

Centro Federal de Educação Tecnológica de Minas Gerais
Campus Divinópolis
Graduação em Engenharia Mecatrônica

Álan Crístoffer e Sousa

IMPLEMENTAÇÃO DE CONTROLADOR MPC EM UM
SISTEMA A PARÂMETROS DISTRIBUÍDOS VIA SUBSISTEMAS
INTERCONECTADOS



Divinópolis

2018

Álan Crístoffer e Sousa

**IMPLEMENTAÇÃO DE CONTROLADOR MPC EM UM
SISTEMA A PARÂMETROS DISTRIBUÍDOS VIA SUBSISTEMAS
INTERCONECTADOS**

Monografia de Trabalho de Conclusão de Curso apresentada ao Colegiado de Graduação em Engenharia Mecatrônica como parte dos requisitos exigidos para a obtenção do título de Engenheiro Mecatrônico.

Eixo de Formação: Modelagem e Controle de Processos.

Orientador: Prof. Dr. Valter Júnior de Souza Leite
Coorientador: Prof. Dr. Ignacio Rubio Scola



Divinópolis

2018

A MINHA FAMÍLIA, QUE SEMPRE
ME APOIOU NESSA CAMINHADA.

Agradecimentos

Agradeço,
aos meus pais por todo o apoio e confiança em mim depositados;
ao Prof. Valter por toda a paciência e empenho nesses 2 anos de orientação;
ao Prof. Ignacio pelas conversas e esclarecimentos;
aos colegas de trabalho mais próximos: Nelson Figueredo, Ângelo Eugênio, Mariella Maia, Michelle Faria, Anderson Bento, Mário Cipriano, Bernardo Amim e Luis Gustavo a convivência descontraída e as trocas de experiências;
aos demais colegas do Grupo de Modelagem e Controle de Sistemas Mecatrônicos a ótima convivência;
a todos que de alguma forma contribuíram com o meu progresso.

Não há nada como um sonho para criar o futuro.

Victor Hugo

Resumo

Controle preditivo baseado em modelo (MPC — *model predictive control*) é uma técnica avançada de controle que permite a inserção de restrições no sinal de controle e de variação dos estados do sistema na etapa de equacionamento do controlador. Seu objetivo é encontrar a trajetória de controle ótima que respeite as restrições impostas. O MPC já está estabelecido em indústrias que lidam com processos multivariáveis de dinâmica lenta, especialmente na indústria petro-química. Sua formulação mais comum utiliza modelos descrito no espaço de estados, o que requer que todos os estados sejam conhecidos. Como nem sempre é possível medi-los, faz-se necessário o uso de observadores, que são técnicas de estimar os estados do sistema a partir dos sinais de entrada e saída por meio do modelo dinâmico do sistema. Utilizando técnicas de modelagem, como por exemplo os métodos lineares ARMAX (do inglês *autoregressive moving average with exogenous inputs*), pode-se obter um modelo a parâmetros concentrados cujos coeficientes dependem do espaço (SPD — sistema a parâmetros distribuídos). Esses modelos SPD pode ser utilizados por observadores para estimar os estados do sistema em pontos de interesse onde não é possível ou viável inserir sensores. Assim pode-se, por exemplo, medir-se a temperatura na extremidade de um sólido e, através desta, recuperar-se a temperatura em algum ponto no meio do sólido, desde que o modelo dinâmico de propagação seja conhecido. Ao combinar o modelo ARMAX, o controlador MPC e o observador de estados é possível controlar uma variável em um ponto diferente daquele sendo medido. Assim propõe-se o desenvolvimento e implementação de um controlador MPC que utilize modelos SPD para realizar o controle de uma variável estimada em um ponto intermediário de um sistema distribuído espacialmente. Para isso será utilizada a planta presente no Laboratório de Sinais e Sistemas do *campus* V do CEFET-MG. O modelo SPD utilizado é o desenvolvido por Barroso (2017). A implementação é feita utilizando os softwares *Moirai* e *Lachesis*, desenvolvidos *in loco*, que foram atualizados de forma a comunicar com o *hardware* da planta que receberam atualizações visando suas melhorias. Pretende-se com este trabalho, que envolve principalmente as áreas de controle e computação, aprofundar os estudos do Grupo de Modelagem e Controle de Sistemas Mecatrônicos na utilização de modelos SPD e do MPC, bem como facilitar futuros trabalhos nesta e outras plantas.

Palavras-chave: Controle preditivo por modelo, sistema a parâmetros distribuídos, observador de Kalman

Abstract

Model Predictive Control (MPC) is an advanced control technique that allows the insertion of restrictions in the control signal and in the variation of the system states in the controller equation stage. Its goal is to find the optimal trajectory of the control signal that respects the restrictions imposed. The MPC is already established in industries dealing with multivariate, slow dynamics systems, especially in the petrochemical industry. Its formulation most commonly uses models described in state space, what requires all states to be known. As it is not always possible to measure them, observers, which are techniques of estimating states from the input and output signals, are employed. Using modeling techniques, such as ARMAX (autoregressive moving average with exogenous inputs), one can obtain a model with concentrated parameters whose coefficients depend on the space (DPS — distributed parameters system). These DPS models can be used to estimate the states of the system at points of interest where it is not possible or feasible to insert sensors. Thus, one can measure, for example, the temperature at the end of a solid and, through an observer, recover the temperature at some point in the middle of the solid, provided that the dynamic propagation is known. When combining the ARMAX model, the MPC controller and the state observer it is possible to control a variable in a different spacial point from that being measured. Thus, the development and implementation of a MPC controller using DPS to control an estimated variable at an intermediate point of a spatially distributed system is proposed. For this the plant used will be the one present in the Laboratory of Signals and Systems at *campus* V of CEFET-MG. The DPS model used is the one developed by Barroso (2017). The implementation is done using the Moirai and Lachesis softwares, developed *in loco*, which have been updated in order to communicate with the hardware of the plant. The software received updates aimed at its improvements. It is intended to this work, which mainly involves the areas of control and computation, to deepen the studies of the Group of Modeling and Control of Systems Mechatronics in the use of DPS and MPC models, as well as facilitate future work in this and other plants.

Keywords: Model predictive control, distributed parameters system, Kalman filter

Sumário

1	Introdução	1
1.1	Definição do Problema	2
1.2	Objetivos	3
1.3	Motivação	4
1.4	Estado da arte	4
2	Revisão da literatura	7
3	Metodologia	9
4	Fundamentação Teórica	11
4.1	Sistemas	11
4.1.1	Tanques	11
4.1.2	Forno	11
4.2	Controlador preditivo por modelo	12
4.2.1	Princípio de funcionamento	12
4.2.2	Restrições de parâmetros	16
4.2.3	Funções de Laguerre	19
4.3	Sistemas a parâmetros distribuídos	25
4.3.1	Modelagem do forno por LPVs interconectados	27
4.4	Observadores	29
4.4.1	Observabilidade	29
4.4.2	Filtro de Kalman	30
4.4.3	Observador exponencial com fator de esquecimento	31

4.5	PI por síntese direta	31
4.6	Índices de desempenho	32
5	Resultados e Discussões	33
5.1	Estudos teóricos	33
5.1.1	Controle preditivo por modelo	33
5.1.2	Modelagem térmica e SPD	35
5.2	Modificação do hardware e implantação da plataforma	36
5.3	Calibração	36
5.4	Síntese e validação do observador	38
5.5	Escolha de N_p e N_c	42
5.6	PI por síntese direta	44
5.7	Melhorias na plataforma	45
6	Considerações finais	49
	Referências	51

Lista de figuras

Figura 1.1 – Sensores da planta	3
Figura 4.1 – Forno utilizado	12
Figura 4.2 – Subsistemas interconectados	25
Figura 4.3 – Subsistema LPV	26
Figura 4.4 – Subsistemas LPV interconectados	27
Figura 5.1 – Sistema de tanques comunicantes	34
Figura 5.2 – Simulação do modelo dos tanques com controlador MPC	34
Figura 5.3 – MPC aplicado ao sistema real	35
Figura 5.4 – SPD simulado	35
Figura 5.5 – SPD simulado com observador exponencial	36
Figura 5.6 – Filtro de Kalman com ganho estático e erro de <i>offset</i> e de seguimento .	39
Figura 5.7 – Filtro de Kalman com matriz $Q = 0$	39
Figura 5.8 – Filtro de Kalman com offset	40
Figura 5.9 – Filtro de Kalman com todas as correções aplicadas	41
Figura 5.10–MPC no Filtro de Kalman com todas as correções aplicadas	42
Figura 5.11–Resposta das 840 combinações de N_p e N	43
Figura 5.12–Resposta do MPC com $N_p = 86$ e $N = 8$	44
Figura 5.13–Comparação das respostas dos MPCs e PIIs	45

Lista de tabelas

Tabela 5.1 – Valores utilizados na calibração	38
Tabela 5.2 – Índices de desempenho	45

Lista de acrônimos e notações

ARMAX	Modelo autoregressivo de média variável com entradas exógenas, do inglês <i>Autoregressive-moving-average with exogenous inputs model</i>
DMC	Controle Dinâmico por Matriz, do inglês <i>Dynamic Matrix Control</i>
EDP	Equação Diferencial Parcial
IMC	Controle por Modelo Interno, do inglês <i>Internal Model Control</i>
LPV	Variação linear de parâmetros, do inglês <i>Linear Parameter Varying</i>
MPC	Controle preditivo por modelo, do inglês <i>Model Predictive Control</i>
SPC	Sistema a Parâmetros Concentrados
SPD	Sistema a Parâmetros Distribuídos

Capítulo 1

Introdução

Controle preditivo por modelo (MPC — *Model Predictive Control*) é uma técnica de controle avançada que utiliza o modelo do sistema para prever a saída em momentos futuros e, com isso, gerar uma estratégia de controle que otimize algum critério quadrático selecionado. Esta técnica pode ser aplicada em modelos contínuos quanto discretos. A abordagem discreta no tempo é usada neste trabalho, pois melhor ilustra o funcionamento do MPC, além de fornecer ao controlador mais tempo para o processamento do próximo sinal de controle. Note que a restrição de tempo para processamento do MPC pode ser crítica em algumas aplicações, pois um problema de otimização deve ser resolvido a cada iteração.

