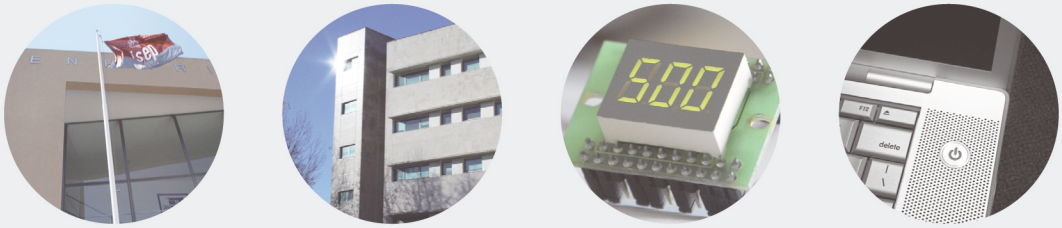
INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA ELECTROTÉCNICA E DE COMPUTADORES



Projeto de Controladores PID Difusos Adaptativos

ANDRÉ FILIPE SANTOS
Setembro de 2016

POLITÉCNICO
DO PORTO

Projeto de controladores PID difusos adaptativos

André Filipe Santos

Departamento de Engenharia Eletrotécnica

Mestrado em Engenharia Eletrotécnica e de Computadores

Área de Especialização em Automação e Sistemas

2016

Este relatório satisfaz, parcialmente, os requisitos que constam da Ficha da Unidade Curricular de Tese/Dissertação (TEDI), do 2º ano, do Mestrado em Engenharia Eletrotécnica e de Computadores

Candidato: André Filipe Santos, Nº 1111309, 1111309@isep.ipp.pt

Orientação científica: Ramiro de Sousa Barbosa, rsb@isep.ipp.pt

Departamento de Engenharia Eletrotécnica

Instituto Superior de Engenharia do Porto

8 de Setembro de 2016

Agradecimentos

Durante esta última etapa do meu percurso académico, foram muitos os momentos que sem o apoio de algumas pessoas não seria possível ter conseguido finalizar.

Agradeço ao meu professor e orientador, Eng. Ramiro Barbosa por todo o apoio constante que me transmitiu, pela disponibilidade com que me ajudou a superar os obstáculos e pela orientação dada no sentido de me ajudar a cumprir todos os objetivos a que me propus com este trabalho. Foi uma pedra fundamental no meu sucesso.

Agradeço à minha família e namorada, pois foram incansáveis desde o princípio. Foram cerca de sete meses bastante complicados e atribulados, mas que graças a vocês, nunca pensei em desistir e dei sempre o melhor de mim. Agradeço toda a paciência que tiveram para comigo e toda a força e incentivo que me deram ao longo deste tempo. Foram vitais para o meu sucesso.

Quero deixar também uma palavra de agradecimento aos meus colegas e amigos que me acompanharam durante todo o meu percurso académico. Foram cinco anos cheios de emoções, de bons momentos e entreajuda que espero que se alonguem por muitos mais anos. Sem vocês, também nada disto seria possível.

Resumo

Os sistemas de controlo estão cada vez mais cimentados no nosso quotidiano, desde as mais sofisticadas aplicações na indústria, até aos mais vulgares eletrodomésticos. A tecnologia moderna tem sofrido um crescimento exponencial, levando consigo o desenvolvimento dos sistemas de controlo modernos. Tal evolução tornou-se possível devido à criação de equipamentos mais complexos e fidedignos, passíveis de serem introduzidos no quotidiano. Para que toda esta tecnologia possa fluir em conjunto, técnicas de controlo PID ou controlo Difuso continuam atualmente a ser amplamente usadas. No entanto, para que se possa evoluir ainda mais, novas tecnologias e sistemas de controlo devem ser desenvolvidos e validados. Este projeto tem como finalidade desenvolver controladores PID-Difusos adaptativos, onde se junta o melhor de dois mundos (controlo PID e controlo Difuso), de modo a se puder controlar sistemas lineares e não lineares. Todo o desenvolvimento e simulações foram realizadas com auxílio do software MATLAB/Simulink. Posteriormente são propostos dois tipos de controladores PID-Difusos adaptativos, que serão sintonizados individualmente e também com recurso aos índices de desempenho *ITAE*, *ITSE*, *IAE* e *ISE*. Estes terão como termo de comparação, um primeiro controlador PID sintonizado com o método de Ziegler-Nichols em malha fechada e um segundo controlador PID sintonizado com recurso à função *pidtune* da MathWorks. Em geral, os controladores PID-Difusos adaptativos mostraram respostas bastante satisfatórias, apresentando melhores desempenhos que os controladores PID desenvolvidos.

Palavras-chave

PID, Lógica Difusa, PID-Difuso Adaptativo, *ITAE*, *ITSE*, *IAE*, *ISE*, MATLAB, Simulink.

Abstract

The control systems are increasingly cemented in our daily lives, from the most sophisticate industry applications to household appliances. Modern technology had an exponential growth, taking with it the development of modern control systems. Such evolution was possible due the creation of complex and trusted equipments which can be introduced in our lives. Aiming a stabilized workflow between all of these different technologies, PID and Fuzzy Logic techniques continue to be used. However, new technologies and control systems must be developed and validated to continue to evolve. The main objective of this project is to develop an Adaptive Fuzzy PID controller, which has the best of two worlds (PID and Fuzzy Logic control) and apply this controller in linear and nonlinear systems. All of the development and simulations was made with software MATLAB/Simulink. Further in this project, two types of Adaptive Fuzzy PID controllers are proposed, which in the first stage will be tuned individually and in a second stage will be tuned with performance index such ITAE, ITSE, IAE and ISE. These simulations will be compared with a PID controller tuned by Ziegler-Nichols closed loop method and with a PID controller tuned by *pidtune* of MATLAB. Overall, the adaptive Fuzzy PID controllers had satisfactory results and presented better performances than other developed controllers.

Keywords

PID, Fuzzy Logic, Adaptive Fuzzy-PID, ITAE, ITSE, IAE, ISE, MATLAB, Simulink.

Conteúdo

Lista de Figuras

Lista de Tabelas

Lista de Acrónimos

CO Saída do Controlador

COG Centro de Gravidade

FLC Controlador Lógico Difuso

FIS Sistema de Inferência Difuso

ISEP Instituto Superior de Engenharia do Porto

IAE *Integral of Absolute Error*

ISE *Integral of Square of the Error*

ITAE *Integral Time multiplied by the Absolute Error*

ITSE *Integral of Time multiplied by the Square Error*

MEEC Mestrado em Engenharia Eletrotécnica e de Computadores

MATLAB *Matrix Laboratory*

NB *Negative Big*

NM *Negative Medium*

NS *Negative Small*

P Controlo proporcional

PI Controlo proporcional-integral

PD Controlo proporcional-derivativo

PID Controlo proporcional-integral-derivativo

PB *Positive Big*

PM *Positive medium*

PS *Positive Small*

TEDI Tese/Dissertação

VP Variável do Processo

ZE *Zero*

Capítulo 1

Introdução

A realização do trabalho, apresentado ao longo desta dissertação, insere-se no âmbito da unidade curricular Tese/Dissertação (TEDI) do 2º ano de Mestrado em Engenharia Eletrotécnica e de Computadores (MEEC), no ramo de Automação e Sistemas, do Instituto Superior de Engenharia do Porto (ISEP). Este projeto, tem por base a realização de controladores PID-Difusos adaptativos, com posterior simulação através do uso do software MATLAB.

1.1 Contextualização

Os sistemas de controlo encontram-se cada vez mais infiltrados no nosso quotidiano, desde as mais sofisticadas aplicações na indústria, até aos mais vulgares eletrodomésticos. Este crescimento exponencial deve-se ao avanço que a tecnologia moderna tem sofrido. Esta tornou possível a criação de equipamentos cada vez mais complexos e fiáveis, capazes de substituir o Homem nas tarefas mais cansativas, mais monótonas e mais exigentes, com o mesmo ou melhor desempenho.

No entanto, a ideia do controlo está associada à atividade humana: os nossos sentidos fornecem indicações ao cérebro, que por sua vez controla os músculos, de modo a que seja executada a tarefa pretendida. Como exemplo, pode-se mencionar a tarefa de conduzir. A trajetória de condução é continuamente controlada pelo cérebro, a partir da imagem fornecida pelos olhos (ninguém de bom senso, conduz um automóvel de olhos fechados!). Transpondo isto para um contexto prático, embora os sistemas passíveis de controlo não possuam características humanas, estes são constituídos por sensores, circuitos de controlo e atuadores que substituem os olhos, cérebro e os músculos humanos, respetivamente.

Contudo, para que haja uma simbiose entre todos estes aplicativos e a capacidade de controlar equipamentos que executam tarefas de grande complexidade, é necessário recorrer a métodos

matemáticos precisos, para que seja possível projetar os seus sistemas de controlo. Com a organização destes métodos, deu-se origem ao aparecimento da teoria do controlo, que se tem vindo a desenvolver muito rapidamente, com o objetivo de satisfazer as mais diversas e complexas necessidades da indústria.

O problema do controlo pode ser exposto, considerando por exemplo, que se pretende manter um navio com uma trajetória constante. Para cumprir esta tarefa, pode-se colocar o navio na trajetória pretendida, bloqueando-se o leme. No entanto, esta solução não é a mais satisfatória, porque este método não tem em conta os desvios que serão provocados, por exemplo, pelo vento e pelas correntes. Posto isto, para se manter o navio com a trajetória desejada, torna-se necessário a existência de uma comparação contínua entre a trajetória real e a pretendida. Com isto, caso haja desvio na trajetória, efetuar-se-á o controlo do leme para se realizar a devida correção da trajetória.

A resposta a este tipo de problemas, na maior parte das vezes, não é simples. A solução clássica deste tipo de dilemas consiste em estabelecer uma relação entre o desvio (ou erro), a ação corretiva (ou variável de controlo) e as características físicas e económicas do sistema a controlar, o que nem sempre é fácil. Deste modo, para se efetuar um controlo eficaz, deve-se ter em conta as características físicas do sistema, isto porque são estas que vão determinar a resposta dinâmica do mesmo.

1.2 Motivação

As técnicas de controlo Proporcional-Integral-Derivativo (PID) continuam a ser amplamente utilizadas em processos industriais. A razão para a sua utilização deve-se essencialmente à sua reconhecida simplicidade e existência de metodologias de sintonia dos correspondentes ganhos. No entanto, no caso de sistemas não lineares torna-se mais difícil a sua implementação. As técnicas de controlo difusas são inerentemente abordagens não lineares, dado que incorporam três fontes principais de não linearidade, nomeadamente, a base de regras, o mecanismo de inferência e os módulos de fuzificação e desfuzificação.

Este paradigma de controlo baseado em lógica difusa tem provado ser uma abordagem real no controlo de vários sistemas lineares, assim como também não lineares e tem vindo a ser sugerido como alternativa às técnicas de controlo convencionais [1]. Estes controladores são conhecidos por apresentarem maior robustez, comparativamente aos controladores convencionais e o seu desempenho ser menos sensível a variações paramétricas do sistema ou a grandezas não modeladas [2]. Para além disso, aplicações recentes dos controladores difusos têm mostrado um grande potencial no contexto de sistemas mal definidos que podem ser convenientemente

controlados por operadores humanos, sem o conhecimento explícito das dinâmicas do sistema [3].

Apesar das grandes potencialidades relativamente à aplicação de controladores difusos em vários contextos, encontrar um conjunto de variáveis linguísticas, regras e fatores de escala, e subsequentemente sintonizá-los, apresenta ainda um desafio que urge dar resposta, devido em grande medida à inexistência de uma abordagem sistemática.

Sendo estes dois métodos que possuem bastantes potencialidades no controlo de sistemas, torna-se um desafio poder juntar o melhor de dois controladores bem conhecidos, com o intuito de obter ainda melhor respostas por parte dos sistemas, sendo estes lineares ou não lineares.

1.3 Objetivos

De uma forma geral, o objetivo deste projeto passa por aprofundar os conhecimentos sobre o controlo PID, controladores difusos e MATLAB. Posteriormente, será realizado um projeto de controladores PID difusos adaptativos, com recurso ao MATLAB.

Dada a complexidade inerente a este objetivo, sentiu-se a necessidade de o subdividir em múltiplas tarefas de realização mais simples, tais como:

- Estudo teórico e prático sobre controladores PID;
- Estudo teórico e prático sobre controladores com lógica difusa;
- Estudo do *software* MATLAB;
- Desenvolvimento de controladores PID-Difusos adaptativos;
- Simulações computacionais;
- Comparação entre todos os tipos de controladores desenvolvidos.

1.4 Calendarização

Na Figura 1.1 é possível observar a distribuição das tarefas realizadas ao longo destes últimos meses de trabalho neste projeto.

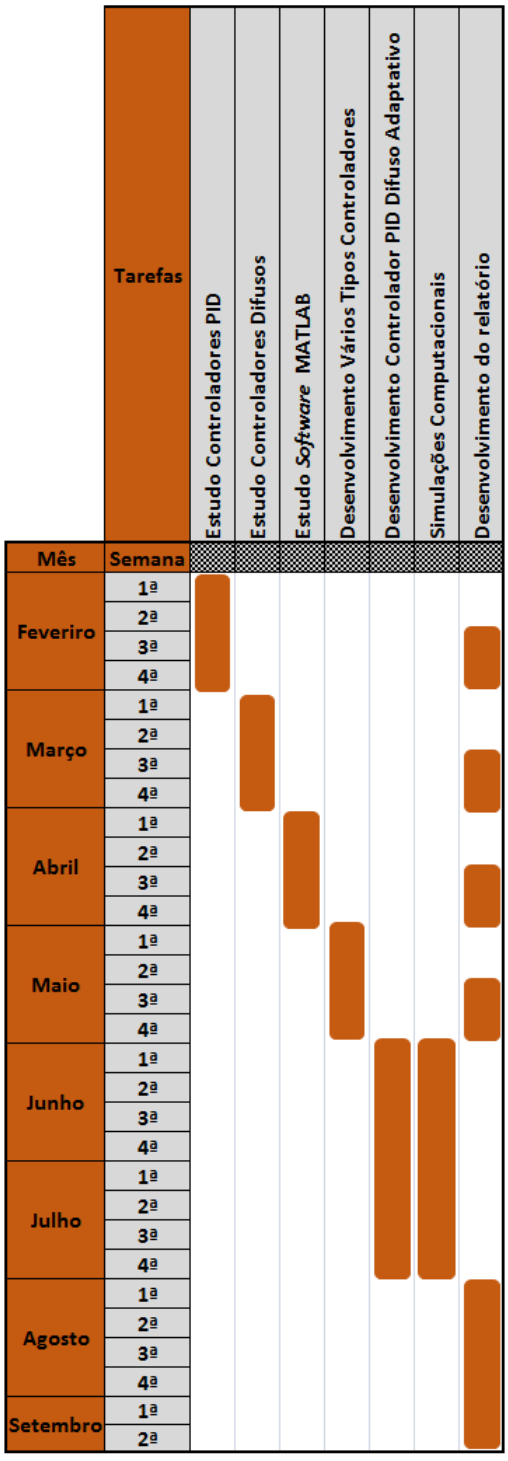


Figura 1.1: Calendarização do trabalho desenvolvido.

1.5 Estrutura do Relatório

O presente relatório está dividido em 6 capítulos. No primeiro capítulo será feita uma abordagem introdutória ao tema deste trabalho. No segundo capítulo será feito um estudo no âmbito de controladores PID, onde estará inerente toda a teoria relacionada com o tema. Já no terceiro capítulo, será realizado um estudo sobre controladores PID Difusos, onde será abordada toda a sua teoria, assim como os tipos de controladores e a sua utilização no MATLAB. No quarto capítulo, irá ser abordado todo o processo do desenvolvimento deste trabalho, desde aos controladores desenvolvidos, até aos métodos utilizados. Já no quinto e sexto capítulos, serão expostos os resultados e as conclusões obtidas, respetivamente.

Capítulo 2

Controladores PID

O PID é uma metodologia de controlo linear, cuja lei de controlo é baseada no erro da variável a controlar. Foi pela primeira vez apresentado por N. Minorsky em 1922. No entanto, atualmente continua a ser o método de controlo mais usado em todo o mundo, tanto por parte da indústria em sistemas de controlo industrial, como na maioria dos controladores comercialmente disponíveis.

Neste capítulo será descrito o controlador PID, as suas principais aplicações, as suas características e os diversos métodos de sintonia.

2.1 Introdução ao PID

A popularidade dos controladores PID, pode ser atribuída com base em vários fatores: são matematicamente simples, de fácil compreensão, fiáveis, e requerem baixa capacidade e custo computacional. Têm tanto destaque, que são muitas as técnicas não lineares utilizadas para o ajuste dos seus parâmetros. Existem mesmo autores que referem que, em processos com perturbações imprevistas e frequentes, o método PID bem ajustado é o que apresenta um melhor desempenho e robustez, exceto nos sistemas com atraso [4]. Por um lado, é verdade que o controlo PID está massivamente estudado na literatura [5][6][7] e é uma das técnicas de controlo mais populares na indústria porque, além das vantagens acima referidas, responde com desempenhos suficientes na maioria dos processos com requisitos pouco exigentes. Por outro lado, também é verdade que o avanço no desenvolvimento das técnicas de controlo não linear e da análise da complexidade deste tipo de sistemas, tem potenciado o controlo de sistemas até requisitos de desempenho totalmente fora do alcance das técnicas PID lineares, independentemente do seu método de ajuste e da configuração com que se apresente. Portanto, não se realça tanto o possível abandono da utilização desta técnica devido ao número de processos que pode

efetivamente controlar, mas antes porque os processos tendem a ser cada vez mais complexos, e pela facilidade de implementação de algumas técnicas não lineares, que são flexíveis ao ponto de regular o desempenho desejado até às solicitações mais exigentes.

Nos métodos práticos de sintonia, o primeiro passo para a utilização de um controlador PID, será a escolha do tipo de controlador a usar. Este poderá conter apenas a componente de ação proporcional (P), a ação proporcional-integral (PI), a ação proporcional-derivativa (PD) ou então a ação proporcional-integra-derivativa (PID). Posteriormente, é necessário fazer o ajuste dos vários parâmetros do controlador. Este ajuste consiste na dedução, tendo em conta a resposta do sistema, quando este é sujeito a entradas específicas com valores que vão permitir o cálculo dos referidos parâmetros. Este procedimento possui a vantagem de não exigir a necessidade de conhecer o modelo do sistema, sendo este, muitas vezes, difícil de determinar. Já no método analítico, procede-se à sintonia dos modos PID para uma aplicação específica, de modo a que determinados critérios de desempenho sejam verificados. Normalmente, este método é usado sempre que se conhece a função de transferência do sistema.

2.1.1 Estrutura do PID

Na Figura 2.1, é possível observar a configuração de um controlador PID em malha fechada.

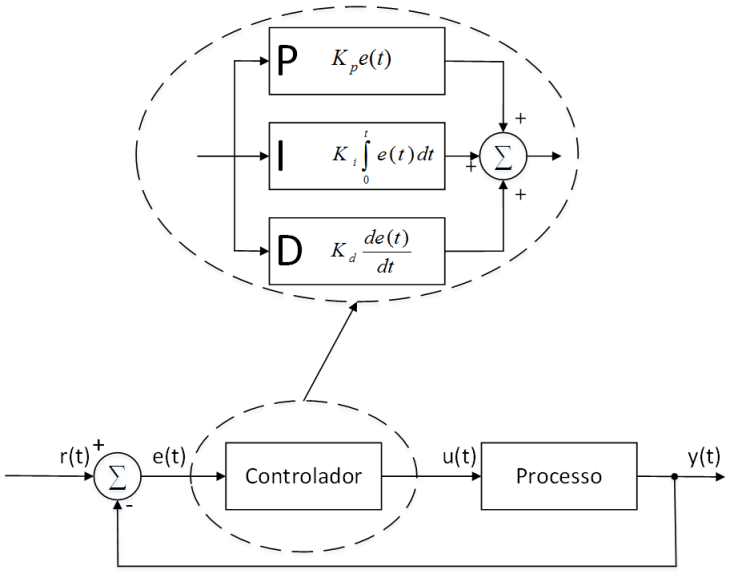


Figura 2.1: Diagrama de blocos do controlador PID.

O objetivo deste tipo de controlador, passa por manter a saída do processo $y(t)$, no valor desejado ou então no valor de referência dado por $u(t)$, eliminando continuamente o erro $e(t)$. Para isso, o controlador irá aplicar ininterruptamente a ação de controlo à entrada do processo [8]. Esta ação é composta pela soma dos termos que o constituem. Ou seja, pela soma do

termo proporcional, integral e derivativo do erro, sendo que o erro é a diferença entre o sinal de entrada e o sinal de saída, dada pela expressão (2.1).

$$e(t) = u(t) - y(t) \tag{2.1}$$

O processo de sintonia do controlo, começa pelos requisitos de desempenho do sistema. O controlo de desempenho do sistema é geralmente medido pela inserção de uma função em degrau, definida como *setpoint*, sendo posteriormente medida a resposta da variável do processo. Geralmente, a resposta é quantificada pelas características da onda de resposta (Figura 2.2). O tempo de subida ou *Rise Time* (T_r) é o tempo que o sistema leva para ir de 10% a 90% do estado estacionário, ou valor final. A sobreelongação máxima ou o *Percent Overshoot* (M_p) é o valor em que a variável do processo ultrapassa o valor final, expresso como uma percentagem do valor final. O tempo de estabelecimento ou o *Settling Time* (T_s), é o tempo necessário para a variável do processo atingir uma determinada percentagem (normalmente 2%) do valor final. Por fim, o erro em regime permanente ou o *Steady-State Error* (e_{ss}), é a diferença final entre as variáveis do processo e o *setpoint* [9].

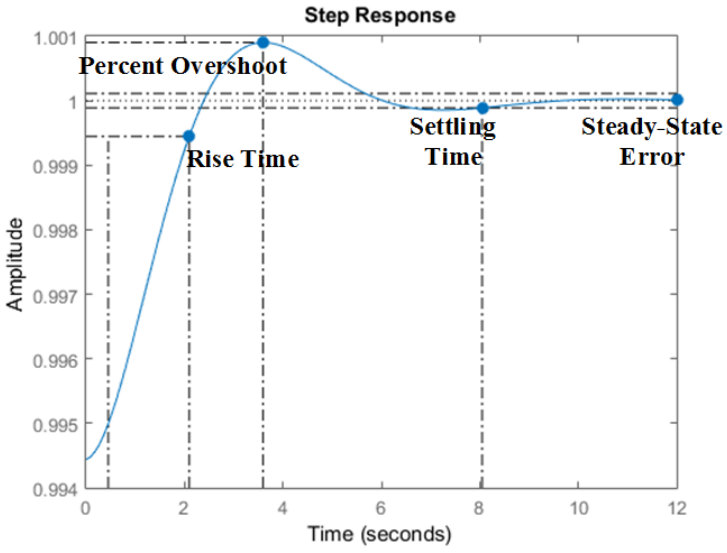


Figura 2.2: Resposta típica de um sistema PID de malha fechada.

De modo a se ajustarem estes parâmetros para que o desempenho do sistema seja o ideal, é necessário alterar os parâmetros do controlador. Ou seja, é preciso alterar o ganho proporcional (K_p), o ganho integrativo (K_i) e o ganho derivativo (K_d) [8].

Assim sendo, tem-se:

$$u(t) = Ação\ de\ Controlo = K_p e(t) + K_i \int_0^t e(t)dt + K_d \frac{de(t)}{dt} \tag{2.2}$$

2.1.2 Ação Proporcional

Na Figura 2.3, é possível observar o diagrama de um controlo apenas com a ação proporcional.

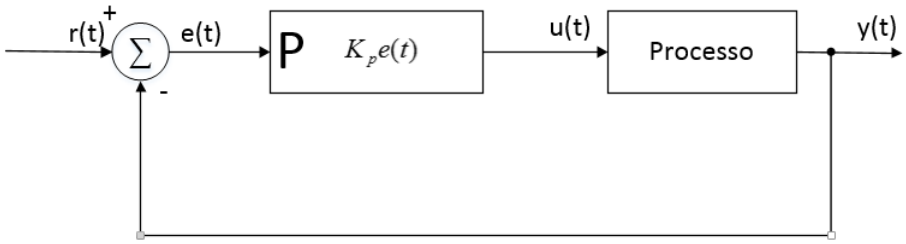


Figura 2.3: Diagrama de blocos de um controlador P.

Este tipo de controlo, é caracterizado pela seguinte expressão:

$$u(t) = K_p e(t) \xrightarrow{L} U(s) = K_p E(s) \tag{2.3}$$

A ação proporcional, como o próprio nome indica, age proporcionalmente ao erro $e(t)$ entre a entrada e a saída do sistema. Sintonizando este parâmetro, quanto maior for o seu valor, menor será o erro em regime permanente. Ou seja, a precisão do sistema em malha fechada é otimizada. O erro $e(t)$ será diminuído com o aumento de K_p, no entanto, nunca poderá ser anulado. Em contrapartida, quanto maior for o ganho de K_p, mais instável o sistema pode ficar [11].

Na Figura 2.4, é possível observar um gráfico com uma entrada de referência (linha azul) e as saídas com o valor de K_p a assumir vários valores.

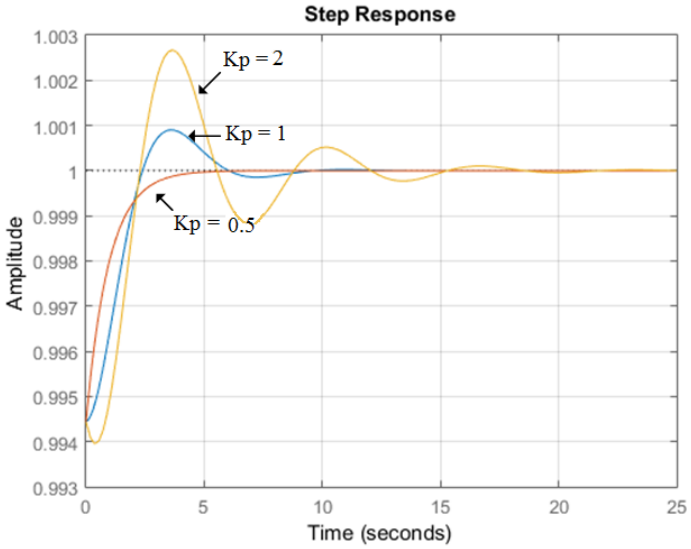


Figura 2.4: Gráfico com a entrada de referência (azul) e as várias saídas para vários valores de K_p.

2.1.3 Ação Proporcional-Integral

Na Figura 2.5, é possível observar o diagrama de um controlo para a ação proporcional-integral.

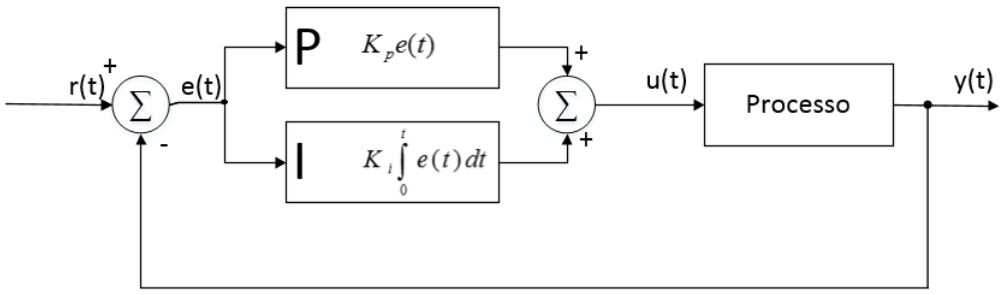


Figura 2.5: Diagrama de blocos de um controlador PI.

Este tipo de controlo, é caracterizado pela seguinte expressão:

$$u(t) = K_p \left[e(t) + \frac{1}{T_i} \int_0^t e(\tau) d\tau \right] \xrightarrow{L} U(s) = K_p \left(1 + \frac{1}{T_i s} \right) E(s) \tag{2.4}$$

Onde T_i é a designação de tempo integral e representa o tempo necessário para que haja uma igualdade entre a ação integral e a ação proporcional.

A ação integral age proporcionalmente à integral do erro do sistema. Esta é responsável por garantir um erro igual a zero em regime permanente para as entradas em degrau, quando o sistema em malha fechada for internamente estável, e rejeitar perturbações aplicadas na entrada

do processo [12]. Na prática, após aplicar este tipo de controlador, o erro em regime permanente é eliminado, independentemente do sistema que se pretende controlar. No entanto, o tempo de estabelecimento irá aumentar, piorando a estabilidade do sistema. Com isto, será necessário reduzir o ganho proporcional, de modo a equilibrar a resposta do sistema.

Na Figura 2.6, é possível observar um gráfico com uma entrada de referência (linha azul) e as saídas com o valor de T_i a assumir vários valores, para um valor de K_p fixo.

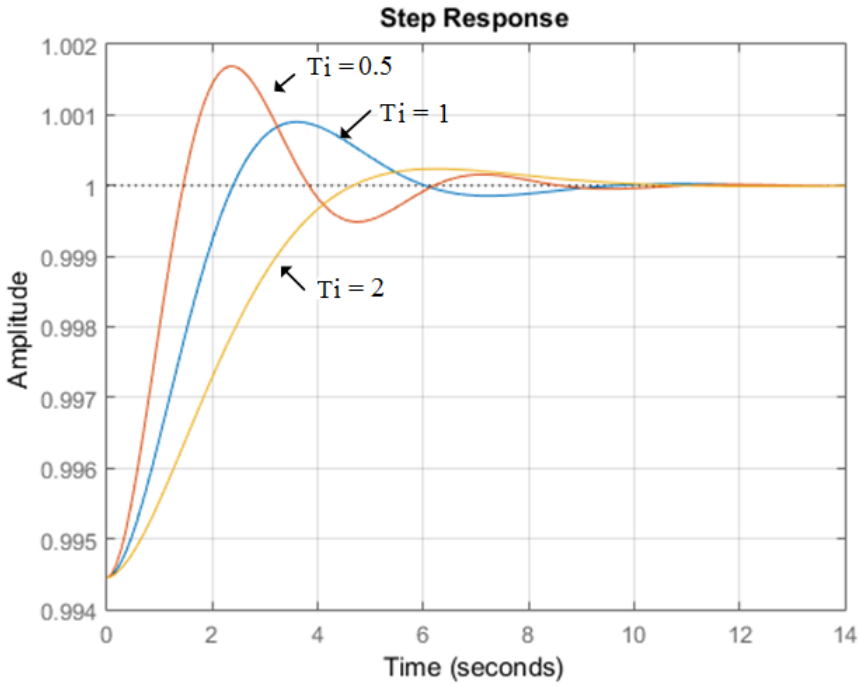


Figura 2.6: Gráfico com a entrada de referência (azul) e as várias saídas para vários valores de T_i.

2.1.4 Ação Proporcional-Derivativa

Na Figura 2.7, é possível observar o diagrama de um controlo para a ação proporcional-derivativa.

Este tipo de controlo, é caracterizado pela seguinte expressão:

$$u(t) = K_p \left[e(t) + T_D \frac{de(t)}{dt} \right] \xrightarrow{L} U(s) = K_p(1 + T_D s) \tag{2.5}$$

Onde T_D é a designação de tempo derivativo e representa a antecipação da ação derivativa relativamente à ação proporcional.

A ação derivativa age proporcionalmente à derivada do erro do sistema e é responsável por melhorar o seu desempenho. Devido à dinâmica do processo, existe um atraso entre a variação

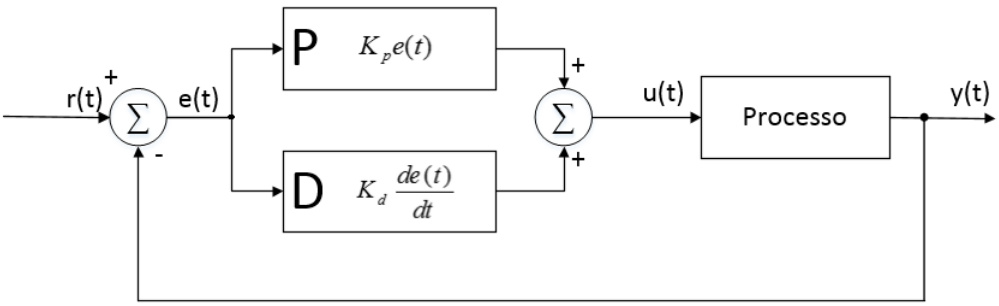


Figura 2.7: Diagrama de blocos de um controlador PD.

do sinal de controlo e a sua influência no sinal de saída. Posto isto, um controlador deste tipo, pode ser interpretado como se o controlo atuasse proporcionalmente sobre a previsão do sinal de erro. Esta previsão é feita, extrapolando a curva do erro, utilizando a sua tangente no instante de tempo t [12], como se pode observar na Figura 2.8.

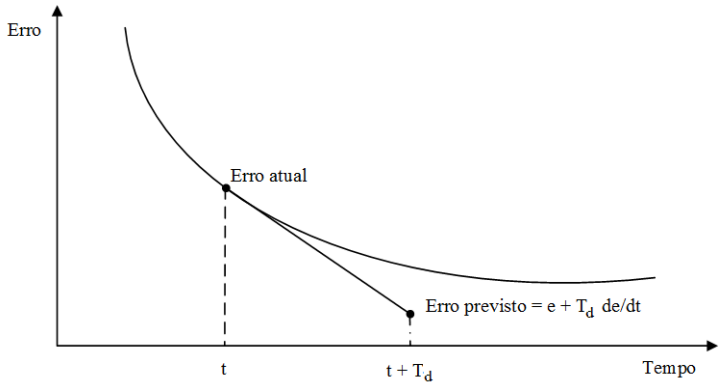


Figura 2.8: Interpretação da ação proporcional-derivativa como ação de controlo preditivo.

Este tipo de controlo não pode ser usado sozinho, pois vai ser proporcional à taxa de variação do erro. A ação de controlo derivativa, quando adicionada a um controlador proporcional, propícia um meio de obter um controlador com alta sensibilidade. Uma vantagem de se usar o controlo em questão, é que este responde à taxa de variação do erro e pode produzir uma correção significativa antes do valor do erro se tornar demasiado grande. Assim, o controlo derivativo, antecipa o erro e inicia uma ação corretiva mais cedo, tendendo a aumentar a estabilidade do sistema [12].

Na Figura 2.9, é possível observar um gráfico com uma entrada de referência (linha azul) e as saídas com o valor de T_d a assumir vários valores, para um valor de K_p fixo.

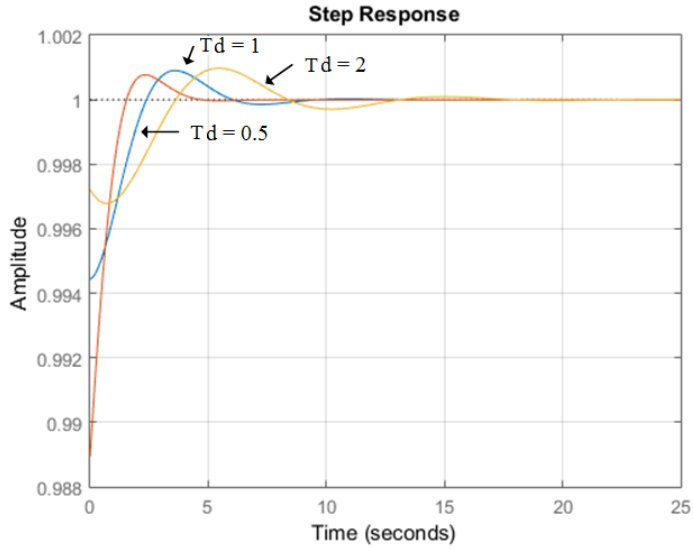


Figura 2.9: Gráfico com a entrada de referência (azul) e as várias saídas para vários valores de T_d.

2.1.5 Ação Proporcional-Integra-Derivativa

Na Figura 2.10, é possível observar o diagrama de um controlo para a ação proporcional-integra-derivativa. Este tipo de controlo é a junção das três ações de controlo mencionadas anteriormente (P, PI e PD).

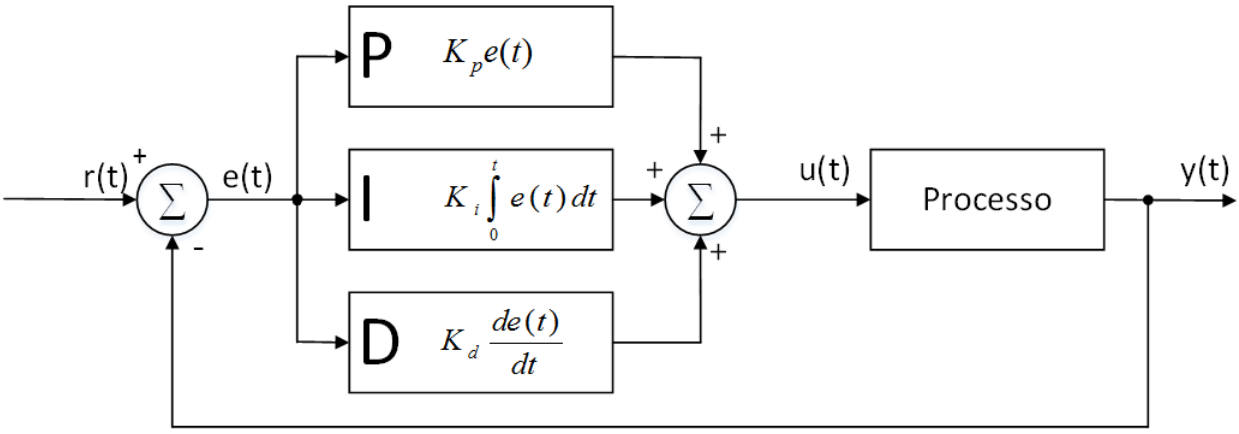


Figura 2.10: Resposta típica de um sistema PID de malha fechada.

Este tipo de controlo, é caracterizado pela seguinte expressão:

$$u(t) = K_p \left[e(t) + \frac{1}{T_i} \int_0^t e(\tau) d\tau + T_D \frac{de(t)}{dt} \right] \xrightarrow{L} U(s) = K_p (1 + \frac{1}{T_i s} + T_D s) E(s) \qquad (2.6)$$

Num controlador PID o parâmetro integral é utilizado para eliminar o erro em regime permanente, proveniente de grandes variações dos parâmetros do sistema. O parâmetro derivativo, com o seu efeito estabilizador, provoca o aumento do ganho e a redução das oscilações, o que leva a uma resposta mais rápida quando comparado com o controlador P e PI [10].

Para facilitar a sintonia de um sistema, apenas com base no efeito que cada parâmetro produz na resposta, pode-se resumir esses mesmos efeitos na Tabela 2.1. No entanto, esta tabela só deve ser usada como referência, pois uma alteração num parâmetro produz efeitos nos restantes.

Tabela 2.1: Dinâmica do sistema, tendo em conta os ganhos do PID (K_p, K_i e K_d) [13].

Ganho	**Tempo de Subida** (T_r)	**Overshoot** (M_p)	**Tempo de Estabele-cimento** (T_s)	**Erro Regime Permanente** (e_{ss})	**Estabilidade**
$\uparrow K_p$	Diminui	Aumenta	Aumenta Pouco	Diminui	Piora
$\uparrow K_i$	Diminui Pouco	Aumenta	Aumenta	Diminui Muito	Piora
$\uparrow K_d$	Diminui Pouco	Diminui	Diminui	Pouca Variação	Melhora

2.2 Métodos de Sintonia de Controladores PID

A essência da sintonia de sistemas de controlo passa por identificar como a dinâmica de um processo reage aos esforços de controlo e, tendo por base os requisitos de desempenho, determinar a dinâmica necessária do algoritmo PID para este eliminar os erros [14]. Independentemente da metodologia do projeto, as seguintes três etapas são comuns aos métodos de identificação da dinâmica do processo e sintonia de controladores PID:

1. O processo é submetido a perturbações no sinal de controlo;
2. A resposta do sistema a este distúrbio é analisada e quantificada;
3. Tendo por base a análise da resposta e as especificações de desempenho, os parâmetros PID são sintonizados [15].

