FCTUC

Universidade de Coimbra
Faculdade de Ciências e Tecnologia
Departamento de Engenharia Electrotécnica e de Computadores

Controlo Difuso Preditivo para Aplicação em Processos Industriais

Nuno Miguel Santos Sousa

Coimbra, 2016

Controlo Difuso Preditivo para Aplicação em Processos Industriais

Orientador: Doutor Rui Alexandre de Matos Araújo
Co-Orientador: Doutor Jérôme Amaro Pires Mendes

Júri:

Presidente: Doutor Rui Pedro Duarte Cortesão
Vogais: Doutor Rui Alexandre de Matos Araújo
Doutor Alberto Jorge Lebre Cardoso

Nuno Miguel Santos Sousa

Dissertação submetida para obtenção do grau de Mestre em Engenharia Electrotécnica e de Computadores

Departamento de Engenharia Electrotécnica e de Computadores

Faculdade de Ciências e Tecnologia
Universidade de Coimbra

Setembro 2016

Agradecimentos

À minha família, pelo seu apoio.
Aos meus amigos, pelos seus momentos de convívio e companheirismo.
À Rita pelo seu suporte e compreensão.
Ao Doutor Jérôme Mendes pela sua disponibilidade e apoio.
Ao Professor Doutor Rui Araújo por me ter concedido este desafio.

Resumo

Nesta dissertação é estudada a aplicação de controlo preditivo e de modelos baseados em redes neuronais difusas recursivas no controlo de processos não lineares. O sistema de controlo aqui estudado possui uma lei de controlo preditivo baseada no algoritmo de controlo GPC. O modelo do processo utilizado por este controlador é baseado num modelo discreto do processo a controlar, gerado em tempo real por uma rede neuronal difusa recursiva. Esta rede agrega um conjunto de regras difusas cujas consequentes contêm modelos lineares. Estes modelos linearizam o modelo do processo em zonas de operação e são agregadas formando um modelo não-linear global que é utilizado pelo controlador GPC. A rede neuronal difusa recursiva é atualizada em cada período de amostragem, sendo as antecedentes das regras difusas atualizadas pelo algoritmo da retro propagação do erro, e as consequentes atualizadas pelo algoritmo dos mínimos quadrados recursivos. Nesta dissertação é também proposto um novo controlador onde foi desenvolvida uma nova lei de adaptação das consequentes das regras difusas de forma a garantir que o erro de seguimento é limitado. Este sistema de controlo proposto, resultou numa publicação em conferência. Com o intuito de validar as técnicas investigadas, são realizados estudos em simulação com vários modelos de processos e em experiências laboratoriais com um grupo motor/gerador.

Palavras-chave: Sistemas Difusos, Sistemas neuro-difusas, Controlo Preditivo, Sistemas Não Lineares, Identificação, Otimização.

Abstract

This dissertation studies the application of predictive control, and models based in recurrent fuzzy neural networks, to the control of non-linear processes. The studied control system is based in predictive control where it is used the GPC algorithm. A discrete-time model is fed to the controller algorithm and is generated in real time by a recurrent fuzzy neural network. This network is built with a set of fuzzy rules where the consequent of each rule is a linear model. These linear models are activated by the fuzzy rules and are aggregated in order to produce a resultant non-linear model of the process to be used by the GPC. The antecedent parts of the rules in the recurrent fuzzy neural network are updated with the back-propagation algorithm, and the consequent parts are updated with the recursive least squares algorithm. In this dissertation is also proposed a controller where it was developed a new consequents adaptation algorithm with the objective of having a bounded steady state error. The proposed controller was published in a conference paper. The developed controllers are tested in simulation environment, and also in a laboratory environment in the control of a motor loaded with a variable mechanical load.

Keywords: Fuzzy Systems, Neuro-Fuzzy Systems, Predictive Control, Nonlinear Systems, Identification, Optimization.

Conteúdo

Lista de Tabelas

Lista de Figuras

Listagens

Símbolos

$a(z^{-1})$	Polinómio dos elementos de entrada do modelo do sistema
$a_j(z^{-1})$	Polinómio da regra difusa j da RFNN
A	Conjunto difuso da antecedente
A_i^j	Variável linguística da RFNN associada com o sinal i e com a regra j
$\mathbf{A}$	Matriz de transição de estado do sistema
$b(z^{-1})$	Polinómio dos elementos de saída do modelo do sistema
$b_j(z^{-1})$	Polinómio da regra difusa j da RFNN
$c(z^{-1})$	Polinómio associado ao sinal de ruído
d	Atraso de transporte entre a entrada e a saída do sistema físico
$e_p(z^{-1})$	Polinómio gerado pelo algoritmo GPC
$e(k)$	Erro de estimação da saída do sistema
$\mathbf{E}(t)$	Vetor de observações de $e(k)$
$f_p(z^{-1})$	Polinómio gerado pelo algoritmo GPC
$\mathbf{F}$	Matriz auxiliar do algoritmo GPC com os coeficientes dos polinómios $f_p(z^{-1})$
$g_p(z^{-1})$	Polinómio gerado pelo algoritmo GPC
$\mathbf{G}$	Matriz auxiliar do algoritmo GPC com os coeficientes dos polinómios $g_p(z^{-1})$
i	Índice das variáveis de entrada $i = 1, \ldots, \bar{n}$ da RFNN
j	Índice das regras difusas $j = 1, \ldots, L$
$J(\mathbf{\Theta}, t)$	Função de custo das consequentes da RFNN
$J(k)$	Função de custo do algoritmo GPC
$\mathbf{K}(k)$	Vetor de ganhos
L	Número de regras difusas
$\mathbf{L}$	Matriz auxiliar do algoritmo GPC com os coeficientes dos polinómios $g_p(z^{-1})$
m_{ij}	Centro da função de pertença gaussiana associada com A_i^j
n	Dimensão do vetor $\mathbf{x}(k)$
$\bar{n}$	Dimensão do vetor $\mathbf{x}_e(k)$ $(\bar{n} = n_y + n_u + 1)$
n_u	Ordem do polinómio $b(z^{-1})$
n_y	Ordem do polinómio $a(z^{-1})$
N_p	Horizonte de predição da saída do sistema

N_u	Horizonte da ação de controlo
$\mathbf{P}$	Matriz definida positiva
$\mathbf{Q}$	Matriz definida positiva
$r(k)$	Sinal de referência
$\mathbf{r}$	Vetor com elementos de referência $r(k)$
R_j	Regra difusa j
$\mathbf{R}(t)$	Matriz de covariâncias
$u(k)$	Sinal de comando
V	Função candidata de Lyapunov
x	Variável de entrada
$x_i(k)$	Variável de entrada contida em $\mathbf{x}(k)$
$\mathbf{x}(k)$	Vetor de variáveis de entrada da RFNN
X	Universo de discurso (domínio) de x
$y(k)$	Saída do sistema físico
$\hat{y}(k)$	Saída processo estimada pela RFNN
$\hat{y}_j(k)$	Saída do processo estimado pela regra difusa j da RFNN
$\mathbf{y}(t)$	Vetor de observações de $y(k)$
$\hat{\mathbf{y}}$	Vetor com previsões de $y(k)$ dentro do horizonte de predição N_p
Δ	Operador diferencial $1 - z^{-1}$
$\Delta\mathbf{u}$	Vetor de ações de controlo futuras $\Delta u(k)$ dentro do horizonte da ação de controlo N_u
$\Delta\mathbf{u}_1$	Vetor de ações de controlo passadas
η	Constante de aprendizagem das antecedentes da RFNN
ϑ_{ij}	Ganho de realimentação na camada 2 da RFNN da regra difusa j e variável i
$\boldsymbol{\theta}^j$	Vetor de parâmetros ajustáveis do modelo
$\boldsymbol{\Theta}$	Agregação dos vetores de parâmetros $\boldsymbol{\theta}^j$
$\hat{\boldsymbol{\Theta}}$	Estimação dos parâmetros do modelo
$\tilde{\boldsymbol{\Theta}}$	Erro de estimação do vetor de parâmetros $\boldsymbol{\Theta}$
$\boldsymbol{\Theta}^*$	Vetor de parâmetros do modelo ótimo
λ	Ganho utilizado no algoritmo GPC
$\mu_j(k)$	Valor de ativação da antecedente da regra difusa j da RFNN
$\mu_{ij}(k)$	Saída de um nó na camada 3 da RFNN da regra difusa j e variável i
$\mu_A(x)$	Valor de pertença do conjunto difuso A
$\xi(k)$	Ruído branco com média nula
σ_{ij}	Largura da função de pertença gaussiana associada com A_i^j
$\varphi(k)$	Função de custo das antecedentes RFNN
ϕ_p	Ganho aplicado ao sinal de referência $r(k)$
$\boldsymbol{\Phi}$	Matriz de ganhos ϕ_p

$\boldsymbol{\Phi}(t)$	Vetor de observações de $\boldsymbol{\Psi}$
$\boldsymbol{\psi}^j$	Vetor de funções utilizadas nos consequentes da RFNN da regra j
$\boldsymbol{\Psi}$	Agregação dos vetores $\boldsymbol{\psi}^j$
$\omega_j(k)$	Valor de ativação normalizado da regra difusa j

Acrónimos

ARMAX	Autoregressive–Moving-Average model with eXogenous inputs
FL	Fuzzy Logic
FNN	Fuzzy Neural Network
FP	Função de Pertença
GPC	Generalized Predictive Control
LD	Lógica Difusa
MBPC	Model Based Predictive Control
NARMAX	Nonlinear Autoregressive–Moving-Average model with eXogenous inputs
NN	Neural Network
PWM	Pulse Width Modulation
RFNN	Recursive Fuzzy Neural Network
RLS	Recursive Least Squares
RMPC	Robust Model Predictive Control
T-S	Sistema difuso do tipo Takagi-Sugeno

Capítulo 1

Introdução

Na industria, em particular no controlo de processos, é frequente encontrar plantas com comportamento não linear. Consoante as características de não linearidade, poderão existir várias abordagens para construir sistemas de controlo para este tipo de plantas [1]. Estas vão desde reduzir o modelo completo do sistema a uma versão linearizada em torno de uma zona de operação e utilizar teoria de controlo de sistemas lineares, a abordagens de controlo não linear. O controlo não linear de sistemas é uma vasta área em forte desenvolvimento.

Nesta dissertação são estudadas abordagens de controlo não linear baseadas em teoria de conjuntos difusos e controlo preditivo.

1.1 Identificação de Sistemas

Ao abordar um problema de controlo, um primeiro obstáculo a ultrapassar é a necessidade de conhecer o sistema que se deseja controlar, ou seja, saber qual o modelo matemático que descreve o seu comportamento. O modelo de um sistema pode ser determinado de forma analítica através de modelação físico-matemática baseada em primeiros princípios, (leis de Newton, por exemplo) ou experimentalmente através de modelos de entrada/saída quando se tem acesso à planta. Ao problema de identificar os parâmetros de um modelo de um sistema (Figura 1.1) com base em dados experimentais dá-se o nome identificação do sistema [2]. De forma simplista, a identificação de uma planta pode ser efetuada fornecendo um sinal conhecido à sua entrada $x(k)$, como por exemplo um impulso ou uma entrada em degrau, e registar o comportamento da sua saída $y(k)$. Estes dados são posteriormente utilizados para ajustar os parâmetros de um dado modelo de forma a aproximar o comportamento real do sistema dentro de determinados requisitos.

Os modelos determinados através de métodos experimentais de identificação são em muitos cenários mais atrativos que os métodos analíticos de modelação. Isto deve-se

Figura 1.1: Modelo de entrada/saída ou de caixa preta.

ao facto de que nem sempre é possível obter um modelo analítico de um processo, por ser um processo moroso e dispendioso, ou por o sistema físico ser demasiado complexo. Outra vantagem dos métodos de identificação, é a possibilidade de a identificação de um sistema poder ser realizada *online*, ou seja em tempo real com o sistema a identificar em funcionamento.

1.2 Adaptação

Na abordagem clássica para o problema de controlo, o controlador é dimensionado em função do modelo do processo. Este dimensionamento é efetuado assumindo que o modelo do processo não varia na zona de operação em que vai funcionar, no entanto podem surgir casos em que na realidade o modelo do processo sofre variações significativas durante o seu normal funcionamento. Estas variações no modelo do processo podem dever-se a alterações físicas do processo ou a perturbações externas. Este cenário coloca algumas dificuldades no dimensionamento do controlador. Uma das abordagens existentes para contornar este problema será sintonizar o controlador de tal forma que este tolere variações no modelo do processo dentro de uma gama admissível. Esta abordagem é conhecida por controlo robusto [1]. Uma desvantagem deste método é o compromisso entre as variações admissíveis no processo e a otimalidade da regulação. Uma outra abordagem com vantagens significativas, em comparação com a utilização de um controlador com características estáticas desde o momento em que foi dimensionado, é construir um controlador cujas características se ajustam em função do processo a ser controlado, isto é, que se adapte ao processo. Este é o chamado de controlo adaptativo [1]. A capacidade de adaptação é utilizada em sistemas de controlo aplicados em processos em que os seus parâmetros são desconhecidos e/ou variam com o tempo. Estas variações de parâmetros devem-se a alterações na dinâmica do sistema devido a variações físicas intrínsecas do processo ou a perturbações externas. Um exemplo recorrente é o de uma aeronave que enquanto executa um voo, sofre uma diminuição da sua massa devido ao consumo do combustível, alterando assim a sua dinâmica de voo. A Figura 1.2 ilustra esquematicamente um sistema de controlo onde está representada a malha de controlo onde o controlador

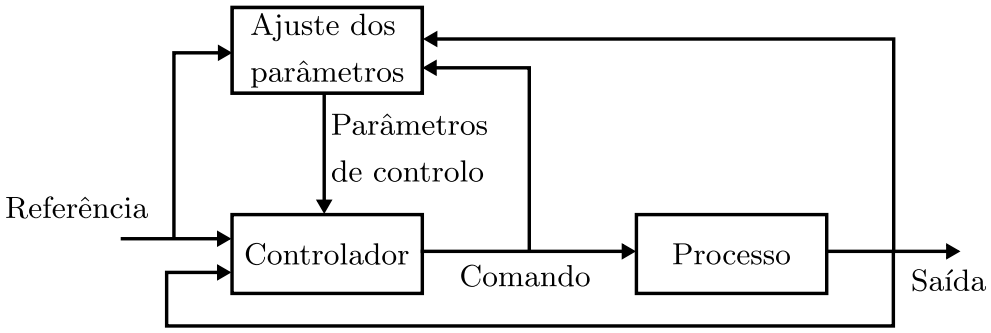


Figura 1.2: Controlo adaptativo.

gera uma ação de controlo que é fornecida ao processo. Em paralelo a lei de controlo é modificada de forma adaptativa em função da dinâmica atual do processo.

1.3 Lógica Difusa

Uma alternativa à identificação clássica de sistemas passa por utilizar modelos linguísticos (ou difusos). A lógica difusa obteve inspiração em sistemas biológicos, nomeadamente na forma como a mente humana interpreta informações incompletas ou imprecisas e toma decisões relevantes com base nestas. O termo difuso advém da sua relação com a lógica clássica, onde tipicamente a função de pertença a um conjunto apenas pode tomar o valor 0 (não pertence) ou 1 (pertence), por oposição a diferentes graus intermédios de verdade de 0 a 1. Tipicamente, a estrutura de um sistema difuso é composta por um módulo que representa a base de conhecimento. Este contém a informação sobre como a entrada se relaciona com a saída do sistema, sob a forma de regras do tipo "Se/Então". Esta informação é posteriormente utilizada para inferir qual é o conjunto de regras mais relevante que se aplica em função dos valores atuais da entrada e da saída. As regras difusas operam sobre quantidades difusas onde grandezas físicas são descritas sob a forma de adjetivos como por exemplo "baixo", "quente", etc. Assim é necessário passar do domínio real para o domínio linguístico e vice-versa de forma ao sistema interagir com o exterior. Estas funções são desempenhadas pelo difusor e pelo desdifusor respetivamente. A Figura 1.3 ilustra esta estrutura típica de um sistema difuso.

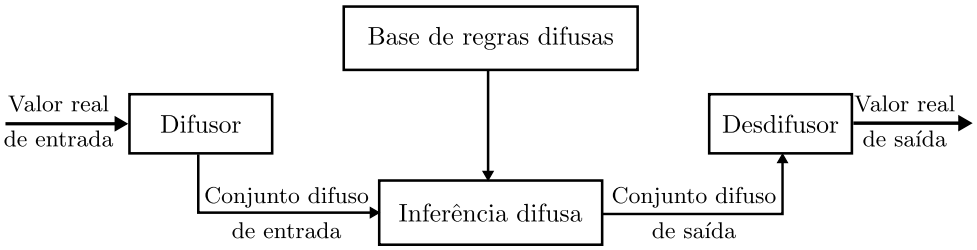


Figura 1.3: Sistema difuso, composto pelos seus blocos principais.

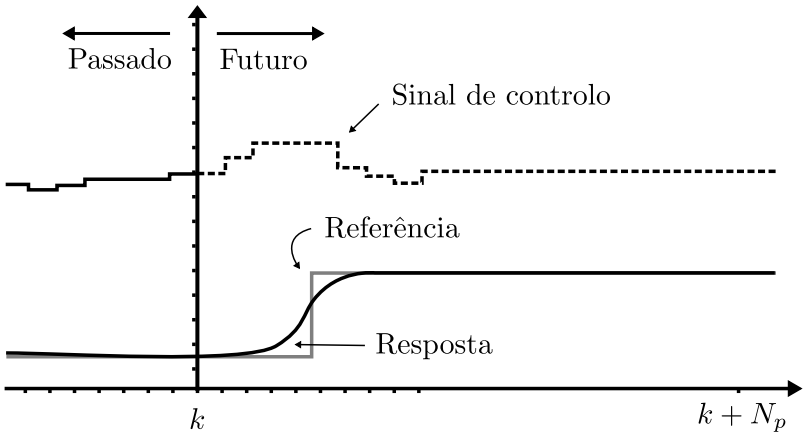


Figura 1.4: Conceito de controlo preditivo.

1.4 Controlo Preditivo

O controlo preditivo é uma técnica [3] que emprega tal como o nome indica, a previsão de algum tipo de informação para gerar a ação de controlo. Ao contrário de outras abordagens, no controlo preditivo o sinal de comando não é determinado tendo em conta o erro atual ou passado da saída da planta face a uma referência desejada, mas sim com uma previsão do erro futuro entre a referência desejada e a saída prevista da planta. O controlo preditivo tem como base o modelo da planta a controlar e a partir deste gerar um modelo de previsão do comportamento da planta para instantes futuros de tempo. Este modelo de previsão serve o propósito de prever a saída da planta, à qual é fornecida uma sequência de comando e é depois obtida uma estimativa da saída. O limite temporal do alcance desta previsão, N_p, é chamado de horizonte de predição. Uma vez obtido o preditor, o esforço de controlo futuro é determinado através de uma minimização de uma função de custo onde é ponderado o erro futuro entre a saída da planta e uma referência desejada, e o esforço de controlo futuro. A Figura 1.4 ilustra este conceito.

1.5 Motivação

O controlo preditivo generalizado (GPC) [3] tornou-se num dos mais poderosos métodos de controlo preditivo utilizados na industria. Este método tem como base a existência de um modelo do processo de forma a obter um modelo preditor deste. O GPC tem um histórico de aplicações em vários processos e com resultados positivos [4], no entanto, os processos onde este foi aplicado eram na sua maioria lineares, simplificando o problema de otimização quadrática utilizada pelo GPC. Foram propostas aplicações do GPC para processos não lineares [5], onde estes eram linearizados localmente na zona de operação.

Dada a sua natureza preditiva, uma característica presente na classe de controladores preditivos baseados no modelo do processo (MBPC, do inglês *Model Based Predictive Control*), incluindo o GPC, é pressuposto de que existe um modelo preciso do processo. Este pressuposto levanta alguns problemas, nomeadamente, a modelação baseada em primeiros princípios de sistemas físicos com dinâmicas complexas é de difícil obtenção, podem existir grandes incertezas nos parâmetros do modelo, ou os sistemas podem demonstrar comportamentos fortemente não lineares. Uma abordagem alternativa na modelação de sistemas não lineares é a utilização de lógica difusa (FL, do inglês *Fuzzy Logic*) ou de redes neuronais (NN, do inglês *Neural Networks*). Sistemas de FL podem ser utilizados para aproximar o modelo da dinâmica de processos não lineares, dado que foi demonstrado que estes são aproximadores universais [6]. Uma variante dos sistemas de FL, os sistemas do tipo Takagi-Sugeno (T-S) [7], tem ganho popularidade graças às consequentes das suas regras difusas consistirem em funções reais, em lugar de conjuntos difusos. Por outro lado, as NN são também aproximadores universais [8], e têm também sido utilizadas na identificação e controlo de sistemas não lineares [9]. As vantagens individuais de FL e de NN podem ser combinadas, originando as redes neuronais difusas (FNN do inglês *Fuzzy Neural Network*) que têm também mostrado resultados positivos na identificação de sistemas. Uma desvantagem deste tipo de sistemas é o facto de não possuírem realimentação interna, o que restringe as classes de processos em que poderão ser aplicados [10]. Uma solução para este problema passa pela utilização de redes neuronais difusas recursivas (RFNN, do inglês *Recursive Fuzzy Neural Network*) que têm ganho popularidade. Estes sistemas possuem realimentações internas que funcionam como elementos de memória, permitindo assim armazenar informação dinâmica e lidar com problemas temporais de forma mais eficiente.

Na identificação de sistemas, o modelo do processo é obtido tendo como base leituras de dados de entrada e de saída do sistema físico obtidas de forma *offline*. Esta abordagem de modo *offline* tem alguns problemas dado que os dados podem ser insuficientes e o modelo resultante pode não ser uma aproximação suficientemente boa do modelo do processo. Esta é uma motivação para o estudo de sistemas de identificação e adaptação *online*

como forma de resolver este problema [2].

Neste contexto, nesta dissertação será estudada a utilização de redes neuronais difusas recursivas (RFNN) para a aprendizagem do modelo do sistema a controlar, que será incorporado no controlador preditivo generalizado (GPC). Duas abordagens de adaptação da RFNN diferentes serão usadas, sendo uma dessas abordagens proposta nesta dissertação.

1.6 Objetivos

Assumindo os problemas e os conceitos considerados anteriormente, esta dissertação tem os seguintes objetivos. Em primeiro lugar, estudar o controlo preditivo, em particular o algoritmo de controlo GPC. Este algoritmo deve ser estudado de forma a se obter um entendimento da sua estrutura e da sua integração num sistema de controlo. Em segundo lugar deverão ser estudados modelos baseados em redes neuronais difusas recursivas (RFNN) para a identificação de modelos de sistemas dinâmicos em tempo real, baseando-se apenas nos seus dados de entrada e de saída. Deverão ser estudados os métodos de ajuste dos seus parâmetros internos que permitam ao modelo identificar um sistema físico com comportamento não linear. Com o objetivo de evoluir o estado da arte é também proposto um sistema de controlo com uma nova lei de adaptação dos parâmetros das consequentes das regras difusas que constituem a rede neuronal difusa recursiva. Por fim, os sistemas de controlo a estudar deverão ser testados em ambiente de simulação e também no controlo de um sistema físico, de forma a avaliar as capacidades do sistema de controlo proposto.

1.7 Organização da Dissertação

O texto encontra-se organizado da seguinte forma. No capítulo 2 são apresentadas de forma sintetizada noções de lógica difusa, de forma a contextualizar alguns conceitos chave utilizados nas secções seguintes. O capítulo 3 descreve a implementação de um controlador preditivo em que o modelo do processo é identificado através de uma rede neuronal difusa recursiva. Os mecanismos de identificação do processo demonstrados neste capítulo são também utilizados numa abordagem diferente, desenvolvida no capítulo 4. Neste capítulo é desenvolvido um novo controlador preditivo, que se propõe ter melhores capacidades de controlo. Os controladores estudados nos capítulos 3 e 4 são testados em ambiente de simulação e também numa planta em laboratório para aferir a sua viabilidade e a sua capacidade real de controlo e identificação de um processo. Estes resultados estão detalhados no capítulo 5.

No processo de desenvolvimento dos controladores estudados foram encontrados alguns

pontos que poderiam ser alvo de melhoria (trabalho futuro) e que não foram contemplados nesta dissertação. O capítulo 6 contém esta informação, e também a conclusão deste trabalho.

Capítulo 2

Lógica Difusa

A Lógica difusa (LD), proposta originalmente por Zadeh [11], inspira-se na ideia de que o conhecimento humano possa ser estruturado sob a forma de proposições lógicas e como estas conseguem representar de forma relevante os fenómenos físicos presentes na Natureza. Como área na engenharia, esta teoria tem assumido um papel cada vez mais importante. As suas aplicações incluem sistemas de controlo, onde conjuntos de proposições lógicas, também chamadas de regras difusas, descrevem a lei de controlo. Também na construção de modelos de sistemas físicos, em especial sistemas que apresentam comportamento não linear, o mesmo tipo de regras pode ser utilizado onde estas descrevem o relacionamento entre as entradas e saídas do sistema.

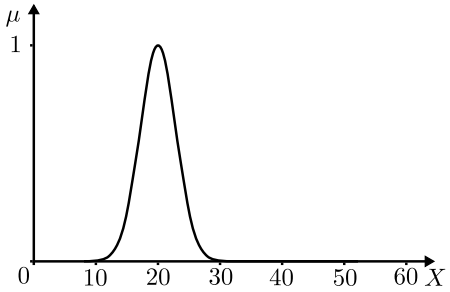
Neste capítulo são abordados, em primeiro lugar, alguns conceitos básicos de Lógica Difusa, nomeadamente conjuntos difusos, inferência difusa, difusão e desdifusão. De seguida é também abordado um tipo especial de sistema difuso desenvolvido por *Takagi* e *Sugeno* com especial interesse para os capítulos seguintes. Este capítulo serve como apresentação de conceitos de lógica difusa antes da sua aplicação na modelação de sistemas dinâmicos que é um dos componentes principais dos controladores implementados nos Capítulos 3 e 4.

Para uma visão mais abrangente da lógica difusa e a sua aplicação em controlo, é recomendada a consulta de um texto de referência como por exemplo [12].

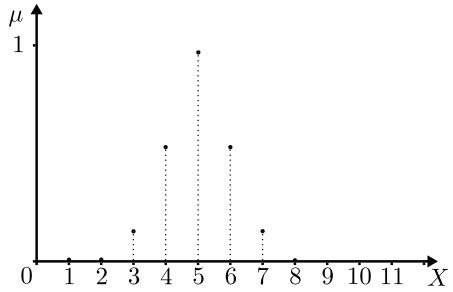
2.1 Conjuntos Difusos

A lógica difusa pode ser vista como uma generalização da lógica clássica. Um exemplo de um conjunto definido em lógica clássica, conjunto clássico, pode ser descrito na forma

$$A = \{x \in X | x > 50\}, \tag{2.1}$$



(a) Continua: Cerca de 20žC (temperatura amena).



(b) Discreta: Número de cadeiras por semestre.

Figura 2.1: Funções de pertença.

em que X representa o universo de x, ou seja, contém todos os valores que x pode tomar, $\mathbb{R}$ por exemplo, e A é um conjunto bem definido com todos os valores de x superiores a 50 contidos no universo de discurso. A lógica difusa estende este conceito tornando a definição de conjunto mais flexível através da definição de graus de pertença dos elementos do universo de discurso, onde os graus de pertença podem tomar valores entre 0 (não pertence) e 1 (pertence). Por exemplo, se definirmos o conjunto "temperatura amena" para caracterizar uma temperatura agradável, este conjunto pode, por exemplo, ser descrito da forma

$$B = \{(x, \mu_B(x))|x \in X\}, \tag{2.2}$$

onde $\mu_B(x)$ representa uma função de pertença (FP). O papel da função de pertença é atribuir a cada elemento x do universo de discurso um grau de pertença $\mu_B(x)$. A Figura 2.1a é um exemplo de uma função de pertença em que o universo de discurso é continuo, por outro lado, na Figura 2.1b o universo de discurso contém apenas valores discretos.
Tal como na lógica clássica, na lógica difusa também existem operações sobre conjuntos difusos, como por exemplo a negação, união e intersecção. Os pontos seguintes descrevem matematicamente as operações lógicas sobre conjuntos difusos.

- **Negação**: A negação, ou complemento, de um conjunto difuso A, denotado por $\overline{A}$ (não A) é definida por:

$$\mu_{\overline{A}}(x) = 1 - \mu_A(x); \tag{2.3}$$

- **União**: A união de dois conjuntos difusos, A e B é o conjunto C, ou notacionalmente $C = A \cup B$ (A *ou* B) e é dada pela seguinte expressão:

$$\mu_{A \cup B}(x) = \max[\mu_A(x), \mu_B(x)]. \tag{2.4}$$

  Esta operação pode ser descrita como o menor conjunto que contém os conjuntos A

e B;

- **Intersecção**: A intersecção de dois conjuntos A e B é o conjunto C, com a notação convencional $C = A \cap B$ (A *e* B). É determinada da forma:

$$\mu_{A\cap B}(x) = \min[\mu_A(x), \mu_B(x)]. \tag{2.5}$$

  A intersecção é interpretada como o maior conjunto que é contido por A e por B em simultâneo.

A Figura 2.2 ilustra a aplicação destes operadores. As equações acima descritas que representam os respetivos operadores lógicos não são únicas, isto é, os operadores lógicos referidos podem ser representados por outras equações que satisfaçam um determinado conjunto de requisitos [12]. Em particular, um operador de negação realiza o mapeamento $c : [0, 1] \to [0, 1]$ tal que um conjunto difuso A é transformado no seu complemento $\overline{A}$, ou seja:

$$c[\mu_A(x)] = \mu_{\overline{A}}(x).$$

Considere-se que a operação de negação deve obedecer aos seguintes requisitos:

- O complemento de um elemento com grau de pertença 1, tem como complemento grau de pertença 0 e vice-versa, ou seja, $c(1) = 0$ e $c(0) = 1$;
- A operação de negação deve ter a seguinte propriedade: se $\mu_A(x) < \mu_B(x)$ então $\mu_{\overline{A}}(x) \geq \mu_{\overline{B}}(x)$.

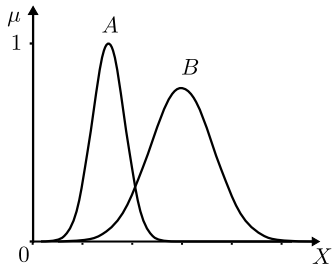
O operador de negação (2.3) satisfaz os requisitos anteriores. Da mesma forma é possível definir outros operadores distintos que também satisfazem estes requisitos. Outro exemplo de um operador de negação válido é dado por

$$\mu_{\overline{A}}(x) = c_\lambda(\mu_A(x)) = \frac{1 - \mu_A(x)}{1 + \lambda\mu_A(x)}, \tag{2.6}$$

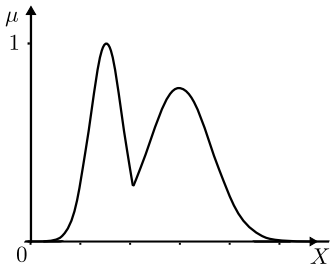
onde $\lambda \in [-1, +\infty[$ é um parâmetro de define o operador de complemento. Para cada valor de λ obtém-se um operador distinto. As equações (2.7) e (2.8) são também alternativas frequentemente utilizadas aos operadores de intersecção e de união respetivamente. Para outras alternativas pode ser consultada a referência [12].

$$\mu_{A\cap B}(x) = \mu_A(x)\mu_B(x). \tag{2.7}$$

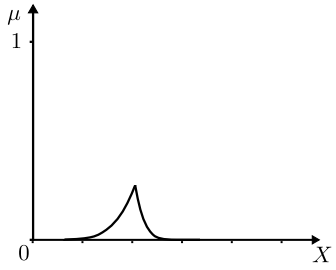
$$\mu_{A\cup B}(x) = \mu_A(x) + \mu_B(x) - \mu_A(x)\mu_B(x). \tag{2.8}$$



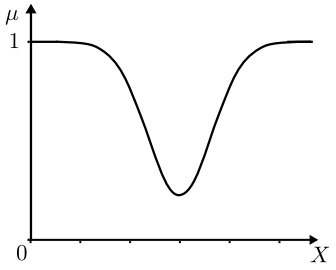
(a) Conjuntos difusos A e B.



(b) União (equação (2.4)): A ou B.



(c) Intersecção (equação (2.5)): A e B.



(d) Negação (equação (2.3)): não B.

Figura 2.2: Ilustração de exemplos de operações sobre conjuntos difusos.

Outra operação que é importante mencionar é a relação difusa de conjuntos difusos. Da lógica clássica, se tomarmos dois (ou mais) conjuntos A e B, o seu produto cartesiano denotado por $A \times B$ é o conjunto de todos os pares ordenados (a, b) onde $a \in A$ e $b \in B$. Uma relação sobre $A \times B$ é tal que para cada elemento (par ordenado) é atribuído o valor de 1 ou 0 caso este satisfaça ou não o critério que define a relação. Por exemplo, se definirmos os conjuntos $A = \{0, 1, 2\}$ e $B = \{0, 1, 2\}$, então o seu produto cartesiano é $A \times B = \{(0,0), (0,1), (0,2), (1,0), (1,1), (1,2), (2,0), (2,1), (2,2)\}$. Podemos definir por exemplo uma relação R sobre o produto cartesiano $A \times B$ tal que "a é igual a b". Assim obtemos o seguinte conjunto $R(A, B) = \{(0,0), (1,1), (2,2)\}$ onde se pode observar que $R(A, B)$ é um subconjunto de $A \times B$ e em cada um dos seus pares ordenados, o primeiro elemento é igual ao segundo.

Uma relação difusa é uma extensão do conceito relação clássica na qual é atribuído um valor de pertença entre 0 e 1 para cada elemento $(a_1, a_2, \ldots, a_n)$ do produto cartesiano $A_1 \times A_2 \times \ldots \times A_n$, ou escrito de forma matemática $\mu_R : A_1 \times A_2 \times \ldots \times A_n \to [0, 1]$. Como exemplo de uma relação difusa podemos definir a relação "a é aproximadamente igual a b" definida sobre o produto cartesiano $A \times B$. Esta relação difusa pode ser representada, por exemplo, pela função de pertença (2.9):

$$\mu_R(a, b) = e^{-|a-b|}. \tag{2.9}$$

2.2 Regras Difusas e Inferência Aproximada

2.2.1 Regras Difusas

Conforme referido anteriormente, o conhecimento humano pode ser representado na forma de expressões condicionais, também chamadas de regras difusas ou implicações difusas, com a forma

$$\text{SE } x \text{ é } A, \text{ ENTÃO } y \text{ é } B. \tag{2.10}$$

A e B são conjuntos difusos que representam um valor linguístico nos universos de discurso X e Y respetivamente. A "x é A" é dado o nome de antecedente da regra difusa e a "y é B" é dado o nome de consequente da regra. A antecedente de uma regra pode tomar mais do que apenas uma proposição tal como em "x_1 é A_1 e x_2 é A_2", onde ambos x_1 e x_2 são variáveis linguísticas [12] e X_1 e X_2 são os respetivos universos de discurso. Assim, uma regra difusa estabelece uma relação entre uma ou mais variáveis de entrada e a variável de saída. Desta forma pode-se descrever uma regra como uma relação difusa R no espaço cartesiano de $X \times Y$, ou seja, para cada par $(x, y) \in X \times Y$ está associado um grau de pertença descrito pela função de pertença $\mu_R(x, y)$.

A interpretação matemática de uma regra difusa pode ser vista como uma generalização da implicação na lógica clássica (ver Tabela 2.1). Enquanto na lógica clássica o resultado de uma regra SE-ENTÃO apenas pode tomar os valores lógicos *verdadeiro* (1) ou *falso* (0), na lógica difusa esse resultadopode passar a tomar valores intermédios de verdade (entre 0 e 1).

Tabela 2.1: Tabela de verdade de uma implicação lógica clássica.

a	b	$a \to b$
V	V	V
V	F	F
F	V	V
F	F	V

Esta generalização, no entanto pode ter duas interpretações. Ao contrário da implicação lógica clássica, que é global, ou seja $A \to B$ é válida para todos os valores lógicos de A e de B, uma implicação difusa pode ser interpretada como tendo validade global ou como tendo validade local. Ao interpretar como tendo validade local considera-se que uma regra difusa apenas é válida quando as proposições A e B tiverem ambas um elevado grau de verdade. As duas interpretações podem ser descritas matematicamente na seguinte forma:

- Validade global: $A \to B = \overline{A} \cup B$

- Validade local: $A \rightarrow B = A \cap B$

A primeira forma contempla todos os casos expressos na Tabela 2.1. A última forma é a adotada neste texto.