Em um modelo discreto o sinal é amostrado a cada Δt segundos. O modelo não varia mais diretamente com o tempo, mas sim com a amostragem k , que se associa com o tempo como $t = k\Delta t$. Assim, em um modelo discreto, ao se passar do tempo 0 para o tempo Δt ou do tempo Δt para o tempo $2\Delta t$, dizemos que se passou um instante.

Utilizando um modelo discreto pode-se então prever a saída do sistema para um instante futuro, se as entradas que foram aplicadas ao sistema nos instantes anteriores forem conhecidas. Pode-se então prever as saídas dos próximos N_p instantes (chamado horizonte de predição) e utilizar técnicas de otimização para obtermos os próximos N_c sinais de controle (chamado horizonte de controle) ótimos que levarão o sistema à referência. Ao utilizar apenas o primeiro sinal de controle e recalcular os sinais de controle ótimos a todo instante, o controlador será capaz de rejeitar ruídos e reagir a mudanças, como a da referência. Esta técnica é chamada de Controle por Horizonte Recessivo e é a base do MPC (WANG, 2009).

A discretização foi feita no tempo, pois altera-se o modelo para que esse não mais dependa do tempo mas sim de um instante. Sistemas que dependem apenas do tempo são chamados de sistemas a parâmetros concentrados (SPC). No entanto existem sistemas que não variam apenas no tempo, mas também no espaço. Um exemplo é a transferência de calor em uma barra metálica, em que a temperatura depende tanto do tempo quanto do ponto no espaço em que se faz a medição. Sistemas que dependem de uma variável espacial são chamados de sistemas a parâmetros distribuídos (SPD). Sua modelagem é feita utilizando derivadas parciais e equações diferenciais parciais (EDP) (CALDEIRA, 2017).

Para ambos tipos de modelos é comum usar a modelagem por espaço de estados ao se trabalhar com MPC. Isso permite projetar mais facilmente sistemas com múltiplas entradas e múltiplas saídas. No entanto, ao se trabalhar com espaço de estados, é necessário que o valor de todos os estados sejam conhecidos, o que nem sempre é possível, seja por limitações econômicas ou até mesmo físicas. Para resolver esse problema utilizam-se observadores.

O observador é um sistema que estima os estados de um modelo baseado nas entradas e saídas do mesmo. O observador de Kalman, também conhecido como filtro de Kalman, é uma formulação que permite estimar sinais de forma ótima na presença de ruídos, seja na leitura da saída ou dos próprios estados (WANG, 2009).

1.1 Definição do Problema

A Figura 1.1 mostra os sensores do forno presente no Laboratório de Sinais e Sistemas. Nela podemos ver os sensores S_1 a S_5 e o fluxo de ar quente q . Imagine que queiramos controlar a temperatura na posição onde encontra-se o sensor S_3 , mas que apenas o sensor S_5 esteja funcionando. Com um modelo SPD podemos ter dois modelos SPC: um que modele o fluxo de temperatura a partir do atuador até o sensor S_3 e um outro que modele de S_3 até S_5 , ambos interconectados. Desta forma, podemos utilizar um observador e a leitura do sensor S_5 para estimar a temperatura no sensor S_3 , e controlá-la mesmo sem fazer sua medição direta.

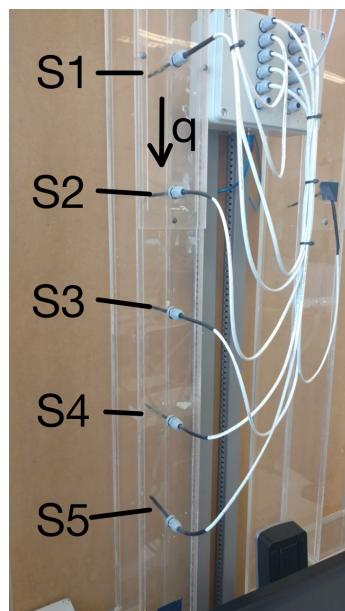


Figura 1.1 – Sensores da planta

1.2 Objetivos

O objetivo desse trabalho é desenvolver um controlador do tipo MPC com restrições na amplitude do sinal de controle e estados do sistema com modelo SPD utilizando observadores do tipo Kalman e implementá-lo utilizando e adaptando a plataforma de controle desenvolvida pelo proponente para uso no forno do laboratório de sinais e sistemas. Para alcançar esse objetivo as seguintes ações foram realizadas:

- modificar a eletrônica da planta: instalar circuitos microprocessados com o intuito de controlar o acionamento do forno, fornecendo um caminho alternativo ao acionamento que já está implementado e permitindo a integração com a plataforma de controle;
- modificar a plataforma de controle: desenvolver driver específico para o forno, de forma a tornar o acionamento dos atuadores e a leitura dos sensores mais simples e direto para os futuros usuários;
- desenvolver o controlador MPC: utilizar o modelo SPD desenvolvido por Barroso (2017) para desenvolver um controlador MPC com restrições na entrada e saída de forma a controlar a temperatura em um ponto diferente daquele onde está fisicamente instalado o sensor;

- implementar o controlador: utilizar a linguagem Python e a plataforma de controle para implementar o controlador e executar o controle da planta;
- comparar o desempenho do controlador: usar índices de desempenho para comparar o desempenho do controlador MPC com controlador PI ou PID, desenvolvido utilizando síntese direta;
- realizar melhorias na plataforma: ao utilizar a plataforma como usuário, espera-se encontrar dificuldades, erros e novas ideias que serão corrigidos/implementadas, como, por exemplo, recuperação de mensagens de erro salvas no banco de dados e exibição para o usuário e verificação de sintaxe de código digitado pelo usuário.

1.3 Motivação

A principal motivação para o trabalho é poder fazer o controle de um processo sem medir diretamente a variável controlada. Isto é interessante pois nem sempre é possível medir diretamente a grandeza que se deseja controlar, seja por restrições físicas (como, por exemplo, colocar um sensor no centro de um alto-forno) ou por questões financeiras.

A escolha do controlador MPC se tornou interessante por ser uma técnica avançada de controle que ainda não é amplamente estudada no *campus V*. Assim, além da possibilidade de estudar uma técnica avançada, este trabalho contribui por ser o primeiro trabalho a utilizar tal controlador no Grupo de Modelagem e Controle de Sistemas Mecatrônicos.

1.4 Estado da arte

O uso de controladores MPC com modelos SPD não é novo. Sua maior aplicação, no entanto, continua sendo na área da química, fabricação de aço e, principalmente na indústria petroquímica. Ele foi desenvolvido inicialmente para controle de processos químicos em que o equipamento é caro, o processo é lento e contém várias entradas, saídas e estados que devem ser restringidas em amplitude e taxa de variação (CAIRANO, 2012).

Este cenário está mudando e outras indústrias — com modelos com poucas entradas e saídas, custo mais baixo e dinâmica mais rápida — estão adotando os controladores MPC. No entanto, isso apresenta alguns desafios no desenvolvimento de controladores, pois o *framework* MPC não foi desenvolvido para tais sistemas e possui limitações como,

por exemplo, o alto custo computacional. Trabalhos já estão sendo desenvolvidos para minimizar ou sanar tais problemas (CAIRANO, 2012).

Em ambientes industriais é comum o uso de computadores lógicos programáveis para fazer a interface com o hardware. No meio acadêmico, no entanto, utiliza-se softwares como o *MATLAB* e *LabVIEW* para a prototipagem rápida. Isto requer que o pesquisador recrie a interface gráfica e controle a aquisição de dados manualmente, o que consome tempo que poderia ser gasto com a pesquisa.

Assim, uma plataforma que faça a interface com o hardware e possibilite a execução de um controlador de forma transparente permite que o pesquisador se concentre nestes estudos e não se preocupe com detalhes da implementação da aquisição dos dados. Tal ferramenta também permitiria ao pesquisador interagir com CLPs, desenvolvendo e testando controladores de forma fácil em uma linguagem simples (Python) sem se preocupar com todas as nuâncias das linguagens LADDER utilizadas nos CLPs, podendo implementar seu controlador nesta linguagem limitada apenas ao final, quando este tiver sido devidamente testado e otimizado.

A aplicação de uma plataforma para controle em todos as plantas do laboratório permite que os usuários possam migrar de uma planta para outra sem dificuldades. Também permite que controladores e observadores desenvolvidos sejam testados em diferentes plantas com poucas modificações, já que as interfaces seriam as mesmas em todas elas.

Capítulo 2

Revisão da literatura

A malha fechada foi formalizada matematicamente em 1868 por Maxwell. Black (1934) demonstrou a utilidade da realimentação negativa nos laboratórios da Bell onde aplicou a malha fechada com realimentação negativa nos amplificadores das linhas de transmissão.