Historicamente, um importante passo no desenvolvimento de metodologias de sintonia de controladores PID, foi dado por Ziegler-Nichols (1942). O método é baseado em caracterizar

a dinâmica do processo por três parâmetros e usar de fórmulas simples. Em contrapartida, o método fornece pouca informação sobre o sistema e produz sistemas em malha fechada com baixo amortecimento e baixa robustez. Ainda assim, este trabalho teve um impacto significativo no controlo de sistemas. Até hoje, a maioria dos fabricantes e/ou utilizadores de controladores PID, aplicam o método, ou variantes do mesmo, na sintonia de controladores.

2.2.1 Método de Ziegler-Nichols

Desenvolvido por J. G. Ziegler e N. B. Nichols, ambos da *Taylor Instruments Companies*, foi o primeiro método de ajuste sistemático dos parâmetros de um controlador PID. Os autores desenvolveram regras empíricas de ajuste dos parâmetros do controlador, baseados em testes práticos realizados em determinados processos com o controlador comercial *Fulscope da Taylor*. Dois métodos clássicos para determinar os parâmetros do controlador PID foram apresentados por Ziegler e Nichols, em 1942. Estes métodos são largamente usados, seja na sua forma original, ou com algumas modificações e consistem em determinar algumas características da dinâmica do processo.

Método da Resposta em Malha Fechada

Neste método, com o controlador P em malha fechada, aumenta-se o ganho proporcional gradualmente, até se obter uma resposta oscilatória com amplitude constante. Neste ponto, determina-se o ganho crítico (K_c) e o período crítico de oscilação (T_c). O K_c é o valor do ganho do controlador P que gerou uma resposta oscilatória sustentada na saída do processo e o T_c será o próprio período do processo oscilante [16]. A Figura 2.11 mostra um exemplo da resposta de um processo durante o ensaio.

Com estes valores de K_c e T_c, observa-se então a Tabela 2.2 proposta por Ziegler Nichols, para se obter a sintonia PID, usando como critério de desempenho uma razão de decaimento igual a $\frac{1}{4}$, entre sucessivos picos da resposta oscilatória do sistema.

Na prática, este método pode levar o processo a variar fora de uma região segura, podendo causar instabilidade do sistema. Daí, este método não ser muito utilizado na prática.

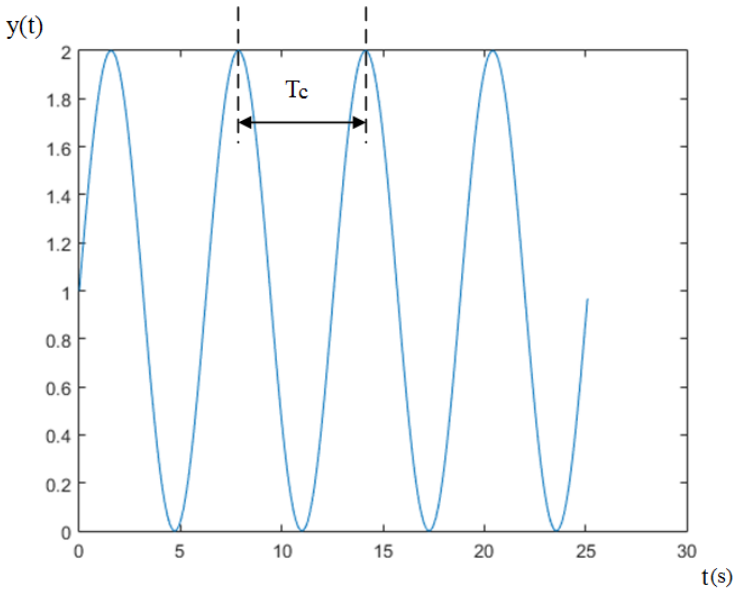


Figura 2.11: Resposta do sistema usando o método da resposta em malha fechada.

Tabela 2.2: Sintonia de Ziegler-Nichols para o método da resposta em malha fechada [16].

Controlador	K_p	T_i	T_d
P	$0,5K_c$	-	-
PI	$0,45K_c$	$\frac{T_c}{1,2}$	-
PID	$0,6K_c$	$\frac{T_c}{2}$	$\frac{T_c}{8}$

Método da Resposta ao Degrau

O outro método apresentado por Ziegler e Nichols baseia-se na informação proveniente do processo na forma da resposta ao degrau do sistema em malha aberta. Neste método, pressupõe-se que a resposta em malha aberta de um sistema a uma entrada em degrau unitário é monótona após um tempo inicial, conforme se pode observar na Figura 2.12. Este pode ser aproximado pela função de transferência de um sistema de primeira ordem com atraso, conforme a função de transferência (2.7).

$$G_p(s) = \frac{K}{\tau s + 1} e^{-\theta s} \tag{2.7}$$

Em que K é o ganho estático do processo, dado por $K = \frac{\Delta y}{\Delta u}$.

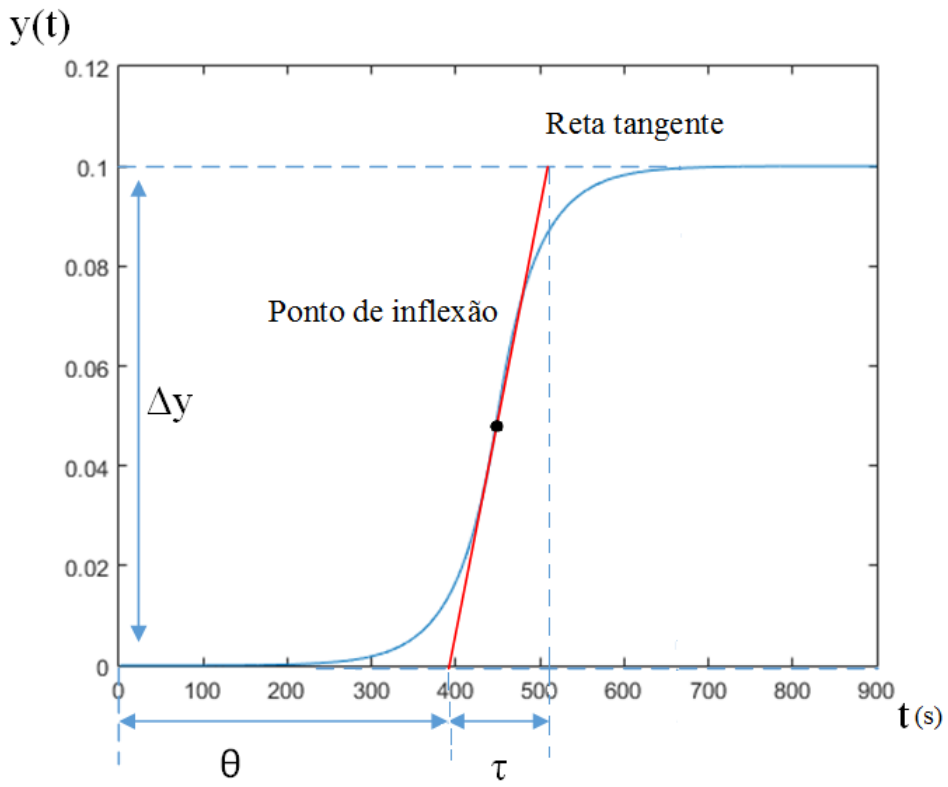


Figura 2.12: Resposta do processo em malha aberta.

A curva de resposta ao degrau será caracterizada por duas constantes: o atraso de transporte (θ) e a constante de tempo (τ). Estes são determinados, através do desenho de uma linha tangente no ponto de inflexão da curva e determinando-se a intersecção desta mesma linha com o eixo do tempo (abcissa) e a linha correspondente ao valor final da saída [16], conforme mostra a Figura 2.12.

Com estes valores de K, τ e θ, observa-se então a Tabela 2.3, proposta por Ziegler e Nichols (1943), que mostra a sintonia PID em função dos parâmetros de um modelo de primeira ordem com atraso ou tempo morto.

2.2.2 Método de Cohen-Coon

Proposto em 1953 pelo engenheiro G. H. Cohen e pelo matemático G. A. Coon, ambos da *Taylor Instruments*. Este é um método também baseado num critério de razão de decaimento de $\frac{1}{4}$ da amplitude do amortecimento da resposta (como o método de Ziegler e Nichols), em resposta a uma perturbação na carga. As regras de Cohen-Coon são adequadas para uma maior variedade de processos que as de Ziegler e Nichols. Embora as regras de Ziegler e Nichols funcionem bem em processos onde o tempo de atraso é inferior a metade da constante de tempo, as regras de

Tabela 2.3: Sintonia de Ziegler-Nichols para o método da resposta ao degrau [16].

Controlador	K_p	T_i	T_d
P	$\frac{\tau}{K\theta}$	-	-
PI	$0,9\left(\frac{\tau}{K\theta}\right)$	$3,33\theta$	-
PID	$1,2\left(\frac{\tau}{K\theta}\right)$	2θ	$0,5\theta$

Cohen-Coon funcionam bem em processos onde o tempo de atraso é inferior a duas vezes a constante de tempo (podendo mesmo ser utilizado para valores superiores, se necessário) [13].

Na sua sintonia, este método faz uso de três parâmetros do processo: ganho (g_p), tempo de atraso (t_d) e constante de tempo (τ). Na Figura 2.13 é possível observar um gráfico referente a um teste de sintonia com o método de Cohen-Coon.

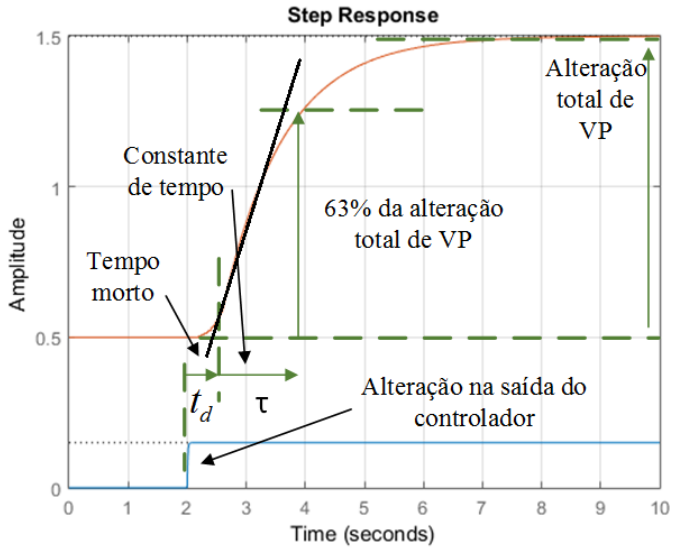


Figura 2.13: Teste de sintonia do método de Cohen-Coon.

Para efetuar a sintonia de um sistema com este método, é necessário seguir os seguintes passos [13]:

1. Colocar o controlador em modo manual e esperar que o processo estabilize;
2. Aplicar uma pequena variação em degrau na saída do controlador (CO) e esperar que a variável de processo (VP) estabilize num novo valor;
3. Converter a variação total de VP em percentagem do alcance do dispositivo de medida;

4. Calcular o ganho do processo (g_p):

$$g_p = \frac{\Delta \text{VP}}{\Delta \text{CO}} 100\% \tag{2.8}$$

5. Determinar o declive máximo da curva de resposta da VP. Isto ocorre no ponto de inflexão (onde a VP interrompe o processo de curvatura ascendente e começa com a curvatura descendente);

6. Determinar o tempo de atraso (t_d);

7. Calcular o valor da VP a 63% da sua variação total. Na curva de resposta da VP, determinar o tempo para o qual a VP atinge este valor;

8. Determinar a constante de tempo (τ);

9. Obter os parâmetros de sintonia do controlador usando as regras de Cohen-Coon, através da Tabela 2.4;

10. Reduzir para metade o ganho do controlador de modo a obter menos oscilação e maior estabilidade;

11. Monitorizar o desempenho do controlador periodicamente.

Tabela 2.4: Parâmetros de sintonia do controlador de Cohen-Coon [13].

Controlador	K_p	T_i	T_d
P	$\frac{1,03}{g_p}(\frac{\tau}{t_d} + 0,34)$	-	-
PI	$\frac{0,9}{g_p}(\frac{\tau}{t_d} + 0,092)$	$3,33t_d\frac{\tau + 0,092t_d}{\tau + 2,22t_d}$	-
PD	$\frac{1,24}{g_p}(\frac{\tau}{t_d} + 0,129)$	-	$0,27t_d\frac{\tau - 0,324t_d}{\tau + 0,129t_d}$
PID	$\frac{1,35}{g_p}(\frac{\tau}{t_d} + 0,185)$	$2,5t_d\frac{\tau + 0,185t_d}{\tau + 0,611t_d}$	$0,37t_d\frac{\tau}{\tau + 0,185t_d}$

Capítulo 3

Controladores Difusos

Um bom ponto de partida será apresentar o testemunho de Steve Marsh, diretor de operações da Motorola, para sublinhar as potencialidades do controlo por lógica difusa: "Eu tinha acabado de chegar ao Japão. A minha primeira reunião de negócios estava agendada para essa mesma tarde mas o avião tinha-se atrasado uma hora. Corri para o hotel ainda com a leve esperança de chegar a tempo para a reunião. O hotel estava com muito movimento. Carreguei no botão para chamar "o elevador"e, após um muito curto espaço de tempo, a porta abriu-se. Carreguei no botão para o 9º piso. A porta fechou-se, e imediatamente abriu-se outra vez. Tendo pouca paciência para elevadores estúpidos (especialmente quando estou com pressa), sussurrei alguma coisa e carreguei novamente no botão para ir para o 9º piso. Mas foi quando percebi que já lá estava. De alguma maneira o elevador tinha-me movimentado 9 pisos de forma incrivelmente rápida e somente percetível usando a concentração. Os meus contactos no Japão disseram-me que o elevador era controlado por lógica difusa e que era 30% mais eficiente que os elevadores tradicionais"[17]. O Prof. Lotfi Zadeh, considerado o pai da lógica difusa, escreveu que "na maior parte dos casos, poderemos desenvolver a mesma solução sem recorrer à lógica difusa, mas com a lógica difusa é mais rápido e barato"[18].

Lotfi Zadeh diz ainda que "a maior contribuição da lógica difusa é a sua metodologia para processamento e programação com palavras – uma metodologia baseada no conceito de variáveis linguísticas e no cálculo de regras difusas"[17]. O conceito fundamental de um controlador lógico difuso (FLC) é que a sua interface com o programador é baseada em variáveis linguísticas que são palavras e não números e, portanto, utiliza uma abstração que "processa com palavras em vez de números". Contrasta com a filosofia das linguagens de programação tradicionais porque se adapta ao raciocínio humano, e não o raciocínio humano à "compreensão do processador". As palavras, como são conceptualmente menos precisas que os números, permitem a exploração da imprecisão à tolerância, tornando a sua utilização mais próxima do raciocínio humano e

com a "consciência"da importância do contexto na resolução dos problemas, como exemplifica exatamente a Figura 3.1.

Figura 3.1: Relação entre precisão e contexto [18].

Este tipo de "controlo inteligente"deve ser visto como um objetivo e não como uma realidade adquirida. De facto, não se pretende substituir um sistema biológico, mas antes replicar funções de um sistema biológico inteligente que ajude a resolver problemas de controlo.

3.1 Introdução ao Controlo Lógico Difuso

O sistema de controlo difuso é um mapeamento não linear estático entre as suas entradas e saídas [19]. Na Figura 3.2, está representado um diagrama de blocos elementar de um sistema de controlo difuso. Este incorpora essencialmente 6 elementos: o pré-processamento, a interface de fuzificação, o mecanismo de inferência, a interface de desfuzificação e o pós-processamento. Muito sucintamente, o bloco de fuzificação converte as entradas em conjuntos difusos, o mecanismo de inferência usa regras difusas de uma base de regras para produzir conclusões difusas e a desfuzificação converte estas conclusões difusas em saídas numéricas.

3.1.1 Conjuntos Difusos

Os conjuntos difusos foram propostos por Zadeh em 1965 como sendo uma extensão dos conjuntos clássicos. Zadeh apercebeu-se que, na realidade, as informações com que trabalhamos são de natureza incerta e que com a teoria dos conjuntos clássicos, se tornava complicada a tradução matemática e objetiva, de algo com caraterísticas difusas.

Na teoria dos conjuntos clássicos, um elemento pertence, ou não, a um determinado conjunto, enquanto que na teoria dos conjuntos difusos pode pertencer, não pertencer ou pertencer

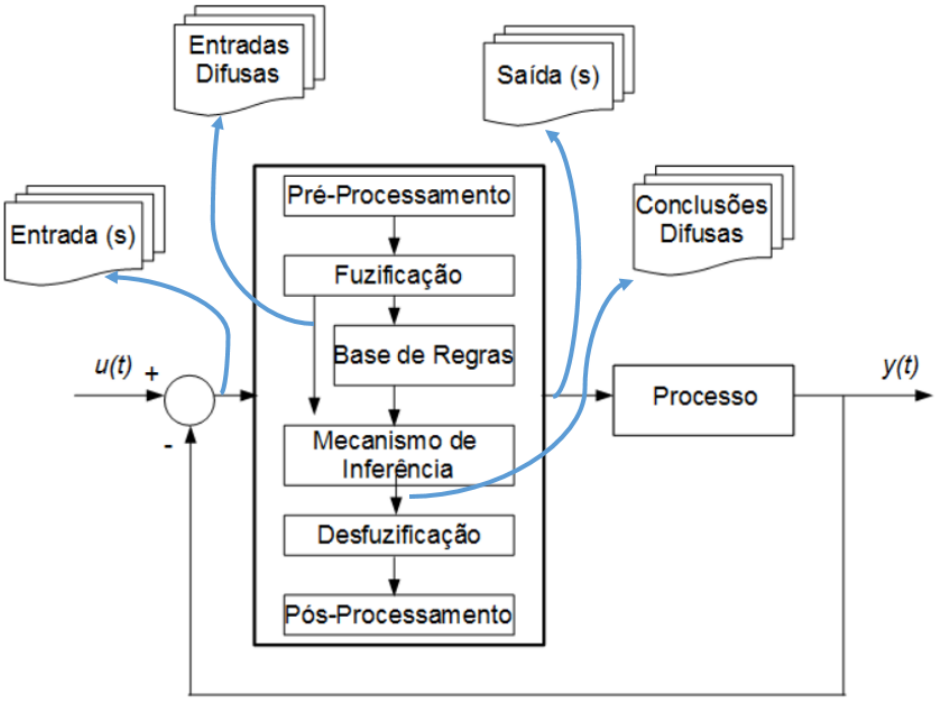


Figura 3.2: Controlador difuso na malha de realimentação.

parcialmente a um determinado conjunto. Esta ambiguidade existente nos conjuntos difusos, constitui uma mais valia na resolução de muitos problemas matemáticos [21]. Um conjunto clássico consiste num conjunto com limites bem definidos. Por exemplo pode-se definir de forma exata o conjunto A, constituído pelos elementos superiores a 10 da seguinte forma:

$$A = \{x | x > 10\} \tag{3.1}$$

Mas se a representação do conjunto A é simples, já a representação do conjunto B constituído pelos elementos x pertencentes ao universo U próximos de 10 é de difícil resolução. Isto, porque não sabemos definir matematicamente o conceito de proximidade. Além disso, o que para uma pessoa pode ser próximo, para outra pessoa pode ser afastado. Contrastando com os conjuntos clássicos, temos os conjuntos difusos que, tal como o nome indica, não possuem limites bem definidos. Nestes conjuntos, a fronteira entre o pertence e o não pertence é feita de forma progressiva e gradual, sendo caracterizada por uma determinada função, designada por função pertença. Podemos então representar o conjunto B por:

$$B = \{(x, \mu_B(x)) | x \in U\} \tag{3.2}$$

Onde $\mu_B(x)$ representa a função de pertença usada para traduzir a evolução do grau de

pertença de x no conjunto B. Esta função faz um mapeamento de cada valor de x do universo U para um valor contínuo no intervalo [22], sendo que este valor é designado pelo grau de pertença. Ou seja, o conjunto B está então dependente da função de pertença usada. Um exemplo possível de uma função que defina o grau de pertença dos elementos x em B é a representada pela equação (3.3) e cujo aspeto se pode visualizar na Figura 3.3.

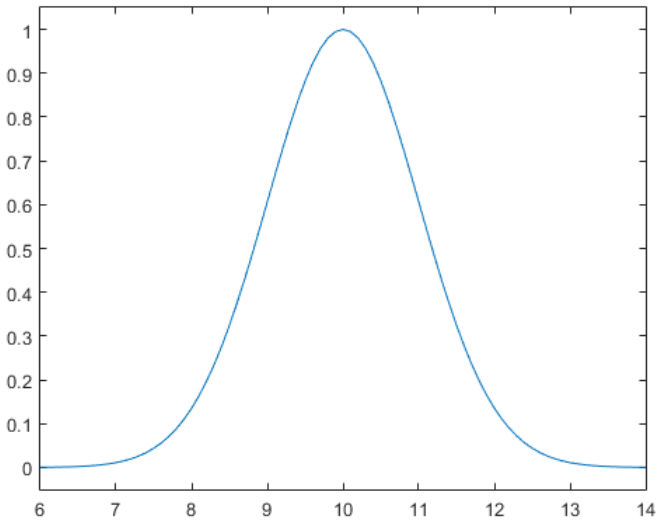


Figura 3.3: Aspeto da função de pertença indicada na equação (3.3).

$$\mu_B(x) = \frac{1}{1 + (x-6)^4} \tag{3.3}$$

Esta situação de indefinição relativamente às fronteiras de um determinado conjunto, acontece frequentemente na realidade. Por exemplo, torna-se difícil definir fronteiras quando se pretende definir grupos para: pessoas altas, casas caras, dias chuvosos, temperaturas baixas, entre outros. Assim, os conjuntos difusos introduzem o conceito de transição contínua e gradual entre membro e não membro, com o objetivo de aumentar a abrangência destes conceitos.

Um outro aspeto importante é o fato de haver a possibilidade de definir várias funções de pertença para o mesmo conjunto difuso. Este aspeto torna-se uma mais valia, pois, quando duas pessoas vão classificar o preço de um imóvel, por exemplo, ambas classificariam de forma diferente a mesma função de pertença. No entanto, a definição dos conjuntos difusos não é arbitrária, pois se para alguns casos uma estimação qualitativa que reflita a ordem dos elementos de um determinado conjunto é suficiente, para outros é necessária uma melhor aproximação que passará pela aplicação de técnicas de aprendizagem, tal como as que são usadas nas redes neuronais.

3.1.2 Variáveis Linguísticas

As variáveis dos sistemas difusos são caracterizadas por serem variáveis, cujos valores não são um número, mas sim uma palavra ou frase. A utilização de uma caracterização linguística em detrimento de uma caracterização numérica prende-se com o facto de a primeira ser menos específica que a segunda. No entanto, em regra geral, a informação linguística não é usada isoladamente, sendo esta na maior parte dos casos combinada com os valores numéricos.

Consideremos a variável linguística u que denota a velocidade de um determinado motor e cujos valores são x, pertencentes a um domínio U. A variável linguística é geralmente decomposta num conjunto de termos $T(u) = \{MuitoBaixa,\ Baixa,\ Normal,\ Alta,\ MuitoAlta\}$ que cobrem o universo de discurso $U = [0;\ 100]$ rpm. Assim, pode-se considerar que a velocidade é muito baixa até 20 rpm, entre 20 rpm e 40 rpm, 40 rpm e 60 rpm, 60 rpm e 80 rpm pode ser considerado baixo, normal e alta, respetivamente e muito alta a partir de 80 rpm. Estes termos podem ser caracterizados por conjuntos difusos, cujas funções pertença são as indicadas na Figura 3.4.

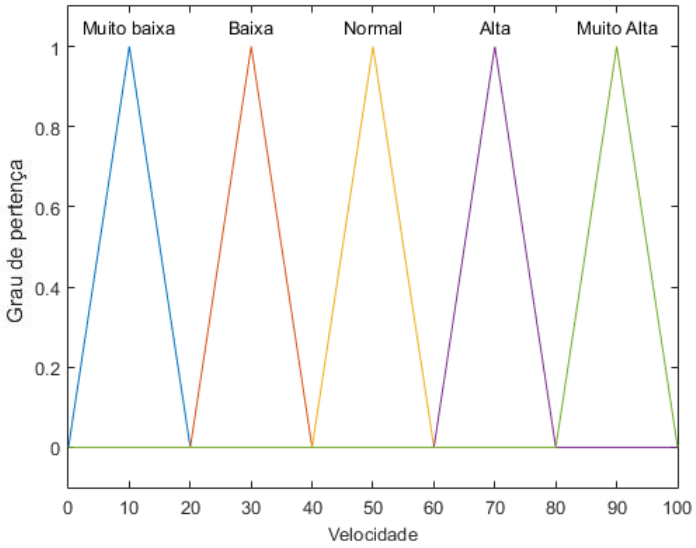


Figura 3.4: Funções de pertença para os termos $T(velocidade)$.

No caso de se ter atribuído outros valores a cada conjunto difuso, poderia acontecer que o mesmo valor de velocidade, poderia pertencer a diferentes conjuntos difusos e, inclusivamente, ter o mesmo grau de pertença em dois conjuntos [23].

3.1.3 Funções de Pertença

A construção das funções de pertença é baseada no bom senso do programador. Uma função de pertença pode ser construída com a forma triangular, trapezoidal, sigmóide, gaussiana ou com a forma de sino. Todas estas formas possuem a característica de apresentar um valor máximo (unitário) para uma região do Universo do Discurso, o qual tende para zero à medida que nos afastamos dessa região caracterizadora do conjunto. Apesar da forma característica de cada uma destas funções, estas possuem um conjunto de parâmetros sintonizáveis, que nos permitem ajustar o seu posicionamento e fatores de forma.

Até à pouco tempo, a seleção do tipo de função a usar, era feita tendo em conta a experiência do utilizador. Com isto, poderia acontecer que dois utilizadores distintos, definissem funções distintas, tendo em conta o conhecimento de cada um. Até mesmo, quando o tipo de função escolhida era o mesmo, estes poderiam-nas dimensionar de formas diferentes, ocupando estas espaços completamente diferentes. No entanto, mais recentemente, as funções são projetadas, recorrendo a procedimentos de otimização que tendem a otimizar a forma da função usada [23].

Um aspeto importante a ter em conta, é o número de funções de pertença usadas para cada variável. Normalmente, o seu número fica a cargo do utilizador. No entanto, sabe-se que quando maior foi o número de funções de pertença, maior será a partição do Universo do Discurso, mas por outro lado, irá trazer mais complexidade e esforço computacional. Outro aspeto a ter em conta, é o fato de estas não terem obrigatoriamente que se sobrepor. No entanto, a grande virtude da lógica difusa reside, precisamente na sua capacidade de descrever ambiguidades, como sejam o copo "meio cheio"ou "meio-vazio". Neste sentido, somos capazes de distribuir as decisões, envolvendo várias classes caracterizadoras da robustez dos sistemas de lógica difusa [23].

De seguida, são apresentadas as definições de cada função de pertença, que são usadas mais regularmente.

Triangular

Esta função é uma das mais frequentemente utilizadas, devido à simplicidade da sua fórmula matemática e à sua eficiência computacional. A função triangular pode ser definida por três parâmetros distintos a,b,c da seguinte forma:

$$\mu_F(x, a, b, c) = max\left(min\left\{\frac{x-a}{b-a}, \frac{c-x}{c-b}\right\}, 0\right) \tag{3.4}$$

Onde a e c representam os pontos extremos da base do triângulo, b representa o centro, *max* a operação máximo e *min* a operação mínimo. De referir que não é obrigatória uma simetria em relação ao centro. A Figura 3.5 mostra um exemplo de uma função triangular μ_F *(x, 10, 40, 60)* [24].

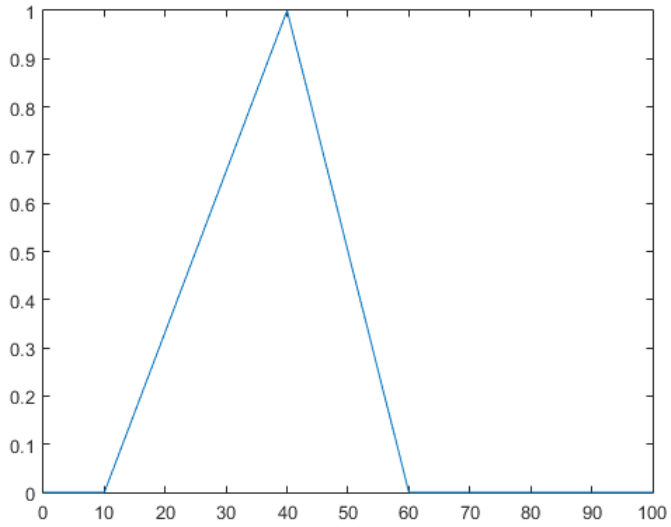


Figura 3.5: Função de pertença triangular com centro em $x = 40$.

Trapezoidal

Pelas mesmas razões da função Triangular, a função de pertença Trapezoidal é também uma das mais usadas. Esta pode ser definida por quatro parâmetros, *a,b,c,d* e representada por:

$$\mu_F(x, a, b, c, d) = max\left(min\left\{\frac{x-a}{b-a}, 1, \frac{d-x}{d-c}\right\}, 0\right) \tag{3.5}$$

Onde a e d representam os extremos da função e b e c o intervalo para o qual o grau de pertença é máximo. Na Figura 3.6 encontra-se representada a função μ_F *(x, 10, 30, 50,70)* [24].

Gaussiana

Esta função apresenta a forma de um gaussiano ou distribuição normal. Esta pode ser especificada por dois parâmetros σ, c através da seguinte expressão:

$$\mu_F(x, \sigma, c) = e^{\left\{-\left[\frac{-0,5(x-c)^2}{\sigma^2}\right]\right\}} \tag{3.6}$$

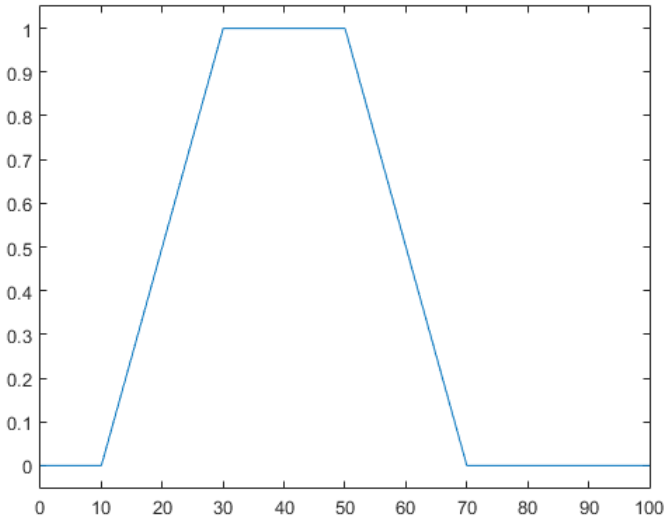


Figura 3.6: Função de pertença trapezoidal.

Onde σ é o desvio padrão da função e c a média. A forma da função gaussiana é a apresentada na Figura 3.7 [24].

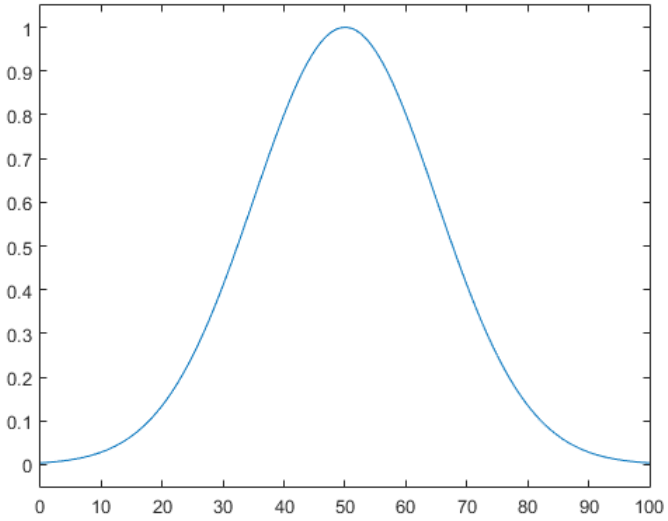


Figura 3.7: Função de pertença do tipo gaussiano para (x, 20, 50).

Sigmóide

A função pertença do tipo sigmóide é definida por:

$$\mu_F(x, a, b) = \frac{1}{1 + e^{[-a(x-b)]}} \tag{3.7}$$

Onde o valor de a controla a inclinação no ponto de cruzamento b. O sinal do parâmetro a define se a função é aberta à direita ou à esquerda e como tal, é apropriada para representar conceitos como "muito alta"ou "muito baixa". A forma da função sigmóide é a apresentada na Figura 3.8 [24].

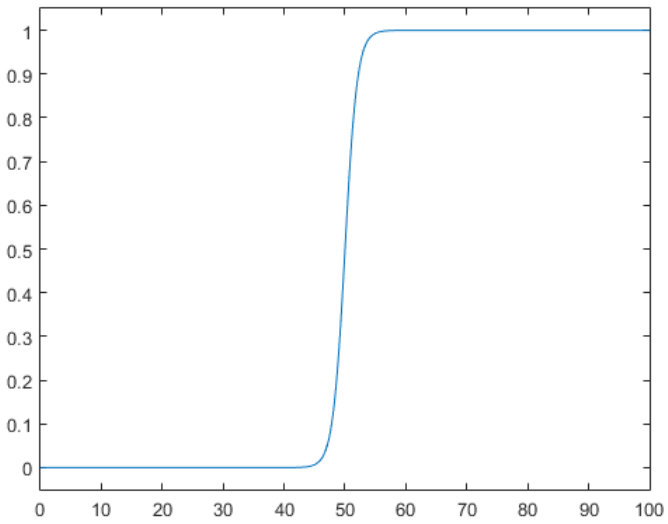


Figura 3.8: Função de pertença sigmóide para $(x,\ 0,5,\ 50)$.

Forma de Sino

Pode ser especificada por três parâmetros a,b,c, segundo a seguinte expressão:

$$\mu_F(x,a,b,c) = \frac{1}{1+|\frac{x-c}{a}|^{2b}} \tag{3.8}$$

Onde c representa o centro, a a largura e b, geralmente definido como sendo positivo, o parâmetro que é usado para controlar a inclinação nos pontos de cruzamento. Na Figura 3.9 está representado uma função com valores de a, b e c de 20, 4 e 50, respetivamente [24].

De referir que poderão ser definidos muitos outros tipos de funções pertença, mais adequadas ao tipo de aplicação pretendido.

3.1.4 Operadores Difusos

Sendo X e Y dois conjuntos e F a relação difusa entre eles, esta é definida da seguinte forma [25]:

$$F = \{((x,y),\mu_F(x,y)) \mid (x,y) \in X \times Y\} \tag{3.9}$$

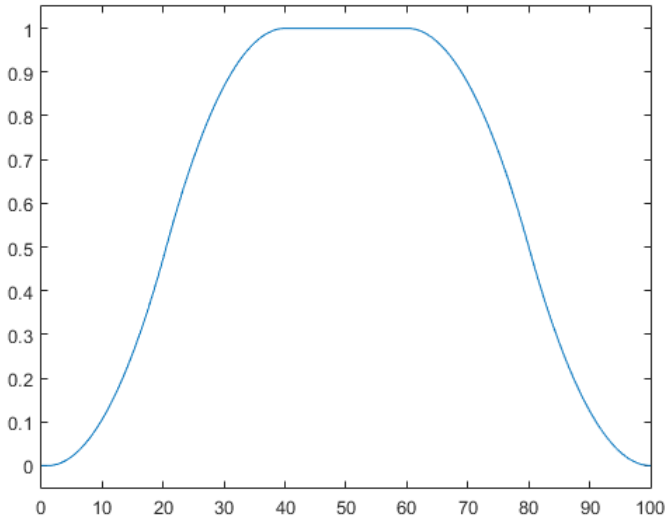


Figura 3.9: Função de pertença em forma de sino para $(x, 20, 4, 50)$.

$$\mu_F : X \times Y[0, 1]$$

Operação de Conjunção ou Intersecção (AND ou T-norma)

Sendo A e B os conjuntos difusos em que $\mathbf{A} \subset \mathbf{X}$ e $\mathbf{B} \subset \mathbf{Y}$, e *(x, y)* $\in XxY$, a operação de conjunção pode ser definida da seguinte forma:

$$\mu_{AND}(x, y) \rightarrow min\left\{\mu_{AND}(x),\ \mu_{AND}(y)\right\} \tag{3.10}$$

Representa uma classe de relações binárias aptas a representarem a operação de intersecção, neste caso o mínimo que devolve o mínimo da intersecção [25]. Na Figura 3.10 é possível observar uma representação da operação de interseção, onde a secção azul e laranja representam as funções de pertença e a secção verde representa a união de ambas as funções.

Operação de Disjunção (OR ou S-norma)

A relação de disjunção pode ser efetuada da seguinte forma:

$$\mu_{OR}(x, y) \rightarrow max\left\{\mu_{OR}(x),\ \mu_{OR}(y)\right\} \tag{3.11}$$

Representa uma classe de relações binárias aptas a representarem a operação de união, neste caso o máximo que devolve o máximo da intersecção [25]. Na Figura 3.11 é possível observar uma representação da operação de disjunção, onde a secção azul e laranja representam as funções de pertença e a secção verde representa a disjunção de ambas as funções.

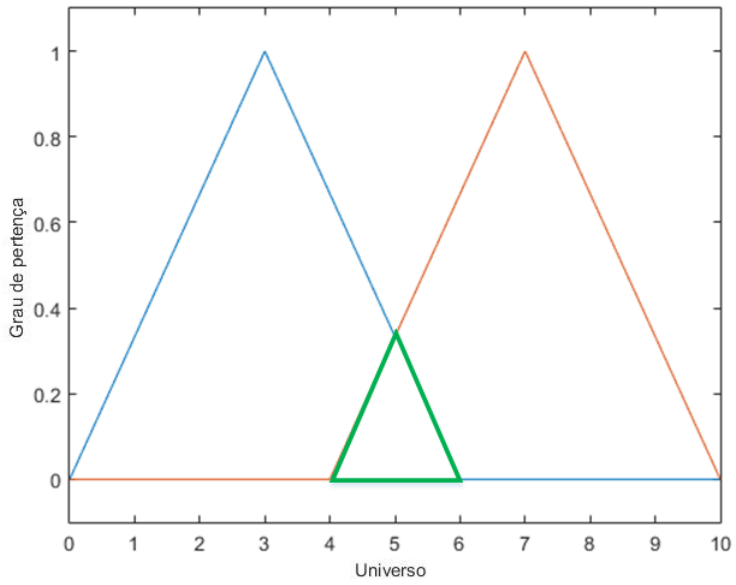


Figura 3.10: Operação de interseção, equivalente ao operador AND.

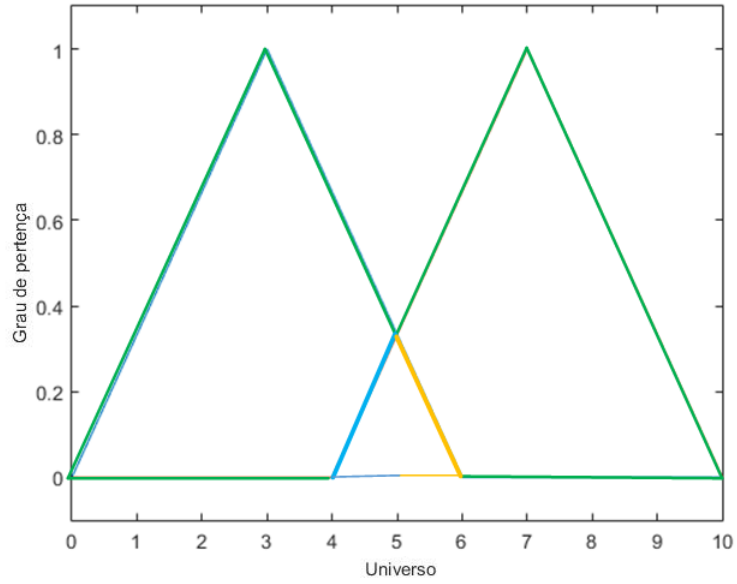


Figura 3.11: Operação de disjunção, equivalente ao operador OR.

Restantes Propriedades dos Operadores

Para que seja possível compreender as restantes propriedades deste tipo de operadores, são enunciadas de seguida as várias propriedades de cada um [25].

