty} e(t) = \lim_{s\to 0} sE(s) = \lim_{s\to 0} \frac{sR(s)}{1+G(s)H(s)} \tag{2.49}$$

onde

$$R(s) = \frac{R_N}{s^{N+1}} \quad \therefore \quad r(t) = R_N \frac{t^N}{N!} u_{-1}(t) \tag{2.50}$$

Usando a fórmula do erro estacionário pode-se facilmente construir a Tabela 2.2.

Tabela 2.2: Valor do erro em termos do tipo do sistema e do sinal de entrada

		$r(t)$		
		Entrada degrau $r(t) = R_0$	Entrada rampa $r(t) = R_1 t$	Entrada parábola $r(t) = 0{,}5R_2 t^2$
N	Sistema tipo 0	e_{ss}	∞	∞
	Sistema tipo 1	0	e_{ss}	∞
	Sistema tipo 2	0	0	e_{ss}

Se o sistema for de tipo 0, o erro em regime permanente terá um valor finito e diferente de zero para uma entrada degrau; para sinais de entrada de ordem superior o erro tenderá a aumentar ilimitadamente com o tempo.

Para sistemas de tipo 1, o erro em regime permanente para uma entrada degrau é zero, e terá um valor finito e diferente de zero para uma entrada rampa; para sinais de entrada de ordem superior o erro tende ao infinito com o tempo.

Em sistemas tipo 2, o erro em regime permanente é zero para entradas degrau e rampa, tem valor finito e diferente de zero para entrada parábola, e para sinais de entrada de ordem superior o erro tende ao infinito com o tempo.

A Tabela 2.2 pode ser estendida para sistemas e entradas de tipo e ordem superiores, embora na prática é rara a ocorrência de sistemas de tipo superior a dois. E pode-se estabelecer a seguinte regra: na diagonal principal da tabela o valor de erro estacionário é finito e diferente de zero; abaixo da diagonal o erro em regime permanente é zero e, acima da diagonal o erro tende ao infinito com o tempo.

Cabe indicar que o erro estacionário tem significado imediato unicamente quando o sistema tiver realimentação unitária; nos demais casos o erro estacionário pode ser calculado, mas o seu significado necessitará de interpretação. Essa interpretação poderá vir através da redução do diagrama de blocos da Figura 2.8 à forma com realimentação unitária, ou da adoção de valores normalizados para as variáveis (sistema p.u.).

2.6 Compensação

Em seqüência serão apresentados brevemente os procedimentos para o projeto e a compensação de sistemas de controle invariantes no tempo, lineares, com uma entrada e uma saída. A compensação é a modificação da dinâmica do sistema para cumprimento das especificações.

2.6.1 Considerações preliminares

Os sistemas de controle são projetados para desempenhar tarefas específicas. Os requisitos impostos aos sistemas de controle normalmente são denominados *especificações de desempenho*. Geralmente são relativos à precisão, estabilidade relativa e velocidade de resposta.

Para problemas rotineiros de projeto, as especificações de desempenho devem ser fornecidas em termos de valores numéricos precisos. Em outros casos, podem ser dadas parcialmente em termos de faixas de valores aceitáveis, as quais podem ser ajustadas durante o desenvolvimento do projeto, já que as especificações solicitadas podem não ser possíveis de serem satisfeitas simultaneamente.

De um modo geral, as especificações de desempenho não devem ser mais restritivas do que o necessário para o desempenho de uma dada tarefa. Se a precisão em operação em regime permanente for a principal exigência de um dado sistema de controle, então não deverão ser exigidas especificações de desempenho desnecessariamente rígidas em relação à resposta transitória, visto que tais especificações exigirão sistemas de controle mais complexos e caros do que o necessário. É importante lembrar que a parte mais importante do projeto de um sistema de controle é estabelecer precisamente as especificações de desempenho de modo que resultem em um sistema de controle adequado à sua finalidade.

O projeto de um sistema de controle consiste de um processo de tentativa e erro tecnicamente orientado, até se chegar a um controlador que leve o sistema a cumprir todas as especificações de desempenho. Após o projeto do sistema, o projetista verifica se todas as especificações de desempenho são satisfeitas. Se não for o caso, então o processo de projeto é repetido, através de alteração dos valores dos parâmetros

ajustáveis, ou modificando a configuração do sistema, até que as especificações sejam cumpridas. Embora o projeto seja baseado em um procedimento de tentativa e erro, a vivência e o conhecimento do projetista desempenham papel importante para que o projeto seja bem-sucedido.

No projeto do sistema de controle normalmente é considerada a dinâmica do sistema controlado como sendo um subsistema de estrutura e parâmetros fixos, e para poder ajustar o sistema de tal forma que as especificações de projeto desejadas sejam cumpridas, é necessária a incorporação de componentes adicionais que possuam parâmetros susceptíveis de serem ajustados. Tais componentes são conhecidos como *compensadores* ou *controladores* e o acréscimo desses elementos ao sistema é chamado de *compensação*.

2.6.2 Compensação em cascata

Na Figura 2.9 é apresentada a estrutura padrão de sistema de controle com realimentação:

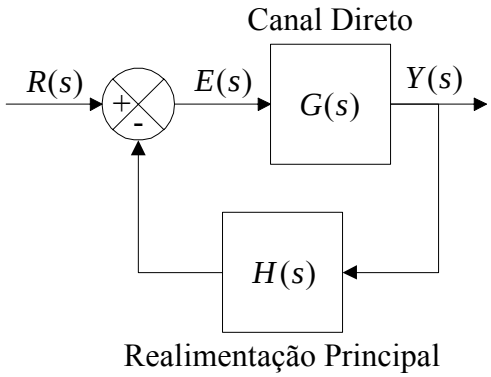


Figura 2.9: Estrutura padrão de sistema de controle com realimentação

Esta é a estrutura fixa do sistema de controle o qual consta dos seguintes blocos e sinais:

- $R(s)$: Sinal de referência ou entrada
- $E(s)$: Erro do sistema $E(s) = R(s)\text{-}Y(s)H(s)$

- $Y(s)$: Sinal de saída
- $G(s)$: bloco do canal direto
- $H(s)$: bloco do transdutor de realimentação (fixo)

O compensador ou controlador mais comumente utilizado é o controlador em cascata, o qual é inserido no canal direto do sistema, tal como é mostrado na Figura 2.10.

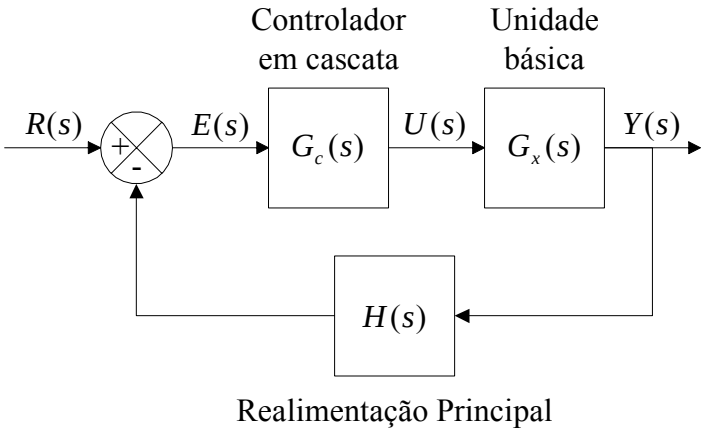


Figura 2.10: Estrutura de controle com controlador em cascata

O controlador em cascata $G_C(s)$ tem parâmetros ajustáveis e recebe o sinal de erro $E(s)$ o qual é modificado de acordo com a dinâmica própria do controlador para gerar o sinal de controle $U(s)$, que por sua vez é o sinal de entrada à unidade básica $G_X(s)$ do sistema.

2.6.3 Compensação por realimentação

Uma alternativa à compensação em cascata é realimentar o sinal de alguns elementos e colocar um compensador no caminho de realimentação interna resultante. Esta compensação é conhecida como compensação por realimentação (ou retroação) e algumas das suas variantes mais utilizadas são mostradas nas Figuras 2.11, 2.12 e 2.13.

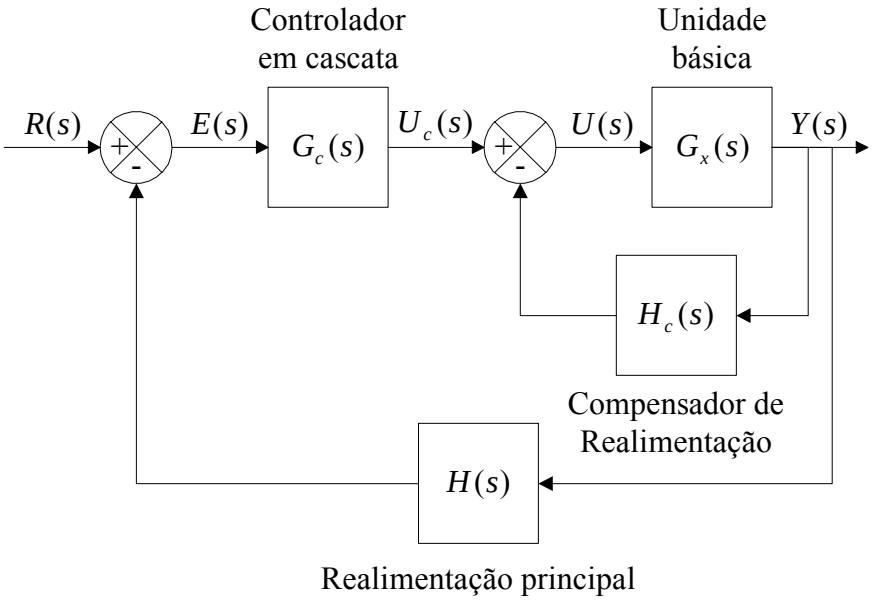


Figura 2.11: Estrutura de controle com controlador em cascata e compensador em realimentação

A Figura 2.11 apresenta um sistema de controle com controlador em cascata $G_C(s)$ e compensador de realimentação $H_C(s)$. O sinal que utiliza o compensador de realimentação é o da saída $Y(s)$. O sinal de entrada à unidade básica é $U(s)$ e o sinal de saída do controlador em cascata é $U_C(s)$.

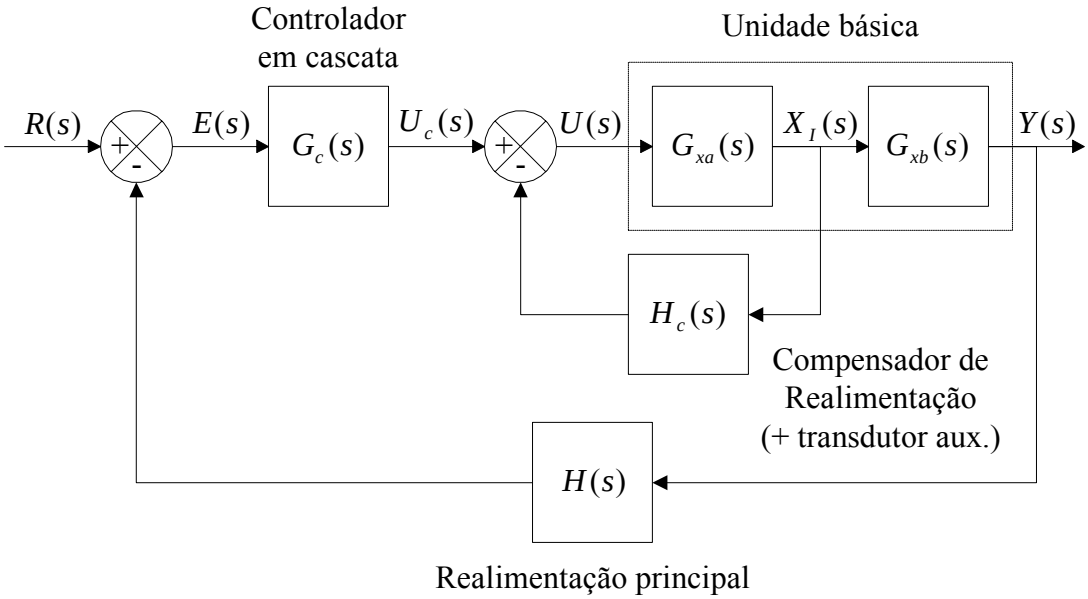


Figura 2.12: Estrutura de controle com controlador em cascata e compensador em realimentação de variável secundaria

Na Figura 2.12 mostra-se uma variante do sistema anterior, onde o sinal que utiliza o compensador de realimentação é o de uma variável intermediária $X_I(s)$ da unidade básica. Usualmente esta variável intermediária é a primeira ou segunda derivada do sinal de saída. Se for o caso de um controle de posição, o sinal utilizado pelo compensador de realimentação seria a velocidade ou a aceleração.

Uma estrutura de controle mais completa é a da Figura 2.13, onde é incluído um controlador em cascata interno $G_{cb}(s)$ na malha de compensação por realimentação, além do controlador em cascata externo $G_{ca}(s)$. Os blocos G_{ca}, G_{cb} e H_c são os blocos que podem conter parâmetros ajustáveis, enquanto que os blocos G_{xa}, G_{xb} e H são os blocos fixos do sistema e possuem parâmetros constantes. Esta estrutura é mais abrangente que as anteriores e pode facilmente se tornar em qualquer uma das estruturas mais simples através de uma escolha adequada das funções de transferências dos blocos. Por causa destas vantagens é que a *toolbox* objeto do presente trabalho foi desenvolvida baseada nesta última estrutura de controle.

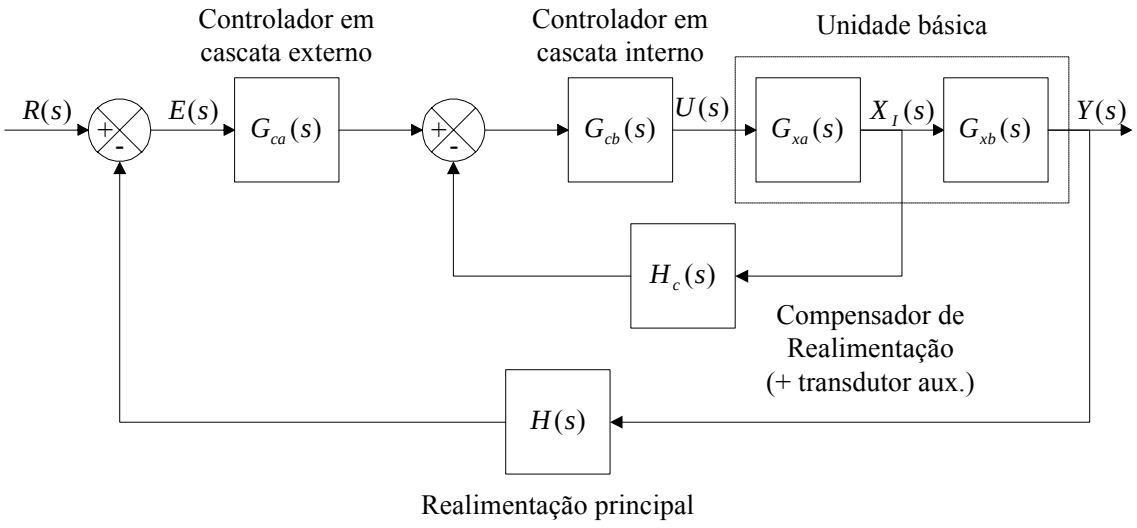


Figura 2.13: Estrutura de controle com controlador em cascata externo e interno, e compensador em realimentação de variável secundaria

2.7 Controle Ótimo

Quando é estipulado um critério ou um conjunto de especificações de desempenho e restrições de funcionamento para um sistema, o não atendimento destas condições dá origem a um problema de controle. Geralmente, a fim de se obter o desempenho desejado para o sistema, torna-se necessário usar um equipamento adicional juntamente com o sistema básico. São empregadas técnicas de projeto convencional ou de controle moderno na determinação da configuração e dos parâmetros do equipamento adicional requerido. Entre as técnicas de controle moderno destaca-se a otimização, que consiste em minimizar um índice de desempenho denominado função custo (por exemplo, $ID = \int_o^\infty e^2 dt$), buscando a obtenção de um único conjunto de parâmetros para o controlador adotado. O desempenho do sistema é então dito *ótimo* com relação à função custo adotada. Em oposição a isto, as técnicas convencionais satisfazem um conjunto preestabelecido de especificações de desempenho. Estes requisitos podem eventualmente ser alcançados por um certo numero de projetos diferentes. Em geral, estes projetos não satisfazem simultaneamente a um critério de desempenho ótimo escolhido. Em conseqüência, tais projetos são ditos *subótimos* [13].

2.7.1 Fundamentos

O advento dos computadores digitais conduziu ao desenvolvimento e à utilização da teoria de controle moderno. Esta teoria permitiu a obtenção de um desempenho ótimo que atende a algum critério de desempenho especificado. Ela envolve a minimização de uma *função custo* (FC). Este método, ao contrário das técnicas de projeto convencionais, faz uso acentuado de análise matemática. A escolha do índice de desempenho é quase sempre baseada em conveniência matemática, isto é, esta escolha permite o projeto analítico do sistema. Embora o método conduza a uma solução matemática única, o desempenho real do sistema pode não apresentar todas as características desejadas. Em outras palavras, o controle ótimo resultante satisfaz a especificação matemática de minimização da FC escolhida, mas pode não conduzir aos valores desejados de *overshoot*, tempo de estabilização, tempo de pico, etc. Deve haver um compromisso entre a escolha de uma função custo que inclua as características desejadas do sistema e que possa ser tratada com um volume aceitável de trabalho de cálculo.

2.7.2 Índices de desempenho

É desejável o estabelecimento de um ID simples que permita julgar "quão bom" é o tempo de resposta do sistema. Este ID deve apresentar três propriedades básicas: confiabilidade, facilidade de aplicação e seletividade. Ele deve ser confiável para uma dada classe de sistemas aos quais possa ser aplicado; deve ser fácil de aplicar e ao mesmo tempo seletivo, de modo que o sistema resultante seja nitidamente ótimo. Uma propriedade desejável é que a minimização do ID atenda algum ou todos os indicadores de desempenho convencionais.

Na Tabela 2.3 são indicados cinco índices de desempenho básicos e as suas formulações:

Tabela 2.3: Índices de desempenho básicos para sistemas de controle

	Formulação
Área de controle	$ID_1 = \int_0^\infty e\, dt$ **(2.51)**
Área de controle ponderada pelo tempo	$ID_2 = \int_0^\infty te\, dt$ **(2.52)**
Integral do erro quadrático (*IEQ*)	$ID_3 = \int_0^\infty e^2\, dt$ **(2.53)**
Integral do erro absoluto (*IEA*)	$ID_4 = \int_0^\infty \lvert e \rvert\, dt$ **(2.54)**
Integral do erro absoluto multiplicado pelo tempo (*IEAT*)	$ID_5 = \int_0^\infty t \lvert e \rvert\, dt$ **(2.55)**

Uma primeira medida da qualidade da resposta transitória de um sistema de controle a uma excitação a degrau é a *área de controle*, mostrada na Figura 2.14. O melhor sistema é aquele que apresenta a área de controle mínima, uma vez que neste caso $y(t)$ está mais próxima de $r(t)$. A área de controle é dada pela integral do erro sobre um intervalo de tempo especificado. Este critério é definido pela Eq. (2.51).

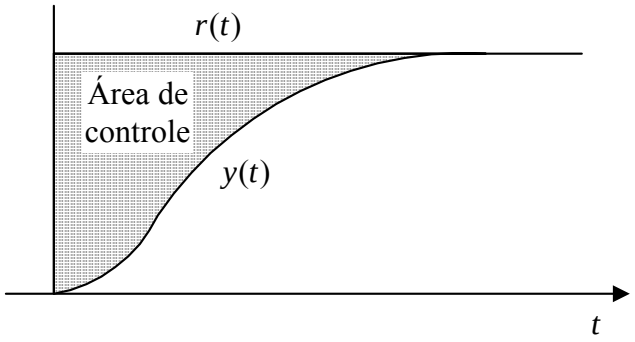


Figura 2.14: Área de controle

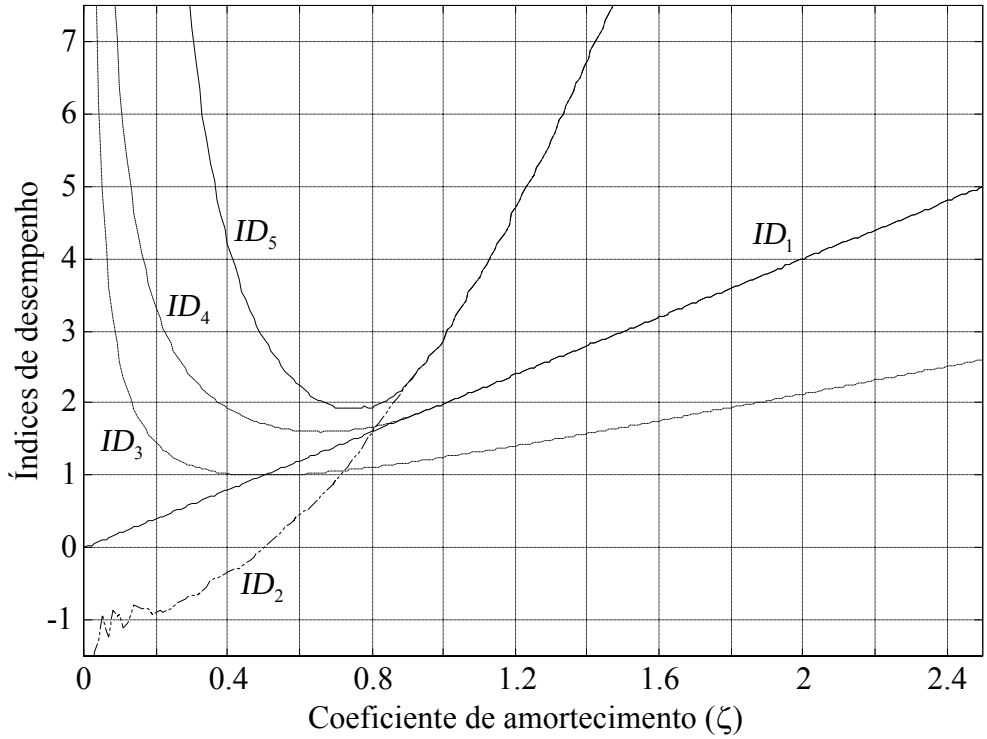


Figura 2.15: Índices de desempenho de sistema de segunda ordem

A Figura 2.15 apresenta o traçado dos vários IDs versus o coeficiente de amortecimento (ζ), para um sistema linear de segunda ordem simples descrito pela relação de controle:

$$\frac{Y(s)}{R(s)} = \frac{\omega_n^{\,2}}{s^2 + 2\zeta\omega_n s + \omega_n^{\,2}} \qquad \textbf{(2.56)}$$

Este sistema é equivalente a um sistema de controle como o da Figura 2.9, para:

$$G(s) = \frac{\omega_n^{\,2}}{s(s + 2\zeta\omega_n)} \quad \text{e} \quad H(s) = 1 \tag{2.57}$$

O erro é a diferença entre o sinal de referência e o de saída para qualquer instante de tempo maior ou igual a zero. Portanto:

$$e(t) = r(t) - y(t), \quad \forall\, t \geq 0 \tag{2.58}$$

Para estabelecer os índices de desempenho considera-se uma referência degrau; para tal caso por tratar-se de um sistema de tipo 1, o sistema apresenta erro estacionário nulo.