No entanto, ao ampliar o sinal, foi ampliado também o ruído. Para resolver este problema iniciaram-se estudos para utilizar técnicas descritas por Laplace, Fourier e Cauchy, que propunham o uso do domínio da frequência para modelar sistemas dinâmicos. O problema passou a ser filtrar as frequências desejadas de forma a não aplicar o ganho da malha fechada no ruído (BRYSON; HO; SIOURIS, 1979).

Nyquist e Bode (1940) criaram ferramentas na década de 1930 que permitem quantificar a estabilidade de um sistema em malha fechada. O uso de tais ferramentas facilitou o desenvolvimento de controladores que alteram a dinâmica do sistema para uma dinâmica desejada. A adição de ferramentas estocásticas permitiu também o desenvolvimento de filtros ótimos.

Com estas técnicas estabeleceu-se o que chamamos de controle clássico. Este é caracterizado pelo desenvolvimento no domínio da frequência e pelo uso de técnicas projetadas para que seus cálculos fossem feitos à mão, ou no máximo com o uso de tabelas (DORF; BISHOP, 2010).

Embora o uso da análise no domínio da frequência tenha sido útil principalmente para sua época, começaram a aparecer sistemas mais complexos, com várias entradas e saídas, ordem elevada e não lineares, que não eram bem descritos no domínio da frequência. Nesta época, estudos no domínio do tempo, utilizando as equações diferenciais, começaram a aparecer, principalmente para resolver problemas da indústria aeroespacial (LYAPUNOV,

1892).

Várias técnicas foram desenvolvidas nesta época. Uma se destaca: DMC — *Dynamic Matrix Control*. Ela visa resolver os problemas de controle multivariável com restrições, típico nas indústrias química e petroquímica. Antes do desenvolvimento dessa técnica o controle era feito por várias malhas em cascata (CUTLER; RAMAKER, 1980).

O MPC gerou impacto na indústria, especialmente a petroquímica (MORARI; LEE, 1999). Provavelmente não há nenhuma empresa extratora de petróleo que não utilize esta técnica ou uma derivada. O desenvolvimento do MPC se deu como uma tentativa de entender o DMC, que parecia desafiar a análise teórica tradicional por ser formulado de maneira não convencional. Por exemplo, outra técnica, a IMC — *Internal Model Control* — falhou em explicar o funcionamento do DMC mas acabou ajudando no desenvolvimento do controle robusto (MORARI; LEE, 1999).

Hoje em dia o controle preditivo pode ser formulado no espaço de estados, mesmo que seja possível sua formulação por, por exemplo, funções de transferência. O controle preditivo por modelo é uma das formas de controle desenvolvidas com a ideia de buscar a trajetória ótima de controle, que observa-se sempre existir em sistemas dinâmicos. Para isso é utilizada a otimização custo, normalmente uma função de energia (MORARI; LEE, 1999; BRYSON; HO; SIOURIS, 1979).

O grande problema destes controladores hoje é o custo computacional quando se necessita de um controlador *on-line*, o que normalmente é o que se deseja, pois estes retornam resultados melhores na presença de restrições. Os estudos atuais seguem duas linhas: melhorar os *solvers* de forma a melhorar o desempenho ou encontrar maneiras de pré-processar as restrições, para que apenas multiplicações matriciais sejam realizadas em tempo real. A primeira linha de pesquisa é a mais explorada hoje em dia (WANG, 2009; ZHANG, 2016).

Capítulo 3

Metodologia

Para que a plataforma de controle possa se comunicar com o forno do laboratório de sinais e sistemas foi necessário modificar o circuito de acionamento da mesma. Para isso foram utilizados circuitos microcontrolados que se comunicam com o circuito de acionamento e sensoriamento presente na mesma.

Esse novo circuito foi programado para se comunicar com a plataforma de controle. Um *driver* foi escrito para a mesma para permitir o controle dos atuadores e leitura dos sensores de forma transparente e específica para este sistema. O *driver* abstrai a interface com o hardware de forma que o usuário não precise se preocupar com detalhes da implementação, como número de porta e pino, tensões de operação, etc.

Também foi necessário calibrar todos os sensores e atuadores, além de conferir o funcionamento dos circuitos elétricos. Todo este procedimento foi feito usando-se a plataforma de controle para a aquisição de dados. Para isso foi necessário estudar os trabalhos de Barroso (2015) e Silva (2014).

Os modelos levantados por Barroso (2017) foram validados no sistema físico. Isto foi necessário para verificar que não houve alterações significativas no sistema desde o seu desenvolvimento. Para o uso destes modelos foi necessário estudar sobre sistemas a parâmetros distribuídos e observadores de Kalman.

O controlador MPC utilizado é discreto no tempo e baseado em modelo em espaço de estados, logo foi necessário estudar sobre controle digital e modelagem em espaço de estados, além da própria técnica de controle. Para isto foram utilizados, principalmente, os livros do Dorf e Bishop (2010), do Liberty (1972), do Wang (2009) e as notas de aula do Professor Patwardhan (2014).

Para o desenvolvimento do controlador foi utilizado o modelo SPD além de determinadas restrições do sinal de controle. Os estados utilizados no controlador são os provenientes do observador do tipo Kalman utilizado pelo Barroso (2017).

Para comparar o desempenho do controlador MPC foram desenvolvidos controladores PI por síntese direta. Utilizou-se os índices IAE e $IA\Delta X$ para realizar uma comparação quantitativa dos controladores.

Todos os controladores foram implementados utilizando a plataforma de controle e a linguagem de programação Python, assim como todos os testes que que usem o acionamento do forno e/ou medição de seus termômetros. Desta forma foram corrigidos na plataforma todos os problemas percebidos na mesma do ponto de vista de usuário. Também foram implementadas várias novas funcionalidades, visando seu aperfeiçoamento.

Capítulo 4

Fundamentação Teórica

Para a realização deste trabalho é necessário conhecimento nas áreas de controle preditivo por modelo e sistemas a parâmetros distribuídos. Este capítulo visa explicar sucintamente esses conceitos.

4.1 Sistemas

Os sistemas de tanques comunicantes e forno utilizados são descritos a seguir.

4.1.1 Tanques

O sistema de tanques é composto por quatro tanques comunicantes. Os dois tanques superiores foram utilizados para teste do controlador. Trata-se de dois recipientes de 200 l com uma válvula entre ambos, que permite a passagem de água de um para o outro. Há também válvulas na saída dos tanques, que permitem que a mesma flua para os tanques inferiores. O atuador bombeia a água para o tanque T1. Essa flui para o tanque T2 e sai do sistema. O objetivo de controle é a altura da coluna d'água no tanque T2.

4.1.2 Forno

O forno presente no Laboratório de Sinais e Sistemas pode ser visto na Figura 4.1. Nela estão demarcados os principais elementos do sistema. Ele é composto por um túnel de acrílico onde o ar entra forçadamente por um *cooler* em uma de suas extremidades. No percurso há 3 resistências, denominadas $R1$, $R2$ e $R3$ e 10 sensores, $S1$ a $S10$.

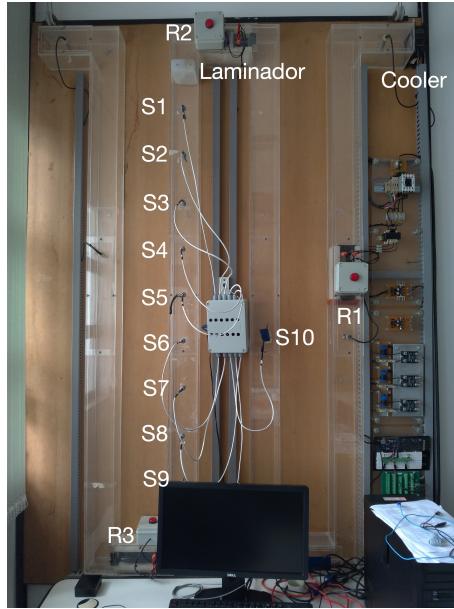


Figura 4.1 – Forno utilizado

Nesse trabalho apenas a resistência $R2$ é utilizada. O sensor $S10$ foi posto antes da mesma para medir a temperatura ambiente. O sensor $S9$, mais distante da resistência, é utilizado para fins de controle, enquanto os outros são medidos apenas para fim de validação do observador, conferindo sua medida com a estimativa do observador.

Um laminador foi inserido logo após a resistência para que o perfil do fluxo de ar nos sensores não seja turbulento. Isso se fez necessário pois o fluxo turbulento torna a relação entre as leituras dos sensores não determinístico.

4.2 Controlador preditivo por modelo

Controle preditivo por modelo é uma técnica comumente utilizada pelas indústrias petroquímica, farmacêutica e do aço. Ela utiliza o modelo para prever a saída do sistema e, com isso, otimizar a trajetória do sinal de controle (WANG, 2009).

4.2.1 Princípio de funcionamento

A formulação MPC mais comum é a em espaço de estados. Um modelo em espaço de estados tem a forma

$$\begin{aligned}\dot{x}(t) &= Ax(t) + Bu(t), \\ y(t) &= Cx(t) + Du(t),\end{aligned}\tag{4.1}$$

em que $A \in R^{n \times n}$ é a matriz dinâmica do sistema, $B \in R^{n \times m}$ e a matriz de entrada, $C \in R^{m \times n}$ é a matriz de saída. $D \in R^{m \times m}$ é normalmente 0. $x(t) \in R^{m \times 1}$ é o vetor de estados e $u(t) \in R^{m \times 1}$ são as entradas do sistema no instante t .

Neste trabalho será utilizada a formulação discreta do controlador preditivo. Esta formulação tem a vantagem de poder ser implementada facilmente em dispositivos digitais, além de permitir certo controle do tempo de amostragem.