Definindo a aplicação da T-norma como $t : [0, 1]\ x\ [0, 1] \rightarrow [0, 1]$, descrevem-se as seguintes propriedades:

1. Monotonicidade: quando $x \leq y$ e $w \leq z$, então $t(x, w) \leq t(y, z)$

2. Comutatividade: $t(x,y) = t(y,x),\ \forall\ x,\ y[0,1]$

3. Associatividade: $t(t(x,y),\ z) = t(x,\ t(y,z)),\ \forall\ x,\ y,\ z[0,1]$

4. Limite ou fronteira : $t(x,0) = 0\ e\ t(x,1) = x\ ,\forall\ \ x\ \in\ [0,1]$

As T-normas são usadas em vez do operador clássico de intersecção $\mu_{A\cap B}(x) = t(\mu_A(x),\ \mu_B(x))$
Eis alguns exemplos de T-normas:

- (Mínimo) $t(x,y) = min(x,\ y)$
- (Diferença limitada) $t(x,\ y) = max(0,\ x+y-1)$
- (Produto algébrico) $t(x,\ y) = x \times y$
- (Produto dástrico) $t(x,\ y) = \begin{cases} min(x,\ y) \text{ quando } max(x,\ y) = 1 \\ 0 \text{ para restantes casos} \end{cases}$

Definindo a aplicação da S-norma como $s\ :\ [0,1]\ \ x\ \ [0,1][0,1]$, descrevem-se as seguintes propriedades:

1. Monotonicidade : quando $x \leq y$ e $w \leq z$, então $s(x,\ w) \leq s(y,\ z)$

2. Comutatividade: $s(x,y) = s(y,\ x),\ \forall\ x,\ y \in [0,\ 1]$

3. Associatividade: $s(s(x,\ y),\ z) = s(x,\ s(y,\ z)),\ \forall\ x,\ y,\ z \in [0,1]$

4. Limite ou fronteira : $s(x,\ 0) = x$ e $s(x,\ 1) = 1,\ \forall\ x \in [0,\ 1]$

As S-normas são usadas em vez do operador clássico de união $\mu_{A\cup B}(x) = s(\mu_A(x), \mu_B(x))$
Eis alguns exemplos de S-normas:

- (Máximo) $s(x,y) = max(x,\ y)$
- (Diferença limitada) $s(x,\ y) = min(1,\ x+y)$
- (Produto algébrico) $s(x,\ y) = x + y - x \times y$
- (Produto dástrico) $s(x,\ y) = \begin{cases} max(x,\ y) \text{ quando } min(x,\ y) = 1 \\ 1 \text{ para restantes casos} \end{cases}$

Existem ainda propriedades importantes dos conjuntos difusos:

- Leis de De Morgan: garantida apenas se a S-norma é derivada de uma T-norma.Ou seja, quando:

$$t(x,y) = 1 - s(1-x,\ 1-y) \tag{3.12}$$

- Lei do Complemento: as relações $A \cup A^C = X$ e $A \cap A^C = 0$, nunca se cumprem.

3.1.5 Estrutura do Controlador Difuso

Assim como outras ferramentas matemáticas, a lógica difusa e os métodos de controlo difusos têm vindo a ser desenvolvidos com o intuito de resolver problemas práticos. Na teoria de controlo de sistemas, se a interpretação difusa de um problema real está correta e se a teoria difusa for desenvolvida corretamente, então os controladores difusos podem ser projetados adequadamente, apresentando várias vantagens aquando o seu funcionamento.

Tal como outras ferramentas matemáticas, a lógica difusa, a teoria dos conjuntos difusos, a modelação difusa e os métodos de controlo difuso, têm sido desenvolvidos com o intuito de solucionar problemas práticos. Nos sistemas de controlo difuso, se a interpretação de um dado problema do mundo real está correto e se a teoria difusa é desenvolvida de forma adequada, então os controladores difusos podem ser projetados e funcionam bastante bem devido às vantagens que apresentam. Todo o processo é então devolvido para o mundo real original, para realizar a automação do sistema pretendido.

Esta é a chamada rotina "Fuzificação - Base de Regas Difusas - Desfuzificação"que é aplicada nos sistemas de controlo difuso. O passo fundamental - operação difusa - é executado por uma regra lógica base, que consiste em regras do tipo "*If-Then*"(Se-Então) que são estabelecidas através de lógica difusa e análise humana do problema físico.

A estrutura geral de um controlador difuso, consiste em três partes fundamentais: a unidade de fuzificação à entrada do controlador, o mecanismo de inferência construído sobre a base de regras que o controlador possui, a unidade de desfuzificação à saída do controlador e um pré-processamento inicial, que será abordado mais à frente. Na Figura 3.12, é possível observar uma estrutura base de um controlador difuso.

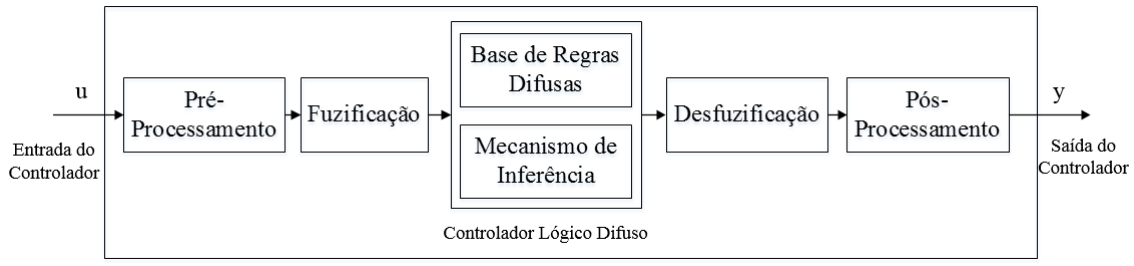


Figura 3.12: Estrutura geral de um controlador difuso.

Pré-Processamento

Normalmente, antes de se converter as entradas numéricas em conjuntos difusos, torna-se necessário realizar filtragem de ruídos, integração e derivação de sinais; normalização ou trans-

formação de determinado universo de discurso Φ_1 para um outro Φ_2, arredondamentos, entre outros. Existe *hardware* que só permite trabalhar com números inteiros, o que pode exigir a alteração do universo de discurso das entradas do controlador difuso como resposta a esta restrição. No entanto, este pré-processamento pode levar o controlador a oscilar em torno da referência ou, até mesmo, à instabilidade.

Fuzificação

São utilizados conjuntos difusos, com o intuito de quantificar a informação da base de regras. Para isso, torna-se necessário especificar como serão convertidas as entradas numéricas $u_i \in \Phi_i$ em conjuntos difusos. O processo de fuzificação é o processo de transformar cada uma das entradas u_i em valores numéricos das funções de pertença, que estão definidas para essas mesmas variáveis. Ou seja, determinar o grau a que pertencem ao conjunto difuso estabelecido, através das funções de pertença. Se for definido Φ_i^{FUZ} o universo de discurso do conjunto de possíveis conjuntos difusos definidos em Φ_i , a fuzificação transforma $u_i \in \Phi_i$ para um conjunto difuso FUZ_i, através de um operador F (equações (3.13) e (3.14)), geralmente do tipo impulso, que gera uma função de pertença que produzirá $u\_FUZ_i(x) \in \Phi_i^{FUZ}$, de acordo com a equação (3.15) [20].

$$F : \Phi_i \rightarrow \Phi_i^{FUZ} \tag{3.13}$$

$$F(u_i) = FUZ_i \tag{3.14}$$

$$u\_FUZ_i(x) = \begin{cases} 1 \ , \ x = u_i \\ 0 \ , \ Outro \end{cases} \tag{3.15}$$

Base de Regras

O bloco designado por base de regras consiste, na realidade, numa base de regras. Ou seja, uma coleção de regras do tipo *If-Then* (Se-Então) que podem ser representadas por:

$$R^l : IF \ u_1 \ is \ F_1^l \ AND \ u_2 \ is \ F_2^l ... u_i \ is \ F_i^l \ THEN \ v \ is \ G^l \tag{3.16}$$

Onde $l = 1, 2, ..., M$, F_i^l e G^l são conjuntos difusos em $U_i \subset \Re$ e $V \subset \Re$, respetivamente, $\mu = col(\mu_1, ..., \mu_i) \ \in \ U_1 \times ... \times U_i$ e $v \in V$. μ é o vetor da variável linguística de entrada, de coordenadas $\mu_i \in U_i$, respetivamente, e v a variável linguística de saída, com $v \in V$. M é o

número total de regras [20].

A base de regras difusas armazena o conhecimento empírico das operações do processo, em termos do domínio de conhecimento. Cada regra l, *If-Then*, define um conjunto difuso $F_1^l \times ... \times F_i^l \rightarrow G^l$ definido no espaço produto $U \times V$. Uma regra deste tipo é interpretada como uma implicação difusa no espaço produto.

Mecanismo de Inferência

O mecanismo de inferência envolve normalmente duas etapas [20]:

- Determinação da relevância de cada regra para o estado das entradas u_i (também designado por *matching*). Ou seja, é o cálculo do grau de satisfação de cada regra;
- Recolha de conclusões, usando o valor das entradas u_i e a informação da base de regras (também designado por *inference step*).

Matching Considerando $CD_1^j \times CD_2^k \times ... \times CD_n^l$ o conjunto difuso representativo da premissa da regra i, o *matching* envolve a execução de dois passos:

1. Combinação das entradas com a premissa das regras. Este ponto envolve a quantificação das premissas, encontrando os conjuntos difusos $MAT_1^j, MAT_2^k, ..., MAT_n^l$ com as funções de pertença:

$$\begin{aligned} u\_MAT_1^j(u_1) =& u\_CD_1^j(u_1) \times u\_FUZ_1^j(u_1) \\ u\_MAT_2^k(u_2) =& u\_CD_2^k(u_2) \times u\_FUZ_2^j(u_2) \\ & ... \\ u\_MAT_n^l(u_n) =& u\_CD_n^l(u_1) \times u\_FUZ_n^n(u_n) \end{aligned} \tag{3.17}$$

   (para todo o j, k ,..., l) que combinam os conjuntos difusos da fuzificação com os conjuntos difusos usados em cada um dos termos das premissas das regras.

2. Determinar a aplicabilidade de cada regra. Ou seja, encontrar as regras relevantes de acordo com o estado atual das entradas. Para isso, são calculados valores de pertença $u_i(u1, u2, ..., u_n)$ para a premissa da regra i, que representam a certeza de cada premissa relativamente ao estado atual das entradas. Quando a fuzificação é efetuada com a função impulso, tem-se:

$$u_i(u1, u2, ..., u_n) = u\_CD_1^j(u_1) \times u\_CD_2^k(u_2) \times ... \times u\_CD_n^l(u_n) \tag{3.18}$$

Onde $u_i(u1, u2, ..., u_n)$ é uma superfície multidimensional que avalia a certeza de cada premissa [20]. Todas as regras têm associado um peso (um número entre 0 e 1) que é aplicado ao valor da premissa.

Inference Step Nesta fase são combinadas as recomendações de todas as regras para construir uma única conclusão. Também aqui são executados dois passos:

1. Aplicação do método de implicação. O peso de cada regra igual a 1 não traduz efeitos na aplicação deste método. O consequente de cada regra é um conjunto difuso representado por uma função de pertença que é reformulada pela premissa. A entrada do processo de implicação é um único resultado numérico dado pela premissa e a saída é um conjunto difuso, que não é mais do que o conjunto difuso do consequente reajustado de acordo com o resultado do *matching*. O objetivo é calcular o "conjunto difuso implicado" IMP_q^i, que especifica a certeza que a saída do processo deva ser y_q dentro do universo de discurso Ψ_q, e que tem como valor de pertença:

$$u\_IMP_q^i(y_q) = u_i(u_1, u_2, ..., u_n) \times u\_CONS_q^p(y_p) \tag{3.19}$$

tendo em conta que $CONS_q^p$ é o conjunto difuso do consequente. Os operadores lógicos utilizados tipicamente são o mínimo e o produto algébrico.

2. Agregação das saídas. A decisão final é baseada no teste de todas as regras R, o que implica que o resultado da aplicação do método de implicação deve ser combinado num único conjunto difuso:

$$u\_IMP_q(y_q) = max\left\{u\_IMP_q^1(y_q), u\_IMP_q^2(y_q), ..., u\_IMP_q^R(y_q) : y_q \in \Psi_q\right\} \tag{3.20}$$

Esta operação apenas pode ocorrer uma vez por cada variável de saída, antecedendo a última fase do processo. A sua entrada é um conjunto de conjuntos difusos que retornaram da aplicação do método de implicação a cada regra. Já a sua saída é simplesmente um conjunto difuso para cada saída do processo y_i.

Desfuzificação

A desfuzificação é a última fase do algoritmo de controlo baseado em lógica difusa. Existe um grande número de estratégias de desfuzificação, onde cada uma delas fornece um meio para

identificar uma saída numérica (não difusa) com base nos resultados da agregação [20]. Ou seja, pretende-se converter as decisões produzidas pelo mecanismo de inferência em ações de controlo "acertadas". A sua entrada é um conjunto difuso (informação de recomendação difusa) e a sua saída é a saída numérica do controlador difuso. O centro de gravidade (COG) é o método mais popular para "desfuzificar"o conjunto difuso recomendado pelo mecanismo de inferência. Se for definido R como sendo o número de regras, CA_i^q como o centro de área das funções de pertença de $CONS_q^p$ associadas aos conjuntos difusos após a aplicação do método de implicação $IMPq^i$ para a regra i, a sua expressão matemática pode-se descrever da seguinte forma:

$$y_q^{COG} = \frac{\sum_{i=1}^{R} CA_i^q \int_{\Psi_q} u\_IMP_q^i(y_q)dy_q}{\int_{\Psi_q} u\_IMP_q^i(y_q)dy_q}, \sum_{i=1}^{R} CA_i^q \int_{\Psi_q} u\_IMP_q^i(y_q)dy_q \neq 0 \tag{3.21}$$

O numerador desta equação será diferente de zero, se existir pelo menos uma regra ativa para todas as possíveis combinações das entradas do sistema difuso, e se os consequentes de todos os conjuntos difusos tiverem área diferente de zero [20].

Na Figura 3.13 pode ser observada uma representação resumida destes conceitos.

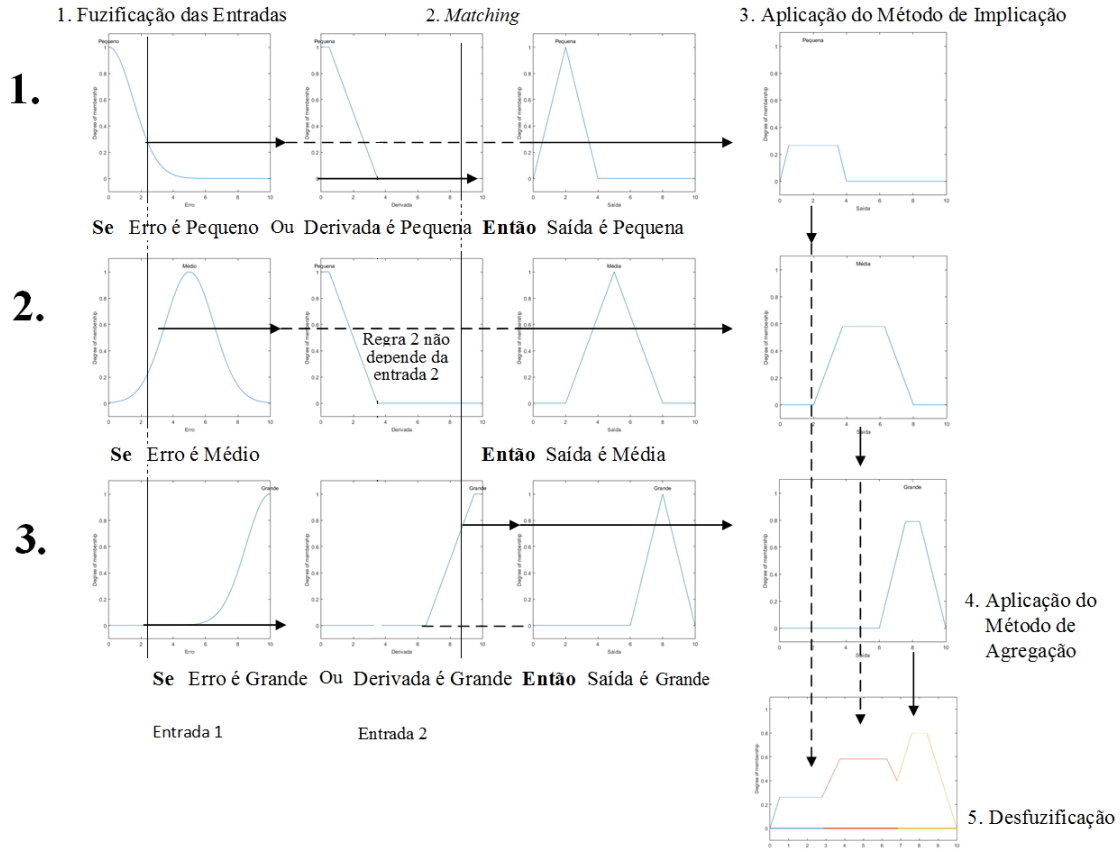


Figura 3.13: Representação gráfica das operações do controlador difuso.

3.2 Controladores PID difusos

Os controladores difusos diferem dos controladores convencionais, devido a utilizarem o conhecimento expresso de uma forma natural e através de um conjunto de regras e variáveis linguísticas, cujos valores são descritos por conjuntos difusos.

A ideia básica subjacente ao controlador PID convencional, passa por escolher a lei de controlo, tendo em conta o erro, o integral do erro e a derivada do erro em ordem ao tempo, de acordo com a equação (2.2), já mencionada no capítulo 2, secção 2.1.1.

Existem dois tipos básicos de controladores de Mamdani que utilizam como variáveis de entrada, o erro e a variação do erro. São estes o controlador PD difuso e o controlador PI, que disponibilizam respetivamente a ação de controlo e o seu incremento. Relativamente ao controlador PID de Mamdani, este pode ser obtido com base na agregação das saídas dos controladores PD e PI.

3.2.1 Controlador PD difuso

O controlador PD difuso utiliza como variáveis linguísticas de entrada, o erro e a variação do erro. Já como variável linguística de saída, apresenta a ação de controlo. A lei de controlo pode ser expressa por:

$$u_k = f(e_k, \Delta e_k) \tag{3.22}$$

Em que $f(\cdot)$ representa o mecanismo interno do controlo difuso, que permite transformar as variáveis de entrada em variáveis de saída.

A estrutura do controlador PD difuso encontra-se na Figura 3.14, onde K_e e $K_{\Delta e}$ são os fatores de escala associados às entradas e K_u o fator de escala associado à saída. Os fatores de escala têm um papel idêntico ao dos ganhos nos controladores convencionais, sendo que possuem um papel fundamental no desempenho e estabilidade do controlador [26]. Isto é, por vezes podem ser fonte de instabilidade e oscilações.

As regras do controlador apresentam a seguinte forma:

$$Regra\ i: Se\ E\ é\ LE^{(i)}\ e\ \Delta E\ é\ L\Delta E^{(i)}\ então\ U\ é\ LU^{(i)} \tag{3.23}$$

Em que $LE^{(i)}$, $L\Delta E^{(i)}$ e $LU^{(i)}$ são os valores linguísticos das variáveis E, ΔE e U, respetivamente.

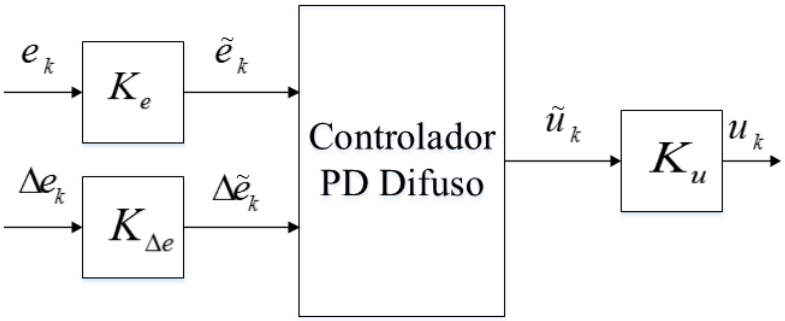


Figura 3.14: Estrutura do controlador PD Difuso.

3.2.2 Controlador PI difuso

O controlador PI difuso é caracterizado por admitir como variáveis de entrada, o erro e a variação do erro, e como variáveis de saída o incremento da ação de controlo. A ação de controlo pode ser expressa do seguinte modo:

$$\Delta u_k = f(e_k, \Delta e_k) \tag{3.24}$$

Em que $f(\cdot)$ representa o mecanismo interno do controlador.

A estrutura do controlador PI difuso encontra-se na Figura 3.15. K_e e $K_{\Delta e}$ são os fatores de escala associados às entradas e $K_{\Delta u}$ o fator de escala associado à saída.

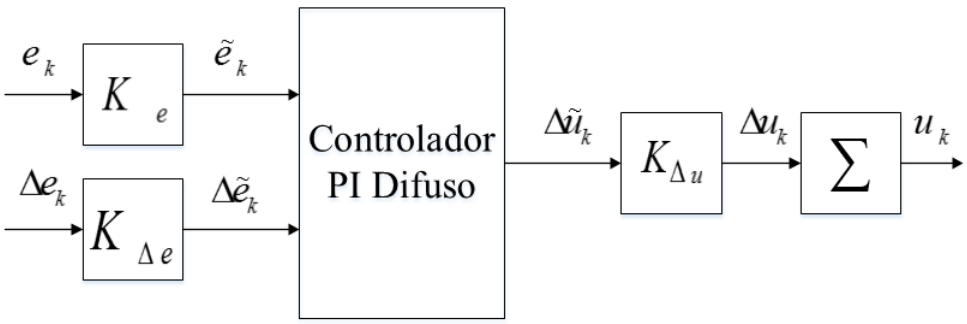


Figura 3.15: Estrutura do controlador PI Difuso.

As regras do controlador apresentam a seguinte forma:

$$Regra\ i: Se\ E\ é\ LE^{(i)}\ e\ \Delta E\ é\ L\Delta E^{(i)}\ então\ \Delta U\ é\ \Delta LU^{(i)} \tag{3.25}$$

Em que $LE^{(i)}$, $L\Delta E^{(i)}$ e $\Delta LU^{(i)}$ são os valores linguísticos das variáveis E, ΔE e ΔU, respetivamente.

Os mecanismos de inferência dos controladores PI e PD possuem funcionamentos análogos, residindo a única diferença na variável de saída. A seleção do tipo de controlo a utilizar é feita de acordo com o seu comportamento em regime transitório e estacionário. Aquando a utilização de um controlador PD, é de extrema dificuldade a eliminação do erro em regime estacionário. Já com a utilização de um controlador PI, torna-se impossível obter um bom desempenho em regime transitório para sistemas de ordem elevada, em virtude da operação interna de integração [26].

3.2.3 Controlador PID difuso

O controlador PID difuso pode ser construído, através da adição de uma terceira variável linguística, para além do erro e da variação do erro. Esta nova variável traduz a segunda diferença do erro, com a variável linguística de saída consistindo na variação da ação de controlo. A lei de controlo do controlador PID difuso pode ser expressa por:

$$\Delta u_k = f(e_k, \Delta e_k, \Delta^2 e_k) \tag{3.26}$$

Em que $f(\cdot)$ representa o mecanismo interno do controlador difuso.

Relativamente à topologia do controlador PID difuso, esta tem a estrutura apresentada na Figura 3.16, onde K_e, $K_{\Delta e}$ e $K_{\Delta^2 e}$ são os fatores de escala associados às entradas e $K_{\Delta u}$ o fator de escala relativo à saída.

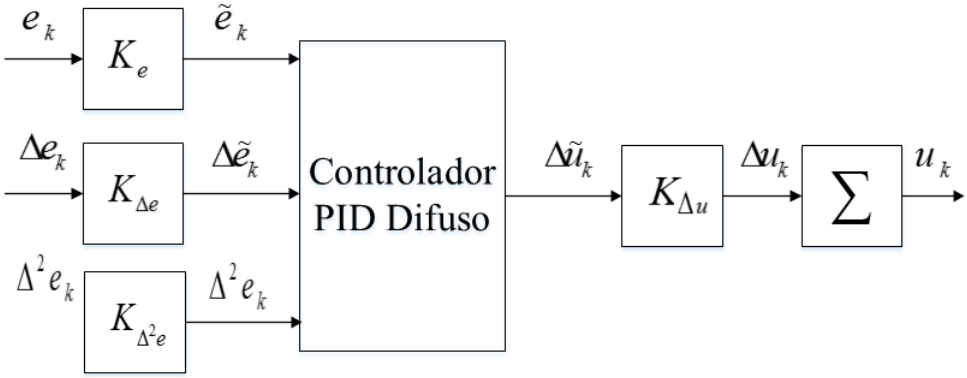


Figura 3.16: Estrutura do controlador PID Difuso.

As regras do controlador PID apresentam a seguinte forma:

$$Regra\ i: Se\ E\ é\ LE^{(i)}\ e\ \Delta E\ é\ L\Delta E^{(i)}\ e\ \Delta^2 E\ é\ L\Delta^2 E^{(i)} então\ \Delta U\ é\ \Delta LU^{(i)} \tag{3.27}$$

Em que $LE^{(i)}$, $L\Delta E^{(i)}$, $L\Delta^2 E^{(i)}$ e $\Delta LU^{(i)}$ são os valores linguísticos das variáveis E, ΔE, $\Delta^2 E$ e ΔU, respetivamente.

3.3 Lógica Difusa no MATLAB

O MATLAB é um software interativo de alto desempenho, sendo que é muito utilizado para cálculo numérico. Este integra análise numérica, cálculo com matrizes, processamento de sinais e construção de gráficos em ambiente fácil de usar, onde os problemas e as soluções são expressas apenas matematicamente, ao contrário da programação tradicional. O MATLAB é caracterizado pelo seu sistema interativo, cujo elemento básico de informação é uma matriz que não requer dimensionamento. Este sistema apresenta a grande vantagem de resolução de muitos problemas numéricos em frações de tempo muito reduzidas.

É possível efetuar o controlo de sistemas com lógica difusa, através do MATLAB, recorrendo a três métodos distintos: fazendo uso da *Fuzzy Toolbox*, pela linha de comandos ou então por Simulink.

3.3.1 *Fuzzy Toolbox*

Aquando a utilização desta *Toolbox* disponibilizada pelo MATLAB, será necessário usar as seguintes ferramentas de construção, edição e visualização de sistemas de inferência difusa (Figura 3.17) [18].

- **Interface do *Fuzzy Logic*** - Esta interface é utilizada para lidar com os problemas de alto nível no sistema, relativamente às variáveis de entrada, de saída, nomes de variáveis, entre outros. O *software* não limita o número de entradas. No entanto, pode ser limitado pela memória disponível no *hardware*.

- **Editor das Funções de Pertença** - Esta interface é utilizada para definir as formas de todas as funções de pertença, associadas a cada variável introduzida pelo utilizador.

- **Editor de Regras** - Esta interface é utilizada para editar a lista de regras, que definem o comportamento do sistema.

- **Visualizador de regras** - Esta interface é utilizada para visualizar o diagrama de inferência difusa. Ou seja, para ver quais regras estão ativas e a influência de cada função no resultado final, por exemplo.

- **Visualizador de Superfície** - Esta interface é utilizada para visualizar a dependência

de uma das saídas sobre as entradas. Isto é, é gerado um mapa da superfície de saída para o sistema.

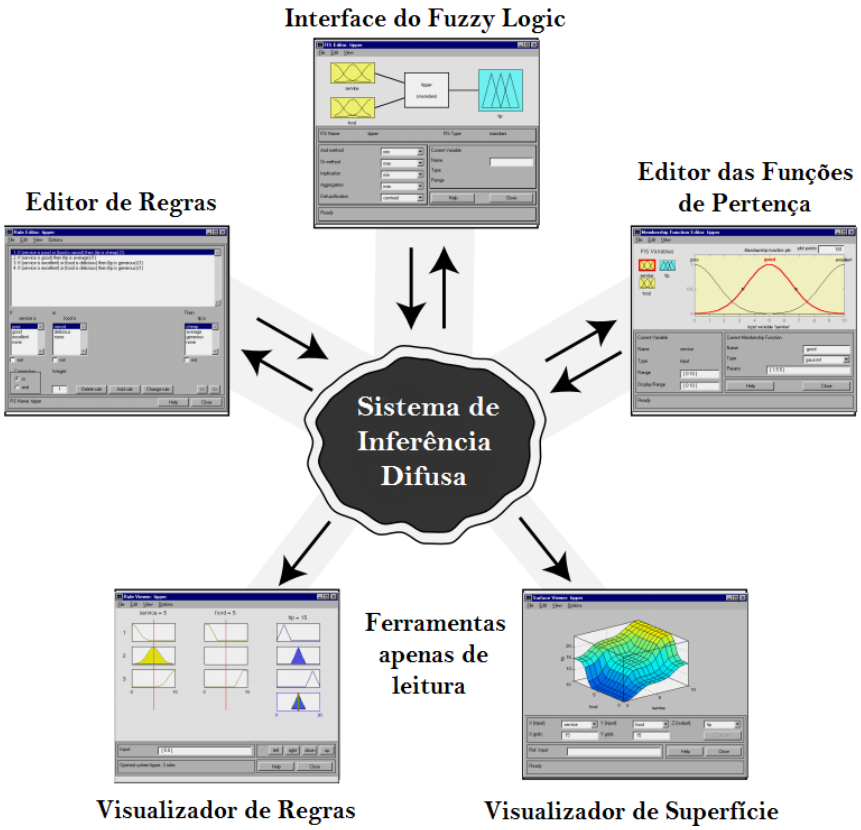


Figura 3.17: Estrutura da *Toolbox* de lógica difusa no MATLAB [18].

No Anexo A é possível encontrar um pequeno manual de funcionamento sobre a lógica difusa no MATLAB/Simulink. Deste faz parte a implementação de lógica difusa através da *Toolbox* fornecida pela MathWorks, através de Simulink e por linha de comandos.

Capítulo 4

Desenvolvimento e Implementação

Este capítulo tem como principal objetivo dar a conhecer todo o sistema de controlo criado durante este processo de pesquisa e desenvolvimento. Será descrito em detalhe cada componente que compõe cada controlador, desde a sua arquitetura até ao nível de *software*. Ao nível da sua arquitetura será abordado cada componente que compõe cada controlador. Ao nível de *software* serão apresentadas as tabelas de regras lógicas difusas que representam a lógica de controlo e ditam o seu comportamento em todas as situações.

4.1 Sistema de Controlo PID

Sendo o controlo PID a forma mais comum de controlo por realimentação, devido à sua simplicidade e excelente desempenho em muitas aplicações, estes controladores são usados em 95% dos processos industriais [27]. Sendo assim, este foi primeiro controlador elaborado neste trabalho, com base na configuração apresentada anteriormente na Figura 2.1.

Como já mencionado anteriormente, a expressão de um controlo clássico PID pode ser dada pela expressão (4.1).

$$\begin{aligned} u(t) &= K_p \left[e(t) + \frac{1}{T_i} \int_0^t e(\tau) d\tau + T_d \frac{de(t)}{dt} \right] \\ &= K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de(t)}{dt} \end{aligned} \tag{4.1}$$

Onde:

$$K_i = \frac{K_p}{T_i} \quad e \quad K_d = K_p T_d \tag{4.2}$$

Sendo os parâmetros K_p, K_i e K_d o ganho proporcional, integral e derivativo, respetivamente. Estes são os ganhos submetidos para a sintonia do controlo PID.

4.1.1 Método de Sintonia de Ziegler-Nichols em Malha Fechada

De modo a fazer a sintonia dos ganhos do controlador PID, foram utilizados dois métodos distintos. Inicialmente foi usado o método de sintonia de Ziegler-Nichols em malha fechada, onde o ganho crítico (K_c) é o ganho necessário e suficiente para o processo entrar numa oscilação periódica. Nesta abordagem, os parâmetros do controlador são calculados a partir do ganho crítico do sistema e do seu respetivo período de oscilação crítica (T_c). Estes parâmetros são obtidos aumentando gradualmente o ganho do controlador proporcional, até que a saída do sistema fechado comece a oscilar com uma amplitude constante. Ou seja, quando a malha de controlo está no limite da estabilidade [27]. De seguida, o ganho e o período críticos são guardados e os parâmetros do controlador PID são determinados por relações pré-determinadas.

Para calcular os ganhos do controlador, foi usado o método de resposta em malha fechada, apresentado na Tabela 4.1.

Tabela 4.1: Parâmetros do controlo PID pelo método de sintonia em malha fechada.

Tipo de Controlador	K_p	T_i	T_d
P	$0,5K_c$	-	-
PI	$0,4K_c$	$0,8T_c$	-
PID	$0,6K_c$	$0,5T_c$	$0,125T_c$

4.1.2 *pidtune* da MathWorks

Numa segunda estância, foi utilizado o algoritmo fornecido pela MathWorks. O *pidtune* é um algoritmo que apresenta respostas bastante razoáveis para modelos lineares e também mais otimizadas comparativamente com o método de Ziegler-Nichols. Normalmente, faz parte dos objetivos de uma sintonia PID típica três aspetos, sendo eles: Estabilidade em malha fechada, desempenho e robustez adequados.

O algoritmo de sintonia usado pelo *pidtune* tem em conta esses três aspetos, de modo a alcançar um bom equilíbrio na resposta do sistema. Por norma, o algoritmo escolhe uma frequência de *crossover* com base na dinâmica do sistema a controlar e com uma margem de fase de 60º. Sempre que é alterado o tempo de resposta, a largura de banda, a resposta transitória ou a margem de fase com recurso à interface *pidtuner*, o algoritmo calcula novos ganhos do PID [28]. Para um dado valor de robustez (margem de fase mínimo), o algoritmo de sintonia escolhe um tipo de controlador que equilibra as duas medidas de desempenho, ou seja, o seguimento de referência e rejeição a distúrbios.

A sintaxe usada é a seguinte:

```
C = pidtune(sys,C0)
```

O parâmetro *sys* representa o sistema dinâmico utilizado para fazer o controlo e C_0 as propriedades definidas para o controlador projetado [29]. Neste caso, C_0 representa um controlo PID com a seguinte estrutura:

```
C = pid(Kp,Ki,Kd)
```

Onde K_p, K_i e K_d representam os ganhos do controlador. Este tipo de controlador tem o modelo da equação (4.3).

$$C = K_p + \frac{K_i}{s} + K_d s \tag{4.3}$$

Relativamente aos argumentos de saída, se o segundo argumento do *pidtune* for C_0, então a saída C será do mesmo tipo que C_0. Assim como a saída também terá o mesmo domínio de tempo que o argumento de entrada *sys* [29].

Na Figura 4.1 é possível observar o esquema relativo ao controlador PID.

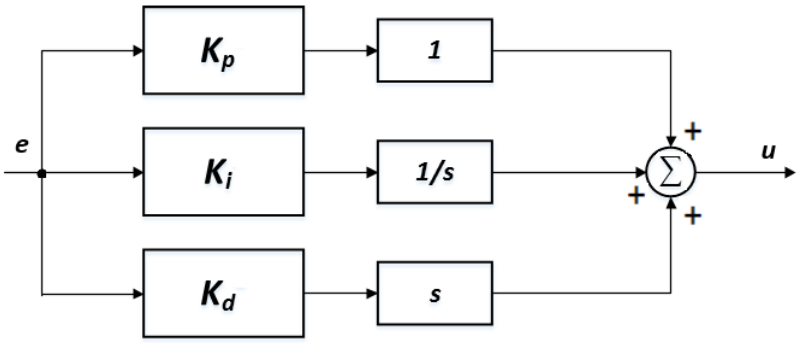


Figura 4.1: Esquema desenvolvido para o controlador PID.

No Anexo B é possível observar o esquema desenvolvido em Simulink para o controlador PID.

4.2 Sistemas de Controlo PID-Difuso Adaptativo

Nesta secção são abordados os dois tipos de controladores PID-Difusos adaptativos desenvolvidos neste trabalho, sendo que estes possuem o melhor de dois controladores bem conhecidos: o controlador lógico difuso e o controlador PID.

4.2.1 Primeiro Controlador

Na Figura 4.2 é possível observar o esquema proposto para o primeiro controlador PID-Difuso adaptativo desenvolvido (adaptado de [30]).

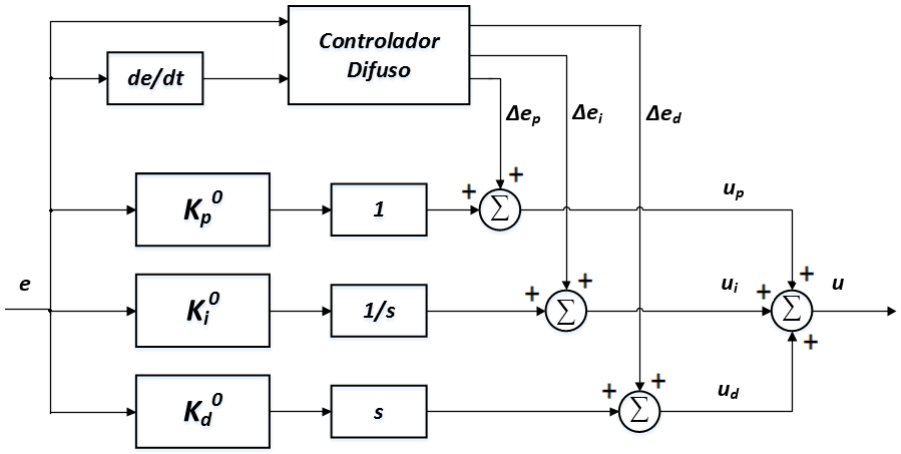


Figura 4.2: Esquema do primeiro controlador PID-Difuso adaptativo proposto.

Este controlador PID-Difuso adaptativo pode ser definido da seguinte forma:

$$\begin{aligned} u(t) &= \left[K_p^0 e(t) + \Delta e_p(t)\right] + \left[K_i^0 \int_0^t e(\tau)d(\tau) + \Delta e_i(t)\right] + \left[K_d^0 \frac{de(t)}{dt} + \Delta e_d(t)\right] \\ &= K_p^0 e(t) + K_i^0 \int_0^t e(\tau)d(\tau) + K_d^0 \frac{de(t)}{dt} + \Delta e_p(t) + \Delta e_i(t) + \Delta e_d(t) \\ &= u^0(t) + \Delta u(t) \end{aligned} \tag{4.4}$$

Onde:

$$u^0(t) = K_p^0 e(t) + K_i^0 \int_0^t e(\tau)d(\tau) + K_d^0 \frac{de(t)}{dt} \tag{4.5}$$

Os parâmetros K_p^0, K_i^0 e K_d^0 são pré-sintonizados e são valores invariantes enquanto o controlador está em funcionamento. Já os parâmetros $\Delta e_p(t)$, $\Delta e_i(t)$ e $\Delta e_d(t)$ são variantes no tempo e adaptados em tempo real.

Para o controlador difuso, a sua saída de controlo pode ser dada por:

$$\Delta u(t) = \Delta e_p(t) + \Delta e_i(t) + \Delta e_d(t) \tag{4.6}$$

Assim, para projetar um controlador adaptativo de modo a obter os sinais $\Delta e_p(t)$, $\Delta e_i(t)$ e $\Delta e_d(t)$, foi usado um controlador difuso. Este controlador difuso é constituído por duas entradas e três saídas. Uma das entradas é dada pelo sinal do erro $e(t)$ e a segunda entrada pela derivada

do sinal do erro $ec(t)$. Já as suas saídas são dadas pelos parâmetros $\Delta e_p(t)$, $\Delta e_i(t)$ e $\Delta e_d(t)$, respetivamente.

No Anexo C é possível observar o esquema desenvolvido em Simulink para o primeiro controlador PID-Difuso adaptativo.

4.2.2 Segundo Controlador

Na Figura 4.3 é possível observar o esquema proposto para o segundo controlador PID-Difuso adaptativo desenvolvido (adaptado de [31]).