Se assumirmos que uma regra difusa tem validade local ($A \rightarrow B = A \cap B$), e se utilizarmos o operador de produto (2.7) para implementar a intersecção de conjuntos, então relação difusa que representa a regra difusa (2.10) tem a seguinte função de pertença:

$$\mu_{R_P}(x, y) = \mu_A(x)\mu_B(y), \tag{2.11}$$

ou utilizando o operador min (2.5):

$$\mu_{R_M}(x, y) = \min[\mu_A(x), \mu_B(y)]. \tag{2.12}$$

As implicações (2.11) e (2.12) são conhecidas por implicações de Mamdani [12]. Podemos generalizar o caso simples da regra difusa (2.10) e contemplar antecedentes compostos, ou seja, a antecedente é composta por n variáveis difusas na forma:

$$\text{SE } x_1 \text{ é } A_1 \text{ e } x_2 \text{ é } A_2 \text{ e } \ldots x_n \text{ é } A_n, \text{ ENTÃO } y \text{ é } B. \tag{2.13}$$

Notemos que, por simplificação de notação, neste capítulo a variável n tem um significado não coincidente com o significado que a variável n tem nas outras partes da dissertação fora deste capítulo. Esta antecedente em (2.13) é descrita por uma relação difusa em $X_1 \times X_2 \times \ldots \times X_n$ tal que $\mu_A(\mathbf{x}) = \mu_{A_1}(x) \wedge \mu_{A_2}(x) \wedge \ldots \wedge \mu_{A_n}(x)$, onde $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ representa o vetor de variáveis linguísticas (difusas) e onde $\wedge$ representa um operador lógico de interseção (min (2.5) ou produto (2.7), por exemplo). Assim podemos reescrever (2.11) e (2.12) da seguinte forma, respetivamente:

$$\mu_{R_P}(\mathbf{x}, y) = \left(\prod_{i=1}^{n} \mu_{A_i}(x_i) \right) \mu_B(y), \tag{2.14}$$

ou

$$\mu_{R_M}(\mathbf{x}, y) = min[\mu_{A_1}(x_1), \ldots, \mu_{A_n}(x_n), \mu_B(y)]. \tag{2.15}$$

2.2.2 Inferência Aproximada

Uma vez geradas as regras que descrevem um sistema é necessário através destas conseguir gerar conclusões com base em dados de entrada. A esse processo é dado o nome de inferência difusa. Na lógica clássica, o *modus ponens* é uma regra de inferência que permite inferir o valor de verdade da proposição B a partir do valor de verdade de A e da implicação (ou regra) $A \rightarrow B$. Por exemplo, consideremos que A é definido por "a

Tabela 2.2: Inferência lógica.

premissa 1 (dado):	x é A
premissa 2 (regra):	SE x é A ENTÃO y é B
consequência (conclusão):	y é B

Tabela 2.3: Inferência lógica aproximada.

premissa 1 (dado):	x é A'
premissa 2 (regra):	SE x é A ENTÃO y é B
consequência (conclusão):	y é B'

temperatura é *elevada*" e que B é definido por "a *pressão* é *elevada*". Se definirmos a implicação lógica "Se A Então B", e se a proposição A for verdadeira, podemos inferir que a proposição B é verdadeira, o que pode ser escrito de forma mais formal como representado na Tabela 2.2. No entanto, é frequente as regras serem utilizadas de forma aproximada. Tomando o exemplo anterior, podemos assumir que se A' for definido por "a *temperatura* é *média*" e se B' for definido por "a *pressão* é *média*", então também pode ser considerado verdadeiro que se A' é verdadeiro logo B' também o é. Os conjuntos A e A', e B e B' pertencem aos mesmos universos de discurso, respetivamente. Esta é a forma generalizada do *modus ponens*, e a este processo é dado o nome de inferência difusa aproximada (ver Tabela 2.3). Este tipo de inferência deve ser tal que quanto mais próximo A' estiver de A, mais próximo estará B' de B.

É necessário ainda determinar a função de pertença da conclusão gerada em cada regra. Esta é obtida através da regra composicional de inferência.

2.2.3 Princípio da Regra Composicional

Para entender o principio da regra composicional de inferência pode-se generalizar a noção de função, onde $y = f(x)$ dita a relação entre x e y.

Quando é fornecido um valor $x = a$ a $y = f(x)$, é possível inferir um valor $y = f(a) = b$ (Figura 2.3a). Generalizando este processo, onde a passa a representar um intervalo e $f(x)$ passa a representar uma função que relaciona intervalos, podemos também inferir um intervalo $b = f(a)$. Se realizarmos uma extensão cilíndrica do intervalo a, do seu domínio X para o domínio $X \times Y$, e realizarmos a sua intersecção I com a curva-intervalo f, b é determinado como sendo a projeção de I sobre o domínio Y (ver Figura 2.3b).

Defina-se A' como sendo um conjunto difuso no universo de discurso X, e F como sendo uma relação difusa no espaço $X \times Y$. Utilizando o mesmo raciocínio anterior, realizamos uma extensão cilíndrica de A' para $X \times Y$, obtendo-se $\mu_{A'_E}(x, y) = \mu_{A'}(x)$. A intersecção entre a extensão cilíndrica de A' e a relação F é descrita por $\mu_{A'_E \cap F}(x, y)$.

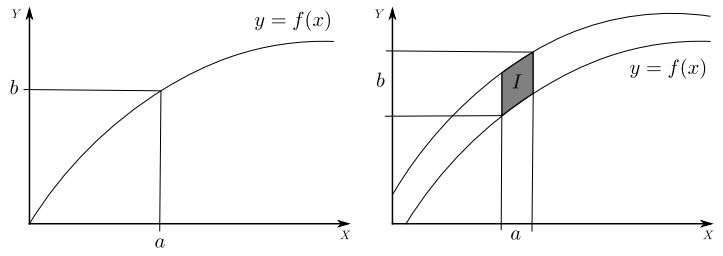


Figura 2.3: Obtenção de $y = b$ a partir de $x = a$ quando, (esquerda) a, b são pontos e $y = f(x)$ é uma curva, e quando (direita) a, b são intervalos e $y = f(x)$ é uma função que relaciona intervalos.

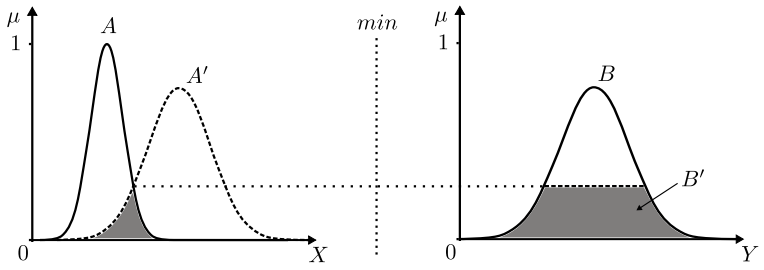


Figura 2.4: Inferência difusa com uma regra simples (2.17).

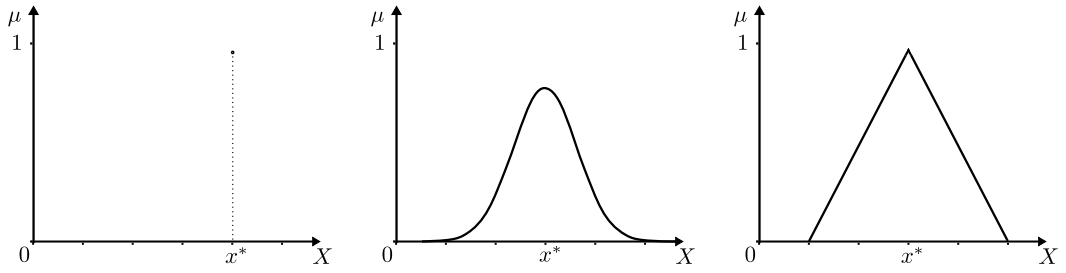
Assim, a conclusão representada pelo conjunto difuso B' é definida sendo a projeção de $\mu_{A'_E \cap F}(x, y)$ em Y dada por $\mu_{B'}(y) = \max_{x \in X}[\mu_{A'_E \cap F}(x, y)]$.

Concretizando o que foi exposto até aqui, é possível gerar as equações para determinar a conclusão das regras difusas. Dada a função de pertença $\mu_{A'}(x)$ que representa a premissa 1 (x é A'), e a função de pertença $\mu_R(x, y)$ da relação difusa que representa a premissa 2 (SE x é A ENTÃO y é B), então a função de pertença $\mu_{B'}(y)$ que descreve a conclusão (y é B') é dada pela expressão (2.16) se for utilizado o operador produto (2.7) na intersecção, ou pela expressão (2.17) se for utilizado o operador min (2.5). Em ambas as configurações é admitido que as regras tem validade local.

$$\mu_{B'}(y) = \max_{x \in X}[\mu_{A'}(x)\mu_R(x, y)] = \max_{x \in X}[\mu_{A'}(x)\mu_A(x)\mu_B(y)]. \tag{2.16}$$

$$\mu_{B'}(y) = \max_{x \in X}[\min[\mu_{A'}(x), \mu_R(x, y)]] = \max_{x \in X}[\min[\mu_{A'}(x), \mu_A(x), \mu_B(y)]]. \tag{2.17}$$

A Figura (2.4) demonstra um exemplo de inferência difusa com uma regra constituída por uma antecedente e uma consequente onde é aplicada a regra composicional (2.17).



(a) Singleto. (b) Gaussiano. (c) Triangular.

Figura 2.5: Difusão de x^* em três difusores distintos.

2.3 Sistemas Difusos

Cada regra difusa descreve apenas um fragmento de informação, no entanto um sistema difuso é mais útil se utilizar mais do que uma regra difusa de forma a agregar todo o conhecimento relevante. Um sistema difuso é constituído por quatro componentes principais, por ordem de fluxo de informação: o difusor, motor de inferência difusa e base de regras difusas, e por último o desdifusor. Estes componentes são descritos nos pontos seguintes.

2.3.1 Difusor

O propósito de um difusor é o de atribuir uma representação sob a forma de um conjunto difuso a uma grandeza numérica (e.g., uma grandeza medida), como por exemplo atribuir à temperatura ambiente 20ºC o valor linguístico representado por um conjunto difuso gaussiano com centro em 20ºC. O difusor é descrito por uma função que faz o mapeamento de um valor $x^* \in X \subset \mathbb{R}$ (caso unidimensional) para um conjunto difuso $A' \in X$, ou seja, atribui a cada valor x^* de uma grandeza real com domínio $X \subset \mathbb{R}$, um conjunto difuso A' no universo de discurso X. De forma mais geral os difusores são empregues como módulo de entrada de um sistema difuso em que este pode ter como entrada um vetor de sinais $\mathbf{x}^* = (x_1^*, x_2^*, \ldots, x_n^*)$ com dimensão n (em $\mathbb{R}^n$).

Os pontos seguintes descrevem alguns difusores.

- **Difusor singleto**: Faz o mapeamento de um valor $\mathbf{x}^* \in X$ para um conjunto difuso $A' \in X$ tal que $\mu_{A'}(\mathbf{x})$ é 1 para $\mathbf{x} = \mathbf{x}^*$ e 0 caso contrário:

$$\mu_{A'}(\mathbf{x}) = \begin{cases} 1, & \text{se} \quad \mathbf{x} = \mathbf{x}^*, \\ 0, & \text{se} \quad \mathbf{x} \neq \mathbf{x}^*. \end{cases} \tag{2.18}$$

- **Difusor gaussiano**: Define uma curva gaussiana com centro em $\mathbf{x}^*$ da seguinte

forma, onde a_i toma um valor positivo e determina a largura da curva gaussiana:

$$\mu_{A'}(\mathbf{x}) = e^{-\left(\frac{x_1 - x_1^*}{a_1}\right)^2} \wedge \ldots \wedge e^{-\left(\frac{x_n - x_n^*}{a_n}\right)^2}, \tag{2.19}$$

onde $\wedge$ representa um operador de intersecção difusa.

- **Difusor triangular**: Define uma função triangular com centro em $\mathbf{x}^*$ e onde o parâmetro b_i toma um valor positivo e determina a largura da função de pertença:

$$\mu_{A'}(\mathbf{x}) = \begin{cases} \left(1 - \frac{|x_1 - x_1^*|}{b_1}\right) \wedge \ldots \wedge \left(1 - \frac{|x_n - x_n^*|}{b_n}\right), & \text{se } |x_i - x_i^*| \leq b_i, i = 1, 2, \ldots, n, \\ 0, & \text{outros casos.} \end{cases} \tag{2.20}$$

Uma propriedade importante destes difusores é que estes tem o valor máximo em $\mathbf{x}^*$ e que $\mu_{A'}(\mathbf{x}^*) = 1$. A Figura 2.5 ilustra exemplos de cada destes três tipos de difusores.

2.3.2 Motor de Inferência Difusa

Na Secção 2.2 foi mostrado como com base em uma premissa e uma regra difusa se pode inferir uma conclusão. No entanto, um sistema difuso com uma regra apenas é de limitada utilidade. Um sistema difuso torna-se mais útil se tiver a capacidade de agregar o conhecimento sob a forma de um conjunto de regras difusas e inferir conclusões com base nestas e nas premissas. Existem algumas abordagens ao problema de agregar várias regras difusas de forma obter conclusões lógicas [12], no entanto por simplicidade aqui é apenas descrita uma dessas abordagens.

Considere-se que a base de regras de um sistema difuso é constituída de forma genérica por L regras (R_j, $j = 1, \ldots, L$). Estas regras são da forma (2.13), reescrita em (2.21) por conveniência.

$$R_j\text{: SE } x_1 \text{ é } A_1^j \text{ e } x_2 \text{ é } A_2^j \text{ e } \ldots x_n \text{ é } A_n^j\text{, ENTÃO } y \text{ é } B^j. \tag{2.21}$$

Estas regras podem ser combinadas através da união das conclusões $\mu_{B'_j}(y)$ obtidas em cada regra individual j, ou seja, $\mu_{B'}(y) = \mu_{B'_1}(y) \vee \mu_{B'_2}(y) \vee \ldots \vee \mu_{B'_L}(y)$ onde $\vee$ é um operador de união difusa (max (2.4) por exemplo) e L representa o número total de regras difusas. Utilizando o modelo de inferência (2.16) para cada regra e a implicação (2.14), onde a antecedente de cada regra tem n variáveis difusas e o operador max é usado para a união de conjuntos, obtemos o seguinte motor de inferência difusa para uma base com

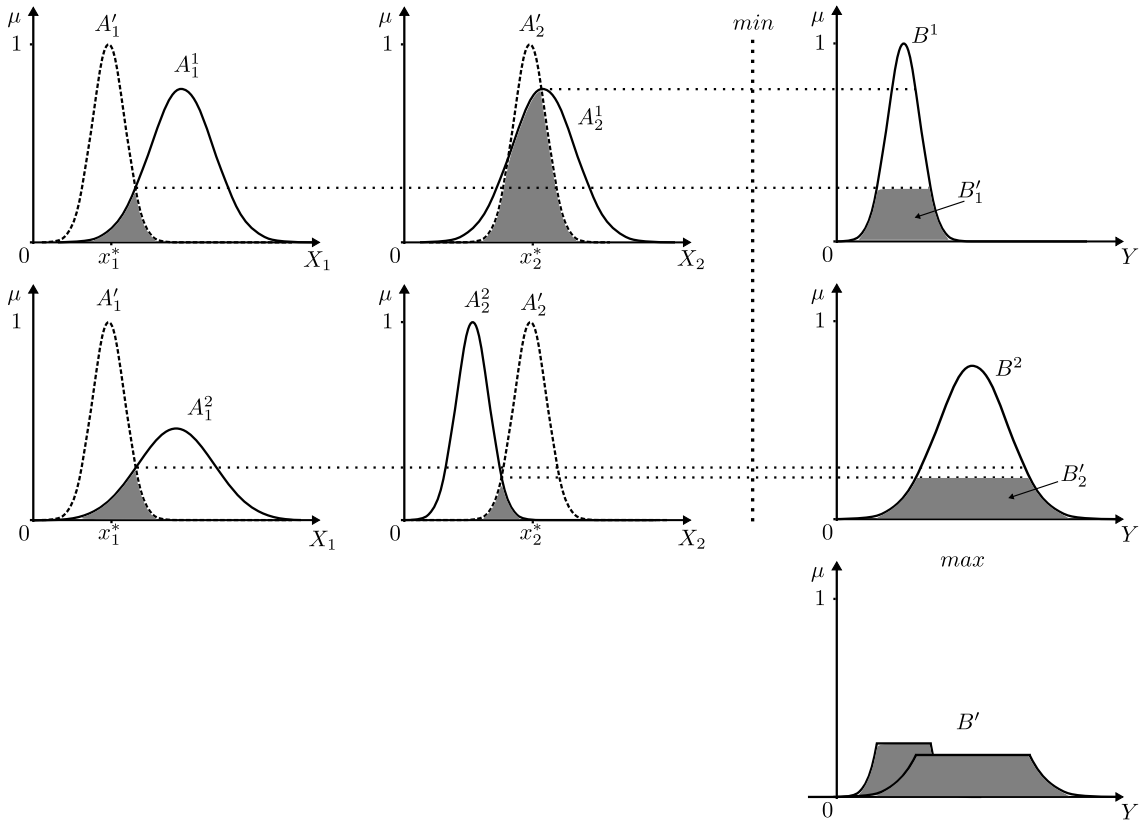


Figura 2.6: Inferência difusa com duas regras difusas e com duas variáveis difusas por antecedente.

L regras:

$$\mu_{B'}(y) = \max_{j=1,\ldots,L}[\mu_{B'_j}(y)] = \max_{j=1,\ldots,L}\left\{\max_{\mathbf{x}\in X}\left[\mu_{A'}(\mathbf{x})\left(\prod_{i=1}^{n}\mu_{A^j_i}(x_i)\right)\mu_{B^j}(y)\right]\right\}. \tag{2.22}$$

Utilizando o modelo de inferência (2.17) com a implicação (2.15), obtém-se alternativamente o seguinte motor de inferência difusa:

$$\mu_{B'}(y) = \max_{j=1,\ldots,L}\left[\mu_{B'_j}(y)\right] = \max_{j=1,\ldots,L}\left\{\max_{\mathbf{x}\in X}\left[\min\left(\mu_{A'}(\mathbf{x}), \mu_{A^j_1}(x_1), \ldots, \mu_{A^j_n}(x_n), \mu_{B^j}(y)\right)\right]\right\}. \tag{2.23}$$

Na Figura 2.6 é ilustrado esquematicamente o processo interno de obtenção da conclusão de um motor de inferência difusa com duas regras ($L = 2$) e duas antecedentes por regra ($n = 2$), ou seja:

$$\begin{aligned} &R_1\text{: SE } x_1 \text{ é } A^1_1 \text{ e } x_2 \text{ é } A^1_2\text{, ENTÃO } y \text{ é } B_1, \\ &R_2\text{: SE } x_1 \text{ é } A^2_1 \text{ e } x_2 \text{ é } A^2_2\text{, ENTÃO } y \text{ é } B_2. \end{aligned} \tag{2.24}$$

Neste sistema é utilizado o motor de inferência difusa (2.23) com o difusor gaussiano (2.19) aplicado sobre dois sinais de entrada x^*_1 e x^*_2.

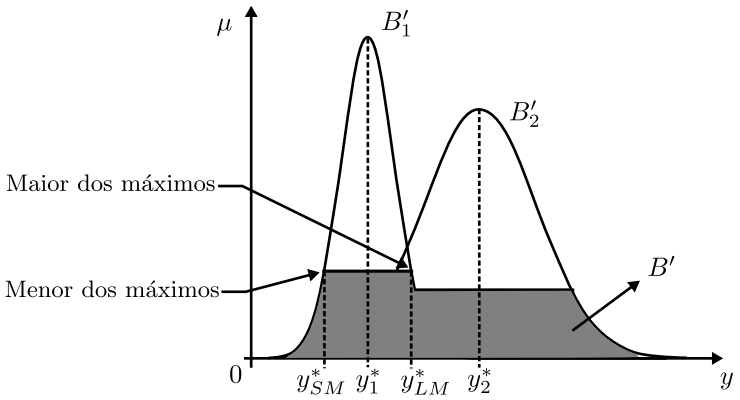


Figura 2.7: Desdifusor: determinação dos desdifusores y^*_{SM} e y^*_{LM} e dos centros y^*_j utilizados no desdifusor de média dos centros.

2.3.3 Desdifusor

A conclusão obtida por um motor de inferência difusa é definida por um conjunto difuso B' e pela respetiva função de pertença $\mu_{B'}(y)$ (ver motores de inferência difusa (2.22) e (2.23)), no entanto em situações práticas onde se lidem com grandezas físicas expressas sob a forma de valores numéricos é de interesse atribuir um significado numérico que melhor descreva B'. É necessário então definir um mapeamento que atribui a um conjunto difuso um valor numérico. A função que realiza este mapeamento é chamada de desdifusor.

O desdifusor é concretizado sob a forma de uma função que realiza o mapeamento de um conjunto difuso $B' \in Y \subset \mathbb{R}$, para um valor $y^* \in Y$. Tal como em outros componentes o desdifusor pode ser implementado matematicamente de mais de uma forma [12]. Os pontos seguintes descrevem alguns desdifusores possíveis.

- **Centro de massa**: Determina o centro de massa do conjunto difuso no seu universo de discurso, considerando que a densidade de massa em cada ponto desse domínio é dada pelo valor da função de pertença do conjunto difuso nesse ponto.A desdifusão por centro de massa é determinada pela equação (2.25).

$$y^*_{CM} = \frac{\int y \mu_{B'}(y) dy}{\int \mu_{B'}(y) dy}. \tag{2.25}$$

  Este desdifusor pode ser visto como uma função que determina o valor esperado da função de pertença de forma semelhante como se obtém o valor médio de uma função de densidade de probabilidade. Este método é computacionalmente mais exigente do que os métodos seguintes.

- **Média dos centros**: Dado que a conclusão B' de um sistema difuso resulta da

agregação das conclusões de cada regra B'_j (ver (2.22) ou (2.23)), pode-se utilizar a média dos centros dos diversos conjuntos-conclusão das diversas regras para determinar o valor de y^*_{CM} que deve resultar da desdifusão:

$$y^*_{MC} = \frac{\sum_{j=1}^{L} y^*_j w_j}{\sum_{j=1}^{L} w_j}. \tag{2.26}$$

O valor de y^*_{CM} é determinado pela equação (2.26) onde y^*_j é o centro de B'_j, e w_j é a altura de B'_j no seu centro, ou seja $w_j = \mu_{B'_j}(y^*_j)$. Este método de desdifusão, ao contrário do método de centro de massa, não determina o valor de y^*_{MC} com base no resultado final da agregação do conjunto de regras $\mu_{B'}(y)$, mas sim com base na localização do centro e amplitude da conclusão $\mu_{B'_j}(y)$ de cada regra individual. A Figura 2.7 ilustra o conjunto difuso B' definido por $\mu_{B'}(y)$ bem como os conjuntos difusos B'_j definidos por $\mu_{B'_j}(y)$ para as regras $j = 1, 2$.

- **Máximo**: Este desdifusor escolhe y^*_M tal que nesse ponto, a pertença $\mu_{B'}(y)$ seja máxima. Se este critério não definir um valor único para y^*_M, então é necessário utilizar outros critérios como o menor dos máximos y^*_{SM}, maior dos máximos y^*_{LM} ou a média destes dois y^*_{MM}. Ver Figura 2.7.

2.3.4 Sistema Difuso

Nas secções anteriores foram descritos os blocos constituintes de um sistema difuso. Com base neste conjunto de blocos é agora possível construir um sistema que interage com o exterior através de um ou mais sinais de entrada e um de saída, e cujo comportamento da saída é definido através de um conjunto de regras difusas. Para concretizar esta noção de sistema difuso, a construção de um sistema em particular é descrita em seguida.

Consideremos que a representação geral de uma regra difusa é dada por (2.21) e que as funções de pertença que descrevem os conjuntos difusos A^j_i e B^j são do tipo gaussiano e são descritas pelas equações (2.27) e (2.28) respetivamente. Os parâmetros m^j_i e m^j definem o centro das funções de pertença e os parâmetros σ^j_i e σ^j definem a largura das funções de pertença.

$$\mu_{A^j_i}(x) = e^{-\left(\frac{x_i - m^j_i}{\sigma^j_i}\right)^2} \tag{2.27}$$

$$\mu_{B^j}(y) = e^{-\left(\frac{y^j - m^j}{\sigma^j}\right)^2} \tag{2.28}$$

Se utilizarmos o motor de inferência difusa (2.22) e difusor singleto (2.18), obtemos o

seguinte motor de inferência difusa ao substituir (2.18) em (2.22):

$$\begin{aligned}\mu_{B'}(y) &= \max_{j=1,\ldots,L}\left\{\max_{\mathbf{x}\in X}\left[\mu_{A'}(\mathbf{x})\prod_{i=1}^{n}\mu_{A_i^j}(x_i)\mu_{B^j}(y)\right]\right\}\\ &= \max_{j=1,\ldots,L}\left[\prod_{i=1}^{n}\mu_{A_i^j}(x_i^*)\mu_{B^j}(y)\right]. \end{aligned} \tag{2.29}$$

O resultado da operação $\max_{\mathbf{x}\in X}$ em (2.29) é alcançado para cada regra difusa j em $x_i = x_i^*$ pela definição do difusor singleto (2.18), onde novamente o vetor de entrada não difuso é dado por $\mathbf{x}^* = (x_1^*, x_2^*, \ldots, x_n^*)$.

Por fim, o último passo necessário para obter um sistema difuso completo consiste em implementar o desdifusor. Considere-se o desdifusor de média dos centros (2.26). Este desdifusor considera a conclusão de cada regra j individualmente e identifica para cada regra o centro y_j^* da consequente e a sua amplitude $w_j = \mu_{B'_j}(y_j^*)$. Desta forma e recordando que a função de pertença da consequente utilizada é gaussiana (2.28) obtém-se que o seu centro é determinado por $y_j^* = m^j$ e que a amplitude é determinada da seguinte forma:

$$w_j = \mu_{B'_j}(y_j^*) = \mu_{B'_j}(m^j) = \prod_{i=1}^{n}\mu_{A_i^j}(x_i^*)\mu_{B^j}(m^j) = \prod_{i=1}^{n}\mu_{A_i^j}(x_i^*). \tag{2.30}$$

Substituindo $y_j^* = m^j$ e (2.30) em (2.26) obtemos o sistema difuso descrito pela equação (2.31).

$$y = \frac{\sum_{j=1}^{L} m^j \prod_{i=1}^{n}\mu_{A_i^j}(x_i^*)}{\sum_{j=1}^{L}\prod_{i=1}^{n}\mu_{A_i^j}(x_i^*)}. \tag{2.31}$$

A equação (2.31) descreve um sistema difuso com o vetor de entrada $\mathbf{x}^* = (x_1^*, x_2^*, \ldots, x_n^*)$ e com saída y. Este sistema é constituído por L regras difusas do tipo (2.21) onde cada regras possui n variáveis de entrada. Na literatura, este tipo de sistema é usualmente chamado de sistema difuso do tipo Mamdani devido à utilização das implicações difusas de Mamdani descritas pelas equações (2.11) e (2.12).

2.4 Sistemas Difusos do Tipo Takagi-Sugeno

Na generalidade, a maioria dos sistemas físicos é governado por equações dinâmicas não lineares, isto é, não podem de forma completa ser representados por um conjunto de equações diferenciais lineares. No entanto, frequentemente a operação destes sistemas físicos é efetuada dentro de uma zona de operação limitada na qual em muitos casos o modelo do sistema pode ser aproximado por um modelo linear. Nestas circunstâncias é possível a utilização de um controlador linear para controlo do processo dentro de uma zona de operação restrita. Quando a zona de operação é alargada e o comportamento

não linear do processo se torna predominante, um modelo linear deixa de poder fornecer uma aproximação satisfatória do processo, sendo neste caso necessário enveredar por metodologias de modelação alternativas.

Os sistemas difusos do tipo Takagi-Sugeno (T-S) propostos originalmente por Takagi e Sugeno [13], são uma outra classe de sistemas difusos que sendo também aproximadores universais [14] podem ser utilizados para criar modelos de sistemas. Estes sistemas diferem dos sistemas do tipo Mamdani na medida em que as suas consequentes não possuem conjuntos difusos, mas sim uma função $f_j()$ das variáveis de entrada do sistema.

Este tipo de sistema, ao contrário do tipo Mamdani, é menos intuitivo dado que a consequente não é representada por uma variável linguística, no entanto uma característica relevante deste sistema é a capacidade de interpolar um conjunto de modelos lineares definidos pelas funções $f_j()$. Esta abordagem pode ser vantajosa na modelação de sistemas não lineares, pois é assim possível definir um conjunto de modelos lineares $f_j()$ que caracterizam as diversas dinâmicas locais de um sistema e agregar todos estes modelos sob a forma de um modelo global válido para toda a gama de operação desejada. Cada modelo $f_j()$ é local a uma zona de funcionamento definida pela antecedente da regra difusa j.

Considere-se que, de forma genérica, uma regra difusa do tipo T-S com n sinais de entrada tem a seguinte forma,

$$\begin{aligned} R_j : \text{SE } x_1(k) \text{ é } A_1^j \text{ e } x_2(k) \text{ é } A_2^j \text{ e } \ldots \text{ e } x_n(k) \text{ é } A_n^j \\ \text{ENTÃO } y_j(k+1) = f_j(x_1(k), x_2(k), \ldots, x_n(k)), \end{aligned} \tag{2.32}$$

onde a antecedente é idêntica à antecedente existente numa implicação difusa do tipo Mamdani, mas a consequente é composta por uma função das variáveis de entrada. Se selecionarmos o difusor singleto (2.18) e o operador de interseção (2.7), podemos definir o peso w_j de cada regra como sendo o valor lógico da antecedente da regra j dado através da expressão (2.33).

$$w_j = \prod_{i=1}^{n} \mu_{A_i^j}(x_i^*(k)). \tag{2.33}$$

A saída de um sistema T-S é determinada através de uma média ponderada dos valores das consequentes de cada regra y_j com o fator de ponderação definido pelo valor dos pesos w_j obtidos a partir das respetivas antecedentes das regras. A equação (2.34) representa o cálculo do valor da saída do modelo T-S.

$$y(k+1) = \frac{\sum_{j=1}^{L} y_j(k+1) w_j}{\sum_{j=1}^{L} w_j} = \frac{\sum_{j=1}^{L} y_j(k+1) \prod_{i=1}^{n} \mu_{A_i^j}(x_i^*(k))}{\sum_{j=1}^{L} \prod_{i=1}^{n} \mu_{A_i^j}(x_i^*(k))} \tag{2.34}$$

Capítulo 3

Controlo Preditivo com Rede Neuronal Difusa Recorrente

Um dos pontos positivos da lógica difusa é a sua capacidade para servir de base para construção e implementação de modelos de sistemas físicos com recurso a regras SE-ENTÃO. Uma aplicação deste tipo de modelação é apresentada neste capítulo, no contexto da apresentação de uma solução para o problema de controlo de sistemas não lineares.

A abordagem desenvolvida neste capítulo foi proposta originalmente em [15] e consiste em utilizar a capacidade das regras difusas em particionar um universo de discurso em subconjuntos. Cada um destes conjuntos irá representar uma zona de operação do sistema não linear global, onde este será representado localmente por um modelo linearizado que aproxima o comportamento do sistema global. Esta função é desempenhada por uma rede neuronal difusa recursiva que tem como resultado final um modelo não linear equivalente do sistema não linear global, sendo este modelo global composto por modelos lineares locais. Esta abordagem permite assim a utilização de metodologias de controlo já existentes, que pressupõem a existência de um modelo linear, no controlo de sistemas não lineares.

As redes neuronais difusas recursivas (do inglês *Recurrent Fuzzy Neural Network* [15], doravante abreviado RFNN) utilizam para além da lógica difusa, que consolida as regras que geram o particionamento do universo de discurso, a metodologia de retropropagação do erro. Esta é metodologia de aprendizagem da área das redes neuronais, e fornece a RFNN da capacidade de esta se adaptar ao longo do tempo, em função de diferentes estímulos externos. Desta forma, a RFNN tem a capacidade de melhorar iterativamente a aproximação do comportamento do sistema não linear com base nos seus sinais de entrada e saída que são lidos. Uma outra característica da RFNN, é o facto de esta ser uma evolução da rede neuronal difusa (do inglês *Fuzzy Neural Network* [16], abreviado FNN), onde são adicionados elementos de memória à estrutura interna de uma FNN. Estes

elementos de memória consistem em malhas fechadas com ganhos associados, e permitem lidar com sistemas dinâmicos de uma forma mais eficiente [16]. A estrutura da RFNN e os seus conceitos serão expostos na secção 3.1.

Nesta dissertação, a motivação principal para construir um modelo com base em RFNN, é a sua aplicação a sistemas de controlo. A metodologia de controlo utilizada nas próximas secções é o controlo preditivo. O controlo preditivo é uma classe de técnicas de controlo utilizada na industria [3]. É particularmente popular na industria química, onde os processos são normalmente caracterizados por tempos de resposta lentos e com atrasos de transporte entre a entrada e a saída [15]. Esta classe de controladores caracteriza-se por utilizar o modelo do processo, frequentemente obtido através de identificação como é o caso da abordagem baseada na RFNN aqui utilizada, para estimar a trajetória futura da saída da planta. Com base nesta previsão é possível estimar o desvio futuro face a uma trajetória desejada para a saída do processo, e assim gerar uma lei de controlo que permita compensar esse desvio. Neste capítulo é abordada uma das técnicas que pertencem a esta classe chamada de controlo preditivo generalizado (do inglês *Generalized Predictive Control*, doravante abreviado por GPC). Foi desenvolvido por Clarke em conjunto com outros [17]. Este tópico será desenvolvido na secção 3.2.

3.1 Rede Neuronal Difusa Recorrente (RFNN)

Na identificação de processos com uma entrada e uma saída é frequente utilizar modelos do tipo ARMAX,

$$y(k) = a(z^{-1})y(k-1) + z^{-d}b(z^{-1})u(k-1) + \frac{1}{\Delta}\xi(k), \tag{3.1}$$

onde $a(z^{-1})$ e $b(z^{-1})$ são os polinómios,

$$\begin{aligned} a(z^{-1}) &= a_1 + a_2 z^{-1} + \ldots + a_{n_y} z^{-(n_y-1)}, \\ b(z^{-1}) &= b_0 + b_1 z^{-1} + \ldots + b_{n_u} z^{-n_u}. \end{aligned} \tag{3.2}$$

$u(k) \in \mathbb{R}$ e $y(k) \in \mathbb{R}$ são a entrada e a saída do processo, $n_u \in \mathbb{N}$ e $n_y \in \mathbb{N}$ são as ordens do polinómio de entrada e de saída respetivamente, $d \in \mathbb{N}$ é o atraso de transporte do processo, $\xi(k) \in \mathbb{R}$ representa ruído branco com média nula, e z^{-1} representa o operador de atraso unitário. O símbolo Δ representa o operador de primeira diferença $\Delta = 1 - z^{-1}$. Estes modelos são aplicáveis em sistemas que demonstrem um comportamento predominantemente linear. Se a dinâmica do sistema for não linear, o modelo anterior é insuficiente.

Uma classe grande de sistemas não lineares pode ser descrito sob a forma de um modelo

NARMAX [18],

$$\begin{aligned} y(k) = f(&y(k-1), y(k-2), \ldots, y(k-n_y), \\ &u(k-d-1), u(k-d-2), \ldots, u(k-d-n_u-1)) + \frac{1}{\Delta}\xi(k), \end{aligned} \tag{3.3}$$

onde $f(\cdot)$ é uma função não linear dos seus argumentos. Normalmente esta é desconhecida, sendo assim importante encontrar uma representação que sirva de estimativa para esta função. Neste texto, propõe-se que $f(\cdot)$ seja representada por uma rede neuronal difusa recursiva (RFNN). Estas redes são aproximadores universais [16], sendo assim apropriadas a este problema. Este tipo de sistema cria um modelo global de um processo, agregando vários modelos lineares locais R_j, cada um associado a um regime de operação distinto, ou zona de funcionamento. A interpolação destes modelos locais é efetuada através de um mecanismo de decisão baseado em lógica difusa. A RFNN pode ser então descrita na forma (ver Figura 3.1),

$$\begin{aligned} R_j : &\text{ SE } x_{a,1}(k) \text{ é } A_1^j, \text{ e } \ldots \text{ e } x_{a,\bar{n}}(k) \text{ é } A_{\bar{n}}^j \\ &\text{ ENTÃO } y_j(k) = a'_j(z^{-1})y(k-1) + z^{-d}b'_j(z^{-1})u(k-1), \end{aligned} \tag{3.4}$$

onde $a'_j(z^{-1})$ e $b'_j(z^{-1})$ são os polinómios,

$$\begin{aligned} a'_j(z^{-1}) &= a_{1j} + a_{2j}z^{-1} + \ldots + a_{n_yj}z^{-(n_y-1)}, \\ b'_j(z^{-1}) &= b_{0j} + b_{1j}z^{-1} + \ldots + b_{n_uj}z^{-n_u}. \end{aligned} \tag{3.5}$$

R_j $(j = 1, \ldots, L)$ representa a regra difusa j, e L representa o número de regras difusas. A RFNN tem como entrada o vetor de variáveis linguísticas [12] $\mathbf{x}_a(k) = [x_{a,1}(k), \ldots, x_{a,\bar{n}}(k)]^T$. Por outro lado, tendo em conta (3.4) define-se o vetor de variáveis de entrada das funções consequentes das regras da RFNN da seguinte forma:

$$\begin{aligned} \mathbf{x}_e(k) &= [x_{e,1}(k), \ldots, x_{e,\bar{n}}(k)]^T \\ &= [y(k-1), y(k-2), \ldots, y(k-n_y), u(k-d-1), \ldots, u(k-d-n_u-1)]^T. \end{aligned} \tag{3.6}$$

Os termos A_i^j $(i = 1, \ldots, \bar{n},\ j = 1, \ldots, L)$ são termos linguísticos descritos por conjuntos difusos [12] que permitem descrever as zonas de operação locais.

Sem perda de generalidade, nesta dissertação assume-se que $\mathbf{x}_a(k) = \mathbf{x}_e(k)$. Se fosse $\mathbf{x}_a(k) \neq \mathbf{x}_e(k)$, vetores compostos por diferentes conjuntos de variáveis, então por (3.4) torna-se claro que se poderia desenvolver a análise definindo um novo vetor de entradas e saídas $\bar{\mathbf{x}}_a(k) = \bar{\mathbf{x}}_e(k)$ que teria como componentes a união das componentes dos vetores $\mathbf{x}_a(k)$ e $\mathbf{x}_e(k)$, e adicionalmente considerar que seriam nulos certos termos antecedentes

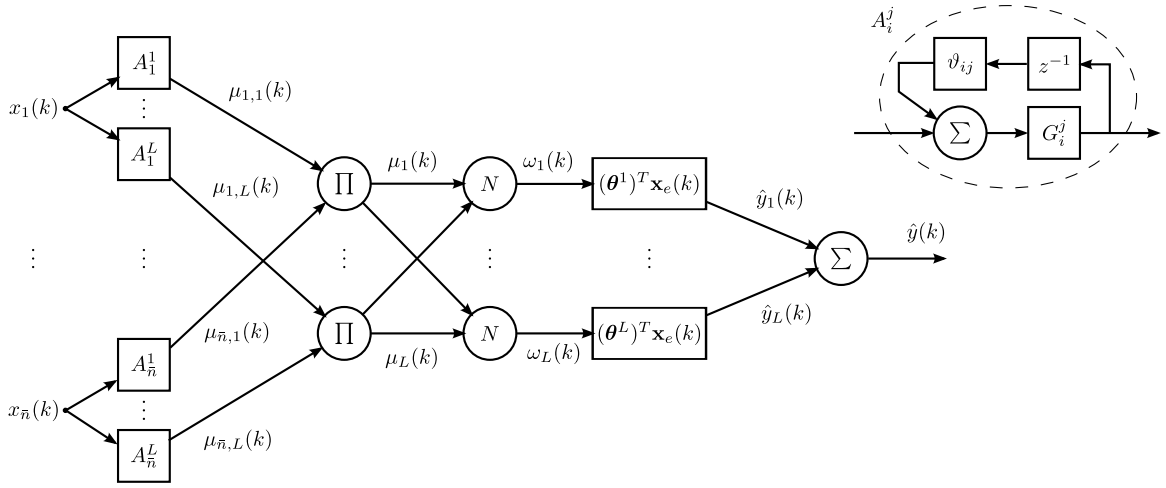


Figura 3.1: Estrutura de uma rede neuronal difusa com nós recursivos.