Ao discretizar um sistema contínuo no tempo altera-se sua formulação para que a matriz de dinâmica do sistema não mais produza a variação do estado, mas sim o valor do estado após Δt segundos. Um intervalo de tempo de Δt segundos é chamado de instante, sendo $t = k\Delta t$. Para discretizar um modelo em espaço de estados usa-se a equação

$$x(k+1) = e^{AT}x(k) + \int_0^T e^{At}dtBu(k). \quad (4.2)$$

Ao discretizar o modelo (4.1) utilizando (4.2) obtem-se

$$\begin{aligned} x(k+1) &= A_dx(k) + B_d u(k), \\ y(k) &= C_dx(k) + D_d u(k). \end{aligned} \quad (4.3)$$

Daqui em diante as matrizes A , B , C e D irão se referir ao sistema discreto no tempo, já que não mais será utilizado o sistema contínuo no tempo.

Uma vez obtido o modelo discreto o mesmo deve ser aumentado com um integrador. Para inserir um integrador no sistema utiliza-se a equação

$$\begin{bmatrix} \Delta x(k+1) \\ y(k+1) \end{bmatrix} = \begin{bmatrix} A & 0_m^T \\ CA & 1 \end{bmatrix} \begin{bmatrix} \Delta x(k) \\ y(k) \end{bmatrix} + \begin{bmatrix} B \\ CB \end{bmatrix} \Delta u(k). \quad (4.4)$$

Observe que os estados e o sinal de entrada foram substituídos por variações dos mesmos. Isso é possível pois

$$\Delta x(k+1) = A\Delta x(k) + B\Delta u(k), \quad (4.5)$$

e é também necessário para o funcionamento do controlador. O motivo para esta reformulação será percebido mais adiante.

Uma vantagem do modelo em espaço de estados discreto é que é possível calcular estados futuros facilmente, sem a necessidade de se resolver uma equação diferencial.

Utilizando deste fato pode-se utilizar o modelo (4.3) para obter uma equação da próxima saída do sistema, através da substituição de $x(k)$ por $x(k+1)$ na equação da saída:

$$y(k+1) = C(Ax(k) + Bu(k)). \quad (4.6)$$

Ao realizar esta substituição uma segunda vez, obtém-se a saída do sistema após 2 instantes:

$$y(k+2) = C(A(Ax(k) + Bu(k)) + Bu(k+1)). \quad (4.7)$$

Com uma nova substituição obtém-se a saída após 3 instantes:

$$y(k+3) = C(A(A(Ax(k) + Bu(k)) + Bu(k+1)) + Bu(k+2)). \quad (4.8)$$

Tem-se então que a saída depende do estado atual e da trajetória de controle. Reescrevendo estas equações de forma matricial para qualquer número N_p de saídas futuras e N_c de entradas futuras, tem-se

$$Y = Fx(k) + \Phi U \quad (4.9)$$

em que

$$Y = \begin{bmatrix} y(k+1) \\ y(k+2) \\ y(k+3) \\ \vdots \\ y(k+N_p) \end{bmatrix}; F = \begin{bmatrix} CA \\ CA^2 \\ CA^3 \\ \vdots \\ CA^{N_p} \end{bmatrix}; U = \begin{bmatrix} u(k+1) \\ u(k+2) \\ u(k+3) \\ \vdots \\ u(k+N_c) \end{bmatrix}; \quad (4.10)$$

$$\Phi = \begin{bmatrix} CB & 0 & 0 & \dots & 0 \\ CAB & CB & 0 & \dots & 0 \\ CA^2B & CAB & CB & \dots & 0 \\ \vdots & & & & \\ CA^{N_p-1}B & CA^{N_p-2}B & CA^{N_p-3}B & \dots & CA^{N_p-N_c}B \end{bmatrix}$$

No entanto, para que essa formulação funcione devemos ter $N_c = N_p$. Isto no entanto não é desejado. Por questões de implementação computacional e matemáticas é preferível que $N_c \ll N_p$. Isso se torna possível se substituirmos U por ΔU , o que é feito na matriz aumentada. Essa substituição é possível, pois, devido à presença do integrador, um valor de $\Delta U = 0$ significa que o último sinal aplicado é mantido.

Para compreender melhor este conceito tome com exemplo um sistema de tanques o qual deseja-se controlar o nível alterando-se a vazão de uma bomba. Para todo $N_p > N_c$ tem-se uma vazão de 0, ou seja, a bomba seria desligada. Isso faria com que o nível caísse toda vez que o sistema alcançasse o equilíbrio. Utilizando ΔU o valor zero significa manter a última vazão apicada, o que faz com que o sistema permaneça em equilíbrio uma vez que o sinal de controle seja zerado.

MPC é uma técnica baseada em otimização. Problemas de otimização são resolvidos encontrando-se o máximo ou mínimo de alguma função. Deve-se então definir uma função que, ao ser minimizada, terá como resultado a trajetória ótima. Este tipo de função é chamada de função de custo e normalmente utiliza-se em MPC uma função quadrática, o que garante a convergência. Escrevendo a função com o intuito de minimizar o erro em regime permanente e a variação do sinal de controle, tem-se

$$J = (R_s - Y)^T (R_s - Y) + \Delta U^T \bar{R} \Delta U, \quad (4.11)$$

em que R_s é o vetor referência com a mesma dimensão do vetor Y e \bar{R} é uma variável que pode ser usada para ponderar a variação do sinal de controle. \bar{R} maior implica em variações menores.

Substituindo (4.9) em (4.11) tem-se

$$J = (R_s - Fx(k))^T (R_s - Fx(k)) - 2\Delta U^T \Phi^T (R_s - Fx(k)) + \Delta U^T (\Phi^T \Phi + \bar{R}) \Delta U. \quad (4.12)$$

Derivando em função da variação do sinal de controle, ΔU :

$$\frac{\partial J}{\partial \Delta U} = -2\Phi^T (R_s - Fx(k)) + 2(\Phi^T \Phi + \bar{R}) \Delta U. \quad (4.13)$$

Igualando $\frac{\partial J}{\partial \Delta U}$ a zero e isolando ΔU :

$$\Delta U = (\Phi^T \Phi + \bar{R})^{-1} \Phi^T (R_s - Fx(k)). \quad (4.14)$$

Ao utilizar $N_c = N_p$, é possível, através da equação (4.14), encontrar os valores ótimos de ΔU que farão o sistema seguir uma referência. No entanto, o controlador não será capaz de rejeitar ruídos. Aplicando apenas o primeiro sinal de controle, medindo ou estimando os estados do sistema e recalculando a trajetória a cada instante é possível contornar este problema.

Esta é a formulação básica do MPC, porém ela não fornece o que pode ser considerada a maior vantagem desta técnica de controle: restrição do sinal de controle, da saída, de estados ou de suas variações. Para isto é necessário voltar na função de custo e resolvê-la de uma maneira que permita a inserção de restrições no equacionamento.

4.2.2 Restrições de parâmetros

Os parâmetros mais comumente restringidos são a saída do sistema, o sinal de controle e a variação do sinal de controle. Estas restrições tem a forma de inequações e podem ser escritas como

$$U^{\min} \leq U \leq U^{\max}, \quad (4.15)$$

$$Y^{\min} \leq Y \leq Y^{\max}, \quad (4.16)$$

e

$$\Delta U^{\min} \leq \Delta U \leq \Delta U^{\max}. \quad (4.17)$$

Usando a equação (4.17) como exemplo, pode-se reescrevê-la como duas inequações:

$$\begin{aligned} -\Delta U &\leq -\Delta U^{\min} \\ \Delta U &\leq \Delta U^{\max}. \end{aligned} \quad (4.18)$$

Em forma matricial,

$$\begin{bmatrix} -I \\ I \end{bmatrix} \Delta U \leq \begin{bmatrix} -\Delta U^{\min} \\ \Delta U^{\max} \end{bmatrix}. \quad (4.19)$$

Para os casos de Y e U , deve-se fazer as devidas substituições de forma que a variável ΔU apareça na inequação. Esta substituição é necessária pois a técnica de otimização irá buscar a solução ótima para apenas uma variável (ΔU), sendo necessário que todas as restrições sejam funções desta variável.