O mínimo de ID_1 ocorre em ζ=0, o que não é obviamente um valor adequado, já que para este valor o sistema está no limite de instabilidade. A minimização da área de controle leva à anulação do coeficiente de amortecimento, resultando numa resposta puramente oscilatória. Portanto, este índice não serve para otimização, e será rejeitado sem maiores considerações.

Uma modificação da área de controle como critério, usa o tempo como fator de ponderação. Este critério, conhecido como *área de controle ponderada pelo tempo*, acrescenta uma penalidade forte para os erros que não desaparecem rapidamente. Como pode ser observado na Figura 2.15 este critério falha do mesmo modo que o critério da área de controle.

A *integral do erro quadrático* (*IEQ*) é um índice de desempenho proposto, no qual ambos os valores do erro, positivos e negativos, presentes na resposta subamortecida aumentam o valor da integral. Isto elimina a deficiência do critério da área de controle. O mínimo do índice ocorre para ζ=0,5, considerado um valor razoável, embora acarrete uma maior ultrapassagem e um maior tempo de acomodação que no caso de ζ=0,7. Por outro lado este índice não é muito seletivo, uma vez que apresenta uma região de quase mínimo bastante ampla (curva muito achatada em torno do mínimo).

O seguinte índice de desempenho proposto é a *integral do erro absoluto* (*IEA*), pelo fato de se usar o valor absoluto do erro, a integral aumenta para erros positivos ou negativos. O valor mínimo do índice ocorre na proximidade ζ=0,7, acarretando um sistema bem amortecido e com uma seletividade maior que o *IEQ*.

A ponderação do valor absoluto do erro pelo tempo conduz ao índice de desempenho conhecido como *integral do erro absoluto multiplicado pelo tempo* (*IEAT*). Esta FC é mais seletiva do que o *IEA* e apresenta também o mínimo em ζ=0,7. Por esse motivo, esta função custo é a que vai ser utilizada no presente trabalho para fins de otimização do sistema de controle.

A revisão feita neste capítulo forma então a base para o procedimento de projeto e ajuste de sistemas de controle pelo Método das Inequações desenvolvido neste trabalho. Esse procedimento é direcionado para sistemas de controle com a configuração dada pelo diagrama de blocos da Figura 2.13, tendo como objetivo a estabilização do sistema e o cumprimento das restrições de funcionamento e especificações de desempenho estabelecidas pelo projetista. Como última etapa do projeto, o projetista terá à sua disposição a opção de otimizar o sistema, através da minimização da função custo *IEAT*.

CAPÍTULO 3

MÉTODO DAS INEQUAÇÕES

O método das inequações (MI) é uma técnica desenvolvida por Zakian e Al-Naib [26] para projeto assistido por computador de sistemas de controle. O método é baseado no principio de que o problema do projeto pode ser formulado em termos de um conjunto de inequações que define as limitações de operação e especificações de desempenho que o sistema deve satisfazer. Nessa linha, o projetista fornece as inequações, e o computador realiza uma busca iterativa, variando automaticamente os parâmetros do sistema de modo a satisfazer as inequações.

3.1 Formulação do Problema

As limitações de construção e operação, e as especificações de desempenho do projeto do sistema de controle devem ser reunidas num sistema de *m* inequações da forma:

$$\phi_i(\mathbf{p}) \le C_i \quad i = 1, 2, ..., m \tag{3.1}$$

onde **p** denota um ponto no espaço *n*-dimensional R^n, de coordenadas reais, de modo que:

$$\mathbf{p} = (p_1, p_2, ..., p_n) \tag{3.2}$$

p_1, p_2, ..., p_n são os parâmetros ajustáveis dos controladores, por isto **p** é denominado *vetor de parâmetros* ou *vetor de projeto,* ϕ_i são funções reais de **p**, e C_i são números reais. Cada Ineq. (3.1) define um conjunto S_i de pontos nesse espaço. Um tal conjunto pode ser denotado por:

$$S_i = \left\{ \mathbf{p} \,\middle|\, \phi_i(\mathbf{p}) \le C_i \right\} \tag{3.3}$$

O contorno de cada um destes conjuntos é definido no espaço *n*-dimensional pela equação:

$$\phi_i(\mathbf{p}) = C_i \tag{3.4}$$

Se existir um ponto **p** em R^n que satisfaça simultaneamente todas as Ineqs. (3.1), então **p** pertence a cada um dos conjuntos S_i. Seja S a interseção de todos os conjuntos S_i.

$$S = \bigcap_{i=1}^{m} S_i \tag{3.5}$$

No exemplo da Figura 3.1 se mostra graficamente a Eq. (3.5) para o caso de três inequações ($m = 3$) e dois parâmetros variáveis ($n = 2$).

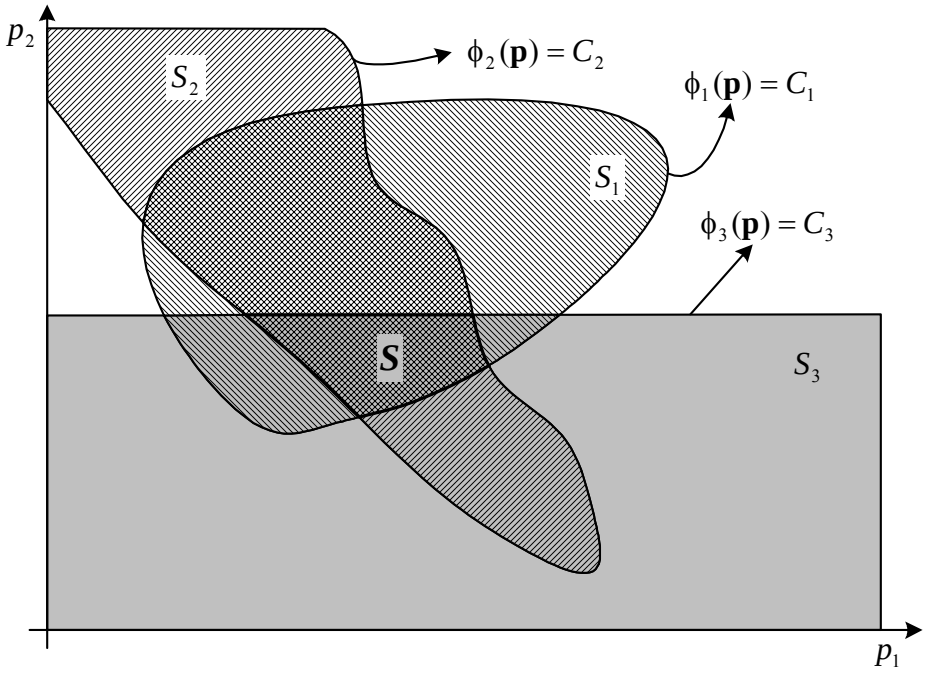


Figura 3.1: Exemplo gráfico do conjunto admissível *S*

É evidente que **p** só satisfará todas as inequações se, e somente se, **p** $\in$ *S*. *S* é denominado o *conjunto admissível*, e qualquer **p** em *S* é um *ponto admissível*. Portanto, após o problema de projeto ter sido formulado pelo sistema de Ineqs. (3.1), sua solução consiste em obter um ponto admissível.

3.2 Processo dos Contornos Móveis

Para resolver o sistema de Ineqs. (3.1), Zakian e Al-Naib [26] desenvolveram uma *busca iterativa*, que parte de um ponto inicial arbitrário $\mathbf{p}^0$ para um ponto admissível, isto é, para qualquer ponto em *S*. O algoritmo, chamado de *Processo dos Contornos Móveis* (PCM), é descrito a seguir.

Seja $\mathbf{p}^k$ o valor de $\mathbf{p}$ após seu *k*-ésimo movimento. A inequação $\phi_i(\mathbf{p}) \le \phi_i(\mathbf{p}^k)$ define um conjunto $S_i^{\,k}$ com o contorno definido por $\phi_i(\mathbf{p}) = \phi_i(\mathbf{p}^k)$. Um movimento é realizado do ponto $\mathbf{p}^k$ ao ponto $\tilde{\mathbf{p}}^k$. Se para cada $i = 1, 2, ..., m$, o contorno definido por $\phi_i(\mathbf{p}) = \phi_i(\tilde{\mathbf{p}}^k)$ é mais próximo, ou não mais distante, do contorno de S_i, então o ponto $\tilde{\mathbf{p}}^k$ é aceito e converte-se no novo ponto $\mathbf{p}^{k+1}$. Depois de um número suficiente de passos com sucesso, o contorno de $S_i^{\,k}$ coincide com o de S_i para cada $i = 1, 2, ..., m$, e o problema está resolvido. Para descrição do processo, são definidas as seguintes equações:

$$C_i^{\,k} = \begin{cases} C_i, & \text{se } \phi_i(\mathbf{p}^k) \le C_i \\ \phi_i(\mathbf{p}^k), & \text{se } \phi_i(\mathbf{p}^k) > C_i \end{cases} \qquad \textbf{(3.6)}$$

$$S_i^{\,k} = \left\{ \mathbf{p} \,\middle|\, \phi_i(\mathbf{p}) \le C_i^{\,k} \right\} \qquad \textbf{(3.7)}$$

$$S^k = \bigcap_{i=1}^{m} S_i^{\,k} \qquad \textbf{(3.8)}$$

O movimento do ponto $\mathbf{p}^k$ ao ponto $\tilde{\mathbf{p}}^k$ é chamado de *tentativa*. Se o ponto $\tilde{\mathbf{p}}^k$ é um sucesso, então $\mathbf{p}^{k+1} = \tilde{\mathbf{p}}^k$; senão, o ponto $\tilde{\mathbf{p}}^k$ é um fracasso e outras tentativas são feitas a partir de $\mathbf{p}^k$ até que um sucesso ocorra.

Um ponto tentativa $\tilde{\mathbf{p}}^k$ é considerado um sucesso se, e somente se:

$$\tilde{C}_i^{\,k} \le C_i^{\,k}, \quad \text{para todo} \quad i = 1, 2, ..., m \qquad \textbf{(3.9)}$$

A Figura 3.2 ilustra como um sucesso é obtido num espaço R^2 para uma inequação.

A cada passo com sucesso o contorno de S^{k+1} fica mais perto do contorno de *S*. O processo estará terminado quando o contorno de S^k coincida com o de *S*, ou seja quando:

$$C_i^{\,k} = C_i, \quad \text{para todo} \quad i = 1, 2, ..., m \tag{3.10}$$

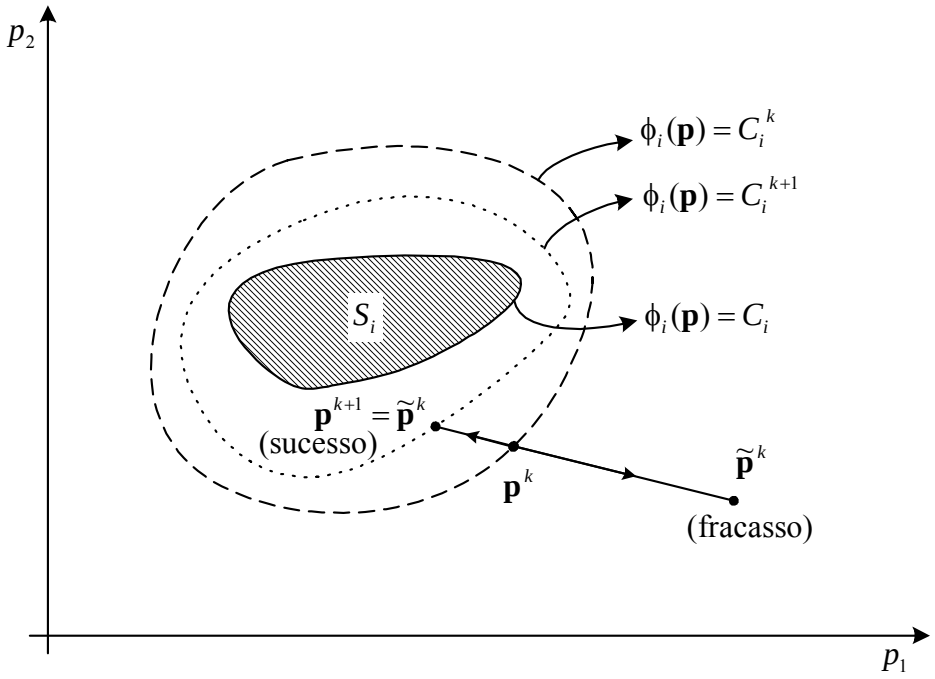


Figura 3.2: Ilustração do PCM

Conceitualmente, é útil imaginar-se o movimento dos contornos, e por esta razão o algoritmo é chamado de *processo dos contornos móveis* (PCM), ou *moving boundaries process* (MBP), em inglês.

3.3 Geração de Tentativas

Para a geração dos pontos tentativas $\tilde{\mathbf{p}}^k$ é utilizado o método sugerido por Rosenbrock [21], o qual é descrito a seguir. Seja $\tilde{\mathbf{p}}^k$ definido pela seguinte equação:

$$\tilde{\mathbf{p}}^k = \mathbf{p}^k + e_i \mathbf{v}_i{}^j \tag{3.11}$$

onde $\mathbf{v}_i{}^j$ para i=1, 2, ..., n denota um conjunto de vetores unitários ortonormais, e e_i corresponde a um conjunto de valores reais.

Se o ponto tentativa gerado $\tilde{\mathbf{p}}^k$ é um sucesso, e_i é substituído por αe_i, sendo α>1. Em caso contrário, se $\tilde{\mathbf{p}}^k$ é um fracasso, e_i é substituído por βe_i, sendo –1<β<0. Em qualquer um dos casos, i é substituído por i+1 na Eq. (3.11). Uma *iteração* (*it*) do PCM é definida como n tentativas consecutivas [26], portanto, depois da última tentativa (quando i=n) da it-ésima iteração, faz-se i=1 e it=it+1 para a próxima tentativa.

Assim que um sucesso seguido de um fracasso tenha acontecido para cada i=1, 2, ..., n então $\mathbf{v}_i{}^j$ é substituído por $\mathbf{v}_i{}^{j+1}$ para i=1, 2, ..., n. Os vetores $\mathbf{v}_i{}^j$ são calculados da seguinte forma:

Seja d_i a soma de todos os valores de e_i que geraram sucesso na j-ésima fase, e seja:

$$\left.\begin{aligned} \mathbf{a}_1 &= d_1 \mathbf{v}_1{}^j + d_2 \mathbf{v}_2{}^j + \ldots + d_n \mathbf{v}_n{}^j \\ \mathbf{a}_2 &= d_2 \mathbf{v}_2{}^j + \ldots + d_n \mathbf{v}_n{}^j \\ &\vdots \\ \mathbf{a}_n &= d_n \mathbf{v}_n{}^j \end{aligned}\right\} \tag{3.12}$$

Obtém-se um conjunto de vetores ortonormais de $\mathbf{a}_i$ através do procedimento de Gram-Schmidt:

$$
\left.
\begin{aligned}
&\mathbf{b}_1 = \mathbf{a}_1 \\
&\mathbf{v}_1{}^{j+1} = \mathbf{b}_1 / \|\mathbf{b}_1\| \\
&\mathbf{b}_2 = \mathbf{a}_2 - \left(\mathbf{a}_2, \mathbf{v}_1{}^{j+1}\right)\mathbf{v}_1{}^{j+1} \\
&\mathbf{v}_2{}^{j+1} = \mathbf{b}_2 / \|\mathbf{b}_2\| \\
&\vdots \\
&\mathbf{b}_n = \mathbf{a}_n - \sum_{k=1}^{n-1}\left(\mathbf{a}_n, \mathbf{v}_k{}^{j+1}\right)\mathbf{v}_k{}^{j+1} \\
&\mathbf{v}_n{}^{j+1} = \mathbf{b}_n / \|\mathbf{b}_n\|
\end{aligned}
\right\} \qquad \textbf{(3.13)}
$$

Onde a notação (**x**,**y**) denota o produto escalar de dois vetores *n*-dimensionais **x** e **y**:

$$\left(\mathbf{x},\mathbf{y}\right) = \mathbf{x}^t\mathbf{y} = x_1y_1 + x_2y_2 + ...+ x_ny_n \qquad \textbf{(3.14)}$$

e $\|\mathbf{x}\|$ denota o comprimento do vetor **x** definido através de:

$$\|\mathbf{x}\| = \left(\mathbf{x},\mathbf{x}\right)^{1/2} \qquad \textbf{(3.15)}$$

Inicialmente, com j=0, e_i e $\mathbf{v}_i{}^j$ são escolhidos arbitrariamente. Porém, a medida que j vai sendo incrementado, a taxa de convergência de S^k a S tende a melhorar, devido a que $\mathbf{v}_1{}^j$ vai progressivamente se orientando na direção de maior avanço, $\mathbf{v}_2{}^j$ na melhor direção normal a $\mathbf{v}_1{}^j$, e assim por diante. As constantes α e β podem ser escolhidos também de forma arbitrária.

Para o presente trabalho foram escolhidos os seguintes valores iniciais:

$$e_i = \begin{cases} 0.1\,p_i, & se\ p_i \neq 0 \\ 0.1, & se\ p_i = 0 \end{cases} \qquad \textbf{(3.16)}$$

$$\mathbf{v}^0 = \begin{matrix} \mathbf{v}_1{}^0 & \cdots & \cdots & \mathbf{v}_n{}^0 \\ \downarrow & & & \downarrow \end{matrix} \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 1 \end{bmatrix} \quad \textbf{(3.17)}$$

As constantes escolhidas foram $\alpha = 3$ e $\beta = -0{,}5$ devido aos bons resultados obtidos com esses valores em trabalhos anteriores [3] [8] [10] [26]. Cabe mencionar que os valores de α e β são implementados na *toolbox* como parâmetros ajustáveis pelo projetista.

CAPÍTULO 4

APLICAÇÃO DO MÉTODO DAS INEQUAÇÕES AO PROJETO DE SISTEMAS DE CONTROLE

Nos capítulos dois e três foram abordadas a teoria de sistemas de controle e o método das inequações. No presente capítulo será mostrado como ambos são acoplados para produzir um método aplicado ao projeto e ajuste de sistemas de controle.

4.1 Equacionamento da Estrutura de Controle

A estrutura de controle utilizada neste trabalho tem como base um sistema invariante no tempo, linear, com entrada e saída simples, que possui compensadores em cascata e realimentação, como é mostrado na Figura 4.1.

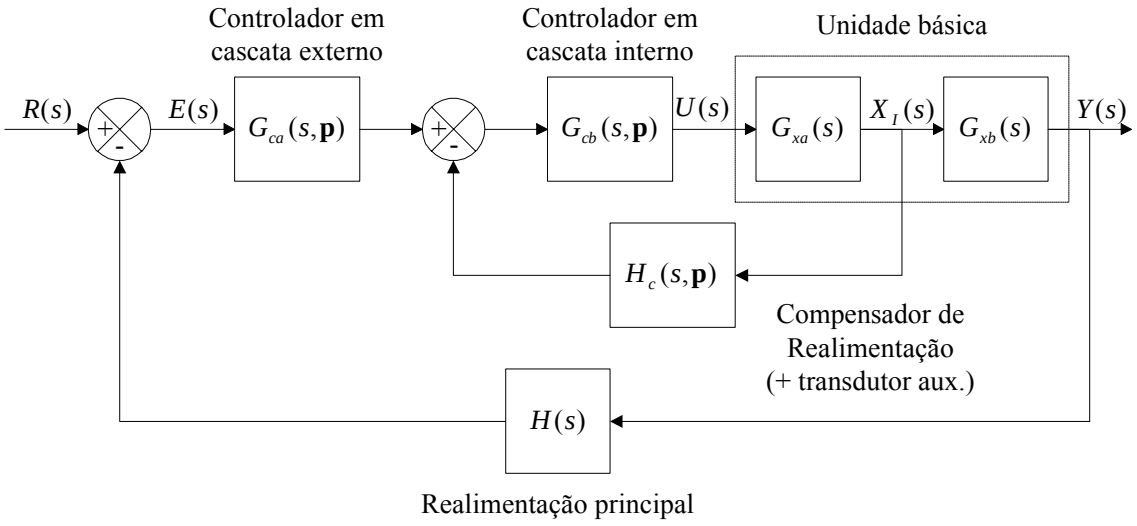


Figura 4.1: Estrutura de controle utilizada no projeto

O sinal de excitação $r(t)$ é um degrau com amplitude R_0 ajustável, e os índices de desempenho do sistema são as características de resposta ao degrau (CRD).

Considera-se que os blocos funcionais são representados por relações de polinômios em s. Os blocos das unidades básicas G_{xa}, G_{xb} e da realimentação externa H são

considerados fixos, ou seja, não possuem parâmetros ajustáveis; os blocos G_{ca}, G_{cb} e H_c fazem parte do controlador a ser projetado, e portanto possuem parâmetros susceptíveis de serem modificados, os mesmos que vão constituir o vetor **p**.

Para a obtenção das CRD do sistema é preciso primeiro definir os sinais de erro $E(s)$, a entrada à unidade básica ou sinal de controle $U(s)$, e a saída do sistema $Y(s)$, em função dos blocos funcionais G_{xa}, G_{xb}, H, G_{ca}, G_{cb} e H_c, e da entrada $R(s)$. Assim tem-se:

$$\left.\begin{aligned} G_{xa} &= \frac{P_1(s)}{P_2(s)}, \quad H = \frac{P_3(s)}{P_4(s)}, \quad G_{ca} = \frac{P_5(s)}{P_6(s)}, \\ H_c &= \frac{P_7(s)}{P_8(s)}, \quad G_{xb} = \frac{P_9(s)}{P_{10}(s)}, \quad G_{cb} = \frac{P_{11}(s)}{P_{12}(s)} \end{aligned}\right\} \tag{4.1}$$

$P_i(s)$ são polinômios em s, de coeficientes reais, constantes ou variáveis. Deste ponto em diante, os polinômios serão escritos simplesmente como P_i, omitindo-se a dependência de s.

Fazendo uso das técnicas de modificação e redução de diagramas de blocos apresentadas na seção 2.3.2, e de simplificações algébricas, têm-se as seguintes equações:

$$\frac{Y(s)}{R(s)} = W_y(s) = \frac{P_1P_4P_5P_8P_9P_{11}}{P_2P_4P_6P_8P_{10}P_{12} + P_1P_4P_6P_7P_{10}P_{11} + P_1P_3P_5P_8P_9P_{11}} \tag{4.2}$$

$$\frac{U(s)}{R(s)} = W_u(s) = \frac{P_2P_4P_5P_8P_{10}P_{11}}{P_2P_4P_6P_8P_{10}P_{12} + P_1P_4P_6P_7P_{10}P_{11} + P_1P_3P_5P_8P_9P_{11}} \tag{4.3}$$

$$\frac{E(s)}{R(s)} = W_e(s) = \frac{P_2P_4P_6P_8P_{10}P_{12} + P_1P_4P_6P_7P_{10}P_{11}}{P_2P_4P_6P_8P_{10}P_{12} + P_1P_4P_6P_7P_{10}P_{11} + P_1P_3P_5P_8P_9P_{11}} \tag{4.4}$$

Note-se que o denominador é o mesmo para as três funções de transferência.