A_i^j e/ou certos coeficientes das funções consequentes. Dado que $\mathbf{x}_a(k) = \mathbf{x}_e(k)$, então temos que as dimensões $\bar{n}$ de ambos os vetores são determinadas por $\bar{n} = n_y + n_u + 1$.

3.1.1 Estrutura Interna

A RFNN foi inicialmente proposta em [16] sendo que a forma aqui apresentada segue a referência [15]. A estrutura da RFNN está apresentada na Figura 3.1. A RFNN é composta por várias camadas com funções distintas que serão descritas de seguida.

Camada 1 Entrada: É a primeira camada, ou camada de entrada na rede. Esta camada tem m nós e é responsável apenas por receber e transferir para a camada seguinte os sinais de entrada $x_{e,i}(k)$.

Camada 2 Funções de pertença: Esta camada é responsável por determinar os valores das funções de pertença antecedentes e contém também uma unidade de memória. Nesta rede são utilizadas funções de pertença do tipo Gaussiano.

A função (3.7) descreve a relação entre a entrada e a saída de um nó A_i^j desta camada, onde i representa o número da entrada da rede, $i = 1, \ldots, \bar{n}$, e j representa o número da regra difusa $j = 1, \ldots, L$. $\mu_{ij}(k)$ representa a saída de um nó.

$$\mu_{ij}(k) = \exp\left[-\frac{(x_{e,i}(k) + \mu_{ij}(k-1)\vartheta_{ij} - m_{ij})^2}{2\sigma_{ij}^2}\right] \tag{3.7}$$

Os termos m_{ij} e σ_{ij} representam o centro e a largura da função de pertença gaussiana (representada na Figura 3.1 pelo bloco G_i^j), respetivamente. Uma característica importante que torna esta rede dinâmica ao invés de ser apenas um mapeamento fixo

entre a entrada e a saída é a inclusão de realimentação da saída desta camada para a sua entrada através de uma linha de atraso com um ganho ϑ_{ij}: especificamente, a saída de um nó desta camada 2, depende não só de uma entrada da RFNN, mas também da saída do próprio neurónio no instante anterior (3.7). m_{ij}, σ_{ij} e ϑ_{ij} são os parâmetros ajustáveis presentes nesta camada.

Camada 3 Antecedentes das regras difusas: A saída desta camada representa o valor de ativação das antecedentes das regras. A sua saída j da Camada 3 é calculada como o produto de certas saídas da camada anterior, de forma a efetuar a operação "e" antecedente da regra j. O valor da saída j ($j = 1, \ldots, L$) é dado por (3.8),

$$\mu_j(k) = \prod_{i=1}^{\bar{n}} \mu_{ij}(k). \tag{3.8}$$

Camada 4 Nível de ativação normalizado: Os valores de ativação das regras são normalizados nesta camada de acordo com a equação (3.9), onde o valor de ativação de uma regra é dividido pela soma dos valores de ativação de todas as regras (tendo (2.34) como inspiração).

$$\omega_j(k) = \frac{\mu_j(k)}{\sum_{j=1}^{L} \mu_j(k)} \tag{3.9}$$

Camada 5 Consequentes das regras difusas: Esta camada contém as consequentes das regras difusas. A consequente de cada regra difusa da RFNN (3.4) possui um modelo linear (3.4 que descreve a dinâmica da planta na zona de operação descrita pela respetiva antecedente da regra difusa j. Este modelo possui um conjunto de parâmetros ajustáveis contidos no vetor

$$\boldsymbol{\theta}^j = \left[a_{1j}, \ldots, a_{n_y j}, b_{0j}, \ldots, b_{n_u j}\right]^T, \tag{3.10}$$

que concretizam o modelo linear e a saída do nó desta camada têm a forma

$$\hat{y}_j(k) = (\boldsymbol{\theta}^j)^T \mathbf{x}_e(k), \tag{3.11}$$

onde $\hat{y}_j(k)$ representa o valor da saída do processo estimado pela regra j.

Camada 6 Saída da rede: A última camada é responsável por determinar $\hat{y}(k)$ que representa a saída estimada do processo que a RFNN está a identificar. Esta é obtida como a soma do resultado de cada uma das regras individuais $\hat{y}_j(k)$ ponderado pelo fator $\omega_j(k)$ (3.9),

$$\hat{y}(k) = \sum_{j=1}^{L} \omega_j(k) \hat{y}_j(k). \tag{3.12}$$

3.1.2 Algoritmo de Aprendizagem

Como foi referido na secção anterior, a RFNN possui internamente parâmetros que são ajustáveis o que lhe permite aproximar o modelo de um processo desconhecido. A aprendizagem destes parâmetros é efetuada em dois passos distintos. No primeiro passo são treinados os parâmetros das antecedentes das regras difusas com recurso a um algoritmo comum no contexto de redes neuronais. Neste caso trata-se do algoritmo de retropropagação do erro baseado num algoritmo de minimização de uma função de custo que representa a qualidade de aproximação da rede, fazendo recurso ao gradiente dessa função de custo. No segundo passo, as consequentes das regras são ajustadas por regressão linear através do método dos mínimos quadrados.

Treino das Antecedentes

Pretende-se ajustar os parâmetros m_{ij}, σ_{ij} e ϑ_{ij} de forma a minimizar uma função de custo baseada no erro de aproximação da rede. Irá ser obtida como resultado uma lei de aprendizagem recursiva que pode ser utilizada *online* durante a operação do sistema.

A função de custo é definida como sendo o quadrado do erro entre a saída real do processo e a saída estimada pela RFNN em cada instante de tempo k. Esta função de custo é determinada por:

$$\varphi(k) = \frac{1}{2}(y(k) - \hat{y}(k))^2, \tag{3.13}$$

onde $y(k)$ e $\hat{y}(k)$ são respetivamente a saída real do processo e a saída estimada pela RFNN no instante de tempo k. O ajuste dos parâmetros consiste numa lei de ajuste onde para cada parâmetro da rede é determinada uma componente de correção. Se definirmos um vetor $\mathbf{W}$ que contém os parâmetros da rede, a lei de ajuste dos parâmetros pode ser definida por (3.14),

$$\mathbf{W}(k+1) = \mathbf{W}(k) + \mathbf{\Delta W(k)}, \tag{3.14}$$

onde $\mathbf{W} = \left[\mathbf{w}_1^T, \mathbf{w}_2^T, \ldots, \mathbf{w}_L^T\right]$ representa o vetor de parâmetros ajustáveis da rede, e $\mathbf{w}_j = [m_{1j}, \ldots, m_{\tilde{n}j}, \sigma_{1j}, \ldots, \sigma_{\tilde{n}j}, \vartheta_{1j}, \ldots, \vartheta_{\tilde{n}j}]$ contem os parâmetros que constituem cada regra j. O termo $\mathbf{\Delta W(k)}$ representa a componente de correção dos parâmetros.

O método utilizado para ajustar os parâmetros da rede é familiarmente utilizado em aprendizagem em redes neuronais e consiste na utilização do gradiente da função de custo. O princípio consiste em corrigir cada parâmetro da rede por uma componente de correção proporcional ao simétrico da derivada parcial da função de custo (3.13) em função de cada componente, em cada instante k. Este fator de correção é definido por

$$\mathbf{\Delta W(k)} = \eta \left(-\frac{\partial \varphi(k)}{\partial \mathbf{W}} \right), \tag{3.15}$$

onde η representa uma constante de aprendizagem. Substituindo a função de custo (3.13) em (3.15), substituindo (3.15) em (3.14) e determinando as derivadas parciais em função de cada parâmetro m_{ij}, σ_{ij} e ϑ_{ij}, obtém-se as seguintes leis de aprendizagem (3.16)-(3.18),

$$m_{ij}(k+1) = m_{ij}(k) - \eta\frac{\partial\varphi(k)}{\partial m_{ij}} = m_{ij}(k) + \eta(y(k)-\hat{y}(k))\frac{\partial\hat{y}(k)}{\partial m_{ij}}, \tag{3.16}$$

$$\sigma_{ij}(k+1) = \sigma_{ij}(k) - \eta\frac{\partial\varphi(k)}{\partial \sigma_{ij}} = \sigma_{ij}(k) + \eta(y(k)-\hat{y}(k))\frac{\partial\hat{y}(k)}{\partial \sigma_{ij}}, \tag{3.17}$$

$$\vartheta_{ij}(k+1) = \vartheta_{ij}(k) - \eta\frac{\partial\varphi(k)}{\partial \vartheta_{ij}} = \vartheta_{ij}(k) + \eta(y(k)-\hat{y}(k))\frac{\partial\hat{y}(k)}{\partial \vartheta_{ij}}, \tag{3.18}$$

onde

$$\frac{\partial\hat{y}(k)}{\partial m_{ij}} = \frac{\partial\hat{y}(k)}{\partial\mu_j(k)}\frac{\partial\mu_j(k)}{\partial\mu_{ij}(k)}\frac{\partial\mu_{ij}(k)}{\partial m_{ij}} = (\hat{y}_j(k)-\hat{y}(k))\omega_j(k)\frac{(x_{e,i}(k)+\mu_{ij}(k-1)\vartheta - m_{ij})}{\sigma_{ij}^2}, \tag{3.19}$$

$$\frac{\partial\hat{y}(k)}{\partial \sigma_{ij}} = \frac{\partial\hat{y}(k)}{\partial\mu_j(k)}\frac{\partial\mu_j(k)}{\partial\mu_{ij}(k)}\frac{\partial\mu_{ij}(k)}{\partial \sigma_{ij}} = (\hat{y}_j(k)-\hat{y}(k))\omega_j(k)\frac{(x_{e,i}(k)+\mu_{ij}(k-1)\vartheta - m_{ij})^2}{\sigma_{ij}^3}, \tag{3.20}$$

$$\frac{\partial\hat{y}(k)}{\partial \vartheta_{ij}} = \frac{\partial\hat{y}(k)}{\partial\mu_j(k)}\frac{\partial\mu_j(k)}{\partial\mu_{ij}(k)}\frac{\partial\mu_{ij}(k)}{\partial \vartheta_{ij}} = (\hat{y}_j(k)-\hat{y}(k))\omega_j(k)\frac{(m_{ij} - x_{e,i}(k)-\mu_{ij}(k-1)\vartheta)}{\sigma_{ij}^2}\mu_{ij}(k-1). \tag{3.21}$$

A convergência da lei de aprendizagem é provada em [15], onde se garante que a RFNN converge se a constante de aprendizagem η for definida por

$$\eta = \frac{\beta}{\Sigma_{j=1}^{L}\Sigma_{i=1}^{\bar{n}}\left(\left(\frac{\partial\hat{y}(k)}{\partial m_{ij}}\right)^2 + \left(\frac{\partial\hat{y}(k)}{\partial \sigma_{ij}}\right)^2 + \left(\frac{\partial\hat{y}(k)}{\partial \vartheta_{ij}}\right)^2\right)}, \tag{3.22}$$

com $0 < \beta < 2$. Em [19], a constante de aprendizagem é definida com $\beta = 1$.

Treino das Consequentes

Cada regra difusa j possui na sua consequente um vetor $\boldsymbol{\theta}^j$ (3.10) de parâmetros que são ajustáveis. Este vetor contém os coeficientes da equação de diferenças (função) (3.11)

que constitui a consequente da regra j. Para todas as regras, estes coeficientes devem ser ajustados de forma a que o modelo gerado pela RFNN melhor se aproxime do processo físico que se pretende identificar/representar.

O método de identificação dos coeficientes consequentes utilizado consiste em minimizar a soma dos quadrados das diferenças entre os valores da saída do processo $y(k)$ e os valores estimados pela RFNN $\hat{y}(k)$. Descrito matematicamente, consiste em minimizar o seguinte índice J:

$$J(\boldsymbol{\Theta}, N) = \frac{1}{2}\sum_{k=1}^{N}(y(k) - \hat{y}(k))^2. \tag{3.23}$$

Este método de estimação é utilizado quando se pretende estimar um modelo linear nos parâmetros, como o descrito por

$$y(k) = \theta^1\psi^1(k) + \theta^2\psi^2(k) + \ldots + \theta^s\psi^s(k) = \boldsymbol{\Theta}^T\boldsymbol{\Psi}(k), \tag{3.24}$$

onde $\boldsymbol{\Theta}^T = [\theta^1, \theta^2, \ldots, \theta^s]$ são os parâmetros do modelo a serem determinados, $\boldsymbol{\Psi}^T(k) = [\psi^1(k), \psi^2(k), \ldots, \psi^s(k)]$ podem ser funções conhecidas que dependem de outras variáveis do sistema, e $y(k)$ é a variável observada no processo.

O modelo gerado pela última camada da RFNN (3.12) pode ser expresso com o formato descrito na equação (3.24). Substituindo a equação (3.11) em (3.12) e expandindo a soma obtém-se:

$$\begin{aligned}
\hat{y}(k) =& \sum_{j=1}^{L}\omega_j(k)\hat{y}_j(k) = \sum_{j=1}^{L}\omega_j(k)\mathbf{x}_e^T(k)\boldsymbol{\theta}^j \\
=& \omega_1(k)\mathbf{x}_e^T(k)\boldsymbol{\theta}^1 + \omega_2(k)\mathbf{x}_e^T(k)\boldsymbol{\theta}^2 + \ldots + \omega_L(k)\mathbf{x}_e^T(k)\boldsymbol{\theta}^L \\
=& a_{1,1}\omega_1(k)y(k-1) + \ldots + a_{n_y,1}\omega_1(k)y(k-n_y) \\
&+ b_{0,1}\omega_1(k)u(k-d-1) + \ldots + b_{n_u,1}\omega_1(k)u(k-d-n_u-1) + \ldots \\
&+ a_{1,L}\omega_L(k)y(k-1) + \ldots + a_{n_y,L}\omega_L(k)y(k-n_y) \\
&+ b_{0,L}\omega_L(k)u(k-d-1) + \ldots + b_{n_u,L}\omega_L(k)u(k-d-n_u-1).
\end{aligned} \tag{3.25}$$

Observando as equações (3.25) e (3.10), podem-se definir os vetores $\boldsymbol{\Theta}$ e $\boldsymbol{\Psi}$:

$$\boldsymbol{\Theta}^T = \left[(\boldsymbol{\theta}^1)^T, (\boldsymbol{\theta}^2)^T, \ldots, (\boldsymbol{\theta}^L)^T\right], \tag{3.26}$$

$$\boldsymbol{\Psi}^T(k) = \left[(\boldsymbol{\psi}^1(k))^T, (\boldsymbol{\psi}^2(k))^T, \ldots, (\boldsymbol{\psi}^L(k))^T\right], \tag{3.27}$$

$$\begin{aligned}
(\boldsymbol{\psi}^j(k))^T =& \omega_j(k)\mathbf{x}_e^T(k) \\
=& [\omega_j(k)y(k-1), \ldots, \omega_j(k)y(k-n_y), \\
& \omega_j(k)\, u(k-d-1), \ldots, \omega_j(k)u(k-d-n_u-1)].
\end{aligned} \tag{3.28}$$

Assim, pode-se reescrever a equação (3.25) na seguinte forma:

$$\hat{y}(k) = \boldsymbol{\Theta}^T \boldsymbol{\Psi}(k), \tag{3.29}$$

onde (3.29) é da forma (3.24), sendo $\boldsymbol{\Theta}$ e $\boldsymbol{\Psi}(k)$ vetores de dimensão $s = nL$.

Os parâmetros do modelo são estimados com base num conjunto de pares $(y(k), \boldsymbol{\Psi}(k))$, observados no processo físico nos instantes $k = 1, 2, \ldots, t$, onde t é o instante de observação corrente. Definam-se as seguintes matrizes,

$$\mathbf{y}(t) = [y(1), y(2), \ldots, y(t)]^T, \tag{3.30}$$

$$\boldsymbol{\Phi}(t) = \begin{bmatrix} \boldsymbol{\Psi}^T(1) \\ \boldsymbol{\Psi}^T(2) \\ \vdots \\ \boldsymbol{\Psi}^T(t) \end{bmatrix} = [\boldsymbol{\Psi}(1), \boldsymbol{\Psi}(2), \ldots, \boldsymbol{\Psi}(t)]^T, \tag{3.31}$$

$$\mathbf{E}(t) = [e(1), e(2), \ldots, e(t)]^T. \tag{3.32}$$

$e(k)$ representa o erro de estimação e é dado por:

$$e(k) = y(k) - \hat{y}(k) = y(k) - \boldsymbol{\Theta}^T \boldsymbol{\Psi}(k). \tag{3.33}$$

Com estas matrizes ((3.30), (3.31), (3.32)) e com a definição (3.33), podemos reescrever a equação (3.23) da seguinte forma,

$$J(\boldsymbol{\Theta}, t) = \frac{1}{2} \sum_{k=1}^{t} e^2(k) = \frac{1}{2} \mathbf{E}^T(t) \mathbf{E}(t) = \frac{1}{2} (\mathbf{y}(t) - \boldsymbol{\Phi}(t)\boldsymbol{\Theta})^T (\mathbf{y}(t) - \boldsymbol{\Phi}(t)\boldsymbol{\Theta}). \tag{3.34}$$

Como a função J é quadrática em $\boldsymbol{\Theta}$, o mínimo desta pode ser encontrado se a sua derivada parcial em $\boldsymbol{\Theta}$ for determinada e a equação resultante igualada a zero, ou seja, pretende-se determinar a solução da seguinte equação:

$$\frac{\partial J(\boldsymbol{\Theta}, t)}{\partial \boldsymbol{\Theta}} = 0. \tag{3.35}$$

A derivada parcial de J em relação a $\boldsymbol{\Theta}$ é dada por:

$$\begin{aligned} \frac{\partial J(\boldsymbol{\Theta}, t)}{\partial \boldsymbol{\Theta}} =& \frac{\partial}{\partial \boldsymbol{\Theta}} \left(\frac{1}{2} (y(t) - \boldsymbol{\Phi}(t)\boldsymbol{\Theta})^T (\mathbf{y}(t) - \boldsymbol{\Phi}(t)\boldsymbol{\Theta}) \right) \\ =& \frac{\partial}{\partial \boldsymbol{\Theta}} \left(\frac{1}{2} (\mathbf{y}^T(t)\mathbf{y}(t) - \mathbf{y}^T(t)\boldsymbol{\Phi}(t)\boldsymbol{\Theta} - \boldsymbol{\Theta}^T \boldsymbol{\Phi}^T(t)\mathbf{y}(t) + \boldsymbol{\Theta}^T \boldsymbol{\Phi}^T(t)\boldsymbol{\Phi}(t)\boldsymbol{\Theta}) \right) \\ =& \boldsymbol{\Phi}^T(t)\boldsymbol{\Phi}(t)\boldsymbol{\Theta} - \boldsymbol{\Phi}^T(t)\mathbf{y}(t). \end{aligned} \tag{3.36}$$

Substituindo (3.36) em (3.35) e resolvendo em ordem a $\boldsymbol{\Theta}$ obtém-se,

$$\begin{aligned}\boldsymbol{\Phi}^T(t)\boldsymbol{\Phi}(t)\boldsymbol{\Theta} =& \boldsymbol{\Phi}^T(t)\mathbf{y}(t)\\ \hat{\boldsymbol{\Theta}} =& (\boldsymbol{\Phi}^T(t)\boldsymbol{\Phi}(t))^{-1}\boldsymbol{\Phi}^T(t)\mathbf{y}(t),\end{aligned} \tag{3.37}$$

em que $\hat{\boldsymbol{\Theta}}$ representa a aproximação/estimativa do vetor $\boldsymbol{\Theta}$. Esta solução apenas é possível se a matriz $(\boldsymbol{\Phi}^T(t)\boldsymbol{\Phi}(t))^{-1}$ for não singular.

A solução para $\boldsymbol{\Theta}$ obtida pela equação (3.37) representa a estimativa que minimiza os quadrados das diferenças entre o valor observado da saída do processo real $y(k)$ e o valor estimado pelo modelo $\hat{y}(k)$, no entanto este algoritmo necessita que um conjunto de pares $(y(k), \boldsymbol{\Psi}(k))$ sejam obtidos para cada estimação. Em sistemas de controlo que operam em tempo real, e em que o modelo do processo seja atualizado em cada instante t de forma a obter a melhor aproximação possível, o método de estimação apresentado não é conveniente dado que este não reutiliza a informação obtida na iteração anterior $t-1$. Uma forma de resolver este problema é reescrever o método para obter a solução que otimiza (3.23) através de um conjunto de equações recursivas.

Defina-se a seguinte matriz de covariâncias

$$\mathbf{R}(t) = (\boldsymbol{\Phi}^T(t)\boldsymbol{\Phi}(t))^{-1} = \left(\sum_{k=1}^{t}\boldsymbol{\Psi}(k)\boldsymbol{\Psi}^T(k)\right)^{-1}. \tag{3.38}$$

Se determinarmos o inverso de $\mathbf{R}(t)$, a equação (3.38) pode ser rescrita da seguinte forma recursiva,

$$\begin{aligned}\mathbf{R}^{-1}(t) =& \boldsymbol{\Phi}^T(t)\boldsymbol{\Phi}(t) = \sum_{k=1}^{t}\boldsymbol{\Psi}(k)\boldsymbol{\Psi}^T(k)\\ =& \sum_{k=1}^{t-1}\boldsymbol{\Psi}(k)\boldsymbol{\Psi}^T(k) + \boldsymbol{\Psi}(t)\boldsymbol{\Psi}^T(t)\\ =& \mathbf{R}^{-1}(t-1) + \boldsymbol{\Psi}(t)\boldsymbol{\Psi}^T(t).\end{aligned} \tag{3.39}$$

Como cada iteração do algoritmo de estimação irá produzir uma nova estimativa dos parâmetros do modelo, pode-se definir o vetor $\hat{\boldsymbol{\Theta}}(t)$ como sendo o vetor de parâmetros estimados no instante t. Com a ajuda desta notação, pode-se rescrever a equação (3.37) numa forma recursiva.

Rescrevendo a equação (3.37) para o instante $t-1$ e substituindo em $\mathbf{R}^{-1}(t-1)$ o

valor obtido pela equação (3.39), obtém-se

$$\begin{aligned}\boldsymbol{\Phi}^T(t-1)\mathbf{y}(t-1) =& \mathbf{R}^{-1}(t-1)\hat{\boldsymbol{\Theta}}(t-1)\\ =& \mathbf{R}^{-1}(t)\hat{\boldsymbol{\Theta}}(t-1) - \boldsymbol{\Psi}(t)\boldsymbol{\Psi}^T(t)\hat{\boldsymbol{\Theta}}(t-1).\end{aligned} \tag{3.40}$$

Rescrevendo novamente a equação (3.37) em $\hat{\boldsymbol{\Theta}}(t)$, temos

$$\begin{aligned}\hat{\boldsymbol{\Theta}}(t) =& \mathbf{R}(t)\boldsymbol{\Phi}^T(t)\mathbf{y}(t) = \mathbf{R}(t)\left(\sum_{k=1}^{t}\boldsymbol{\Psi}(k)y(k)\right)\\ =& \mathbf{R}(t)\left(\boldsymbol{\Phi}^T(t-1)\mathbf{y}(t-1) + \boldsymbol{\Psi}(t)y(t)\right),\end{aligned} \tag{3.41}$$

e substituindo a equação (3.40) em (3.41), obtemos a seguinte forma recursiva para determinar $\hat{\boldsymbol{\Theta}}(t)$,

$$\begin{aligned}\hat{\boldsymbol{\Theta}}(t) =& \hat{\boldsymbol{\Theta}}(t-1) - \mathbf{R}(t)\boldsymbol{\Psi}(t)\boldsymbol{\Psi}^T(t)\hat{\boldsymbol{\Theta}}(t-1) + \mathbf{R}(t)\boldsymbol{\Psi}(t)y(t)\\ =& \hat{\boldsymbol{\Theta}}(t-1) + \mathbf{R}(t)\boldsymbol{\Psi}(t)\left(y(t) - \boldsymbol{\Psi}^T(t)\hat{\boldsymbol{\Theta}}(t-1)\right)\\ =& \hat{\boldsymbol{\Theta}}(t-1) + \mathbf{K}(t)e(t),\end{aligned} \tag{3.42}$$

onde,

$$\mathbf{K}(t) = \mathbf{R}(t)\boldsymbol{\Psi}(t), \tag{3.43}$$

$$e(t) = y(t) - \boldsymbol{\Psi}^T(t)\hat{\boldsymbol{\Theta}}(t-1). \tag{3.44}$$

É necessária ainda uma forma de determinar recursivamente $\mathbf{R}(t)$ e não $\mathbf{R}^{-1}(t)$ como já definido em (3.39). Considere-se a seguinte identidade [1],

$$(\mathbf{A} + \mathbf{BCD})^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{B}(\mathbf{C}^{-1} + \mathbf{DA}^{-1}\mathbf{B})^{-1}\mathbf{DA}^{-1}, \tag{3.45}$$

onde as dimensões das matrizes são definidas da seguinte forma, $\mathbf{A}$ é $n \times n$, $\mathbf{C}$ é $n \times m$, $\mathbf{C}$ é $m \times m$, e $\mathbf{V}$ é $m \times n$. Invertendo ambos os lados da equação (3.39) temos

$$\mathbf{R}(t) = \left(\mathbf{R}^{-1}(t-1) + \boldsymbol{\Psi}(t)\boldsymbol{\Psi}^T(t)\right)^{-1}. \tag{3.46}$$

Aplicando a identidade (3.45) a (3.46), obtemos a seguinte forma recursiva de cálculo de $\mathbf{R}(t)$,

$$\mathbf{R}(t) = \mathbf{R}(t-1) - \mathbf{R}(t-1)\boldsymbol{\Psi}(t)\left(\mathbf{I} + \boldsymbol{\Psi}^T(t)\mathbf{R}(t-1)\boldsymbol{\Psi}(t)\right)^{-1}\boldsymbol{\Psi}^T(t)\mathbf{R}(t-1). \tag{3.47}$$

Com esta definição recursiva de $\mathbf{R}(t)$, o vetor $\mathbf{K}(t)$ pode também ser reescrito de forma a

não depender de $\mathbf{R}(t)$. Substituindo a equação (3.47) em (3.43), obtém-se:

$$\begin{aligned}
\mathbf{K}(t) =&\mathbf{R}(t)\boldsymbol{\Psi}(t)\\
=&\mathbf{R}(t-1)\boldsymbol{\Psi}(t) - \mathbf{R}(t-1)\boldsymbol{\Psi}(t)\left(\mathbf{I}+\boldsymbol{\Psi}^T(t)\mathbf{R}(t-1)\boldsymbol{\Psi}(t)\right)^{-1}\boldsymbol{\Psi}^T(t)\mathbf{R}(t-1)\boldsymbol{\Psi}(t)\\
=&\mathbf{R}(t-1)\boldsymbol{\Psi}(t)\left(\mathbf{I}+\boldsymbol{\Psi}^T(t)\mathbf{R}(t-1)\boldsymbol{\Psi}(t)\right)^{-1}\left(\mathbf{I}+\boldsymbol{\Psi}^T(t)\mathbf{R}(t-1)\boldsymbol{\Psi}(t)\right)\\
&-\mathbf{R}(t-1)\boldsymbol{\Psi}(t)\left(\mathbf{I}+\boldsymbol{\Psi}^T(t)\mathbf{R}(t-1)\boldsymbol{\Psi}(t)\right)^{-1}\boldsymbol{\Psi}^T(t)\mathbf{R}(t-1)\boldsymbol{\Psi}(t)\\
=&\mathbf{R}(t-1)\boldsymbol{\Psi}(t)\left(\mathbf{I}+\boldsymbol{\Psi}^T(t)\mathbf{R}(t-1)\boldsymbol{\Psi}(t)\right)^{-1}\\
&\qquad\left(\mathbf{I}+\boldsymbol{\Psi}^T(t)\mathbf{R}(t-1)\boldsymbol{\Psi}(t)-\boldsymbol{\Psi}^T(t)\mathbf{R}(t-1)\boldsymbol{\Psi}(t)\right)\\
=&\mathbf{R}(t-1)\boldsymbol{\Psi}(t)\left(\mathbf{I}+\boldsymbol{\Psi}^T(t)\mathbf{R}(t-1)\boldsymbol{\Psi}(t)\right)^{-1}. \qquad (3.48)
\end{aligned}$$

O algoritmo recursivo de estimação dos parâmetros $\hat{\boldsymbol{\Theta}}$ do modelo pode agora ser resumido nas seguintes equações aqui repetidas para referência,

$$\hat{\boldsymbol{\Theta}}(t) =\hat{\boldsymbol{\Theta}}(t-1) + \mathbf{K}(t)\left(y(t) - \boldsymbol{\Psi}^T(t)\hat{\boldsymbol{\Theta}}(t-1)\right), \qquad (3.49)$$

$$\mathbf{R}(t) =\mathbf{R}(t-1) - \mathbf{K}(t)\boldsymbol{\Psi}^T(t)\mathbf{R}(t-1), \qquad (3.50)$$

$$\mathbf{K}(t) =\mathbf{R}(t-1)\boldsymbol{\Psi}(t)\left(\mathbf{I}+\boldsymbol{\Psi}^T(t)\mathbf{R}(t-1)\boldsymbol{\Psi}(t)\right)^{-1}. \qquad (3.51)$$

Como as equações anteriores são recursivas é necessário definir quais são as suas condições iniciais, ou seja, é necessário definir $\hat{\boldsymbol{\Theta}}(t_0)$ e $\mathbf{R}(t_0)$. Recordando a definição de $\mathbf{R}(t)$ (3.38), se definir-mos uma sequência de t_s observações (3.29), $\boldsymbol{\Psi}(1)$, $\boldsymbol{\Psi}(1)$, ..., $\boldsymbol{\Psi}(t_s)$ tal que a matriz das covariâncias $\boldsymbol{\Phi}^T(t_s)\boldsymbol{\Phi}(t_s)$ seja invertível, é possível definir $\mathbf{R}(t_0)$ e $\hat{\boldsymbol{\Theta}}(t_0)$,

$$\mathbf{R}(t_0) =(\boldsymbol{\Phi}^T(t_s)\boldsymbol{\Phi}(t_s))^{-1}, \qquad (3.52)$$

$$\hat{\boldsymbol{\Theta}}(t_0) =\mathbf{R}(t_0)\boldsymbol{\Phi}^T(t_0)\mathbf{y}(t_0). \qquad (3.53)$$

A equação (3.53) é obtida substituindo (3.52) em (3.37). Para $t > t_0$, são utilizadas as equações recursivas definidas anteriormente (3.49), (3.50), (3.51).

Dada a dificuldade em determinar eficazmente o valor de t_s a utilizar de forma a garantir a invertibilidade de $\boldsymbol{\Phi}^T(t_s)\boldsymbol{\Phi}(t_s)$, podemos em alternativa assumir que não existe nenhum conhecimento inicial acerca da planta e utilizar a seguinte inicialização [1]:

$$\mathbf{R}(t_0) =\delta\mathbf{I}, \qquad (3.54)$$

$$\hat{\boldsymbol{\Theta}}(t_0) =\mathbf{0}, \qquad (3.55)$$

onde $\mathbf{I}$ representa a matriz identidade e δ representa um escalar.

3.2 Controlo Preditivo

Nesta secção é desenvolvido um controlador GPC. São adotadas algumas simplificações [19] com o fim de tonar o controlador computacionalmente menos exigente.

3.2.1 Modelo do Processo

Considere-se o seguinte modelo,

$$a(z^{-1})y(k) = z^{-d}b(z^{-1})u(k-1) + \epsilon(k), \tag{3.56}$$

onde, $u(k) \in \mathbb{R}$ representa a entrada, $y(k) \in \mathbb{R}$ representa a saída e $\epsilon(k) \in \mathbb{R}$ representa o ruído que afeta a saída. O valor $d \in \mathbb{N}$ representa o atraso de transporte entre a entrada e a saída. Os polinómios a e b tem respetivamente ordem n_y e n_u:

$$a(z^{-1}) = 1 + a_1 z^{-1} + \ldots + a_{n_y} z^{-n_y}, \tag{3.57}$$

$$b(z^{-1}) = b_0 + b_1 z^{-1} + \ldots + b_{n_u} z^{-n_u}. \tag{3.58}$$

Na literatura [3], [17], é comum o ruído ser modelado por um processo do tipo média móvel,

$$\epsilon(k) = c(z^{-1})\xi(k), \tag{3.59}$$

onde,

$$c(z^{-1}) = 1 + c_1 z^{-1} + \ldots + c_{n_\xi} z^{-n_\xi}, \tag{3.60}$$

e $\xi(k)$ representa um sinal de ruído branco com média nula.

No entanto, se as perturbações forem não estacionárias, por exemplo, se contiverem degraus ou se o seu valor médio variar com o tempo, é utilizado o seguinte modelo:

$$\epsilon(k) = \frac{c(z^{-1})}{\Delta}\xi(k), \tag{3.61}$$

onde Δ é o operador diferencial $1 - z^{-1}$. Nos desenvolvimentos seguintes é assumido por simplicidade que $c(z^{-1}) = 1$. Desta forma é utilizado o seguinte modelo de processo:

$$a(z^{-1})y(k) = z^{-d}b(z^{-1})u(k-1) + \frac{1}{\Delta}\xi(k). \tag{3.62}$$

3.2.2 Predição da Saída

O algoritmo de controlo GPC baseia-se no objetivo de gerar uma lei de comando que minimize uma série do tipo,

$$[\hat{y}(k+p|k) - r(k+p)]^2 + [\hat{y}(k+p+1|k) - r(k+p+1)]^2 + \ldots, \tag{3.63}$$

ou seja, regular a saída futura da planta de forma a convergir para uma trajetória r desejada. Como se pretende controlar um processo cuja resposta apenas se irá refletir no futuro, é necessário encontrar a melhor estimativa possível para os valores futuros da saída da planta. Surge então o problema de estimar a saída de um processo com um modelo do tipo (3.62) para instantes futuros.

Como ponto de partida pode-se reescrever a equação (3.62) para instantes de tempo futuros ($p > 0$),

$$y(k+p) = \frac{b(z^{-1})}{a(z^{-1})}u(k+p-d-1) + \frac{1}{\Delta a(z^{-1})}\xi(k+p). \tag{3.64}$$

Como se observa em (3.64), para estimar a saída do processo y em instantes de tempo futuros, é necessário determinar a melhor estimativa da componente de ruído ξ. É assumido que ξ representa ruído gaussiano com média nula, logo a sua melhor estimativa é nula. Assim, podemos isolar no último membro de (3.64) os termos que dependem de valores futuros de ξ e substitui-los pela sua melhor estimativa que é 0, ou seja, $\xi(k+p) = 0$ para $p > 0$.

Efetuando a divisão polinomial do último termo de (3.64) que multiplica pelo sinal de ruído $\xi(k)$, podemos obter a fatorização (3.65),

$$\frac{1}{\Delta a(z^{-1})} = e_p(z^{-1}) + z^{-p}\frac{f_p(z^{-1})}{\Delta a(z^{-1})}, \tag{3.65}$$

de onde resultam o polinómio e_p que representa o resultado da operação de divisão de 1 por $\Delta a(z^{-1})$ em (3.64), e o polinómio $z^{-p}f_p$ que representa o resto dessa divisão,

$$e_p(z^{-1}) = e_{0,p} + e_{1,p}z^{-1} + \ldots + e_{p-1,p}z^{-(p-1)}, \tag{3.66}$$

$$f_p(z^{-1}) = f_{0,p} + f_{1,p}z^{-1} + \ldots + f_{n_y,p}z^{-n_y}. \tag{3.67}$$

A equação (3.65) é uma equação diofantina e a sua solução (equações (3.66) e (3.67)) é derivada no anexo A.2.

Se multiplicarmos (3.62) por $\Delta z^p e_p(z^{-1})$, obtemos,

$$\Delta z^p e_p(z^{-1})a(z^{-1})y(k) = z^{p-d}e_p(z^{-1})b(z^{-1})\Delta u(k-1) + z^p e_p(z^{-1})\xi(k). \tag{3.68}$$

Ao multiplicar (3.65) por $\Delta a(z^{-1})$ e reordenando os seus termos obtemos

$$\Delta e_p(z^{-1})a(z^{-1}) = 1 - z^{-p}f_p(z^{-1}). \tag{3.69}$$

Substituindo (3.69) em (3.68), podemos obter o seguinte modelo de predição de p passos no futuro:

$$z^p(1 - z^{-p}f_p(z^{-1}))y(k) = z^{p-d}e_p(z^{-1})b(z^{-1})\Delta u(k-1) + z^p e_p(z^{-1})\xi(k). \tag{3.70}$$

Simplificando de forma a evidenciar a saída do modelo p mostras no futuro $y(k+p)$ obtém-se:

$$y(k+p) = g_p(z^{-1})\Delta u(k+p-d-1) + f_p(z^{-1})y(k) + e_p(z^{-1})\xi(k+p). \tag{3.71}$$

Em (3.71), o polinómio $g_p(z^{-1})$ é dado por $g_p(z^{-1}) = e_p(z^{-1})b(z^{-1})$.