Para Y deve-se substituir a equação (4.9) em (4.16), isolar $\Phi\Delta U$ e reescrevê-la de forma matricial:

$$\begin{bmatrix} -\Phi \\ \Phi \end{bmatrix} \Delta U \leq \begin{bmatrix} -Y^{\min} + Fx(k) \\ Y^{\max} - Fx(k) \end{bmatrix}. \quad (4.20)$$

Para a restrição de U , lembrando que $u(k) = u(k-1) + \Delta u(k)$, tem-se

$$\begin{bmatrix} u(k) \\ u(k+1) \\ \vdots \\ u(k+N_c-1) \end{bmatrix} = \begin{bmatrix} I \\ I \\ \vdots \\ I \end{bmatrix} u(k-1) + \begin{bmatrix} I & 0 & \dots & 0 \\ I & I & \dots & 0 \\ \vdots & & & \\ I & I & \dots & I \end{bmatrix} \begin{bmatrix} \Delta u(k) \\ \Delta u(k+1) \\ \vdots \\ \Delta u(k+N_c-1) \end{bmatrix}, \quad (4.21)$$

que pode ser reescrita como

$$\begin{bmatrix} -C_1 u(k-1) - C_2 \Delta U \\ C_1 u(k-1) + C_2 \Delta U \end{bmatrix} \leq \begin{bmatrix} -U^{\min} \\ U^{\max} \end{bmatrix}. \quad (4.22)$$

Reduzindo estas restrições em uma única inequação matricial, obtem-se

$$\begin{bmatrix} M_1 \\ M_2 \\ M_3 \end{bmatrix} \Delta U \leq \begin{bmatrix} N_1 \\ N_2 \\ N_3 \end{bmatrix}, \quad (4.23)$$

sendo

$$\begin{aligned} M_1 &= \begin{bmatrix} -C_2 \\ C_2 \end{bmatrix}; N_1 = \begin{bmatrix} -U^{\min} + C_1 u(k-1) \\ U^{\max} + C_1 u(k-1) \end{bmatrix}; M_2 = \begin{bmatrix} -I \\ I \end{bmatrix}; \\ N_2 &= \begin{bmatrix} -\Delta U^{\min} \\ \Delta U^{\max} \end{bmatrix}; M_3 = \begin{bmatrix} -\Phi \\ \Phi \end{bmatrix}; N_3 = \begin{bmatrix} -Y^{\min} + Fx(k) \\ Y^{\max} - Fx(k) \end{bmatrix}. \end{aligned} \quad (4.24)$$

Pode-se inserir outras restrições, como em estados, por exemplo, seguindo essa lógica de formulação. Obtem-se então todas as restrições descritas na forma

$$M\Delta U \leq \gamma. \quad (4.25)$$

Este formato de restrições é comumente utilizado em problemas de otimização. Os métodos de programação quadrática, multiplicadores de lagrange e até mesmo o método

SIMPLEX usam restrições escritas nesse formato. Deve-se então escolher um método de otimização e ajustar a função de custo à aquela necessária para usar o método. Como a função de custo é quadrática, convém escolher um método que seja capaz de otimizar problemas quadráticos.

O método de programação quadrática tem uma função de custo, dada por

$$J = \frac{1}{2}x^TEx + x^TF, \quad (4.26)$$

que se assemelha à que foi definida, como pode ser visto na equação (4.26). Vale notar que as variáveis e constantes desta função não tem relação com as variáveis definidas anteriormente, mas são as comumente utilizadas por matemáticos ao trabalhar com estes métodos.

Reescrevendo (4.11) no formato de (4.26), encontra-se

$$E = \Phi^T\Phi + \bar{R} \quad (4.27)$$

$$F = \Phi^T(Fx - R_s), \quad (4.28)$$

no qual E e F à esquerda da igualdade vem da equação (4.26) e Φ , F , R_s e \bar{R} à direita da igualdade da formulação do MPC.

Explicar o funcionamento do método de programação quadrática foge do escopo deste trabalho. Existem bibliotecas que implementam este método em MATLAB e Python como, por exemplo, a CVXOPT¹. Para usar o *solver* de programação quadrática definido nessas bibliotecas deve-se definir as matrizes E , F , M e γ .

Analizando a equação (4.23) pode-se ver que suas dimensões não são iguais, isto é, a matriz M não é quadrada. Isso ocorre pois é possível ter mais restrições do que variáveis. Isso não é um problema já que as restrições são inequações, logo uma mais de uma restrição pode ser satisfeita se tentar satisfazer apenas uma delas, mais restritiva.

Em outras palavras, apenas um subconjunto de restrições estará ativo em um dado momento. Existem técnicas para determinar qual é o conjunto ativo. Isto, no entanto, não é necessário, já que o *solver* irá encontrar a mesma solução para o conjunto completo que para o subconjunto ativo de restrições. A vantagem de encontrar o conjunto ativo é reduzir o custo computacional da otimização.

¹ <https://cvxopt.org/>

4.2.3 Funções de Laguerre

O modelo aumentado adotado até aqui torna os cálculos extremamente onerosos computacionalmente conforme aumentamos o horizonte de controle. Teoricamente os resultados retornados seriam os mesmos independente da dimensão do sistema. No entanto, diferenças numéricas e a presença de ruídos e incertezas fazem com que os resultados não sejam iguais. Torna-se então interessante que o modelo utilize um horizonte de controle maior. Uma solução adotada é o uso de funções de Laguerre, que diminui o número de parâmetros necessários para descrever uma grande janela de controle.

Redes de Laguerre discretas no tempo modelam um sinal utilizando uma série de impulsos. Retomando o vetor de variações do sinal de controle

$$\Delta U = \begin{bmatrix} \Delta u(k_i) & \Delta u(k_i + 1) & \Delta u(k_i + 2) & \dots & \Delta u(k_i + N_c - 1) \end{bmatrix}^T, \quad (4.29)$$

podemos escrever os elementos dentro de ΔU como

$$\Delta u(k + i) = \begin{bmatrix} \delta(i) & \delta(i - 1) & \dots & \delta(i - N_c + 1) \end{bmatrix} \Delta U, \quad (4.30)$$

sendo $\delta(i)$ a função de impulso discreto, sendo $\delta(0) = 1$ e $\delta(i \neq 0) = 0$. A rede de Laguerre discreta é composta por uma série de sistemas Γ da forma

$$\Gamma_N(z) = \frac{\sqrt{1-a^2}}{1-az^{-1}} \frac{z^{-1}-a}{1-az^{-1}}^{N-1}, \quad (4.31)$$

em que a é a localização do polo da rede, variando de 0 a 1.

Tomando

$$L(k) = \begin{bmatrix} l_1(k) & l_2(k) & l_3(k) & \dots & l_N(k) \end{bmatrix}^T \quad (4.32)$$

pode-se escrever

$$L(k+1) = A_\ell L(k), \quad (4.33)$$

se for definido $\beta = 1 - a^2$,

$$L(0)^T = \sqrt{\beta} \begin{bmatrix} 1 & -a & a^2 & -a^3 & \dots & (-1)^{N-1} a^{N-1} \end{bmatrix} \quad (4.34)$$

e

$$A_\ell = \begin{bmatrix} a & 0 & 0 & 0 & \dots & 0 \\ \beta & a & 0 & 0 & \dots & 0 \\ -a\beta & \beta & a & 0 & \dots & 0 \\ a^2\beta & -a\beta & \beta & a & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \dots & \vdots \end{bmatrix}. \quad (4.35)$$

No caso especial $a = 0$, a matriz A_ℓ se torna

$$A_\ell = \begin{bmatrix} 0 & 0 & 0 \dots & 0 & 0 \\ 1 & 0 & 0 \dots & 0 & 0 \\ 0 & 1 & 0 \dots & 0 & 0 \\ 0 & 0 & 1 \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \dots & 0 & 1 & 0 \end{bmatrix}, \quad (4.36)$$

e o vetor de condição inicial se torna

$$L(0)^T = \begin{bmatrix} 1 & 0 & 0 & 0 & \dots & 0 \end{bmatrix}, \quad (4.37)$$

que é exatamente o caso utilizado até a seção anterior, ou seja, o caso estudado até então é um caso especial da formulação por Laguerre.

Um sistema pode ser descrito utilizando uma rede de funções de laguerre ao encontrar-se os coeficientes c_i tal que, para N termos,

$$H(k) = c_1(k)l_1(k) + c_2(k)l_2(k) + \dots + c_N(k)l_N(k). \quad (4.38)$$

Assim, o problema de descrever o sistema se torna um problema de determinação de coeficientes. O seguinte código Python calcula os coeficientes e o conjunto de funções L para um determinado sistema. Note que a entrada *sys* é um *tuple* do tipo (A, B, C, D, dt).

```

1 def lagd(a, N):
2     ''' Retorna as matrizes A1 e L0 usadas no cálculo dos coeficientes '''
3     toeplitz = sp.linalg.toeplitz
4     beta = 1 - a * a
5     A1 = np.tril(toeplitz([a] + [(-1)**i * a**i * beta for i in range(N - 1)]))
6     L0 = np.sqrt(beta) * np.array([(-1)**i * a**i for i in range(N)])
7     return A1, L0.reshape(N, 1)
8
9 def lagfit(sys, a, N_sim):
10    '''
11    Retorna os coeficientes e matriz L para esta simulação
12
13    sys = sistema (ver help(sp.signal.dimpulse))
14    a = peso ajustável

```

```

15     N = número de variáveis para descrever o sistema
16     N_sim = número de pontos na simulação
17     ...
18     ts, (ys, ) = sp.signal.dimpulse(sys, n=N_sim)
19     A1, L0 = lagd(a, N)
20     L = np.zeros((N_sim, N, 1))
21     L[0, :] = L0
22     for k in range(1, N_sim):
23         L[k, :] = A1 @ L[k - 1, :]
24         cs = [L[:, i].reshape(1, N_sim) @ ys for i in range(N)]
25     return cs, L

```

A trajetória de controle definida como uma sequência de variações do sinal de controle (Δu) é vista como uma sequência de impulsos aplicados ao sistema. Dessa forma, um conjunto de funções de Laguerre pode ser utilizado para descrever essa dinâmica, resultando em

$$\Delta u(k_i + k) = \sum_{j=1}^N c_j(k_i) l_j(k). \quad (4.39)$$

Retomando a Equação (4.32), pode-se reescrever essa equação como

$$\Delta u(K_i + k) = L(k)^T \eta, \quad (4.40)$$

sendo

$$\eta = \begin{bmatrix} c_1 & c_2 & \dots & c_N \end{bmatrix}^T. \quad (4.41)$$

Dessa forma, $\Delta u(k_i + k)$ passa a depender dos parâmetros η , que são os mesmos para todo Δu , e $L(k)$, que é calculado offline. Assim, pode-se substituir a otimização de ΔU pela minimização de η , que tem tamanho N . Note que a relação entre o tamanho de η e ΔU é a dimensão de $L(k)$ e a quantidade de funções utilizadas, o que permite descrever mais variações de sinal de controle com menos variáveis.