4.2 Calculo de Funcionais

Uma vez definidas as funções de transferência do sistema de controle, devem ser calculadas as características das respostas do sistema que vão ser utilizadas para medir o seu desempenho.

Com exceção do erro estacionário que, será calculado para uma referência que vai depender do tipo do sistema, os funcionais utilizados no projeto serão calculados para um degrau de magnitude R_0 aplicado à referência , ou seja:

$$r(t) = R_0 u_{-1}(t) \quad \therefore \quad R(s) = \frac{R_0}{s} \tag{4.5}$$

A resposta a degrau, por depender do vetor de parâmetros ajustáveis, será denotada por:

$$y(t,\mathbf{p}) = R_0 \mathcal{L}^{-1}\left[\frac{W_y(s,\mathbf{p})}{s}\right] \tag{4.6}$$

Para denotar as características de resposta dinâmica do sistema, que dependem do valor dos parâmetros ajustáveis dos controladores, será utilizado o termo *funcional*, isto é, que dependem de **p**. Cada funcional vai ser denotado como A_i.

Os funcionais considerados tem relação com as características da resposta ao degrau (CRD), os valores de saída máximo e mínimo do controlador, a estabilidade do sistema, o erro em regime permanente, e a integral do erro absoluto multiplicada pelo tempo (*IEAT*), e são definidos a seguir:

Tempo de subida (t_r):

$$A_1(\mathbf{p}) = t_{0.9}(\mathbf{p}) = \min\left\{ t \,\middle|\, \frac{y(t,\mathbf{p})}{y(\infty,\mathbf{p})} = 0.9 \right\} \tag{4.7}$$

Tempo de pico (t_p):

$$A_2(\mathbf{p}) = \left\{ t_p(\mathbf{p}) \,\middle|\, y(t_p, \mathbf{p}) = \max_t y(t, \mathbf{p}) \right\} \tag{4.8}$$

O tempo de pico é definido quando a resposta possui pico máximo, senão o tempo de pico tende ao infinito de acordo com a sua definição (Ver seção 2.5.2).

Ultrapassagem máxima (M_0):

$$A_3(\mathbf{p}) = M_0(\mathbf{p}) = \sup_{t \geq 0} \left[\frac{y(t, \mathbf{p})}{y(\infty, \mathbf{p})} \right] - 1 \tag{4.9}$$

onde "sup" denota supremo.

Tempo de acomodação (t_s):

$$A_4(\mathbf{p}) = t_s(\mathbf{p}) = \min \left\{ t_b \,\middle|\, \left| \frac{y(t, \mathbf{p})}{y(\infty, \mathbf{p})} - 1 \right| \leq 0.02 \,, \quad \forall\, t \geq t_b \right\} \tag{4.10}$$

Período de oscilação aparente (T_d):

$$A_5(\mathbf{p}) = T_d(\mathbf{p}) = \min \left\{ T \,\middle|\, y(t_0 + T, \mathbf{p}) = y(t_0, \mathbf{p}), \quad \frac{y'(t_0 + T, \mathbf{p})}{y(\infty, \mathbf{p})} > 0 \right\} \tag{4.11}$$

Retrocesso(L_p):

$$A_6(\mathbf{p}) = L_p(\mathbf{p}) = \left| \inf_{t \geq 0} \left[\frac{y(t, \mathbf{p})}{y(\infty, \mathbf{p})} \right] \right| \tag{4.12}$$

onde "inf" denota ínfimo.

Saída em regime permanente (y_{rp}):

$$A_7(\mathbf{p}) = y_{rp}(\mathbf{p}) = y(\infty, \mathbf{p}) = \lim_{t\to\infty} y(t, \mathbf{p}) = R_0 \lim_{s\to 0} W_y(s, \mathbf{p}) \tag{4.13}$$

Saída máxima do controlador ($u_{\max}$):

$$A_8(\mathbf{p}) = u_{\max}(\mathbf{p}) = \sup_{t\geq 0} u(t, \mathbf{p}) \tag{4.14}$$

Saída mínima do controlador ($u_{\min}$):

$$A_9(\mathbf{p}) = u_{\min}(\mathbf{p}) = \inf_{t\geq 0} u(t, \mathbf{p}) \tag{4.15}$$

Abscissa de Estabilidade (σ_s):

$$A_{10}(\mathbf{p}) = \sigma_s(\mathbf{p}) = \max\left\{ \operatorname{Re}(s) \,\middle|\, \frac{1}{W_y(s, \mathbf{p})} = 0 \right\} \tag{4.16}$$

Erro estacionário (e_{ss}):

Para a determinação do erro estacionário, é preciso conhecer antes o tipo N do sistema (número de integradores do canal direto $G(s) = Y(s)/E(s)$), para assim saber qual sinal de entrada vai produzir uma saída com erro estacionário finito e diferente de zero. Este sinal é:

$$r(t) = R_N \frac{t^N}{N!} u_{-1}(t) \quad \therefore \quad R(s) = \frac{R_N}{s^{N+1}} \tag{4.17}$$

onde R_N é a magnitude do sinal de entrada definida pelo usuário.

O tipo do sistema também pode ser dado pelo número de zeros na origem contidos na função de transferência $W_e(s) = E(s)/R(s)$ entre o erro e a referência:

$$W_e(s,\mathbf{p}) = \frac{p_0 s^N (s-p_1)(s-p_2)...(s-p_w)}{(s-p_{w+1})(s-p_{w+2})...(s-p_{w+l})} \tag{4.18}$$

então:

$$A_{11}(\mathbf{p}) = e_{ss}(\mathbf{p}) = \lim_{s\to 0} R_N \frac{W_e(s,\mathbf{p})}{s^N} = (-1)^{w+l} R_N \frac{p_0 p_1 p_2 \ \cdots p_w}{p_{w+1} p_{w+2} \ \cdots p_{w+l}} \tag{4.19}$$

Integral do erro transitório absoluto multiplicado pelo tempo (*IEAT*):

$$A_{12}(\mathbf{p}) = IEAT(\mathbf{p}) = \int_0^\infty t \left| y(t,\mathbf{p}) - y(\infty,\mathbf{p}) \right| dt \tag{4.20}$$

A definição original da *IEAT* considera apenas o erro, e não o erro transitório, pois partia-se do pressuposto de que $e(\infty,\mathbf{p}) = 0$, para entrada degrau. Como este trabalho também abrange sistemas tipo zero, para os quais existe erro estacionário para entrada degrau, é necessário considerar-se o erro transitório de modo a garantir a existência da integral na Eq. (4.20). Para entradas degrau, é fácil verificar-se que

$$e(t,\mathbf{p}) - e(\infty,\mathbf{p}) = y(\infty,\mathbf{p}) - y(t,\mathbf{p}) \tag{4.21}$$

4.3 Formulação do Sistema de Inequações

Com a utilização dos funcionais previamente estabelecidos, as especificações ou índices de desempenho e restrições do sistema podem ser formulados na forma de um sistema de inequações da forma apresentada nas Ineqs. (3.1). Por exemplo, deseja-se que a abscissa de estabilidade seja menor que zero, o qual é condição fundamental para que o sistema seja estável e assim possa funcionar. Então:

$$\phi_1(\mathbf{p}) \le C_1, \text{ onde } \begin{cases} \phi_1(\mathbf{p}) = A_{10}(\mathbf{p}) \\ C_1 = 0 \end{cases} \tag{4.22}$$

Se adicionalmente é preciso de tempo de acomodação menor que 5 segundos e retrocesso menor que 2 unidades, tem-se o seguinte:

$$\phi_2(\mathbf{p}) \leq C_2, \text{ onde } \begin{cases} \phi_2(\mathbf{p}) = A_4(\mathbf{p}) \\ C_2 = 5 \end{cases} \quad \textbf{(4.23)}$$

$$\phi_3(\mathbf{p}) \leq C_3, \text{ onde } \begin{cases} \phi_3(\mathbf{p}) = A_6(\mathbf{p}) \\ C_3 = 2 \end{cases} \quad \textbf{(4.24)}$$

Adicionalmente às especificações de desempenho do sistema, devem ser incluídas restrições aos valores que podem assumir os parâmetros variáveis, como sempre acontece nos sistemas reais. Assim, para cada um dos parâmetros variáveis serão acrescentadas duas inequações, que vão servir para definir o valor máximo e mínimo de cada um deles. No caso de dois parâmetros variáveis com faixas entre 50 e 80, e –1 e 2, respectivamente, as inequações adicionais seriam as seguintes:

$$\phi_4(\mathbf{p}) \leq C_4, \textit{ onde } \begin{cases} \phi_4(\mathbf{p}) = -(p_1) \\ C_4 = -(50) \end{cases} \quad \textbf{(4.25)}$$

$$\phi_5(\mathbf{p}) \leq C_5, \textit{ onde } \begin{cases} \phi_5(\mathbf{p}) = p_1 \\ C_5 = 80 \end{cases} \quad \textbf{(4.26)}$$

$$\phi_6(\mathbf{p}) \leq C_6, \textit{ onde } \begin{cases} \phi_6(\mathbf{p}) = -(p_2) \\ C_6 = -(-1) \end{cases} \quad \textbf{(4.27)}$$

$$\phi_7(\mathbf{p}) \leq C_7, \textit{ onde } \begin{cases} \phi_7(\mathbf{p}) = p_2 \\ C_7 = 2 \end{cases} \quad \textbf{(4.28)}$$

4.4 Adaptação do Processo dos Contornos Móveis ao Problema

Uma vez que o problema de controle foi formulado como um conjunto de inequações, ele pode ser resolvido utilizando o Processo dos Contornos Móveis (PCM). O algoritmo

PCM, tal como foi descrito na seção 3.2, é utilizado para obter uma solução $\mathbf{p}^k$ do conjunto de Ineqs. (3.1), partindo de um ponto inicial $\mathbf{p}^0$ arbitrário.

Inicialmente, para cada novo valor de $\tilde{\mathbf{p}}^k$ devem ser obtidas as funções de transferência $W_y(s)$, $W_u(s)$ e $W_e(s)$.

Com as funções de transferência definidas, são obtidos os valores dos funcionais $A_i(\mathbf{p})$. É importante notar que a maioria dos funcionais só tem significado quando o sistema é estável, ou seja, quando a abscissa de estabilidade é menor que zero e a função de transferência $W_y(s)$ é própria. Por causa disto, o funcional $A_{10}(\mathbf{p})$, que representa a abscissa de estabilidade, é calculado primeiro, e os demais funcionais são calculados somente se $A_{10}(\mathbf{p})$ é menor que zero e $W_y(s)$ é própria.

No caso do PCM ser iniciado com um ponto $\mathbf{p}^0$ que não produza como resultado um sistema estável, as primeiras iterações do processo terão a tarefa de encontrar um ponto $\mathbf{p}^k$ que resulte em um sistema estável.

Uma vez que o sistema foi estabilizado, o processo iterativo continua, chegando num ponto $\mathbf{p}^k$ onde todas as inequações são satisfeitas, ou no qual é alcançado o limite de iterações estabelecido pelo usuário da *toolbox* de projeto.

4.5 Estratégias de Projeto

O tipo de sistema de controle estudado consiste de duas partes distintas que interagem entre si: a unidade básica e o controlador. A unidade básica inclui somente as partes que não são susceptíveis de modificações ao longo do projeto, enquanto que o controlador é caracterizado pelas estruturas das funções de transferência de seus blocos, e pelo vetor $\mathbf{p}$ de parâmetros variáveis, que deve satisfazer restrições e índices de desempenho representados por um sistema de inequações.

Em situações onde a forma do controlador não é completamente conhecida, o uso repetido do PCM, utilizando diferentes estruturas de controladores, proporciona informação que ajuda na escolha do controlador que ajusta-se melhor ao sistema em questão.

Outro problema a ser enfrentado pelo projetista é o de formular um conjunto de inequações realístico. Em particular este problema é refletido na escolha dos limites C_i (i=1, 2, ... , m). Se os limites estiverem muito baixos, o conjunto solução S será vazio. Por outro lado, se os limites forem muito altos, melhoras adicionais no projeto podem ser possíveis. Neste caso, uma nova execução do PCM pode resolver o problema. Usualmente uma execução prévia do PCM dá uma boa indicação de se os limites estão muito baixos ou altos, e assim eles podem ser ajustados e o processo executado novamente.

Se, depois de um numero razoável de iterações, **p** está dentro de alguns conjuntos S_i, mas encontra-se fora dos conjuntos restantes, sem lograr progressos adicionais, então, supondo que o progresso não tenha sido impedido pela interferência de um mínimo local de $\phi_i(\mathbf{p})$, S é um conjunto vazio, e uma solução aproximada do conjunto de inequações terá sido obtida. O fato de S ser conjunto vazio pode significar uma formulação inadequada do problema; neste caso o projetista deve reformular o problema, adotando um dos procedimentos seguintes, ou, ainda, uma combinação dos mesmos:

Por exemplo, pode-se aumentar o número de parâmetros variáveis na estrutura do controlador. Alternativamente, os limites C_i podem ser aumentados de forma que reflitam a importância relativa das inequações. Com a reformulação, é mais factível que o conjunto S não seja vazio.

Se, depois de umas poucas iterações, **p** consegue estar dentro de S e está bem no interior do contorno de alguns dos conjuntos S_i, isto pode ser interpretado como um indicativo de que os limites C_i estão altos demais, e podem ser diminuídos pelo menos até o ponto em que $\mathbf{p}^k$ consegue ficar dentro de todos os novos limites. Alternativamente, o projetista pode optar por reformular o problema, adicionando uma nova inequação m+1 às já existentes, com a finalidade de reduzir o tamanho da região admissível S.

4.6 Otimização do Projeto

Uma das principais contribuições do presente trabalho é usar o PCM para otimizar o projeto do controlador, através da minimização do *IEAT*.

Um projeto de controle utilizando o MI é finalizado quando é encontrado um ponto $\mathbf{p}^k$ dentro da região admissível S (se S não for vazia). Qualquer ponto dentro de S é uma solução admissível ao problema. É possível encontrar entre as soluções admissíveis uma que seja a melhor delas, ou, em outras palavras, a solução ótima, respeitadas as restrições do projeto.

Para otimizar o projeto é acrescentada uma inequação $\phi_{m+1}(\mathbf{p}) \leq C_{m+1}$ ao sistema, onde o valor de C_{m+1} vai definir um subconjunto de pontos S_{m+1} vazio. O ponto inicial da nova execução é o ponto final da execução anterior, com o qual assegura-se que as respostas encontradas vão todas satisfazer as restrições do projeto original:

$$\mathbf{p}^0{}_{\text{otimização}} = \mathbf{p}^k{}_{\text{original}} \tag{4.29}$$

Ao executar o MI com as modificações indicadas, o PCM vai chegar até o número máximo de iterações sem satisfazer a nova inequação m+1. No final do processo é obtido um novo ponto $\mathbf{p}^k$, o qual é a solução mais próxima à inequação m+1 e que cumpre com o conjunto de inequações do problema original; em outras palavras é obtida a solução ótima do problema com restrições.

Este procedimento é adotado neste trabalho, com a inequação:

$$A_{12}(\mathbf{p}) \leq 0 \tag{4.30}$$

onde $A_{12}(\mathbf{p})$ é o funcional dado pela integral do erro absoluto multiplicado pelo tempo (*IEAT*), tentando fazê-lo chegar a zero. É evidente que isto é impossível, visto que $IEAT > 0$, por definição, mas quando se chegar a um vetor $\mathbf{p}$ para o qual seja impossível reduzir $A_{12}(\mathbf{p})$, ter-se-á minimizado a *IEAT*. O sistema estará então otimizado com relação ao *IEAT*, respeitadas as restrições impostas pelas demais inequações.

CAPÍTULO 5

IMPLEMENTAÇÃO DO PROGRAMA DE PROJETO UTILIZANDO O MATLAB

Neste capítulo é apresentado brevemente o aplicativo *MATLAB*, o qual foi utilizado no presente trabalho para implementação do sistema computacional do MI; após essa apresentação, é explicada em detalhe a implementação do sistema.

5.1 Que É o MATLAB?

O Sistema MATLAB é uma linguagem de alto desempenho para computação técnica e científica. Ele integra cálculo, visualização e programação em um ambiente de fácil uso onde os problemas e soluções são expressos em uma notação matemática bastante familiar a engenheiros. Usos típicos incluem:

- Matemática e cálculo.
- Desenvolvimento de algoritmos.
- Modelagem e simulação.
- Análise, exploração, e visualização dos dados.
- Gráficos científicos e de engenharia.
- Desenvolvimento de aplicações, incluindo construção de interface gráfica com o usuário.

MATLAB é um sistema interativo onde o elemento básico de dados é uma matriz que não requer dimensionamento. Isto permite resolver muitos problemas de computação científica, especialmente aqueles envolvendo matrizes e vetores, em uma fração do tempo necessário para escrever o programa numa linguagem escalar e não interativa, tais como C e Fortran.

O nome MATLAB vem de *"matrix laboratory"*. MATLAB foi originalmente escrito para prover um acesso fácil aos programas de cálculo matricial desenvolvidos pelos

projetos LINPACK e EISPACK, os quais representam o estado da arte no que se refere a programas para cálculo matricial [24].

MATLAB tem evoluído com os anos com contribuições de muitos usuários. Em ambientes universitários, é a ferramenta padrão para o ensino de cursos de matemática básica e avançada, engenharia, e ciências. Na indústria, MATLAB é a ferramenta escolhida para pesquisa, desenvolvimento e análise de alta produtividade.

MATLAB é caracterizado por possuir uma família de soluções para aplicações específicas chamadas *toolboxe*s.

Um aspecto muito importante para a maioria dos usuários de MATLAB é que as *toolboxes* permitem aprender e aplicar tecnologias especializadas. As *toolboxes* são coleções de funções MATLAB (*M-files*) que estendem o ambiente do MATLAB para resolver problemas específicos de certas áreas. As áreas nas quais tem-se *toolboxes* disponíveis incluem processamento de sinais, sistemas de controle, redes neurais, lógica nebulosa (ou difusa), simulação e muitas outras.

Finalmente, e mais importante, ferramentas não disponíveis nas *toolboxes* existentes, podem ser criadas escrevendo-se novos *M-files*.

5.1.1 *Toolbox* de sistema de controle

Na *toolbox* de sistemas de controle [18] é utilizada a estrutura de matrizes e são construídas, na linguagem básica de MATLAB, funções especializadas da engenharia de controle. A *toolbox* de sistemas de controle é uma coleção de comandos e funções em forma de *M-files*, nos quais são implementadas técnicas de projeto, modelagem e análise de sistemas de controle. Os sistemas de controle podem ser modelados como funções de transferência, ou como blocos zero–pólo–ganho, ou como equações de estado, permitindo a utilização de técnicas de controle clássico e moderno. Podem ser manipulados sistemas de tempo contínuo ou discreto; são incluídos comandos para conversão entre os vários tipos de modelos. Respostas no domínio do tempo, da freqüência e lugares das raízes de equações características podem ser calculados e

apresentados graficamente. Outras funções permitem alocação de pólos, controle ótimo e estimação.

Para o presente trabalho, foi utilizado o seguinte grupo de funções da *toolbox* de controle: DCGAIN, ZERO, POLE, STEP, ISPROPER e ZPK. A descrição de cada função e a sua forma de utilização são apresentadas no apêndice A.2.

5.1.2 *Toolbox* CACCON

A *toolbox* CACCON [7] é um conjunto de comandos MATLAB para análise e projeto de sistemas de controle. É concebida para ser uma *toolbox* complementar à *toolbox* de sistema de controle, e contém comandos para controle baseado em lugar das raízes e análise no domínio do tempo. Contem também comandos gráficos para permitir ao usuário elaborar gráficos bidimensionais, com maior controle dos parâmetros da saída gráfica.

Os comandos utilizados da *toolbox* CACCON são: SOMAPOLI e CARACRD. A descrição de cada função e a sua forma de utilização são apresentadas no apêndice A.1.

5.1.3 *Toolbox* de matemática simbólica

A *toolbox* de matemática simbólica [23] incorpora o cálculo simbólico ao ambiente de MATLAB. Os cálculos que a *toolbox* incorpora são: cálculo diferencial e integral, álgebra linear, simplificação de expressões algébricas, solução de equações diferenciais algébricas, avaliação numérica de expressões matemáticas, transformadas de Fourier, Laplace, Z e as suas correspondentes antitransformadas, e outras funções especiais da matemática clássica aplicada.

Os comandos utilizados da *toolbox* de matemática simbólica são: SYMS, SYM2POLY e COLLECT. A descrição de cada função e a sua forma de utilização são apresentadas no apêndice A.3.

5.1.4 Interface gráfica com o usuário (GUI)

Boas ferramentas computacionais convertem o trabalho em algo agradável. A interface gráfica com o usuário (GUI, *Graphics User Interface*) faz com que o trabalho de utilização de uma aplicação em computador seja uma experiência agradável, com menor tempo dispendido na entrada de dados, e maior tempo para o análise de resultados gráficos e numéricos.

Ao se facilitar o uso de uma ferramenta computacional através da GUI, permite-se que usuários interessados na aplicação científica possam ter acesso a ela, sem precisar de profundos conhecimentos de computação ou de conhecimentos prévios do seu funcionamento.

Muitas das ferramentas de GUI não estavam disponíveis no MATLAB há pouco tempo atrás, e por isto é importante ressaltar que o presente trabalho, além das melhoras implementadas no plano conceitual da sistemática de projeto, apresenta uma contribuição importante ao colocar à disposição uma aplicação com GUI que é fácil de usar e com a qual é possível fazer rapidamente modificações no projeto e em suas especificações, convertendo o trabalho do projetista em uma experiência amena de interação contínua com a *toolbox*, onde podem ser exploradas muitas opções, sem chegar a fatigar o projetista.

5.2 Implementação do Programa de Projeto

A *toolbox* do projeto está composta de onze sub-rotinas, as quais são classificadas de acordo com a função que elas realizam, da seguinte forma:

Sub-rotinas da interface gráfica com o usuário (GUI):

mi.m

mi_config.m

mi_f.m

imp.m

ajuda.m

Sub-rotinas do processo dos contornos móveis (PCM):

pcm.m

gram_s.m

tex2dad.m

Sub-rotinas de funcionais do sistema de controle (FSC):

fnal.m

pol2ft.m

nd2ft.m

No diagrama da Figura 5.1 é ilustrada a interação entre as diferentes sub-rotinas da *toolbox*.