Como o polinómio $e_p(z^{-1})$ tem grau $p-1$ (3.66), a sua multiplicação pelo sinal de ruído resulta em que $e_p(z^{-1})\xi(k+p)$ toma apenas termos futuros do sinal de ruído, ou seja, valores de $\xi(k+p)$ para $p \geq 1$. Como foi já referido, a melhor estimativa de $\xi(k+p)$ é 0 dado que se trata de ruído branco com média nula, assim chegamos à seguinte versão do modelo de predição de (3.62) estimado com dados disponíveis até ao instante k:

$$\hat{y}(k+p|k) = g_p(z^{-1})\Delta u(k+p-d-1) + f_p(z^{-1})y(k). \tag{3.72}$$

3.2.3 Lei de Controlo Preditivo

Seguindo a análise de [15], considere-se a seguinte função de custo:

$$J(k) = \sum_{p=d+1}^{N_p} \left(\hat{y}(k+p|k) - \phi_p r(k+p)\right)^2 + \sum_{p=d}^{d+N_u-1} \Big(q(z^{-1})\Delta u(k+p-d|k)\Big)^2, \tag{3.73}$$

onde ϕ_p é uma sequência de ganhos $(\phi_{d+1}, \cdots, \phi_{N_p})$, $\Delta = 1 - z^{-1}$ é o operador de diferença e $q(z^{-1})$ é o polinómio $q(z^{-1}) = q_0 + q_1 z^{-1} + \cdots + q_{n_u+d}z^{-n_u-d}$. N_p e N_u são os horizontes de predição da saída e da ação de controlo, respetivamente. A lei de controlo é obtida encontrando uma solução para o sinal de comando u que minimiza a função de custo.

Como uma ação de controlo no instante de tempo k terá resultado na saída do processo após $d+1$ amostras, apenas é necessário considerar os valores da saída a partir do instante $d+1$. No entanto se o atraso de transporte d não for conhecido, este assume-se como nulo [17]. Os parâmetros N_p e N_u são chamados respetivamente de horizonte de predição e horizonte de controlo. Em [3] estes parâmetros são definidos como $N_p = d+N$ e $N_u = N$, no entanto em outros textos tais como [17], [15], considera-se $N_u < N$ com o objetivo de

reduzir o esforço computacional. Neste capítulo é utilizada a segunda abordagem com $N_u = 1$. O polinómio $q(z^{-1})$ é interpretado como uma sequência de ganhos da ação de controlo. É importante também colocar o esforço de comando $u(k)$ na função de custo. De outro modo a minimização do erro da saída poderia resultar em esforços de controlo demasiado elevados caso o esforço de controlo não fosse também penalizado na função de custo. Com o objetivo de minimizar a função de custo (3.73), considere-se os desenvolvimentos seguintes.

Ao expandir os polinómios nos dois termos do segundo membro de (3.72), estes tem a forma dada por (3.74) (a ordem do polinómio $g_p(z^{-1}) = e_p(z^{-1})b(z^{-1})$, é n_u+p-1). Assim podemos separar o primeiro termo do segundo membro de (3.73) em termos passados de Δu (na forma $\Delta u(k+p)$, com $p < 0$) e em termos futuros de Δu (na forma $\Delta u(k+p)$, com $p \geq 0$):

$$\begin{aligned} g_p(z^{-1})\Delta u(k+p-d-1) =& g_{0,p}\Delta u(k+p-d-1) + g_{1,p}\Delta u(k+p-d-2) + \\ & \cdots + g_{n_u+p-1,p}\Delta u(k-n_u-d), \\ f_p(z^{-1})y(k) =& f_{0,p}y(k) + f_{1,p}y(k-1) + \cdots + f_{n_y,p}y(k-n_y). \end{aligned} \tag{3.74}$$

Podemos escrever $\hat{y}(k+p|k)$ sob a forma matricial se agregarmos todos os seus valores de $p = d+1$ até $p = d+N$ na seguinte forma. Defina-se o vetor coluna $\hat{\mathbf{y}}$ (3.75) que contém todos os elementos de $\hat{y}(k+p|k)$ para $p = d+1, \ldots, d+N$. Com a separação dos termos de Δu em termos passados e futuros, são criados também os vetores coluna $\Delta\mathbf{u}_1$ (3.76) e $\Delta\mathbf{u}$ (3.77) respetivamente.

$$\hat{\mathbf{y}} = [\hat{y}(k+d+1), \hat{y}(k+d+2), \ldots, \hat{y}(k+d+N)]^T, \tag{3.75}$$

$$\Delta\mathbf{u} = [\Delta u(k), \Delta u(k+1), \ldots, \Delta u(k+N-1)]^T, \tag{3.76}$$

$$\Delta\mathbf{u}_1 = [\Delta u(k-1), \Delta u(k-2), \ldots, \Delta u(k-n_u-d)]^T, \tag{3.77}$$

$$\mathbf{y} = [y(k), y(k-1), \ldots, y(k-n_y)]^T. \tag{3.78}$$

Se para cada valor de $p = d+1, \ldots, d+N$, os coeficientes do polinómio $g_p(z^{-1})$ em (3.74) forem agrupados em coeficientes que multiplicam por valores do tipo $\Delta u(k+p)$, com $p < 0$, e por coeficientes que multiplicam valores do tipo $\Delta u(k+p)$, com $p \geq 0$, então podemos respetivamente definir as matrizes $\mathbf{L}$ e $\mathbf{G}$, onde em cada linha de ambas matrizes estão agrupados os coeficientes de $g_p(z^{-1})$ correspondes a um valor particular de

p, conforme expresso na seguinte equação.

$$\hat{\mathbf{y}} = \underbrace{\begin{bmatrix} g_{0,d+1} & 0 & \cdots & 0 \\ g_{1,d+2} & g_{0,d+2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ g_{N-1,d+N} & g_{N-2,d+N} & \cdots & g_{0,d+N} \end{bmatrix}}_{\mathbf{G}} \Delta\mathbf{u} \tag{3.79}$$

$$+ \underbrace{\begin{bmatrix} g_{1,d+1} & g_{2,d+1} & \cdots & g_{n_u+d,d+1} \\ g_{2,d+2} & g_{3,d+2} & \cdots & g_{n_u+d+1,d+2} \\ \vdots & \vdots & \ddots & \vdots \\ g_{N,d+N} & g_{N+1,d+N} & \cdots & g_{n_u+d+N-1,d+N} \end{bmatrix}}_{\mathbf{L}} \Delta\mathbf{u}_1 + \underbrace{\begin{bmatrix} f_{0,d+1} & \cdots & f_{n_y,d+1} \\ f_{0,d+2} & \cdots & f_{n_y,d+2} \\ \vdots & \ddots & \vdots \\ f_{0,d+N} & \cdots & f_{n_y,d+N} \end{bmatrix}}_{\mathbf{F}} \mathbf{y},$$

ou de forma mais compacta:

$$\hat{\mathbf{y}} = \mathbf{G}\Delta\mathbf{u} + \mathbf{L}\Delta\mathbf{u}_1 + \mathbf{F}\mathbf{y}, \tag{3.80}$$

onde as matrizes $\mathbf{G}$, $\mathbf{L}$ e $\mathbf{F}$ foram definidas em (3.79).

Olhando para o primeiro termo do segundo membro da função de custo (3.73), verifica-se que este penaliza o erro de seguimento entre $\hat{y}$ e a referência desejada r sob a forma da soma das diferenças,

$$\left[\hat{y}(k+d+1|k) - \phi_{d+1} r(k+d+1)\right]^2 + \ldots + \left[\hat{y}(k+d+N|k) - \phi_{d+N} r(k+d+N)\right]^2.$$

Esta soma de diferenças pode ser também representada na forma matricial se definirmos o vetor $\mathbf{r}$ (3.81) que contém os valores de r,

$$\mathbf{r} = \left[r(k+d+1), r(k+d+2), \ldots, r(k+d+N)\right]^T, \tag{3.81}$$

e a matriz $\mathbf{\Phi}$ (3.82) que contém na sua diagonal os ganhos ϕ_p,

$$\mathbf{\Phi} = \begin{bmatrix} \phi_{d+1} & 0 & \cdots & 0 \\ 0 & \phi_{d+2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \phi_{d+N} \end{bmatrix}. \tag{3.82}$$

Se subtrairmos $\mathbf{\Phi r}$ de ambos os membros de (3.80), obtemos (3.83) que representa o erro

de seguimento que se pretende minimizar sob a forma matricial,

$$\hat{\mathbf{y}} - \mathbf{\Phi r} = \mathbf{G}\Delta\mathbf{u} + \mathbf{L}\Delta\mathbf{u}_1 + \mathbf{F}\mathbf{y} - \mathbf{\Phi r}. \tag{3.83}$$

O segundo termo de (3.73) contém os elementos do sinal de comando. Para $N_u = 1$, este termo fica com a forma $(q(z^{-1})\Delta u(k|k))$ e a sua representação matricial é obtida rescrevendo sob a forma da multiplicação dos seguintes dois vetores $\mathbf{Q}$ e $\Delta\mathbf{u}$:

$$\Delta\mathbf{u} = \begin{bmatrix} \Delta\mathbf{u} \\ \Delta\mathbf{u}_1 \end{bmatrix}, \mathbf{Q} = \begin{bmatrix} \mathbf{q} \\ \mathbf{q}_1 \end{bmatrix}, \tag{3.84}$$

onde os vetores $\Delta\mathbf{u}$ e $\Delta\mathbf{u}_1$ estão definidos em (3.76) e (3.77), respetivamente. Os coeficientes do polinómio $q(z^{-1})$ que multiplicam por elementos de $\Delta u(k+p)$ para $p \geq 0$ estão agrupados no vetor $\mathbf{q}$ (3.85) e os restantes estão no vetor $\mathbf{q}_1$ (3.86),

$$\mathbf{q} = [q_0, q_1, \ldots, q_{N_u-1}]^T, \tag{3.85}$$

$$\mathbf{q}_1 = [q_{-1}, q_{-2}, \ldots, q_{-n_u-d}]^T. \tag{3.86}$$

Rescrevendo a função de custo (3.73) na forma matricial utilizando os resultados anteriores, obtemos (3.87),

$$\begin{aligned} J(k) = &\left(\mathbf{F}\mathbf{y} + \mathbf{G}\Delta\mathbf{u} + \mathbf{L}\Delta\mathbf{u}_1 - \mathbf{\Phi r}\right)^T \left(\mathbf{F}\mathbf{y} + \mathbf{G}\Delta\mathbf{u} + \mathbf{L}\Delta\mathbf{u}_1 - \mathbf{\Phi r}\right) + \\ &\left(\mathbf{Q}^T\Delta\mathbf{u}\right)^T \left(\mathbf{Q}^T\Delta\mathbf{u}\right). \end{aligned} \tag{3.87}$$

Ao realizar a multiplicação dos membros da equação (3.87) e considerando a descrição dos vetores em (3.84), obtém-se:

$$\begin{aligned} J(k) = &\left(\Delta\mathbf{u}^T\mathbf{G}^T + \Delta\mathbf{u}_1^T\mathbf{L}^T + \mathbf{y}^T\mathbf{F}^T - \mathbf{r}^T\mathbf{\Phi}^T\right)\left(\mathbf{G}\Delta\mathbf{u} + \mathbf{L}\Delta\mathbf{u}_1 + \mathbf{F}\mathbf{y} - \mathbf{\Phi r}\right) + \\ &\left(\Delta\mathbf{u}^T\mathbf{Q}\right)\left(\mathbf{Q}^T\Delta\mathbf{u}\right) \\ = &\Delta\mathbf{u}^T\left(\mathbf{G}^T\mathbf{G} + \mathbf{q}\,\mathbf{q}^T\right)\Delta\mathbf{u} + \Delta\mathbf{u}_1^T\left(\mathbf{L}^T\mathbf{L} + \mathbf{q}_1\,\mathbf{q}_1^T\right)\Delta\mathbf{u}_1 + \\ &\left(2\Delta\mathbf{u}_1^T\mathbf{L}^T\mathbf{G} + 2\mathbf{y}^T\mathbf{F}^T\mathbf{G} - 2\mathbf{r}^T\mathbf{\Phi}^T\mathbf{G} + 2\Delta\mathbf{u}_1^T\mathbf{q}_1\,\mathbf{q}^T\right)\Delta\mathbf{u} + \\ &2\Delta\mathbf{u}_1^T\mathbf{L}^T\mathbf{F}\mathbf{y} - 2\mathbf{r}^T\mathbf{\Phi}^T\mathbf{F}\mathbf{y} - 2\mathbf{r}^T\mathbf{\Phi}^T\mathbf{L}\Delta\mathbf{u}_1 + \mathbf{r}^T\mathbf{\Phi}^T\mathbf{\Phi r} + \mathbf{y}^T\mathbf{F}^T\mathbf{F}\mathbf{y}. \end{aligned} \tag{3.88}$$

A ação de controlo futura ótima que minimiza a função de custo é obtida anulando a derivada parcial de $J(k)$ em relação a $\Delta\mathbf{u}$, ou seja:

$$\frac{\partial J(k)}{\partial(\Delta\mathbf{u})} = 0. \tag{3.89}$$

Assim, substituindo (3.88) em (3.89) e resolvendo a equação (3.89) chega-se ao seguinte resultado:

$$2\left(\mathbf{G}^T\mathbf{G} + \mathbf{q}\ \mathbf{q}^T\right)\Delta\mathbf{u} + 2\Delta\mathbf{u}_1^T\mathbf{L}^T\mathbf{G} + 2\mathbf{y}^T\mathbf{F}^T\mathbf{G} - 2\mathbf{r}^T\mathbf{\Phi}^T\mathbf{G} + 2\Delta\mathbf{u}_1^T\mathbf{q}_1\ \mathbf{q}^T = 0, \quad (3.90)$$

ou reordenando os seus termos, obtém-se (3.91),

$$\left(\mathbf{G}^T\mathbf{G} + \mathbf{q}\ \mathbf{q}^T\right)\Delta\mathbf{u} = \mathbf{G}^T\left(\mathbf{\Phi}\mathbf{r} - \mathbf{F}\mathbf{y}\right) - \left(\mathbf{G}^T\mathbf{L} + \mathbf{q}\ \mathbf{q}_1^T\right)\Delta\mathbf{u}_1. \quad (3.91)$$

Analisando a equação (3.91), pode-se verificar que o último termo pode ser anulado através do ajuste do vetor de ganhos $\mathbf{q}_1$, gerando assim a lei de controlo simplificada de [15]. Este resultado é obtido escolhendo os valores do vetor de ganhos $\mathbf{q}_1$ de forma a satisfazer a seguinte restrição (3.92),

$$\mathbf{G}^T\mathbf{L} = -\mathbf{q}\ \mathbf{q}_1^T. \quad (3.92)$$

A lei de controlo é obtida substituindo (3.92) em (3.91) e reescrevendo a equação de forma a colocar $\Delta\mathbf{u}$ em evidência (3.93),

$$\Delta\mathbf{u} = \left(\mathbf{G}^T\mathbf{G} + \mathbf{q}\ \mathbf{q}^T\right)^{-1}\mathbf{G}^T\left(\mathbf{\Phi}\mathbf{r} - \mathbf{F}\mathbf{y}\right). \quad (3.93)$$

De forma a reduzir a carga computacional, o horizonte de controlo N_u é definido com $N_u = 1$. Isto significa que apenas a ação de controlo $\Delta u(k)$ é não nula, sendo as restantes $\Delta u(k+1) = 0, \ldots, \Delta u(k+N-1) = 0$. Como desta forma o vetor $\Delta\mathbf{u}$ (3.76) passa apenas a ter o seu primeiro elemento não nulo, a equação (3.93) pode ser simplificada.

Redefina-se a matriz $\mathbf{G}$ (3.79) apenas à sua primeira coluna,

$$\mathbf{G} = \left[g_{0,d+1},\ g_{1,d+2},\ \ldots,\ g_{N-1,d+N}\right]^T, \quad (3.94)$$

e o vetor $\Delta\mathbf{u}$ a escalar $\Delta\mathbf{u} = \Delta u(k)$. O vetor $\mathbf{q}$ (3.85) é reduzido também a escalar $\mathbf{q} = q_0$. Desta forma a inversão matricial presente em (3.93) torna-se na inversão de um escalar apenas. A lei de controlo (3.93) pode ser então reescrita na seguinte forma final:

$$u(k) = \Delta u(k) + u(k-1) = \mathbf{K}\left(\mathbf{\Phi}\mathbf{r} - \mathbf{F}\mathbf{y}\right) + u(k-1), \quad (3.95)$$

onde $\mathbf{K} = \left(\mathbf{G}^T\mathbf{G} + \lambda\right)^{-1}\mathbf{G}^T$ e $\lambda = q_0^2$. A prova de estabilidade deste controlador é feita em [15].

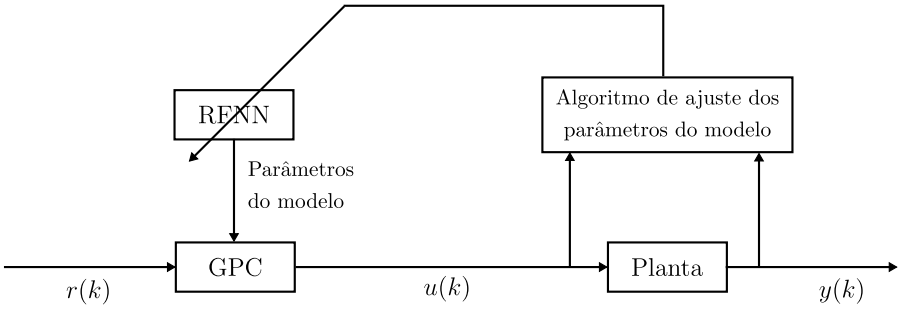


Figura 3.2: Estrutura do Controlador Preditivo com Rede Neuronal Difusa Recorrente.

3.3 Controlador Preditivo com Rede Neuronal Difusa Recorrente

Depois de consolidados os dois componentes principais do controlador nas secções anteriores deste capítulo, o modelo gerado pela RFNN e a lei de controlo preditivo, é necessário definir como estes interagem e estabelecer a metodologia necessária para aplicar este controlador no controlo de um processo em tempo real.

A Figura 3.2 representa o controlador GPC em conjunto com a RFNN, integrados na malha de controlo com o processo a controlar.

O controlador preditivo desenvolvido na Secção 3.2 deriva as suas expressões com base num modelo linear da forma (3.56). No entanto o modelo gerado pela RFNN (3.12) na Secção 3.1 tem uma forma diferente. Utilizando o desenvolvimento do modelo (3.12) feito em (3.25), é possível identificar os coeficientes a_i e b_i com a seguinte análise,

$$\begin{aligned}
\hat{y}(k) =& a_{1,1}\omega_1(k)y(k-1) + \ldots + a_{n_y,1}\omega_1(k)y(k-n_y) \\
&+ b_{0,1}\omega_1(k)u(k-d-1) + \ldots + b_{n_u,1}\omega_1(k)u(k-d-n_u-1) \\
&+ \ldots + a_{1,L}\omega_L(k)y(k-1) + \ldots + a_{n_y,L}\omega_L(k)y(k-n_y) \\
&+ b_{0,L}\omega_L(k)u(k-d-1) + \ldots + b_{n_u,L}\omega_L(k)u(k-d-n_u-1) \\
=& \underbrace{\left(\sum_{j=1}^{L} a_{1,j}\omega_j(k)\right)}_{\bar{a}_1} y(k-1) + \ldots + \underbrace{\left(\sum_{j=1}^{L} a_{n_y,j}\omega_j(k)\right)}_{\bar{a}_{n_y}} y(k-n_y) \\
&+ \underbrace{\left(\sum_{j=1}^{L} b_{o,j}\omega_j(k)\right)}_{\bar{b}_0} u(k-d-1) + \ldots + \underbrace{\left(\sum_{j=1}^{L} b_{n_u,j}\omega_j(k)\right)}_{\bar{b}_{n_u}} u(k-d-n_u-1)
\end{aligned}$$

$$= \left(\bar{a}_1 z^{-1} + \ldots + \bar{a}_{n_y} z^{-n_y}\right) y(k) + z^{-d} \left(\bar{b}_0 + \ldots + \bar{b}_{n_u} z^{-n_u}\right) u(k-1). \tag{3.96}$$

A equação (3.96) constitui uma forma alternativa do modelo não-linear da RFNN (3.12), (3.25). A partir do modelo da RFNN (3.12), (3.25), (3.96), pode-se extrair um modelo local/instantâneo linear (3.56), (3.57), (3.58), onde os parâmetros são dados por (3.97), (3.98). Especificamente, comparando o resultado final de (3.96) com o modelo (3.56) e os seus polinómios (3.57) e (3.58), podemos fazer as seguintes definições,

$$a_i = -\bar{a}_i = -\sum_{j=1}^{L} a_{ij}\omega_j(k), \tag{3.97}$$

$$b_i = \bar{b}_i = \sum_{j=1}^{L} b_{ij}\omega_j(k). \tag{3.98}$$

A sequência de passos necessários para aplicar o controlador desenvolvido neste capítulo a um sistema de tempo real é a seguinte:

Passo 1 Definir os parâmetros do modelo da RFNN n_y, n_u, d, L, e do controlador preditivo N_p e λ.

Passo 2 Inicializar as regras difusas, definindo o valor inicial para as antecedentes m_{ij}, σ_{ij}, ϑ_{ij}, e para as consequentes $\hat{\boldsymbol{\Theta}}(0)$. É necessário inicializar também a matriz de covariâncias $\mathbf{R}(0)$ utilizada no algoritmo de aprendizagem das consequentes.

Passo 3 Obter o valor da saída do processo y e do sinal de referência r.

Passo 4 Atualizar os parâmetros do modelo da RFNN com (3.16), (3.17), (3.18), (3.49), (3.50), (3.51).

Passo 5 Utilizar (3.97) e (3.98) para determinar os coeficientes de (3.57) e (3.58), e consequentemente determinar o próximo incremento na ação de controlo Δu com (3.95).

Passo 6 Aplicar o incremento Δu como ação controlo aplicada sobre o processo.

Passo 7 Repetir a sequência de passos 3 a 6 com um período de amostragem T.

O controlador desenvolvido neste capítulo foi proposto originalmente em [15].

Capítulo 4

Controlo Preditivo com Rede Neuronal Difusa Recorrente (II)

Neste capítulo é proposto um controlador preditivo baseado no modelo do processo.

O controlador aqui proposto, à semelhança do que foi desenvolvido no Capítulo 3, utiliza o modelo gerado por uma RFNN e o controlo é baseado no algoritmo GPC original [17], [3]. O desenvolvimento aqui proposto, consiste em uma nova lei de adaptação dos consequentes das regras da RFNN que é derivada diretamente pelos critérios de estabilidade de Lyapunov, provando assim em simultâneo a estabilidade do sistema de malha fechada. No seguinte de desenvolvimento, também são obtidos resultados que permitem provar que o erro de seguimento é limitado. Este capítulo teve como resultado uma publicação em conferência [20] (ver anexo A.1).

Neste capítulo são em primeiro lugar expostos de forma breve alguns conceitos do algoritmo de controlo GPC original de Clarke [17] necessários nos pontos seguintes. O resto do capítulo é dedicado a desenvolver a lei de controlo baseada em GPC e a lei de adaptação baseada em teoria de estabilidade de Lyapunov.

4.1 Controlo Preditivo

Nas secções 3.2.1 e 3.2.2 do Capítulo 3 foram desenvolvidos alguns resultados de base do algoritmo GPC. Esta secção vai utilizar esses desenvolvimentos para derivar a lei de controlo GPC utilizada neste capítulo.

Considere-se a função de custo (4.1),

$$J(k) = \sum_{p=d+1}^{N_p} \left(\hat{y}(k+p|k) - \phi_p r(k+p)\right)^2 + \sum_{p=1}^{N_u} \lambda_p \left(\Delta u(k+p-1|k)\right)^2. \qquad (4.1)$$

Esta função de custo é idêntica a (3.73), com a exceção do segundo membro onde λ_p é uma sequência de ganhos aplicados ao sinal de comando.

Considere-se a seguinte representação matricial do modelo de predição da saída do processo (3.72) para valores de $p = d + 1$ até ao horizonte de predição $p = N_p = d + N$, e com o horizonte de controlo $N_u = N$,

$$\hat{\mathbf{y}} = \mathbf{G}\Delta\mathbf{u} + \mathbf{F}(z^{-1})y(k) + \mathbf{L}(z^{-1})\Delta u(k-1), \tag{4.2}$$

onde $\hat{\mathbf{y}}$, $\mathbf{G}$, $\Delta\mathbf{u}$, $\mathbf{F}$ e $\mathbf{L}$ são definidos da seguinte forma,

$$\hat{\mathbf{y}} = \begin{bmatrix} \hat{y}(k+d+1) \\ \hat{y}(k+d+2) \\ \vdots \\ \hat{y}(k+d+N) \end{bmatrix}, \quad \Delta\mathbf{u} = \begin{bmatrix} \Delta u(k) \\ \Delta u(k+1) \\ \vdots \\ \Delta u(k+N-1) \end{bmatrix}, \tag{4.3}$$

$$\mathbf{G} = \begin{bmatrix} g_{0,d+1} & 0 & \cdots & 0 \\ g_{1,d+2} & g_{0,d+2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ g_{N-1,d+N} & g_{N-2,d+N} & \cdots & g_{0,d+N} \end{bmatrix}, \quad \mathbf{F}(z^{-1}) = \begin{bmatrix} f_{d+1}(z^{-1}) \\ f_{d+2}(z^{-1}) \\ \vdots \\ f_{d+N}(z^{-1}) \end{bmatrix}, \tag{4.4}$$

$$\mathbf{L}(z^{-1}) = \begin{bmatrix} \left(g_{d+1}(z^{-1}) - g_{0,d+1}\right) z \\ \left(g_{d+2}(z^{-1}) - g_{0,d+2} - g_{1,d+2}z^{-1}\right) z^2 \\ \vdots \\ \left(g_{d+N}(z^{-1}) - g_{0,d+N} - \cdots - g_{N-2,d+N}z^{-(N-2)} - g_{N-1,d+N}z^{-(N-1)}\right) z^N \end{bmatrix}. \tag{4.5}$$

Utilizando a equação (4.2) em conjunto com (3.82) e (3.81), é possível reescrever a função de custo (4.1) na seguinte forma matricial, de forma idêntica ao realizado na Secção 3.2.3,

$$\begin{aligned} J(k) = &\left[\mathbf{G}\Delta\mathbf{u} + \mathbf{F}(z^{-1})y(k) + \mathbf{L}(z^{-1})\Delta u(k-1) - \mathbf{\Phi r}\right]^T \\ &\left[\mathbf{G}\Delta\mathbf{u} + \mathbf{F}(z^{-1})y(k) + \mathbf{L}(z^{-1})\Delta u(k-1) - \mathbf{\Phi r}\right] + \lambda \left[\Delta\mathbf{u}\right]^T \left[\Delta\mathbf{u}\right]. \end{aligned} \tag{4.6}$$

De forma a simplificar a lei de comando, a sequência de ganhos λ_p é definida como sendo constante em (4.6), com valor λ.

A lei de controlo ótima é obtida minimizando a função de custo (4.6). Para este fim

é utilizada a derivada parcial de $J(k)$ em ordem a $\Delta\mathbf{u}$,

$$\frac{\partial J(k)}{\partial(\Delta\mathbf{u})} = 0. \tag{4.7}$$

Expandindo a equação (4.6) e substituindo o resultado em (4.7), obtém-se a seguinte equação,

$$\mathbf{G}^T\left(\mathbf{F}(z^{-1})y(k) + \mathbf{L}(z^{-1})\Delta u(k-1) - \mathbf{\Phi r}\right) + \left(\mathbf{G}^T\mathbf{G} + \lambda\mathbf{I}\right)\Delta\mathbf{u} = 0, \tag{4.8}$$

onde $\mathbf{I}$ é a matriz identidade. Colocando o vetor de ações de controlo futuras $\Delta\mathbf{u}$ em evidência em (4.8), obtém-se a lei de controlo (4.9),

$$\Delta\mathbf{u} = \left(\mathbf{G}^T\mathbf{G} + \lambda\mathbf{I}\right)^{-1}\mathbf{G}^T\left(\mathbf{\Phi r} - \mathbf{F}(z^{-1})y(k) - \mathbf{L}(z^{-1})\Delta u(k-1)\right). \tag{4.9}$$

O vetor de ações de controlo $\Delta\mathbf{u}$ (4.9) contém todas as ações de controlo para os instantes de tempo futuros dentro do horizonte de controlo N_u. No entanto, em cada período de amostragem apenas é aplicado ao processo sob controlo, o primeiro elemento de $\Delta\mathbf{u}$, $\Delta u(k)$. Obtém-se assim o seguinte resultado,

$$u(k) = \Delta u(k) + u(k-1) = \mathbf{K}\left(\mathbf{\Phi r} - \mathbf{F}(z^{-1})y(k) - \mathbf{L}(z^{-1})\Delta u(k-1)\right) + u(k-1). \tag{4.10}$$

onde $\mathbf{K} = [1\ 0\ 0\ \ldots\ 0]_{1\times N_u}\left(\mathbf{G}^T\mathbf{G} + \lambda\mathbf{I}\right)^{-1}\mathbf{G}^T$.

Nos desenvolvimentos anteriores foi assumido que o horizonte de controlo $N_u = N$, no entanto este pode também assumir valores $N_u < N$. Se for definido o horizonte de controlo com $N_u < N$, então no modelo (4.2) é adequado considerar que o vetor $\Delta\mathbf{u}$ (4.3) possui apenas os primeiros N_u elementos não nulos, e que a matriz $\mathbf{G}$ (4.4) possui as primeiras N_u colunas não nulas.

4.2 Lei de Controlo Preditivo

Na Secção 3.1.2 foi descrito o desenvolvimento da RFNN e a sua capacidade de representar sistemas não-lineares na forma (3.29), aqui reescrita em (4.11) para referência,

$$\hat{y}(k) = \hat{\mathbf{\Theta}}^T\mathbf{\Psi}(k). \tag{4.11}$$

Pressuposto 1. *Existe um vetor de parâmetros ótimos $\mathbf{\Theta}^*$, que resulta em o modelo produzido pela RFNN (3.29) ser uma representação perfeita do modelo não linear da planta (3.3) [21], excetuando o ruído $\xi(k)$.*

Considerando o Pressuposto 1, isto é, assumindo que o erro de modelação é desprezável, a planta real (3.3) pode ter uma representação do tipo (4.11), ou seja,

$$y^*(k) = (\boldsymbol{\Theta}^*)^T \boldsymbol{\Psi}(k), \tag{4.12}$$

onde $\boldsymbol{\Theta}^* = \left[(\boldsymbol{\theta}^{1*})^T, (\boldsymbol{\theta}^{2*})^T, \ldots, (\boldsymbol{\theta}^{L*})^T\right]^T$. É assumido que o vetor de parâmetros ótimos em (4.12) não é conhecido, desta forma um modelo aproximado para a planta pode ser definido como,

$$\hat{y}(k) = \hat{\boldsymbol{\Theta}}^T(k) \boldsymbol{\Psi}(k), \tag{4.13}$$

onde $\hat{\boldsymbol{\Theta}} = \left[(\hat{\boldsymbol{\theta}}^1)^T, (\hat{\boldsymbol{\theta}}^2)^T, \ldots, (\hat{\boldsymbol{\theta}}^L)^T\right]^T$ é um vetor de parâmetros que representa uma estimativa de $\boldsymbol{\Theta}^*$.

Pressuposto 2. *O vetor de parâmetros estimados do modelo $\hat{\boldsymbol{\Theta}}$, está restrito a uma região limitada $\Omega_{\boldsymbol{\Theta}}$, onde $m_{\boldsymbol{\Theta}}$ é uma constante positiva, ou seja,*

$$\Omega_{\boldsymbol{\Theta}} = \{\boldsymbol{\Theta} \in \mathbb{R}^n : \|\boldsymbol{\Theta}\| \leq m_{\boldsymbol{\Theta}}\}. \tag{4.14}$$

O erro de aproximação da RFNN, $d_y(k) = \hat{y}(k) - y^*(k) = \hat{\boldsymbol{\Theta}}^T(k)\boldsymbol{\Psi}(k) - (\boldsymbol{\Theta}^*)^T\boldsymbol{\Psi}(k) = \tilde{\boldsymbol{\Theta}}(k)\boldsymbol{\Psi}(k)$, onde $\tilde{\boldsymbol{\Theta}}(k) = \hat{\boldsymbol{\Theta}}(k) - \boldsymbol{\Theta}^*$, satisfaz a seguinte inequação, onde β é uma constante positiva,

$$\sup_{\mathbf{x}_e(k) \in \Omega_{\mathbf{x}_e}} |d_y(k)| \leq \beta, \tag{4.15}$$

onde se assume que $\mathbf{x}_e(k)$ (3.6) toma valores $\mathbf{x}_e(k) \in \Omega_{\mathbf{x}_e}$.

Para definir a arquitetura do controlador difuso preditivo proposto, considere-se a classe de sistemas dinâmicos não lineares em tempo discreto, modelados da seguinte forma:

$$\begin{aligned} x_1(k+1) &= x_2(k), \\ x_2(k+1) &= x_3(k), \\ &\vdots \\ x_{n-1}(k+1) &= x_n(k), \\ x_n(k+1) &= f[\mathbf{x}(k)] + g[\mathbf{x}(k)]u(k), \\ y(k+1) &= x_n(k+1), \end{aligned} \tag{4.16}$$

onde $\mathbf{x}(k) = [x_1(k), x_2(k), \ldots, x_n(k)]^T$ é o vetor de estado, $u \in \mathbb{R}$ e $y \in \mathbb{R}$ são a entrada e a saída da planta, respetivamente, e $f[\mathbf{x}(k)]$ e $g[\mathbf{x}(k)]$ são funções não lineares desconhecidas.

Pressuposto 3. *$|g(\mathbf{x}(k))| > \delta$, onde δ representa um numero real positivo pequeno. Isto implica que o grau relativo do modelo gerado pela RFNN e consequentemente, o grau*

relativo da planta é unitário [21].

Sem perda de generalidade, é assumido que $g[\mathbf{x}(k)] > 0$. De forma a simplificar as operações computacionais é utilizado $g[\mathbf{x}(k)] = g > 0$ como sendo uma constante.

Pressuposto 4. *A trajetória de referência $r(k)$ satisfaz,*

$$|r(k)| \leq U, \tag{4.17}$$

onde U é um limite conhecido [21].

Com o Pressuposto 2 pode-se assumir que o modelo (4.16) é invertível, e consequentemente definir a seguinte lei de controlo [21],

$$u_*(k) = \frac{1}{g}\left\{-f[\mathbf{x}(k)] + r(k+1) + \mathbf{k}^T\mathbf{e}(k)\right\}, \tag{4.18}$$

onde $\mathbf{e}(k)$ é o vetor (4.19) e $e(k)$ representa o erro de seguimento do modelo (4.20),

$$\mathbf{e}(k) = [e(k-n+1), \ldots, e(k-1), e(k)]^T, \tag{4.19}$$

$$e(k) = r(k) - y(k). \tag{4.20}$$

O vetor de coeficientes $\mathbf{k} = [k_n, \ldots, k_1]^T$ é definido tal que os zeros do polinómio,

$$k(z) = z^n + k_1 z^{n-1} + \ldots + k_n, \tag{4.21}$$

se encontrem dentro do círculo de raio unitário centrado na origem do plano z.

A lei de controlo (4.18) é ideal dado que se as funções $f[\mathbf{x}(k)]$ e $g[\mathbf{x}(k)]$ forem conhecidas, ao substituir $u_*(k)$ (4.18) no modelo do processo (4.16) em $u(k)$, o comportamento não linear do sistema é cancelado, resultando em

$$y(k+1) = r(k+1) + \mathbf{k}^T\mathbf{e}(k). \tag{4.22}$$

Substituindo (4.20) em (4.22), obtém-se a seguinte equação da dinâmica do erro do sistema,

$$e(k+1) = -\mathbf{k}^T\mathbf{e}(k). \tag{4.23}$$

Desta forma é obtido um sistema de malha fechada governado por (4.23) e que tem uma dinâmica de erro definida por $\mathbf{k}$.

Como as funções $f[\mathbf{x}(k)]$ e $g[\mathbf{x}(k)]$ não são normalmente conhecidas, é necessário escrever $e(k+1)$ de outra forma. Reescrevendo (4.18) colocando $r(k+1)$ em evidência, e

substituindo em (4.20), obtém-se,

$$e(k+1) = g \times u_*(k) + f[\mathbf{x}(k)] - \mathbf{k}^T\mathbf{e}(k) - y(k+1). \tag{4.24}$$

Substituindo também $y(k+1)$ (4.16) em (4.24) obtém-se a seguinte equação da dinâmica do erro,

$$e(k+1) = -\mathbf{k}^T\mathbf{e}(k) + g \times (u_*(k) - u(k)). \tag{4.25}$$

Se forem definidas as seguintes matrizes $\mathbf{\Lambda}$, e $\mathbf{b}_g$:

$$\mathbf{\Lambda} = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \\ -k_n & -k_{n-1} & -k_{n-2} & \dots & -k_1 \end{bmatrix}, \quad \mathbf{b}_g = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ g \end{bmatrix}, \tag{4.26}$$

então podemos reescrever a equação (4.25) na seguinte forma matricial:

$$\mathbf{e}(k+1) = \mathbf{\Lambda}\mathbf{e}(k) + \mathbf{b}_g\,(u_*(k) - u(k)). \tag{4.27}$$

Considere-se a lei de controlo do algoritmo GPC (4.10), rescrita aqui para referência (4.28),

$$u(k) = \mathbf{K}\left(\mathbf{\Phi r} - \mathbf{F}(z^{-1})y(k) - \mathbf{L}(z^{-1})\Delta u(k-1)\right) + u(k-1). \tag{4.28}$$

Substituindo o modelo da planta (4.11) em (4.28), obtém-se a seguinte lei de controlo, onde o vetor de parâmetros $\hat{\mathbf{\Theta}}$ do modelo se encontra explicito,

$$u(k) = u\left[\mathbf{x}_e, \hat{\mathbf{\Theta}}\right] = \mathbf{K}\left(\mathbf{\Phi r} - \mathbf{F}(z^{-1})\hat{\mathbf{\Theta}}^T(k)\mathbf{\Psi}(k) - \mathbf{L}(z^{-1})\Delta u(k-1)\right) + u(k-1). \tag{4.29}$$

Defina-se o vetor de parâmetros ótimos $\mathbf{\Theta}^*$ tal que o erro entre $u\left[\mathbf{x}_e, \hat{\mathbf{\Theta}}\right]$ (4.29) e a lei de controlo ótimo u_* (4.18) seja mínimo, da seguinte forma:

$$\mathbf{\Theta}^* = \text{argmin}_{\hat{\mathbf{\Theta}} \in \Omega_{\mathbf{\Theta}}} \left[\sup_{\mathbf{x}_e \in \Omega_{\mathbf{x}_e}} \left\| u\left[\mathbf{x}_e, \hat{\mathbf{\Theta}}\right] - u_* \right\| \right], \tag{4.30}$$

onde $\Omega_{\mathbf{\Theta}}$ e $\Omega_{\mathbf{x}_e}$ representam os conjunto de valores admissíveis para $\hat{\mathbf{\Theta}}$ e $\mathbf{x}_e$, respetivamente. Assim, com (4.30), a lei de controlo ótimo u_* é aproximada com erro mínimo por $u[\mathbf{x}_e, \mathbf{\Theta}^*]$, isto é,

$$u_*(k) \approx u^*(k) = u[\mathbf{x}_e, \mathbf{\Theta}^*]. \tag{4.31}$$

Substituindo $u_*(k)$ pela sua aproximação $u^*(k)$ em (4.27), e utilizando (4.29), obtemos

a seguinte equação da dinâmica do erro do sistema,

$$\begin{aligned}
\mathbf{e}(k+1) &= \mathbf{\Lambda e}(k) + \mathbf{b}_g \left(u^*(k) - u(k)\right) \\
&= \mathbf{\Lambda e}(k) + \mathbf{b}_g \Big(\mathbf{K}\Big(\mathbf{\Phi r} - \mathbf{F}(z^{-1})(\mathbf{\Theta}^*)^T\mathbf{\Psi}(k) - \mathbf{L}(z^{-1})\Delta u(k-1)\Big) + u(k-1) \\
&\quad -\mathbf{K}\Big(\mathbf{\Phi r} - \mathbf{F}(z^{-1})\hat{\mathbf{\Theta}}^T(k)\mathbf{\Psi}(k) - \mathbf{L}(z^{-1})\Delta u(k-1)\Big) - u(k-1)\Big) \\
&= \mathbf{\Lambda e}(k) + \mathbf{b}_g\mathbf{KF}\left(\hat{\mathbf{\Theta}}^T(k) - \left(\mathbf{\Theta}^*\right)^T\right)\mathbf{\Psi}(k) \\
&= \mathbf{\Lambda e}(k) + \mathbf{b}_g\mathbf{KF}\tilde{\mathbf{\Theta}}^T(k)\mathbf{\Psi}(k), \qquad (4.32)
\end{aligned}$$

onde $\tilde{\mathbf{\Theta}}(k) = \hat{\mathbf{\Theta}} - \mathbf{\Theta}^*$. A equação (4.32) descreve a relação entre o erro $\mathbf{e}$ do sistema e o erro $\tilde{\mathbf{\Theta}}$ nos parâmetros do modelo.