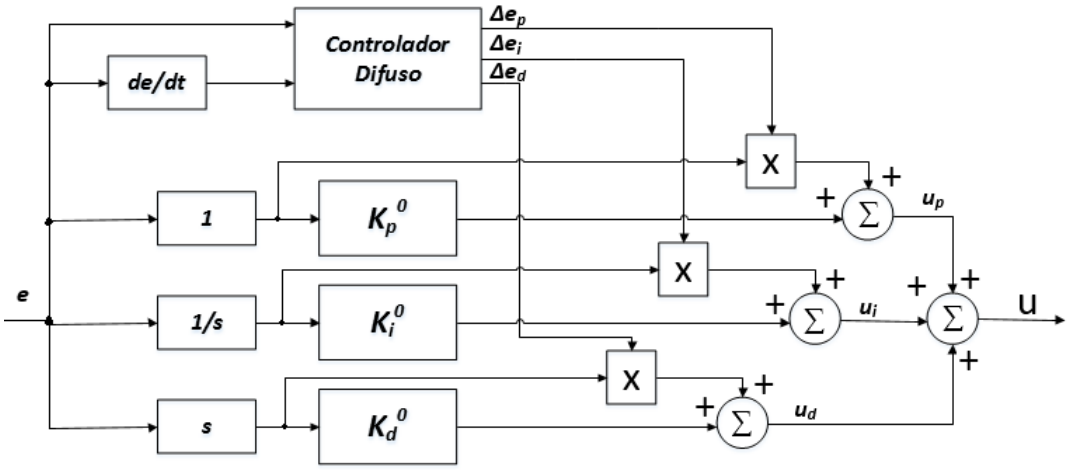


Figura 4.3: Esquema do segundo controlador PID-Difuso adaptativo proposto.

Este controlador PID-Difuso adaptativo pode ser definido da seguinte forma:

$$\begin{aligned} u(t) &= \left[\left(K_p^0 e(t)\right) + \left(\Delta e_p(t)\right)\right] + \left[\left(K_i^0 \int_0^t e(\tau)d\tau\right) + \left(\Delta e_i(t) \int_0^t e(\tau)d\tau\right)\right] + \\ &\quad + \left[\left(K_d^0 \frac{de(t)}{dt}\right) + \left(\Delta e_d(t) \frac{de(t)}{dt}\right)\right] \\ &= \left[K_p^0 + \Delta e_p(t)\right] e(t) + \left[K_i^0 + \Delta e_i(t)\right] \int_0^t e(\tau)d\tau + \left[K_d^0 + \Delta e_d(t)\right] \frac{de(t)}{dt} \end{aligned} \tag{4.7}$$

Os parâmetros K_p^0, K_i^0 e K_d^0 são pré-sintonizados e são valores invariantes no tempo, enquanto o controlador está em funcionamento. Já os parâmetros $\Delta e_p(t)$, $\Delta e_i(t)$ e $\Delta e_d(t)$ são variantes no tempo e adaptados em tempo real, sendo estes os valores obtidos da saída do controlador difuso. Assim, a adaptação de $\Delta e_p(t)$, $\Delta e_i(t)$ e $\Delta e_d(t)$ significa também a adaptação da saída do controlador.

Assim como no primeiro controlador PID-Difuso adaptativo, foi usado um controlador difuso como base para projetar este segundo controlador adaptativo. Este controlador difuso é também

constituído por duas entradas e três saídas. Uma das entradas é dada pelo sinal do erro $e(t)$ e a segunda entrada pela derivada do sinal do erro $ec(t)$. Já as suas saídas são dadas pelos parâmetros $\Delta e_p(t)$, $\Delta e_i(t)$ e $\Delta e_d(t)$, respetivamente.

No Anexo D é possível observar o esquema desenvolvido em Simulink para o segundo controlador PID-Difuso adaptativo.

4.3 Regras Difusas

Neste trabalho, todos os intervalos difusos das entradas e saídas do controlador difuso são dados por $\{NB,\ NM,\ NS,\ ZE,\ PS,\ PM,\ PB\}$, correspondendo a *Negative Big*, *Negative Medium*, *Negative Small*, *Zero*, *Positive Small*, *Positive Medium* e *Positive Big*, respetivamente. Relativamente aos universos de discurso, foram atribuídos os intervalos $[-6,\ 6]$ às entradas e e ec e $[-1,\ 1]$ às saídas $\Delta e_p(t)$, $\Delta e_i(t)$ e $\Delta e_d(t)$. Estes valores foram atribuídos por método empírico, sendo escolhidos por terem sido os que forneceram melhores resultados. Na Figura 4.4 é possível observar as funções de pertença usadas para as entradas e e ec e na Figura 4.5 as funções de pertença atribuídas a $\Delta e_p(t)$, $\Delta e_i(t)$ e $\Delta e_d(t)$. Nas Tabelas 4.2, 4.3 e 4.4 estão expostas as regras atribuídas aos parâmetros Δe_p, Δe_i e Δe_d, respetivamente.

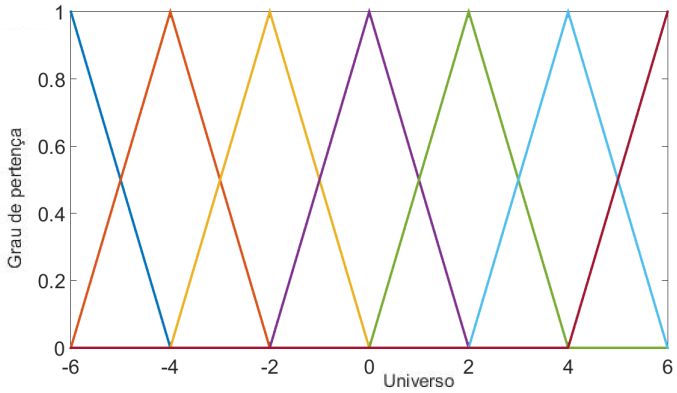


Figura 4.4: Funções de pertença atribuídas aos parâmetros $e(t)$ e $ec(t)$.

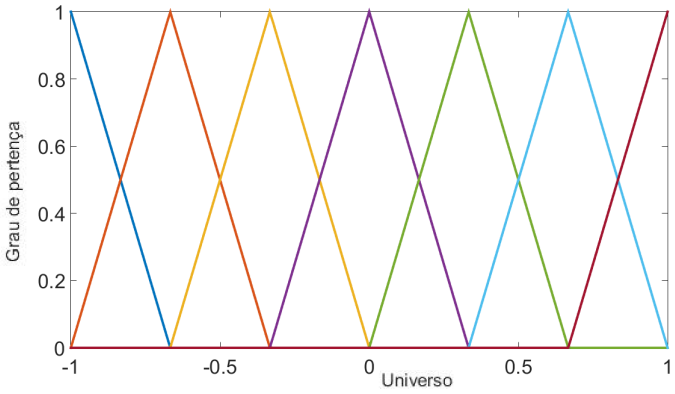


Figura 4.5: Funções de pertença atribuídas aos parâmetros $\Delta e_p(t)$, $\Delta e_i(t)$ e $\Delta e_d(t)$.

Tabela 4.2: Regras para Δe_p.

Δe_p							
ec/e	**NB**	**NM**	**NS**	**ZE**	**PS**	**PM**	**PB**
NB	PB	PB	PM	PM	PS	PS	ZE
NM	PB	PB	PM	PM	PS	ZE	ZE
NS	PM	PM	PM	PS	ZE	NS	NM
ZE	PM	PS	PS	ZE	NS	NM	NM
PS	PS	PS	ZE	NS	NS	NM	NM
PM	ZE	ZE	NS	NM	NM	NM	NB
PB	ZE	NS	NS	NM	NM	NB	NB

Tabela 4.3: Regras para Δe_i.

Δe_i							
ec/e	**NB**	**NM**	**NS**	**ZE**	**PS**	**PM**	**PB**
NB	NB	NB	NB	NM	NM	ZE	ZE
NM	NB	NB	NM	NM	NS	ZE	ZE
NS	NM	NM	NS	NS	ZE	PS	PS
ZE	NM	NS	NS	ZE	PS	PS	PM
PS	NS	NS	ZE	PS	PS	PM	PM
PM	ZE	ZE	PS	PM	PM	PB	PB
PB	ZE	ZE	PS	PM	PB	PB	PB

Tabela 4.4: Regras para Δe_d.

Δe_d							
ec/e	**NB**	**NM**	**NS**	**ZE**	**PS**	**PM**	**PB**
NB	PS	PS	ZE	ZE	ZE	PB	PB
NM	NS	NS	NS	NS	ZE	NS	PM
NS	NB	NB	NM	NS	ZE	PS	PM
ZE	ZE	ZE	ZE	ZE	ZE	ZE	ZE
PS	NB	NM	NS	NS	ZE	PS	PS
PM	NM	NS	NS	NS	ZE	PS	PS
PB	PS	ZE	ZE	ZE	ZE	PB	PB

De acordo com as regras difusas apresentadas nas tabelas anteriores, a sintonia dos parâmetros Δe_p, Δe_i e Δe_d segue as seguintes três regras fundamentais [30]:

1. Se $|e|$ é alto, então Δe_p deve ser alto e Δe_d deve ser pequeno, para que o sistema responda rapidamente. No entanto, a ação integral deve ser limitada, normalmente $\Delta e_i = 0$, de modo a evitar um *overshoot* elevado.

2. Se $|e|$ é moderado, então Δe_p deve ser pequeno. Isto, porque o parâmetro Δe_d é o mais importante na obtenção de um *overshoot* reduzido.

3. Se $|e|$ é pequeno, então Δe_p e Δe_i devem ser altos para proporcionar ao sistema uma resposta em regime permanente mais aceitável. Quando $|ec|$ é alto, Δe_d deve ser pequeno. Deste modo, o sistema consegue evitar oscilações perto do *setpoint*.

Relativamente às regras criadas para fazer o controlo do sistema, estas seguem a seguinte sintaxe:

Se e é NB e ec é NB, então Δe_p é PB, Δe_i é NB e Δe_d é PS.

Nas Figuras 4.6, 4.7 e 4.8 é possível observar as superfícies de controlo de cada variável Δe_p, Δe_i e Δe_d, respetivamente.

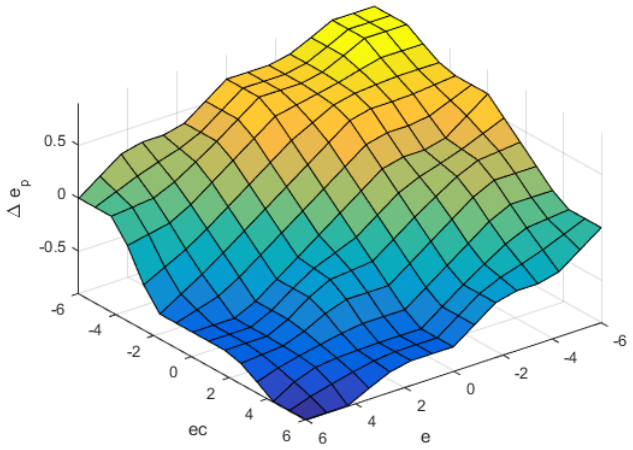


Figura 4.6: Superfícies de controlo da variável Δe_p.

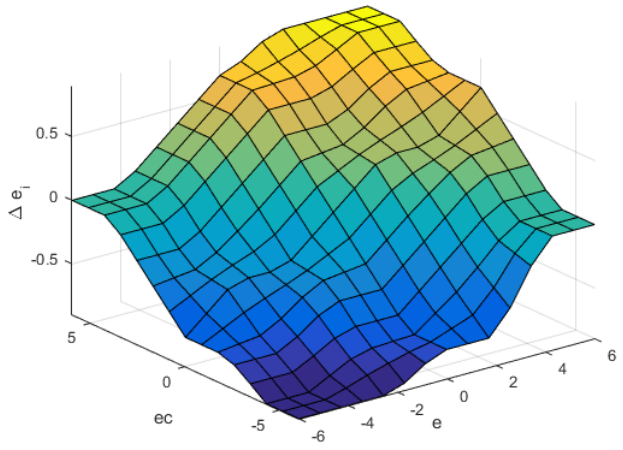


Figura 4.7: Superfícies de controlo da variável Δe_i.

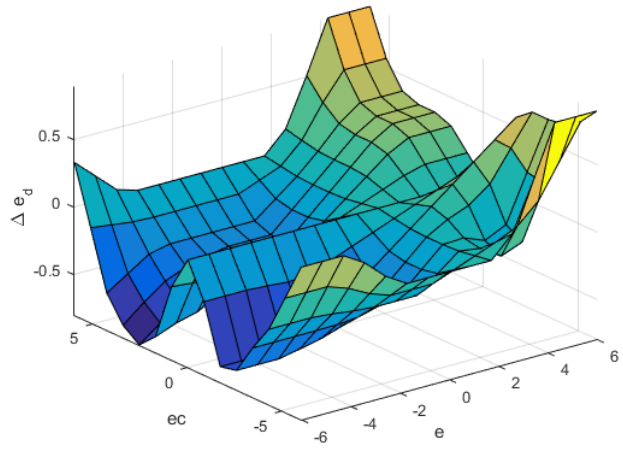


Figura 4.8: Superfícies de controlo da variável Δe_d.

4.4 Índices de Desempenho

A medição do erro permite fazer uma avaliação de como a sintonia do sistema está a ser avaliada. Se a sintonia for realizada corretamente, o erro é baixo. No entanto, se a sintonia for fraca, o valor do erro será maior. Tendo isso em mente e considerando a resposta do sistema, os índices de desempenho podem ser alcançados através da medição do erro da resposta em degrau do sistema.

Existem vários métodos de medição do índice de desempenho. No entanto, este trabalho foi focado em quatro índices de desempenho, sendo eles [32] [33]:

1. Integral do tempo multiplicado pelo erro absoluto ($ITAE$);
2. Integral do tempo multiplicado pelo quadrado do erro ($ITSE$);
3. Integral do erro absoluto (IAE);

4. Integral do erro quadrático (ISE).

O índice de desempenho $ITAE$ é usado para minimizar o impacto do erro inicial da resposta em degrau e para amplificar o erro em estado estacionário. Este índice de desempenho é o que oferece a melhor seletividade e cuja expressão matemática é:

$$ITAE = \int_0^{Tsim} t|e(t)dt \tag{4.8}$$

Um outro critério usado, sendo este semelhante ao $ITAE$, é o $ITSE$. A sua fórmula matemática é dada por:

$$ITSE = \int_0^{Tsim} te^2(t)dt \tag{4.9}$$

O critério IAE mede o desvio absoluto do erro de referência ao longo do tempo, sendo a sua expressão matemática dada por:

$$IAE = \int_0^{Tsim} |e(t)dt \tag{4.10}$$

Por fim, também foi usado o critério ISE. A principal desvantagem deste índice, é o facto de amplificar grandes erros e, consequentemente, levar o sistema a grandes oscilações. A sua representação matemática é dada por:

$$ISE = \int_0^{Tsim} e^2(t)dt \tag{4.11}$$

Em todos os índices de desempenho, o parâmetro T_{sim} corresponde ao tempo de simulação e $e = r - y$ é o erro, como a diferença entre o sinal de referência e a resposta de saída.

4.5 Algoritmo de Minimização do Erro

De modo a utilizar os índices de desempenho mencionados anteriormente para otimizar as respostas dos sistemas a controlar, foi utilizado o algoritmo de minimização *fminsearch* do MATLAB. Este algoritmo tem como base o método *Nelder-Mead*, sendo este um método numérico bastante comum na descoberta dos valores máximos e mínimos de uma função objetivo. Este é normalmente aplicado a problemas de otimização não-lineares, onde as suas derivadas não são possíveis de conhecer. No entanto, o método *Nelder-Mead* é considerado uma heurística de pesquisa que pode eventualmente convergir para valores não estacionários [34][35].

Este método aproxima-se de um ótimo local de um dado problema com n variáveis quando

a função objetivo varia suavemente e é unimodal. Uma possível variação deste algoritmo para tentar minimizar uma dada função $f(x)$, onde $x \in \mathbb{R}^n$ e os pontos de teste são $x_1, ..., x_{n+1}$, o algoritmo passa pelos seguintes três passos [36]:

1. **Ordem inicial:**

   De acordo com os valores dos vértices $f(x_1) \leq f(x_2) \leq ... \leq f(x_{n+1})$.

2. **Calculo de x_0:**

   O centro geométrico de todos os pontos exceto x_{n+1}.

3. **Reflexão:**

   $x_r = x_0 + \alpha(x_0 - x_{n+1})(\alpha > 0)$ se o ponto refletido for melhor que o segundo pior, mas não melhor do que o melhor (ex: $f(x_1) \leq f(x_r) \leq ... \leq f(x_n)$, então o pior ponto x_{n+1} é substituído pelo seu valor refletido x_r e volta novamente ao primeiro passo.

4. **Expansão:**

   Se o ponto refletido é o melhor ponto até ao momento $f(x_r) < f(x_1)$, então é calculado a sua expansão de seguida $x_e = x_0 + \gamma(x_r - x_0)(\gamma > 0)$. Se o ponto expandido for melhor que o ponto refletido, então $f(x_e) < f(x_r)$ e de seguida é substituído o pior ponto x_{n+1} pelo ponto expandido x_e e volta ao primeiro passo. Caso contrário, é substituído o pior ponto x_{n+1} pelo ponto refletido x_r e volta ao primeiro passo.

5. **Contração:**

   Neste passo é certo que $f(x_r) > f(x_n)$. Então é contraído o ponto $x_c = x_0 + \rho(x_{n+1} - x)(0 < \rho \leq 0,5)$. Caso o ponto contraído seja melhor que o pior ponto (ex: $f(x_c) < f(x_{n+1})$), então o pior ponto x_{n+1} é substituído pelo ponto contraído x_c e volta novamente ao primeiro passo.

6. **Encolher:**

   O melhor ponto é substituído por $x_i = x_1 + \sigma(x_i - x_1)$ para todos $i \in 2, ..., n + 1$ e volta ao primeiro passo.

Sabendo que a função *fminsearch* recorre a um método de procura por um mínimo local de uma função escalar de diversas variáveis, a partir de um valor inicial previamente estipulado e que geralmente é referida como otimização não-linear, foi feito uso da seguinte sintaxe no MATLAB:

```
[x,fval] = fminsearch(fun,x0,options)
```

A função *fminsearch* começa o seu processamento no ponto inicial $x0$ e retorna o valor ótimo local da função fun através da variável x e o valor da função objetivo através da variável $fval$.

4.6 Controlos Efetuados

Ao longo das experiências efetuadas, foram realizados vários tipos de controlo, com a finalidade de encontrar a melhor resposta do sistema a controlar. Segue uma explicação, pela ordem de implementação, de cada tipo de controlo implementado.

4.6.1 PID com Método de Ziegler-Nichols

Inicialmente, sendo este o método de controlo mais conhecido, foram feitos testes com a implementação do método de Ziegler-Nichols no controlo PID. Foi utilizado o método de malha fechada, onde se seguiram os passos já mencionados anteriormente na subsecção 4.1.1. No capítulo seguinte, onde serão apresentados os resultados obtidos, este será identificado na legenda de cada gráfico e nas tabelas pela sigla "ZN".

A Figura 4.9 apresenta o fluxograma da obtenção dos resultados pelo método de Ziegler-Nichols.

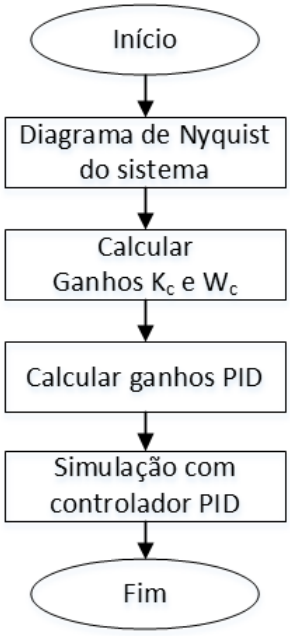


Figura 4.9: Fluxograma correspondente ao processamento do PID com Método de Ziegler-Nichols.

4.6.2 PID com o *pidtune* do MATLAB

Posteriormente, de modo a obter uma melhor resposta ao controlo PID, foi utilizada a função *pidtune* da *MathWorks*, já mencionado na sub subsecção 4.1.2. Sendo este um método de controlo PID mais otimizado que o anterior, foram descartados os ganhos obtidos com o método de Ziegler-Nichols e usados os ganhos obtidos com o *pidtune* nas experiências posteriores com o controlador PID-Difuso adaptativo. Nos gráficos e tabelas correspondentes aos resultados, este método será identificado pela sigla "*PID*".

Na Figura 4.10 é apresentado o fluxograma da obtenção dos resultados pelo método PID com o *pidtune* do MATLAB.

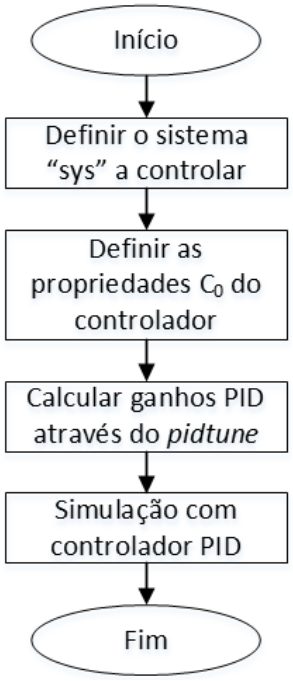


Figura 4.10: Fluxograma correspondente ao processamento do PID com o *pidtune* do MATLAB.

4.6.3 PID-Difuso Adaptativo

Seguidamente, usando os ganhos de K_p, K_i e K_d obtidos com o controlo PID através do método *pidtune*, foi implementado o controlo PID-Difuso adaptativo. Neste método, o controlador utiliza os ganhos do PID e a resposta do sistema é adaptada ao longo do tempo através do controlador difuso implementado em paralelo. Nos gráficos e tabelas correspondentes aos resultados, este método será identificado pela sigla "*Fuzzy PID*".

Na Figura 4.11 é possível observar o fluxograma da obtenção dos resultados pelo método PID-Difuso Adaptativo.

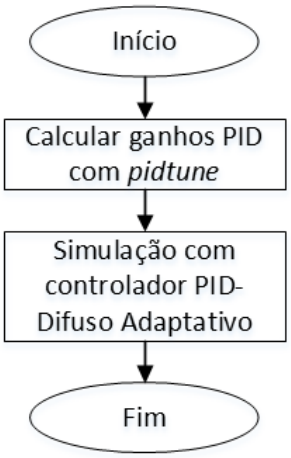


Figura 4.11: Fluxograma correspondente ao processamento do PID-Difuso Adaptativo.

4.6.4 PID-Difuso Adaptativo com Método Ótimo

Na fase seguinte, foram implementados os métodos ótimos ao controlador PID-Difuso adaptativo. Usando o método PID-Difuso adaptativo anteriormente mencionado, foram aplicados em paralelo os métodos ótimos mencionados na subsecção 4.4. Ou seja, nesta implementação foram efetuadas duas otimizações, com o intuito de obter uma melhor resposta do sistema a controlar. Tendo sido o sistema já otimizado uma primeira vez com a sintonia PID, este foi novamente otimizado com os índices de desempenho a atuar paralelamente com o controlador PID-Difuso adaptativo. Nos gráficos e tabelas correspondentes aos resultados, este método será identificado como "*Fuzzy PID X*", onde "X"corresponde ao método ótimo usado (ITAE, ITSE, IAE ou IAE).

Na Figura 4.12 está exposto o fluxograma da obtenção dos resultados pelo método PID-Difuso Adaptativo com Método Ótimo.

4.6.5 PID com Método Ótimo

Posteriormente, foram implementados os métodos ótimos ao controlador PID. Nesta abordagem, os ganhos de K_p, K_i e K_d iniciais são inicializados todos com o valor de 1. Aquando o funcionamento do controlador PID em paralelo com os índices de desempenho, os ganhos serão otimizados utilizando a minimização do erro, obtendo-se assim uma melhor resposta, fazendo apenas uma única otimização ao controlador. Nos gráficos e tabelas correspondentes aos resultados, este método será identificado como "*PID X*", onde "X"corresponde ao método ótimo usado (ITAE, ITSE, IAE ou IAE).

É possível observar na Figura 4.13 o fluxograma da obtenção dos resultados pelo método

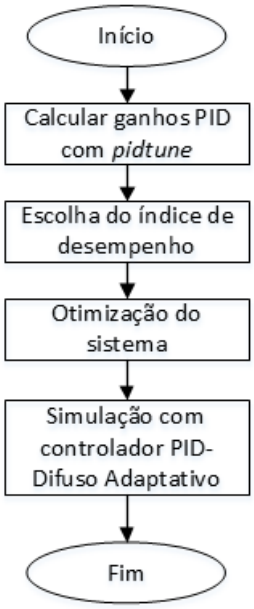


Figura 4.12: Fluxograma correspondente ao processamento do PID-Difuso Adaptativo com Método Ótimo.

PID com Método Ótimo.

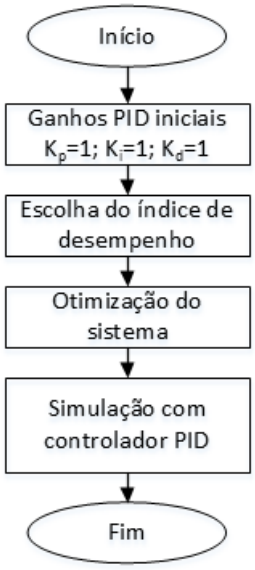


Figura 4.13: Fluxograma correspondente ao processamento do PID com Método Ótimo.

4.6.6 PID-Difuso Adaptativo com Ganhos do PID Ótimo

Seguidamente, assim como na terceira abordagem já mencionada, usando os ganhos de K_p, K_i e K_d obtidos com o controlo PID através dos métodos ótimos (abordagem anterior), foi implementado o controlo PID-Difuso adaptativo. Neste método, o controlador utiliza os ganhos

do PID otimizado com os índices de desempenho e a resposta do sistema é adaptada ao longo do tempo através do controlador difuso adaptativo implementado em paralelo. Nos gráficos e tabelas correspondentes aos resultados, este método será identificado como "*Fuzzy PID Ótimo*".

A Figura 4.14 apresenta o fluxograma da obtenção dos resultados pelo método PID-Difuso Adaptativo com Ganhos do PID Ótimo.

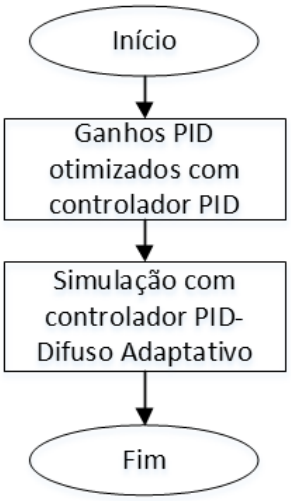


Figura 4.14: Fluxograma correspondente ao processamento do PID-Difuso Adaptativo com Ganhos do PID Ótimo.

Capítulo 5

Resultados

Neste capítulo serão apresentados todos os resultados obtidos nas diversas simulações realizadas através do Simulink do MATLAB e que permitiram validar os algoritmos de controlo propostos. As simulações levadas a cabo tiveram incidência sobre sistemas de várias ordens, com e sem atraso, lineares e não lineares. Os resultados são apresentados sob a forma gráfica para uma análise e interpretação lógica amigável, onde estará patente a diferença entre ambos os controladores.

Os resultados obtidos serão comentados tendo em consideração três das especificações fundamentais na análise de sistemas, a saber: o "Tempo de Subida, T_r", "*Overshoot*, M_p"e o "Tempo de Estabelecimento, T_s".

As simulações foram efetuadas com recurso aos controladores desenvolvidos e já mencionados anteriormente. O primeiro controlador PID-Difuso adaptativo a ser usado é o referido na subsecção 4.2.1, na Figura 4.2. O segundo controlador PID-Difuso adaptativo utilizado é o mencionado na subsecção 4.2.2, na Figura 4.3. Como forma de comparação com ambos os controladors PID-Difusos adaptativos, é utilizado o controlador PID mencionado na subsecção 4.1.2, na Figura 4.1.

5.1 Sistemas Lineares

Os primeiros sistemas estudados, foram sistemas lineares no tempo, de várias ordens, com e sem atraso. Para a escolha dos sistemas a controlar, foram tidos em conta as referências [30] e [37].

Para todos os sistemas estudados nesta secção, os resultados serão apresentados através de gráficos com as respostas dos sistemas e de tabelas com os valores dos ganhos K_p, K_i, K_d e os valores de M_p, T_s e T_r. Os gráficos apresentados encontram-se separados por índices de

desempenho e ordenados pela ordem *ITAE*, *ITSE*, *IAE* e *ISE*. Estes têm sempre por base de comparação o controlo PID sintonizado pelo método de Ziegler-Nichols, o PID sintonizado com o *pidtune* e o controlo PID-Difuso adaptativo não otimizado e sintonizado com os parâmetros do último controlo PID mencionado. As respostas a estes métodos base são as mesmas, independentemente do controlador PID-Difuso adaptativo que se esteja a abordar.

5.1.1 Primeiro Sistema

A função de transferência utilizada para a primeira simulação, foi a seguinte [37]:

$$P_1(s) = \frac{e^{-s}}{(sT+1)^2} \tag{5.1}$$

Onde se assumiu $T = 0,5\ s$.

Primeiro Controlador

Na Figura 5.1, é possível observar as simulações efetuadas para este primeiro sistema. A Tabela 5.1, apresenta os parâmetros temporais do sistema, para os gráficos obtidos anteriormente.

No primeiro sistema a ser simulado, observou-se nos métodos de controlo base, que se obteve uma resposta com um menor *overshoot* com o método de sintonia de Ziegler-Nichols, comparativamente ao controlo PID-Difuso adaptativo e consequentemente ao do método de sintonia *pidtune*. No entanto, a resposta obtida mostrou-se mais oscilatória nos primeiros 10 segundos da simulação, em comparação com os outros dois métodos mencionados. Contudo, como se pode observar através da Figura 5.1 e da Tabela 5.1, estas são respostas pouco aceitáveis para um bom método de controlo.

Com a aplicação dos índices de desempenho nas restantes simulações, obteve-se respostas bastante dispersas umas das outras. Analisando a Figura 5.1 e a Tabela 5.1, é possível concluir que a simulação mais eficaz e com melhor resposta, foi a que se utilizou a abordagem com o controlo PID-Difuso adaptativo otimizado pelo índice de desempenho *ITAE*. Com esta abordagem, obteve-se um *overshoot* de $4,0333\%$ e um sistema mais estável.

Já para com os restantes índices de desempenho *ITSE*, *IAE* e *ISE*, obteve-se respostas bastante semelhantes para ambas as simulações. Contudo, foram respostas menos conseguidas para que seja considerado um sistema com um bom controlo.

Na Figura 5.2 é possível observar a resposta do sistema com a introdução de uma perturbação de valor igual a $0,8$, aos 10 segundos de simulação. Ambos os métodos aplicados ao sistema apresentam respostas similares.

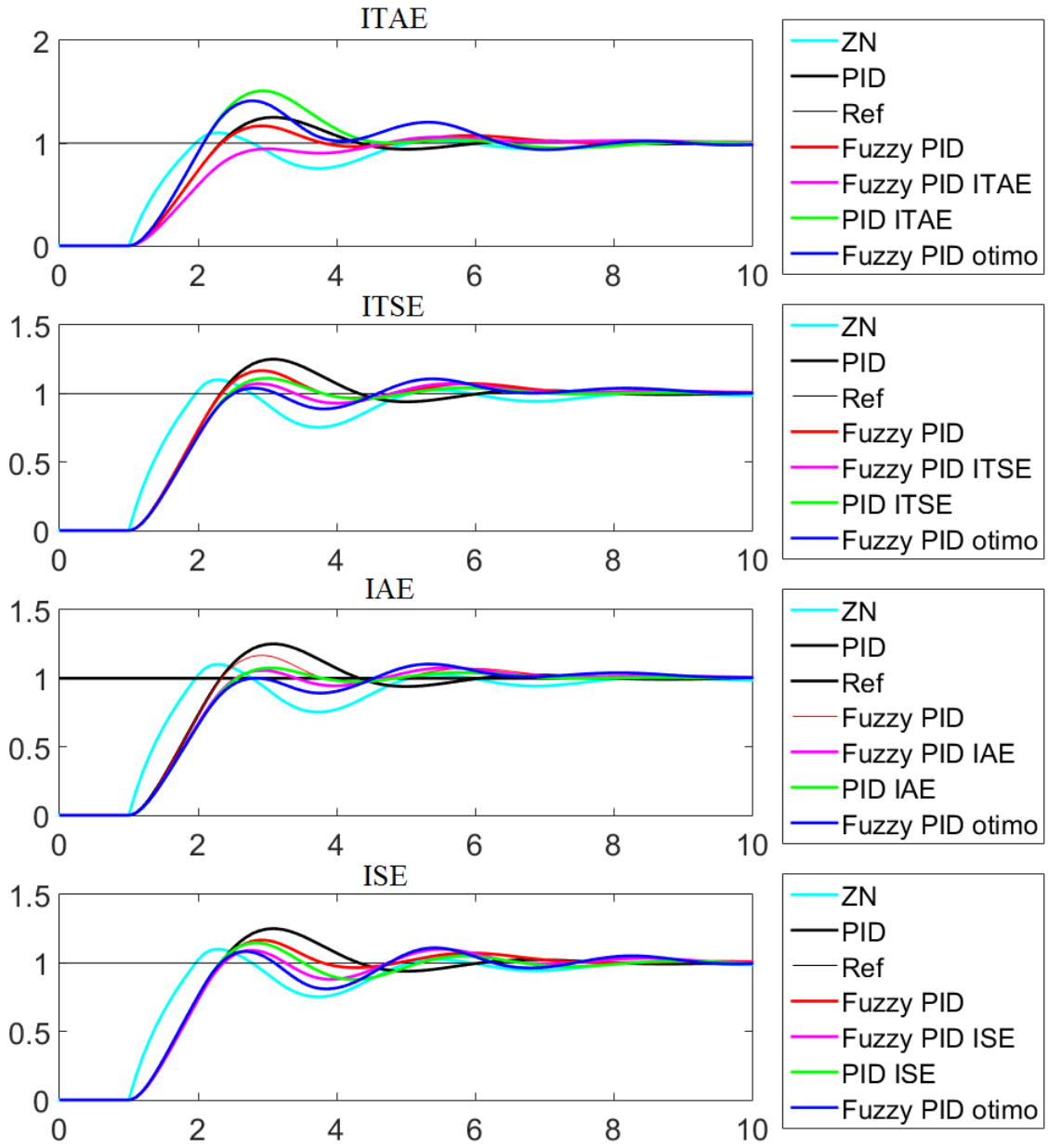


Figura 5.1: Análise temporal do primeiro sistema com uso do primeiro controlador.

Analisando com mais pormenor, o controlo através do método PID-Difuso adaptativo sem qualquer tipo de otimização mostrou-se mais eficaz na resposta à perturbação. Nesta simulação, o método PID-Difuso adaptativo otimizado por *ITAE* também apresentou uma boa resposta, reagindo até mais rápido na tentativa de devolver o sistema ao seu *setpoint*. Contudo apresentou alguma oscilação antes do sistema voltar a estabilizar.

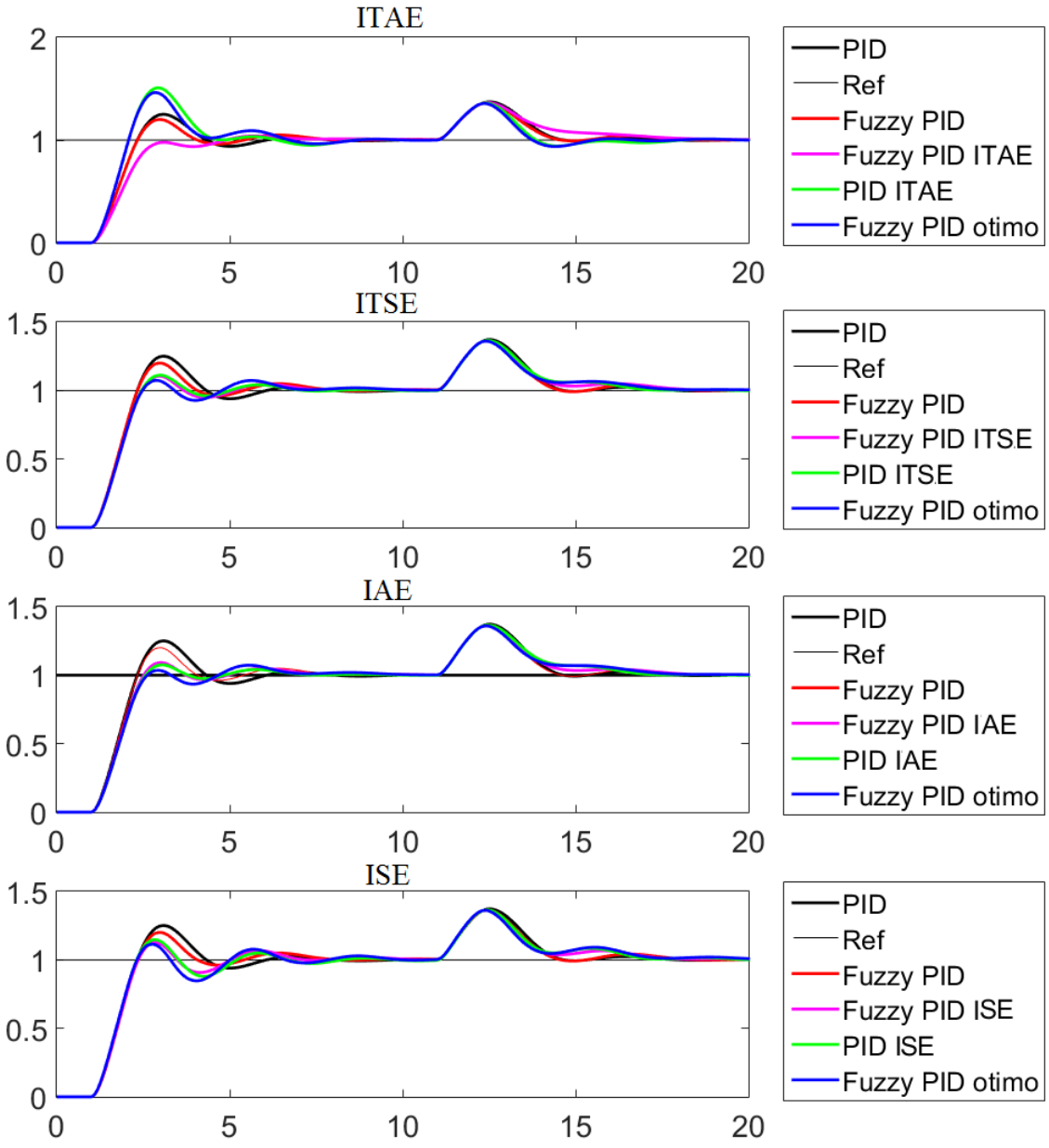


Figura 5.2: Análise temporal do primeiro sistema com perturbação, com uso do primeiro controlador.

Tabela 5.1: Valores dos ganhos K_p, K_i, K_d e valores de M_p, T_s e T_r do primeiro sistema, com o primeiro controlador.

	$\mathbf{K_p}$	$\mathbf{K_i}$	$\mathbf{K_d}$	$\mathbf{M_p}(\%)$	$\mathbf{T_r}(\mathbf{s})$	$\mathbf{T_s}(\mathbf{s})$
ZN	1,0441	1,8258	0,4382	10,143	0,7616	7,7336
PID	0,9231	0,6785	0,314	24,996	0,9290	7,1433
Fuzzy PID	0,9231	0,6785	0,314	15,673	0,9402	6,8860
Fuzzy PID ITAE	0,7521	0,5295	0,3414	5,1542	1,3361	6,3028
Fuzzy PID ITSE	0,8816	0,6063	0,3425	6,5522	1,0187	6,5873
Fuzzy PID IAE	0,8486	0,6062	0,3399	6,9111	1,0544	6,6436
Fuzzy PID ISE	0,9545	0,6199	0,4183	9,8370	0,9412	9,0831
PID ITAE	1,0505	1,0454	0,5743	50,793	0,7514	8,1524
PID ITSE	0,8879	0,6056	0,4381	11,204	1,0000	6,5590
PID IAE	0,8398	0,5882	0,4345	7,5533	1,0618	6,5600
PID ISE	1,0079	0,5897	0,4703	15,144	0,9087	7,6007
Fuzzy PID Ótimo ITAE	1,0505	1,0454	0,5743	43,355	0,7416	8,9957
Fuzzy PID Ótimo ITSE	0,8879	0,6056	0,4381	10,342	1,0137	8,8846
Fuzzy PID Ótimo IAE	0,8398	0,5882	0,4345	9,7915	1,0908	8,7707
Fuzzy PID Ótimo ISE	1,0079	0,5897	0,4703	11,586	0,9102	9,0600

Segundo Controlador

Na Figura 5.3, é possível observar as simulações efetuadas para este primeiro sistema. Na Tabela 5.2, apresenta-se os parâmetros temporais do sistema, para os gráficos obtidos anteriormente.