A função de custo deve ser reescrita. Como a prova é extensa, apenas os resultados serão apresentados. Caso o leitor deseje verificar o raciocínio, a prova completa é apresentada no capítulo 3 do livro da Wang (2009).

A função de custo continua sendo a mesma utilizada anteriormente para resolução através de programação quadrática, porém suas constantes e variável objetivo são alteradas para refletir o modelo de Laguerre, resultando em

$$J = \eta^T \Omega \eta + 2\eta^T \Psi x(k_i), \quad (4.42)$$

em que

$$\Omega = \sum_{m=1}^{N_p} \Phi(m) Q \Phi(m)^T + R_L, \quad (4.43)$$

$$\Psi = \sum_{m=1}^{N_p} \Phi(m) Q A^m, \quad (4.44)$$

$$\Phi^T = \sum_{j=0}^{m-1} A^{m-j-1} B L_k(j)^T. \quad (4.45)$$

Para fins de simulação,

$$\eta = \frac{L(0)}{\|L(0)\|^2} u, \quad (4.46)$$

$$x(k+1) = Ax(k) + BL(0)^T \eta. \quad (4.47)$$

O seguinte código calcula as constantes Ω , Ψ e Φ :

```

1 def laguerre_phi(m, A, B, L):
2     mp = np.linalg.matrix_power
3     return np.sum((mp(A, m - i - 1) @ B @ L[i].T for i in range(m))).T
4
5
6 def omega_psi(Q, RL, A, B, L, Np):
7     mp = np.linalg.matrix_power
8     phis = [0] + [laguerre_phi(m, A, B, L) for m in range(1, Np + 1)]
9     omega = np.sum((phis[m] @ Q @ phis[m].T for m in range(1, Np + 1))) + RL
10    psi = np.sum((phis[m] @ Q @ mp(A, m) for m in range(1, Np + 1)))
11    return omega, psi

```

As restrições também devem ser reescritas, mapeando o máximos e mínimos desejados em ΔU , U e Y para η . Assim, a restrição em ΔU pode ser reescrita como

$$M\eta \leq \Delta U^{\max} \quad (4.48)$$

$$-M\eta \leq \Delta U^{\min}, \quad (4.49)$$

em que

$$M = \begin{bmatrix} L_1^T & 0 & \dots & 0 \\ 0 & L_2^T & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & L_m^T \end{bmatrix}. \quad (4.50)$$

A restrição da amplitude do sinal de controle é dada por

$$M\eta \leq U^{\max} - u(k-1) \quad (4.51)$$

$$-M\eta \leq -U^{\max} + u(k-1), \quad (4.52)$$

em que

$$M = \begin{bmatrix} \sum_{i=0}^{k-1} L_1^T & 0 & \dots & 0 \\ 0 & \sum_{i=0}^{k-1} L_2^T & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \sum_{i=0}^{k-1} L_m^T \end{bmatrix}. \quad (4.53)$$

A restrição da amplitude da saída se torna

$$M\eta \leq Y^{\max} - CAx(k-1) \quad (4.54)$$

$$-M\eta \leq -Y^{\max} + CAx(k-1), \quad (4.55)$$

em que

$$M = \begin{bmatrix} CBL_1^T & 0 & \dots & 0 \\ 0 & CBL_2^T & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & CBL_m^T \end{bmatrix}. \quad (4.56)$$

O código Python a seguir calcula essas restrições para os primeiros termos, no formato utilizado pela função de minimização quadrática.

```

1 def restrictions_laguerre(G, umin, umax, dumin, dumax, ymin, ymax, x, u, L, N):
2     # ΔU
3     M1 = np.vstack((-L[0:N].T[0], L[0:N].T[0]))
4     N1 = np.vstack((-dumin * np.ones((N, 1)), dumax * np.ones((N, 1))))
5     # U
6     a = [np.sum((L[i].T for i in range(n)), axis=0) for n in range(1, N + 1)]
7     M2 = np.vstack((-np.vstack(a), np.vstack(a)))
8     N2 = np.vstack((-umin * np.ones((N, 1)) + u, umax * np.ones((N, 1)) - u))
9     # Y
10    A, B, C, D = G
11    M3 = np.vstack((-C @ B @ L[0].T, C @ B @ L[0].T))
12    N3 = np.vstack((-ymin + C @ A @ x, ymax - C @ A @ x))
13    #
14    M = np.vstack((M1, M2, M3))

```

```
15     N = np.vstack((N1, N2, N3))
16     return M, N
```

4.3 Sistemas a parâmetros distribuídos

Neste trabalho é estudada apenas a abordagem proposta por Barroso (2017). Ela consiste basicamente em unir duas propostas já existentes, a SPD por subsistemas interconectados e a baseada em modelos lineares a parâmetros variantes no espaço (Spatial LPV, do inglês *Spatial Linear Parameter Varying*).

SPD por subsistemas interconectados consiste em modelar um problema que varia no espaço, ou seja, um SPD, utilizando vários sistemas a parâmetros concentrados. Assim cada SPC representa a saída do sistema em um ponto no espaço. Esta abordagem, no entanto, é limitada, já que os pontos no espaço onde pode-se fazer a estimativa dos estados é fixada pelo modelo. Para fazer a estimação em outros pontos é necessário remodelar o sistema.

Na Figura 4.2 pode-se ver um esquemático do funcionamento desta aproximação. O sistema varia no espaço de 0 a L . Para obter estados intermediários, desenvolve-se vários modelos que descrevem a dinâmica de um ponto intermediário s_i até um ponto s_j , utilizando modelos SPC para tal.

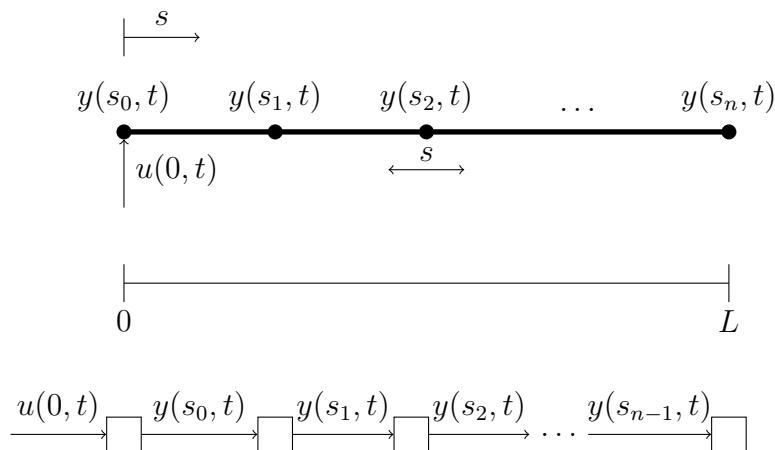


Figura 4.2 – Subsistemas interconectados

Já o método de modelos lineares a parâmetros variantes no espaço propõe a criação de um modelo que irá descrever a dinâmica do sistema do ponto inicial até determinado ponto espacial que é definido como uma variável, chamada de *scheduling*. Assim, alterando esta variável obtém-se como saída o estado em um ponto diferente no espaço.

A Figura 4.3 contém o esquemático de um modelo LPV. Nele o mesmo sistema é descrito por apenas um modelo que tem um parâmetro que altera o ponto no espaço ao

qual a saída se refere. Assim pode-se obter o estado em vários pontos no espaço sem a necessidade de remodelar o sistema.

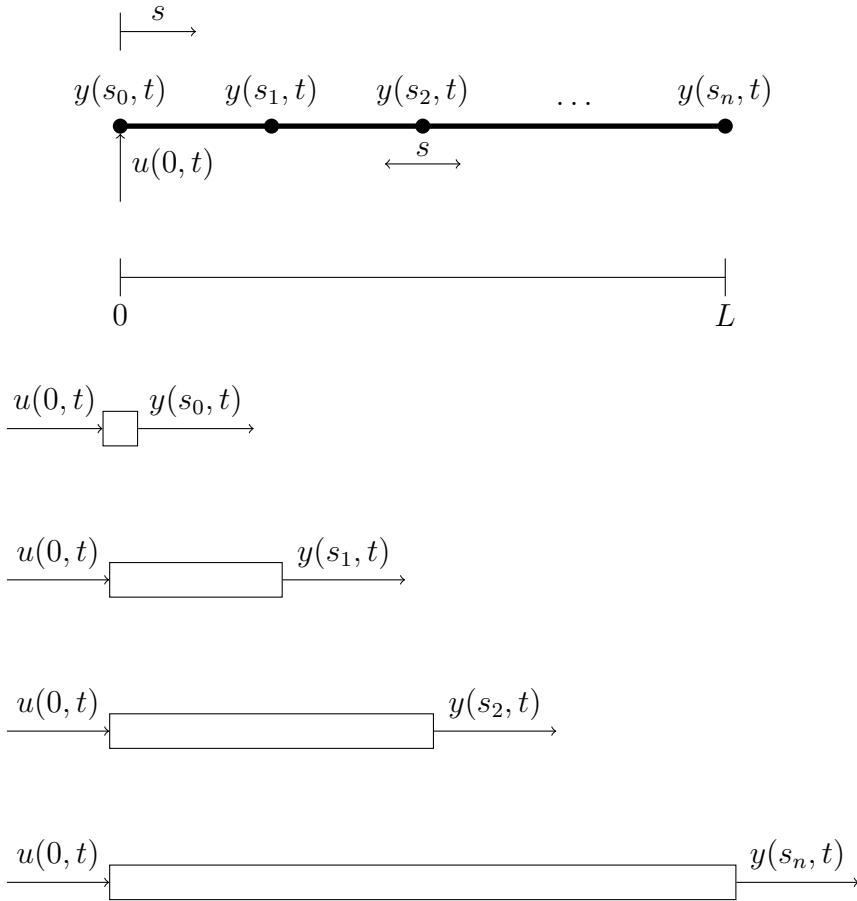


Figura 4.3 – Subsistema LPV

Ao combinar as duas técnicas, Barroso (2017) propõe que se use dois modelos LPV complementares como subsistemas interconectados. O primeiro modelo descreve a dinâmica do sistema do atuador até um ponto x enquanto o segundo modelo descreve a dinâmica desse ponto até uma saída fixa.