4.3 Lei de Adaptação

O objetivo da lei de adaptação consiste em minimizar conjuntamente o erro de seguimento do sistema $\mathbf{e}$ e o erro no vetor de parâmetros $\tilde{\mathbf{\Theta}}$ (ajustando $\hat{\mathbf{\Theta}}$). A abordagem utilizada neste controlador utiliza teoria de estabilidade de Lyapunov, onde através da aplicação de critérios de estabilidade é desenvolvida uma lei de adaptação para os parâmetros $\hat{\mathbf{\Theta}}$ que torne o sistema de malha fechada estável.

Para referência, nos desenvolvimentos seguintes, alguns resultados da teoria de estabilidade de Lyapunov são de seguida mencionados [1], [22], [23].

Considere-se um sistema discreto do tipo

$$\mathbf{x}(k+1) = \mathbf{f}(\mathbf{x}(k)), \qquad (4.33)$$

onde $\mathbf{x}(k) \in \mathbb{R}^n$, $\mathbf{f}(\cdot) : \mathbb{R}^n \to \mathbb{R}^n$ e $\mathbf{f}(\cdot)$ é continua. É assumido que o ponto $\mathbf{x} = \mathbf{0} \in \mathbb{R}^n$ é um ponto de equilíbrio, isto é, $\mathbf{f}(\mathbf{0}) = \mathbf{0}$.

A solução $\mathbf{x}(k) = \mathbf{0} \in \mathbb{R}^n$ do sistema (4.33) é dita estável se para um dado valor $\epsilon > 0$, existe um valor $\delta(\epsilon) > 0$, tal que a seguinte relação se verifique,

$$\|\mathbf{x}(k_0)\| < \delta \Rightarrow \|\mathbf{x}(k)\| < \epsilon,\ \forall k_0 < k.$$

A solução $\mathbf{x}(k) = \mathbf{0} \in \mathbb{R}^n$ de (4.33) é dita assimptoticamente estável se for estável e existir um δ tal que a seguinte propriedade se verifique:

$$\|\mathbf{x}(k_0)\| < \delta \Rightarrow \lim_{k\to\infty} \|\mathbf{x}(k)\| = 0.$$

A solução $\mathbf{x}(k) = \mathbf{0} \in \mathbb{R}^n$ de (4.33) é globalmente estável, ou globalmente assimptoticamente estável, se, respetivamente, for estável, ou assimptoticamente estável, para qualquer

valor de $\mathbf{x}(k_0)$.

A teoria de estabilidade de Lyapunov baseia-se em encontrar uma função $V(\mathbf{x})$, chamada de função de Lyapunov para o sistema (4.33), tal que esta satisfaça um conjunto de propriedades. Se estas propriedades forem satisfeitas, então pode concluir-se que o sistema (4.33) é estável. O Teorema 1 é chamado de método direto de Lyapunov aplicado em sistemas discretos [1].

Teorema 1. *Considere-se que o ponto* $\mathbf{x} = \mathbf{0} \in \mathbb{R}^n$ *é um ponto de equilíbrio de* $\mathbf{x}(k+1) = \mathbf{f}(\mathbf{x}(k))$. *Se existir uma função continua* $V(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}$, *que obedeça às seguintes propriedades,*

(a) $V(\mathbf{0}) = 0$ *e* $V(\mathbf{x}) > 0$, $\forall \mathbf{x} \neq \mathbf{0}$,

(b) $\Delta V(\mathbf{x}(k)) = V(\mathbf{f}(\mathbf{x}(k))) - V(\mathbf{x}(k)) < 0$, $\forall \mathbf{x}(k)$,

(c) $\|\mathbf{x}\| \to \infty \Rightarrow V(\mathbf{x}) \to \infty$,

então o ponto $\mathbf{x} = \mathbf{0} \in \mathbb{R}^n$ *é globalmente assimptoticamente estável. Se em (b) a condição for* $\Delta V(\mathbf{x}(k)) \leq 0$, *então o ponto* $\mathbf{x} = \mathbf{0} \in \mathbb{R}^n$ *é apenas globalmente estável*

Considere-se agora em particular a análise de estabilidade de Lyapunov em sistemas lineares do tipo (4.34),

$$\mathbf{x}(k+1) = \mathbf{A}\mathbf{x}(k). \tag{4.34}$$

Se for escolhida a função de Lyapunov (4.35),

$$V(\mathbf{x}) = \mathbf{x}^T\mathbf{P}\mathbf{x}, \tag{4.35}$$

obtém-se o seguinte desenvolvimento para $\Delta V(\mathbf{x}(k))$:

$$\begin{aligned} \Delta V(\mathbf{x}(k)) &= V(\mathbf{x}(k+1)) - V(\mathbf{x}(k)) = \mathbf{x}^T(k)\mathbf{A}^T\mathbf{P}\mathbf{A}\mathbf{x}(k) - \mathbf{x}^T(k)\mathbf{P}\mathbf{x}(k) \\ &= \mathbf{x}^T(k)\left(\mathbf{A}^T\mathbf{P}\mathbf{A} - \mathbf{P}\right)\mathbf{x}(k) = -\mathbf{x}^T(k)\mathbf{Q}\mathbf{x}(k), \end{aligned} \tag{4.36}$$

onde

$$\mathbf{A}^T\mathbf{P}\mathbf{A} - \mathbf{P} = -\mathbf{Q} \tag{4.37}$$

Antes de avançar para o teorema de estabilidade para sistemas lineares, é necessária a seguinte definição de matriz definida positiva. Uma matriz $\mathbf{Q}$ é definida positiva se a seguinte relação se verificar,

$$\mathbf{x}^T\mathbf{Q}\mathbf{x} > 0, \forall \mathbf{x} \neq 0.$$

Teorema 2. *[23] Considere-se o sistema (4.33), com* $\mathbf{A}$ *não-singular. Uma condição necessária e suficiente para que o estado de equilíbrio* $\mathbf{x} = \mathbf{0}$ *do sistema seja assimptoticamente estável é que, dada alguma matriz simétrica definida-positiva* $\mathbf{Q}$, *existe uma matriz*

definida positiva $\mathbf{P}$ *tal que (4.37) se verifica. A função escalar* $V(\mathbf{x})$ *é uma função de Lyapunov. Se* $\Delta V(\mathbf{x}(k))$ *(4.36) não for identicamente nula ao longo de alguma sequência de solução de (4.33), então* $\mathbf{Q}$ *pode ser escolhida como sendo semi-definida positiva.*

Com base nos resultados anteriores, pode-se agora definir a lei de adaptação. Considere-se a seguinte função de Lyapunov para o sistema definido por (4.32),

$$V(\mathbf{e}(k), \tilde{\mathbf{\Theta}}(k-1)) = \frac{1}{2}\mathbf{e}^T(k)\mathbf{P}\mathbf{e}(k) + \frac{1}{2\gamma}\tilde{\mathbf{\Theta}}^T(k-1)\tilde{\mathbf{\Theta}}(k-1), \tag{4.38}$$

onde γ é uma constante positiva e $\mathbf{P}$ é uma matriz $n \times n$ simétrica definida positiva.

Esta função possui um segundo termo que é função do erro no vetor dos parâmetros do modelo $\tilde{\mathbf{\Theta}}(k)$. Quando o erro de identificação é nulo, ou seja, quando $\tilde{\mathbf{\Theta}}(k) = \hat{\mathbf{\Theta}}(k) - \mathbf{\Theta}^* = \mathbf{0}$, a função (4.38) tem a forma de (4.35). Esta abordagem é conhecida por método de síntese de Lyapunov [12], dado que é gerada uma lei de adaptação para $\tilde{\mathbf{\Theta}}(k)$ de forma a satisfazer $\Delta V(\mathbf{e}(k), \tilde{\mathbf{\Theta}}(k-1)) \leq 0$ (Teorema 1).

Pressuposto 5. *Existe uma constante* $\alpha > 0$ *e uma matriz definida positiva* $\mathbf{P}$ *tal que para a matriz* $\mathbf{\Lambda}$ *(4.26), a seguinte equação se verifica [21]:*

$$\mathbf{\Lambda}^T\mathbf{P}\mathbf{\Lambda} - \mathbf{P} \leq -\alpha\mathbf{I} < \mathbf{0}. \tag{4.39}$$

De forma a estudar em que condições o estado $\mathbf{e}(k) = \mathbf{0}$ é globalmente estável para a equação (4.32), e em simultâneo determinar uma lei de adaptação para $\hat{\mathbf{\Theta}}(k)$ de forma a essas condições de estabilidade existirem, é necessário avaliar $\Delta V(\mathbf{e}(k), \tilde{\mathbf{\Theta}}(k-1))$.

Obtém-se assim o seguinte desenvolvimento (4.40)-(4.45). A equação (4.41) é obtida ao realizar a primeira diferença em k de (4.38). Ao substituir (4.32) em (4.41), e definindo $\rho = \mathbf{b}_g\mathbf{K}\mathbf{F}\tilde{\mathbf{\Theta}}^T(k)\mathbf{\Psi}(k)$, obtém-se (4.42). Considerando o Pressuposto 5, obtém-se (4.43). Como a matriz $\mathbf{P}$ é simétrica, tem-se que $\rho^T\mathbf{P}\mathbf{\Lambda}\mathbf{e}(k) = (\mathbf{\Lambda}\mathbf{e}(k))^T\mathbf{P}\rho$, obtendo-se assim após algumas manipulações a equação (4.45).

$$\begin{aligned}
\Delta V(\mathbf{e}(k), \tilde{\mathbf{\Theta}}(k-1)) &= V(\mathbf{e}(k+1), \tilde{\mathbf{\Theta}}(k)) - V(\mathbf{e}(k), \tilde{\mathbf{\Theta}}(k-1)) && (4.40)\\
&= \frac{1}{2}\mathbf{e}^T(k+1)\mathbf{P}\mathbf{e}(k+1) - \frac{1}{2}\mathbf{e}^T(k)\mathbf{P}\mathbf{e}(k)\\
&\quad + \frac{1}{2\gamma}\tilde{\mathbf{\Theta}}^T(k)\tilde{\mathbf{\Theta}}(k) - \frac{1}{2\gamma}\tilde{\mathbf{\Theta}}^T(k-1)\tilde{\mathbf{\Theta}}(k-1)\\
&= \frac{1}{2}\mathbf{e}^T(k+1)\mathbf{P}\mathbf{e}(k+1) - \frac{1}{2}\mathbf{e}^T(k)\mathbf{P}\mathbf{e}(k)\\
&\quad - \frac{1}{2\gamma}\left(\tilde{\mathbf{\Theta}}(k) - \tilde{\mathbf{\Theta}}(k-1)\right)^T\left(\tilde{\mathbf{\Theta}}(k) - \tilde{\mathbf{\Theta}}(k-1)\right)\\
&\quad + \frac{1}{\gamma}\tilde{\mathbf{\Theta}}^T(k)\tilde{\mathbf{\Theta}}(k) - \frac{1}{\gamma}\tilde{\mathbf{\Theta}}^T(k)\tilde{\mathbf{\Theta}}(k-1) && (4.41)
\end{aligned}$$

$$
\begin{aligned}
&= \frac{1}{2}\mathbf{e}^T(k)\left(\mathbf{\Lambda}^T\mathbf{P}\mathbf{\Lambda}-\mathbf{P}\right)\mathbf{e}(k)+\rho^T\mathbf{P}\mathbf{\Lambda}\mathbf{e}(k)\\
&\quad -\frac{1}{2\gamma}\left(\tilde{\mathbf{\Theta}}(k)-\tilde{\mathbf{\Theta}}(k-1)\right)^T\left(\tilde{\mathbf{\Theta}}(k)-\tilde{\mathbf{\Theta}}(k-1)\right)\\
&\quad +\frac{1}{\gamma}\tilde{\mathbf{\Theta}}^T(k)\left(\tilde{\mathbf{\Theta}}(k)-\tilde{\mathbf{\Theta}}(k-1)\right)+\frac{1}{2}\rho^T\mathbf{P}\rho \qquad (4.42)\\
&\leq -\frac{1}{2}\alpha\mathbf{e}^T(k)\mathbf{e}(k)+\rho^T\mathbf{P}\mathbf{\Lambda}\mathbf{e}(k)+\frac{1}{2}\rho^T\mathbf{P}\rho\\
&\quad +\frac{1}{\gamma}\tilde{\mathbf{\Theta}}^T(k)\left(\tilde{\mathbf{\Theta}}(k)-\tilde{\mathbf{\Theta}}(k-1)\right) \qquad (4.43)\\
&= -\frac{1}{2}\alpha\mathbf{e}^T(k)\mathbf{e}(k)+\left(\mathbf{\Lambda}\mathbf{e}(k)\right)^T\mathbf{P}\rho+\frac{1}{2}\rho^T\mathbf{P}\rho\\
&\quad +\frac{1}{\gamma}\tilde{\mathbf{\Theta}}^T(k)\left(\tilde{\mathbf{\Theta}}(k)-\tilde{\mathbf{\Theta}}(k-1)\right)\\
&= -\frac{1}{2}\alpha\mathbf{e}^T(k)\mathbf{e}(k)+\frac{1}{2}\rho^T\mathbf{P}\rho\\
&\quad +\tilde{\mathbf{\Theta}}^T(k)\left(\mathbf{\Lambda}\mathbf{e}(k)\right)^T\mathbf{P}\mathbf{b}_g\mathbf{K}\mathbf{F}\mathbf{\Psi}(k)\\
&\quad +\frac{1}{\gamma}\tilde{\mathbf{\Theta}}^T(k)\left(\tilde{\mathbf{\Theta}}(k)-\tilde{\mathbf{\Theta}}(k-1)\right) \qquad (4.44)\\
&= -\frac{1}{2}\alpha\mathbf{e}^T(k)\mathbf{e}(k)+\frac{1}{2}\rho^T\mathbf{P}\rho\\
&\quad +\tilde{\mathbf{\Theta}}^T(k)\left\{\left(\mathbf{\Lambda}\mathbf{e}(k)\right)^T\mathbf{P}\mathbf{b}_g\mathbf{K}\mathbf{F}\mathbf{\Psi}(k)+\right.\\
&\quad \left.\frac{1}{\gamma}\left(\tilde{\mathbf{\Theta}}(k)-\tilde{\mathbf{\Theta}}(k-1)\right)\right\} \qquad (4.45)
\end{aligned}
$$

Ao analisar o resultado (4.45), verifica-se que se for definida uma lei de adaptação para $\tilde{\mathbf{\Theta}}$ de forma a que o último termo da equação seja nulo, obtém-se a seguinte equação:

$$
\begin{aligned}
\left(\mathbf{\Lambda}\mathbf{e}(k)\right)^T\mathbf{P}\mathbf{b}_g\mathbf{K}\mathbf{F}\mathbf{\Psi}(k)+\frac{1}{\gamma}\left(\tilde{\mathbf{\Theta}}(k)-\tilde{\mathbf{\Theta}}(k-1)\right)&=0\\
\hat{\mathbf{\Theta}}(k)=\hat{\mathbf{\Theta}}(k-1)-\gamma\left(\mathbf{\Lambda}\mathbf{e}(k)\right)^T\mathbf{P}\mathbf{b}_g\mathbf{K}\mathbf{F}\mathbf{\Psi}(k),& \qquad (4.46)
\end{aligned}
$$

onde γ é interpretado como um ganho de adaptação. Este passo (4.46) é necessário para que a condição $\Delta V(\mathbf{e}(k),\tilde{\mathbf{\Theta}}(k-1))\leq 0$ seja satisfeita. Esta abordagem é frequentemente chamada de método de síntese de Lyapunov [12].

De forma a assegurar que $\hat{\mathbf{\Theta}}(k)$ está limitado ao conjunto definido por (4.14) segundo o Pressuposto 2, a lei de adaptação (4.46) é modificada tomando a seguinte forma final:

$$
\hat{\mathbf{\Theta}}(k)=\begin{cases}\hat{\mathbf{\Theta}}(k-1)-\gamma\left(\mathbf{\Lambda}\mathbf{e}(k)\right)^T\mathbf{P}\mathbf{b}_g\mathbf{K}\mathbf{F}\mathbf{\Psi}(k), & \|\hat{\mathbf{\Theta}}(k)\|<m_{\mathbf{\Theta}},\\ \hat{\mathbf{\Theta}}(k-1), & \|\hat{\mathbf{\Theta}}(k)\|\geq m_{\mathbf{\Theta}}.\end{cases} \qquad (4.47)
$$

Substituindo a lei de adaptação (4.46) em (4.45), $\Delta V(\mathbf{e}(k),\tilde{\mathbf{\Theta}}(k-1))$ pode ser rescrita

na seguinte forma:

$$\begin{aligned}\Delta V(\mathbf{e}(k), \tilde{\mathbf{\Theta}}(k-1)) &\leq -\frac{1}{2}\alpha \mathbf{e}^T(k)\mathbf{e}(k) + \frac{1}{2}\rho^T \mathbf{P}\rho \\ &\leq -a\|\mathbf{e}(k)\|^2 + \sigma_{max}(\mathbf{P})\|\rho\|^2, \end{aligned} \tag{4.48}$$

onde $\sigma_{max}(\mathbf{P})$ representa o maior valor próprio de $\mathbf{P}$, $a = \alpha/2$, e

$$\begin{aligned}\|\rho\| &= \|\mathbf{b}_g \mathbf{K}\mathbf{F}\tilde{\mathbf{\Theta}}^T(k)\mathbf{\Psi}(k)\|, \\ &\leq \|\mathbf{b}_g\|\|\mathbf{K}\|\|\mathbf{F}\|\|\tilde{\mathbf{\Theta}}^T(k)\mathbf{\Psi}(k)\|. \end{aligned} \tag{4.49}$$

Aplicando hipótese (4.15) do Pressuposto 2 a (4.49), obtém-se a seguinte relação:

$$\|\rho\| \leq \|\mathbf{b}_g\|\|\mathbf{K}\|\|\mathbf{F}\|\beta. \tag{4.50}$$

Sumariando os resultados anteriores, do Pressuposto 2, os parâmetros do modelo $\hat{\mathbf{\Theta}}$ estão limitados, consequentemente, de (3.9), (3.10), (3.26), (3.27), (3.28), (3.97), (3.98), (3.67), (3.74), e (4.4), os elementos de $\mathbf{F}$ e $\mathbf{G}$ são também limitados. Assim, de (4.26) e (4.50), existe uma constante ρ_c [24] tal que,

$$\|\rho\| \leq \rho_c. \tag{4.51}$$

Por fim, aplicando o resultado (4.51) a (4.48), pode-se concluir que $\Delta V(\mathbf{e}(k), \tilde{\mathbf{\Theta}}(k-1)) \leq 0$ está dentro da bola [24] definida por,

$$\left\{ \mathbf{e}(k) : \|\mathbf{e}(k)\| < \epsilon = \sqrt{\frac{\sigma_{max}(\mathbf{P})}{a}}\rho_c \right\}. \tag{4.52}$$

Teorema 3. *Considere-se o sistema de malha fechada constituído pela planta (4.12), controlador (4.28) e pela lei de adaptação (4.47). Se os Pressupostos 1, 2, 3, 4, 5 se verificarem, então o erro de seguimento* $\mathbf{e}(k)$ *é limitado superiormente por* ϵ *definido por (4.52).*

A prova do Teorema 3 é realizada pela análise anterior e por (4.52).

4.4 Controlador Preditivo com Rede Neuronal Difusa Recorrente (II)

Neste capítulo foi proposto e desenvolvido um controlador baseado em GPC e com um modelo do tipo RFNN. A principal diferença entre este controlador e o exposto no Capítulo

3, deve-se à lei de adaptação dos consequentes da RFNN e à lei de controlo que permitem obter um erro de seguimento limitado.

A Figura 4.1 representa de forma esquemática a estrutura do sistema de controlo em malha fechada.

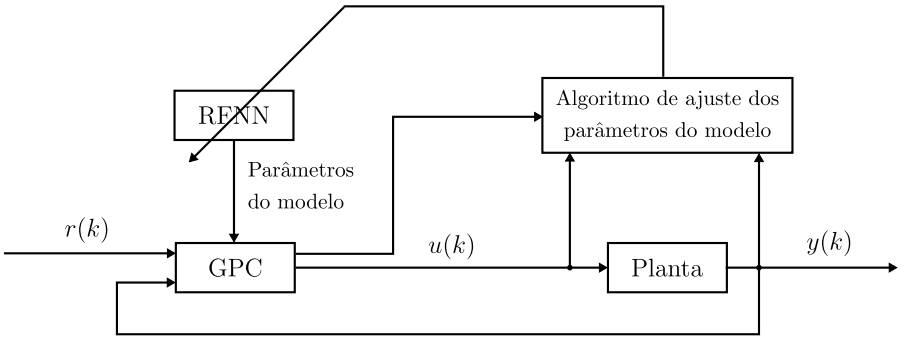


Figura 4.1: Estrutura do Controlador Preditivo com Rede Neuronal Difusa Recorrente.

A sequência de passos necessários para aplicar o controlador desenvolvido neste capítulo a um sistema de controlo em tempo real é a seguinte:

Passo 1 Definir os parâmetros do modelo da RFNN n_y, n_u, d, L, do controlador preditivo N_p, N_u, λ, e da lei de adaptação dos consequentes n, $\mathbf{k}$, g, $\mathbf{P}$, γ.

Passo 2 Inicializar as regras difusas, definindo o valor inicial para as antecedentes m_{ij}, σ_{ij}, ϑ_{ij}, e para as consequentes $\hat{\mathbf{\Theta}}(0)$.

Passo 3 Obter o valor da saída do processo y e do sinal de referência r.

Passo 4 Atualizar os parâmetros do modelo da RFNN com (3.16), (3.17), (3.18), (3.49), (4.47).

Passo 5 Utilizar (3.97) e (3.98) para determinar os coeficientes de (3.57) e (3.58), e consequentemente determinar o próximo incremento na ação de controlo Δu com (4.10).

Passo 6 Aplicar o incremento Δu para obter a ação controlo aplicada sobre o processo.

Passo 7 Repetir a sequência de passos 3 a 6 com um período de amostragem T.

Capítulo 5

Resultados Experimentais

De forma a validar os dois controladores desenvolvidos nos Capítulos 3 e 4, estes são testados em ambiente de simulação MATLAB e também em ambiente de laboratório no controlo de uma planta real. O objetivo dos testes é analisar a capacidade dos controladores com vista a forçar que a saída das plantas siga uma referência desejada, e analisar também a capacidade dos controladores na rejeição de perturbações na saída das plantas.

No ambiente de simulação, os controladores são testados em três plantas distintas com características não lineares, de forma a permitir avaliar as suas capacidades de identificação e de controlo. A identificação das plantas feita pelos controladores é efetuada *online*, ou seja, estes partem da condição inicial onde não têm internamente qualquer informação sobre o modelo da planta. Nos testes implementados em laboratório foi utilizada como planta de teste um motor elétrico com uma carga variável acoplada no seu eixo.

5.1 Testes em Ambiente de Simulação

Nesta secção são analisados os testes efetuados em simulação. O comportamento dos dois controladores desenvolvidos será analisado e comparado entre si para cada uma das três plantas em teste.

5.1.1 Planta I

Neste teste é simulado o modelo da planta não linear descrita na equação (5.1) [15]. Este descreve a relação entre o caudal de água que entra num tanque e o nível da água no seu

interior. O recipiente possui uma válvula de escoamento na sua base.

$$\begin{aligned} y(k) = {} & 0.9722y(k-1) + 0.3578u(k-1) \\ & - 0.1295u(k-2) - 0.3103y(k-1)u(k-1) \\ & - 0.04228y^2(k-2) + 0.1663y(k-2)u(k-2) \\ & - 0.03259y^2(k-1)y(k-2) - 0.3513y^2(k-1)u(k-2) \\ & + 0.3084y(k-1)y(k-2)u(k-2) \\ & + 0.1087y(k-2)u(k-1)u(k-2) + \nu(k) \end{aligned} \tag{5.1}$$

Neste cenário de teste, ambos os controladores foram testados com o sinal de referência $r(k)$ (5.2), sendo também aplicada uma perturbação em degrau $\nu(k)$ (5.2) na saída do processo.

$$r(k) = \begin{cases} 1.0, & 0 < k \leq 400, \\ 0.0, & 400 < k \leq 800, \end{cases} \qquad \nu(k) = \begin{cases} 0.0, & 0 < k < 200, \\ 0.05, & 200 \leq k < 600, \\ 0.2, & 600 \leq k \leq 800. \end{cases} \tag{5.2}$$

A configuração dos controladores foi realizada da seguinte forma. Em ambos os controladores, o número de regras da RFNN foi definido com $L = 3$, as ordens dos polinómios de entrada e de saída e o atraso de transporte foram definidos com $n_u = 1$, $n_y = 2$ e $d = 0$, respetivamente. Estes parâmetros resultam no seguinte vetor de entrada da RFNN,

$$\mathbf{x}_e(k) = [y(k-1), y(k-2), u(k-1), u(k-2)]\,.$$

O atraso de transporte pode ser estimado experimentalmente, medindo o tempo (em períodos de amostragem) que decorre desde o instante em que é aplicado um estímulo à entrada da planta, e o instante em que se verifica uma reação significativa da planta a esse estímulo. Em ambiente de simulação é possível determinar este atraso de transporte desta forma, ou em alternativa, como temos acesso ao modelo da planta basta analisar a equação (5.1) para determinar $d = 0$. Os parâmetros L, n_u, e n_y da RFNN foram escolhidos por tentativa e erro.

O universo de discurso do sinal de saída da planta $y(k)$ é definido com o domínio $Y = [-3, 3]$, e o universo de discurso do sinal de entrada $u(k)$ com domínio $U = [-3, 3]$. A RFNN possui também parâmetros internos que devem ser inicializados de forma apropriada, nomeadamente os diversos parâmetros ϑ_{ij}, m_{ij} e σ_{ij}. Estes parâmetros foram inicializados de modo a que os universos de discurso Y e U sejam totalmente cobertos pelas regras difusas no início da aprendizagem da RFNN. Assim, o ganho de realimentação ϑ_{ij} é inicializado com o valor 1 dado que ainda não há informação acerca do comporta-

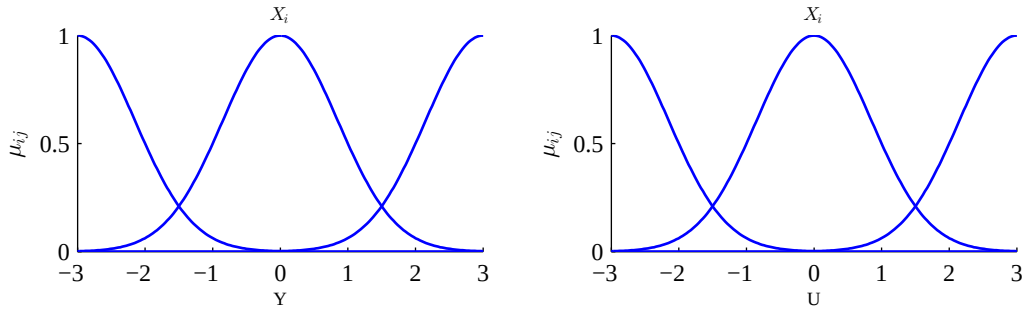


Figura 5.1: Funções de pertença μ_{ij} antecedentes da RFNN com parâmetros iniciais para cada uma das regras difusas R_j e variáveis difusas X_i.

mento da planta, m_{ij} é definido de forma a que os centros das funções de pertença em cada variável difusa estejam distribuídos de forma uniforme, por fim o parâmetro σ_{ij} é definido de modo a ajustar a largura inicial das funções de pertença de forma a que o universo de discurso de cada variável esteja totalmente coberto por pelo menos uma regra. A Figura 5.1 ilustra este conjunto de condições iniciais. A figura é interpretada da seguinte forma, o gráfico da esquerda representa as funções de pertença de cada um dos elementos do vetor $\mathbf{x}_e(k)$ que contêm uma versão atrasada de y (i.e. que contém $y(k-j)$, para algum j), o da direita representa as funções de pertença de cada um dos elementos em do vetor $\mathbf{x}_e(k)$ que contêm uma versão atrasada de u. Cada curva representa conjunto difuso antecedente X_i relacionado com o elemento x_i contido em $\mathbf{x}_e(k)$, na regra difusa j. Como neste caso estamos a lidar com $L = 3$, ou seja, temos 3 regras difusas, cada variável de entrada na rede é representada por 3 conjuntos difusos, um por cada regra difusa j. A curva gaussiana mais à esquerda pertence à regra difusa $j = 1$, e a curva gaussiana mais à direita pertence à regra difusa $j = 3$.

O valor inicial das consequentes das regras é estabelecido de forma distinta em ambos o controladores I (desenvolvido no Capítulo 3) e II (desenvolvido no Capítulo 4). No controlador I, o vetor dos parâmetros das consequentes $\hat{\mathbf{\Theta}}$ é inicializado com todos os seus elementos iguais a 0 (3.55) e a matriz de covariâncias $\mathbf{R}$ é inicializada usando (3.54), com um valor alto de δ forma a indicar a incerteza inicial ao algoritmo de mínimos quadrados. Neste teste, a matriz $\mathbf{R}$ foi inicializada com $\delta = 100$. No Controlador II, cada elemento do vetor $\hat{\mathbf{\Theta}}$ é inicializado com um valor diferente de 0, tendo sido utilizado o valor 0.25.

O conjunto de parâmetros de controlo encontrado para cada controlador é distinto. No Controlador I, os parâmetros de controlo são os seguintes: horizonte de predição $N_p = 10$ e ganho do controlador $\lambda = 10$. No Controlador II, foram definidos os seguintes valores para os parâmetros: $N_p = 10$, $N_u = 1$, $\lambda = 0.9$, $n = 1$, $\mathbf{k} = [k_1] = 1$, $g = 1$, $\mathbf{P} = 1$ e $\gamma = 0.03$. Estes parâmetros de configuração do controlador II resultam no vetor de estado (4.16) $\mathbf{x}(k) = [x_1(k)] = y(k)$.

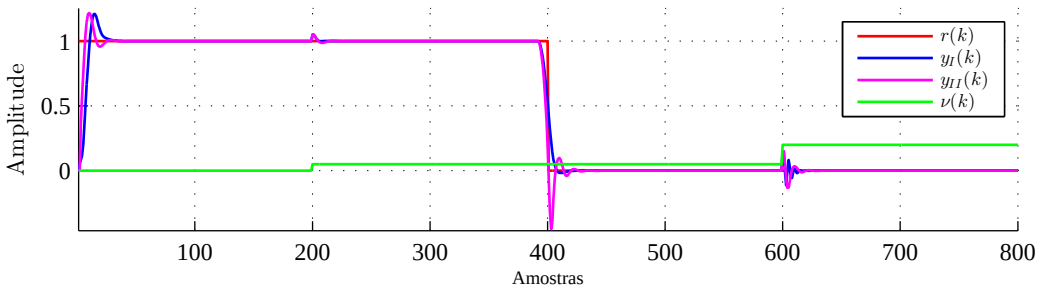


Figura 5.2: Resultado da ação de controlo de ambos os controladores, I e II, sobre a planta I.

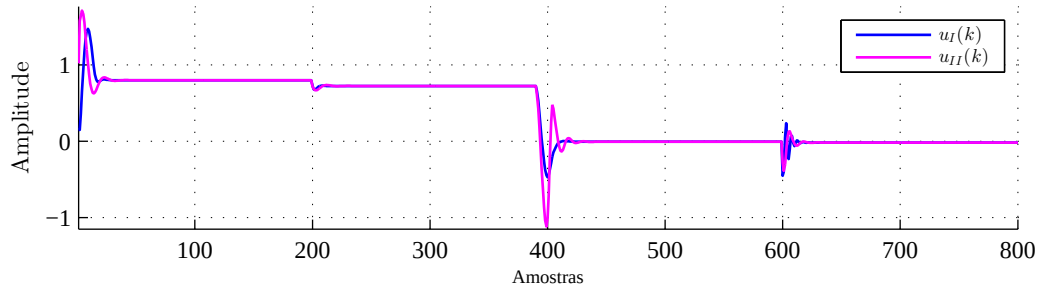


Figura 5.3: Sinais de comando gerados por ambos os controladores, I e II, no controlo da planta I.

Estes valores encontrados para os parâmetros de controlo foram definidos em simulação, após várias execuções do modelo tendo assim encontrado os valores que produzem os resultados mais satisfatórios, ou seja, aqueles que permitiram obter uma melhor capacidade de seguimento da planta.

Análise dos Resultados

A Figura 5.2 demonstra o seguimento da referência $r(k)$ pela planta descrita pela equação (5.1), quando controlada por cada um dos dois controladores desenvolvidos. O esforço de controlo gerado pelos controladores I e II está ilustrado na Figura 5.3.

Ambos os controladores demonstram ser capazes de forçar a planta descrita pela equação (5.1) no seguimento da referência $r(k)$. O tempo de resposta a alterações no sinal de referência $r(k)$, ou à aplicação da perturbação $\nu(k)$ é idêntico nos dois controladores, no entanto existe uma discrepância notória na forma como os controladores convergem para $r(k)$, em específico, o controlador II apresenta oscilações amortecidas. Este comportamento é particularmente visível no instante $k = 400$.

Como é visível na reposta de ambos os controladores, o arranque dos sistemas a partir da condição inicial onde não existe ainda qualquer informação acerca da planta, não

apresenta problemas, convergindo em ambos os casos a saída do processo para a referência.

A evolução dos coeficientes do modelo local/instantâneo linearizado (3.56) (cf. Secção 3.3) identificado pelo controlador I está representada na Figuras A.1 e A.2, no Anexo A.3.1. Como foi referido anteriormente, os coeficientes são inicializados com o valor 0, sendo depois ajustados de forma a identificar o modelo do processo a controlar. Verifica-se, ao analisar as evoluções do coeficientes, que na presença de perturbações, ou de mudanças do ponto de operação do processo, estes são ajustados de forma a considerar esta alterações na planta.

De forma idêntica, a evolução dos coeficientes identificados pelo controlador II está representada nas Figuras A.3 e A.4. Os coeficientes utilizados pelo controlador II são inicializados com o valor 0.25.