Relativamente aos métodos aplicados, os índices de desempenho *ITAE* e *IAE* foram os que apresentaram melhores resultados, sendo estes ligeiramente semelhantes. Analisando a Figura 5.3 e a Tabela 5.2 mais ao pormenor, pode-se concluir que a melhor resposta para este sistema foi obtida com recurso ao controlo com o método PID-Difuso adaptativo otimizado pelo *ITAE*. No entanto, o método PID-Difuso adaptativo sem qualquer tipo de otimização, também apresenta uma resposta muito semelhante e bastante aceitável.

Na Figura 5.4 é possível observar a resposta do sistema com a introdução de uma perturbação de valor igual a $0,8$, aos 10 segundos de simulação. Ambos os métodos aplicados ao sistema apresentam respostas similares. No entanto, o método PID-Difuso adaptativo, com os ganhos ganhos K_p, K_i e K_d já otimizados pelo *ITAE* apresentou uma melhor resposta, comparativamente com os restantes métodos simulados.

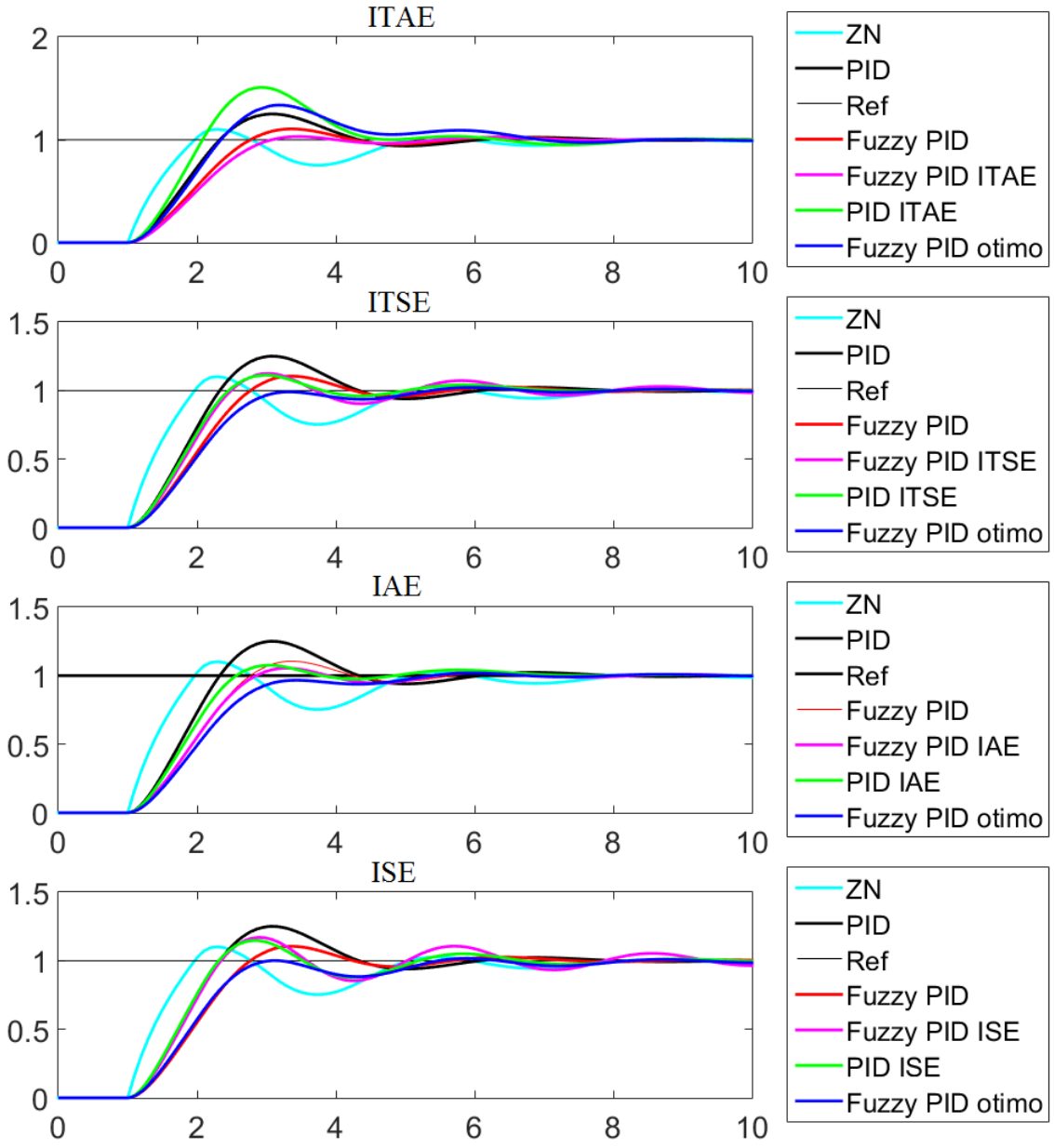


Figura 5.3: Análise temporal do primeiro sistema com uso do segundo controlador.

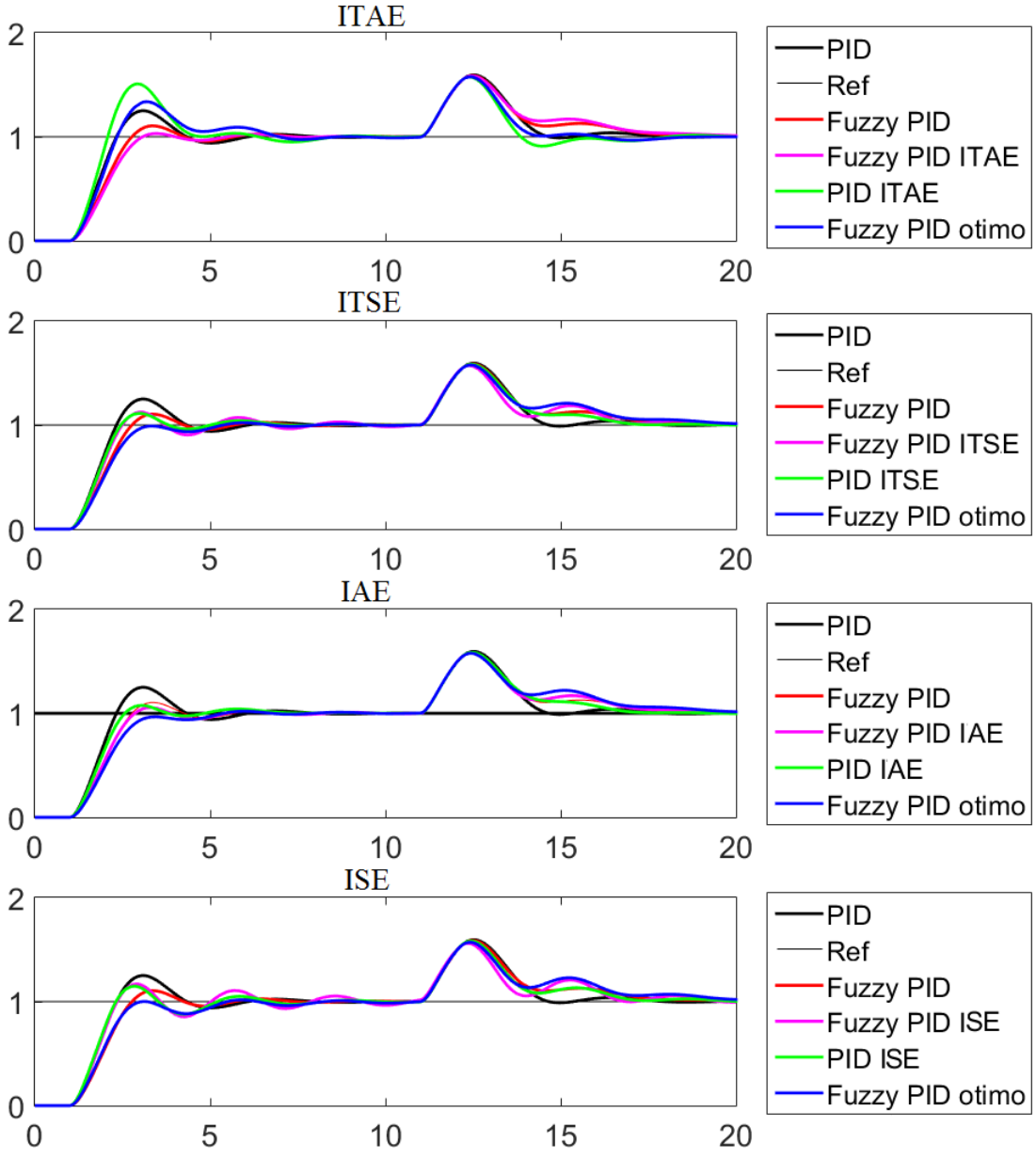


Figura 5.4: Análise temporal do primeiro sistema com perturbação, com uso do segundo controlador.

Tabela 5.2: Valores dos ganhos K_p, K_i, K_d e valores de M_p, T_s e T_r do primeiro sistema, com o segundo controlador.

	$\mathbf{K_p}$	$\mathbf{K_i}$	$\mathbf{K_d}$	$\mathbf{M_p}(\%)$	$\mathbf{T_r}(\mathbf{s})$	$\mathbf{T_s}(\mathbf{s})$
ZN	1,0441	1,8258	0,4382	10,143	0,7616	7,7336
PID	0,9231	0,6785	0,314	24,996	0,9290	7,1433
Fuzzy PID	0,9231	0,6784	0,3140	10,108	1,2200	6,6590
Fuzzy PID ITAE	0,8487	0,6276	0,3520	2,9953	1,3707	6,4429
Fuzzy PID ITSE	1,1315	0,7238	0,5402	14,239	1,0063	9,4245
Fuzzy PID IAE	0,9372	0,6486	0,3909	5,5511	1,2433	6,5359
Fuzzy PID ISE	1,2587	0,7511	0,6128	21,242	0,8994	9,5317
PID ITAE	1,0505	1,0454	0,5743	50,793	0,7514	8,1524
PID ITSE	0,8879	0,6056	0,4381	11,204	1,0000	6,5590
PID IAE	0,8398	0,5882	0,4345	7,5533	1,0618	6,5600
PID ISE	1,0079	0,5897	0,4703	15,144	0,9087	7,6007
Fuzzy PID Ótimo ITAE	1,0505	1,0454	0,5743	34,935	0,9221	6,8461
Fuzzy PID Ótimo ITSE	0,8879	0,6056	0,4381	2,6554	1,3760	6,3105
Fuzzy PID Ótimo IAE	0,8398	0,5882	0,4345	2,1972	1,4915	6,1673
Fuzzy PID Ótimo ISE	1,0079	0,5897	0,4703	3,1945	1,2223	9,0471

5.1.2 Segundo Sistema

A função de transferência utilizada para a segunda simulação, foi a seguinte [37]:

$$P_2(s) = \frac{1}{(s+1)(sT+1)^2} \tag{5.2}$$

Onde se assumiu $T = 0,5\ s$.

Primeiro Controlador

Na Figura 5.5, é possível observar as simulações efetuadas para este segundo sistema. Na Tabela 5.3, é possível observar os parâmetros temporais do sistema, para os gráficos obtidos anteriormente.

No segundo sistema a controlar, observou-se que as respostas nos métodos de controlo base seguiram o rumo já esperado. Ou seja, o controlo PID-Difuso adaptativo não otimizado apresentou novamente uma melhor resposta comparativamente com o controlo PID com o *pidtune*, assim como com o PID pelo método de Ziegler-Nichols. Chega-se facilmente a esta conclusão, observando a discrepância existente entre as respostas na Figura 5.5 e através dos parâmetros temporais do sistema expostos na Tabela 5.3.

Relativamente às simulações com recurso aos índices de desempenho, tanto para o *ITAE*,

como para o *IAE*, o sistema apresentou respostas ligeiramente similares. No entanto, o *ITAE* mostrou-se mais uma vez o índice de desempenho mais eficaz, fornecendo um controlo mais eficiente. O mesmo se pode comprovar, analisando a Figura 5.5, assim como a Tabela 5.3. Já os restantes índices de desempenho originaram respostas mais oscilatórias.

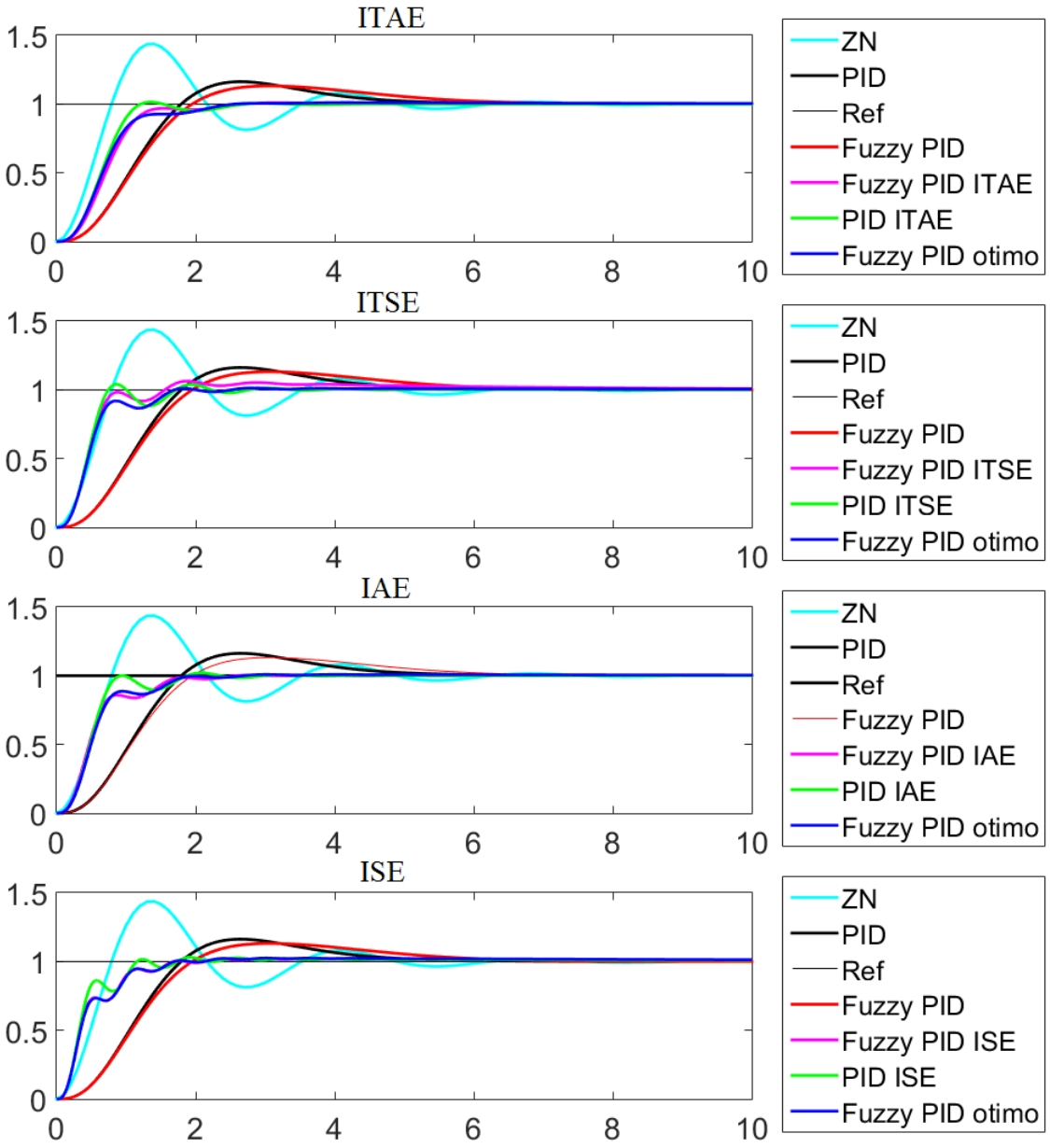


Figura 5.5: Análise temporal do segundo sistema com uso do primeiro controlador.

Na Figura 5.6 é possível observar a resposta do sistema com a introdução de uma perturbação de valor igual a $0,8$, aos 10 segundos de simulação. Ambos os métodos aplicados ao sistema apresentam respostas similares. No entanto, os métodos ITSE e ISE demonstram melhor eficácia na resposta a perturbações, mesmo comparativamente ao PID-Difuso adaptativo não otimizado.

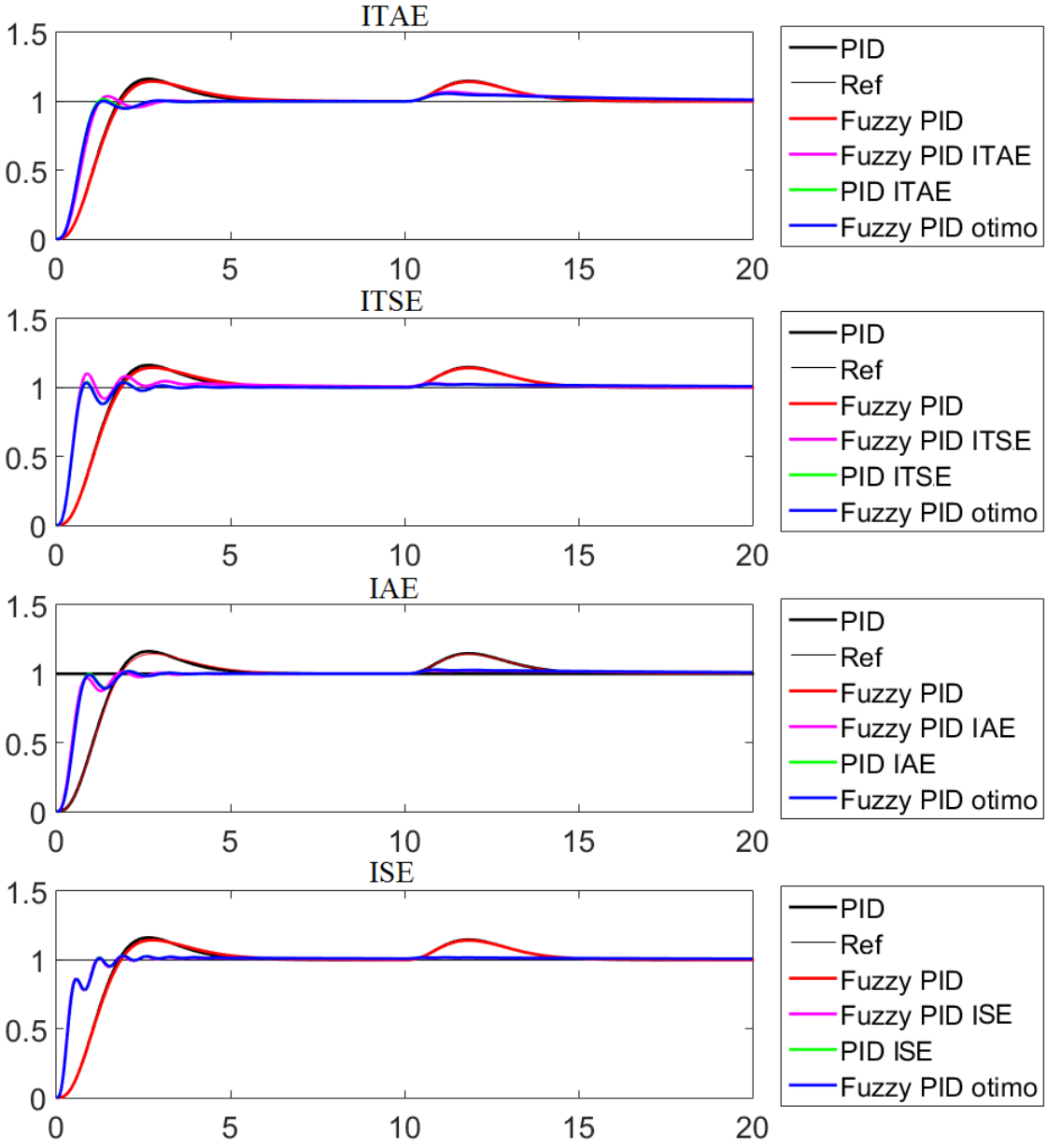


Figura 5.6: Análise temporal do segundo sistema com perturbação.

Tabela 5.3: Valores dos ganhos K_p, K_i, K_d e valores de M_p, T_s e T_r do segundo sistema, com o primeiro controlador.

	$\mathbf{K_p}$	$\mathbf{K_i}$	$\mathbf{K_d}$	$\mathbf{M_p}(\%)$	$\mathbf{T_r}(\mathbf{s})$	$\mathbf{T_s}(\mathbf{s})$
ZN	5,4001	1,1107	0,2666	43,762	0,5306	5,8982
PID	2,1899	1,3751	0,8719	16,269	1,1056	5,0516
Fuzzy PID	2,1899	1,3751	0,8719	13,160	1,2026	6,0737
Fuzzy PID ITAE	6,4500	1,4055	2,6538	0,6722	0,8359	2,4484
Fuzzy PID ITSE	19,758	4,4536	7,2949	5,4177	0,4799	5,1832
Fuzzy PID IAE	19,463	1,8552	8,2716	0,2666	1,1863	2,3369
Fuzzy PID ISE	50,969	4,3131	20,803	0,9842	0,8600	2,0238
PID ITAE	7,7763	1,4646	3,4782	1,5199	0,6891	2,4419
PID ITSE	20,269	2,2616	7,7960	3,9443	0,4312	2,5693
PID IAE	16,444	1,9074	6,9393	1,9584	0,4965	1,7926
PID ISE	48,853	4,5755	20,311	2,2093	0,8614	1,9617
Fuzzy PID Ótimo ITAE	7,7763	1,4646	3,4782	0,6830	0,8823	2,3261
Fuzzy PID Ótimo ITSE	20,269	2,2616	7,7960	0,8891	0,5475	1,6207
Fuzzy PID Ótimo IAE	16,444	1,9074	6,9393	0,4602	1,2336	1,7916
Fuzzy PID Ótimo ISE	48,853	4,5755	20,311	1,1853	0,8738	1,6545

Segundo Controlador

A Figura 5.7, apresenta as simulações efetuadas para este segundo sistema. Na Tabela 5.4, é possível observar os parâmetros temporais do sistema, para os gráficos obtidos anteriormente.

Com a utilização dos índices de desempenho neste sistema, o *ITAE* foi o que apresentou uma melhor resposta, tanto com o método PID-Difuso adaptativo otimizado, como com o método PID-Difuso adaptativo com os ganhos de K_p, K_i, K_d já otimizados. Por outro lado, os restantes índices de desempenho apresentaram respostas bastante oscilatórias no tempo, como se pode observar na Figura 5.7. No entanto, sendo o *ITAE* o índice de desempenho o que apresentou melhores resultados, as respostas variaram ligeiramente com o método aplicado. Utilizando o método PID-Difuso adaptativo com recurso aos ganhos da sintonia *pidtune* e com posterior otimização, o sistema apresenta uma ligeira oscilação no período inicial, embora estabilize rapidamente. Já com o método PID-Difuso adaptativo com recurso aos ganhos do *pidtune* já otimizados, apresentou uma resposta melhor e mais estável.

Na Figura 5.8 é possível observar a resposta do sistema com a introdução de uma perturbação de valor igual a $0,8$ aos 10 segundos de simulação. Ambos os métodos aplicados apresentam respostas similares. No entanto, pode-se analisar esta simulação por duas vertentes. Analisando só a parte da resposta à perturbação, os métodos aplicados com recurso aos índices de desempenho

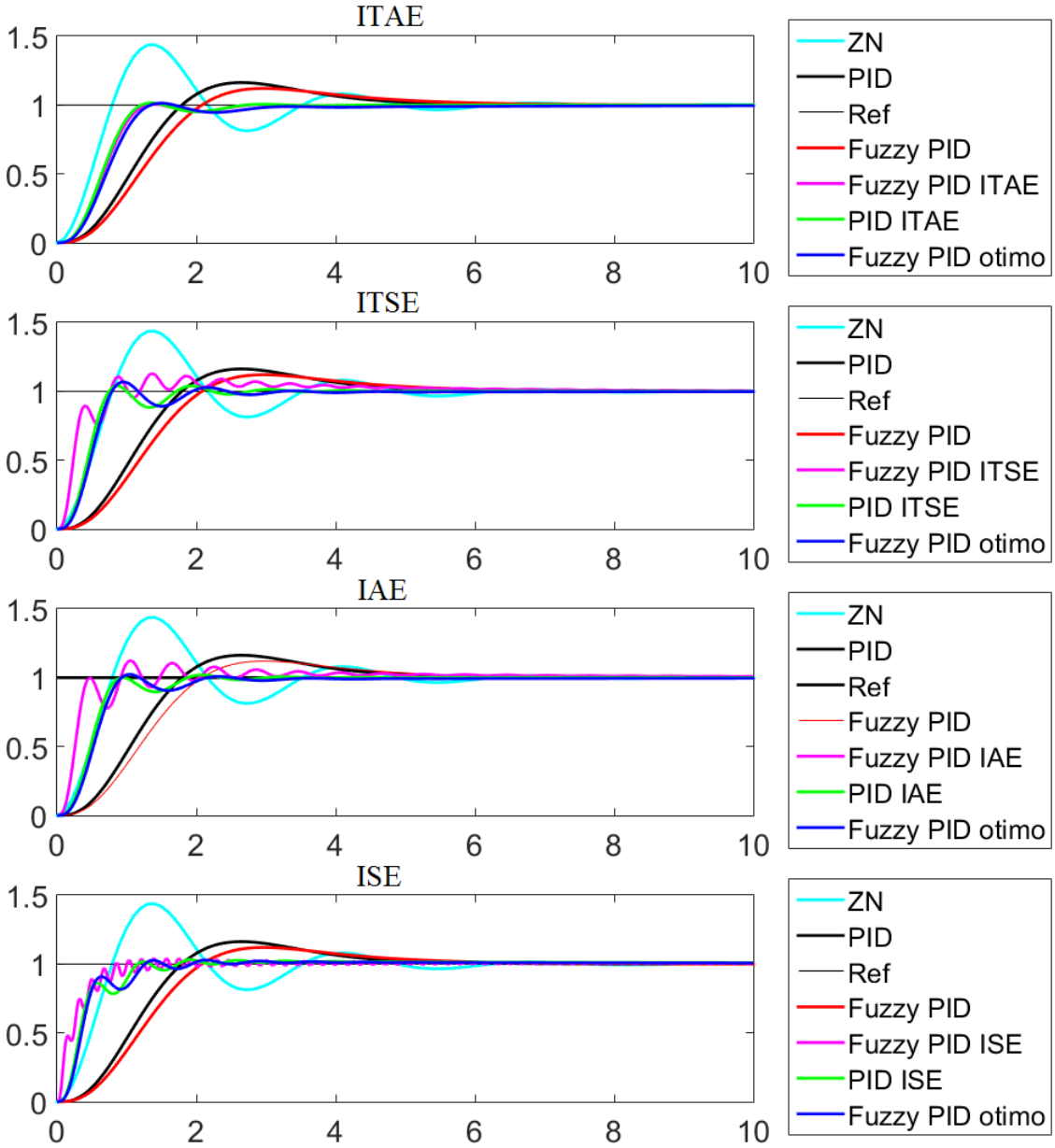


Figura 5.7: Análise temporal do segundo sistema com uso do segundo controlador.

ITSE e *IAE* apresentaram uma melhor resposta. Por outro lado, se a análise for feita desde o início da simulação, incluindo a resposta à perturbação, o índice de desempenho *ITAE* foi o que apresentou melhores resultados, pois os restantes índices apresentam respostas bastante oscilatórias no período inicial de simulação, levando assim a um pior controlo do sistema.

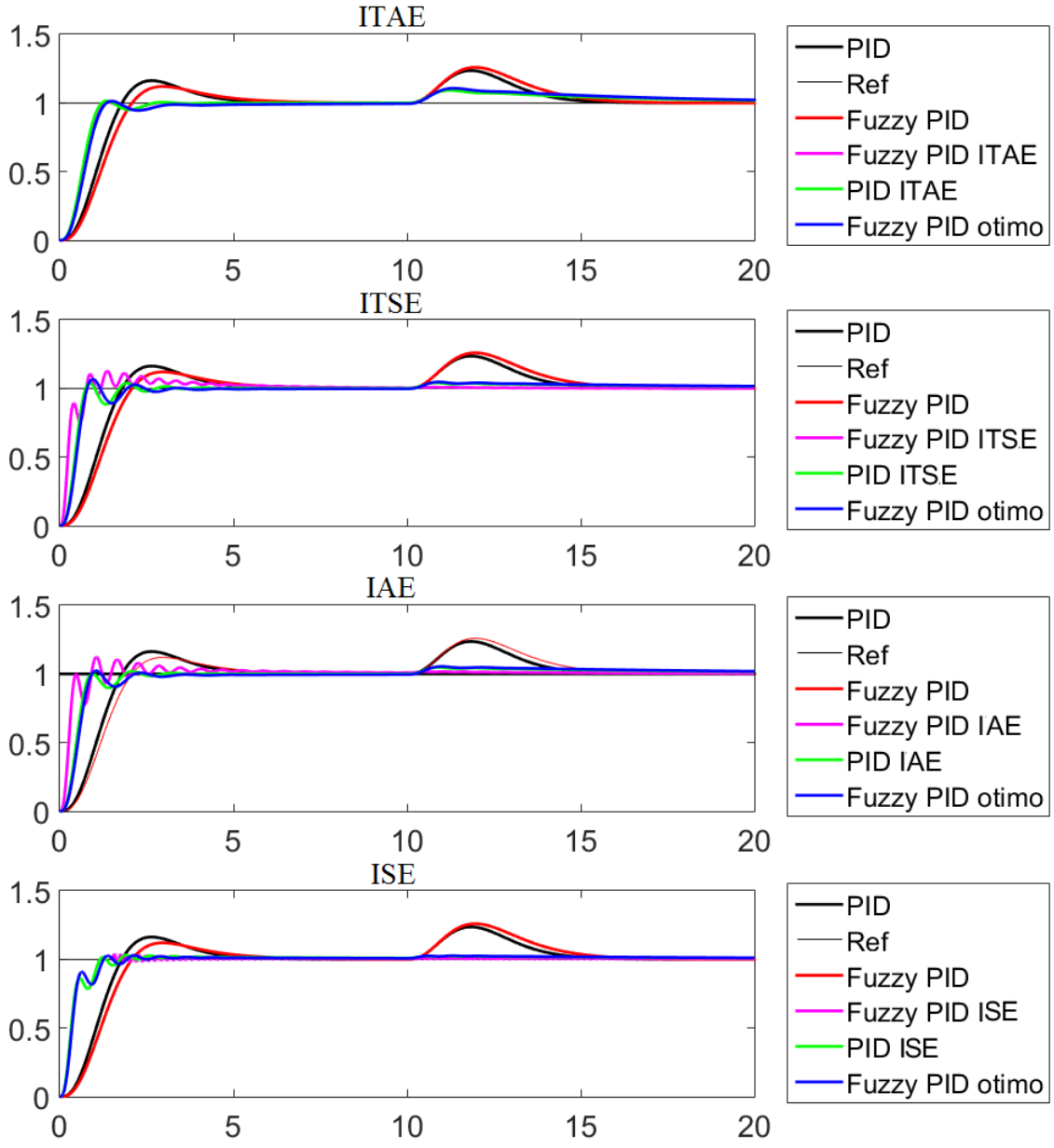


Figura 5.8: Análise temporal do segundo sistema com perturbação, com uso do segundo controlador.

Tabela 5.4: Valores dos ganhos K_p, K_i, K_d e valores de M_p, T_s e T_r do segundo sistema, com o segundo controlador.

	$\mathbf{K_p}$	$\mathbf{K_i}$	$\mathbf{K_d}$	$\mathbf{M_p}(\%)$	$\mathbf{T_r}(\mathbf{s})$	$\mathbf{T_s}(\mathbf{s})$
ZN	5,4001	1,1107	0,2666	43,762	0,5306	5,8982
PID	2,1899	1,3751	0,8719	16,269	1,1056	5,0516
Fuzzy PID	2,1899	1,3751	0,8718	12,009	1,2827	5,6721
Fuzzy PID ITAE	9,2643	1,7780	4,3406	0,5848	0,7165	2,4620
Fuzzy PID ITSE	150,66	47,220	49,504	12,502	0,5974	5,4003
Fuzzy PID IAE	106,44	18,513	33,188	11,391	0,2527	4,7015
Fuzzy PID ISE	1309,4	37,578	380,49	3,0329	0,5774	2,9275
PID ITAE	7,7763	1,4646	3,4782	1,5199	0,6891	2,4419
PID ITSE	20,269	2,2616	7,7960	3,9443	0,4312	2,5693
PID IAE	16,444	1,9074	6,9393	1,9584	0,4965	1,7926
PID ISE	48,853	4,5755	20,311	2,2093	0,8614	1,9617
Fuzzy PID Ótimo ITAE	7,7763	1,4646	3,4782	1,8348	0,7602	2,8400
Fuzzy PID Ótimo ITSE	20,269	2,2616	7,7960	7,1030	0,4605	2,8668
Fuzzy PID Ótimo IAE	16,444	1,9074	6,9393	2,7769	0,5290	2,0078
Fuzzy PID Ótimo ISE	48,853	4,5755	20,311	2,1168	0,4379	2,1807

5.1.3 Terceiro Sistema

A função de transferência utilizada para a terceira simulação, foi a seguinte [37]:

$$P_3(s) = \frac{1}{(s+1)((sT)^2 + 1.4sT + 1)} \tag{5.3}$$

Onde se assumiu $T = 0,5\ s$.

Primeiro Controlador

Na Figura 5.9, é possível observar as simulações efetuadas para este terceiro sistema. Na Tabela 5.5, é possível observar os parâmetros temporais do sistema, para os gráficos obtidos anteriormente.

Neste último sistema, obteve-se as respostas esperadas para os métodos de controlo base. Ou seja, conseguiu-se novamente uma melhor resposta do sistema para o controlo PID-Difuso adaptativo sem otimizações, comparativamente com o controlo PID (métodos *pidtune* e Ziegler-Nichols).

Já com a implementação dos métodos ótimos, obteve-se respostas bastante idênticas com os índices de desempenho *ITSE* e *IAE*. No entanto, o índice de desempenho *ITAE* foi o que apresentou novamente melhores resultados para ambos os métodos aplicados, apresentando

também valores de *overshoot* mais baixos (Tabela 5.5). Com os restantes indices de desempenho, embora os valores de *overshoot* também sejam reduzidos, estes produziram respostas bastante oscilatórias e pouco convincentes.

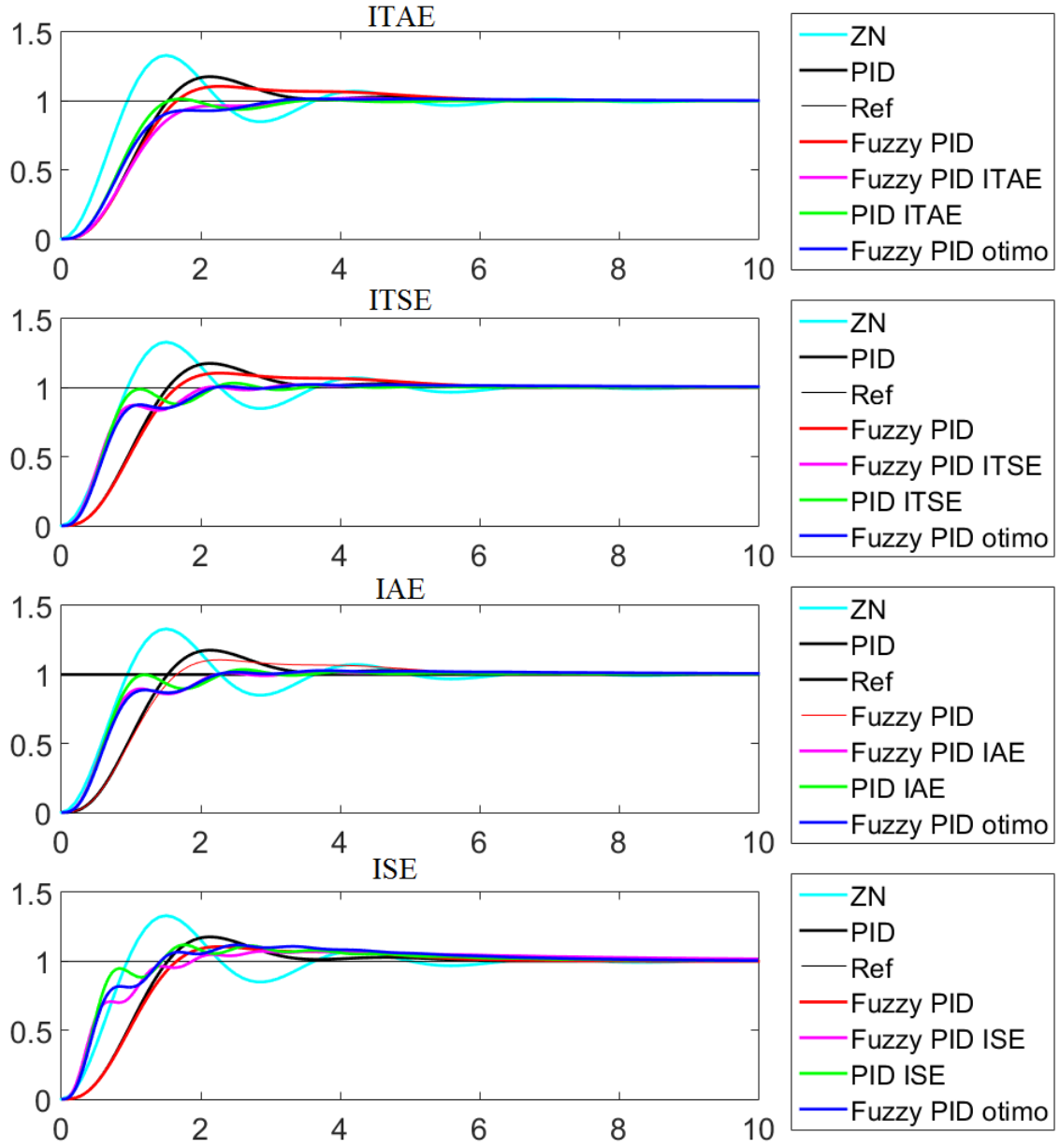


Figura 5.9: Análise temporal do terceiro sistema com uso do primeiro controlador.

Na Figura 5.10 é possível observar a resposta do sistema com a introdução de uma perturbação de valor igual a $0,8$, aos 10 segundos de simulação. A análise a esta simulação é bastante idêntica à já feita no sistema anterior. Embora os métodos aplicados com recurso aos índices de desempenho *ITSE* e *ISE* tenham fornecido melhores desempenhos na resposta à perturbação, os seus instantes iniciais de simulação não permitem que estes sejam bons métodos de controlo. Por outro lado, o *ITAE* apresentou-se como o melhor índice de desempenho,

apresentando uma resposta mais constante ao longo de toda a simulação.