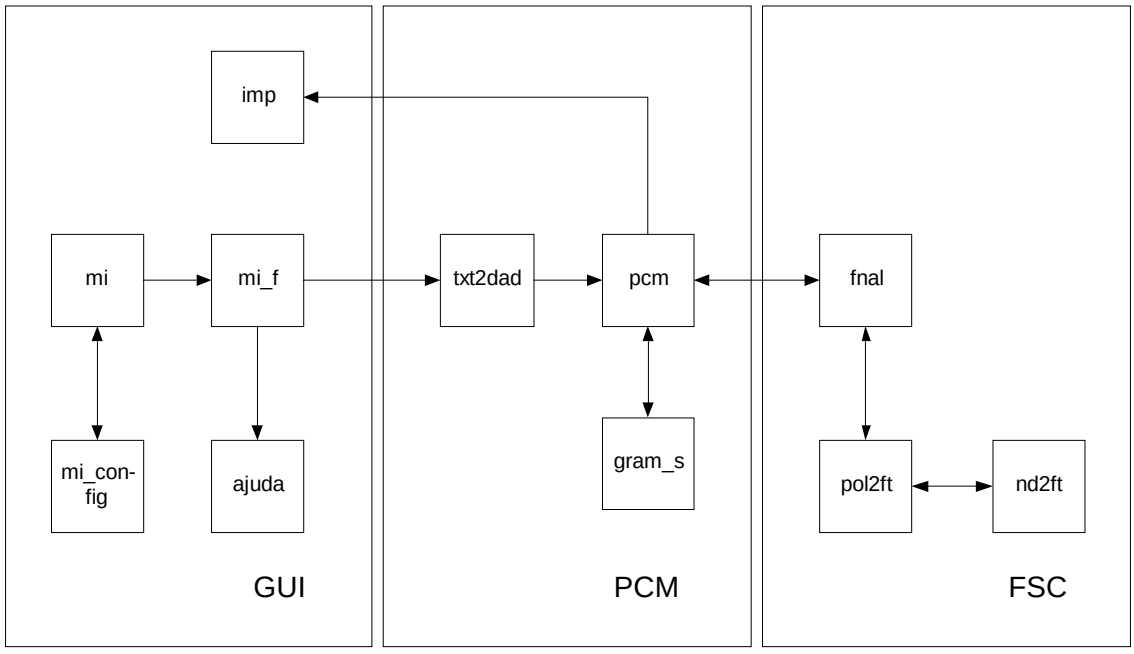


Figura 5.1: Diagrama de blocos de sub-rotinas do MI

As sub-rotinas da GUI foram as últimas a serem desenvolvidas, porém, são as primeiras a serem executadas. Nelas está estruturada a janela para entrada de dados e exibição de resultados, o arquivamento e a recuperação de projetos, e a ajuda ao usuário.

A parte do PCM é a segunda a entrar em ação, após os dados do projeto terem sido digitados pelo usuário, e ordenada a execução do PCM. A primeira ação dentro deste

bloco é converter os dados de entrada, que até este ponto têm o formato de seqüências de caracteres ("strings"), em dados numéricos ou fórmulas que vão ser utilizadas na estruturação do sistema de inequações, nas funções de transferência do sistema, nos valores iniciais dos parâmetros, e no número máximo de iterações. Com os dados de entrada já convertidos a um formato adequado, o PCM inicia o processo de busca da solução do problema tal como foi visto nas seções anteriores.

Para cada ponto de teste $\tilde{\mathbf{p}}^k$ que o PCM gera, é feita a consulta ao bloco FSC para obtenção dos valores dos funcionais do sistema de controle avaliados no ponto de teste. No bloco FSC são recebidos os valores do ponto de teste $\tilde{\mathbf{p}}^k$ e as funções de transferência da estrutura de controle na forma de polinômios $P_i(\mathbf{p},s)$. Os valores de $\tilde{\mathbf{p}}^k$ são substituídos nos polinômios $P_i(\mathbf{p},s)$, com o qual tem-se polinômios $P_i(s)$ onde todos os coeficientes são constantes. Utilizando os polinômios $P_i(s)$ previamente determinados, são obtidas as funções de transferência $W_y(s)$, $W_u(s)$ e $W_e(s)$. Estas funções de transferência são utilizadas para o cálculo das funções no tempo $y(t)$ e $u(t)$, e dos funcionais do sistema de controle.

Para calcular as funções no tempo $y(t)$ e $u(t)$ é utilizada a função *step* da *Control System Toolbox* [18]. Esta função gera, através de simulação, um conjunto de pontos representando o valor da função no domínio do tempo; trata-se portanto de uma representação aproximada em tempo discreto da função real contínua no tempo. O tempo de simulação e o intervalo de integração são escolhidos a fim de garantir resultados válidos com o mínimo esforço computacional, ou seja: escolha de um tempo de simulação que seja o menor possível mas que permita que a componente transitória do sinal de saída alcance uma ordem de grandeza desprezível; e escolha de um intervalo de integração que seja o maior possível mas que permita realizar uma representação dos transitórios razoavelmente aproximada das respostas do sistema contínuo.

Os funcionais do sistema de controle são calculados com ajuda da função *caracrd* da *Toolbox CACCON* [7]. Os valores calculados são devolvidos ao PCM para a continuação do processo.

O processo de busca no PCM termina somente quando todas as inequações são satisfeitas, ou quando o número de iterações atinge o valor máximo estabelecido pelo usuário.

Durante a execução do PCM, é continuamente enviada informação ao usuário através do GUI. Para cada ponto bem-sucedido $\mathbf{p}^k$, são apresentados os gráficos do sinal de saída do sistema $y(t)$ e do controlador $u(t)$; também são mostrados os valores dos funcionais $\mathbf{A}(\mathbf{p}^k)$ obtidos.

Ao final de cada execução do PCM é enviado ao *workspace* (área de memória accessível ao usuário) do *MATLAB* uma série de variáveis úteis ao projetista para análise e para criação de gráficos adicionais aos oferecidos na *toolbox*. As variáveis enviadas ao *workspace* são as seguintes:

- **A**: Vetor com os funcionais da última iteração com sucesso na mesma ordem que aparecem na janela do MI.
- **P**: Vetor com os polinômios das funções de transferência.
- **ph**: Matriz com todos os pontos testados pelo PCM, incluindo um índice adicional para indicar sucesso (*índice*=1)ou fracasso (*índice*=0).
- **pf**: Último ponto bem sucedido obtido pelo PCM.
- **t**: Vetor com valores do tempo dos gráficos da última iteração com sucesso.
- **u**: Vetor do sinal de controle no tempo.
- **y**: Vetor do sinal de saída do sistema no tempo.

5.2.1 Interface com o usuário (GUI)

A interface gráfica com o usuário (GUI) está composta das sub-rotinas: *mi.m*, *mi_config.m*, *mi_f.m*, *ajuda.m* e *imp.m*. Elas são o caminho de comunicação entre o projetista e o algoritmo do PCM, e constituem uma parte secundária do projeto, se comparada com o PCM ou FSC, mas desempenha um papel fundamental na formulação de cada projeto, pois facilita enormemente a entrada de dados e a visualização dos resultados, convertendo as tediosas repetições de tentativa e erro num processo rápido e dinâmico.

Na Figura 5.2 é mostrada a janela do GUI no momento de iniciar a *toolbox*, a qual chama-se de Método das Inequações (MI). Na parte superior esquerda da janela encontram-se seis botões de comando: **Abrir**, **Salvar**, **Salvar Como**, **Fechar**, **Parâmetros** e **Ajuda**, os quais são descritos a seguir:

Abrir: recupera sessões arquivadas no disco.

Salvar: atualiza o último arquivo de dados aberto com os dados apresentados na janela.

Salvar Como: arquiva os dados apresentados na janela num arquivo no disco.

Fechar: fecha a janela do MI.

Parâmetros: abre a janela que permite modificar parâmetros internos da *toolbox*.

Ajuda: abre janela de texto com ajuda e orientações ao projetista.

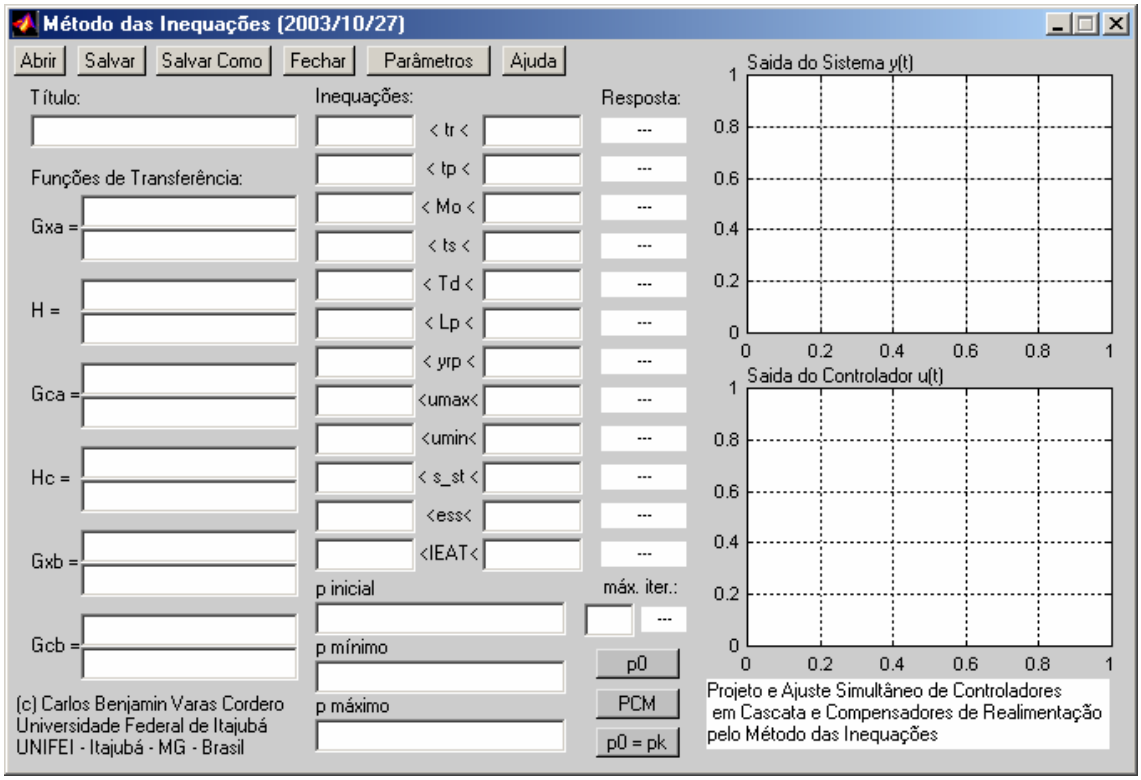


Figura 5.2: Janela do MI ao inicio

A janela **Parâmetros MI** ilustrada na Figura 5.3 permite ao projetista modificar os seguintes parâmetros internos do MI:

R_0: Amplitude do degrau de entrada.

R_N: Amplitude da excitação utilizada para o cálculo do erro estacionário.

α, β: Parâmetros internos do PCM.

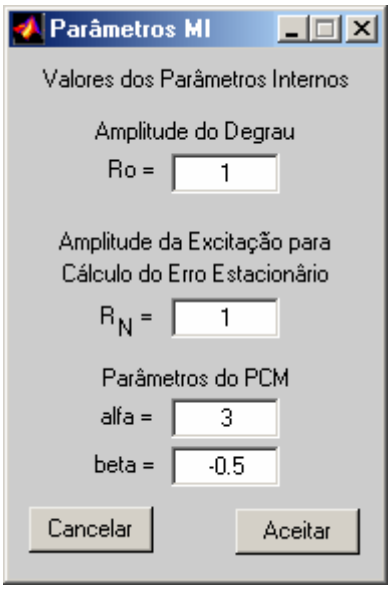


Figura 5.3: Janela Parâmetros MI

Na parte esquerda da janela principal são mostradas linhas para entrada dos dados do problema, e na parte direita são apresentados os resultados da execução.

Os dados de entrada são:

Título: título do projeto.

Funções de Transferência: A entrada das funções de transferência é feita em forma de polinômios, onde a função de transferência de cada bloco é definida pelos polinômios do numerador e do denominador. Os polinômios devem ser expressos em função de *s*, a freqüência complexa da transformada de Laplace; os parâmetros variáveis dos controladores devem ser expressos como elementos do vetor **p**. A Figura 5.4 ilustra uma entrada de dados típica de funções de transferência.

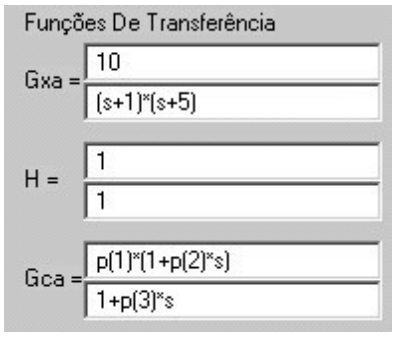


Figura 5.4: Exemplo de entrada de funções de transferência no MI

Na Figura 5.4 são observadas as seguintes funções de transferência:

$$G_{xa}(s) = \frac{P_1(s)}{P_2(s)} = \frac{10}{(s+1)(s+5)} \tag{5.1}$$

$$H(s) = \frac{P_3(s)}{P_4(s)} = \frac{1}{1} = 1 \tag{5.2}$$

$$G_{ca}(s) = \frac{P_5(s)}{P_6(s)} = \frac{p_1(1+p_2 s)}{1+p_3 s} \tag{5.3}$$

Nota-se que o formato de entrada de dados dos polinômios é muito flexível e familiar aos usuários de programas de cálculo matemático.

Inequações: No campo de inequações, devem ser introduzidos os limites inferiores e superiores para os valores dos índices de desempenho, deixando-se em branco os campos dos limites que não serão especificados. Podem ser definidos limites máximos e mínimos para cada um dos doze funcionais que são considerados na *toolbox*; estes limites serão os valores C_i do conjunto de inequações, e definirão os contornos das regiões admissíveis S_i. Cabe ressaltar que somente serão criadas inequações no caso do campo conter algum valor. Os campos em branco não são levados em conta no momento de se estruturar o conjunto de inequações. A Figura 5.5 ilustra um exemplo de como são introduzidos os limites para as inequações.

Figura 5.5: Exemplo de entrada de limites de inequações no MI

No exemplo são especificados tempo de subida menor que 1 seg., ultrapassagem máxima menor que 0,01 e tempo de estabilização menor que 3 seg.

p inicial, p mínimo, p máximo: Nestes campos indicam-se o valor inicial dos parâmetros variáveis (**p**) e os seus limites máximo e mínimo. Os campos devem-se preencher com vetores coluna, segundo a notação MATLAB, ou seja entre colchetes e separados cada um com ponto e vírgula. O número de elementos do vetor deve ser igual ao número de parâmetros variáveis, p_i, definidos nas funções de transferência.

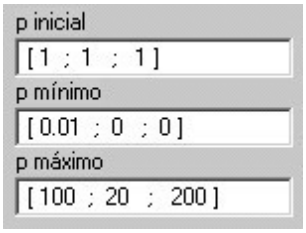


Figura 5.6: Exemplo de entrada do vetor inicial $\mathbf{p}^0$ e seus limites máximo e mínimo

A Figura 5.6 ilustra um exemplo para um problema de três parâmetros variáveis onde:

$$\mathbf{p}^0 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \tag{5.4}$$

$$0.01 \leq p_1 \leq 100 \tag{5.5}$$

$$0 \leq p_2 \leq 20 \tag{5.6}$$

$$0 \leq p_3 \leq 200 \tag{5.7}$$

Máx. iter.: O dado "número máximo de iterações" é necessário para interromper o PCM em problemas para os quais não existe uma região viável *S*, ou do algoritmo ter falhado na tarefa de encontrar um ponto da região. Na prática tem-se observado que é bom escolher valores pequenos (menor que 5) para as primeiras experiências, podendo-se aumentar o número máximo de iterações na medida que os índices de desempenho forem se aproximando das faixas estabelecidas pelas inequações.

Os três botões na parte inferior da janela, **p0**, **PCM** e **p0 = pk**, executam as seguintes funções:

p0:	Calcula os funcionais e gráficos de $y(t)$ e $u(t)$ no ponto inicial $\mathbf{p}^0$.
PCM:	Executa o PCM até obter uma solução ou até atingir o número máximo de iterações.
p0 = pk:	Copia os valores do último ponto com sucesso achado na execução anterior do PCM, na janela de entrada do ponto inicial.

No lado direito da janela são mostrados os resultados do projeto: exibidos os valores dos funcionais **A** na coluna resposta, os gráficos de resposta do sistema $y(t)$, e do sinal de controle $u(t)$, e o valor do vetor $\mathbf{p}^k$. Resultados parciais são apresentados durante a execução do PCM: cada vez que o algoritmo encontra um novo ponto $\mathbf{p}^k$, os valores de resposta e os gráficos vão sendo atualizados, assim oferecendo ao projetista uma visão dinâmica da evolução dos índices de desempenho em relação aos limites estabelecidos nas inequações. Adicionalmente, o número da iteração também é mostrado, para acompanhamento do avanço da execução do algoritmo. Junto com o valor do vetor $\mathbf{p}^k$ também é mostrado o tipo *N* do sistema.

Na Figura 5.7 é mostrada a janela do MI ao final da execução de um exemplo; observa-se que o PCM foi executado com sucesso, todos os valores de resposta encontram-se dentro dos limites estabelecidos, o número de iterações realizadas foi de 16, e a mensagem *"Processo terminado com sucesso"* indica que foi encontrado um ponto dentro da região admissível *S*.

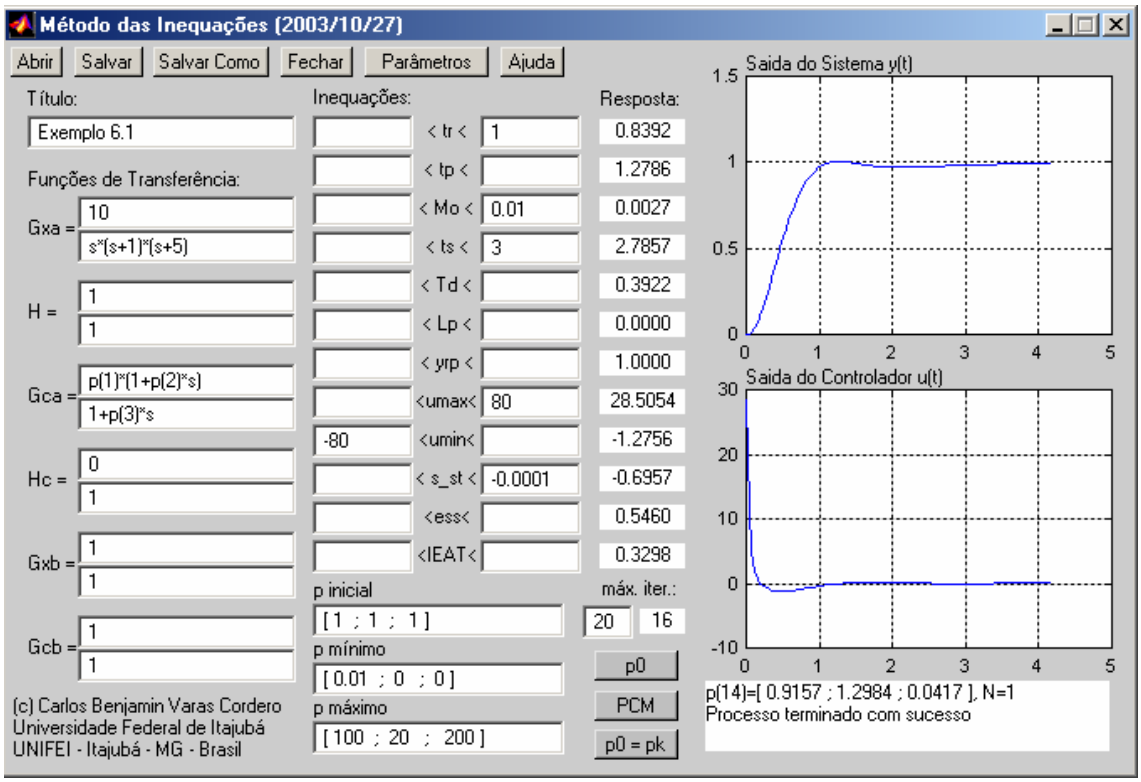


Figura 5.7: Janela do MI após da execução do PCM

5.2.2 O processo dos contornos móveis (PCM)

O PCM é composto das seguintes sub-rotinas: *pcm.m*, *gram_s.m*, e *tex2dad.m*. Em primeiro lugar a sub-rotina *tex2dad.m* capta os dados provenientes do bloco GUI e estrutura-os para utilização no algoritmo do PCM. A informação de entrada recebida do GUI consiste em seqüências de caracteres, as quais podem representar textos, equações, variáveis, ou valores numéricos. Utilizando funções de conversão de texto e da *toolbox* de matemática simbólica [23], os dados são convertidos ao formato que o PCM utiliza. Após a conversão tem-se como resultados os dados de entrada seguintes:

P = polinômios das funções de transferência (alfanumérico)

S = conjunto de inequações (alfanumérico)

$\mathbf{p}^0$ = vetor do ponto inicial (numérico)

$iter_{max}$ = número máximo de iterações (numérico)

P é um vetor de doze elementos, que são a representação alfanumérica dos polinômios do numerador e do denominador das seis funções de transferência que definem a estrutura do sistema.

S é uma matriz onde cada fila representa uma inequação; o número de filas de **S** é variável, dependendo do número de restrições definidas pelo projetista. Além das inequações relativas às características dinâmicas do sistema, são incrementadas duas inequações por cada parâmetro variável definido, as quais são utilizadas para definir os limites mínimos e máximos de cada parâmetro.

P, **S**, $\mathbf{p}^0$ e $iter_{\max}$ são os valores de entrada da sub-rotina *pcm.m*. O comando, antes de iniciar realmente o PCM, faz uma primeira avaliação do sistema no ponto inicial $\mathbf{p}^0$; se o ponto estiver fora da região admissível *S*, então o PCM, como foi descrito na seção 3.2, é iniciado. O processo irá obtendo novos pontos iterativamente, utilizando o algoritmo proposto por Rosenbrock [21], descrito na seção 3.3; os pontos tenderão a se aproximar cada vez mais de *S*, até que seja encontrado um ponto dentro de *S*, ou até que o número de iterações alcance o limite estabelecido na variável $iter_{\max}$. Nesse momento, termina a execução do PCM, e os resultados do último ponto permanecem na janela de saída.

A ortogonalização de vetores requerida para a geração dos pontos de teste é feita em uma sub-rotina especial chamada *gram_s.m*, na qual é implementado o algoritmo que cria uma base de vetores ortonormais para um conjunto dado de *n* vetores no espaço R^n.

Os valores dos funcionais do sistema não são calculados dentro do bloco PCM, e nem as saídas do sistema e do controlador. O PCM simplesmente chama o bloco FSC, fornecendo como dado de entrada o valor do ponto de teste $\tilde{\mathbf{p}}^k$, e os polinômios das funções de transferência do sistema **P**, e recebendo como resposta os valores dos funcionais **A** e a resposta no tempo do sinal de saída do sistema $y(t)$, e do sinal de controle $u(t)$.