5.1.2 Planta II

A planta não linear expressa pela equação (5.3) [25] é utilizada para testar a capacidade de seguimento dos dois controladores.

$$y(k) = \frac{y(k-1)y(k-2)(y(k-1)+2.5)}{1+y^2(k-1)+y^2(k-2)} + 1.25u(k-3) + \nu(k). \tag{5.3}$$

Novamente os controladores são testados no controlo da planta de forma a que a saída desta siga uma referência $r(k)$ (5.4). De forma a testar a robustez da ação de controlo face a perturbações na saída do processo, estas são também simuladas na forma de uma perturbação em degrau $\nu(k)$ (5.4).

$$r(k) = \begin{cases} 4.5, & 0 < k \leq 200, \\ -4.5, & 200 < k \leq 400, \\ 4.5, & 400 < k \leq 800, \end{cases} \qquad \nu(k) = \begin{cases} 0, & 0 < k < 500, \\ 1, & 500 \leq k \leq 800. \end{cases} \tag{5.4}$$

Os parâmetros de configuração foram definidos da seguinte forma. Em ambos os controladores o número de regras na RFNN foi definido como $L = 3$, as ordens dos polinómios de entrada e de saída e o atraso de transporte foram definidas com $n_u = 2$, $n_y = 2$ e $d = 2$ respetivamente. O valor escolhido para o atraso de transporte deve-se a verificar que a saída do processo $y(k)$ (5.3) é afetada por sinais de entrada decorridos em $u(k-3)$. Estes parâmetros resultam no seguinte vetor de entrada da RFNN:

$$\mathbf{x}_e(k) = [y(k-1), y(k-2), u(k-3), u(k-4), u(k-5)]\,.$$

Os universos de discurso dos elementos do vetor $\mathbf{x}_e(k)$ foram definidos da seguinte forma. O universo de discurso dos elementos de $y(k)$ foi definido com $Y = [-8, 8]$ e o

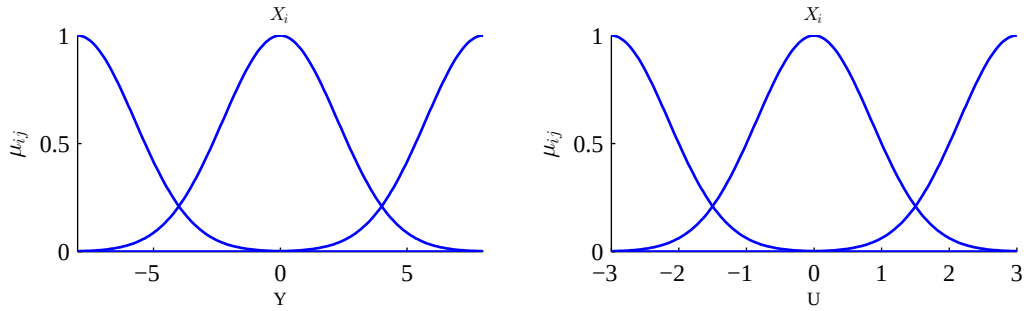


Figura 5.4: Funções de pertença μ_{ij} antecedentes da RFNN com parâmetros iniciais para cada uma das regras difusas R_j e variáveis difusas X_i.

universo de discurso dos elementos de $u(k)$ foi definido com $U = [-3, 3]$. O valor inicial das antecedentes e das consequentes das regras é estabelecido da mesma forma que a utilizada na secção 5.1.1. A Figura 5.4 ilustra as condições iniciais das antecedentes das regras. Na inicialização dos consequentes, foram utilizados os valores $\delta = 100$ para inicializar $\mathbf{R}$ de acordo com (3.54) em ambos os controladores I e II, e $\hat{\mathbf{\Theta}}$ é inicializado com todos os seus elementos iguais a 0 no controlador I, e com os seus elementos iguais a 0.25 no controlador II.

O conjunto de parâmetros de controlo definidos para o Controlador I são os seguintes: $N_p = 6$ e $\lambda = 89$. No Controlador II, foram definidos os seguintes valores para os parâmetros: $N_p = 12$, $N_u = 1$, $\lambda = 132$, $n = 1$, $\mathbf{k} = [k_1] = 1$, $g = 1$, $\mathbf{P} = 1$, $\gamma = 0.008$. Desta forma, o vetor de estado definido no controlador II (4.16) tem a forma $\mathbf{x}(k) = [x_1(k)] = y(k)$.

Análise dos Resultados

A Figura 5.5 representa o comportamento da planta (5.3) quando controlada por cada um dos controladores I e II, e a Figura 5.6 mostra a respetiva evolução do esforço de controlo gerado por cada controlador.

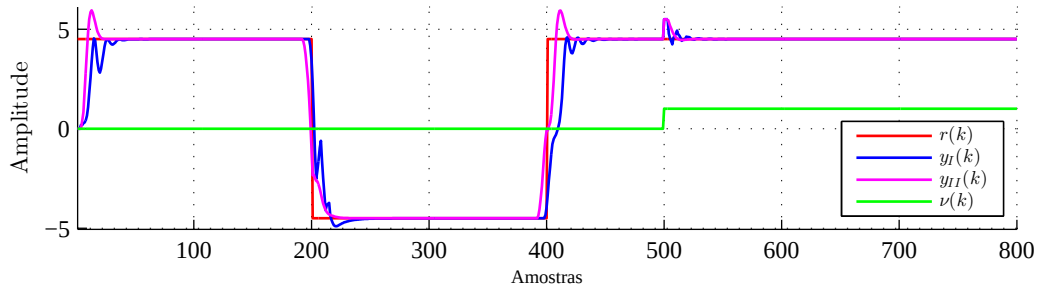


Figura 5.5: Resultado da ação de controlo de ambos os controladores, I e II, sobre a planta II.

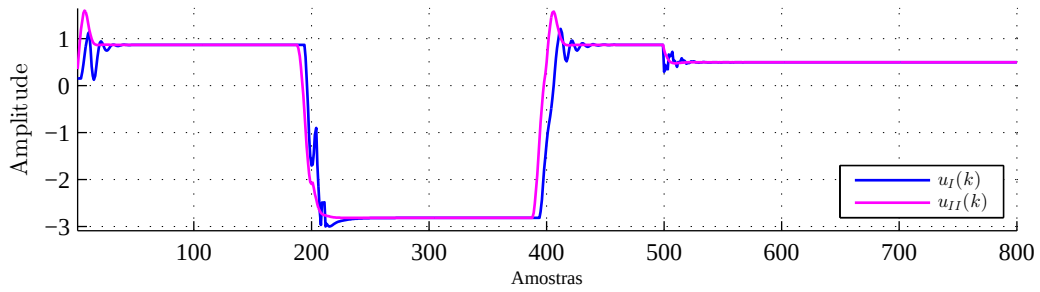


Figura 5.6: Sinais de comando gerados por ambos os controladores, I e II, no controlo da planta II.

Pode-se verificar que ambos os controladores conseguem convergir a saída da planta para a referência $r(k)$ sem erro estacionário, no entanto, é visível que ambos demonstram comportamentos diferentes nos momentos de transição do sinal de referência e de aplicação da perturbação $\nu(k)$. O controlador I apresenta maiores oscilações amortecidas que o controlador II nos momentos de transição, e maiores tempos de convergência. Por outro lado o controlador gera um *overshoot* significativo nos momentos de transição.

A evolução dos coeficientes local/instantâneo linearizado (3.56) (cf. Secção 3.3) identificados pelos controladores I e II está representada nas Figuras A.5 e A.6 relativas ao controlador I, e nas Figuras A.7 e A.8 relativas ao controlador II (Anexo A.3.2).

5.1.3 Planta III

À semelhança dos testes anteriores, a seguinte planta não linear descrita pela equação (5.5) [19] é utilizada para testar os controladores I e II.

$$y(k) = \frac{y(k-1)}{1+y^2(k-1)} + u^3(k-6) + 0.2u^3(k-7) + \nu(k) \tag{5.5}$$

Mais uma vez, o objetivo da lei de controlo é regular a saída da planta $y(k)$ de forma a que esta siga com o menor erro possível uma referência desejada $r(k)$ (5.6). A saída da planta estará sujeita também a uma perturbação em degrau $\nu(k)$ (5.6).

$$r(k) = \begin{cases} 1, & 0 < k \leq 600, \\ -1, & 600 < k \leq 1200, \end{cases} \qquad \nu(k) = \begin{cases} 0, & 0 < k < 400, \\ -0.1, & 400 \leq k < 700, \\ 0.25, & 700 \leq k \leq 1200. \end{cases} \tag{5.6}$$

A inicialização dos parâmetros internos de ambos os controladores é efetuada da seguinte forma. Os parâmetros de configuração da RFNN no controlador I e no controlador II são definidos com, número de regras $L = 3$, a ordem dos polinómios de entrada e de

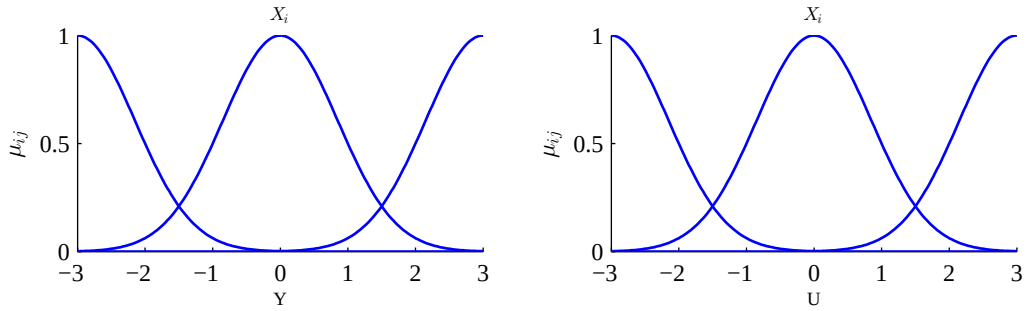


Figura 5.7: Funções de pertença μ_{ij} antecedentes da RFNN com parâmetros iniciais para cada uma das regras difusas R_j e variáveis difusas X_i.

saída do modelo identificado são respetivamente $n_u = 2$, $n_y = 2$. Observando o modelo da planta III (5.5), verifica-se que existe um atraso de transporte $d = 5$. Estes parâmetros geram o seguinte vetor de entrada na rede RFNN,

$$\mathbf{x}_e(k) = [y(k-1), y(k-2), u(k-6), u(k-7), u(k-8)]\,.$$

O universo de discurso foi definido com $U = [-3, 3]$ para o sinal de entrada da planta, e com $Y = [-3, 3]$ para o sinal de saída da planta. Os parâmetros das antecedentes da regras da RFNN são inicializados da mesma forma que a utilizada nas secções anteriores (Secções 5.1.1, e 5.1.2). A Figura 5.7 ilustras as condições iniciais das antecedentes. Na inicialização dos consequentes, foi utilizado o valor $\delta = 100$ para inicializar $\mathbf{R}$ de acordo com (3.54) em ambos os controladores I e II, e $\hat{\mathbf{\Theta}}$ é inicializado com todos os seus elementos iguais a 0 no controlador I, e com os seus elementos iguais a 0.25 no controlador II.

Os parâmetros de controlo definidos para o controlador I são, horizonte de predição $N_p = 10$ e ganho do controlador $\lambda = 220$. No controlador II, foram definidos os seguintes valores para os parâmetros de controlo: $N_p = 15$, $N_u = 1$, $\lambda = 80$, $n = 1$, $\mathbf{k} = [k_1] = 1$, $g = 1$, $\mathbf{P} = 1$, $\gamma = 0.0001$. Estes parâmetros de configuração do controlador II resultam no vetor de estado (4.16) $\mathbf{x}(k) = [x_1(k)] = y(k)$.

Análise dos Resultados

Na Figura 5.8 estão representados os resultados do seguimento da referência $r(k)$ pela planta III, quando atuada por cada um dos controladores I e II com os parâmetros anteriormente descritos. O esforço de controlo de cada um dos controladores I e II está representado na Figura 5.9.

Durante o controlo da planta III, é efetuada a identificação dos parâmetros do modelo local/instantâneo linearizado (3.56) (cf. Secção 3.3) com ordem n_y e n_u e atraso de transporte d anteriormente descritos. A evolução destes parâmetros está representada

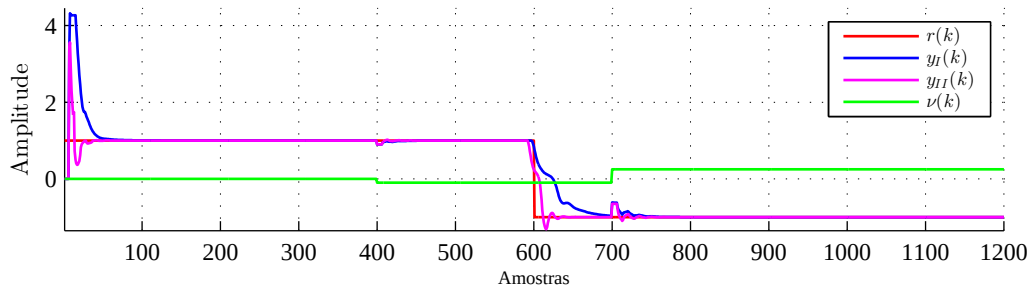


Figura 5.8: Resultado da ação de controlo de ambos os controladores, I e II, sobre a planta III.

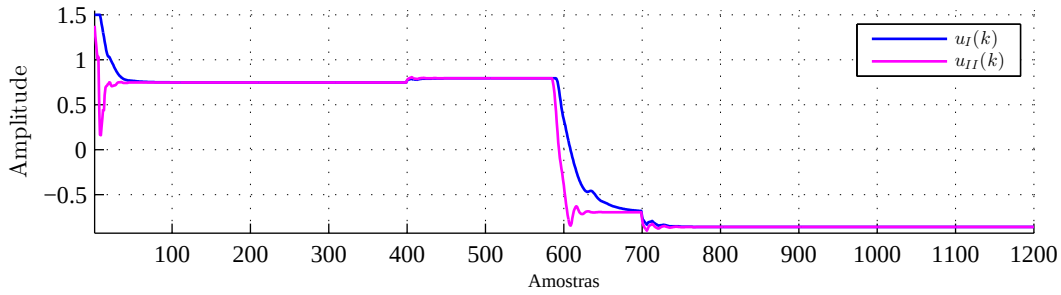


Figura 5.9: Sinais de comando gerados por ambos os controladores, I e II, no controlo da planta III.

nas Figuras A.9 e A.10 relativas ao controlador I, e nas Figuras A.11 e A.12 relativas ao controlador II (Anexo A.3.3).

A simulação dos controladores I e II no controlo da planta III apresenta resultados satisfatórios como é visível nas Figuras 5.8 e 5.9. Ambos os controladores demonstram ser capazes de controlar a saída da planta na presença de mudanças na amplitude do sinal de referência $r(k)$ e de perturbações na saída $\nu(k)$. O comportamento de ambos os controladores é semelhante, existem no entanto algumas diferenças significativas. O controlador I apresenta uma resposta mais lenta que o controlador II na mudança do sinal de referência, mas ambos aparentam ter um comportamento idêntico na rejeição das perturbações. De uma forma geral o controlador II aparenta ter uma resposta mais suave que o controlador I.

5.2 Testes em Laboratório

Após os testes efetuados em ambiente de simulação, nas secções anteriores, nesta secção pretende-se agora verificar o comportamento de ambos os controladores desenvolvidos no controlo de um sistema real em laboratório.

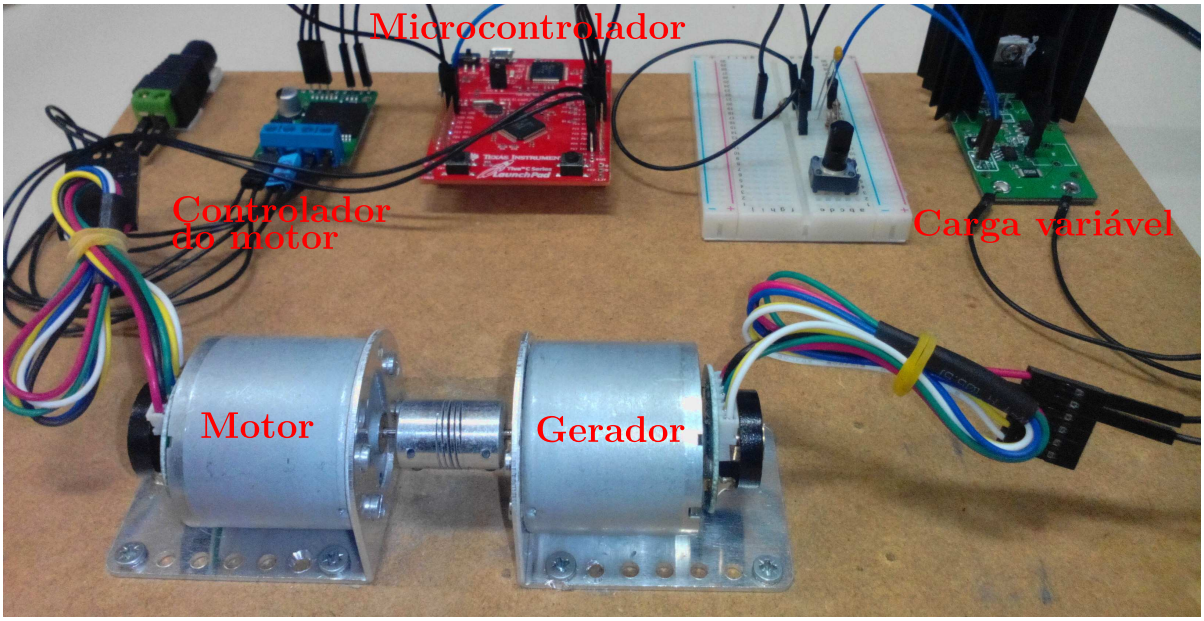


Figura 5.10: Imagem do sistema motor/gerador e carga elétrica.

O sistema utilizado para testar a capacidade de seguimento dos controladores, consiste num grupo motor (M_1)/gerador (M_2) onde estes estão acoplados diretamente através do seu eixo motriz (ver Figura 5.10). O motor M_1 é controlado pelo microcontrolador através de um controlador dedicado (*driver*). O microcontrolador gera um sinal PWM que é convertido e amplificado pelo controlador dedicado e depois aplicado ao motor. O motor M_2 encontra-se a funcionar como gerador e está eletricamente a alimentar uma carga elétrica variável C_1, onde a quantidade de energia dissipada pode ser ajustada pelo microcontrolador por via de um sinal PWM.

O microcontrolador utilizado tem como função gerar os sinais PWM que permitem controlar o motor M_1 e a carga elétrica variável C_1. O microntrolador comunica com um PC através de uma ligação série, e através desta recebe os valores numéricos para ajustar os sinais PWM do motor e da carga. O motor possui um *encoder* acoplado no seu eixo que permite medir a sua velocidade de rotação. Este sinal é enviado pelo microcontrolador pela porta série para o PC, para ser processado pelo algoritmo de controlo.

O *encoder* utilizado tem uma resolução de 64 passos por revolução, e tem um limite de 11000 rotações por minuto. A velocidade de rotação é medida em unidades de [pp/(100 ms)] (pulsos por 100 [ms]). O controlo do motor e da carga elétrica são efetuados com comandos numéricos com valor entre 0 e 4096.

O objetivo de controlo consiste em colocar a velocidade de rotação $y(k)$ do motor M_1 a seguir uma referência desejada $r(k)$ com o menor erro possível. A função do gerador é a de fornecer um carga variável ao motor a que está acoplado, criando assim perturbações $\nu(k)$ na velocidade angular do motor. A Figura 5.11 ilustra este sistema de forma esquemática.

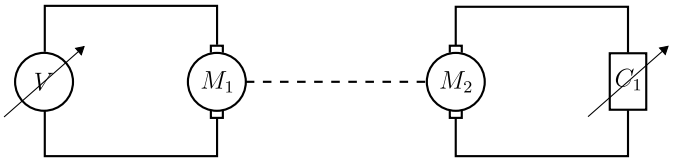


Figura 5.11: Esquema do sistema motor/gerador e carga elétrica.

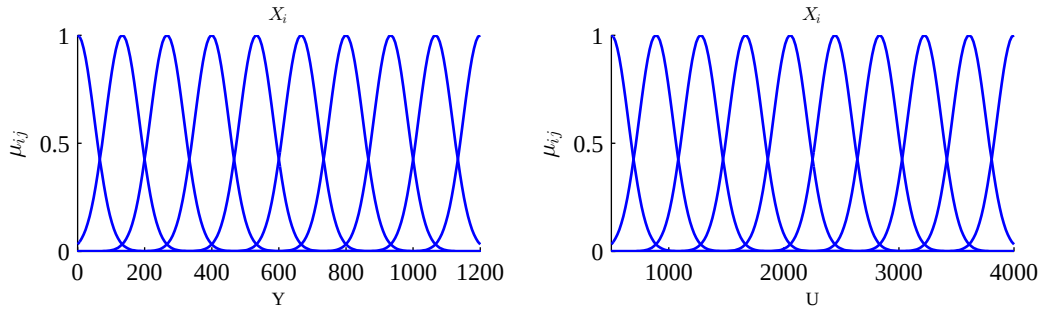


Figura 5.12: Funções de pertença μ_{ij} antecedentes da RFNN com parâmetros iniciais para cada uma das regras difusas R_j e variáveis difusas X_i.

O cenário de teste é concretizado da seguinte forma. O sinal de referência $r(k)$ é definido por (5.7), e o sinal de perturbação $\nu(k)$ é definido por (5.7).

$$r(k) = \begin{cases} 500, & 0 < k \leq 200, \\ 650, & 200 < k \leq 400, \\ 400, & 400 < k \leq 800, \\ 550, & 800 < k \leq 1300, \end{cases} \qquad \nu(k) = \begin{cases} 0, & 0 < k < 600, \\ 4000, & 600 \leq k \leq 1000, \\ 0, & 1000 < k \leq 1300. \end{cases} \tag{5.7}$$

Os parâmetros de configuração da RFNN utilizados por ambos os controladores, I e II, são definidos da seguinte forma. O número de regras é definido com $L = 10$, as ordens dos polinómios de entrada e de saída, e atraso de transporte são definidos com $n_u = 2$, $n_y = 2$ e $d = 0$ respetivamente. Estes parâmetros resultam no seguinte vetor de entrada da RFNN,

$$\mathbf{x}_e(k) = [y(k-1), y(k-2), u(k-1), u(k-2), u(k-3)]\,.$$

Os universos de discurso de cada um dos elementos do vetor $\mathbf{x}_e(k)$ que representam a saída da planta $y(k)$ são $Y = [0, 1200]$ e os elementos que representam a entrada $u(k)$ têm universo de discurso $U = [500, 4000]$. A Figura 5.12 ilustra as condições iniciais dos conjuntos difusos X_i de cada um dos elementos i do vetor de entrada da RFNN, definidos para todas as regras difusas j.

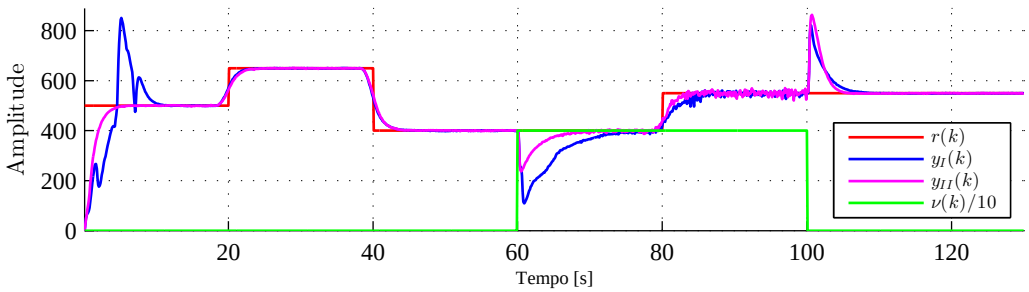


Figura 5.13: Resultado da ação de controlo de ambos os controladores, I e II, sobre o motor.

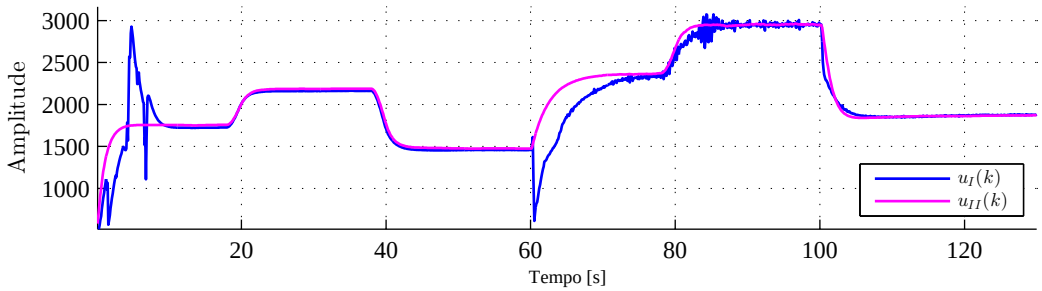


Figura 5.14: Sinais de comando gerados por ambos os controladores, I e II, no controlo do motor.

Da mesma forma que a utilizada nas secções anteriores, o valor inicial das consequentes das regras é inicializada de forma distinta em ambos os controladores. No controlador I, o vetor dos parâmetros das consequentes $\hat{\boldsymbol{\Theta}}$ é inicializado com todos os seus elementos iguais a 0 (3.55) e a matriz de covariâncias $\mathbf{R}$ é inicializada com um valor alto de δ (3.54), tendo sido inicializado com o valor $\delta = 100$. No controlador II, cada elemento do vetor $\hat{\boldsymbol{\Theta}}$ é inicializado com um valor diferente de 0, tendo sido utilizado o valor 0.1.

Os parâmetros de controlo para o controlador I são definidos por: $N_p = 20$ e $\lambda = 20$. Os parâmetros de configuração do controlador II são definidos por: $N_p = 20$, $N_u = 1$, $\lambda = 30$, $n = 1$, $\mathbf{k} = [k_1] = 0.5$, $g = 1$, $\mathbf{P} = 1$, $\gamma = 10^{-8}$. O vetor de estado definido no controlador II fica então com a forma (4.16) $\mathbf{x}(k) = [x_1(k)] = y(k)$. O período de amostragem utilizado pelo ciclo de controlo foi $T = 0.1$ [s].

Análise dos Resultados

A Figura 5.13 ilustra o seguimento da saída da planta em relação à referência desejada, quando controlada por cada um dos controladores. Os respetivos sinais de comando estão ilustrados na Figura 5.14. Os parâmetros do modelo local/instantâneo linearizado (3.56) da planta (cf. Secção 3.3) ao longo de toda a experiência, são ilustrados nas Figuras A.13

e A.14 quando o controlador I é utilizado, e nas Figuras A.15 e A.16, no teste com o controlador II.

Observando o comportamento de ambos os controladores na Figura 5.13, é possível verificar que ambos conseguem controlar o processo de forma a este convergir para o sinal de referência desejado. Existem algumas discrepâncias entre a resposta de ambos os controladores. É visível que o controlador II conseguiu obter melhores resultados de controlo que o controlador I em algumas circunstâncias, nomeadamente, o controlador II demonstrou um menor (desprezável) tempo de identificação, e consequentemente de controlo da planta, em oposição ao controlador I que necessitou de cerca de 9 segundos até a saída da planta convergir para o sinal de referência. Aos 60 segundos, onde é aplicada a perturbação na carga, o controlador I gera um maior *undershoot* que o controlador II, e aos 80 segundos é visível o surgimento de oscilações na carga que são mais eficientemente removidas pelo controlador II.

Capítulo 6

Conclusão e Trabalho Futuro

6.1 Conclusão

Nos capítulos 3 e 4 foram desenvolvidos dois controladores baseados em lógica difusa e em controlo preditivo. A primeira abordagem foi baseada num controlador GPC modificado [19] e numa rede neuronal difusa recursiva. A segunda abordagem foi baseada na primeira, mas com a substituição da lei de adaptação das consequentes por uma nova lei de adaptação que torna o sistema estável segundo a teoria de estabilidade de Lyapunov. Esta segunda abordagem deu origem a um artigo publicado em conferência [20] (Anexo A.1).

Ambos os controladores foram testados em simulação e em laboratório com sucesso, mostrado que têm potencial para utilização em sistemas reais. Os resultados obtidos demonstram que é possível utilizar controlo preditivo em controlo de processos sem que o modelo do processo seja explicitamente conhecido. A ausência de informação inicial acerca do processo é colmatada pela sua aquisição durante a execução de controlo, onde o modelo do processo controlado é identificado através de algoritmos de identificação. Este processo de aprendizagem não é totalmente automático: é necessário algum ajuste de parâmetros internos quando aplicado a um processo distinto, e consequentemente várias iterações são necessárias até ser obtido o melhor conjunto de parâmetros de configuração que forneçam os melhores resultados de regulação.

Observando os resultados obtidos no Capítulo 5, pode-se concluir que o controlador II apresenta na maioria dos cenários um melhor comportamento com uma resposta mais suave, com menor oscilação, à variação do sinal de referência, ou à variação de perturbações aplicadas no processo. Este produz, no entanto, maior *overshoot* que o controlador I em alguns dos casos.

6.2 Trabalho Futuro

Ambos os controladores tiveram um comportamento satisfatório em simulação e em laboratório. Isto significa que há ainda margem para a melhoria destes. Assim, os seguintes pontos são considerados importantes em futuras melhorias:

- Identificar melhores heurísticas para a inicialização dos parâmetros m_{ij}, ϑ_{ij}, e σ_{ij} das antecedentes da RFNN. Não existe neste momento uma metodologia rigorosa que permita definir estes parâmetros, podendo no pior dos casos ser escolhidos valores iniciais para estes parâmetros que gerem soluções sub-ótimas na obtenção do modelo gerado pela RFNN.

- Identificar uma heurística genérica e prática para ajustar os parâmetros de controlo nos controladores I e II. À semelhança do que foi mencionado no ponto anterior, os parâmetros de controlo também necessitam de uma metodologia bem definida para a sua configuração. Durante o teste dos controladores foram sendo definidas algumas regras empíricas para ajustar os parâmetros N_p e λ, onde por exemplo se verificou que um valor de λ maior produz um sistema mais lento, mas um λ menor produz um sistema mais rápido a reagir e ao mesmo tempo com maior probabilidade de conter oscilações na sua saída.

- Refinar a lei de adaptação do controlador II (4.47), ou eventualmente reformular a função candidata de Lyapunov (4.38), de forma a tentar diminuir o número de parâmetros de configuração: na utilização apresentada a lei de adaptação possui vários parâmetros de configuração, nomeadamente n, $\mathbf{k}$, g, $\mathbf{P}$, γ.

Bibliografia

[1] K. J. Astrom and B. Wittenmark, *Adaptive Control*, 2nd ed. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1994.

[2] L. Ljung, *System Identification: Theory for the User*, 2nd ed. Upper Saddle River, NJ, USA: Prentice Hall PTR, 1999.

[3] E. F. Camacho and C. Bordons, *Model Predictive Control*, 2nd ed., ser. Advanced Textbooks in Control and Signal Processing. London: Springer-Verlag, July 2007.

[4] D. W. Clarke, "Application of generalized predictive control to industrial process," *IEEE Control Systems Magazine*, vol. 8, no. 2, pp. 49–55, 1988.

[5] J. A. Rossiter, B. Kouvaritakis, and R. M. Dunnett, "Application of generalised predictive control to a boiler-turbine unit for electricity generation," *IEE Proceedings Part D*, vol. 138, no. 1, pp. 59–67, 1991.

[6] B. Kosko, "Fuzzy systems as universal approximators," *IEEE Trans. Comput.*, vol. 43, no. 11, pp. 1329–1333, Nov. 1994.

[7] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 15, no. 1, pp. 116–132, February 1985.

[8] K. M. Hornik, M. Stinchcombea, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, March 1989.

[9] S. S. Ge, *Nonlinear Identification and Control - a Neural Network Approach.* London: Springer, 2001.

[10] C.-H. Lee and C.-C. Teng, "Identification and control of dynamic systems using recurrent fuzzy neural networks," *IEEE Transactions on Fuzzy Systems*, vol. 8, no. 4, pp. 349–366, August 2000.

[11] L. A. Zadeh, "Fuzzy sets," *Information and Control*, vol. 8, no. 3, pp. 338–353, June 1965.

[12] L.-X. Wang, *A Course in Fuzzy Systems and Control*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1997.

[13] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 15, no. 1, pp. 116–132, February 1985.

[14] C. Fantuzzi and R. Rovatti, "On the approximation capabilities of the homogeneous takagi-sugeno model," in *Proc. of the Fifth IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'96)*, 1996, pp. 1067–1072.

[15] C.-H. Lu and C.-C. Tsai, "Generalized predictive control using recurrent fuzzy neural networks for industrial processes," *Journal of Process Control*, vol. 17, no. 1, pp. 83–82, January 2007.

[16] C.-H. Lee and C.-C. Teng, "Identification and control of dynamic systems using recurrent fuzzy neural networks," *IEEE Transactions on Fuzzy Systems*, vol. 8, no. 4, pp. 349–366, 2000.

[17] D. W. Clarke, C. Mohtadi, and P. S. Tuffs, "Generalized predictive control – part i. the basic algorithm, part ii extensions and interpretations," *Automatica*, vol. 23, no. 2, pp. 137–160, March 1987.

[18] T. A. Johansen and B. A. Foss, "A narmax model representation for adaptive control based on local models," *Modeling, Identification and Control*, vol. 13, no. 1, pp. 25–39, 1992.

[19] C.-H. Lu and C.-C. Tsai, "Adaptive predictive control with recurrent neural network for industrial processes: An application to temperature control of a variable-frequency oil-cooling machine," *IEEE Transactions on Industrial Electronics*, vol. 55, no. 3, pp. 1366–1375, 2008.

[20] J. Mendes, N. Sousa, and R. Araújo, "Adaptive predictive control with recurrent fuzzy neural network for industrial processes," in *Proc. 16th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2011)*. Toulouse, France: IEEE, September 5-9 2011, pp. 1–8.

[21] R. Qi and M. A. Brdys, "Stable indirect adaptive control based on discrete-time t–s fuzzy model," *Fuzzy Sets Syst.*, vol. 159, no. 8, pp. 900–925, 2008.

[22] W. M. Haddad and V. Chellaboina, *Nonlinear Dynamical Systems and Control: A Lyapunov-Based Approach*. Princeton, NJ, USA: Princeton University Press, 2008.

[23] K. Ogata, *Discrete-Time Control Systems*, 2nd ed. Englewood Cliffs, NJ, USA: Prentice–Hall International, Inc, 1995.

[24] J. Mendes and R. Araújo, "Adaptive fuzzy model-based predictive control: a lyapunov approach," in *Proc. 9th Portuguese Conference on Automatic Control (CONTROLO 2010)*, Coimbra, Portugal, September 8-10 2010.

[25] H.-R. Sun, P. Han, and S. Jiao, "A predictive control strategy based on fuzzy system," in *Proc. 2004 IEEE International Conference on Information Reuse and Integration (IRI 2004)*. IEEE, 2004, pp. 549–552.

Apêndice A

Anexos

A.1 Adaptive Predictive Control With Recurrent Fuzzy Neural Network for Industrial Processes

This regular paper was presented as part of the main technical program at IEEE ETFA'2011

Adaptive Predictive Control With Recurrent Fuzzy Neural Network for Industrial Processes

Jérôme Mendes, Nuno Sousa, and Rui Araújo
Institute of Systems and Robotics (ISR-UC), and
Department of Electrical and Computer Engineering (DEEC-UC),
University of Coimbra, Pólo II, PT-3030-290 Coimbra
jermendes@isr.uc.pt, rui@isr.uc.pt

Abstract

The paper proposes an adaptive fuzzy predictive control method. The proposed controller is based on the Generalized predictive control (GPC) algorithm, and a recurrent fuzzy neural network (RFNN) is used to approximate the unknown nonlinear plant. To provide good accuracy in identification of unknown model parameters, an online adaptive law is proposed to adapt the consequent part of the RFNN, and its antecedent part is adapted by backpropagation method. The stability of closed-loop control system is studied and proved via the Lyapunov stability theory. A nonlinear laboratory-scale liquid-level process is used to validate and demonstrate the performance of the proposed control. The simulation results show that the proposed method has good performance and disturbance rejection capacity in industrial processes and outperforms the PID and the classical GPC controllers.

1 Introduction

Generalized predictive control (GPC) [2] has become one of the most popular and powerful control methods in industry. It is a model-based control method where a plant model is used to obtain a predictor model. The GPC has been applied in various plants, and has shown good performance results [3]. However the most plants where GPC has been applied were linear because the quadratic optimization problem involved in GPC is easily solved for the linear prediction case. Previous research has presented GPC for nonlinear plants [16], where was proposed the linearisation of the plants [16]. The disadvantage of GPC, as common factor of all Model Based Predictive Control (MBPC) is its assumption of an accurate model.

This assumption may present problems, because many complex plants are difficult to be modelled mathematically based in physical laws, or have large uncertainties and strong nonlinearities. An alternative to modelling nonlinear plants are fuzzy logic systems and neural networks (NNs). Fuzzy systems may be used to approximate unknown nonlinear functions of the plant. This is theoretically supported by the fact that fuzzy logic systems are universal approximators [8]. Takagi-Sugeno (T-S) [17] fuzzy models have gained much popularity because of their rule consequent structure, that is a real-valued function. On the other hand, NNs are, also, universal approximators [7] and due to this propriety they have been applied to identification and control of nonlinear systems [5]. The merits of both neural and fuzzy systems can be integrated in a fuzzy neural network (FNN), that have been shown to obtain successful results and very effective for system identification. However, a major drawback of the FNN is that its application domain is limited to static problems due to its feedforward network structure [10]. To deal with this problem, interest in using recurrent fuzzy neural networks (RFNN) has been growing in recent years. RFNNs use internal feedback connections as internal memories, which can temporarily store dynamic information and cope with temporal problems efficiently.

In off-line training algorithms the model can be obtained from input-output data collected from a plant. However, this collected dataset can be limited and the obtained model may not provide adequate accuracy. This motivates the introduction of on-line adaptive control methodologies to solve the problem.

In [14] a novel robust model predictive controller for uncertain linear time-varying systems is presented. The proposed algorithm is based on a dual-mode paradigm in which a linear controller is designed to achieve robust stability, good performance according to user defined tuning matrices and offset-free control. However, there are many applications in industry where plant models are nonlinear and it does not consider online auto-adaptation mechanisms to take into account and overcome complex and/or unknown time-varying plant behavior and system disturbances, and in particular to improve performance under such conditions.

A modified generalized predictive control strategy for plants described by second order plus dead time models is proposed in [13]. The proposed scheme is less computationally demanding than the conventional GPC methods

978-1-4577-0018-7/11/$26.00

and is easy to implement for practitioners in industry. In [6] a multiple model predictive control (MMPC) strategy based on T–S fuzzy models for temperature control of air-handling unit in heating, ventilating, and air-conditioning (HVAC) systems is presented. In [11] a Locomotive Brake Control Method based on T-S Fuzzy Modeling Predictive Control is proposed. A fuzzy clustering method is used to determine initial parameters, and a back-propagation algorithm used for parameters adaptation by off-line learning. However, these methods require some knowledge about the model of the process to be controlled, in the form of a model close to the real model of the process. Such knowledge can be difficult to extract in complex industrial processes.

In this paper, a new adaptive model-based predictive tracking controller using recurrent fuzzy neural network is proposed for a class of nonlinear discrete-time processes. The proposed controller is based on GPC algorithm and uses a discrete-time RFNN model adapted on-line. The antecedent part is adapted by backpropagation learning algorithm and the consequent part is adapted by a proposed adaptation law. This controller is able to ensure that the tracking error remains bounded. The stability of closed-loop control system is studied and proved via the Lyapunov stability theory. A diagram of the proposed adaptive fuzzy generalized predictive control (AFGPC) approach is represented in Fig. 1. In the presently proposed

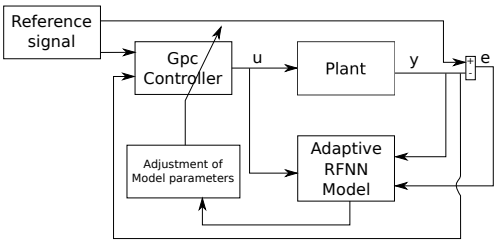


Figure 1: A generic schematic diagram of the proposed control AFGPC.

method, human knowledge or a physical model about the plant model are not necessary. Nevertheless, initial human knowledge about the plant can be integrated in the method.