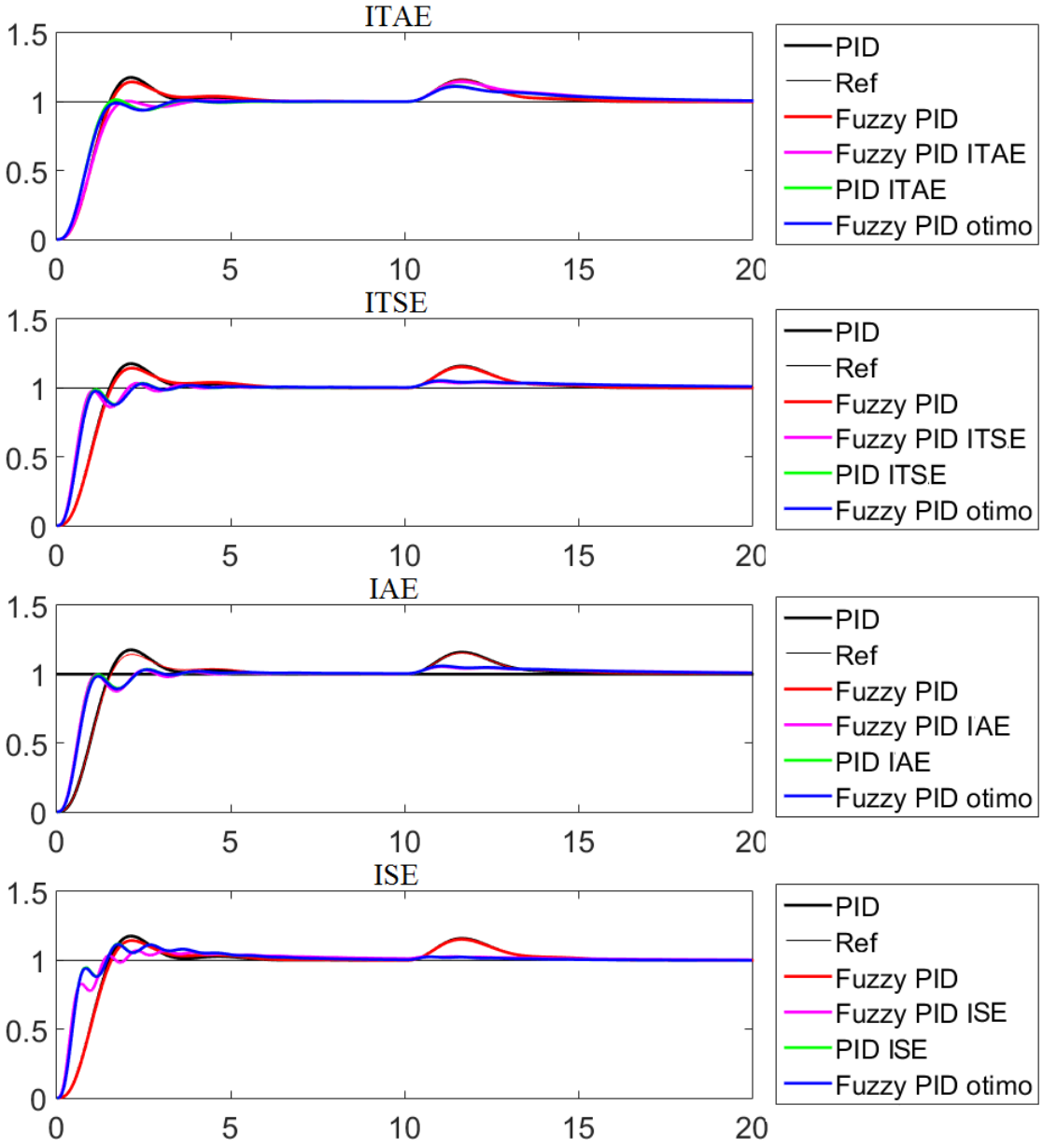


Figura 5.10: Análise temporal do terceiro sistema com perturbação, com uso do primeiro controlador.

Tabela 5.5: Valores dos ganhos K_p, K_i, K_d e valores de M_p, T_s e T_r do terceiro sistema, com o primeiro controlador.

	$\mathbf{K_p}$	$\mathbf{K_i}$	$\mathbf{K_d}$	$\mathbf{M_p}(\%)$	$\mathbf{T_r}(\mathbf{s})$	$\mathbf{T_s}(\mathbf{s})$
ZN	3,276	1,2047	0,2891	33,073	0,6319	5,9974
PID	2,2144	1,4651	0,8367	17,627	0,9147	5,2699
Fuzzy PID	2,2144	1,4651	0,8367	10,596	0,9903	5,6222
Fuzzy PID ITAE	2,4338	1,0213	1,0200	1,6824	1,1709	3,0758
Fuzzy PID ITSE	10,390	1,7407	5,0856	1,4203	1,4407	2,7531
Fuzzy PID IAE	8,4832	1,5988	4,1065	1,3804	1,5303	2,1334
Fuzzy PID ISE	25,024	5,3281	13,645	5,4289	1,0364	6,8007
PID ITAE	3,5089	1,1384	1,6623	1,4679	0,8738	3,1377
PID ITSE	8,7778	1,6742	4,4019	3,0892	0,6054	2,6599
PID IAE	7,7595	1,6784	3,9159	3,2222	0,6329	2,8373
PID ISE	18,189	6,6205	9,6109	11,494	0,4845	6,1149
Fuzzy PID Ótimo ITAE	3,5089	1,1384	1,6623	1,1860	1,0791	2,8701
Fuzzy PID Ótimo ITSE	8,7778	1,6742	4,4019	1,6176	1,5025	2,0984
Fuzzy PID Ótimo IAE	7,7595	1,6784	3,9159	2,0784	1,5283	3,8870
Fuzzy PID Ótimo ISE	18,189	6,6205	9,6109	11,292	1,0375	6,8506

Segundo Controlador

Na Figura 5.11 e na Tabela 5.6 é possível observar as simulações efetuadas para este terceiro sistema e os seus parâmetros temporais, respetivamente.

Relativamente à simulação com recurso ao método PID-Difuso adaptativo não otimizado, este apresentou melhores resultados comparativamente aos métodos PID (*pidtune* e Ziegler-Nichols), como era de esperar.

Quanto aos índices de desempenho, o *ITAE* mostrou-se novamente mais eficaz no controlo do sistema. Tanto com o método do PID-Difuso adaptativo otimizado como com o método PID-Difuso adaptativo com os ganhos do *pidtune* já otimizados, apresentaram respostas bastante satisfatórias e com um *overshoot* bastante baixo. Já os restantes índices apresentaram novamente respostas oscilatórias e pouco satisfatórias.

Na Figura 5.12 é possível observar a resposta do sistema com a introdução de uma perturbação de valor igual a $0,8$ aos 10 segundos de simulação. Ambos os métodos aplicados ao sistema apresentam respostas similares. No entanto, analisando a resposta à perturbação, os índices de desempenho *ITSE* e *ISE* apresentaram uma maior eficácia, comparativamente aos outros índices e aos métodos base. No entanto, analisando todo o tempo de simulação, desde os instantes iniciais, o índice de desempenho *ITAE* assume-se como melhor resposta, pois embora a sua resposta à perturbação também seja satisfatória, os instantes iniciais de simulação

apresentam melhores resultados comparativamente com os restantes índices.

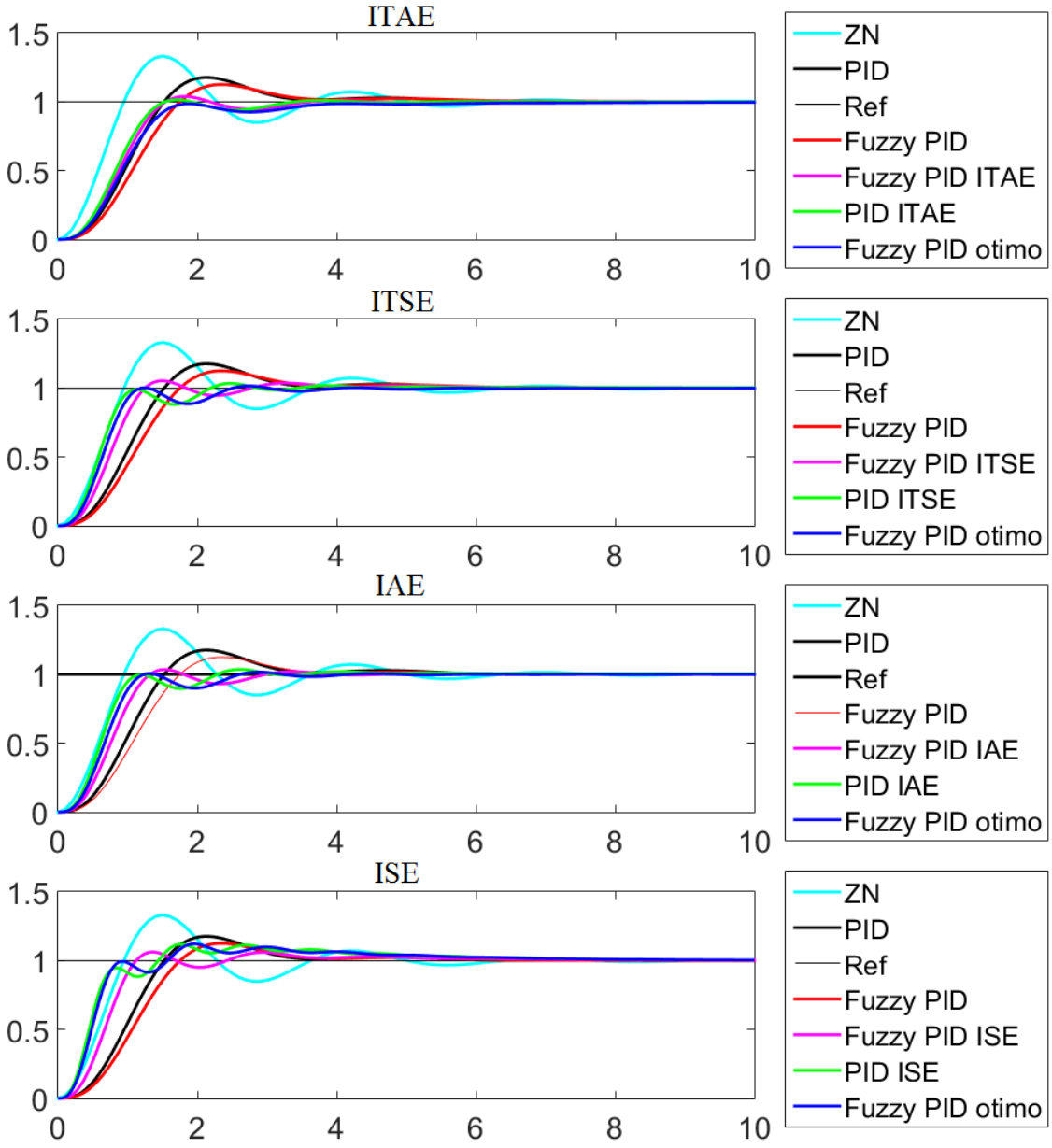


Figura 5.11: Análise temporal do terceiro sistema com uso do segundo controlador.

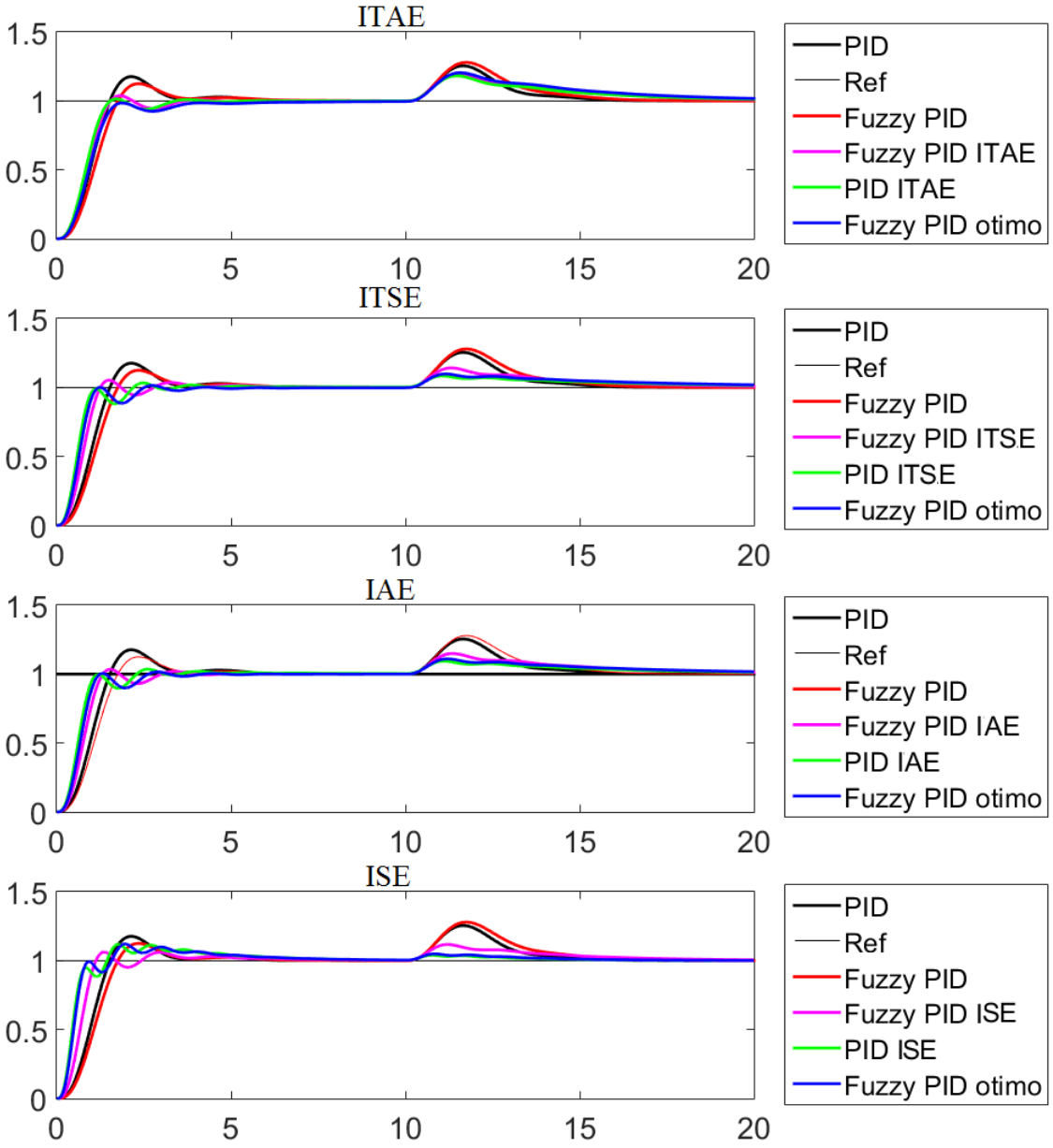


Figura 5.12: Análise temporal do terceiro sistema com perturbação, com uso do segundo controlador.

Tabela 5.6: Valores dos ganhos K_p, K_i, K_d e valores de M_p, T_s e T_r do terceiro sistema, com o segundo controlador.

	$\mathbf{K_p}$	$\mathbf{K_i}$	$\mathbf{K_d}$	$\mathbf{M_p}(\%)$	$\mathbf{T_r}(\mathbf{s})$	$\mathbf{T_s}(\mathbf{s})$
ZN	3,276	1,2047	0,2891	33,073	0,6319	5,9974
PID	2,2144	1,4651	0,8367	17,627	0,9147	5,2699
Fuzzy PID	2,2143	1,4651	0,8366	12,491	1,0517	5,3085
Fuzzy PID ITAE	3,8026	1,3787	1,7137	3,8139	0,9031	3,3370
Fuzzy PID ITSE	5,9237	1,8896	2,8858	5,0328	0,7347	3,6529
Fuzzy PID IAE	5,5438	1,6274	2,6830	3,4088	0,7686	2,8275
Fuzzy PID ISE	7,4496	2,4284	3,7113	5,8687	0,6607	4,9711
PID ITAE	3,5089	1,1384	1,6623	1,4679	0,8738	3,1377
PID ITSE	8,7778	1,6742	4,4019	3,0892	0,6054	2,6599
PID IAE	7,7595	1,6784	3,9159	3,2222	0,6329	2,8373
PID ISE	18,189	6,6205	9,6109	11,494	0,4845	6,1149
Fuzzy PID Ótimo ITAE	3,5089	1,1384	1,6623	0	1,0036	3,6721
Fuzzy PID Ótimo ITSE	8,7778	1,6742	4,4019	1,7042	0,6516	3,5700
Fuzzy PID Ótimo IAE	7,7595	1,6784	3,9159	1,6820	0,6860	2,4928
Fuzzy PID Ótimo ISE	18,189	6,6205	9,6109	11,708	0,4917	6,2792

5.2 Sistemas Não Lineares

Numa fase posterior, foram estudados sistemas não lineares no tempo. Para a escolha dos sistemas a controlar, foram tidos em conta as referências [38] e [27].

Para todos os sistemas estudados nesta secção, os resultados serão apresentados através de gráficos com as respostas dos sistemas e de tabelas com os valores dos ganhos K_p, K_i, K_d e os valores de M_p, T_s e T_r. Os gráficos apresentados encontram-se separados por índices de desempenho e ordenados pela ordem *ITAE*, *ITSE*, *IAE* e *ISE*. Estes têm sempre por base de comparação o controlo PID em malha fechada de Ziegler-Nichols e o controlo PID-Difuso adaptativo não otimizado e sintonizado com os parâmetros do controlo PID. As respostas a estes métodos base são as mesmas, independentemente do controlador PID-Difuso adaptativo que se esteja a abordar.

Para o controlo PID através do método em malha fechada de Ziegler-Nichols, foi necessário colocar o sistema em oscilação, aumentando o valor do ganho K_p. Após isto e depois de calcular o período crítico, aplicou-se os dados da Tabela 2.2 mencionada no capítulo 2 referente aos controladores PID.

5.2.1 Primeiro Sistema

O primeiro sistema não linear utilizado para a primeira simulação, foi o seguinte [38]:

$$\frac{d^2y(t)}{dt^2} + \frac{dy(t)}{dt} + 0,25y^2 = u(t-L) \tag{5.4}$$

Com $L = 0,5$.

A representação no espaço de estados do sistema (5.4) é dada por:

$$\begin{cases} \dot{x}_1(t) = x_2(t) \\ \dot{x}_2(t) = -x_2(t) - 0,25x_1(t) + u(t-L) \end{cases} \tag{5.5}$$

O diagrama de blocos do sistema é apresentado na Figura 5.13.

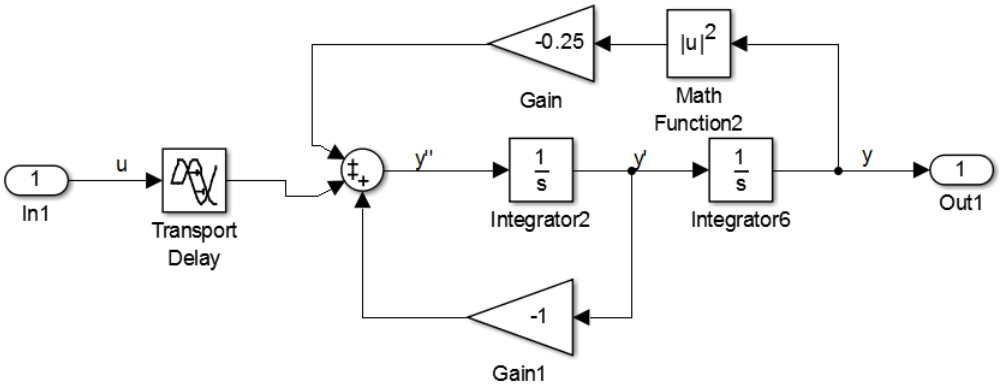


Figura 5.13: Diagrama de blocos do primeiro sistema não linear.

Primeiro Controlador

Na Figura 5.14, é possível observar as simulações efetuadas para este primeiro sistema. Na Tabela 5.7, é possível observar os parâmetros temporais do sistema, para os gráficos obtidos anteriormente.

Neste sistema, observou-se nos métodos de controlo base, que se obteve uma melhor resposta com o uso do controlador PID-Difuso adaptativo não otimizado, comparativamente ao controlo PID com o método de sintonia Ziegler-Nichols. O mesmo se pode comprovar, analisando a Tabela 5.7, onde se pode observar que o controlador PID-Difuso adaptativo não otimizado obteve um *overshoot* relativamente inferior ao controlo PID. No entanto, este ainda é um valor relativamente alto para um controlo eficaz do sistema.

Relativamente aos índices de desempenho aplicados, neste sistema o *IAE* foi o que apresentou melhor resposta com o método PID-Difuso adaptativo utilizando os ganhos do PID já otimizados. Observando a Tabela 5.7 e a Figura 5.14 é possível ver que é uma resposta minimamente estável e com um *overshoot* reduzido. Já para os restantes índices de desempe-

nho, embora algumas respostas tenham *overshoot* relativamente baixo, estas apresentaram-se bastante oscilatórias no período inicial do controlo.

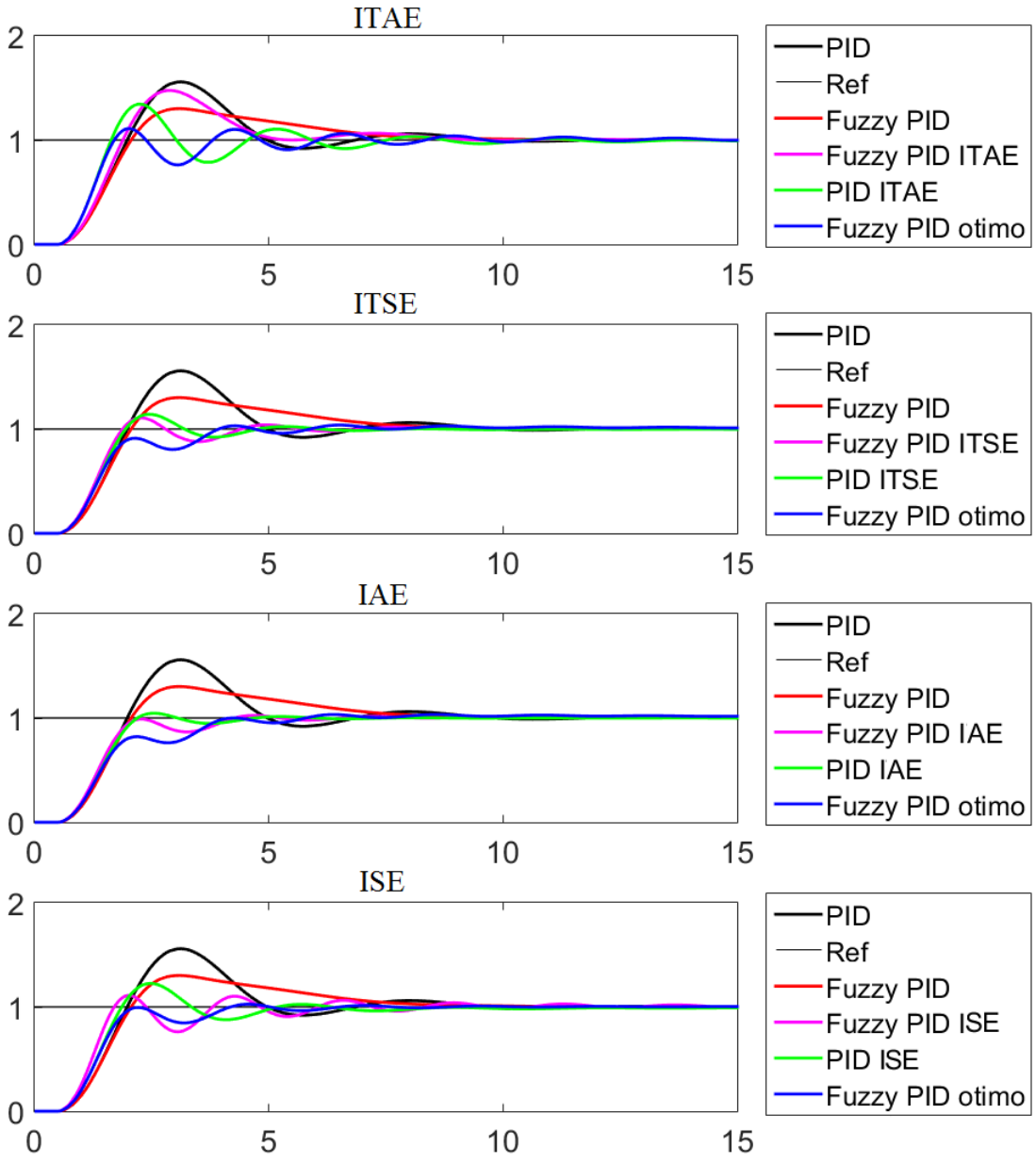


Figura 5.14: Análise temporal do primeiro sistema não linear com uso do primeiro controlador.

Na Figura 5.15 é possível observar a resposta do sistema com a introdução de uma perturbação de valor igual a $0,8$ aos 20 segundos de simulação. Para este sistema, a melhor resposta obteve-se com a utilização do PID-Difuso adaptativo não otimizado. A utilização dos índices de desempenho na resposta a perturbações não se mostrou eficaz, isto porque todos eles apresentaram respostas pouco satisfatórias por demorarem imenso tempo a devolver o sistema novamente ao valor de *setpoint*.

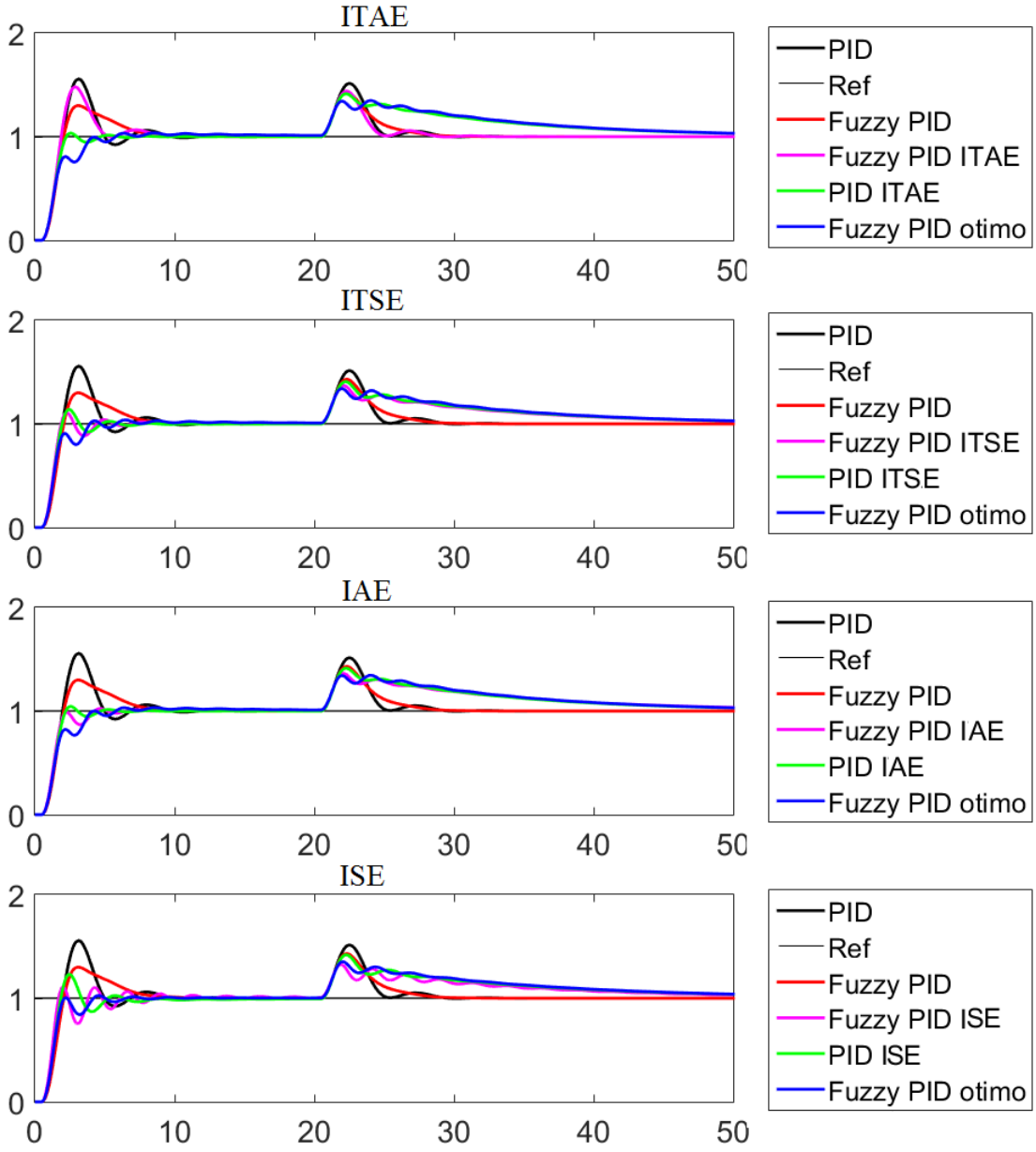


Figura 5.15: Análise temporal do primeiro sistema não linear com perturbação, com uso do primeiro controlador.

Tabela 5.7: Valores dos ganhos K_p, K_i, K_d e valores de M_p, T_s e T_r do primeiro sistema não linear, com o primeiro controlador.

	$\mathbf{K_p}$	$\mathbf{K_i}$	$\mathbf{K_d}$	$\mathbf{M_p}(\%)$	$\mathbf{T_r}(\mathbf{s})$	$\mathbf{T_s}(\mathbf{s})$
PID	1,32	0,601	0,7246	55,745	0,9735	9,2356
Fuzzy PID	1,32	0,601	0,7246	30,430	1,0297	8,7707
Fuzzy PID ITAE	1,4874	0,7251	0,6044	47,715	0,9036	8,6612
Fuzzy PID ITSE	1,962	0,2052	0,9945	10,981	0,9046	6,5038
Fuzzy PID IAE	1,7831	0,1778	1,105	2,1063	1,0423	6,1388
Fuzzy PID ISE	2,3848	0,2155	1,3911	12,142	0,7820	11,669
PID ITAE	1,5892	0,173	1,4266	3,1035	1,1026	4,4084
PID ITSE	1,809	0,1903	1,3683	14,266	0,9442	5,6225
PID IAE	1,6067	0,175	1,4121	4,9156	1,0815	4,4077
PID ISE	1,884	0,1724	1,1708	23,394	0,8938	7,8865
Fuzzy PID Ótimo ITAE	1,5892	0,173	1,4266	2,2585	2,7527	8,7936
Fuzzy PID Ótimo ITSE	1,809	0,1903	1,3683	3,0227	1,2565	9,0708
Fuzzy PID Ótimo IAE	1,6067	0,175	1,4121	2,3094	2,7595	8,8658
Fuzzy PID Ótimo ISE	1,884	0,1724	1,1708	2,1529	0,9905	6,1534

Segundo Controlador

A Figura 5.16, apresenta as simulações efetuadas para este primeiro sistema. Já na Tabela 5.8, é possível observar os parâmetros temporais do sistema, para os gráficos obtidos anteriormente.

Quanto aos índices de desempenho usados, para o método PID-Difuso adaptativo otimizado, o *ITAE* apresentou uma melhor resposta. Já com o método PID-Difuso adaptativo com os ganhos da sintonia Ziegler-Nichols já otimizados, todos os índices apresentaram respostas similares. No entanto, o *ITAE* e o*ITSE* mostraram-se ligeiramente melhores por atingir o *setpoint* no período inicial do controlo e estabilizar posteriormente. Pelo contrário o *IAE* apenas ficou perto do *setpoint* no período inicial e estabilizou posteriormente e o *ISE* apresentou respostas um pouco oscilatórias.

Na Figura 5.17 é possível observar a resposta do sistema com a introdução de uma perturbação de valor igual a $0,8$ aos 20 segundos de simulação. Para este sistema, a melhor resposta obteve-se com a utilização do PID-Difuso não otimizado, pois com os índices de desempenho, esta demorou a estabilizar novamente.

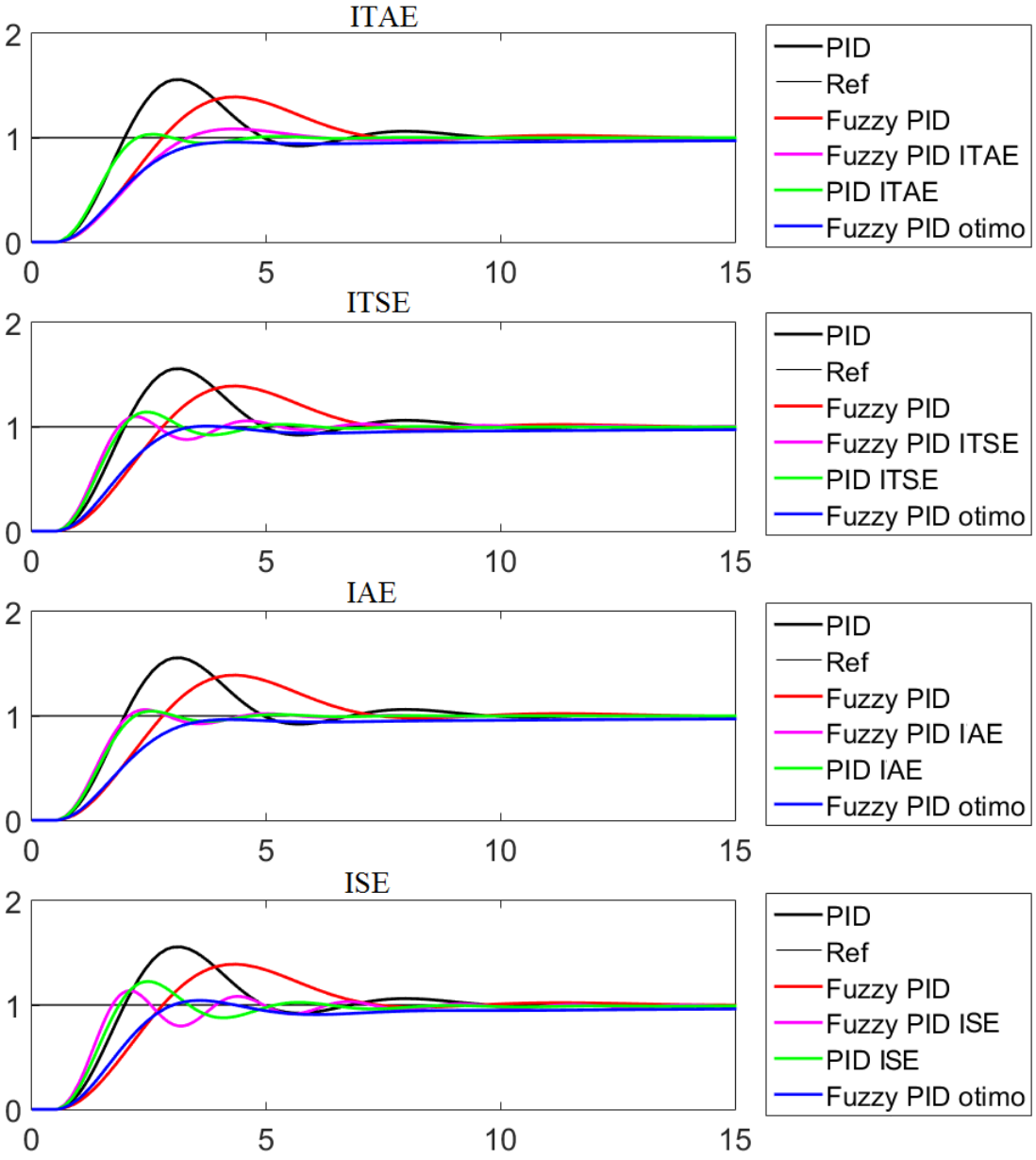


Figura 5.16: Análise temporal do primeiro sistema não linear com uso do segundo controlador.

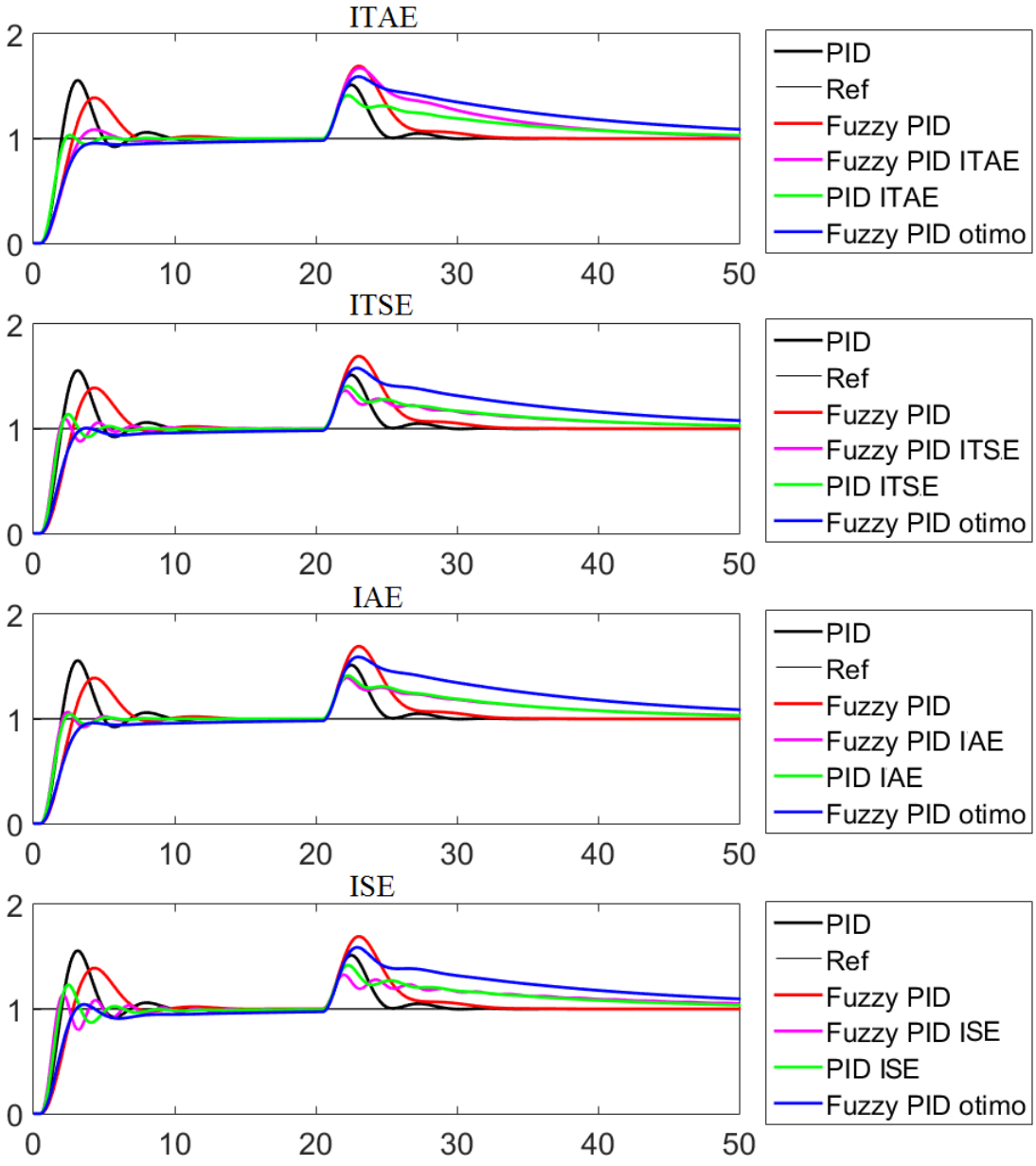


Figura 5.17: Análise temporal do primeiro sistema não linear com perturbação, com uso do segundo controlador.

Tabela 5.8: Valores dos ganhos K_p, K_i, K_d e valores de M_p, T_s e T_r do primeiro sistema não linear, com o segundo controlador.

	$\mathbf{K_p}$	$\mathbf{K_i}$	$\mathbf{K_d}$	$\mathbf{M_p}(\%)$	$\mathbf{T_r}(\mathbf{s})$	$\mathbf{T_s}(\mathbf{s})$
PID	1,32	0,601	0,7246	55,745	0,9735	9,2356
Fuzzy PID	1,32	0,601	0,7246	39,226	1,5217	11,923
Fuzzy PID ITAE	1,419	0,2624	0,9372	8,7164	1,8621	5,8013
Fuzzy PID ITSE	4,0877	0,4046	3,6222	9,3781	0,8808	7,2999
Fuzzy PID IAE	3,5123	0,3562	3,0667	6,1444	0,9992	5,2021
Fuzzy PID ISE	4,6483	0,3238	4,0255	15,302	0,7860	9,3863
PID ITAE	1,5892	0,173	1,4266	3,3437	1,1063	4,3835
PID ITSE	1,809	0,1903	1,3683	14,330	0,9467	5,6497
PID IAE	1,6067	0,175	1,4121	4,6013	1,0816	4,3913
PID ISE	1,884	0,1724	1,1708	23,610	0,8898	7,8115
Fuzzy PID Ótimo ITAE	1,5892	0,173	1,4266	0	2,0304	8,1559
Fuzzy PID Ótimo ITSE	1,809	0,1903	1,3683	3,4670	1,6795	7,8080
Fuzzy PID Ótimo IAE	1,6067	0,175	1,4121	0	1,9833	8,1459
Fuzzy PID Ótimo ISE	1,884	0,1724	1,1708	8,4736	1,5281	8,0589

5.2.2 Segundo Sistema

Para a segunda simulação foi usado o seguinte sistema não linear e variante no tempo [27]:

$$\frac{d^2y(t)}{dt^2} + e^{-0,2t}\frac{dy(t)}{dt} + e^{-5t}sin(2t+6)y = u(t) \tag{5.6}$$

A sua representação no espaço de estados é dada por:

$$\begin{cases} \dot{x}_1(t) = x_2(t) \\ \dot{x}_2(t) = -e^{-0,2t}x_2(t) - e^{-5t}sin(2t+6)x_1(t) + u(t) \end{cases} \tag{5.7}$$

O diagrama de blocos do sistema é apresentado na Figura 5.18.