Durante a execução do PCM também são feitas chamadas ao bloco GUI. Cada vez que um ponto de teste é bem-sucedido, os valores e gráficos de resposta do sistema são

atualizados na janela do GUI. Para melhor compreensão do funcionamento do algoritmo do PCM, na Figura 5.8 é apresentado o seu fluxograma simplificado.

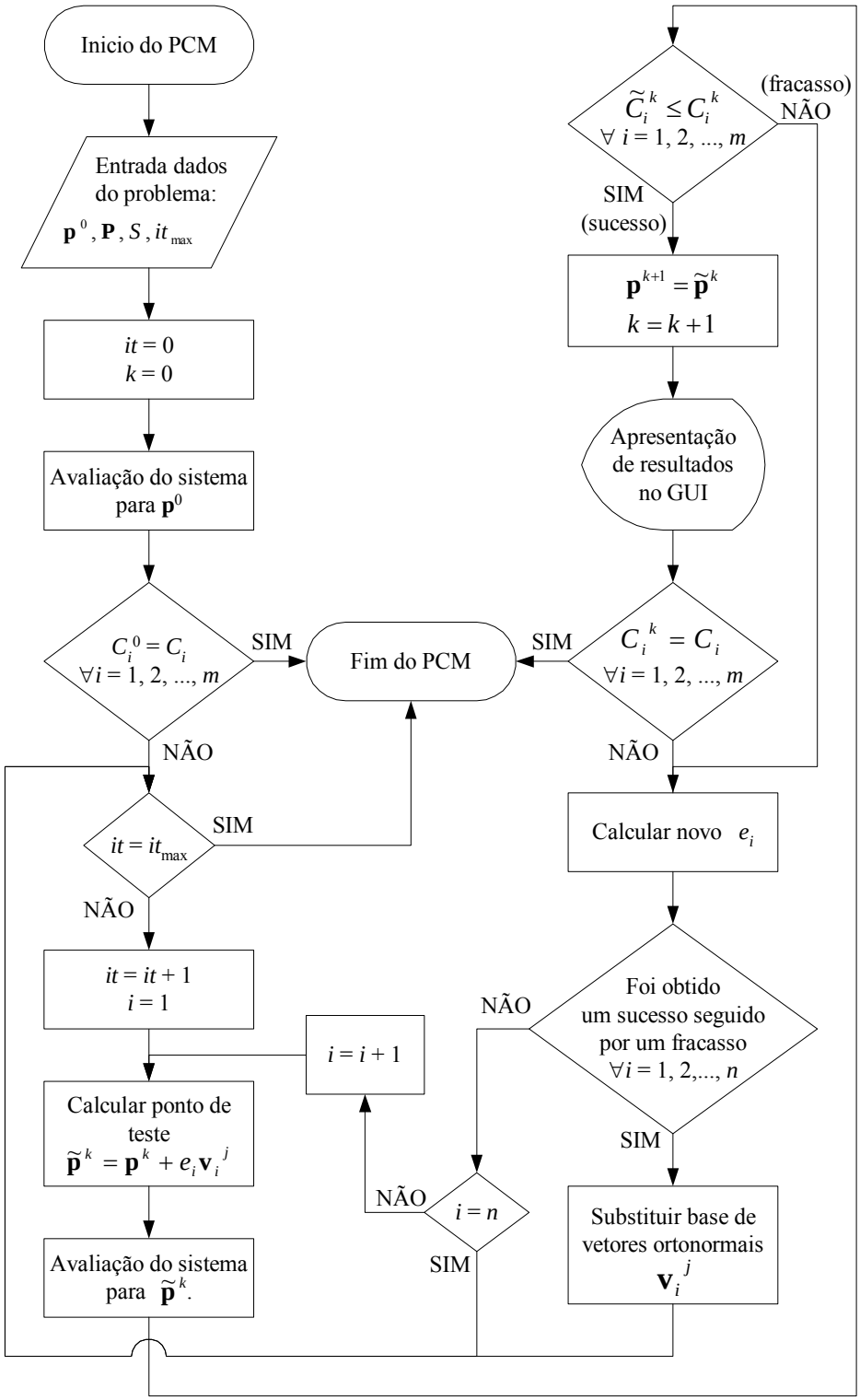


Figura 5.8: Fluxograma simplificado do PCM

5.2.3 Funcionais do sistema de controle (FSC)

O bloco FSC está composto das sub-rotinas: *fnal.m*, *pol2ft.m*, *nd2ft.m*. A sub-rotina *pol2ft.m* recebe os dados do ponto de teste $\tilde{\mathbf{p}}^k$, e os polinômios das funções de transferência do sistema $\mathbf{P}(s,\mathbf{p})$. Substituindo-se os valores de $\tilde{\mathbf{p}}^k$ nos polinômios $\mathbf{P}(s,\mathbf{p})$, obtém-se um conjunto de polinômios $\mathbf{P}(s)$, com coeficientes constantes. Com os polinômios $\mathbf{P}(s)$ são calculadas as funções de transferência $W_y(s)$, $W_u(s)$ e $W_e(s)$, utilizando-se as Eqs. (4.2), (4.3) e (4.4). As funções de transferência são devolvidas na sub-rotina *fnal.m*, para o cálculo dos funcionais.

A sub-rotina *nd2ft.m* é um complemento da *pol2ft.m* que faz o cancelamento de pólos e zeros nas funções de transferência $W_y(s)$, $W_u(s)$ e $W_e(s)$, nos casos de igualdade.

Na sub-rotina *fnal.m* é efetuada uma comprovação prévia de que o sistema de controle é estável, ou seja, de que $W_y(s)$ é própria (Ineq.(2.23)) e todos os seus pólos têm parte real negativa (Ineq. (2.24)). Por causa desta comprovação, o funcional $A_{10}(\mathbf{p})$ (Eq. (4.16)), que é a abscissa de estabilidade, é calculado antes dos outros funcionais.

Se o sistema for estável, a sub-rotina *fnal.m* continua com o cálculo dos outros funcionais do sistema (Eqs. (4.7) a (4.20)). No caso do sistema ser instável, a sub-rotina termina e devolve ao PCM unicamente o valor da abscissa de estabilidade.

Para o cálculo dos funcionais é preciso gerar a resposta no tempo das funções de transferência $W_y(s)$ e $W_u(s)$. Para isto é utilizada a função *step* da *Control System Toolbox* [18]; esta função gera, através de simulação, um conjunto de pontos que representam $y(t)$ e $u(t)$; o tempo de simulação (T_{sim}) e o intervalo de integração (h) são calculados através das seguintes equações:

$$\sigma_s = \max\left\{ \mathrm{Re}(s) \,\middle|\, \frac{1}{W_y(s)} = 0 \right\} \qquad \textbf{(5.8)}$$

onde $\sigma_s < 0$ é a *abscissa de estabilidade* do sistema

$$T_{sim} = \frac{10}{|\sigma_s|} \tag{5.9}$$

$$\sigma_{\min} = \min\left\{ \mathrm{Re}(s) \left| \frac{1}{W_y(s)} = 0 \right. \right\} \tag{5.10}$$

$$T_{\min,r} = \frac{1}{|\sigma_{\min}|} \tag{5.11}$$

$$\omega_{\max} = \max\left\{ \mathrm{Im}(s) \left| \frac{1}{W_y(s)} = 0 \right. \right\} \tag{5.12}$$

$$T_{\min} = \frac{2\pi}{\omega_{\max}} \tag{5.13}$$

$$h = \max\left\{ \min\left\{ \frac{T_{\min,r}}{10}, \frac{T_{\min}}{60} \right\}, \frac{T_{sim}}{1000} \right\} \tag{5.14}$$

Os valores de T_{sim} e h calculados vão garantir uma simulação com resultados válidos e um mínimo de esforço computacional.

As CRD's são obtidas através da analise da saída $y(t)$, feita pela função *caracrd* da *Toolbox CACCON* [7].

Para o cálculo da *IEAT* é utilizada a função *trapz* do *MATLAB* [24] a qual realiza uma integração numérica através do método trapezoidal.

Uma vez calculados os funcionais **A** e as respostas no tempo de $W_y(s)$ e $W_u(s)$, todos estes dados são retornados ao PCM para continuar o processo.

CAPÍTULO 6

EXEMPLOS DE APLICAÇÃO

Nos capítulos anteriores foi apresentado o MI aplicado ao projeto de sistemas de controle e a sua implementação computacional. No capítulo presente são apresentados vários exemplos de projeto e ajuste de controladores em cascata e em retroação, utilizando a *toolbox* desenvolvida. Os exemplos foram executados em um computador pessoal com processador Intel Celeron de 800Mhz.

6.1 Controlador em Cascata

O primeiro exemplo mostrado na Figura 6.1, é um sistema básico de controle com realimentação unitária. Este exemplo tem sido investigado por vários autores, como, por exemplo, D'Azzo e Houpis [13], Zakian e Al-Naib [26], Coelho [10], e Bejarano [3].

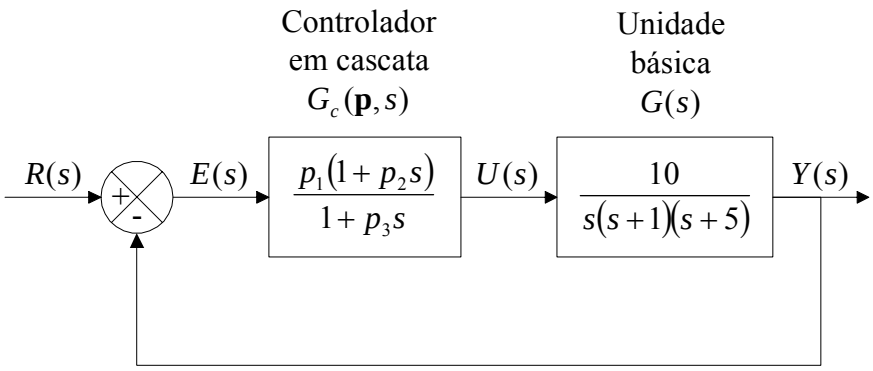


Figura 6.1: Exemplo 1 - Sistema de controle com controlador em cascata

Outros detalhes do projeto são:

$$\text{Especificações de projeto} \begin{cases} t_r \le 1\ seg \\ M_0 \le 0.01\,(1\%) \\ t_s \le 3\ seg \\ -80 \le u(t) \le 80 \\ 0.01 \le p_1 \le 100 \\ 0 \le p_2 \le 20 \\ 0 \le p_3 \le 100 \end{cases} \quad \textbf{(6.1)}$$

O ponto de partida para o PCM é o seguinte:

$$\mathbf{p}^0 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \quad \textbf{(6.2)}$$

Para este ponto o sistema é estável, mas a resposta ao degrau é lenta e muito oscilatória.

Para construir a estrutura de controle do exemplo, a qual é mais simples do que a estrutura utilizada na *toolbox* MI, alguns blocos devem ser eliminados; isto é conseguido com funções de transferência adequadas para os vários blocos do DB padrão da Figura 4.1. O DB padrão, adaptado a este exemplo, é mostrado na Figura 6.2.

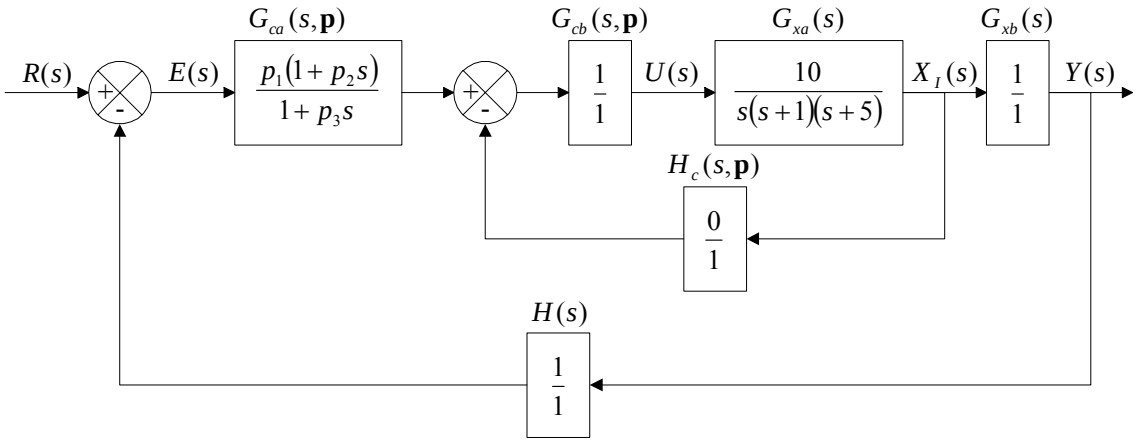


Figura 6.2: Exemplo 1 - Diagrama de blocos padronizado para o MI

Na Figura 6.3 é mostrada a entrada de dados feita na janela do MI e o resultado final. Note-se que o número de iterações selecionado é 20.

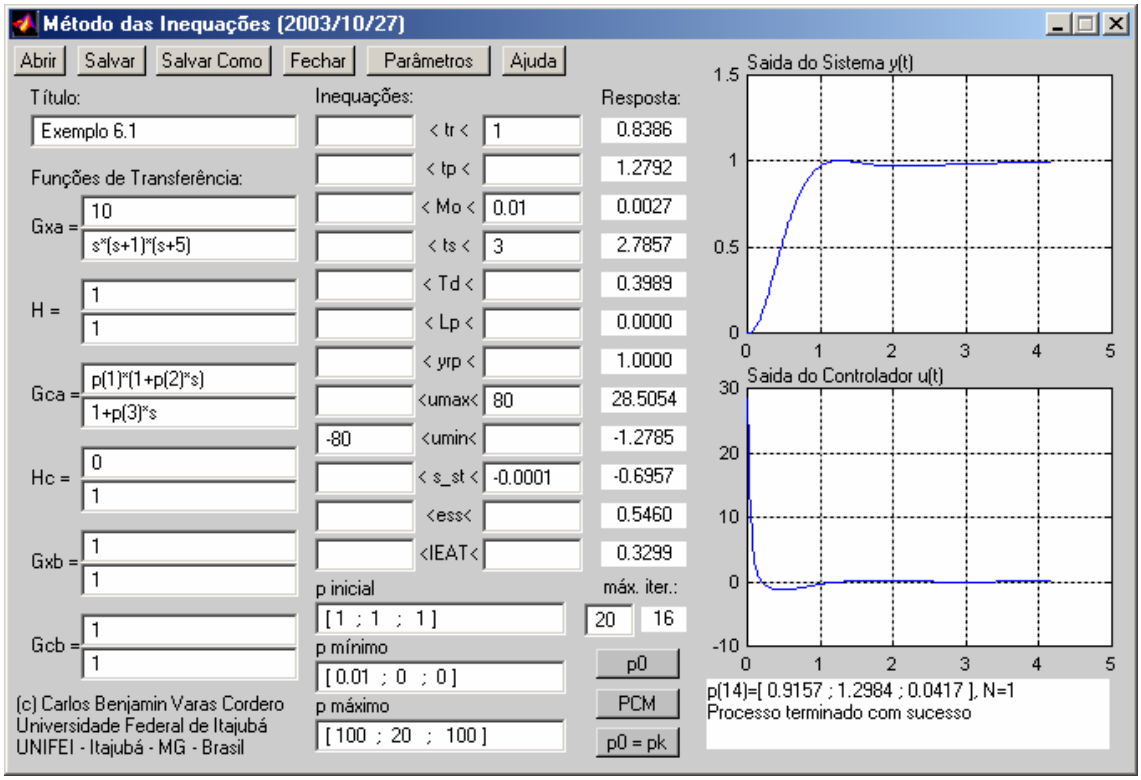


Figura 6.3: Exemplo 1 - Janela do MI após a primeira execução do PCM

Uma vez introduzidos todos os dados pressiona-se o botão **PCM** para dar inicio ao processo iterativo. Pode-se observar que após 16 iterações o computador encontra a seguinte solução para o problema:

$$\mathbf{p}^{14} = \begin{bmatrix} 0.9157 \\ 1.2984 \\ 0.0417 \end{bmatrix} \tag{6.3}$$

$$\text{Funcionais da resposta} \begin{cases} t_r = 0.8386\ seg \\ M_0 = 0.0027\ (0.27\%) \\ t_s = 2.7857\ seg \\ u_{\max} = 28.5054 \\ u_{\min} = -1.2785 \end{cases} \tag{6.4}$$

O superíndice 14 indica o número de pontos bem-sucedidos, durante a execução do PCM.

O passo seguinte no processo de projeto é encontrar uma solução ótima para o problema, dentro do conjunto de soluções possíveis *S*. Portanto, o PCM é executado novamente, desta vez tomando como ponto de partida o ponto da solução anterior (Eq. (6.3)) e adicionando a seguinte inequação:

$$IEAT \leq 0 \tag{6.5}$$

A Ineq. (6.5) expressa uma condição impossível de satisfazer, para sistemas reais; o PCM tentará, sem sucesso, encontrar um ponto $\mathbf{p}^k$ que atenda esta especificação de projeto além das anteriores. No processo o PCM buscará pontos que reduzam o valor do *IEAT*, tentando anulá-lo, o que é impossível. Cada tentativa $\tilde{\mathbf{p}}^k$ só é aceita como um novo ponto $\mathbf{p}^k$ quando é conseguida uma nova redução do *IEAT*. Quando o número de iterações chegar ao seu limite máximo, e ficar claro para o projetista que iterações adicionais não levarão a novas reduções do *IEAT*, considera-se que o último valor obtido para o *IEAT* é o mínimo possível, e que o ponto $\mathbf{p}^k$ correspondente é o ponto ótimo.

Na Figura 6.4 é mostrada a nova execução do MI. Foi escolhido um número máximo de iterações suficientemente grande para levar o sistema a um ponto onde qualquer redução no *IEAT* é desprezível.

A resposta encontrada nesta nova execução é:

$$\mathbf{p}^{64} = \begin{bmatrix} 0.9909 \\ 1.1155 \\ 0.0138 \end{bmatrix} \tag{6.6}$$

$$\text{Funcionais da resposta} \begin{cases} t_r = 0.8729\ seg \\ M_0 = 0.0085\ (0.85\%) \\ t_s = 1.0942\ seg \\ u_{\max} = 80.0000 \\ u_{\min} = -1.0168 \\ IEAT = 0.1980 \end{cases} \tag{6.7}$$

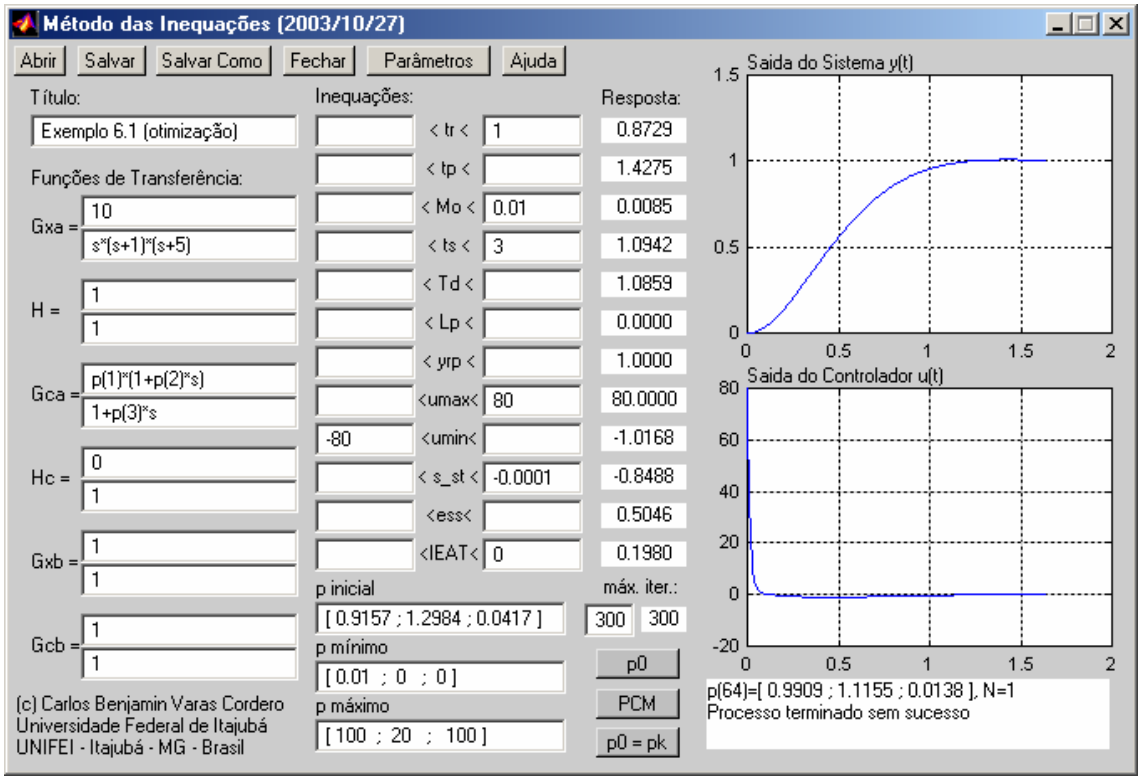


Figura 6.4: Exemplo 1 - Janela do MI após otimização

O ponto encontrado (Eq. (6.6)) é um ponto que satisfaz as especificações de projeto originais (Ineqs. (6.1)) e que é ótimo com base no *IEAT*.

Na Tabela 6.1 são apresentadas as especificações de projeto e os resultados obtidos para os pontos $\mathbf{p}^0$, $\mathbf{p}^{14}$ e $\mathbf{p}^{64}$. Na Figura 6.5 podem ser comparados os gráficos de $y(t,\mathbf{p}^k)$ versus t, para os três casos. Observa-se que as melhores características são as da resposta $y(t,\mathbf{p}^{64})$, tal como era esperado.