Um esquemático desta técnica é apresentado na Figura 4.4. Nele pode-se ver que foi utilizado a técnica de subsistemas interconectados, no entanto, os subsistemas utilizados são LPV e complementares. Isso significa que ao alterar o valor da variável s_l altera-se ambos sistemas para que o primeiro forneça o estado em s_l e o segundo a saída em L , dada a saída s_l . Desta forma o ponto s_l é móvel, podendo ser alterado a qualquer momento.

A principal vantagem desta abordagem é poder utilizar um sensor na saída do segundo modelo LPV e utilizar um observador para estimar o estado de qualquer ponto entre o atuador e o sensor, selecionando o ponto através apenas da alteração da variável *scheduling*, s_l .

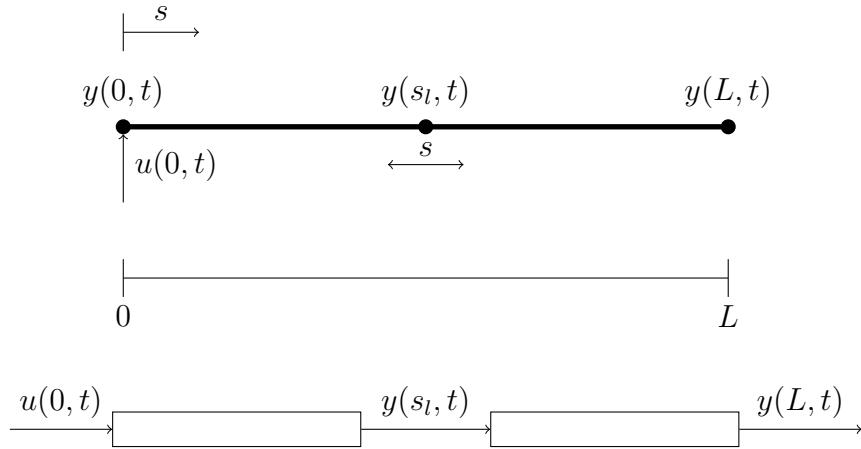


Figura 4.4 – Subsistemas LPV interconectados

4.3.1 Modelagem do forno por LPVs interconectados

Para modelar o sistema foram utilizados dois modelos ARMAX (Modelo autoregressivo de média variável com entradas exógenas, do inglês *Autoregressive-moving-average with exogenous inputs*), um de 0 até s_ℓ e outro de s_ℓ até L . A formulação geral para estes modelos é

$$y(s_\ell, k) = - \sum_{i_y=1}^{n_y} a_{i_y}(s_\ell) y(0, k - i_y) + \sum_{i_u=n_k}^{n_k+n_u-1} b_{i_u}(s_\ell) y(s_\ell, k - i_u) + \sum_{i_v=1}^{n_v} c_{i_v}(s_\ell) v(s_\ell, k - i_v) + v(s_\ell, k) \quad (4.57)$$

e

$$y(L, k) = - \sum_{i_y=1}^{n_y} \alpha_{i_y}(\bar{s}_\ell) y(L, k - i_y) + \sum_{i_u=n_k}^{n_k+n_u-1} \beta_{i_u}(\bar{s}_\ell) y(\bar{s}_\ell, k - i_u) + \sum_{i_v=1}^{n_v} \gamma_{i_v}(\bar{s}_\ell) v(\bar{s}_\ell, k - i_v) + v(\bar{s}_\ell, k). \quad (4.58)$$

Após um dimensionamento inicial obtém-se

$$y(s_\ell, k) = -a_1(s_\ell) y(s_\ell, k - 1) + b_1(s_\ell) y(0, k - 1) + c_1(s_\ell) v(s_\ell, k - 1) + c_2(s_\ell) v(s_\ell, k - 2) + v(s_\ell, k) \quad (4.59)$$

e

$$y(L, k) = -\alpha_1(\bar{s}_\ell) y(L, k - 1) + \beta_0(\bar{s}_\ell) y(\bar{s}_\ell, k) + \beta_1(\bar{s}_\ell) y(\bar{s}_\ell, k - 1) + \gamma_1(\bar{s}_\ell) v(\bar{s}_\ell, k - 1) + \gamma_2(\bar{s}_\ell) v(\bar{s}_\ell, k - 2) + v(\bar{s}_\ell, k), \quad (4.60)$$

sendo os coeficientes na forma

$$a(s_\ell) = \sum_{i_a=0}^{n_a} a_{i_a} s_\ell^{i_a}. \quad (4.61)$$

Obtém-se assim as funções scheduling de 3 (4.62), 5 (4.63) e 9 (4.64) posições, obtidas utilizando a técnica de mínimos quadrados para ajustar os coeficientes (BARROSO, 2017).

$$\begin{aligned}
 b1 &= 1.3291 \times 10^{-9} s_\ell^2 - 2.2196 \times 10^{-5} s_\ell + 0.0563 \\
 a1 &= -1.5101 \times 10^{-8} s_\ell^2 + 1.3591 \times 10^{-5} s_\ell - 0.9118 \\
 w1 &= 1.6636 \times 10^{-8} s_\ell^2 - 3.2243 \times 10^{-5} s_\ell + 0.0213 \\
 \beta_0 &= -2.5839 \times 10^{-4} \bar{s}_\ell + 0.9036 \\
 \beta_1 &= 2.3909 \times 10^{-4} \bar{s}_\ell - 0.8628 \\
 \alpha_1 &= -1.6436 \times 10^{-5} \bar{s}_\ell - 0.9559 \\
 w2 &= 4.9739 \times 10^{-6} \bar{s}_\ell + 0.0029
 \end{aligned} \tag{4.62}$$

$$\begin{aligned}
 b1 &= -3.3299 \times 10^{-9} s_\ell^2 - 1.5982 \times 10^{-5} s_\ell + 0.05711 \\
 a1 &= -3.0164 \times 10^{-8} s_\ell^2 + 3.3929 \times 10^{-5} s_\ell - 0.91092 \\
 w1 &= 1.4014 \times 10^{-8} s_\ell^2 - 2.6372 \times 10^{-5} s_\ell + 0.01952 \\
 \beta_0 &= -4.8261 \times 10^{-8} \bar{s}_\ell^2 - 1.3554 \times 10^{-4} \bar{s}_\ell + 0.8171 \\
 \beta_1 &= 5.7969 \times 10^{-8} \bar{s}_\ell^2 + 8.1357 \times 10^{-5} \bar{s}_\ell - 0.7485 \\
 \alpha_1 &= 2.9771 \times 10^{-8} \bar{s}_\ell^2 - 8.6561 \times 10^{-5} \bar{s}_\ell - 0.9149 \\
 w2 &= 3.8259 \times 10^{-9} \bar{s}_\ell^2 - 3.1247 \times 10^{-6} \bar{s}_\ell + 0.0068
 \end{aligned} \tag{4.63}$$

$$\begin{aligned}
 b1 &= -4.2490 \times 10^{-12} s_\ell^3 + 3.4291 \times 10^{-9} s_\ell^2 - 1.5999 \times 10^{-5} s_\ell + 0.0568 \\
 a1 &= 9.5100 \times 10^{-12} s_\ell^3 - 4.7554 \times 10^{-8} s_\ell^2 + 4.3890 \times 10^{-5} s_\ell - 0.9112 \\
 w1 &= -2.3370 \times 10^{-11} s_\ell^3 + 6.6704 \times 10^{-8} s_\ell^2 - 5.7759 \times 10^{-5} s_\ell + 0.0237 \\
 \beta_0 &= 2.0229 \times 10^{-10} \bar{s}_\ell^3 - 4.7445 \times 10^{-7} \bar{s}_\ell^2 + 1.1916 \times 10^{-4} \bar{s}_\ell + 0.7532 \\
 \beta_1 &= -2.5343 \times 10^{-10} \bar{s}_\ell^3 + 6.2283 \times 10^{-7} \bar{s}_\ell^2 - 2.8674 \times 10^{-4} \bar{s}_\ell - 0.6551 \\
 \alpha_1 &= -5.5237 \times 10^{-11} \bar{s}_\ell^3 + 1.4785 \times 10^{-7} \bar{s}_\ell^2 - 1.5578 \times 10^{-4} \bar{s}_\ell - 0.8992 \\
 w2 &= 8.1120 \times 10^{-11} \bar{s}_\ell^3 - 1.8802 \times 10^{-8} \bar{s}_\ell^2 + 1.6462 \times 10^{-4} \bar{s}_\ell + 0.0017
 \end{aligned} \tag{4.64}$$

As equações (4.60) e (4.59) podem ser reescritas no formato de espaço de estados como

$$\begin{aligned} x(k) &= \begin{bmatrix} x_1(L, k) \\ x_2(s_\ell, k) \end{bmatrix} \\ x(k+1) &= \begin{bmatrix} -\alpha(\bar{s}_\ell) & \beta_1(\bar{s}_\ell) - \beta_0(\bar{s}_\ell)a_1(s_\ell) \\ 0 & -\alpha(s_\ell) \end{bmatrix} x(k) + \begin{bmatrix} \beta_0(\bar{s}_\ell)b_1(s_\ell) \\ b_1(s_\ell) \end{bmatrix} u(0, k) \\ y(L, k) &= \begin{bmatrix} 1 & 0 \end{bmatrix} x(k). \end{aligned} \quad (4.65)$$

Os modelos levantados por Barroso (2017) estão discretizados com tempo de amostragem de 5 segundos.