Primeiro Controlador

As simulações efetuadas para este segundo sistema não linear encontram-se representadas na Figura 5.19. Na Tabela 5.9, apresenta-se os parâmetros temporais do sistema, para os gráficos obtidos anteriormente.

Neste sistema, observou-se nos métodos de controlo base, que se obteve novamente uma melhor resposta com o uso do controlador PID-Difuso adaptativo não otimizado, comparativamente ao controlo PID com o método de sintonia Ziegler-Nichols. O mesmo se pode comprovar, analisando a Tabela 5.9, onde se pode observar que o controlador PID-Difuso adaptativo não

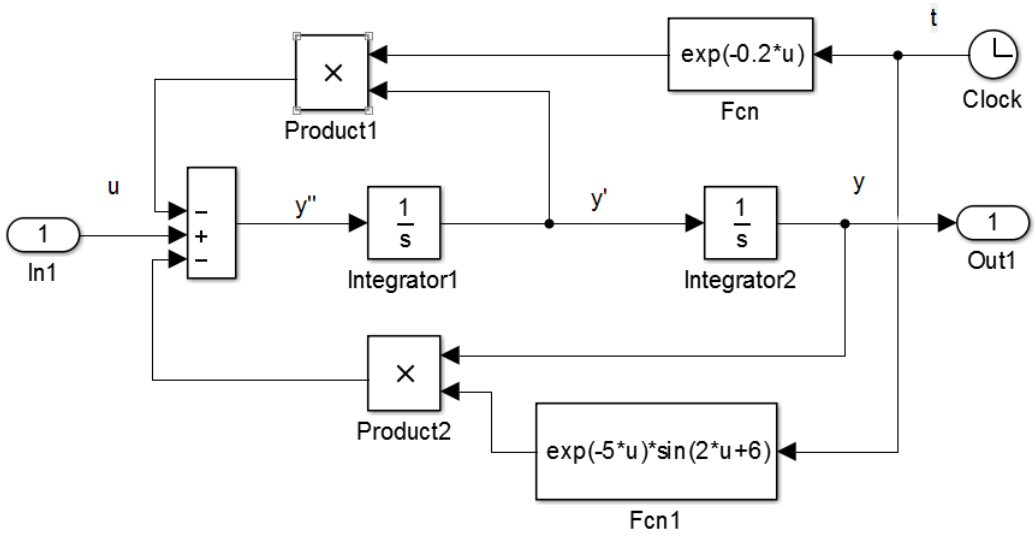


Figura 5.18: Diagrama de blocos do segundo sistema não linear.

otimizado obteve um *overshoot* relativamente inferior controlo PID. No entanto, este ainda é valor relativamente alto para um controlo eficaz do sistema.

Quanto aos restantes métodos aplicadas com o auxílio dos índices de desempenho, apenas o *ITAE* se mostrou pouco viável para este sistema. Todos os restantes índices apresentaram respostas semelhantes, sendo o *ISE* o mais eficaz para este controlo. Já quanto ao método utilizado, o que apresentou uma melhor resposta, foi o PID-Difuso adaptativo com uso dos ganhos do PID já otimizados, tendo também valores de *overshoot* mais baixos, como se pode comprovar na Tabela 5.9.

Na Figura 5.20 é possível observar a resposta do sistema com a introdução de uma perturbação de valor igual a $0,8$ aos 20 segundos de simulação. Para este sistema, a melhor resposta obteve-se com a utilização do PID-Difuso adaptativo não otimizado. A otimização com recurso aos índices de desempenho mostrou-se pouco eficaz neste caso. Com o índice *ITAE* obteve-se uma boa resposta, sendo esta ainda melhor que a resposta do PID-Difuso adaptativo não otimizado. No entanto, a pior resposta que este índice de desempenho apresentou nos instantes iniciais do controlo, torna este um controlo não adequado. Já com os restantes índices de desempenho, o sistema não conseguiu voltar novamente ao valor de *setpoint* após a perturbação no sistema.

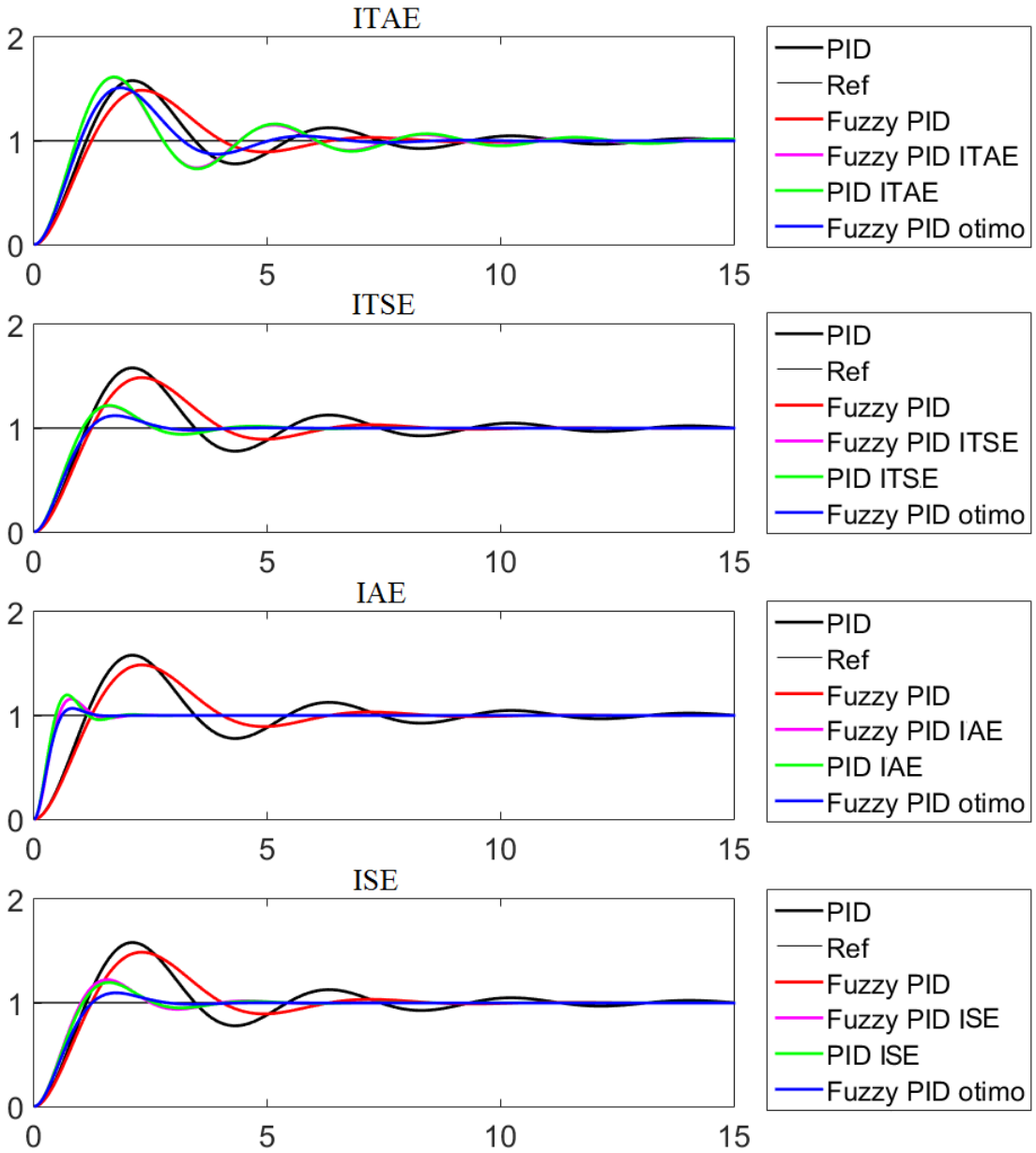


Figura 5.19: Análise temporal do segundo sistema não linear, com uso do primeiro controlador.

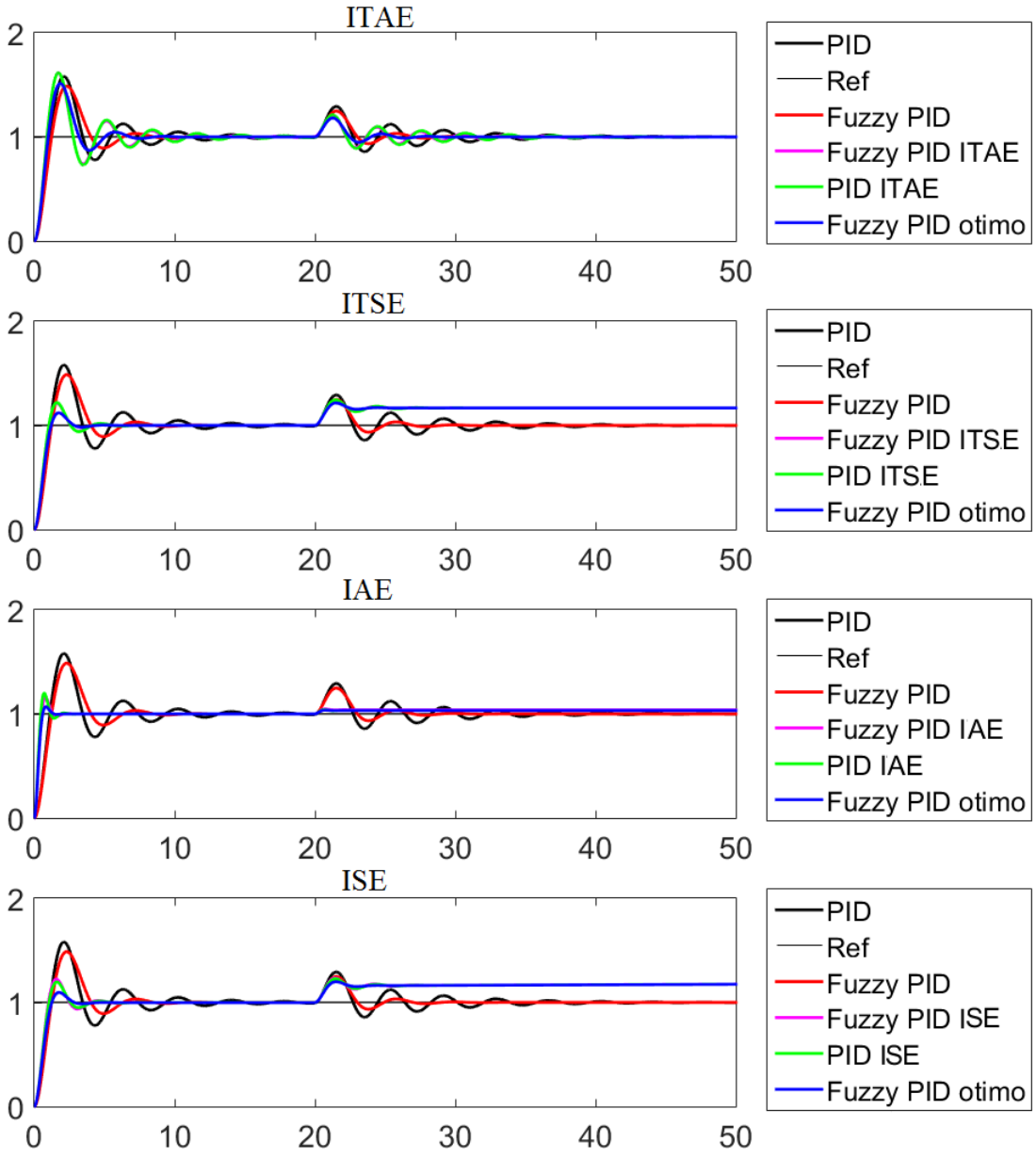


Figura 5.20: Análise temporal do segundo sistema não linear com perturbação, com uso do primeiro controlador.

Tabela 5.9: Valores dos ganhos K_p, K_i, K_d e valores de M_p, T_s e T_r do segundo sistema não linear, com o primeiro controlador.

	$\mathbf{K_p}$	$\mathbf{K_i}$	$\mathbf{K_d}$	$\mathbf{M_p}(\%)$	$\mathbf{T_r}(\mathbf{s})$	$\mathbf{T_s}(\mathbf{s})$
PID	3	2,1171	1,0627	57,813	0,7563	14,160
Fuzzy PID	3	2,1171	1,0627	48,757	0,8405	7,8709
Fuzzy PID ITAE	4,243	3,5386	0,7788	60,130	0,6118	13,431
Fuzzy PID ITSE	4,7653	0	0,6875	21,434	0,7065	3,7933
Fuzzy PID IAE	20,567	-1,773	2,3669	15,917	0,3596	1,7349
Fuzzy PID ISE	4,957	-0,015	0,6784	22,458	0,6828	3,7248
PID ITAE	4,2522	3,5336	1,1677	59,094	0,6169	13,910
PID ITSE	4,7652	0	1,0312	21,863	0,7055	3,8414
PID IAE	24,610	-0,000	3,5728	19,762	0,3139	1,6690
PID ISE	4,957	-0,015	1,1706	19,712	0,7096	3,7540
Fuzzy PID Ótimo ITAE	4,2522	3,5336	1,1677	51,253	0,6725	6,4540
Fuzzy PID Ótimo ITSE	4,7652	0	1,0312	12,064	0,8114	2,6342
Fuzzy PID Ótimo IAE	24,610	0	3,5728	6,8275	0,3935	1,1809
Fuzzy PID Ótimo ISE	4,957	-0,015	1,1706	9,7345	0,8326	2,6164

Segundo Controlador

Na Figura 5.21, é possível observar as simulações efetuadas para este segundo sistema não linear. A Tabela 5.10, apresenta os parâmetros temporais do sistema, para os gráficos obtidos anteriormente.

No que toca aos índices de desempenho utilizados de modo a otimizar a resposta do sistema, o *ISE* foi o que apresentou melhores resultados aquando a utilização do método PID-Difuso adaptativo com recurso aos ganhos PID já otimizados.

Na Figura 5.22 é possível observar a resposta do sistema com a introdução de uma perturbação de valor igual a $0,8$ aos 20 segundos de simulação. Para este sistema, a melhor resposta obteve-se novamente com a utilização do PID-Difuso adaptativo não otimizado. Assim como no primeiro controlador, a otimização com recurso aos índices de desempenho mostrou-se pouco eficaz neste caso. Com o método ITAE obteve-se uma boa resposta à perturbação. Contudo, a pior resposta que este índice de desempenho apresentou no período inicial do controlo, torna este um controlo não adequado. Já com os restantes índices de desempenho, o sistema não conseguiu voltar novamente ao valor de *setpoint* após a perturbação no sistema.

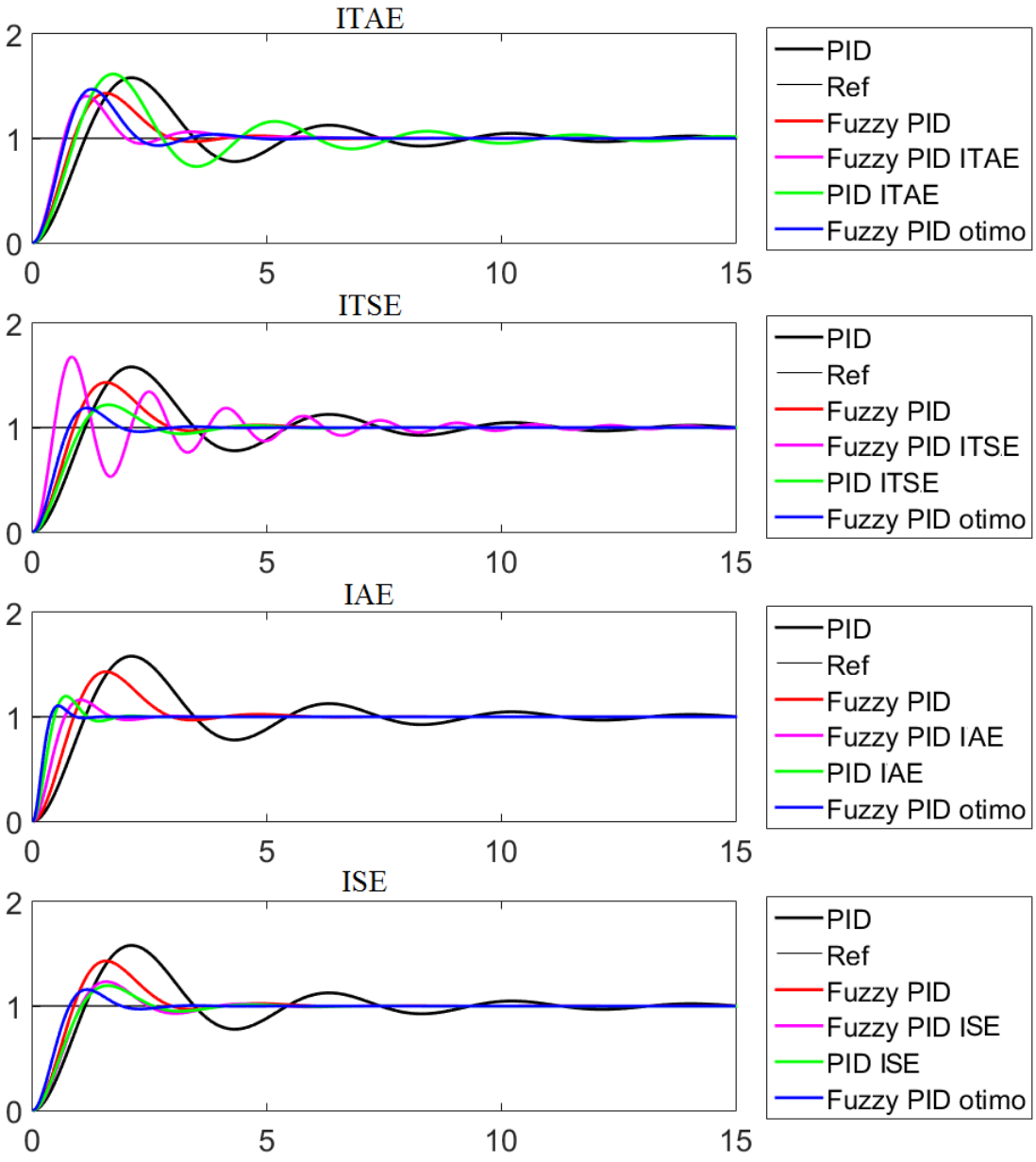


Figura 5.21: Análise temporal do segundo sistema não linear com uso do segundo controlador.

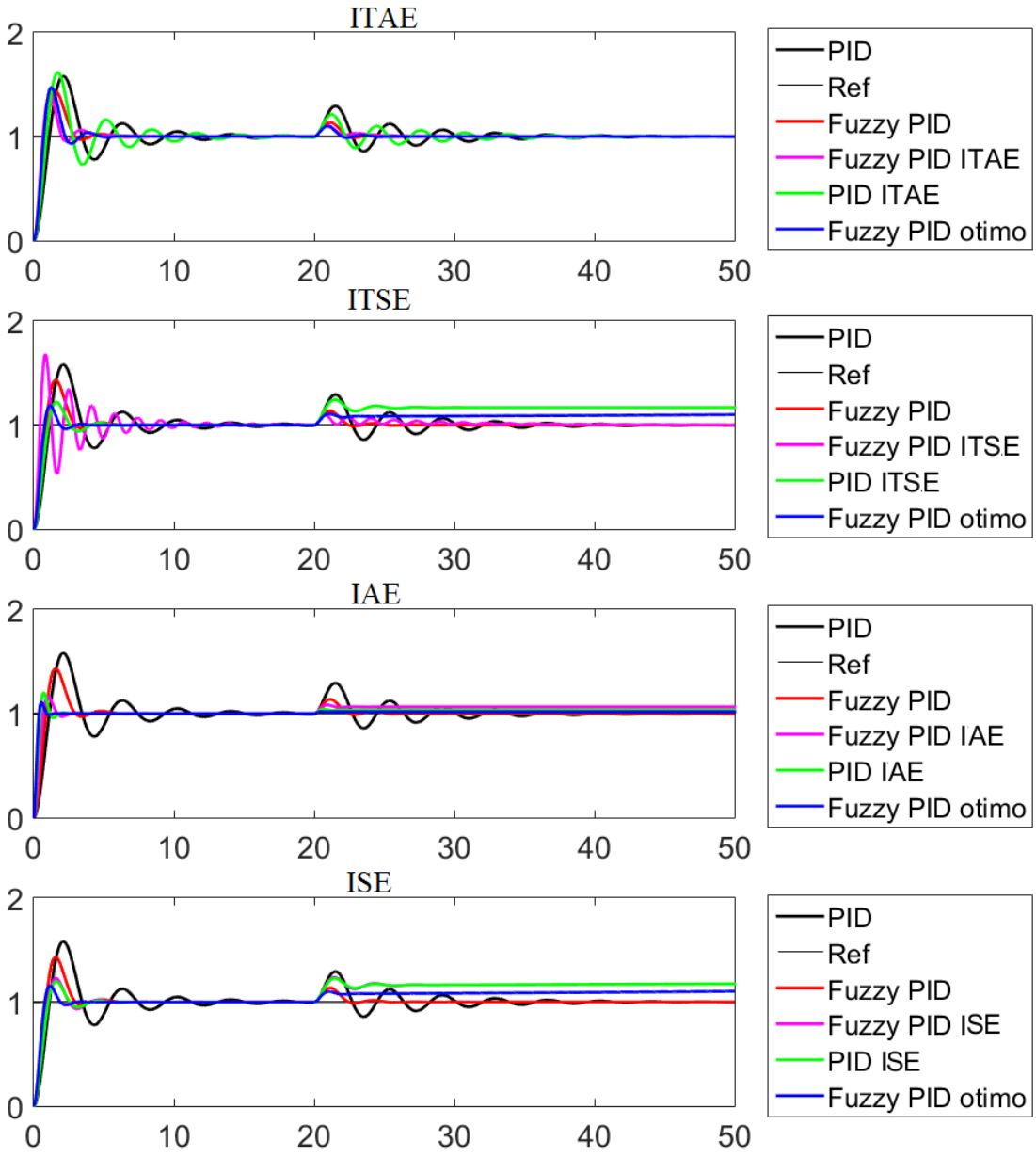


Figura 5.22: Análise temporal do segundo sistema não linear com perturbação, com uso do segundo controlador.

Tabela 5.10: Valores dos ganhos K_p, K_i, K_d e valores de M_p, T_s e T_r do segundo sistema não linear, com o segundo controlador.

	$\mathbf{K_p}$	$\mathbf{K_i}$	$\mathbf{K_d}$	$\mathbf{M_p}(\%)$	$\mathbf{T_r}(\mathbf{s})$	$\mathbf{T_s}(\mathbf{s})$
PID	3	2,1171	1,0627	57,813	0,7563	14,160
Fuzzy PID	3	2,1171	1,0627	43,250	0,5944	5,1654
Fuzzy PID ITAE	4,7194	2,0076	0,9233	40,408	0,4510	4,1316
Fuzzy PID ITSE	7,3861	0,8427	0,2374	68,889	0,3005	14,101
Fuzzy PID IAE	6,2692	8,4728	1,3012	16,255	0,4633	2,2936
Fuzzy PID ISE	2,4785	-0,007	0,5088	23,553	0,6920	4,7162
PID ITAE	4,2522	3,5336	1,1677	59,094	0,6169	13,910
PID ITSE	4,7652	0	1,0312	21,863	0,7055	3,8414
PID IAE	24,610	0	3,5728	19,762	0,3139	1,6690
PID ISE	4,957	-0,015	1,1706	19,712	0,7096	3,7540
Fuzzy PID Ótimo ITAE	4,2522	3,5336	1,1677	47,249	0,4743	4,3865
Fuzzy PID Ótimo ITSE	4,7652	0	1,0312	18,887	0,5202	2,6762
Fuzzy PID Ótimo IAE	24,610	0	3,5728	10,561	0,2570	0,8329
Fuzzy PID Ótimo ISE	4,957	-0,015	1,1706	15,914	0,5309	2,5610

5.3 Comparação de Resultados

Nesta secção será feita uma análise e comparação direta entre o controlador PID sintonizado pelo *pidtune* (sistemas lineares) e pelo método de Ziegler-Nichols (sistemas não lineares) com o controlador PID-Difuso adaptativo não otimizado através dos dois controladores desenvolvidos.

5.3.1 Sistemas Lineares

Na Figura 5.23 é apresentada uma comparação entre o controlo PID pelo método *pidtune* e o controlo PID-Difuso adaptativo não otimizado com recurso aos dois controladores propostos neste trabalho, correspondentes ao sistema linear $P_1(s)$ (5.1).

Como é possível observar, para este sistema e em regra para os sistemas não lineares estudados neste projeto, o segundo controlador PID-Difuso adaptativo não otimizado apresentou uma resposta mais eficaz, comparativamente com o primeiro controlador PID-Difuso adaptativo não otimizado e consequentemente com o controlador PID. Relativamente aos valores obtidos, o segundo controlador apresentou um *overshoot* de $10,10\%$, enquanto que o primeiro controlador obteve um *overshoot* de $15,67\%$. Já o controlador PID possui $24,99\%$. Já relativamente aos valores de T_s e T_r, ambos os controladores apresentaram resultados bastante similares.

Relativamente à resposta à perturbação, o primeiro controlador apresentou uma melhor resposta comparativamente com os restantes, atingindo a estabilidade mais rapidamente.

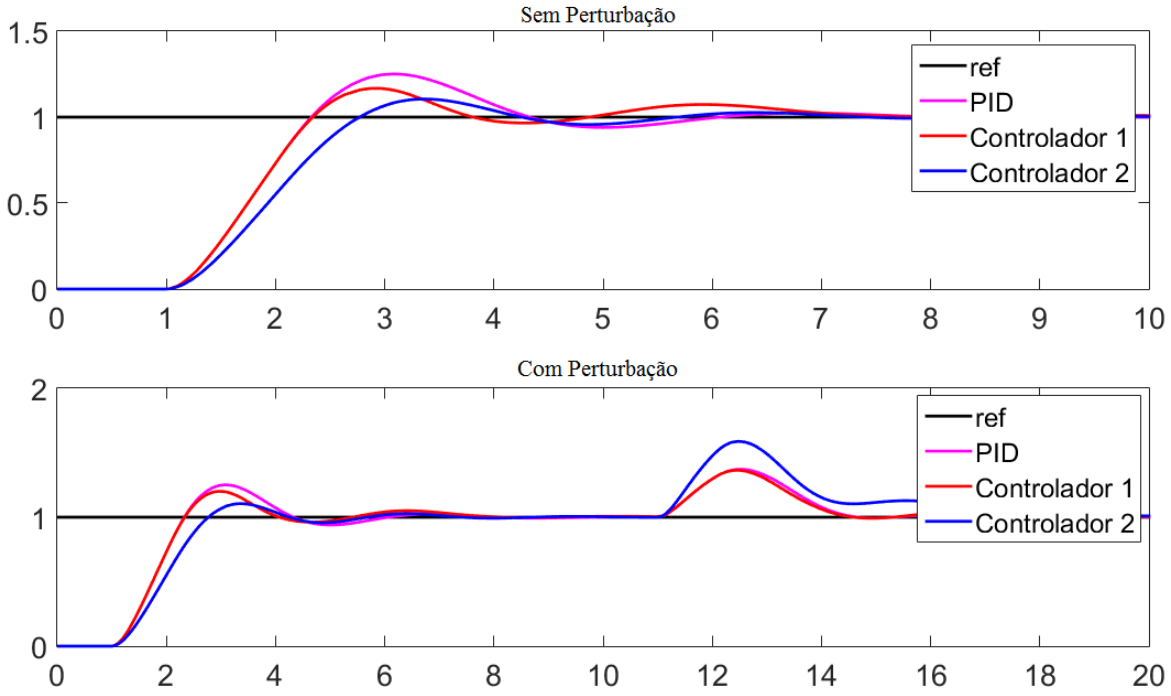


Figura 5.23: Comparação de respostas obtidas para os controlos do sistema linear $P_1(s)$.

5.3.2 Sistemas Não Lineares

A Figura 5.24 apresenta uma comparação entre o controlo PID pelo método de Ziegler-Nichols e o controlo PID-Difuso adaptativo não otimizado com recurso aos dois controladores propostos neste trabalho, correspondentes ao primeiro sistema não linear (5.4).

Através da Figura 5.24 é possível concluir visualmente que o controlador PID foi o que apresentou uma resposta menos eficaz. Já relativamente aos controladores PID-Difusos adaptativos não otimizados, foi obtida uma melhor resposta com o primeiro controlador.

Pode-se fundamentar estas conclusões através da Tabela 5.8 e 5.7. Para o primeiro controlador, foi obtido um valor de *overshoot* de $30,43\%$ enquanto que para o segundo controlador o valor foi de $39,22\%$. Embora a resposta do segundo controlador tenha estabilizado mais rapidamente em comparação com o primeiro controlador, ambas as respostas atingiram o *setpoint* praticamente ao mesmo tempo. Deste modo e juntando todos os aspetos mais relevantes, é possível afirmar que o primeiro controlador foi mais eficaz para este tipo de sistema não linear.

Relativamente à resposta à perturbação, o primeiro controlador continuou a apresentar uma melhor resposta comparativamente com os restantes.

É possível observar na Figura 5.25 a comparação realizada entre o controlo PID pelo método de Ziegler-Nichols e o controlo PID-Difuso adaptativo não otimizado com recurso aos dois controladores desenvolvidos, correspondentes ao segundo sistema não linear e invariante no

tempo (5.6).

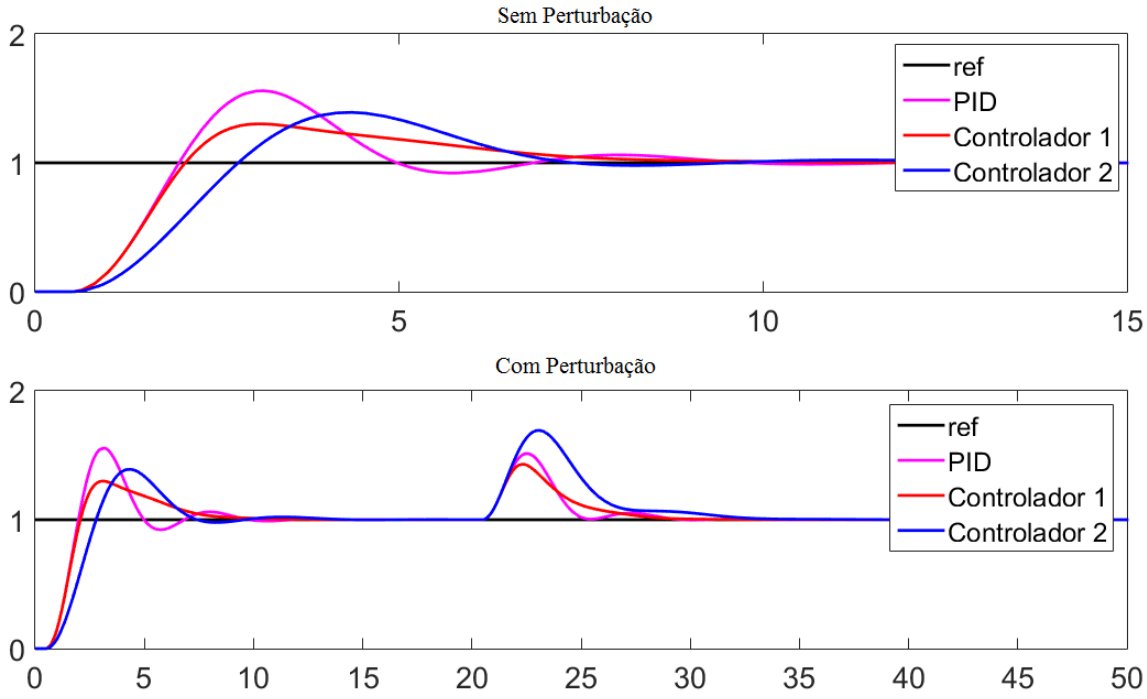


Figura 5.24: Comparação de respostas obtidas para os controlos do primeiro sistema não linear.

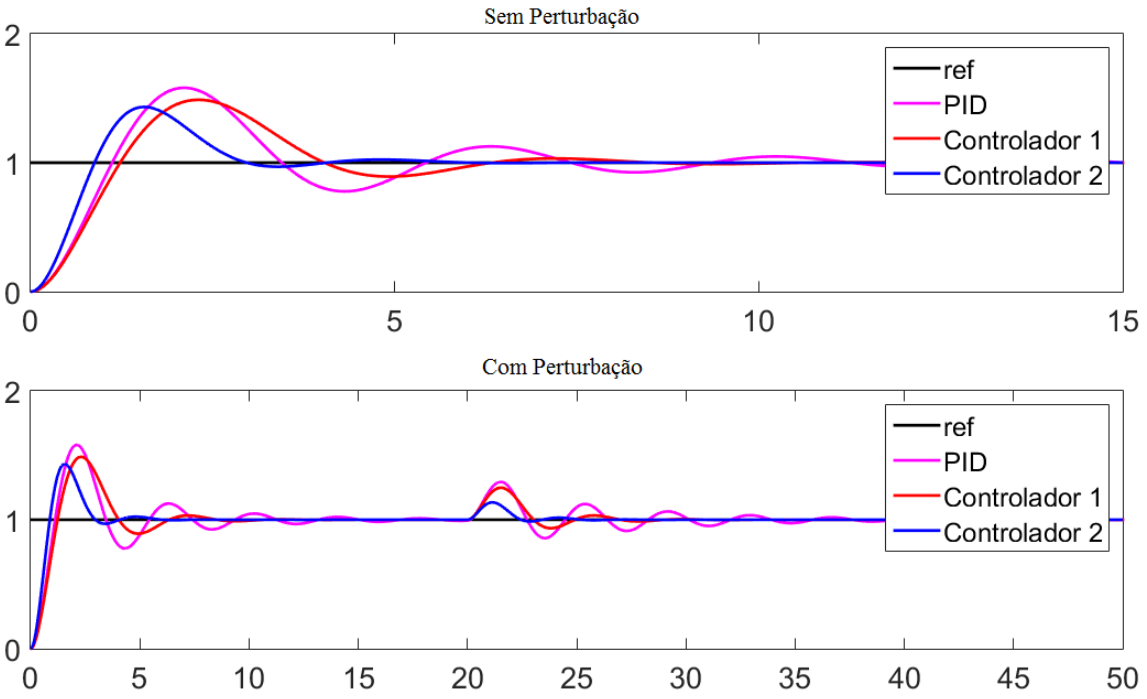


Figura 5.25: Comparação de respostas obtidas para os controlos do segundo sistema não linear e variante no tempo.

Assim como aconteceu no primeiro sistema não linear, o controlador PID foi o que apresentou a pior resposta de controlo. Já quanto aos controladores PID-Difusos adaptativos não otimizados, o segundo controlador foi o que obteve uma melhor resposta do sistema, obtendo um menor *overshoot* $(43,25\%)$ e estabilizando mais rapidamente que o primeiro controlador. Relativamente ao primeiro controlador, este obteve um *overshoot* de $48,75\%$ e uma resposta ligeiramente mais oscilatória.

Relativamente à resposta à perturbação, o segundo controlador continuou a apresentar uma melhor resposta comparativamente com os restantes.

Capítulo 6

Conclusões

Este trabalho tinha como objetivo principal, o desenvolvimento de controladores PID-Difusos adaptativos. O qual foi atingido com resultados bastante satisfatórios, tendo em conta o tempo e os recursos disponíveis. Como complemento, foram utilizados diferentes métodos de controlo e de otimização, de modo a poder existir um termo de comparação sobre a precisão e estabilidade de ambos os métodos.

Inicialmente, começou-se por fazer um estudo mais aprofundado sobre controladores PID e controladores lógicos difusos, assim como a sua aplicação no MATLAB. Este *software* de cálculo, em conjunto com o *software* de simulação Simulink, mostraram-se poderosas ferramentas de desenvolvimento de trabalho, sem os quais seria muito mais difícil de alcançar os resultados obtidos ao longo deste projeto.

No início do desenvolvimento dos controladores, começou-se por implementar um controlador PID sintonizado pelo método de Ziegler-Nichols em malha fechada. Sendo este um método bastante conhecido e utilizado na indústria atualmente, seria uma boa escolha para ter como controlo base para os métodos a implementar subsequentemente. Posteriormente, como este último método não fornece respostas muito satisfatórias no que diz respeito a um bom controlo de um sistema, foi implementado um segundo controlador PID, mas com recurso à função *pidtune* do MATLAB. A razão para se ter implementado este segundo método PID, prende-se com o facto desta função originar respostas mais otimizadas comparativamente com o método de Ziegler-Nichols em malha fechada.

Numa fase seguinte foi desenvolvida um primeiro controlador PID-Difuso adaptativo, que possuía como base um controlador PID. Os ganhos deste eram sintonizados previamente, sendo considerados como o "núcleo"do controlador. Como estes valores são invariantes no tempo e de modo a fazer a adaptação do sistema, foi introduzido um sistema lógico difuso, que adapta os ganhos das suas saídas em tempo real, adaptando assim a saída do controlador em torno dos

valores do "núcleo"do sistema.

Posteriormente, foi desenvolvido um segundo controlador PID-Difuso adaptativo. Este segundo controlador tem como base o mesmo processo que o anterior. Ou seja, possui também um controlador PID pré-sintonizado com valores invariantes no tempo e um controlador lógico difuso que tem como função adaptar em tempo real a saída do controlador.

Relativamente às regras difusas, foi desenvolvida uma base de 49 regras, cujos intervalos de pertença foram escolhidos por método empírico. Já na escolha das funções de pertença, foram utilizadas funções triangulares por estas serem as mais predominantes nos métodos de controlo difuso e por apresentarem bons resultados, mesmo na indústria atual. Quanto à construção das regras teve-se como base alguns aspetos, como a limitação da ação integral, de modo a ser evitado um *overshoot* bastante elevado e especial atenção à saída derivativa do controlador lógico difuso, por esta ser essencial na obtenção de um *overshoot* reduzido e tentar evitar oscilações perto do valor de *setpoint*. De salientar que a base de regras utilizada foi a mesma para ambos os controladores, tanto para os sistemas lineares como não lineares.

Quanto aos resultados obtidos ao longo das simulações, o segundo controlador PID-Difuso adaptativo mostrou-se mais eficaz que o primeiro controlador, comparativamente ao controlador PID para os sistemas lineares estudados. Enquanto que com o primeiro controlador PID-Difuso adaptativo não otimizado se tenha obtido respostas bastante semelhantes comparativamente com o controlador PID com ganhos do *pidtune*, embora ligeiramente melhores, já com o segundo controlador PID-Difuso adaptativo não otimizado as respostas eram notoriamente melhores e mais eficazes.

Posteriormente e ainda para os sistemas lineares, com a otimização dos controladores com recurso aos índices de desempenho, notou-se uma predominância de certos índices a fornecer melhores respostas que outros. Como foi possível observar no capítulo referente aos resultados, para ambos os controladores PID-Difusos adaptativos, o índice de desempenho $ITAE$ foi o que mais vezes apresentou melhores respostas. Já com a inclusão de uma perturbação, o índice de desempenho $ITSE$ também passou a estar a par do $ITAE$ na predominância de melhores respostas. No entanto, visto que os instantes iniciais das respostas obtidas com o índice $ITSE$ eram muitas vezes pouco estáveis, é possível assim afirmar que o $ITAE$ é o índice de desempenho mais constante das suas respostas.

Relativamente os sistemas não lineares, sendo estes sistemas naturalmente mais difíceis de controlar, os índices de desempenho não se mostraram tão fiáveis na apresentação de respostas. Para ambos os sistemas, o índice de desempenho IAE foi o mais predominante a fornecer melhores respostas nos instantes iniciais das simulações. Já relativamente às respostas à perturbação

inserida nos sistemas, os índices de desempenho mostraram-se pouco eficazes. Enquanto que para o primeiro sistema não linear, este demorou imenso tempo a voltar ao seu ponto de estabilidade, já no segundo sistema não linear e sendo este variante no tempo, os índices de desempenho não conseguiram devolver o sistema ao seu valor de *setpoint*.