Tabela 6.1: Exemplo 1 - Especificações e resultados obtidos

Especificações	$\mathbf{p}^0$	$\mathbf{p}^{14}$	$\mathbf{P}^{64}$
$0.01 \le p_1 \le 100$	1	0.9157	0.9909
$0 \le p_2 \le 20$	1	1.2984	1.1155
$0 \le p_3 \le 100$	1	0.0417	0.0138
$t_r \le 1$	1.428	0.839	0.873
$M_0 \le 0.01$	0.486	0.0027	0.0085
$t_s \le 3$	12.71	2.786	1.094
$-80 \le u_{\min}$	-0.486	-1.28	-1.02
$u_{\max} \le 80$	1	28.51	80.00
IEAT	7.8417	0.3299	0.1980
No. Iterações	0	16	300

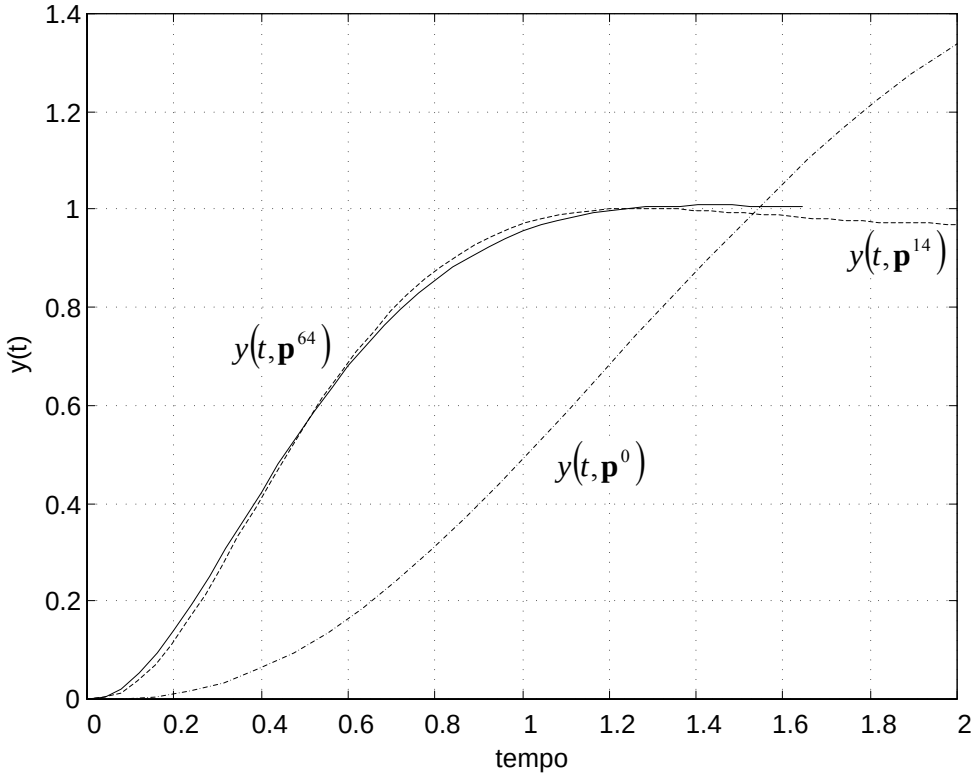


Figura 6.5: Exemplo 1 – Gráficos de resposta no tempo

6.2 Controladores em Cascata e Realimentação Tacométrica

Utilizando a mesma unidade básica do exemplo da Figura 6.1 e controlador em cascata combinado com realimentação tacométrica, tem-se o sistema de controle mostrado na Figura 6.6.

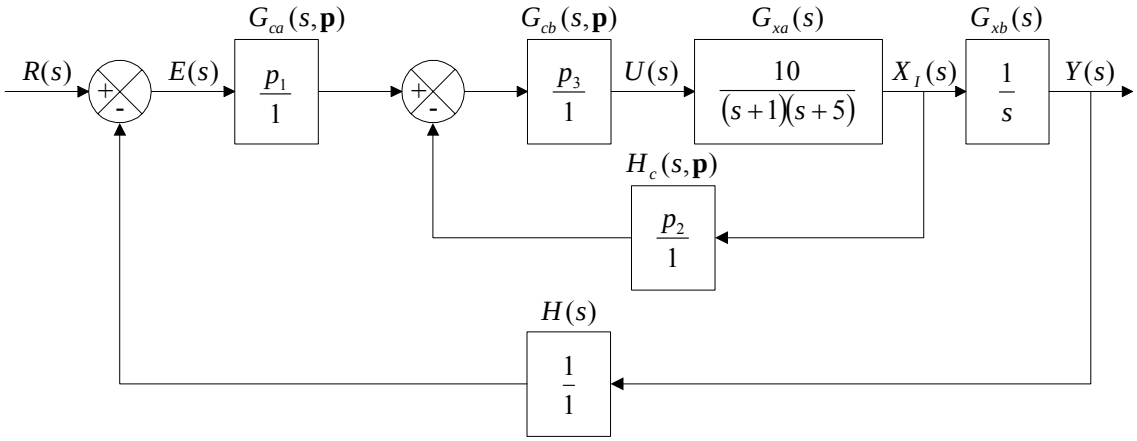


Figura 6.6: Exemplo 2 – Diagrama de blocos padronizado para o MI

As especificações de projeto e o ponto inicial são os mesmos do exemplo anterior, com a diferença de que os valores dos parâmetros variáveis p_i devem estar entre 0 e 100. Na Figura 6.7 são mostradas a entrada de dados feita na janela do MI e os resultados da primeira execução.

$$\text{Especificações de projeto} \begin{cases} t_r \le 1\ seg \\ M_0 \le 0.01\ (1\%) \\ t_s \le 3\ seg \\ -80 \le u(t) \le 80 \\ 0 \le p_i \le 100,\ para\ i = 1,2,3 \end{cases} \quad \textbf{(6.8)}$$

$$\text{Ponto de partida: } \mathbf{p}^0 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \quad \textbf{(6.9)}$$

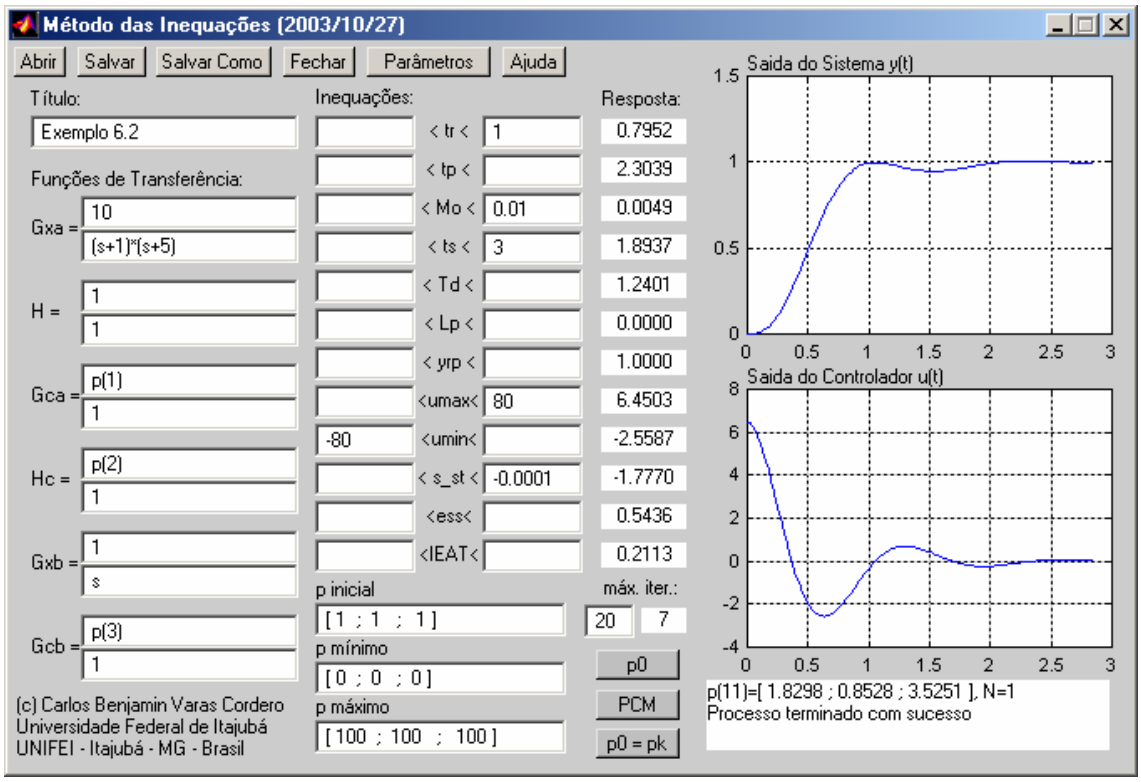


Figura 6.7: Exemplo 2 - Janela do MI após primeira execução do PCM

Utilizando o ponto encontrado na primeira execução, o processo é executado novamente incluindo a restrição para otimização, como é mostrado na Figura 6.8.

Na Tabela 6.2 são resumidas as especificações de projeto e os resultados obtidos para os pontos $\mathbf{p}^0$, $\mathbf{p}^{11}$ e $\mathbf{p}^{109}$. Na Figura 6.9 podem ser comparados os gráficos de $y(t,\mathbf{p}^k)$ versus t, para os três casos. Observa-se no gráfico que $y(t,\mathbf{p}^{109})$ apresenta uma componente oscilatória mais acentuada que $y(t,\mathbf{p}^{11})$, mas o *IEAT* é menor, e portanto o ponto $\mathbf{p}^{109}$ é ótimo com relação ao *IEAT* respeitando as restrições de projeto, o que não significa que o projetista seja obrigado a acatar este resultado: ele pode optar pelo ponto $\mathbf{p}^{11}$, para o qual a parcela oscilatória da resposta é bem menos acentuada.

Comparando os exemplos 1 e 2 observa-se que o *IEAT* cai de 0,1980 para 0,1229 com a utilização do compensador em realimentação, mantendo-se os funcionais restantes dentro das faixas estabelecidas.

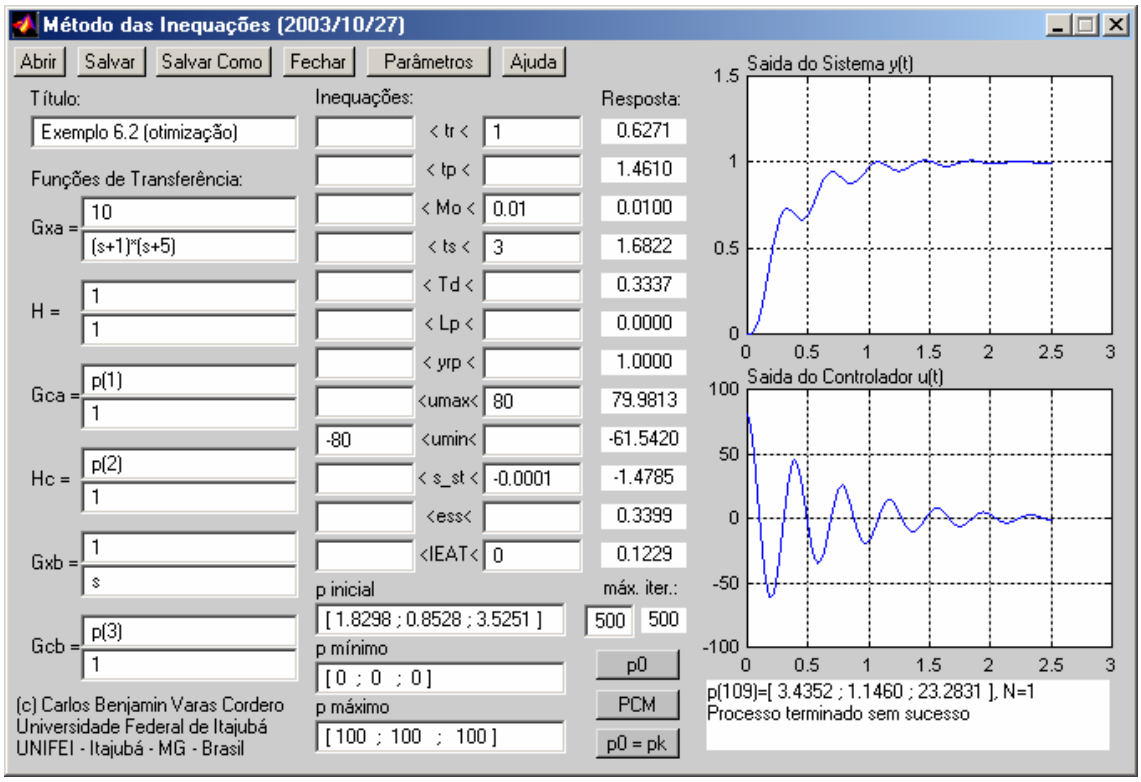


Figura 6.8: Exemplo 2 - Janela do MI após otimização

Tabela 6.2: Exemplo 2 - Especificações e resultados obtidos

Especificações	$\mathbf{p}^0$	$\mathbf{p}^{11}$	$\mathbf{P}^{109}$
$0 \le p_1 \le 100$	1	1.83	3.435
$0 \le p_2 \le 100$	1	0.853	1.146
$0 \le p_3 \le 100$	1	3.53	23.28
$t_r \le 1$	2.81	0.80	0.63
$M_0 \le 0.01$	0	0.0049	0.01
$t_s \le 3$	4.42	1.89	1.68
$-80 \le u_{\min}$	-0.017	-2.56	-61.54
$u_{\max} \le 80$	1	6.45	79.98
IEAT	1.649	0.211	0.1229
No. Iterações	0	7	500

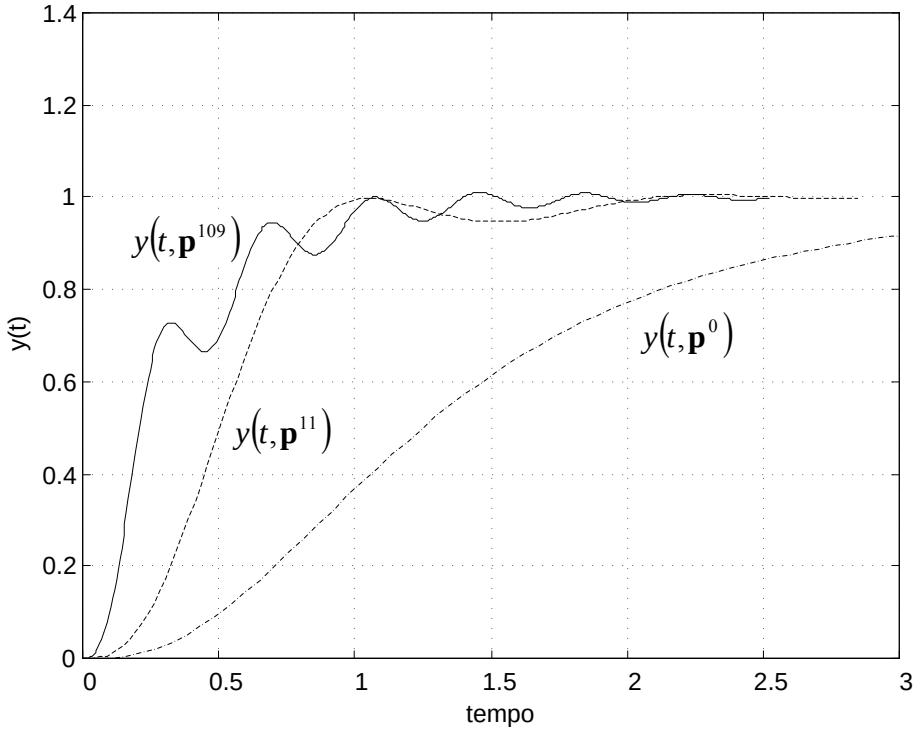


Figura 6.9: Exemplo 2 – Gráficos de resposta no tempo

6.3 Otimização de um Sistema de Segunda Ordem

Nesta seção o programa do MI é aplicado para otimizar um sistema de segunda ordem com a seguinte função de transferência global:

$$\frac{Y(s)}{R(s)} = \frac{1}{s^2 + 2\zeta s + 1} \tag{6.10}$$

O diagrama de blocos padrão da *toolbox* MI que resulta nesta função de transferência é o mostrado na Figura 6.10. Demonstra-se a seguir que realmente esse diagrama tem a FT global dada pela Eq. (6.10).

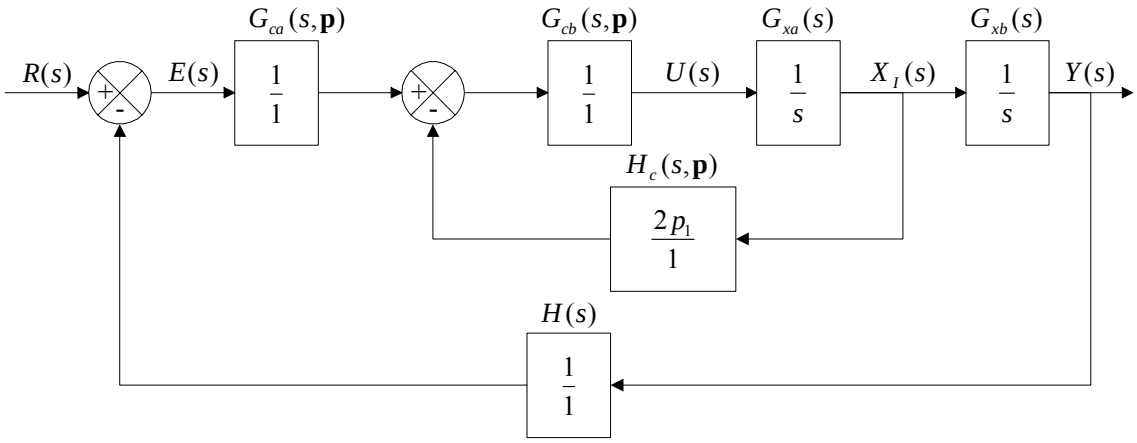


Figura 6.10: Exemplo 3 – Diagrama de blocos padrão do MI de um sistema de segunda ordem.

No diagrama de blocos da Figura 6.10 tem-se:

$$\frac{Y(s)}{E(s)} = G(s) = 1\frac{1\cdot\dfrac{1}{s}}{1+1\cdot\dfrac{1}{s}\cdot 2p_1}\frac{1}{s} = \frac{1}{s^2+2p_1 s} \tag{6.11}$$

Então a função de transferência global será

$$\frac{Y(s)}{R(s)} = \frac{G(s)}{1+G(s)} = \frac{\dfrac{1}{s^2+2p_1 s}}{1+\dfrac{1}{s^2+2p_1 s}} = \frac{1}{s^2+2p_1 s+1} \tag{6.12}$$

Comparando-se as Eqs. (6.10) e (6.12) vê-se que

$$p_1 = \zeta \tag{6.13}$$

Este sistema tem então um único parâmetro a ajustar, ou seja, p_1. Sabe-se que este sistema é estável para $\zeta>0$. Para que somente a função de otimização seja ativada para a execução do PCM, são especificadas apenas as seguintes inequações:

$$\text{Especificações de projeto}\begin{cases} M_o \leq 0{,}99\ (99\%) \\ IEAT \leq 0\ \text{(otimização)} \end{cases} \qquad \textbf{(6.14)}$$

A especificação $M_o \leq 0{,}99$ é uma especificação bastante folgada (frouxa), e praticamente não terá nenhum efeito na execução do PCM, em sua tentativa de zerar o *IEAT*, o que, na realidade levará à sua minimização, o que, por si só, garante uma ultrapassagem muito abaixo do limite especificado de 99 %, que, diga-se de passagem, seria um resultado inaceitável. Para estas duas inequações, o PCM foi executado para dois pontos iniciais diferentes: $p_1^0 = 0{,}01$ e $p_1^0 = 0{,}707$, com um limite de 200 iterações. Os resultados são mostrados nas Figuras 6.11 e 6.12, respectivamente. Note-se que em ambos os casos obteve-se $p_1 = \zeta = 0{,}7556$, com $IEAT = 1.9492$ e $M_o = 0{,}0267$ (2,67%). Alerta-se o leitor que em problemas mais complexos o MI geralmente leva a resultados diferentes para pontos iniciais diferentes do vetor de parâmetros **p**. Neste caso específico o resultado final foi o mesmo porque o sistema considerado neste exemplo possui um único parâmetro ajustável (ζ) e o *IEAT*, como função de ζ, é uma função perfeitamente convexa, ou seja, com um único mínimo. O objetivo deste exemplo, que é bastante utilizado na literatura [13] [20], foi mostrar que, com a inclusão da inequação $IEAT \leq 0$, o MI também cumpre a tarefa de fazer o ajuste ótimo do sistema.

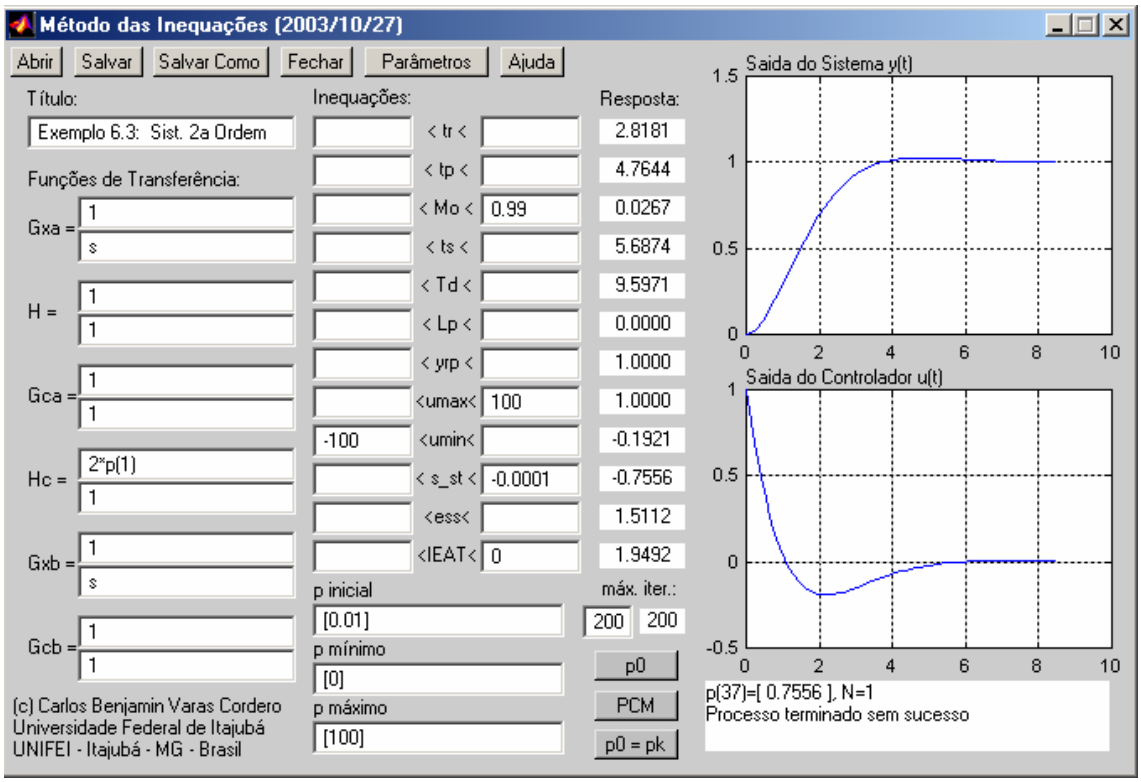


Figura 6.11: Exemplo 3 – Otimização do sistema de 2ª ordem para $p_1^0 = \zeta^0 = 0{,}01$.