The paper is organized as follows. Section 2 presents a nonlinear systems modelling method using RFNN models and describe the learning algorithm for the rule's antecedents. Section 3 presents a brief overview of GPC. The proposed control method is described in Section 4. In Section 5, the results of simulations are presented and analysed. Finally, Section 6 makes concluding remarks.

2 Nonlinear Systems Modelling Using RFNN Models

A large class of nonlinear processes can be represented by the following NARMAX model [9]:

$$y(k) = f[y(k-1), y(k-2), \ldots, y(k-n_y), \\ u(k-d-1), \ldots, u(k-d-n_u)] + \zeta(k), \quad (1)$$

where $u(\cdot) : \mathbb{N} \to \mathbb{R}$ and $y(\cdot) : \mathbb{N} \to \mathbb{R}$ are the process input and output, $n_u \in \mathbb{N}$ and $n_y \in \mathbb{N}$ are the orders of input and output respectively, $d \in \mathbb{N}$ is the time-delay of the system, and $\zeta(k) \in \mathbb{R}$ is a sequence of zero-mean Gaussian white noise. In the discrete-time nonlinear SISO plant (1), $f(\cdot) : \mathbb{R}^n \to \mathbb{R}$, where $n = n_y + n_u$, represents a nonlinear mapping which is assumed to be unknown. $f(\cdot)$ will be approximated by a RFNN system. To design the RFNN model, the global operation of the nonlinear system (1) can be accurately approximated by a set of local affine models R_i. Thus, system (1) can be described by a RFNN model defined as:

$$\begin{aligned} R_i: \quad & \text{IF } x_1(k) \text{ is } A_1^i, \text{ and} \ldots \text{and } x_n(k) \text{ is } A_n^i \\ & \text{THEN } y_i(k) = a_i(z^{-1})y(k-1) + \\ & \qquad b_i(z^{-1})u(k-d-1) + \zeta(k), \\ & i = 1, \ldots, N, \end{aligned} \quad (2)$$

where R_i $(i = 1, 2, \ldots, N)$ represents the i-th fuzzy rule, N is the number of rules,

$$\begin{aligned} a_i(z^{-1}) &= a_{1i} + a_{2i}z^{-1} + \ldots + a_{n_y i}z^{-(n_y-1)}, \\ b_i(z^{-1}) &= b_{1i} + b_{2i}z^{-1} + \ldots + b_{n_u i}z^{-(n_u-1)}, \end{aligned} \quad (3)$$

and $u(k)$ is the control output. $x_1(k), \ldots, x_n(k)$ are the input variables of the RFNN system - they can be any variables chosen by the designer [e.g. $y(k-1)$, $u(k-1)$, or other]. A_j^i $(j = 1, \ldots, n)$ are linguistic terms characterized by fuzzy membership functions $\mu_{A_j^i}(k)$, which describe the local operating regions of the plant. A

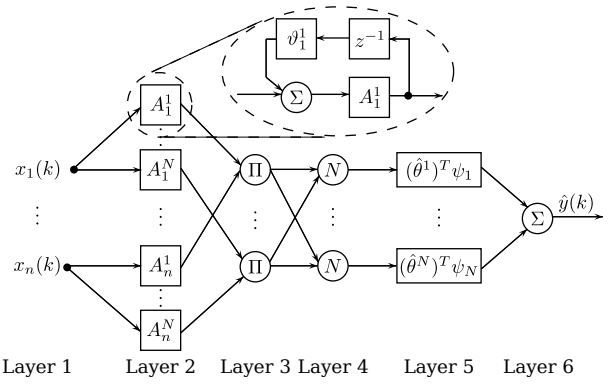


Figure 2: The configuration of RFNN architecture.

schematic diagram of the RFNN is shown in Fig. 2. It is organized in 6 layers which are described as follows. Layer 1 accepts input variables, and its nodes represent linguistic variables. The input values are transmited directly to the next layer. In layer 2, each node, labeled

A_j^i, performs a membership function and acts as a unit of memory. The Gaussian function is adopted here as a membership function. Thus, we have

$$\mu_{A_j^i}(k) = exp\left[-\frac{(x_j(k) + \mu_{A_j^i}(k-1)\vartheta_j^i - m_j^i)^2}{2(\sigma_j^i)^2}\right], \quad (4)$$

where m_j^i and σ_j^i are the center and a mesure of the width of the Gaussian membership function. An important characteristic of this layer is that ϑ_j^i denotes the link gain in the feedback loop. m_j^i, σ_j^i and ϑ_j^i are the adjustable parameters (antecedent parameters). At layer 3, the output of each node gives the product of all the incoming signals, i.e.

$$\bar{\mu}_i(k) = \prod_{j=1}^{n} \mu_{A_j^i}(k). \quad (5)$$

Layer 4 is responsible for the normalization of the activation values of the rules. They are calculated as the ratio of the activation value of every rule to the sum of those of all rules. Layer 5 represents the consequents of the fuzzy rules. Each node have a node function $(\hat{\theta}_i)^T\psi^i$, where θ^i is the adjustable parameter vector (model parameters) and ψ^i depends on the RFNN inputs. At last, the single node in layer 6 computes the overall output as the summation of all incoming signals from the precedent layer.

In the sequel, deterministic models will be considered $[\zeta(k) = 0]$. Thus, from (2) $y(k)$ can be rewritten as

$$\begin{aligned} y(k) &= \sum_{i=1}^{N} \bar{\omega}^i[\mathbf{x}(k)](\theta^i)^T\mathbf{x}_e(k), \quad (6)\\ &= \mathbf{\Theta}^T\mathbf{\Psi}(\mathbf{k}), \end{aligned}$$

where, for $i = 1, \ldots, N$,

$$\begin{aligned} \mathbf{x}(k) &= [x_1(k), x_2(k), \ldots, x_n(k)]^T, \quad (7)\\ \bar{\omega}^i[\mathbf{x}(k)] &= \frac{\prod_{j=1}^{n}\mu_{A_j^i}(k)}{\sum_{i=1}^{N}\prod_{j=1}^{n}\mu_{A_j^i}(k)}, \quad (8)\\ \theta^i &= \left[a_{1i}, \ldots, a_{n_y i}, b_{1i}, \ldots, b_{n_u i}\right]^T, \quad (9)\\ \mathbf{\Theta} &= \left[(\theta^1)^T, (\theta^2)^T, \ldots, (\theta^N)^T\right]^T, \quad (10)\\ \mathbf{x}_e(k) &= [y(k-1), \ldots, y(k-n_y),\\ &\quad u(k-d-1), \ldots, u(k-d-n_u)]^T,\\ \psi^i &= \left[\left(\bar{\omega}^i[\mathbf{x}(k)]\right)\mathbf{x}_e^T(k)\right],\\ \mathbf{\Psi}(k) &= \left[\psi^1, \ldots, \psi^N\right]^T. \end{aligned}$$

Assumption 1 *There exists an (optimal) model parameter vector $\mathbf{\Theta}_*$ that makes model (6) become a perfect representation of the real plant (1).*

Taking into account Assumption 1, i.e. assuming there is no modelling error, and using (6), then the real plant (1) can be represented as

$$y_*(k) = \mathbf{\Theta}_*^T\mathbf{\Psi}(k), \quad (11)$$

where $\mathbf{\Theta}_* = \left[(\theta_*^1)^T, (\theta_*^2)^T, \ldots, (\theta_*^N)^T\right]^T$. It is assumed that the parameters vector $\mathbf{\Theta}_*$ in (11) is unknown. Thus, an approximate model for $y(k)$ is defined as

$$\begin{aligned} \hat{y}(k) &= \sum_{i=1}^{N} \bar{\omega}^i[\mathbf{x}(k)](\hat{\theta}^i)^T\mathbf{x}_e(k), \quad (12)\\ &= \hat{\mathbf{\Theta}}^T(k)\mathbf{\Psi}(k), \end{aligned}$$

where $\hat{\mathbf{\Theta}} = \left[(\hat{\theta}^1)^T, (\hat{\theta}^2)^T, \ldots, (\hat{\theta}^N)^T\right]^T$ is a vector of adjustable parameters which is an estimate of $\mathbf{\Theta}_*$, and $\hat{\theta}^i = \left[\hat{a}_{1i}, \ldots, \hat{a}_{n_y i}, \hat{b}_{1i}, \ldots, \hat{b}_{1n_u}\right]^T$.

Assumption 2 *The parameters vector $\hat{\mathbf{\Theta}}$ belongs to the following bounded region $\Omega_{\mathbf{\Theta}}$, where $m_{\mathbf{\Theta}}$ is a positive constant:*

$$\Omega_{\mathbf{\Theta}} = \{\mathbf{\Theta} \in \mathbb{R}^N : \|\mathbf{\Theta}\| \leq m_{\mathbf{\Theta}}\}, \quad (13)$$

and the approximator error, $d_y(k) = \hat{y}(k) - y_*(k) = \hat{\mathbf{\Theta}}^T(k)\mathbf{\Psi}(k) - \mathbf{\Theta}_*^T\mathbf{\Psi}(k) = \tilde{\mathbf{\Theta}}(k)\mathbf{\Psi}(k)$, where $\tilde{\mathbf{\Theta}}(k) = \hat{\mathbf{\Theta}}(k) - \mathbf{\Theta}_*$, satisfies the following inequality, where δ is a small positive value,

$$\sup_{x \in \Omega_x} |d_y(k)| \leq \delta. \quad (14)$$

2.1 Learning algorithm for the antecedents part of the RFNN

In order to adjust the parameters of the antecedents for identification of an unkown plant, it is used the backpropagation (BP) learning algorithm. The adaptation of the consequent parts of the fuzzy rules is addressed in Section 4. The goal is to derive a learning algorithm for the antecedent parameters m_j^i, σ_j^i and ϑ_j^i, in order to minimize the error funtion

$$\epsilon(k) = \frac{1}{2}(y(k) - \hat{y}(k))^2, \quad (15)$$

where $y(k)$ and $\hat{y}(k)$ are respectively the output of the real plant and the output of the RFNN at the current instant k. The BP algorithm can be written briefly as the recursive equation

$$H(k+1) = H(k) + \Delta H(k) = H(k) + \eta\left(-\frac{\partial\epsilon(k)}{\partial H}\right), \quad (16)$$

where η represents the learning rate and $H = \left[h_1^T, h_2^T, \ldots, h_N^T\right]$ is the parameter vector, with $h_i = \left[m_1^i, \ldots, m_n^i, \sigma_1^i, \ldots, \sigma_n^i, \vartheta_1^i, \ldots, \vartheta_n^i\right]$. The updating laws for m_j^i, σ_j^i and ϑ_j^i are expressed by

$$\begin{aligned} m_j^i(k+1) &= m_j^i(k) - \eta\frac{\partial\epsilon(k)}{\partial m_j^i},\\ &= m_j^i(k) + \eta(y(k) - \hat{y}(k))\frac{\partial\hat{y}(k)}{\partial m_j^i}, \quad (17)\\ \sigma_j^i(k+1) &= \sigma_j^i(k) - \eta\frac{\partial\epsilon(k)}{\partial\sigma_j^i},\\ &= \sigma_j^i(k) + \eta(y(k) - \hat{y}(k))\frac{\partial\hat{y}(k)}{\partial m_j^i}, \quad (18)\\ \vartheta_j^i(k+1) &= \vartheta_j^i(k) - \eta\frac{\partial\epsilon(k)}{\partial\vartheta_j^i},\\ &= \vartheta_j^i(k) + \eta(y(k) - \hat{y}(k))\frac{\partial\hat{y}(k)}{\partial m_j^i}, \quad (19) \end{aligned}$$

where

$$\frac{\partial \hat{y}(k)}{\partial m_j^i} = \frac{\partial \hat{y}(k)}{\partial \bar{\mu}_i(k)} \frac{\partial \bar{\mu}_i(k)}{\partial \mu_{A_j^i}(k)} \frac{\partial \mu_{A_j^i}(k)}{\partial m_j^i}$$
$$= (\hat{y}_i(k) - \hat{y}(k))\bar{\omega}^i[\mathbf{x}(k)] \frac{(x_j(k) + \mu_{A_j^i}(k-1)\vartheta_j^i - m_j^i)}{(\sigma_j^i)^2}, \quad (20)$$

$$\frac{\partial \hat{y}(k)}{\partial \sigma_j^i} = \frac{\partial \hat{y}(k)}{\partial \bar{\mu}_i(k)} \frac{\partial \bar{\mu}_i(k)}{\partial \mu_{A_j^i}(k)} \frac{\partial \mu_{A_j^i}(k)}{\partial \sigma_j^i}$$
$$= (\hat{y}_i(k) - \hat{y}(k))\bar{\omega}^i[\mathbf{x}(k)] \frac{(x_j(k) + \mu_{A_j^i}(k-1)\vartheta_j^i - m_j^i)^2}{(\sigma_j^i)^3}, \quad (21)$$

$$\frac{\partial \hat{y}(k)}{\partial \vartheta_j^i} = \frac{\partial \hat{y}(k)}{\partial \bar{\mu}_i(k)} \frac{\partial \bar{\mu}_i(k)}{\partial \mu_{A_j^i}(k)} \frac{\partial \mu_{A_j^i}(k)}{\partial \vartheta_j^i}$$
$$= (\hat{y}_i(k) - \hat{y}(k))\bar{\omega}^i[\mathbf{x}(k)] \frac{(m_j^i - x_j(k) - \mu_{A_j^i}(k-1)\vartheta_j^i)}{(\sigma_j^i)^2} \times \mu_{A_j^i}(k-1). \quad (22)$$

and $\hat{y}_i(k) = (\theta^i)^T \psi^i$.

The convergence of the RFNN is proved in [12]. The output of the RFNN is convergent if its learning rate η statisfies the equation (23)

$$\eta = \frac{\beta}{\Sigma_{i=1}^N \Sigma_{j=1}^n \left(\left(\frac{\partial \hat{y}(k)}{\partial m_j^i} \right)^2 + \left(\frac{\partial \hat{y}(k)}{\partial \sigma_j^i} \right)^2 + \left(\frac{\partial \hat{y}(k)}{\partial \vartheta_j^i} \right)^2 \right)}, \quad (23)$$

where $0 < \beta < 2$.

3 Predictive Control Law

The adaptive fuzzy generalized predictive control (AFGPC) developed in this paper is motivated from the GPC strategy [2]. For completeness this section briefly overviews the GPC. It is assumed that the plant model is of the form (6), which can be rewritten as follows:

$$\bar{a}(z^{-1})y(k) = \bar{b}(z^{-1})u(k-d-1), \quad (24)$$

where

$$\bar{a}(z^{-1}) = 1 - \bar{a}_1 z^{-1} - \ldots - \bar{a}_{n_y} z^{-n_y}, \quad (25)$$
$$\bar{b}(z^{-1}) = \bar{b}_1 + \bar{b}_2 z^{-1} + \ldots + \bar{b}_{n_u} z^{-(n_u-1)}, \quad (26)$$
$$\bar{a}_j = \sum_{i=1}^N \bar{\omega}^i[\mathbf{x}(k)] a_{ji}, \quad (27)$$
$$\bar{b}_j = \sum_{i=1}^N \bar{\omega}^i[\mathbf{x}(k)] b_{ji}. \quad (28)$$

The GPC control laws is obtained to minimize the following cost function

$$J(k) = \sum_{p=d+1}^{N_p} \left[\hat{y}(k+p|k) - \phi_p r(k+p)\right]^2 + \sum_{p=d+1}^{d+N_u} \left[\lambda(p)\Delta u(k+p-d-1|k)\right]^2, \quad (29)$$

where $\hat{y}(k+p|k)$ is an optimum p-step ahead prediction of the system on instant k, $r(k+p)$ is the future reference trajectory, ϕ_p is the feed forward gain for p steps ahead ($p = d+1, \ldots, N_p$), $\Delta = 1 - z^{-1}$, and $\lambda(p)$ is a positive weighting sequence that considers the future behaviour, usually constant values are considered. N_p and N_u are output and control horizons, respectively. Consider the following Diophantine equation (30):

$$1 = \Delta e_p(z^{-1})\bar{a}(z^{-1}) + z^{-p} f_p(z^{-1}), \quad (30)$$
$$e_p(z^{-1}) = 1 + e_{p,1} z^{-1} + \ldots + e_{p,p-1} z^{-(p-1)}, \quad (31)$$
$$f_p(z^{-1}) = f_{p,0} + f_{p,1} z^{-1} + \ldots + f_{p,n_y} z^{-n_y}, \quad (32)$$

where $e_p(z^{-1})$ and $f_p(z^{-1})$ can be obtained dividing 1 by $\Delta\bar{a}(z^{-1})$ until the remainder can be factorized as $z^{-p} f_p(z^{-1})$ (see [2] for more details). The quotient of the division is the polynomial $e_p(z^{-1})$. Multiplying (24) by $\Delta z^p e_p(z^{-1})$ yields

$$\Delta z^p e_p(z^{-1})\bar{a}(z^{-1})y(k) = \Delta z^p e_p(z^{-1})\bar{b}(z^{-1})u(k-d-1). \quad (33)$$

Defining

$$g_p(z^{-1}) = e_p(z^{-1})\bar{b}(z^{-1}), \quad (34)$$
$$= g_{p,0} + g_{p,1} z^{-1} + \ldots + g_{p,p+n_u-1} z^{-(p+n_u-1)},$$

and substituting (30) and (34) in (33) yields

$$y(k+p|k) = f_p(z^{-1})y(k) + g_p(z^{-1})\Delta u(k+p-d-1). \quad (35)$$

Thus, the best prediction of $y(k+p|k)$ is

$$\hat{y}(k+p|k) = f_p(z^{-1})y(k) + g_p(z^{-1})\Delta u(k+p-d-1). \quad (36)$$

A simple and efficient way to obtain the polynomials $e_p(z^{-1})$ and $f_p(z^{-1})$ is to use recursion of the Diophantine equation that has been demonstrated in [4]. To reduce the computation costs, $N_u = 1$ is chosen, and thus $\lambda(p)$ in (29) is simplified to λ, and is considered $\Delta u(k+1) = \ldots = \Delta u(k+N_u-1) = 0$. Thus (36) can be rewritten as

$$\mathbf{y}(k) = \mathbf{G}_1 u(k) + \mathbf{F}(z^{-1})y(k) + \mathbf{L}(z^{-1}), \quad (37)$$

where

$$\mathbf{y}(k) = \begin{bmatrix} \hat{y}(k+d+1) \\ \hat{y}(k+d+2) \\ \vdots \\ \hat{y}(k+N_p) \end{bmatrix}, \mathbf{F} = \begin{bmatrix} f_{d+1}(z^{-1}) \\ f_{d+2}(z^{-1}) \\ \vdots \\ f_{N_p}(z^{-1}) \end{bmatrix}, \quad (38)$$

$$\mathbf{L} = \begin{bmatrix} \left[g_{d+1}(z^{-1}) - \bar{g}_{d+1}(z^{-1})\right] z\Delta u(k-1) \\ \left[g_{d+2}(z^{-1}) - \bar{g}_{d+2}(z^{-1})\right] z^2\Delta u(k-1) \\ \vdots \\ \left[g_{N_p}(z^{-1}) - \bar{g}_{N_p}(z^{-1})\right] z^{N_p}\Delta u(k-1) \end{bmatrix},$$

$$\bar{g}_p(z^{-1}) = g_{p,0} + g_{p,1} z^{-1} + \ldots + g_{p,p-d-1} z^{d+1-p}.$$
$$\mathbf{G}_1 = \left[g_{1,0}, g_{2,1}, \ldots, g_{N_p,N_p-1}\right]^T. \quad (39)$$

Using (37), (29) can be rewritten as

$$J_{eq}(k) = [\mathbf{F}y(k) + \mathbf{G}_1\Delta u(k) + \mathbf{L} - \mathbf{\Phi R}]^T [\mathbf{F}y(k) + \mathbf{G}_1\Delta u(k) + \mathbf{L} - \mathbf{\Phi R}] + [\lambda\Delta u(k)]^2, \quad (40)$$

By minimizing $J_{eq}(k)$, the following optimum control increment is obtained (see [2]):

$$\Delta u(k) = \mathbf{K}[\mathbf{\Phi R} - \mathbf{F}y(k) - \mathbf{L}], \quad (41)$$

where $\mathbf{K} = (\mathbf{G}_1^T\mathbf{G}_1 + \lambda)^{-1}\mathbf{G}_1^T$.

4 Adaptive Predictive Fuzzy Control

This section explains how the RFNN model (Sections 2) will be used in the predictive control law (Section 3) such that the model parameters (consequent part of the RFNN) can be adapted in novel a Adaptive Predictive Fuzzy Control framework. From here, the antecedent part computed in Section 2.1 are considered fixed. The closed-loop dynamic error equation will be determined, and will be chosen an adaptive law with the goal of minimizing the tracking error $\mathbf{e}$ and the parameters error $\tilde{\mathbf{\Theta}}$ by the minimization of a candidate Lyapunov function.

To design the adaptive predictive fuzzy control architecture consider a class of nonlinear discrete-time dynamic systems modelled by:

$$\begin{aligned} x_n(k+1) &= f[\mathbf{x}(k)] + g[\mathbf{x}(k)]u(k), \quad (42)\\ y(k+1) &= x_n(k+1), \end{aligned}$$

where $\mathbf{x}(k)\hat{=}[x_1(k), x_2(k), \ldots, x_n(k)]$ is the state vector, u is the control input, y is the output of the system, and $f[\mathbf{x}(k)]$ and $g[\mathbf{x}(k)]$ are unknown functions.

Assumption 3 *[15] $|g(\mathbf{x}(k))| > \delta$, where δ is a small real positive number, which implies that the relative degree of the T-S fuzzy model and, consequently, the relative degree of the plant are both equal to one.*

Without loss of generality, it is assumed that $g[\mathbf{x}(k)] > 0$. To simplify the process of computer calculation it is considered that $g[\mathbf{x}(k)] = g > 0$ is constant.

Assumption 4 *[15] The reference trajectory $r(k)$ satisfies*

$$|r(k+1)| \leq U, \quad (43)$$

where U is a known bound.

Let $\mathbf{k} = [k_n, \ldots, k_1]^T$ be chosen such that the zeros of polynomial $k_z = z^n + k_1 z^{n-1} + \ldots + k_n$ are inside in the unit circle centered at the origin of the z plane, and choose the control law

$$u_*(k) = \frac{1}{g}\left\{-f[\mathbf{x}(k)] + r(k+1) + \mathbf{k}^T\mathbf{e}(k)\right\}, \quad (44)$$

where $r(k)$ is the reference model output signal, and

$$\begin{aligned} \mathbf{e}(k) &= [e(k-n-1), \ldots, e(k-1), e(k)]^T, \quad (45)\\ e(k) &= r(k) - y(k). \quad (46) \end{aligned}$$

Substituting (46) into (42) and after some manipulation with (44), the following closed-loop dynamic equation is obtained:

$$e(k+1) = -\mathbf{k}^T\mathbf{e}(k) + g[u_*(k) - u(k)]. \quad (47)$$

Let

$$\mathbf{\Lambda} = \begin{bmatrix} 0 & 1 & 0 & \ldots & 0 \\ 0 & 0 & 0 & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \ldots & 1 \\ -k_n & -k_{n-1} & \ldots & \ldots & -k_1 \end{bmatrix}, \mathbf{b}_g = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ g \end{bmatrix}, \quad (48)$$

then (47) can be rewritten into vector form as

$$\mathbf{e}(k+1) = \mathbf{\Lambda}\mathbf{e}(k) + \mathbf{b}_g[u_*(k) - u(k)]. \quad (49)$$

Assuming that $f[\mathbf{x}(k)]$ is unknown, rewritting (41) as

$$u(k) = \mathbf{K}\left[\mathbf{\Phi R} - \mathbf{F}y(k) - \mathbf{L}\right] + u(k-1), \quad (50)$$

and substituting $y(k)$ by its RFNN approximation $\hat{y}(k)$ (12), then the control law is designed as

$$\begin{aligned} u(k) &= u\left[\mathbf{x}_e(k), \hat{\mathbf{\Theta}}\right], \quad (51)\\ &= \mathbf{K}\left[\mathbf{\Phi R} - \mathbf{F}\hat{y}(k) - \mathbf{L}\right] + u(k-1),\\ &= \mathbf{K}\left[\mathbf{\Phi R} - \mathbf{F}\hat{\mathbf{\Theta}}^T(k)\mathbf{\Psi}(k) - \mathbf{L}\right] + u(k-1). \end{aligned}$$

Define the optimal parameter vector

$$\mathbf{\Theta}_* = \text{argmin}_{\hat{\mathbf{\Theta}}\in\Omega_{\mathbf{\Theta}}}\left[\sup_{\mathbf{x}_e\in\Omega_{\mathbf{x}_e}}\left\|u(\mathbf{x}_e, \hat{\mathbf{\Theta}}) - u_*\right\|\right], \quad (52)$$

where $\Omega_{\mathbf{\Theta}}$ and $\Omega_{\mathbf{x}_e}$ are the sets of admissible values of $\hat{\mathbf{\Theta}}$ and $\mathbf{x}_e$ respectively. Thus, from (52), with a minimum approximation error

$$u_*(k) \approx u^*(k) = u\left[\mathbf{x}_e(k), \mathbf{\Theta}_*\right]. \quad (53)$$

where $u^*(k)$ is defined as the optimal command. By substituting $u^*(k)$ (53) for $u_*(k)$, (49) is rewritten as

$$\begin{aligned} \mathbf{e}(k+1) &= \mathbf{\Lambda}\mathbf{e}(k) + \mathbf{b}_g\left[u^*(k) - u(k)\right],\\ &= \mathbf{\Lambda}\mathbf{e}(k) + \mathbf{b}_g\mathbf{K}\mathbf{F}\tilde{\mathbf{\Theta}}^T(k)\mathbf{\Psi}(k), \quad (54) \end{aligned}$$

where $\tilde{\mathbf{\Theta}}(k) = \hat{\mathbf{\Theta}}(k) - \mathbf{\Theta}_*$.

Assumption 5 *[15] There exist $\alpha > 0$ and a positive-definite symmetric matrix $\mathbf{P}$ such that for matrix $\mathbf{\Lambda}$ (48),*

$$\mathbf{\Lambda}^T\mathbf{P}\mathbf{\Lambda} - \mathbf{P} \leq -\alpha\mathbf{I} < \mathbf{0}. \quad (55)$$

Consider the candidate Lyapunov function for system (54),

$$V(k) = \frac{1}{2}\mathbf{e}^T(k)\mathbf{P}\mathbf{e}(k) + \frac{1}{2\gamma}\tilde{\mathbf{\Theta}}^T(k-1)\tilde{\mathbf{\Theta}}(k-1), \quad (56)$$

where γ is a positive constant and $\mathbf{P}$ is a positive-definite symmetric $n \times n$ matrix meeting Assimption 5.

In order to minimize the tracking error $\mathbf{e}$ and the parameter error $\tilde{\mathbf{\Theta}}$, equation (56) will be minimized. To decrease $V(k)$ it is necessary ensure that $\Delta V(k) < 0$. $\Delta V(k)$ will be analysed and calculated in (57)-(62). In the

next analysis of stability the antecedent part(17), (18) and (19) of RFNN ((17), (18) and (19)) are considered fixed. Taking the first time difference of (56) and with some manipulations (58) is deduced. Using (54) and defining $\rho = \mathbf{b}_g\mathbf{KF}\tilde{\mathbf{\Theta}}^T(k)\mathbf{\Psi}(k)$, (59) can be obtained. By Assumption 5, (60) can be written. Since $\mathbf{P}$ is symmetric, then $\rho^T\mathbf{P\Lambda e}(k) = [\mathbf{\Lambda e}(k)]^T\mathbf{P}\rho$ and with some manipulations (61) is derived. Then, with some manipulations (62) is obtained.

$$
\begin{aligned}
& \Delta V(k) = V(k+1) - V(k), && (57)\\
= \;& \frac{1}{2}\mathbf{e}^T(k+1)\mathbf{Pe}(k+1) - \frac{1}{2}\mathbf{e}^T(k)\mathbf{Pe}(k)\\
& + \frac{1}{2\gamma}\tilde{\mathbf{\Theta}}^T(k)\tilde{\mathbf{\Theta}}(k) - \frac{1}{2\gamma}\tilde{\mathbf{\Theta}}^T(k-1)\tilde{\mathbf{\Theta}}(k-1),\\
= \;& \frac{1}{2}\mathbf{e}^T(k+1)\mathbf{Pe}(k+1) - \frac{1}{2}\mathbf{e}^T(k)\mathbf{Pe}(k)\\
& - \frac{1}{2\gamma}\left[\tilde{\mathbf{\Theta}}(k) - \tilde{\mathbf{\Theta}}(k-1)\right]^T\left[\tilde{\mathbf{\Theta}}(k) - \tilde{\mathbf{\Theta}}(k-1)\right]\\
& + \frac{1}{\gamma}\tilde{\mathbf{\Theta}}^T(k)\tilde{\mathbf{\Theta}}(k) - \frac{1}{\gamma}\tilde{\mathbf{\Theta}}^T(k)\tilde{\mathbf{\Theta}}(k-1), && (58)\\
= \;& \frac{1}{2}\mathbf{e}^T(k)\left(\mathbf{\Lambda}^T\mathbf{P\Lambda} - \mathbf{P}\right)\mathbf{e}(k) + \rho^T\mathbf{P\Lambda e}(k)\\
& - \frac{1}{2\gamma}\left[\tilde{\mathbf{\Theta}}(k) - \tilde{\mathbf{\Theta}}(k-1)\right]^T\left[\tilde{\mathbf{\Theta}}(k) - \tilde{\mathbf{\Theta}}(k-1)\right]\\
& + \frac{1}{\gamma}\tilde{\mathbf{\Theta}}^T(k)\left[\tilde{\mathbf{\Theta}}(k) - \tilde{\mathbf{\Theta}}(k-1)\right] + \frac{1}{2}\rho^T\mathbf{P}\rho, && (59)\\
\leq \;& -\frac{1}{2}\alpha\mathbf{e}^T(k)\mathbf{e}(k) + \rho^T\mathbf{P\Lambda e}(k) + \frac{1}{2}\rho^T\mathbf{P}\rho\\
& + \frac{1}{\gamma}\tilde{\mathbf{\Theta}}^T(k)\left[\tilde{\mathbf{\Theta}}(k) - \tilde{\mathbf{\Theta}}(k-1)\right], && (60)\\
= \;& -\frac{1}{2}\alpha\mathbf{e}^T(k)\mathbf{e}(k) + [\mathbf{\Lambda e}(k)]^T\mathbf{P}\rho + \frac{1}{2}\rho^T\mathbf{P}\rho\\
& + \frac{1}{\gamma}\tilde{\mathbf{\Theta}}^T(k)\left[\tilde{\mathbf{\Theta}}(k) - \tilde{\mathbf{\Theta}}(k-1)\right],\\
= \;& -\frac{1}{2}\alpha\mathbf{e}^T(k)\mathbf{e}(k) + \frac{1}{2}\rho^T\mathbf{P}\rho\\
& + \tilde{\mathbf{\Theta}}^T(k)\left[\mathbf{\Lambda e}(k)\right]^T\mathbf{P}\mathbf{b}_g\mathbf{KF\Psi}(k)\\
& + \frac{1}{\gamma}\tilde{\mathbf{\Theta}}^T(k)\left[\tilde{\mathbf{\Theta}}(k) - \tilde{\mathbf{\Theta}}(k-1)\right] && (61)\\
= \;& -\frac{1}{2}\alpha\mathbf{e}^T(k)\mathbf{e}(k) + \frac{1}{2}\rho^T\mathbf{P}\rho\\
& + \tilde{\mathbf{\Theta}}^T(k)\left\{\left[\mathbf{\Lambda e}(k)\right]^T\mathbf{P}\mathbf{b}_g\mathbf{KF\Psi}(k) + \right. && (62)\\
& \left.\frac{1}{\gamma}\left[\tilde{\mathbf{\Theta}}(k) - \tilde{\mathbf{\Theta}}(k-1)\right]\right\}
\end{aligned}
$$

To minimize $V(k)$, the following parameters adaptation law is chosen such that the third term in (62) is zero

$$
\begin{aligned}
& [\mathbf{\Lambda e}(k)]^T\mathbf{P}\mathbf{b}_g\mathbf{KF\Psi}(k) + \frac{1}{\gamma}\left[\tilde{\mathbf{\Theta}}(k) - \tilde{\mathbf{\Theta}}(k-1)\right] = 0,\\
& \hat{\mathbf{\Theta}}(k) = \hat{\mathbf{\Theta}}(k-1) - \gamma\left[\mathbf{\Lambda e}(k)\right]^T\mathbf{P}\mathbf{b}_g\mathbf{KF\Psi}(k), \quad (63)
\end{aligned}
$$

where γ is an adaptation gain. To explicitly enforce Assumption 2, ensuring that $\hat{\mathbf{\Theta}}(k)$ is bounded to (13), the following adaptation law is used:

$$
\hat{\mathbf{\Theta}}(k) = \begin{cases} \hat{\mathbf{\Theta}}(k-1) - \\ \quad \gamma\left[\mathbf{\Lambda e}(k)\right]^T\mathbf{P}\mathbf{b}_g\mathbf{KF\Psi}(k), & \|\hat{\mathbf{\Theta}}(k)\| < m_{\mathbf{\Theta}}, \\ \hat{\mathbf{\Theta}}(k-1), & \|\hat{\mathbf{\Theta}}(k)\| \geq m_{\mathbf{\Theta}}. \end{cases} \quad (64)
$$

Algorithm 1 Proposed adaptive fuzzy generalized predictive control algorithm.

1. Design control parameters: n_u, n_y, d, N_p, λ, g, $\mathbf{k}$, $\mathbf{P}$, γ, and β.
2. Construct the fuzzy rule base: input variables, respective membership functions ($m_j^i(0)$, $\sigma_j^i(0)$ and $\vartheta_j^i(0)$) and the fuzzy rules.
3. (Initialization) Initialize the model parameters (10) (with or without human knowledge about the plant model) and initialize $u(0)$.
4. Compute $\bar{a}(z^{-1})$ and $\bar{b}(z^{-1})$ using (25) and (26), respectively.
5. Compute control signal $u(k)$ with (41): $\Delta u(k) = \mathbf{K}(\mathbf{\Phi R} - \mathbf{F}y(k) - \mathbf{L})$.
6. Adapt the antecedent parameters (16) using (17), (18) and (19), and the consequent part (model parameters a_{ji} and b_{ji} of (3)) with (64).
7. Go to step 4.

This adaptive law (64) permits the adaption of the consequente part of RFNN (model parameters of (2)) which, with some manipulation explained in Secs. 2 and 3, will be used to obtain $\bar{a}(z^{-1})$ and $\bar{b}(z^{-1})$ in (24). Finally, as explained in Sec. 3, the optimum control increment (41) is obtained.

Using (63), equation (62) can be rewritten as

$$
\begin{aligned}
\Delta V(k) &\leq -\frac{1}{2}\alpha\mathbf{e}^T(k)\mathbf{e}(k) + \frac{1}{2}\rho^T\mathbf{P}\rho\\
&\leq -a\|\mathbf{e}(k)\|^2 + \sigma_{max}(\mathbf{P})\|\rho\|^2, \quad (65)
\end{aligned}
$$

where $\sigma_{max}(\mathbf{P})$ is the largest singular value of $\mathbf{P}$, $a = \alpha/2$, and

$$
\begin{aligned}
\|\rho\| &= \|\mathbf{b}_g\mathbf{KF}\tilde{\mathbf{\Theta}}^T(k)\mathbf{\Psi}(k)\|,\\
&\leq \|\mathbf{b}_g\|\,\|\mathbf{K}\|\,\|\mathbf{F}\|\,\|\tilde{\mathbf{\Theta}}^T(k)\mathbf{\Psi}(k)\|. \quad (66)
\end{aligned}
$$

Using Assumption 2, (66) is rewritten as

$$
\|\rho\| \leq \|\mathbf{b}_g\|\,\|\mathbf{K}\|\,\|\mathbf{F}\|\,\delta. \quad (67)
$$

From Assumption 2, the model parameters, i.e. the components of $\hat{\mathbf{\Theta}}$ are bounded. Thus, from (8)-(10), (25)-(28), (32), (34), and (39), all the elements of $\mathbf{F}$ and $\mathbf{G}$ are bounded.

Consequently, from (48), (67), there exists a positive constant ρ_c such that

$$
\|\rho\| \leq \rho_c. \quad (68)
$$

From (65) and (68), it is concluded that $\Delta V(k) \leq 0$ outside the ball

$$
\left\{\mathbf{e}(k) : \|\mathbf{e}(k)\| < \epsilon = \sqrt{\frac{\sigma_{max}(\mathbf{P})}{a}}\rho_c\right\}. \quad (69)
$$

Theorem 1 *Consider the closed loop system consisting of the plant (11), controller (41) and parameter adaptation law (64). If Assumptions 1-5 hold, then the plant tracking error vector* $\mathbf{e}(k)$ *is bounded above by* ϵ *defined in (69).*

The proof of Theorem 1 is given by the above analysis and by (69).

Algorithm 1 summarizes the design and operation of the proposed adaptive fuzzy generalized predictive control method.

5 Simulation Results

This section presents simulation results to validate the theoretical developments and to demonstrate the performance of the proposed adaptive predictive fuzzy control scheme in nonlinear systems. The control of a laboratory-scale liquid-level process will be simulated. In these simulations, to test the reference tracking performance, parameters convergence, and disturbance rejection capacity, the reference input $r(k)$ is changed with time and a load disturbance $v(k)$ is applied. The proposed controller, AFGPC, will be compared with two classical controllers tipically used in industry processes: PID and GPC.