De um modo geral, para os sistemas não lineares, o controlo através do método PID-Difuso adaptativo não otimizado mostrou-se mais eficaz. Enquanto que para o primeiro sistema não linear se tenha obtido uma melhor resposta com o primeiro controlador, já para o segundo sistema não linear foi o segundo controlador o que apresentou melhores resultados. No entanto, ambos apresentam boas respostas e melhores que o controlador PID, tornando-os mais fiáveis e eficazes.

De uma visão generalizada, o controlador PID-Difuso adaptativo mostrou-se notoriamente mais eficaz que um comum controlador PID sintonizado pelo método de Ziegler-Nichols em malha fechada. Deste modo, torna-se assim uma boa opção pela sua flexibilidade de controlo e relativa facilidade de implementação. Já a sua otimização através dos índices de desempenho, apenas se mostrou mais eficaz na aplicação em sistemas lineares, obtendo assim melhores respostas.

Relativamente a todos os resultados obtidos, visto que estão de acordo com os objetivos e ideias traçadas inicialmente, pode-se concluir de forma global que, tanto para os sistemas lineares como não lineares, consegue-se melhores resultados com o controlador PID-Difuso adaptativo do que com o PID. A grande vantagem dos controladores PID-Difusos adaptativos é a não necessidade de modelação do sistema. São escolhidas as variáveis de entrada e de saída e definidas regras linguísticas de como estas variáveis interagem entre si, mais ou menos de forma a modelar a decisão humana. Como o projeto destes controladores tem um fator subjetivo, a experiência do projetista é um dado importante.

Com a elaboração desta tese, foi possível adquirir conhecimentos associados à lógica difusa e, consequentemente, no projeto e implementação de controladores difusos. O enquadramento da lógica difusa no controlo de sistemas permitiu solidificar algumas bases que possuía da área de controlo de sistemas, assim como adquirir novos conhecimentos que serão fundamentais para o meu futuro.

Como melhorias futuras para este trabalho seria desejável melhorar o desempenho dos controladores propostos neste projeto. Sabe-se que um sistema difuso depende em grande parte da sua base de regras, do tipo de agregação, do tipo de colapsagem, do número de conjuntos difusos, do tipo de inferência e do tipo de funções de pertença. Assim, para o melhoramento do desempenho dos sistemas apresentados ao longo deste projeto, o ajuste destes valores mencionados

poderia fazer parte de um dos melhoramentos futuros a fazer.

Um outro melhoramento futuro que se poderá ter em conta, é a aplicação destes conceitos e destes controladores na robótica e não só. Uma primeira aplicação poderia passar por efetuar o controlo de velocidade de um sistema servo.

Bibliografia

[1] G. Feng, "A survey on analysis and design of model-based fuzzy control systems", 2006.

[2] H. O. Wang, K. Tanaka, e M. F. Griffin, "An approach to fuzzy control of nonlinear systems: stability and design issues", 1996.

[3] C. C. Lee, "Fuzzy logic in control systems: fuzzy logic controller", IEEE JSMC, 1990.

[4] F. Castrillón, M. Osorio, R.Vásquez, "Comparison between different methods for tuning PID controllers", Proceedings of the 2006 International Conference on Dynamics, Instrumentation and Control CDIC. Querétaro, México, 2006.

[5] K. Ogata, "Engenharia de Controlo Moderno", Prentice Hall, 4ª edição, São Paulo, 2003.

[6] J. Carvalho, "Sistemas de Controle Automático", LTC Editora, Rio de Janeiro, ISBN, 200.

[7] J. Carneiro, "Modelação e Controlo de Atuadores Pneumáticos Utilizando Redes Neuronais Artificiais", Tese de Doutoramento em Engenharia Mecânica, Faculdade de Engenharia da Universidade do Porto, Portugal, 2007.

[8] P. Joaquim Jesus, "Sintonia do controlador PID, com Algoritmo de Otimização por Grupo de Partículas.", Dissertação de Mestrado em Engenharia Eletrotécnica e de Computadores, Universidade de Trás-os-Montes e Alto Douro, 2008. Acedido em: 31 Janeiro 2016. Disponível em: https://repositorio.utad.pt/bitstream/10348/185/1/msc_jjpereira.pdf

[9] C. Graham Goodwin, F. Stefan Graebe, E. Mario Salgado, "Control System Design", Prentice Hall, 2001.

[10] National Instruments, "Explicando a Teoria PID", Dezembro 2011. Acedido em: 31 Janeiro 2016. Disponível em: http://www.ni.com/white-paper/3782/pt/

[11] M. Marcelo, "Controle PID com aproximação Digital para utilização no PIC", MicroControlado 2012. Acedido em: 31 Janeiro 2016. Disponível em: http://microcontrolado.com/controle-pid-no-pic/

[12] G. Sérgio Augusto Pereira, "Comparação entre Métodos de Identificação De Plantas com Respostas ao Degrau Monotonicamente Crescentes e Sintomas de Controladores PID", Projeto de Grau de Engenheiro Eletricista, Universidade Federal do Rio de Janeiro, 2008.

[13] B. Ramiro de Sousa, "Controlo PID - Parte 1", SISCA - Sistemas de Controlo Avançado, Ano letivo 2015/16.

[14] V. VanDore, "Auto-tunning Control Using Ziegler Nichols", 9th IEEE/IAS International Conference on Industry Applycations - INDUSCON, 2006.

[15] Johnson, Michael A. Mohammad Moradi., "PID Control - new Identification and Design Methods", Springer-verlag, London, 2005.

[16] P. Jan Erik Mont Gomery, "Aplicação Prática do Método de Sintonia de Controladores PID Utilizando o Método do Relé com Histerese", Dissertação de Mestrado de Pós-Graduação em Engenharia Elétrica e de Computação, Área de Automação e Sistemas, Universidade federal do Rio Grande do Norte Centro de Tecnologia, 2012.

[17] T.Heske, "Fuzzy Logic for Real World Design", Annabooks, San Diego, USA, 1996.

[18] The MathWorks, "Fuzzy Logic Toolbox - User's Guide MATLAB", Version 2.2.22 (Release 2015b), 2015. Acedido em: 1 Fevereiro 2016. Disponível em: http://cn.mathworks.com/help/pdf_doc/fuzzy/fuzzy.pdf

[19] H. Ali, S. Noor, S. Bashi, M. Marhaban, "A Review of Pneumatic Actuators (Modeling and Control)", Australian Journal of Basic and Applied Sciences, 2009.

[20] K. Passino, S. Yurkovich, "Fuzzy Control", Addison Wesley Longman, Menlo Park, CA, 1998.

[21] S. K. Pal, S. Mitra, "Multilayer Perceptron, Fuzzy-Sets, and Classification", Ieee Transactions on Neural Networks, vol. 3, 1992.

[22] R. R. Yager, "On A General-Class of Fuzzy-Connectives", Fuzzy Sets and Systems, vol. 4, 1980.

[23] J. M. Mendel, "Fuzzy logic systems for engineering: A tutorial", Proceedings of the IEEE, vol. 83, 1995.

[24] J. S. R.Jang, C. T. SUN, "Neuro-fuzzy modeling and control", Proceedings of the IEEE, vol. 83, 1995.

[25] F. Martin McNeill, Ellen Thro, "Fuzzy Logic, A Practical Approach", Ap Professional, 1994.

[26] L. Catarina Inês Marques, "Aplicação de técnicas de controlo óptimo difuso em ambientes distribuídos", Dissertação para obtenção do Grau de Mestre em Engenharia Eletrotécnica e de Computadores, Faculdade de Ciências e Tecnologia Universidade Nova de Lisboa, Março de 2012.

[27] X. Dingyu, C. YangQuan, P. Derek Atherton,"Linear Feedback Control", Society for Industrial and Applied Mathematics, 2007.

[28] The MathWorks, Inc, "PID Tuning Algorithm", Documentation, R2016a.

[29] The MathWorks, Inc, "pidtune", Documentation, R2016a.

[30] Y. Yuanhui, Y. Wailing, W. Mingchun, Y. Qiwen, X. Yuncan, "A New Type of Adaptive Fuzzy PID Controller", Jiangsu Key Laboratory of Power Transmission and Distribution Equipment Technology College of Computer and Information, Hohai University, 2010.

[31] J. Jin, H. Huang, J. Sun, Y. Pang, "Study on Fuzzy Self-Adaptive PID Control System of Biomass Boiler Drum Water", College of Electric and Information Engineering, Zhengzhou University of Light Industry, Zhengzhou, China, Journal of Sustainable Bioenergy Systems, 2013.

[32] KJ. Astrom, T.Hägglund ,"PID Controllers: theory, design, and tuning", Instrument Society of America, USA, 2nd Edition,1995.

[33] CD. Richard, HB. Robert, "Modern Control Systems", Prentice Hall, 12th Edition, 2010.

[34] Powell, Michael J. D., "On Search Directions for Minimization Algorithms", Mathematical Programming, 1973.

[35] Y., Wen Ci, "Positive basis and a class of direct search techniques". Scientia Sinica,1979.

[36] Nelder. John, R. Mead, "A simplex method for function minimization", Computer Journal, 1965.

[37] P. Larsson, T. Hagglund, "Comparison Between Robust PID and Predictive PI Controllers with Constrained Control Signal Noise", Department of Automatic Control, Lund University, Lund, Sweden.

[38] O. Karasakal, M. Guzelkaya, I. Eksin, E. Yesil, "An error-based on-line rule weight adjustment method for fuzzy PID controllers", Istanbul Technical University, Faculty of Electrical and Electronics Engineering, Control Engineering Department, Maslak, TR-34469 Istanbul, Turkey, 2011.

Anexo A

A.1. Tutorial *Fuzzy Logic* MATLAB/Simulink

A.1.1 Fuzzy Toolbox

Interface do *Fuzzy Logic*

A interface do *Fuzzy Logic* disponibiliza informações sobre o sistema de inferência difuso (FIS). Para abrir esta *Toolbox*, é necessário digitar o comando ***"fuzzy"*** na linha de comandos do MATLAB.

Na Figura A.1 é possível observar a interface do *Fuzzy Logic*, assim como a descrição de cada secção da mesma.

Para adicionar uma nova variável de entrada/saída, deve-se selecionar a opção -*Edit*-, posteriormente a opção -*Add Variable*-, seguido de -*Input/Output*-.

Editor das Funções de Pertença

Esta é a ferramenta que permite exibir e editar todas as funções de pertença associadas a todas as variáveis de entrada e de saída. Na Figura A.2 é possível observar a interface do editor das funções de pertença, assim como a descrição de cada secção da mesma.

Editor de Regras

Para abrir a interface do editor de regras, é necessário selecionar a opção -*Edit*-, seguido da opção -*Rules*-.

Na Figura A.3 é possível observar a interface do editor de regras, assim como a descrição de cada secção da mesma.

Com base nas descrições das variáveis de entrada e saída já definidas, o editor de regras permite a construção automática das regras. Esta interface permite:

- Criar regras, selecionando as variáveis de entrada e saída nos respetivos campos, selecionando posteriormente o botão *"Add Rule"*.

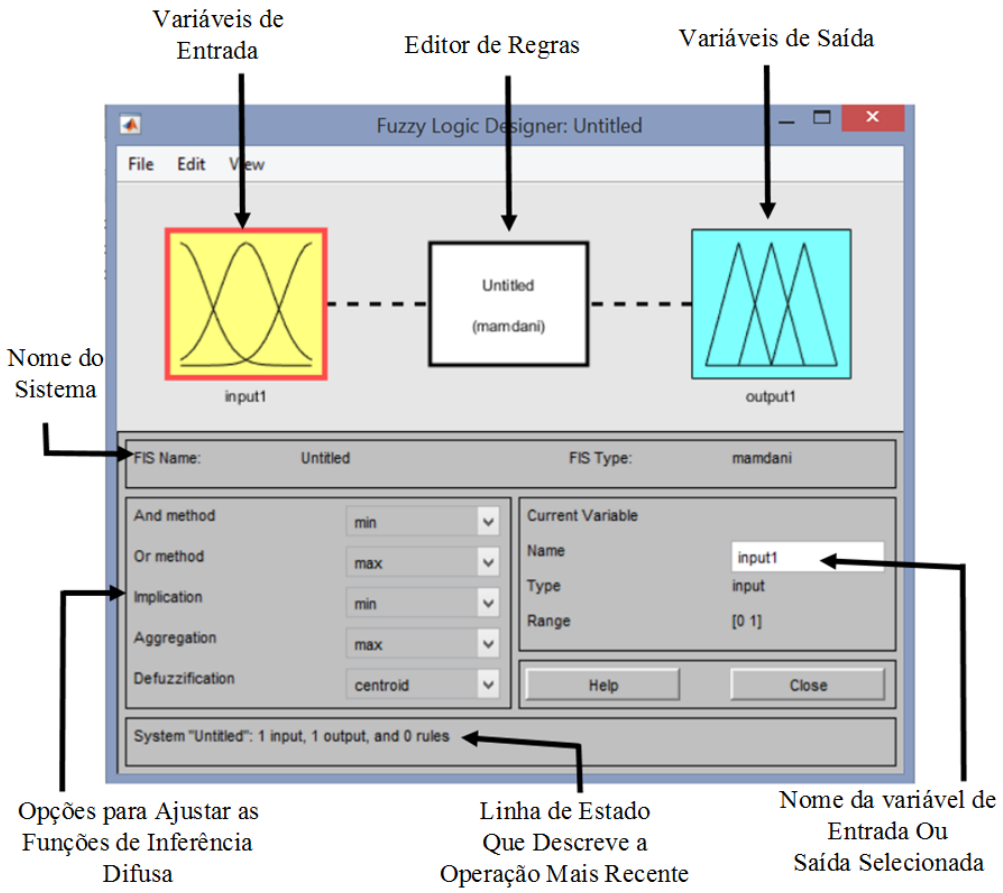


Figura A.1: Estrutura da interface do *Fuzzy Logic* e respetiva descrição de cada secção.

- Eliminar uma regra já criada, selecionando o botão *"Delete Rule"*.
- Editar uma regra já criada, selecionando o botão *"Edit Rule"*.

Visualizador de Regras

Para abrir a interface do visualizador de regras, é necessário selecionar a opção -*View*-, seguido da opção -*Rules*-.

Na Figura A.4 é possível observar a interface do visualizador de regras.

A interface do visualizador de regras, mostra um mapa de todo o processo de inferência difuso do sistema, baseado no editor de regras mencionado anteriormente. Os três gráficos representados no topo da figura, representam o antecedente e o consequente da primeira regra. Cada regra está representada ao longo de cada linha, e cada coluna representa uma variável. Já o número de cada regra, está exposto do lado esquerdo de cada linha.

As duas primeiras colunas apresentadas, mostram as funções de pertença relativamente ao antecedente. Ou seja, referente à parte *"if"* de cada regra. Já a terceira coluna, mostra as

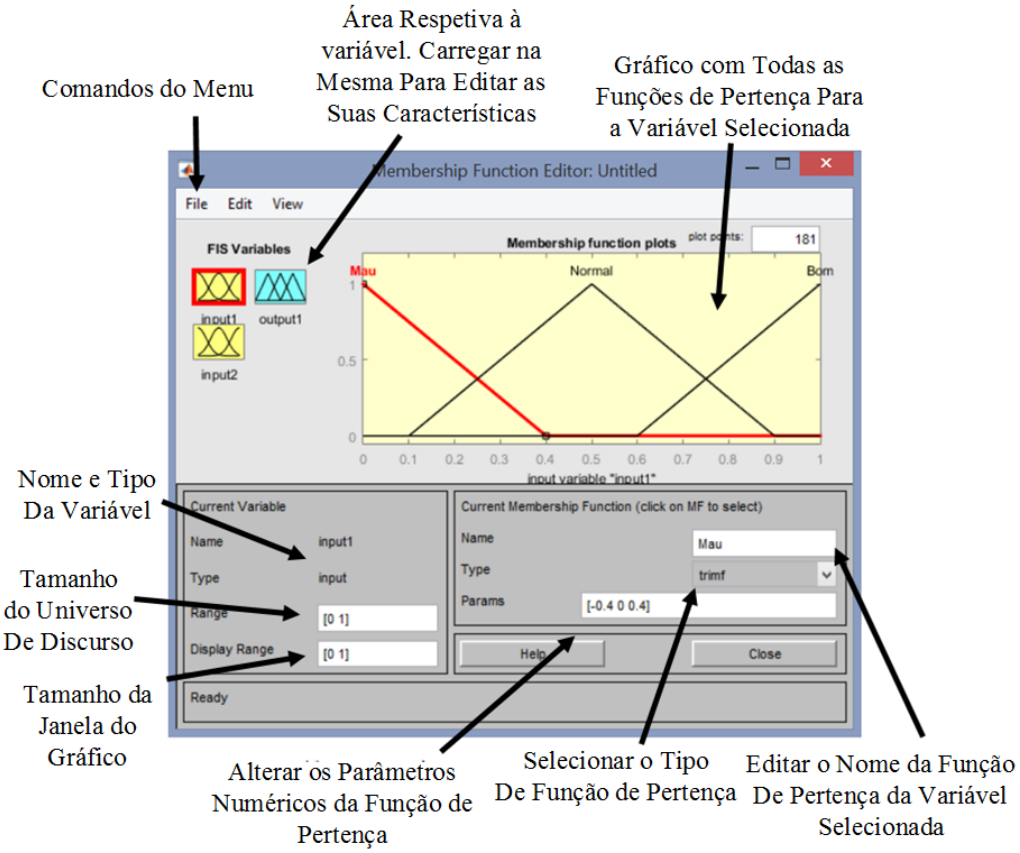


Figura A.2: Estrutura do editor das funções de pertença e respetiva descrição de cada secção.

funções de pertença relativas ao consequente. Ou seja, referente à parte *"then"* de cada regra [18].

Visualizador de Superfície

Para abrir a interface do visualizador de superfície, é necessário selecionar a opção -*View*-, seguido da opção -*Surface*-.

Na Figura A.5 é possível observar a interface do visualizador de superfície.

Ao abrir o visualizador de superfície, é possível ver uma curva tridimensional que representa o mapeamento das regras criadas anteriormente. Este visualizador possui menus de *"drop-down" X(input)*, *Y(input)* e *Z(output)*, que permitem selecionar qualquer uma das entradas e saídas para realizar o gráfico. Imediatamente abaixo, estão situados o *X grids* e o *Y grids*, que permitem especificar o número de eixos que se pretende incluir. Esta capacidade permite que se mantenha o tempo de cálculo razoável, para problemas complexos.

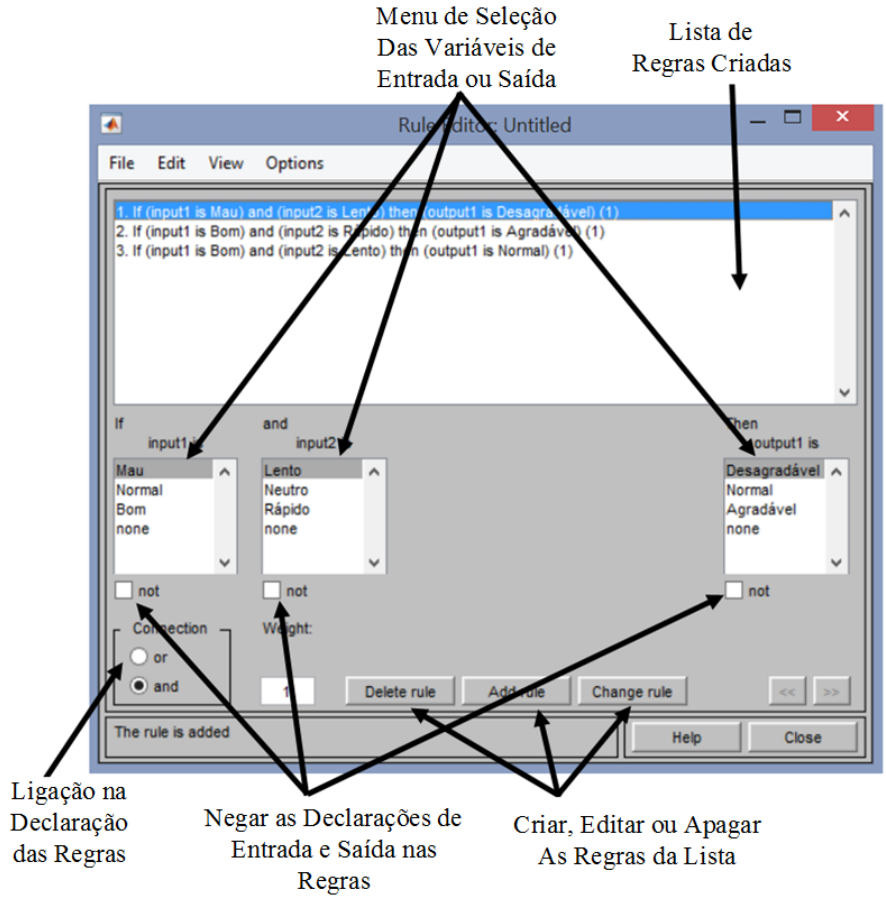


Figura A.3: Estrutura do editor de regras e respetiva descrição de cada secção.

A.1.2 Linha de Comandos

Uma outra forma de realizar o controlo de um sistema através da lógica difusa, é recorrendo à linha de comandos. O FIS é sempre moldado como uma estrutura MATLAB . Para carregar este sistema, recorre-se ao seguinte comando:

```
a = readfis('teste_fuzzy.fis')
```

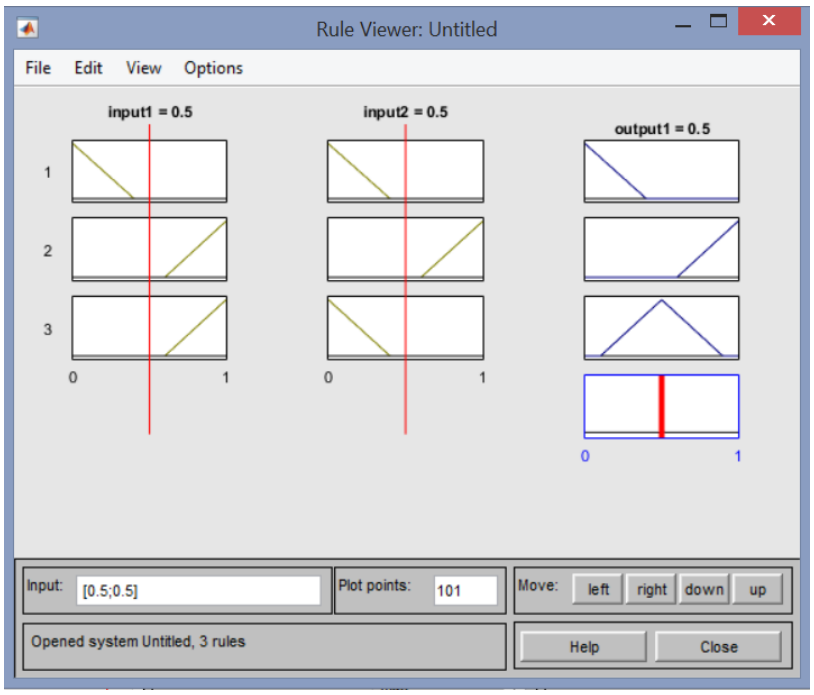


Figura A.4: Estrutura do visualizador de regras.

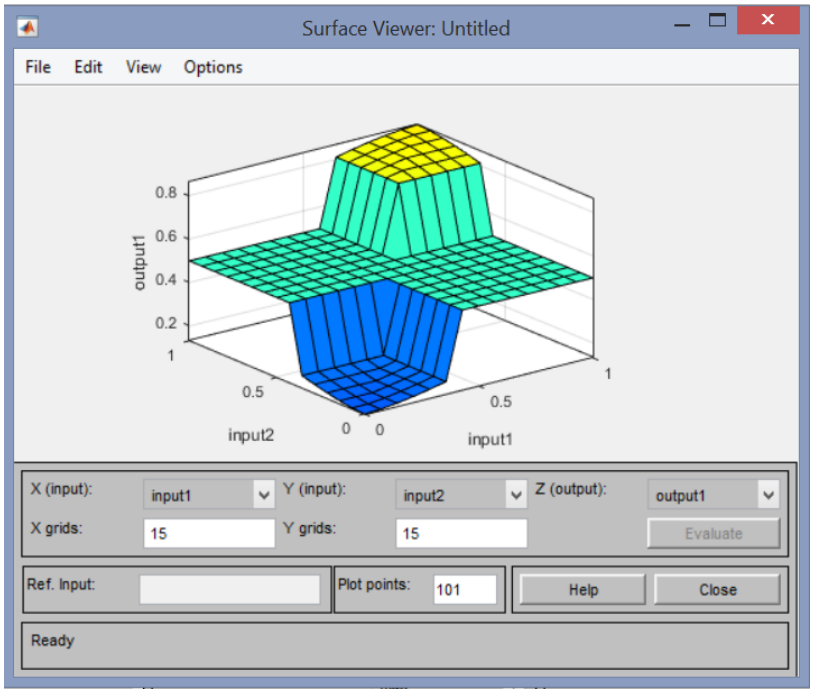


Figura A.5: Estrutura do visualizador de superfície.

Este comando, retorna o seguinte resultado:

```
a =

            name: 'teste_fuzzy'
            type: 'mamdani'
       andMethod: 'min'
        orMethod: 'max'
    defuzzMethod: 'centroid'
       impMethod: 'min'
       aggMethod: 'max'
           input: [1x2 struct]
          output: [1x1 struct]
            rule: [1x3 struct]
```

Como a variável "a"designa o sistema difuso em causa, é possível exibir qualquer uma das interfaces da *Fuzzy Toolbox*, diretamente da linha de comandos. Para isso, é necessário usar os seguintes comandos:

- *fuzzyLogicDesigner (a)* - Exibe a interface do *Fuzzy Logic*;
- *mfedit (a)* - Exibe o editor das funções de pertença;
- *ruleedit (a)* - Exibe o editor de regras;
- *ruleview (a)* - Exibe o visualizador de regras;
- *surfview (a)* - Exibe o visualizador de superfície.

Construir um sistema do zero

É possível criar um FIS, usando a linha de comandos, em vez da *Toolbox* dedicada à lógica difusa. Por exemplo, para construir o sistema inteiramente através da linha de comandos, é necessário utilizar os seguintes comandos:

- ***newwfis*** - Cria um novo FIS;
- ***addvar*** - Adiciona uma nova variável ao FIS;
- ***addmf*** - Adiciona uma nova função de pertença ao FIS;
- ***addrule*** - Adiciona uma nova regra ao FIS.

Cada variável de entrada ou de saída, correspondem a um determinado número identificador, assim como cada função de pertença. Relativamente às regras, estas podem ser construídas, tendo em conta a seguinte estrutura:

If *input1 is* MF1 or *input2 is* MF3, *then output1 is* MF2 (*weight* = 1)

Esta regra é transformada numa estrutura, de acordo com a seguinte lógica: Se houver m variáveis de entrada e n variáveis de saída, então o primeiro m vetor da estrutura de regras corresponde às entradas de *1* a m [18].

O próximo excerto de código, exemplifica como se pode construir um FIS, através da linha de comandos.

```
% Construcao do sistema difuso:
Ex = newfis('Ex','mamdani','prod','probor','prod','sum');

% Definir a entrada "Aulas":
Ex = addvar(Ex,'input','Aulas',[-20 20]);
Ex = addmf(Ex,'input',1,'Fracas','trimf',[-30 -20 0]);
Ex = addmf(Ex,'input',1,'Boas','trimf',[-20 0 20]);
Ex = addmf(Ex,'input',1,'Excelentes','trimf',[0 20 30]);

% Definir a entrada "Professor":
Ex = addvar(Ex,'input','Professor',[-20 20]);
Ex = addmf(Ex,'input',2,'Mau','trimf',[-30 -20 0]);
Ex = addmf(Ex,'input',2,'Bom','trimf',[-20 0 20]);
Ex = addmf(Ex,'input',2,'Excelente','trimf',[0 20 30]);

% Definir a saida "Aprendizagem":
Ex = addvar(Ex,'output','u',[-30 30]);
Ex = addmf(Ex,'output',1,'Mau','trimf',[-30 -30 -30]);
Ex = addmf(Ex,'output',1,'Fraca','trimf',[-20 -20 -20]);
Ex = addmf(Ex,'output',1,'Neutra','trimf',[0 0 0]);
Ex = addmf(Ex,'output',1,'Boa','trimf',[20 20 20]);
Ex = addmf(Ex,'output',1,'Excelente','trimf',[30 30 30]);

Lista_regras = [1 1 1 1 1;...   % Regra 1
                1 2 2 1 1;...   % Regra 2
                1 3 3 1 1;...   % Regra 3
                2 1 2 1 1;...   % Regra 4
                2 2 3 1 1;...   % Regra 5
                2 3 4 1 1;...   % Regra 6
                3 1 3 1 1;...   % Regra 7
                3 2 4 1 1;...   % Regra 8
                3 3 5 1 1];     % Regra 9
```

```
Ex = addrule(Ex,Lista_regras);
```

Relativamente às regras criadas, estas seguem a seguinte estrutura:

MF (Aulas) - MF (Professor) - MF (Aprendizagem) - Peso - AND (Operador)

A.1.3 Simulink

Para se construir um sistema de lógica difusa, usando o Simulink, é necessário usar o bloco *"Fuzzy Logic Controller"* (Figura A.6). Pode-se ter acesso a este através da biblioteca do Simulink, ou em alternativa, digitando *"fuzblock"* na janela de comandos do MATLAB.

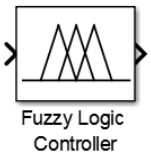


Figura A.6: Bloco *"Fuzzy Logic Controller"* disponível na biblioteca do *Simulink*.

O bloco mencionado anteriormente, usa como parâmetro um ficheiro de um sistema de inferência difuso, gravado como *".fis"*. Para isso, é necessário abrir a caixa de diálogo *"Function Block Parameters: Fuzzy Logic Controller"* e introduzir na mesma o ficheiro *".fis"* pretendido, como mostra a Figura A.7.

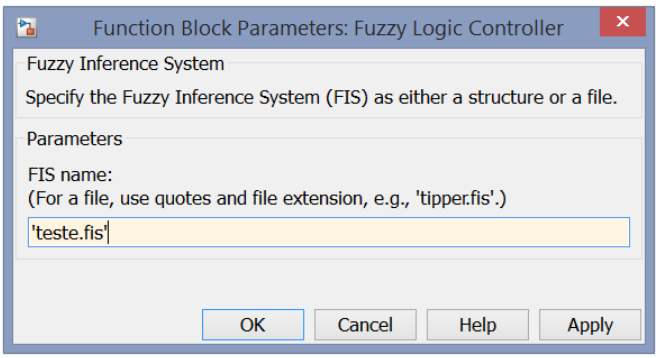


Figura A.7: Caixa de diálogo *"Function Block Parameters: Fuzzy Logic Controller"*.

Anexo B

B.1. Estrutura do Controlador PID

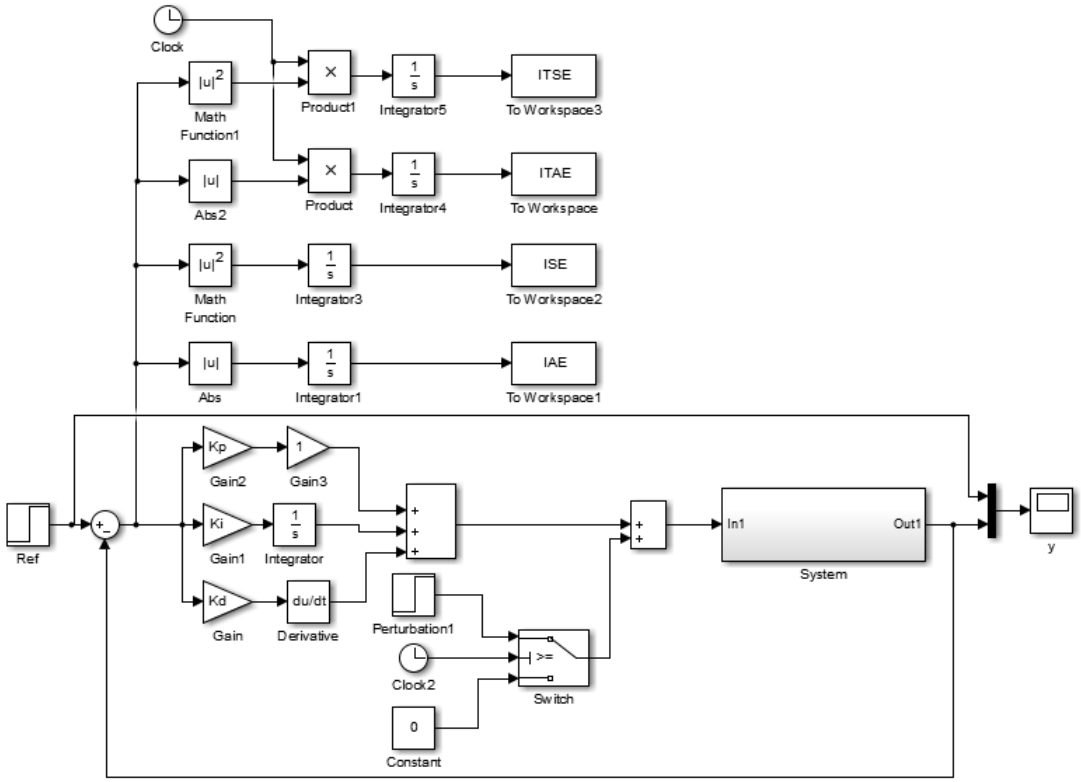


Figura B.1: Estrutura desenvolvida em Simulink para o controlador PID.

Anexo C

C.1. Estrutura do Primeiro Controlador PID-Difuso Adaptativo

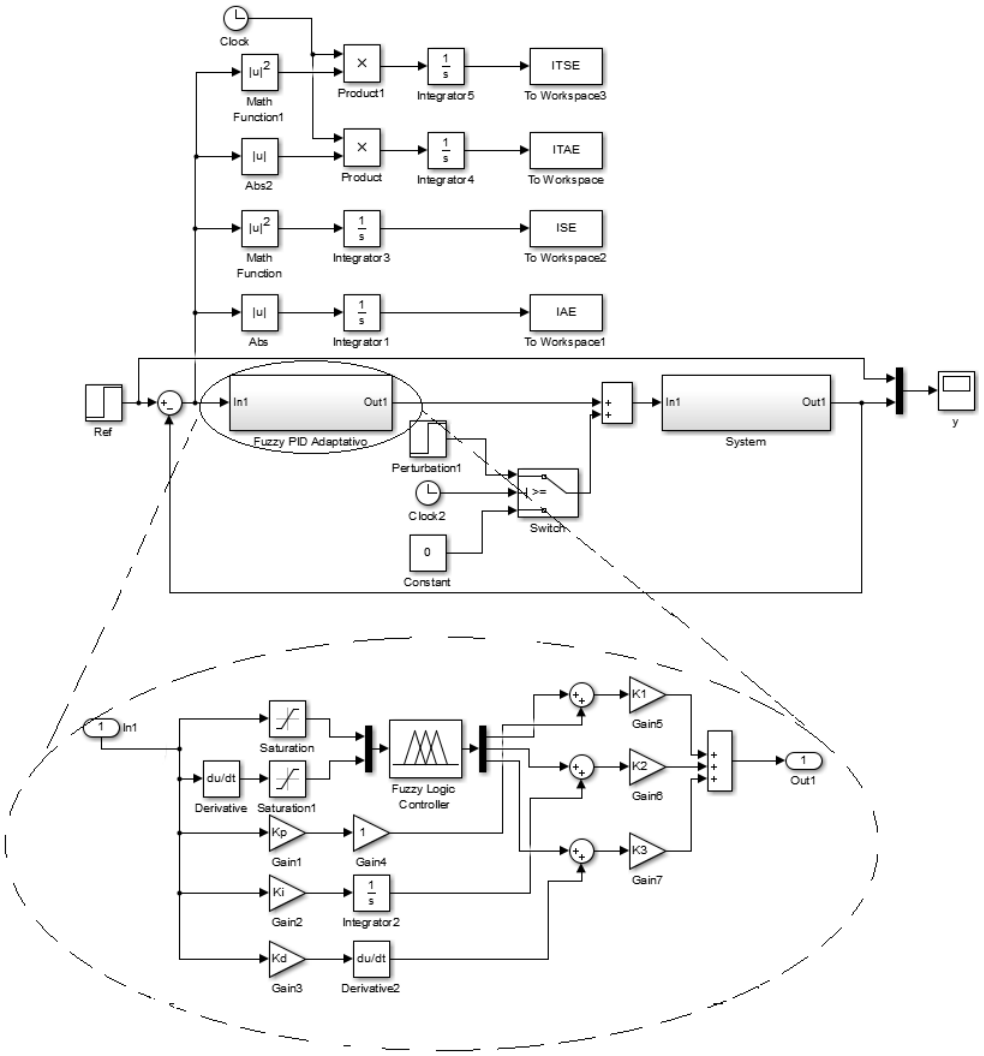


Figura C.1: Estrutura desenvolvida em Simulink para o primeiro controlador PID-Difuso Adaptativo.

Anexo D

D.1. Estrutura do Segundo Controlador PID-Difuso Adaptativo

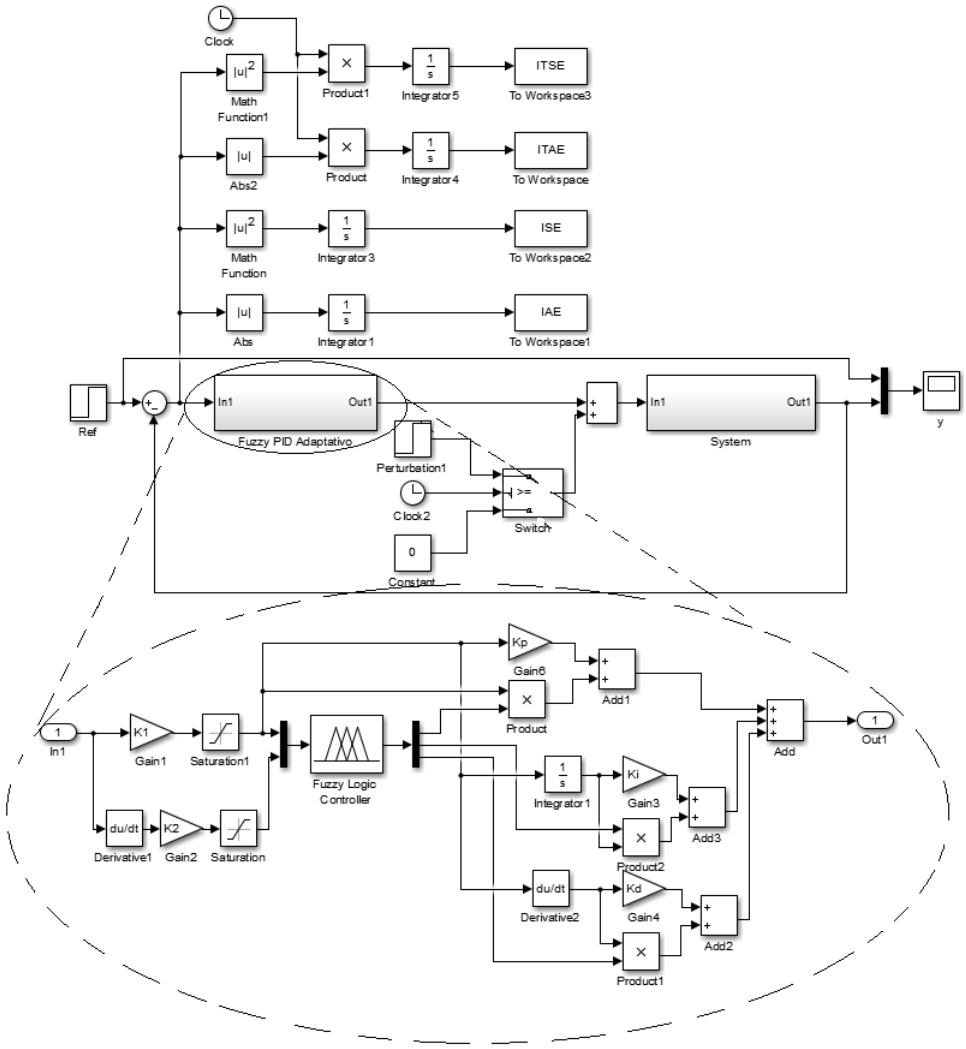


Figura D.1: Estrutura desenvolvida em Simulink para o segundo controlador PID-Difuso Adaptativo.