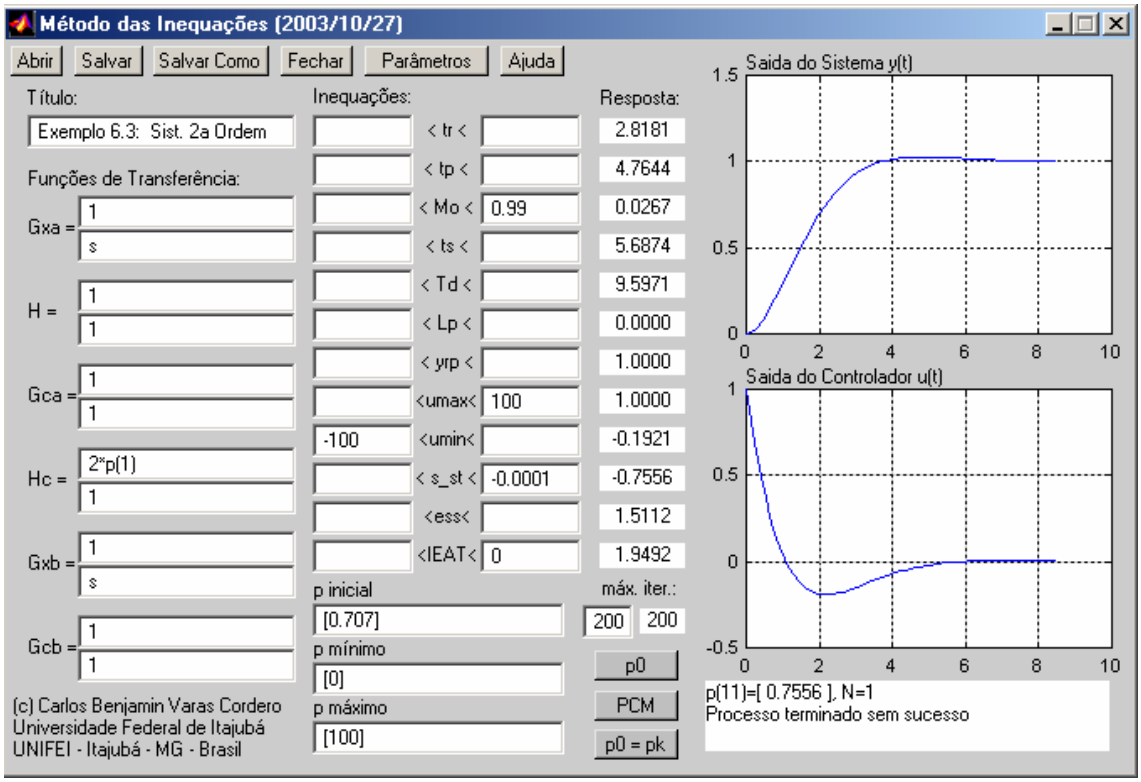


Figura 6.12: Exemplo 3 – Otimização do sistema de 2ª ordem para $p_1^0 = \zeta^0 = 0{,}707$.

6.4 Projeto de Controlador para um Sistema Esfera-Viga (Ball-Beam)

Nesta seção a *toolbox* do MI é aplicada para projetar o controlador de um sistema Esfera-Viga (*Ball-Beam*) muito utilizado em laboratórios de controle [19]. O sistema consiste de uma esfera que pode rodar livremente ao longo de uma viga, como é mostrado na Figura 6.13. Através de um servo-motor é controlado o ângulo de inclinação da viga. O objetivo do problema é projetar um controlador capaz de manipular a posição da esfera na viga.

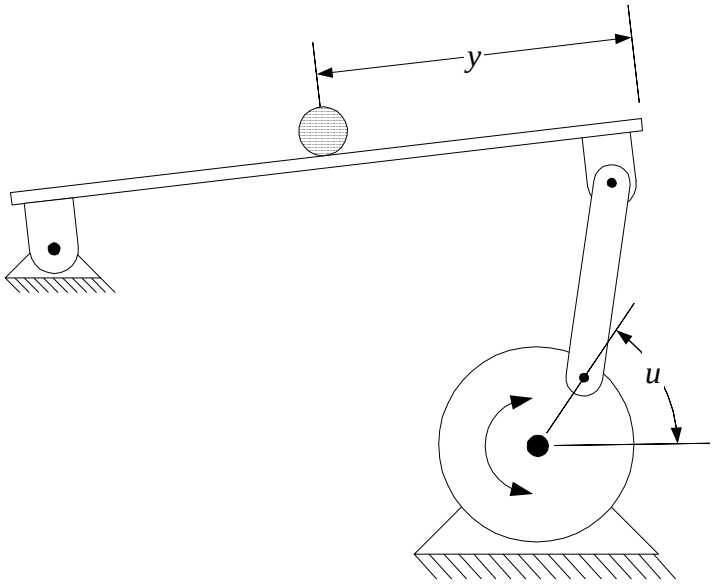


Figura 6.13: Exemplo 4 – Esquema simplificado do projeto *Ball-Beam*

A função de transferência do sistema que relaciona o ângulo do servo-motor (u) com a posição da esfera (y) é a seguinte:

$$\frac{Y(s)}{U(s)} = \frac{0.21}{s^2} \tag{6.15}$$

Pode-se observar que se trata de um sistema de tipo dois, o que caracteriza um sistema instável.

Deseja-se que o sistema controlado atinja as seguintes especificações:

$$\text{Especificações de projeto} \begin{cases} M_o \leq 0{,}05\ (5\%) \\ t_s \leq 3seg \\ u_{\max} \leq 20 \end{cases} \qquad \textbf{(6.16)}$$

Por tratar-se de um sistema tipo dois é conveniente utilizar um compensador na realimentação da primeira derivada do sinal de saída (*y*), que representa a velocidade da esfera; com isto se reduz o tipo do sistema, tornando-o mais estável. Em conjunto com o compensador de realimentação é considerado um controlador proporcional em cascata, como se observa na Figura 6.14.

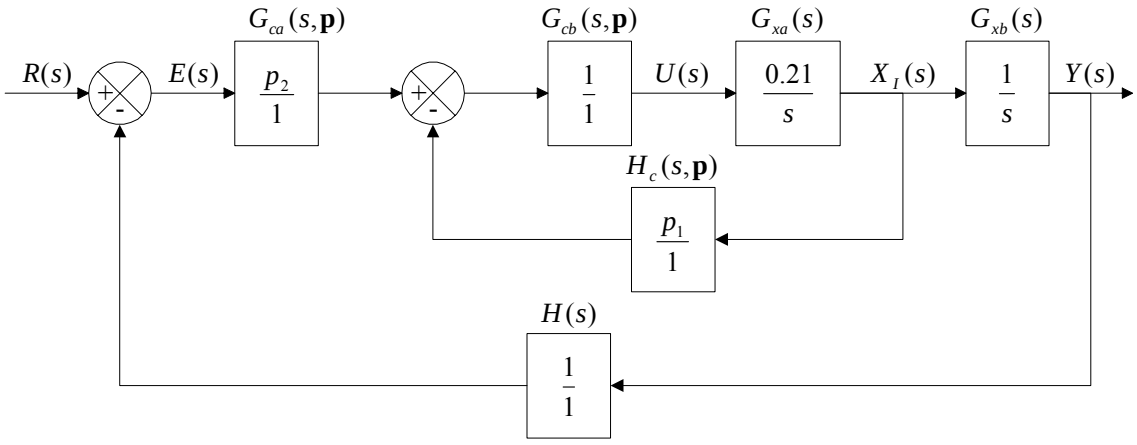


Figura 6.14: Exemplo 4 – Diagrama de Blocos do Sistema *Ball-Beam*

Para este sistema a *toolbox* MI foi executada com o ponto inicial: $\mathbf{p}^0 = [1;1]$ e número máximo de iterações de 100. Os resultados são mostrados na Figura 6.15. Note-se que o PCM levou 68 iterações para encontrar um ponto admissível $\mathbf{p}^{45} = [9.929;7.8927]$. Posteriormente é executado o MI tomando como ponto de partida o ponto admissível encontrado na execução anterior e acrescentando-se a inequação $IEAT \leq 0$ para otimizar o sistema: O número máximo de iterações é igual a 200 para assegurar que a resposta final do PCM seja mais próxima ao ótimo real. Os resultados do processo iterativo de otimização são mostrados na Figura 6.16.

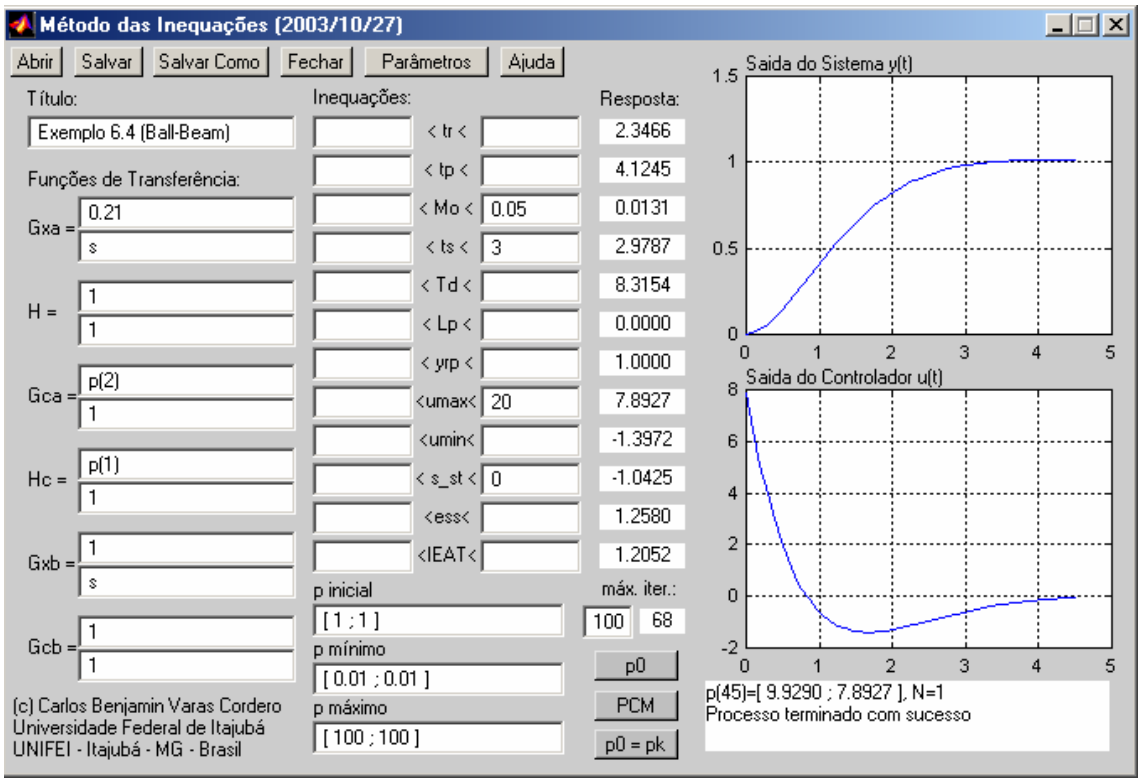


Figura 6.15: Exemplo 4 – Primeira execução do MI

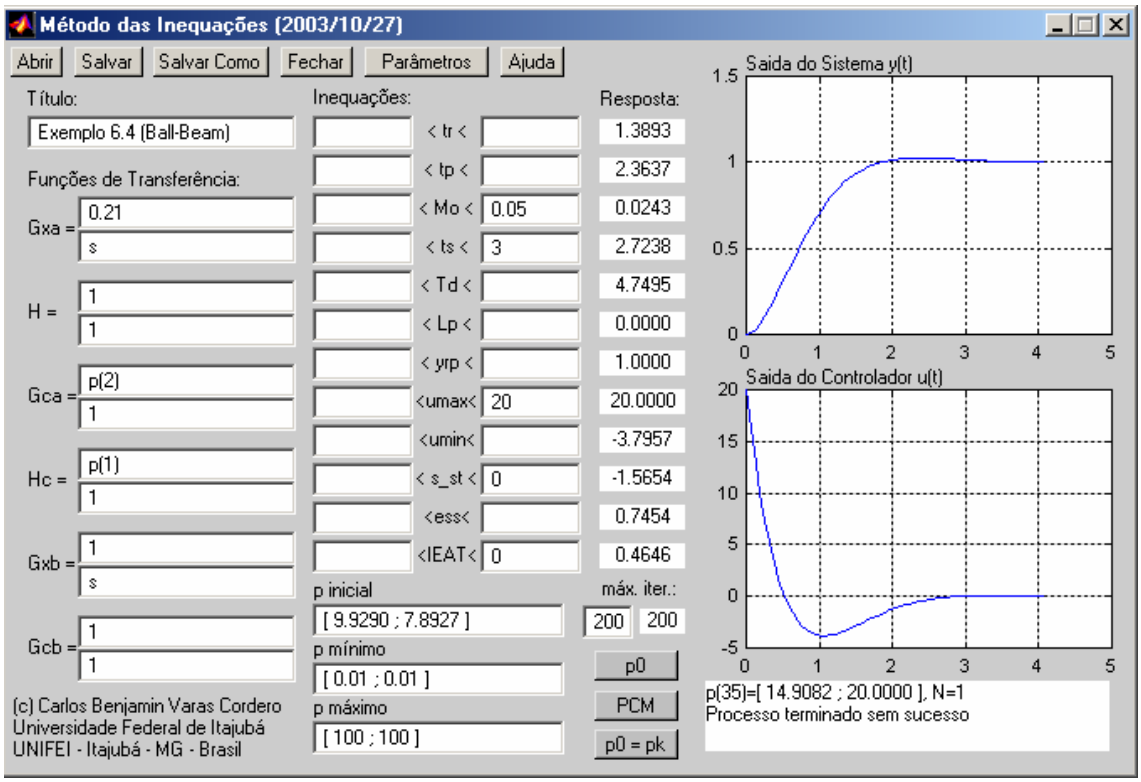


Figura 6.16: Exemplo 4 – Segunda execução do MI (otimização)

Com a finalidade de mostrar graficamente a execução do PCM, para este exemplo foram elaborados gráficos de curvas de nível dos funcionais considerados ($M_0, t_s, u_{\max}, IEAT$) em função dos parâmetros variáveis (p_1, p_2), tal como pode ser observado nas Figuras 6.17 a 6.20. Nos gráficos observa-se em preto a trajetória de sucessos conseguida pelo PCM, e em vermelho os pontos tentativas que corresponderam a fracassos.

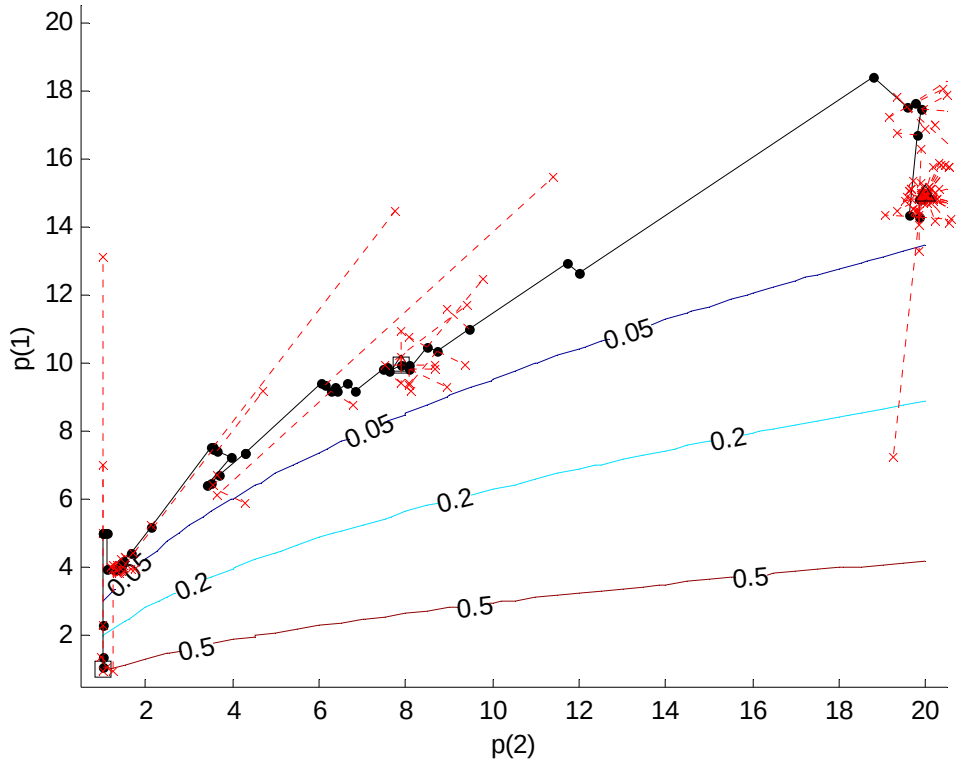


Figura 6.17: Exemplo 4 – Curvas de Nível do M_0 vs. p

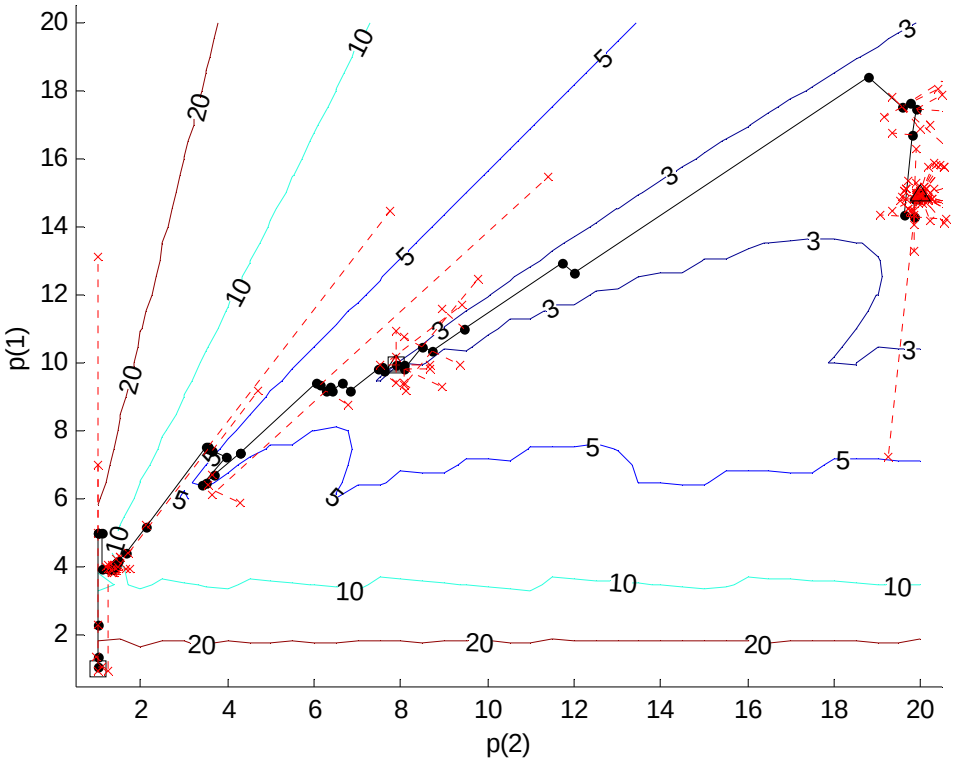


Figura 6.18: Exemplo 4 – Curvas de Nível do t_s vs. p

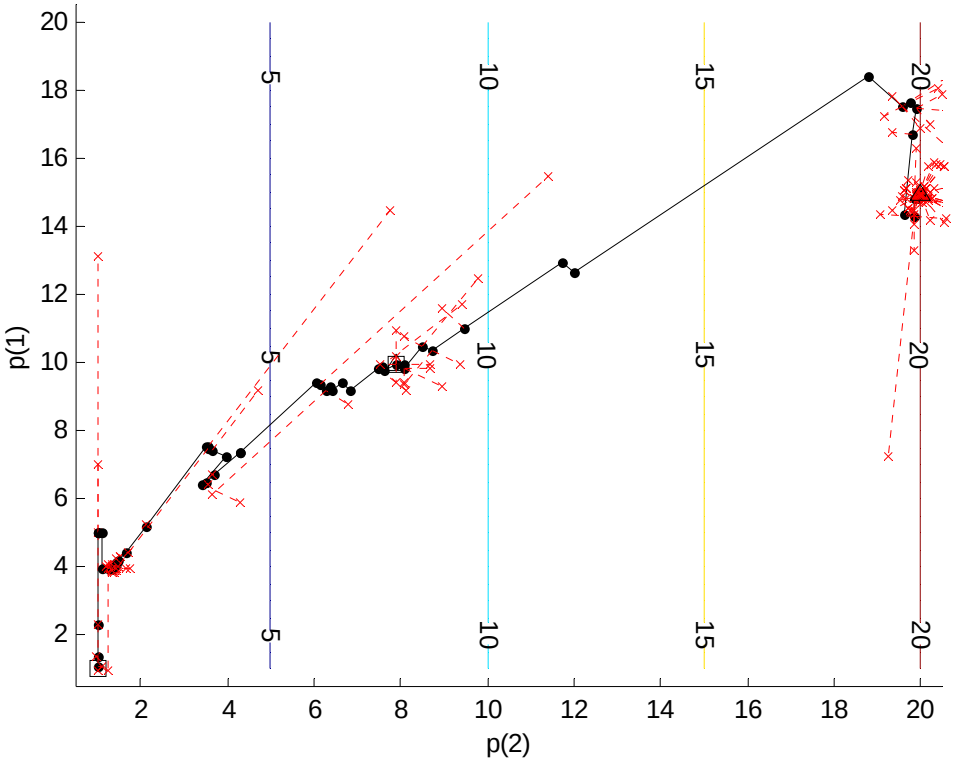


Figura 6.19: Exemplo 4 – Curvas de Nível de u_{max} vs. p

As Figuras 6.17 a 6.19 mostram as duas execuções do PCM. Como pode ser observado na primeira execução, o PCM conseguiu satisfazer as especificações de M_0, t_s e u_{max}, começando em $\mathbf{p}^0 = [1;1]$ e terminando no ponto $\mathbf{p}^{45} = [9.9;7.9]$; a segunda execução inicia-se no ponto final da execução anterior, ou seja $\mathbf{p}^0 = [9.9;7.9]$, para o qual as especificações do projeto já são cumpridas, e termina no ponto $\mathbf{p}^{35} = [14.9;20]$ no qual ocorre a mínima *IEAT* que satisfaz as restrições iniciais do projeto.

A Figura 6.20 mostra a trajetória da execução de otimização do PCM sobreposta ao gráfico de curvas de nível do *IEAT*. Observa-se claramente como o PCM foi eficaz ao encontrar o mínimo *IEAT* (0,46) dentro da região admissível *S* (em cinza) definida pelas especificações iniciais do projeto (M_0, t_s e u_{max}).