5.1 Control of a Laboratory-Scale Liquid-Level Process

The plant model used in this simulation was identified [1] as follows:

$$\begin{aligned} y(k) \quad = \quad & 0.9722y(k-1) + 0.3578u(k-1) \\ & -0.1295u(k-2) - 0.04228y^2(k-2) \\ & -0.3103y(k-1)u(k-1) \\ & +0.1087y(k-2)u(k-1)u(k-2) \\ & -0.03259y^2(k-1)y(k-2) \\ & -0.3513y^2(k-1)u(k-2) \\ & +0.3084y(k-1)y(k-2)u(k-2) \\ & +0.1663y(k-2)u(k-2) + v(k), \end{aligned} \tag{70}$$

where $v(k)$ is an external disturbance.
The simulated system consists of a DC pump to feed water into a conical flask that in turn feeds a tank. The input to the system is the voltage of the pump motor, and the output is the height of the water in the conical flask.

In order to test the control system, the following input reference is used

$$r(k) = \begin{cases} 1, & 0 < k \leq 400, \\ 0, & 400 < k \leq 800, \end{cases} \tag{71}$$

and the load disturbance $v(k)$ was applied with $v(k) = 0.05$ for $200 < k < 600$, $v(k) = 0.2$ for $k \geq 600$, and $v(k) = 0$, otherwise. The output order, input order and transport delay of the model are respectively $n_u = 2$, $n_y = 2$ and $d = 1$. The input vector of the RFNN was chosen as $\mathbf{x}(k) = [y(k-1), y(k-2), u(k-1), u(k-2)]^T$, where each input variable has a universe of discourse in the interval $[-3, 3]$. The number of rules of the RFNN is chosen as $N = 3$. To represent the initial absence of knowledge about the plant (70), the parameters of the consequents of the rules (2), components of (10), are initialized as 0.001. The initialization in the antecedents were chosen as $[m_1^i(0), m_2^i(0), m_3^i(0)] = [-2, 0, 2]$ and $\sigma_j^i(0) = 1$, the feedback gain $\vartheta_j^i(0) = 0$. The learning constant β in the BP algorithm is set to $\beta = 1$.The parameters of the AFGPC are specified as: $N_p = 10$, $\lambda = 10$, $g = 1$, $k_1 = 0.9$, $\mathbf{P} = 1$ and gain $\gamma = 0.8$. To compare the proposed controller performance with other classical methods, it was also simulated a PID and a GPC controllers. Similarly to [12], the parameters of the PID controller were chosen as $K_P = 1$, $K_I = 0.2$ and $K_D = 0.01$. For the classical GPC $N_p = 20$, $N_u = 20$, and $\lambda = 3$ were chosen. The linear model parameters used in the GPC controller were obtained with the Reaction Curve Method from [2] which gave the coefficient vectors $a = [1.0000, -0.8007]$ and $b = [0.2258]$.

5.2 Analysis of Results

From the results shown in Fig. 3, it can be seen that

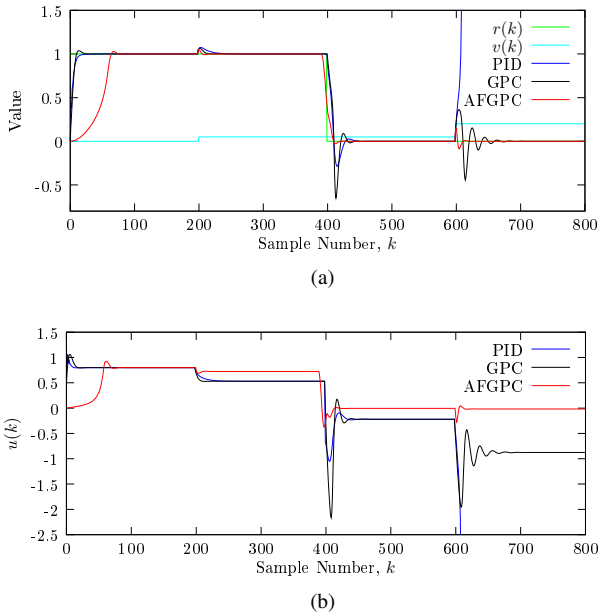


Figure 3: Results of the control of (70): (a) reference signal $r(k)$, load disturbance $v(k)$, and the outputs $y(k)$ of PID, GPC and the proposed controller AFGPC, and (b) applied command signal $u(k)$ of PID, GPC and of the proposed controller AFGPC.

the proposed controller is able to adequately (attain and) control the system output at the desired reference $r(k)$. In terms of initial response of the controller, it can be observed that although there is no initial model knowledge (parameters initialized 0.001), the controller reaches the desired reference signal. When the load disturbance $v(k)$ is applied at $k = 600$, there is an overshoot in system response. As can be seen the controller eliminates this disturbance. The PID and classical GPC controllers system have a good performance in the initial response and quickly reacts to overcome the disturbance with a small value [$v(k) = 0.05$] that is applied for $200 < k < 600$, but when a disturbance with a larger value [$v(k) = 0.2$] is applied for $k \geq 600$, then the PID has a unstable tracking performance, and the classical GPC controller produces a large overshoot and a slowly damped oscilatory system response. Thus, it is concluded that the proposed controller (AFGPC) outperforms the controllers PID and the classical GPC.

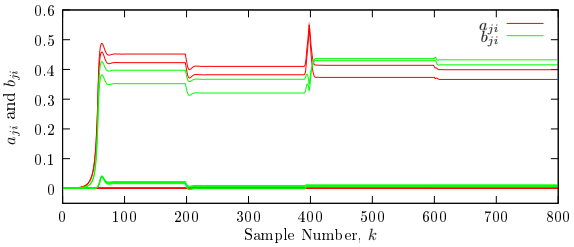


Figure 4: Control of (70): Temporal evolution of the adjustable parameters a_{1i} and a_{2i}; and b_{1i} and b_{2i}.

The temporal evolution of the adjustable parameters of the consequent part of RFNN (model parameters) are shown in Fig. 4. The model parameters, after being initialized with values near zero (0.001), are then adjusted taking in account the desired response. When the load disturbance is applied, the parameters are again adjusted taking into account the corresponding changes in the system. This illustrates that the adaptation mechanisms worked adequately.

6 Conclusion

This paper has proposed a new adaptive model-based predictive controller for a class of nonlinear discrete-time process. The proposed controller is based on the GPC algorithm and uses a RFNN model that is adapted online. To provide a good accuracy in identification of unknown model parameters, an online adaptive law to adapt the model parameters was proposed, and its antecedent part is adapted by backpropagation method. It was demonstrated that the tracking error remains bounded. The stability of closed-loop control system was studied and proved via the Lyapunov stability theory. The simulation results have shown that the proposed method is able to adequately control the plant without human knowledge about the plant model, and has good tracking performance and disturbance rejection capacity. The results indicate that the proposed controller outperforms the PID and the classical GPC controllers. This evidence suggests that the proposed controller could be a good option for industrial process control. As can be seen in the simulations, the adjustable parameters are adjusted for control of the unknown plant and taking into account changes in the system.

Acknowledgement

This work was supported by Mais Centro Operacional Program, financed by European Regional Development Fund (ERDF), and Agência de Inovação (AdI) under Project SInCACI/3120/2009.

Jérôme Mendes is supported by Fundação para a Ciência e a Tecnologia (FCT) under grant SFRH/BD/63383/2009.

References

[1] M. S. Ahmed. Neural-net-based direct adaptive control for a class of nonlinear plants. *IEEE Transactions on Automatic Control*, 45(1):119–124, 2000.

[2] E. F. Camacho and C. Bordons. *Model Predictive Control*. Springer-Verlag, February 1998.

[3] D. W. Clarke. Application of generalized predictive control to industrial process. *IEEE Control Systems Magazine*, 8(2):49–55, 1988.

[4] D. W. Clarke, C. Mohtadi, and P. S. Tuffs. Generalized predictive control – part i. the basic algorithm. *Automatica*, 23(2):137–148, March 1987.

[5] S. S. Ge. *Nonlinear Identification and Control - a Neural Network Approach*. Springer, London, 2001.

[6] M. He, W.-J. Cai, and S.-Y. Li. Multiple fuzzy model-based temperature predictive control for hvac systems. *Information Sciences*, 169(1-2):155–174, 2005.

[7] K. M. Hornik, M. Stinchcombea, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, March 1989.

[8] B. Kosko. Fuzzy systems as universal approximators. *IEEE Transactions on Computers*, 43(11):1329–1333, 1994.

[9] M. Lazar and O. Pastravanu. A neural predictive controller for non-linear systems. *Mathematics and Computers in Simulation*, 60(3-5):315–324, September 2002.

[10] C.-H. Lee and C.-C. Teng. Identification and control of dynamic systems using recurrent fuzzy neural networks. *IEEE Transactions on Fuzzy Systems*, 8(4):349–366, August 2000.

[11] J. Liu, Z. Huang, W. Liu, Y. Yang, and H. Tong. Locomotive brake control method based on t-s fuzzy modeling predictive control. In *Proceedings of the 2008 Second International Symposium on Intelligent Information Technology Application*, pages 602–607, Washington, DC, USA, 2008. IEEE Computer Society.

[12] C.-H. Lu and C.-C. Tsai. Generalized predictive control using recurrent fuzzy neural networks for industrial processes. *Journal of Process Control*, 17(1):83–82, January 2007.

[13] A. Neshasteriz, A. K. Sedigh, and H. Sadjadian. Generalized predictive control and tuning of industrial processes with second order plus dead time models. *Journal of Process Control*, 20(1):63–72, January 2010.

[14] G. Pannocchia. Robust model predictive control with guaranteed setpoint tracking. *Journal of Process Control*, 14(8):927–937, May 2004.

[15] R. Qi and M. A. Brdys. Stable indirect adaptive control based on discrete-time t-s fuzzy model. *Fuzzy Sets Systems*, 159(8):900–925, 2008.

[16] J. A. Rossiter, B. Kouvaritakis, and R. M. Dunnett. Application of generalised predictive control to a boiler-turbine unit for electricity generation. *IEE Proceedings Part D*, 138(1):59–67, 1991.

[17] T. Takagi and M. Sugeno. Fuzzy identification of systems and its applications to modeling and control. *IEEE Transactions on Systems, Man, and Cybernetics*, 15(1):116–132, February 1985.

A.2 Solução da Equação Diofantina

No capítulo 3 foi referida a necessidade de resolver uma equação do tipo:

$$\frac{1}{\Delta a(z^{-1})} = e_p(z^{-1}) + z^{-p}\frac{f_p(z^{-1})}{\Delta a(z^{-1})}. \tag{A.1}$$

A equação (A.1) é usualmente chamada de equação diofantina e no contexto deste trabalho é utilizada para isolar os termos futuros e termos passados da sequência de ruído (ver Secção 3.2.2). Assim, pretende-se expandir a divisão polinomial $1/\Delta a(z^{-1})$ em dois termos representados pelos polinómios e_p e f_p dado p.

Comecemos pelo seguinte exemplo em que o polinómio a é dado por $a(z^{-1}) = 1 - 0.8z^{-1}$. O polinómio $\Delta a(z^{-1})$ é dado por $\Delta a(z^{-1}) = (1 - z^{-1})(1 - 0.8z^{-1}) = 1 - 1.8z^{-1} + 0.8z^{-2}$. A divisão longa $1/\Delta a(z^{-1})$ é mostrada em (A.2) para $p = 3$.

$$\begin{array}{rl|l}
1 & & \overline{1-1.8z^{-1}\ +0.8z^{-2}} \\
\cline{1-1}
-1+1.8z^{-1}\ -0.8z^{-2} & & 1+1.8z^{-1}+2.44z^{-2} \\
\cline{1-1}
\quad 1.8z^{-1}\ -0.8z^{-2} & & \\
\cline{1-1}
\quad -1.8z^{-1}+3.24z^{-2}\ -1.44z^{-3} & & \\
\cline{1-1}
\quad\quad 2.44z^{-2}\ -1.44z^{-3} & & \\
\cline{1-1}
\quad\quad -2.44z^{-2}+4.392z^{-3}-1.952z^{-4} & & \\
\cline{1-1}
\quad\quad\quad 2.952z^{-3}-1.952z^{-4} & &
\end{array} \tag{A.2}$$

Observado os resultados para cada iteração p de (A.2), os polinómios e_p e f_p são obtidos da seguinte forma, o polinómio e_p assume o valor do resultado da divisão, e o resto da divisão é interpretado como $f_p z^{-p}$. Assim, como foram efetuadas 3 divisões, os polinómios e_p e f_p foram determinados para $p = 1, 2, 3$. Os polinómios obtidos no exemplo estão representados em (A.3).

$$\begin{aligned}
e_1(z^{-1}) &= 1, \\
f_1(z^{-1}) &= 1.8 - 0.8z^{-1} \\
e_2(z^{-1}) &= 1 + 1.8z^{-1}, \\
f_2(z^{-1}) &= 2.44 - 1.44z^{-1} \\
e_3(z^{-1}) &= 1 + 1.8z^{-1} + 2.44z^{-2}, \\
f_3(z^{-1}) &= 2.952 - 1.952z^{-1}.
\end{aligned} \tag{A.3}$$

Para mais facilmente compreender o processo de divisão e extrair um algoritmo computacional, cada iteração da divisão no exemplo anterior é representado em (A.4), (A.5) e

(A.6) para $p = 1, 2, 3$ respetivamente.

$$\begin{aligned}\frac{1}{1 - 1.8z^{-1} + 0.8z^{-2}} &= 1 + \frac{-(-1.8z^{-1} + 0.8z^{-2})1}{1 - 1.8z^{-1} + 0.8z^{-2}} \\ &= 1 + \frac{1.8 - 0.8z^{-1}}{1 - 1.8z^{-1} + 0.8z^{-2}} z^{-1},\end{aligned} \tag{A.4}$$

$$\begin{aligned}\frac{1}{1 - 1.8z^{-1} + 0.8z^{-2}} &= 1 + 1.8z^{-1} + \frac{-0.8z^{-2} - (-1.8z^{-1} + 0.8z^{-2})1.8z^{-1}}{1 - 1.8z^{-1} + 0.8z^{-2}} \\ &= 1 + 1.8z^{-1} + \frac{2.44 - 1.44z^{-1}}{1 - 1.8z^{-1} + 0.8z^{-2}} z^{-2},\end{aligned} \tag{A.5}$$

$$\begin{aligned}\frac{1}{1 - 1.8z^{-1} + 0.8z^{-2}} &= 1 + 1.8z^{-1} + 2.44z^{-2} + \frac{1.44z^{-3} - (-1.8z^{-1} + 0.8z^{-2})2.44z^{-2}}{1 - 1.8z^{-1} + 0.8z^{-2}} \\ &= 1 + 1.8z^{-1} + 2.44z^{-2} + \frac{2.952 - 1.952z^{-1}}{1 - 1.8z^{-1} + 0.8z^{-2}} z^{-3}.\end{aligned} \tag{A.6}$$

Definamos os seguintes parâmetros iniciais, $f_0 = 1$ e $e_1 = 1$. Definamos também os valores $\bar{f}_p$ e $\bar{\Delta}a$, em que estes diferem de f_p e Δa apenas no primeiro termo que é omitido. Assim, olhando por exemplo para (A.4), verificamos o seguinte

$$e_1 = 1, \tag{A.7}$$

$$f_1 z^{-1} = -(-1.8z^{-1} + 0.8z^{-2})1 = (1.8 - 0.8z^{-1})z^{-1},$$

$$\bar{f}_0 - \bar{\Delta}a f_{0,0}. \tag{A.8}$$

Onde $f_{0,0}$ é o primeiro termo de f_0. Notar que como $f_0 = 1$, $\bar{f}_0$ é nulo. De forma idêntica para $p = 2$, temos:

$$\begin{aligned}e_2 &= 1 + 1.8z^{-1}, \\ &= 1 + f_{1,0}z^{-1},\end{aligned} \tag{A.9}$$

$$\begin{aligned}f_2 z^{-2} &= -0.8z^{-2} - (-1.8z^{-1} + 0.8z^{-2})1.8z^{-1}, \\ &= (2.44 - 1.44z^{-1})z^{-2}, \\ &= \bar{f}_1 - \bar{\Delta}a f_{1,0}.\end{aligned} \tag{A.10}$$

Generalizando os resultados anteriores, obtém-se as seguintes equações recursivas para

determinar a solução da equação diofantina (A.1):

$$\begin{aligned} f_0 &= 1, \\ e_1 &= 1, \\ f_p z^{-p} &= \bar{f}_{p-1} - \bar{\Delta a} f_{p-1,0}, \\ e_p &= e_{p-1} + f_{p-1,0} z^{-p-1}. \end{aligned} \tag{A.11}$$

O grau dos polinómios f_p e_p é dado por $n_{\Delta a} - 1 = n_a$ e $p - 1$ respetivamente, onde $n_{\Delta a}$ e n_a são os graus dos polinómios $\Delta a(z^{-1})$ e $a(z^{-1})$ respetivamente.

A listagem A.1 implementa em linguagem *MATLAB* a implementação computacional das equações representadas em (A.11).

Listagem A.1: Algoritmo da solução da equação diofantina em código *MATLAB*.

```
function [F_out, G_out] = Diophantine_recursion(A, B, d, Np)
% Funcao que constroi os polinomios F e G para o horizonte de predicao
% do controlador GPC.
%
% Argumentos:   A       - Polinomio de saida do modelo CARIMA da planta
%               B       - Polinomio de entrada do modelo CARIMA da planta;
%               d       - Atraso entre a entrada e a saida da planta;
%               Np      - Horizonte de predicao do controlador GPC;
%
% Saida:        F       - Polinomios obtidos por recurscao da equacao
%                         Diofantina;
%               G       - Polinomo auxiliar do modelo de predicao.


%% Inicializacoes
nA = length(A);             % ordem de delta_A
nB = length(B)-1;           % ordem de B
F = zeros(Np, nA);          % matriz com os coeficientes de F
G = zeros(Np, nB+Np);       % matriz com os coeficientes de G
delta_A = zeros(1, nA+1);   % polinomio A multiplicado por (1 - z^-1)


%% Algoritmo

% multiplicar A por (1 - z^-1);
```

```
26 delta_A(1) = A(1); delta_A(nA+1) = -A(nA);
27 for i = 2:nA
28     delta_A(i) = A(i) - A(i-1);
29 end
30
31 % Contruir os polinomios E, F e G para todas as previsoes ate Np
32 for p = 1:Np
33     % Polinomio F
34     for i = 1:nA
35         if p == 1   % polinomio inicial
36             F(p, i) = -delta_A(i+1);
37             continue;
38         end
39
40         if i == nA % o numero de coeficientes de F e inferior ao de A por
               1
41             F(p, i) = -F(p-1, 1)*delta_A(i+1);
42             continue;
43         end
44
45         F(p, i) = F(p-1, i+1) - F(p-1, 1)*delta_A(i+1);
46     end
47
48     % Polinomio G
49     if p == 1   % polinomio inicial
50         G(1, 1:(nB+1)) = B;
51     else    % casos seguintes
52         G(p, :) = G(p-1, :);
53
54         for i = 1:(nB+1)
55             G(p, p+i-1) = G(p-1, p+i-1) + F(p-1, 1)*B(i);
56         end
57     end
58
59 end
60
61 % % Remover da matrizes F, E e G os polinomios de predicoes anteriores a d
```

```
62 F_out = F((d+1):Np, :);
63 G_out = G((d+1):Np, :);
```

A.3 Coeficientes do Modelos Identificados pelos Controladores

A.3.1 Testes em Ambiente de Simulação - Planta I

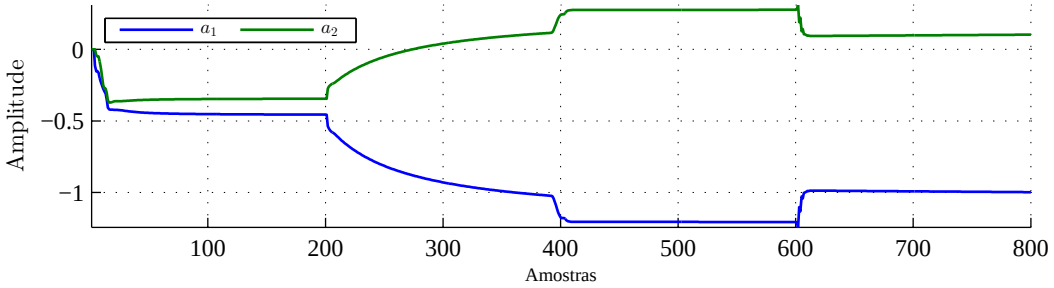


Figura A.1: Coeficientes do modelo, do polinómio $a(z^{-1})$ (3.56) (o coeficiente a_0 não está representado porque assume sempre o valor 1), identificado pelo controlador I.

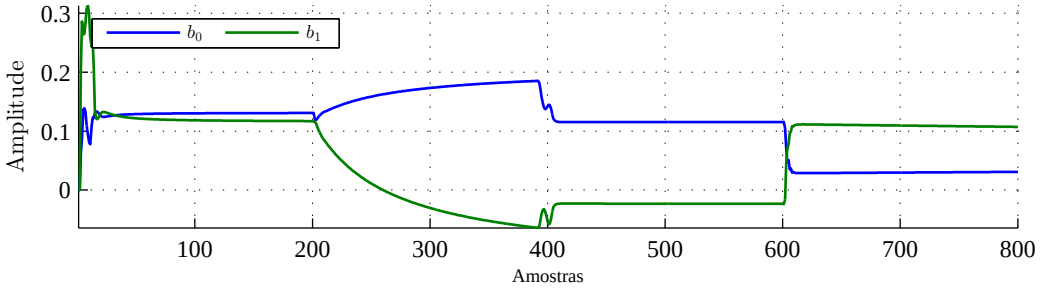


Figura A.2: Coeficientes do modelo, do polinómio $b(z^{-1})$ (3.56), identificado pelo controlador I.

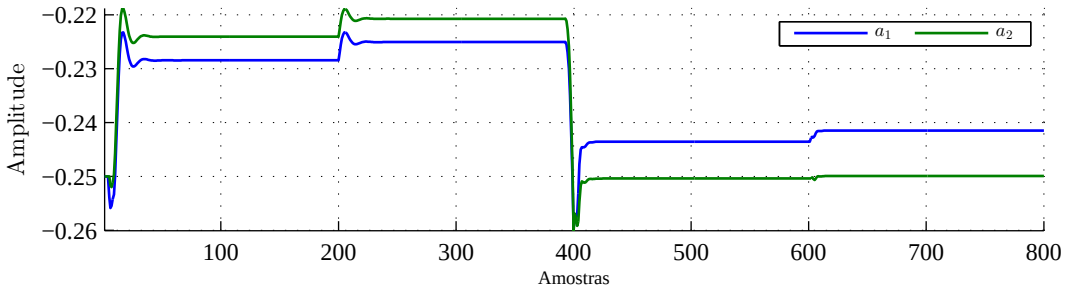


Figura A.3: Coeficientes do modelo, do polinómio $a(z^{-1})$ (3.56) (o coeficiente a_0 não está representado porque assume sempre o valor 1), identificado pelo controlador II.

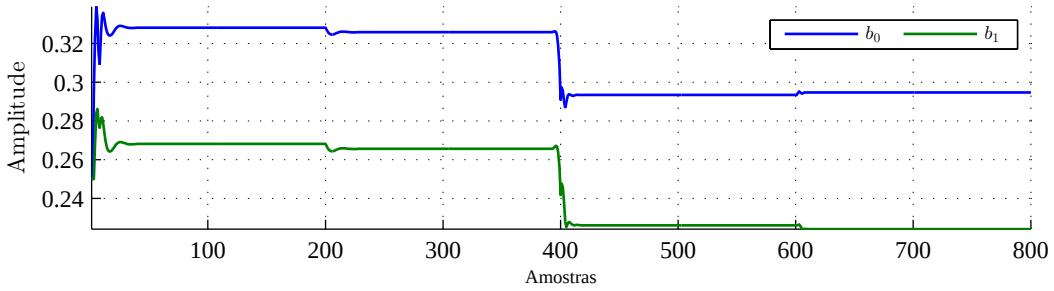


Figura A.4: Coeficientes do modelo, do polinómio $b(z^{-1})$ (3.56), identificado pelo controlador II.

A.3.2 Testes em Ambiente de Simulação - Planta II

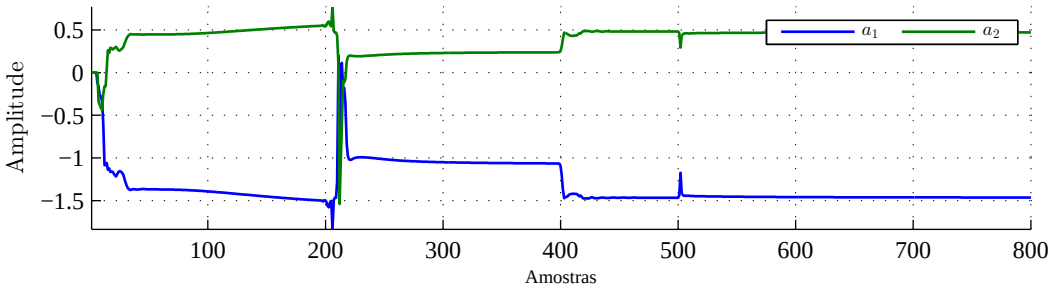


Figura A.5: Coeficientes do modelo, do polinómio $a(z^{-1})$ (3.56) (o coeficiente a_0 não está representado porque assume sempre o valor 1), identificado pelo controlador I.

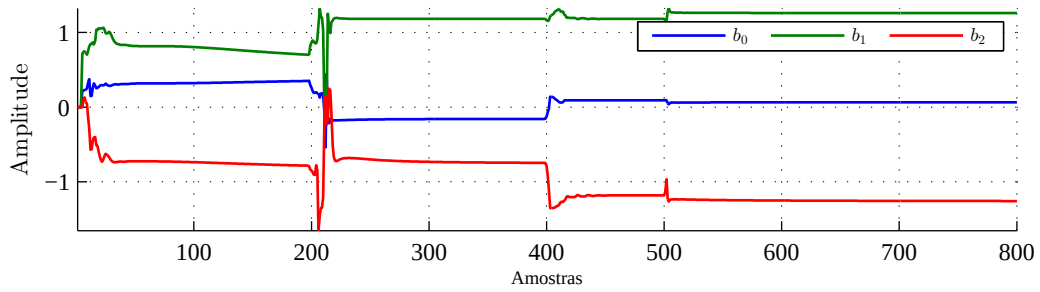


Figura A.6: Coeficientes do modelo, do polinómio $b(z^{-1})$ (3.56), identificado pelo controlador I.

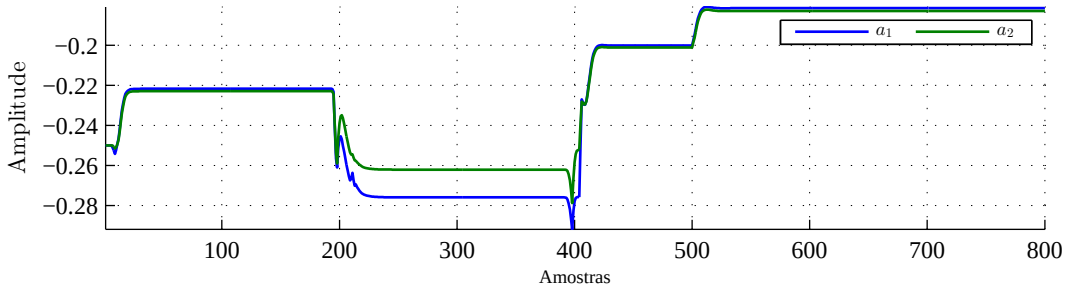


Figura A.7: Coeficientes do modelo, do polinómio $a(z^{-1})$ (3.56) (o coeficiente a_0 não está representado porque assume sempre o valor 1), identificado pelo controlador II.

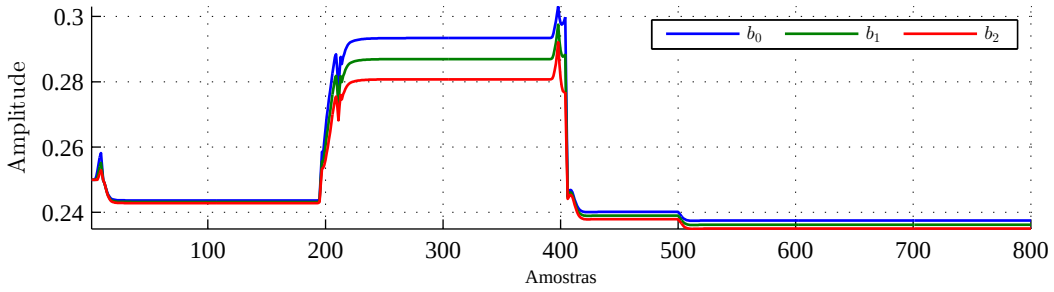


Figura A.8: Coeficientes do modelo, do polinómio $b(z^{-1})$ (3.56), identificado pelo controlador II.

A.3.3 Testes em Ambiente de Simulação - Planta III

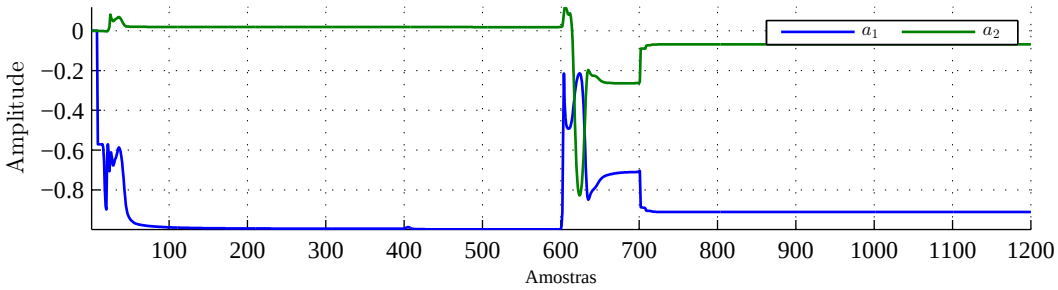


Figura A.9: Coeficientes do modelo, do polinómio $a(z^{-1})$ (3.56) (o coeficiente a_0 não está representado porque assume sempre o valor 1), identificado pelo controlador I.

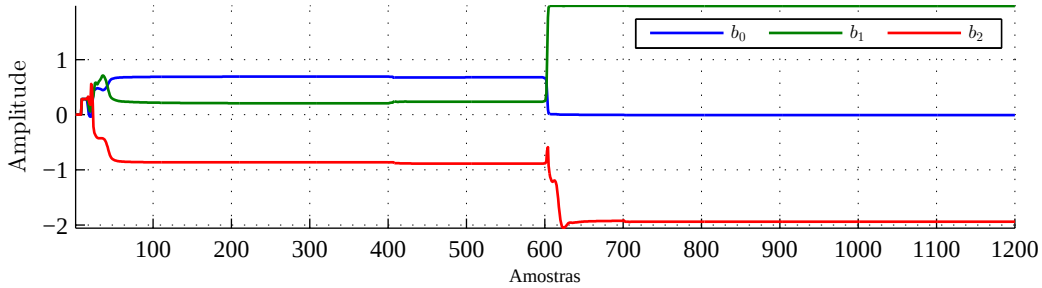


Figura A.10: Coeficientes do modelo, do polinómio $b(z^{-1})$ (3.56), identificado pelo controlador I.

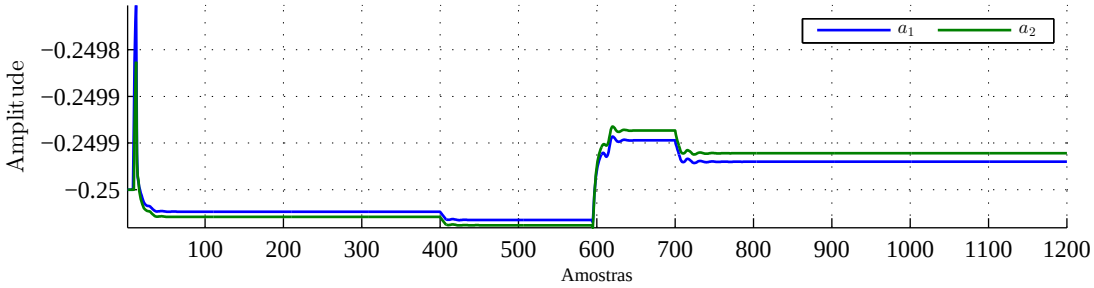


Figura A.11: Coeficientes do modelo, do polinómio $a(z^{-1})$ (3.56) (o coeficiente a_0 não está representado porque assume sempre o valor 1), identificado pelo controlador II.

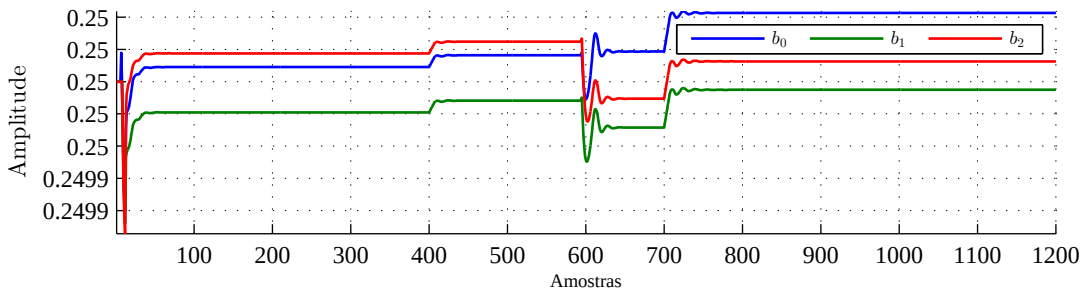


Figura A.12: Coeficientes do modelo, do polinómio $b(z^{-1})$ (3.56), identificado pelo controlador II.

A.3.4 Testes em Laboratório

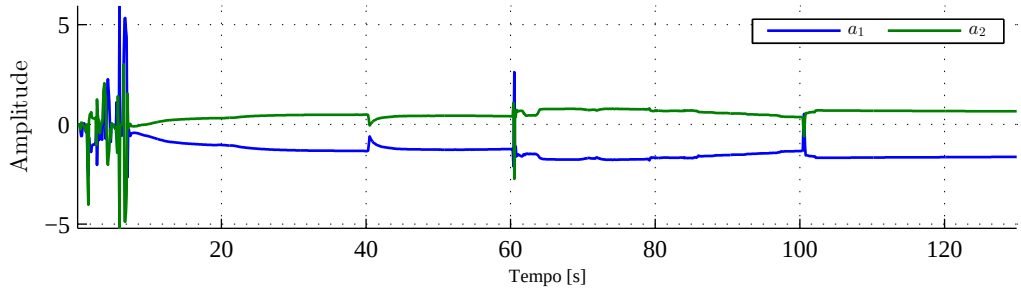


Figura A.13: Coeficientes do modelo, do polinómio $a(z^{-1})$ (3.56) (o coeficiente a_0 não está representado porque assume sempre o valor 1), identificado pelo controlador I.

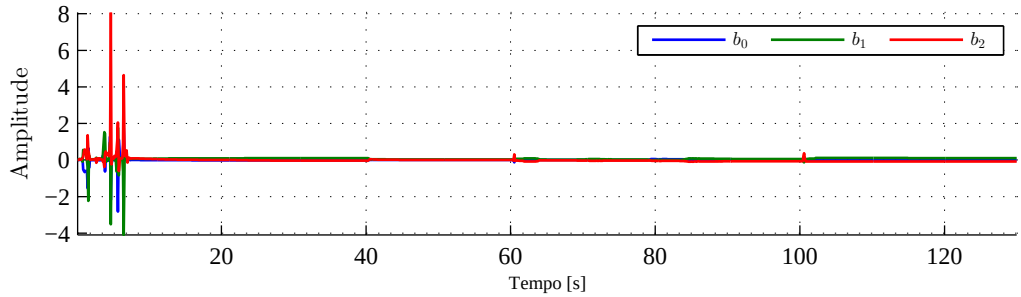


Figura A.14: Coeficientes do modelo, do polinómio $b(z^{-1})$ (3.56), identificado pelo controlador I.

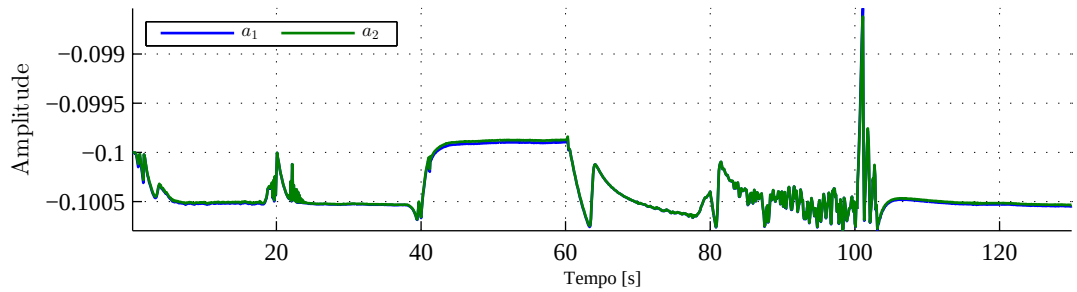


Figura A.15: Coeficientes do modelo, do polinómio $a(z^{-1})$ (3.56) (o coeficiente a_0 não está representado porque assume sempre o valor 1), identificado pelo controlador II.

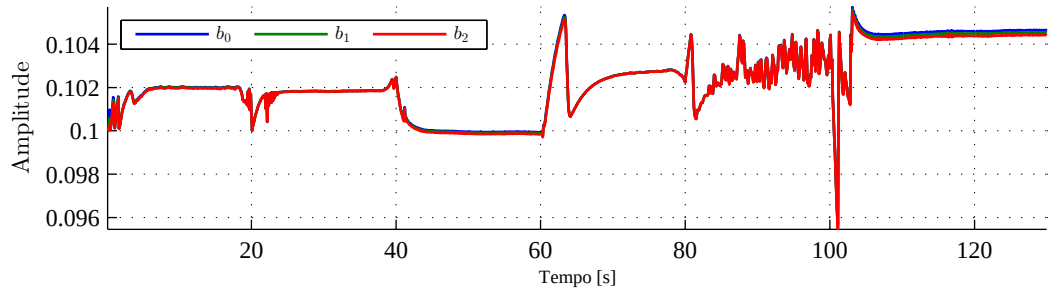


Figura A.16: Coeficientes do modelo, do polinómio $b(z^{-1})$ (3.56), identificado pelo controlador II.