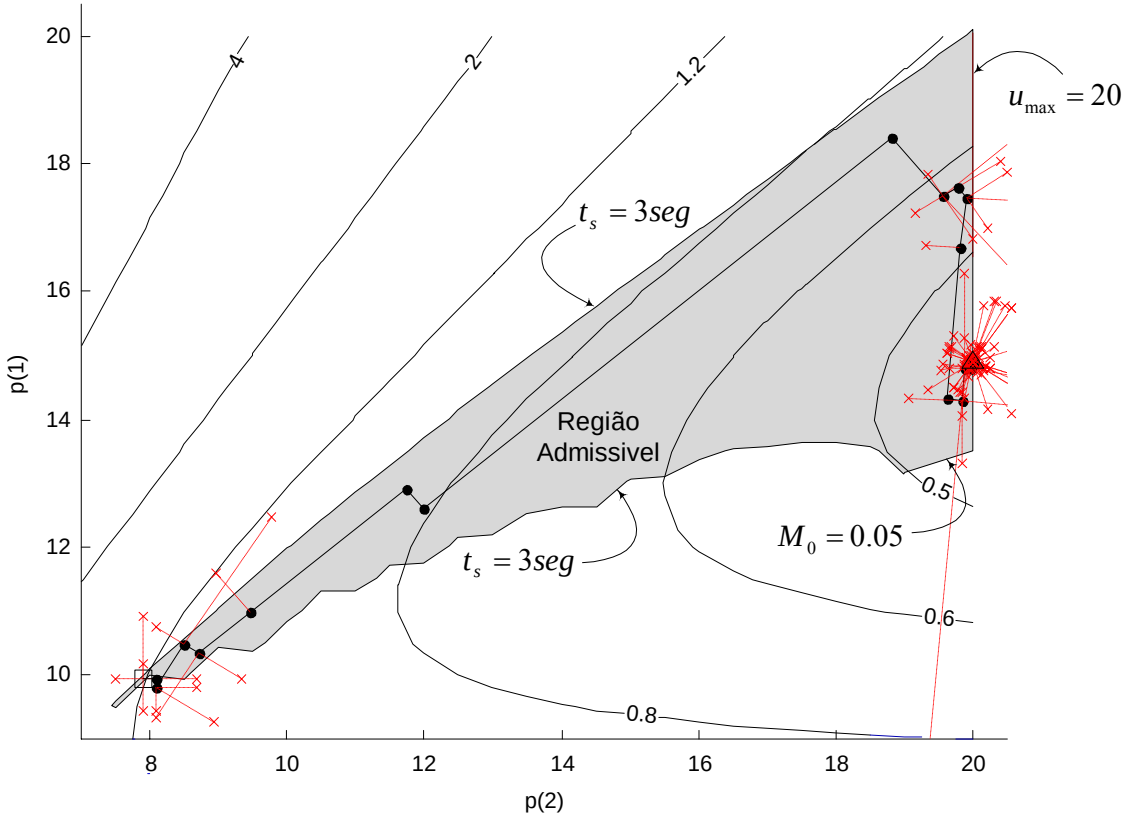


Figura 6.20: Exemplo 4 – Curvas de Nível de *IEAT* vs. P

CAPÍTULO 7

CONCLUSÕES

7.1 Trabalho Desenvolvido

O trabalho apresentado tratou do desenvolvimento de um conjunto de rotinas e comandos computacionais para projeto e ajuste de controladores de sistemas dinâmicos pelo Método das Inequações (MI), voltado para a estrutura convencional de sistemas de controle realimentados, para controladores em cascata e compensadores de realimentação. Esses comandos e rotinas foram desenvolvidos para execução dentro do ambiente computacional MATLAB, mas todos eles são ativados a partir da janela de interface gráfica chamada "Método das Inequações", a partir da qual são feitas a entrada de dados do sistema e todas as demais operações de projeto e ajuste dos controladores e de análise do sistema global. Dentro deste contexto, o conjunto de comandos e rotinas desenvolvido, como parte deste trabalho, será denominado simplesmente *toolbox*.

A base para o desenvolvimento deste trabalho foi a dissertação de mestrado de Bejarano [3]. Deve ser ressaltado que nesta nova versão a *toolbox* desenvolvida utiliza a interface gráfica com o usuário do *Matlab*, que tem a vantagem de ser muito amigável com o projetista, o que evidentemente é uma facilidade para que eventuais interessados no MI venham a aplicá-lo de forma generalizada.

7.2 Aspectos Relevantes

O ponto central deste trabalho é a adoção da estrutura padrão de sistemas de controle mostrada na Figura 2.13, na qual o controlador possui dois controladores em cascata e mais um compensador de realimentação, o que é uma evolução bastante considerável em relação às versões anteriores dos programas de projeto pelo MI para sistemas de controle univariáveis [3] [10] [26]. Nessas versões o controlador possuía unicamente um controlador em cascata, além do detector de erro. A estrutura de controlador para a qual foi desenvolvida este trabalho é bem mais completa, e pode portanto ser aplicada a

um espectro bem mais amplo de problemas de projeto de sistemas de controle. O caso de sistemas com um único controlador em cascata passa a ser um caso particular da estrutura completa aqui considerada. Note-se que o primeiro exemplo de aplicação (seção 6.1) foi o mesmo utilizado em trabalhos anteriores do MI [3] [10] [26], com resultados semelhantes.

Uma outra evolução importante foi a possibilidade de se manter alguns dos parâmetros com valores numéricos fixos. Isto pode ser importante, para o caso de se desejar incluir no controlador filtros passa-baixas com constantes de tempo ou freqüências de corte fixas, ou de se adotar valores fixos para outros parâmetros, com base em alguma outra técnica de ajuste, deixando a cargo do MI o ajuste dos demais parâmetros. Isto representa uma flexibilidade adicional em relação às versões anteriores, além de uma redução de esforço computacional, já que naquelas versões todos os parâmetros teriam de ser ajustados pelo MI, o que, de certa forma, cerceava a liberdade do projetista para fixar alguns dos parâmetros com base em outras técnicas de ajuste ou na sua própria experiência.

Outra contribuição importante é o desenvolvimento de uma janela de *interface* gráfica com o usuário, para entrada de dados, acompanhamento da execução do PCM, inclusive com a visualização dos gráficos da resposta no tempo da saída do sistema e do sinal de controle, à medida que novos pontos bem sucedidos do vetor de parâmetros vão sendo obtidos, e de saída de dados. A *toolbox* utiliza os recursos da *Symbolic Math Toolbox* [23], para que a entrada de dados das funções de transferência de cada bloco do sistema de controle (blocos da unidade básica e do controlador) seja feita em forma simbólica, como se o usuário estivesse escrevendo expressões matemáticas, quando são então definidos quais parâmetros do controlador são fixos e quais são os variáveis (ver os exemplos do Capítulo 6). Os dados de entrada também devem incluir limites para os parâmetros variáveis, e fica a critério do usuário a especificação de um limite inferior e um superior para cada um dos funcionais usados para as especificações de desempenho e as restrições de funcionamento. Quando um determinado limite C_i é omitido, o comando apropriado da *toolbox* exclui do PCM a respectiva inequação, o que dá à *interface* gráfica um aspecto bastante interessante, que poderia até ser considerado como uma característica de inteligência artificial.

Outra melhora da *toolbox* atual é a inclusão do funcional *IEAT*, o qual permite fazer a otimização de um controlador, obtido com execuções anteriores do PCM. A otimização é ativada quando é especificado o limite superior para o *IEAT* igual a zero, ou seja, quando é incluida a inequação $IEAT \leq 0$, a qual é insolúvel, já que, pela sua própria definição, $IEAT > 0$. Então, na busca de zerar ou tornar o *IEAT* negativo, o PCM naturalmente buscará um vetor de parâmetros que o minimize, sujeito às restrições estabelecidas pelas demais inequações. Deve ser observado que a otimização em si dará melhores resultados quando as restrições estabelecidas pelas demais inequações forem mais flexíveis (folgadas), no sentido de que os demais funcionais sejam mantidos com facilidade dentro dos limites estabelecidos.

A *toolbox* de projeto mostrou-se eficiente na determinação de valores viáveis e ótimos para os parâmetros, nos casos examinados. Inclusive, quando o ponto inicial escolhido é um ponto para o qual o sistema é instável, o MI pode encontrar um ponto que o estabilize, e a partir deste busca outros pontos para os quais as demais inequações sejam satisfeitas.

Uma deficiência encontrada no MI é a de que a resposta encontrada é muito sensível ao ponto inicial escolhido. Devido ao fato dos funcionais serem na sua maioria funções multi-modais, nas quais existem mínimos locais, a região de busca da solução simultânea das inequações pode ficar confinada na vizinhança de mínimos locais de alguns dos funcionais que fazem parte do sistema de inequações. Fazer várias tentativas com pontos iniciais diferentes é uma forma de compensar essa deficiência do MI. Por exemplo, numa primeira etapa poderiam ser utilizadas técnicas convencionais de ajuste e projeto de sistemas de controle (lugar das raízes, resposta em freqüência, etc.) para se encontrar um ponto inicial que proporcione ao MI boas possibilidades de ser bem-sucedido.

A *toolbox* desenvolvida para projeto e ajuste de controladores pelo MI contempla o problema básico de projeto de sistemas de controle univariáveis com controladores combinando controle em cascata e compensação por realimentação, com base em especificações da resposta a degrau. Vários problemas práticos estão enquadrados neste problema básico, e a disponibilidade de uma *toolbox* que resolva este problema diretamente no domínio do tempo é, sem dúvida nenhuma, um recurso que poderá

agilizar bastante os trabalhos de projeto de sistemas de controle, principalmente quando comparado aos métodos convencionais de ajuste e projeto, tais como lugar das raízes, resposta em freqüência, alocação de pólos e outros.

7.3 Sugestões para Trabalhos Futuros

Futuros trabalhos poderão ser direcionados para ajuste de controladores de sistemas que não estão enquadrados na estrutura padrão para a qual foi direcionado este trabalho, tais como sistemas multivariáveis, não lineares, sistemas de controle digital, e para conjuntos quaisquer de sinais de entrada (referências e distúrbios), dentro das propostas apresentadas na Nova Formulação para o Método das Inequações [27] e de Projeto de Sistemas de Controle Robusto pelo Método das Inequações [10].

Uma forma de desenvolver uma *toolbox* com essas finalidades poderia ser através da criação de uma interface entre o MI e o *Simulink*. No *Simulink* seria modelado o sistema de controle com ajuda dos blocos de funções existentes nessa *toolbox*. Em cada iteração do PCM o sistema seria simulado no *Simulink* em modo lote (*batch*), o que pode ser feito com o comando ***sim*** (também do *Simulink*), com os valores de parâmetros determinados no PCM. Com a simulação do sistema feita, seriam calculados os funcionais respectivos, e que seriam então retornados ao PCM para a continuação do processo. Para sistemas de grande porte, cuja dimensão inviabilize a representação através de modelos *Simulink*, existe a possibilidade de modelagem através de outros comandos *MATLAB* (*M-files*), que possam ser combinados com o comando de implementação do PCM, para estender a idéia de projeto e ajuste de sistemas com base nas premissas que constituem a base do Método das Inequações.

Outros métodos de busca e otimização podem ser utilizados em conjunto com o PCM para melhorar o desempenho da *toolbox*, a fim de reduzir ou eliminar a dependência do resultado em relação ao ponto inicial escolhido. Uma alternativa viável é a utilização de algoritmos genéticos [16] [17], que produzam um mapeamento inicial da região viável, e escolham como ponto inicial para execução do PCM o ponto que possua o melhor desempenho à luz das especificações do projeto. Desta forma, a indicação do ponto de início do PCM seria opcional para o projetista, podendo deixar que o respectivo comando escolha o melhor ponto encontrado pelo algorítmo genético. Inclusive

poder-se-ia implementar uma opção para realizar um mapeamento inicial da região de busca em função dos funcionais de interesse, e apresentar o resultado em forma de gráfico de colinas, isto sempre que o espaço de busca esteja limitado a duas dimensões.

A *toolbox* atual ou suas futuras versões podem ser melhoradas levando em conta as seguintes considerações:

- Oferecer opções de especificação de outros índices de desempenho, que poderiam ser os relativos ao domínio da frequência, ou ao lugar das raizes da FT de malha fechada, como por exemplo margem de ganho, margem de fase, coeficiente de amortecimento, etc.
- Dar ao usuário a opção de interromper o processo num ponto qualquer para fazer ajustes nos dados de entrada e recomeçar ou continuar a busca, fazendo uso dos novos valores.
- Desenvolver sub-rotinas para a validação dos dados de entrada, antes do PCM ser iniciado, afim de detectar possíveis erros de digitação ou de dados do modelo.

Finalmente, é apropriado reproduzir aqui as sugestões de Bejarano [3], de que a filosofia do Método das Inequações, e a sua aplicação, podem ser estendidas para muito além da área de sistemas de controle, pois qualquer problema que possa ser formulado em termos de um conjunto de inequações, a ser resolvido através de ajuste de parâmetros, pode ser resolvido pelo MI. As adaptações para outras áreas consistirão na identificação e separação adequada das partes do sistema com estruturas e valores de parâmetros fixos, e daquelas cujos parâmetros possam ser modificados, ou que possam receber novos componentes também ajustáveis. O outro passo importante é a definição das grandezas e índices de desempenho sujeitos a limites, das formas de calculá-los, e de seus limites, o que naturalmente levará à definição das inequações a serem satisfeitas através do ajuste de parâmetros. Cumpridas estas etapas, o problema estará pronto para ser resolvido pelo MI, com a utilização do Processo dos Contornos Móveis, ou de outro algoritmo para solução numérica de sistemas de inequações. Acredita-se que, sem grande esforço, é possível imaginar e identificar uma grande variedade de problemas de engenharia, bem como de outras áreas das Ciências Exatas, que possam ser formulados em termos de conjuntos de inequações.

BIBLIOGRAFIA

[1] **APOSTOL, T.M.**: *Mathematical Analysis,* 2ª edição, Addison-Wesley Publishing Co., Reading, MA, EUA, 1974.

[2] **BACH, R.E.Jun.**: *A Practical Approach to Control System Optimization*, Proceedings of the IFAC Tokio Symposium on Systems Engineering for Control-System Design, Japan, pp. 129-135, 1965.

[3] **BEJARANO, J.C.**: *Projeto e Ajuste Computacional de Controladores de Sistemas Dinâmicos pelo Método das Inequações*, Dissertação de Mestrado, Escola Federal de Engenharia de Itajubá, Itajubá, MG, Brasil, Dezembro de 2000.

[4] **CHEN,C.T.**: *Analysis and Synthesis of Linear Control Systems*, Holt, Rinehart and Wilson, Inc., New York, EUA, 1975.

[5] **CLOSE, M.C., & FREDERICK, D.K.**: *Modeling and Analysis of Dynamic Systems*, Houghton Mifflin, Boston, EUA, 1978.

[6] **COELHO, C.A.D.**: *Análise e Simulação de Sistemas Dinâmicos (apostila)*, 1ª edição, EFEI – Escola Federal de Engenharia de Itajubá, Itajubá, MG, Brasil, Dezembro de 1996.

[7] **COELHO, C.A.D.**: *CACCON Control Toolbox for the MATLAB Program (Version 3.2) – User's Manual*, EFEI – Escola Federal de Engenharia de Itajubá, Itajubá, MG, Outubro de 1997.

[8] **COELHO, C.A.D.**: *Compensation of the Speed Governor of a Water Turbine by the Method of Inequalities*, ASME Journal of Dynamic Systems, Measurement and Control, Vol. 101, No. 3, pp.205-211, Setembro de 1979.

[9] **COELHO, C.A.D.**: *Computer-Aided Control System Analysis and Parameter Setting Based on Root-Locus Properties*, Anais do XII Congresso Brasileiro de Automática – XII CBA, 14 a 18/09/1998, Uberlândia, MG, Vol. VI, pp. 1923-1928, 1998.

[10] **COELHO, C.A.D.**: *Design of Robust Control Systems by The Method of Inequalities,* Ph.D. Thesis, The University of Manchester Institute of Science and Technology, Manchester, Inglaterra, Julho de 1980.

[11] **COELHO, C.A.D.**: *Dominant pole Placement with Maximum Zero/Pole Ratio Phase-Lead Controllers*, Proceedings of the 1998 American Control Conference, Philadelphia, PA, EUA, 24 a 26/06/1998, Vol. 2, pp. 1159-1164, 1998.

[12] **COELHO, C.A.D.**: *Root-Locus Based MATLAB Commands for Control System Design*, 1997 MATLAB Conference, San Jose, CA, EUA, The MathWorks, Inc., http://www.mathworks.com/conf/user_contrib/coelho/, Natick,MA, EUA, 1997.

[13] **D'AZZO, J.J. & HOUPIS, C.H.**: *Análise e Projeto de Sistemas de controle Lineares*, 2ª edição, Editora Guanabara Dois S.A., Rio de Janeiro, RJ, 1984.

[14] **DAKESSIAN, V.G.**: *Design of Dynamic Control Systems by the Method of Inequalities*, Design Exercise Report, UMIST, Control Systems Centre, Manchester, Inglaterra, Março de 1980.

[15] **ELGERD, O.E.**: *Electric Energy Systems Theory*, T.M.H. Edition, Tata McGraw-Hill, New Delhi, Índia, 1973.

[16] **FONSECA, C.M. & FLEMING, P.J.**: *Multiobjective Genetic Algorithms Made Easy: Sharing and Mating Restrictions*, Genetic Algorithms in Engineering Systems, 12-14 September 1995 Conference Publication No.414, IEE, pp. 45-52, 1995.

[17] **FONSECA, C.M. & FLEMING, P.J.**: *Multiobjective Genetic Algorithms*, IEE, Savoy Place, London, UK, pp. 6/1-6/5, 1993.

[18] **LAUB, A.J. & LITTLE, J.N.**: *Control System Toolbox for Use with MATLAB - User's Guide*, The MathWorks, Inc., Natick, MA, EUA, 1986.

[19] **MESSNER, B. & TILBURY, D.**: *Control Tutorials for MATLAB and Simulink*, Addison-Wesley Publishing Company, EUA, 1998.

[20] **OGATA, K.**: *Engenharia de Controle Moderno*, 3ª edição, Editora Prentice- Hall do Brasil Ltda., Rio de Janeiro, RJ, 1998.

[21] **ROSENBROCK, H.H.**: *An Automatic Method for Finding the Greatest or Least Value of a Function*, The Computer Journal, Vol. 3, pp. 175-184, 1960.

[22] **THE MATHWORKS, INC.**: *Simulink User's Guide, The Student Edition of MATLAB: Version 5.3 User's Guide*, The MathWorks, Inc., Natick, MA, EUA, Abril de 1999.

[23] **THE MATHWORKS**, **INC.**: *Symbolic Math Toolbox User's* Guide, 1998.

[24] **THE MATHWORKS, INC.**: *The Student Edition of MATLAB: Version 5.3 User's Guide*, Prentice Hall, Englewood Cliffs, NJ, 1999.

[25] **WILSON, D.R.**: *Modern Practice in Servo Design*, Pergamon Press, Oxford, Inglaterra, 1970.

[26] **ZAKIAN, V. & AL-NAIB, U.**: *Design of Dynamical and Control Systems by the Method of Inequalities*, Proceedings IEE, Vol. 120, pp. 1421-1427, Novembro de 1973.

[27] **ZAKIAN, V.**: *New Formulation for the Method of Inequalities*, Proceedings IEE, Vol. 126. No. 6, pp. 579-584, Junho de 1979.

[28] **ZAKIAN, V.**: *The performance and Sensitivity of Classical Control Systems*, International Journal of Systems Science, Vol. 9, No. 3, pp. 343-355, 1977.

APÊNDICE

COMANDOS RELEVANTES MAIS UTILIZADOS DAS *TOOLBOXES* CACCON,"*CONTROL SYSTEM*" E "*SYMBOLIC MATH*" DO MATLAB.

Este apêndice inclui a descrição resumida dos principais comandos MATLAB da *Toolbox* CACCON [7], da *Control System toolbox* [18] e da *Symbolic Math toolbox* [23] do MATLAB, usados no programa de Projeto de Controle.

A.1 Comandos da Toolbox CACCON

1 - Comando "SOMAPOLI"

SINTAXE : *c = somapoli (a, b, epse)*

AÇÃO : Soma de dois polinômios: $c(s) = a(s) + b(s)$

a, b = vetores com os coeficientes de cada parcela.

c = vetor com os coeficientes do polinômio soma.

epse = precisão de ponto flutuante, usada para verificação do grau do polinômio resultante; se o módulo do coeficiente do termo de grau mais alto for inferior a *epse*, esse coeficiente é arredondado para zero e o grau do polinômio é reduzido; se *epse* não for especificado, é utilizado o valor padrão de 10^{-14}.

2 - Comando "CARACRD"

SINTAXE : *desemp = caracrd (t, y, yrp, epse, indt*)

AÇÃO : Determina os índices de desempenho da resposta a degrau $y(t)$

t = vetor de tempos; supõe-se que $t(1) = 0, y(1) = 0$

yrp = $y(\infty)$ não pode ser nulo

desemp = [*tr tp Mo% ts Td Lp%*]

tr = Tempo de subida (tempo necessário para a resposta atingir 90% do valor final pela primeira vez).

tp = Tempo de pico.

Mo% = Ultrapassagem máxima (percentual)

Ts = tempo de acomodação (tempo requerido para a resposta alcançar e permanecer dentro de uma faixa de 2% ou 5% do valor final).

epse = Tolerância (2% ou 5%).

Td = Período de oscilação.

Lp% = Ultrapassagem mínima (percentual).

A.2 Comandos da Control System Toolbox

1 - Comando "STEP"

SINTAXE : [*yout*] = *step* (*Num, Den, TY*)

AÇÃO : Calcula a resposta a Degrau da Função de Transferência *G (s) = Num (s) / Den (s),* onde Num e Den contêm os coeficientes do polinômio em potências decrescentes de *s*.

Num = Conjunto de coeficientes do polinômio do numerador da FT.

Den = Conjunto de coeficientes do polinômio do comum denominador de FT.

TY = Vetor *TY(i), i = 1,2, ...* , dos *t(i)* tempos de resposta de $y(t)$

2 - Comando "DCGAIN"

SINTAXE : *k = dcgain(sys)*

AÇÃO : Calcula o ganho estacionário do sistema linear invariante no tempo *sys.*

sys = Função de transferência do sistema.
k = ganho estacionário do sistema.

3 - Comando "ZERO"

SINTAXE : [*z*] = *zero(sys)*

AÇÃO : Calcula os zeros da FT *sys.*

sys = Função de transferência do sistema SLI e SISO.
z = vetor contendo os zeros da FT.

4 - Comando "POLE"

SINTAXE : [*p*] = *pole(sys)*

AÇÃO : Calcula os pólos da FT *sys.*

sys = Função de transferência do sistema SLI e SISO.
p = vetor contendo os pólos da FT.

5 - Comando "ISPROPER"

SINTAXE : *isproper*(*sys*)

AÇÃO : Determina se a FT *sys.*é própria ou não. O comando devolve 1 se a função é própria e 0 em caso contrário.

sys = Função de transferência do sistema SLI e SISO.

6 - Comando "ZPK"

SINTAXE : *sys* = *zpk*(z, p, k)

AÇÃO : Cria um sistema de controle *sys* com zeros *z*, pólos *p* e ganho *k.*

sys = Função de transferência do sistema SLI e SISO.
z = vetor contendo os zeros da FT.
p = vetor contendo os pólos da FT.
k = valor do ganho estacionário da FT.

A.3 Comandos da Symbolic Math Toolbox

1 - Comando "SYMS"

SINTAXE : *syms arg1 arg2 …*

AÇÃO : Cria os objetos simbólicos .*arg1, arg2, ...*

2 - Comando "SYM2POLY"

SINTAXE : *p = sym2poly(P)*

AÇÃO : Retorna um vetor contendo os coeficientes do polinômio simbólico *P*.

p = vetor de coeficientes.
P = polinômio simbólico.

3 - Comando "COLLECT"

SINTAXE : *s2 = collect(s1)*

AÇÃO : Reescreve uma expressão simbólica em termos da potencia da variável independente.

s1 = expressão simbólica original.
s2 = expressão simbólica reescrita em termos da potencia da variável independente.