4.4 Observadores

Observadores são modelos que reconstroem informação em tempo real. Sua função é recuperar as informações dos estados a partir do modelo do sistema, a entrada, a saída e o último estado estimado. Eles funcionam como malha fechada, utilizando o modelo para predizer o próximo estado e um ganho para corrigir o erro da saída estimada em relação à saída real.

Existem vários tipos de observadores. Neste trabalho serão utilizados dois: o filtro de Kalman e o observador exponencial com fator de esquecimento. O filtro de Kalman é talvez o mais utilizado e funciona tanto como observador quanto filtro de ruídos. Porém sua formulação requer a covariância das variações dos estados. Já o observador exponencial requer apenas a covariância da saída e possui uma variável para ajustar a velocidade de convergência facilmente.

4.4.1 Observabilidade

Observabilidade é a medida de quão bem os estados internos de um sistema podem ser estimados a partir de suas saídas. Formalmente um sistema é dito observável se para qualquer sequência de estados e sinais de controle, o estado atual pode ser determinado em tempo finito usando apenas as saídas. Se o sistema não é observável, então existe pelo menos um estado que não pode ser determinado apenas pela saída.

Para sistemas invariantes no tempo em espaço de estados com n estados, se o posto da matriz L_o da equação (4.66) for igual a n , o sistema é dito observável. A ideia por traz deste teste é que, se a matriz contém n linhas linearmente independentes, então os n estados podem ser vistos como combinações lineares da saída.

$$L_o = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{n-1} \end{bmatrix} \quad (4.66)$$

No caso desse trabalho o sistema é observável.

4.4.2 Filtro de Kalman

O filtro de Kalman é um observador estocástico ótimo discreto no tempo desenvolvido por Kalman (1960). Assumindo que um sistema passa a estar sujeito a ruídos gaussianos de média nula e não correlacionados v e w , podemos descrever seu comportamento como

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) + v(k) \\ y(k) &= Cx(k) + Du(k) + w(k). \end{aligned} \quad (4.67)$$

Para sistemas descritos desta forma, o Teorema 1 define o observador de Kalman.

Teorema 1 *Para qualquer sequência de entrada regularmente persistente para o par (A, C) o seguinte sistema:*

$$\begin{aligned} P(\infty) &= A(P(\infty) - P(\infty)C^T(\Gamma + CP(\infty)C^T)^{-1}CP(\infty))A^T + \Theta \\ K_{ob}(\infty) &= AP(\infty)C^T(\Gamma + CP(\infty)C^T)^{-1} \\ \hat{x}(k+1) &= A\hat{x}(k) + Bu(k) + K_{ob}(\infty)(y(k) - C\hat{x}(k)) - Du(k) \end{aligned} \quad (4.68)$$

provê um observador ótimo para o sistema (4.67).

Na equação (4.68), Γ e Θ são as matrizes de covariância da saída e da variação dos estados, respectivamente. O primeiro pode ser facilmente medido. Basta realizar uma leitura fixa com o sensor por várias amostras e calcular a covariância do vetor de valores resultantes. O segundo exige que seja possível medir todos os estados, o que nem sempre

acontece. Sendo possível, mede-se todos os estados em um valor fixo por várias amostras e calcula-se a covariança da diferença dos estados ($x(k) - x(k-1)$ para cada estado). Caso não seja possível deve-se procurar formas de estimar a covariança ou encontrar valores manualmente que produzam resultados satisfatórios.

$P(\infty)$ e $K_{ob}(\infty)$ são os valores P e K_{ob} que estabilizam as equações, ou seja, os valores em que a saída não mais se altera. Assim o conjunto de equações (4.68) é convergente para os valores de P e K_{ob} , e esses podem ser obtidos através do cálculo iterativo de $P(\infty)$ até sua convergência.

4.4.3 Observador exponencial com fator de esquecimento

O observador exponencial com fator de esquecimento (TICLEA; BESANÇON, 2009) também é um observador do tipo Kalman. Isso pode ser visto em sua estrutura. Sua formulação é dada pelo Teorema 2.

Teorema 2 *Dado os sistema (4.67), e assumindo que uma sequência de entrada, $u(k)$, é regularmente persistente para o par (A, C) e torna A invertível, então, um observador globalmente exponencialmente convergente é dado por:*

$$\begin{aligned} P(\infty) &= \delta^{-1} A(P(\infty) - KCP(\infty))A^T \\ K_{ob}(\infty) &= AP(\infty)C^T(\Gamma + CP(\infty)C^T)^{-1} \\ \hat{x}(k+1) &= A\hat{x}(k) + Bu(k) + K_{ob}(\infty)(y(k) - C\hat{x}(k)) - Du(k). \end{aligned} \quad (4.69)$$

Na equação (4.69), δ é um número real que controla a velocidade de convergência do observador. Quanto menor δ , mais rápido o observador converge.

4.5 PI por síntese direta

O desenvolvimento de controladores PI por síntese direta é uma forma de obter controladores a partir da resposta desejada do sistema em malha fechada. Isso é feito igualando a equação da malha fechada com o controlador à uma resposta de grau compátilvel desejada. Por exemplo, o controlador PI pode ser escrito como

$$C(s) = K_c(1 + \frac{1}{T_i s}), \quad (4.70)$$

em que K_c e T_i são parâmetros do controlador. Para um sistema de primeira ordem a equação de malha fechada com esse controlador é dada por

$$\frac{Y(s)}{R(s)} = \frac{\frac{T_i}{K_c}s}{(\frac{\tau T_i}{KK_c} - T_i)s^2 + (\frac{T_i}{KK_c} + T_i)s + 1}. \quad (4.71)$$

Se a dinâmica de malha fechada desejada for definida como

$$\left(\frac{Y(s)}{R(s)} \right)_d = \frac{K_d s}{(T_d s + 1)^2}, \quad (4.72)$$

pode-se igualar as respostas de forma a encontrar os valores de K_c e T_i que tornam a equação verdadeira.

Há várias receitas de controladores PID e suas variações seguindo esse princípio. Elas se diferenciam pela forma como o controlador ou a resposta desejada é escrita e pela forma como os parâmetros são isolados.

4.6 Índices de desempenho

A comparação entre controladores não é uma tarefa fácil, dado que não existe uma forma de definir os conceitos melhor e pior para os mesmos. Utiliza-se portanto índices que quantificam a resposta com base em algum fator, como o tempo de transitório, tempo de subida, erro em regime permanente, e variação do sinal de controle, entre outros.

Para comparar os controladores propostos utilizou-se dois índices, o $IA\Delta X$ e o IAE. O primeiro mede a variação do sinal, e foi utilizado para medir a variação do sinal de controle e da saída. Seu cálculo se dá conforme a Equação (4.73).

$$IA\Delta X(x \in R^N) = \sum_{i=2}^N |x_i - x_{i-1}| \quad (4.73)$$

O índice IAE quantifica a resposta do sistema de acordo com a integral da norma do erro. Ele pode ser utilizado para comparar tanto o transitório quanto o regime permanente do sistema. No entanto, o sinal utilizado no cálculo deve conter apenas o transitório ou apenas o regime permanente do sinal, e jamais uma mistura dos dois. Sua fórmula pode ser vista na Equação (4.74), onde r é a referência utilizada na obtenção da curva de saída y .

$$IAE(r, y) = \int |r - y| dt \quad (4.74)$$

Capítulo 5

Resultados e Discussões

Neste capítulo são apresentados os resultados obtidos.

5.1 Estudos teóricos

5.1.1 Controle preditivo por modelo

O livro-texto utilizado para os estudos do controle preditivo por modelo foi o do Wang (2009), com as outras referências sendo usadas de forma complementar. Para a execução deste projeto propõe-se o estudo dos capítulos 1 a 4, que tratam do caso discreto.

Foram estudados os capítulos 1 e 2, que apresentam o conceito do controlador e a formulação com restrições, respectivamente. Como os estudos teóricos de MPC foram iniciados antes da alteração da eletrônica do forno, a técnica foi testada utilizando os tanques comunicantes (Figura 5.1) presentes no Laboratório de Sinais e Sistemas.

Os tanques utilizados são o 1º e 2º, que estão localizados na parte superior e marcados como T1 e T2. A bomba insere água no primeiro tanque e esta escoa para o segundo através de uma válvula localizada na parte inferior. A água é então removida através de uma válvula no fundo do tanque 2. O objetivo de controle é o nível do tanque 2.

Uma simulação do controlador MPC controlando o modelo desse sistema pode ser visto na Figura 5.2. Observa-se que o sinal de controle é mantido em 100% por quase todo o transitório, e é então abaixado para o valor de equilíbrio, fazendo com que o tanque encha de forma rápida mas minimizando o overshoot.

Ao inserir este controlador no sistema real (Figura 5.3) obtem-se uma resposta parecida com a da simulação, porém mais lenta. Essa diferença pode ser explicada por erro de



Figura 5.1 – Sistema de tanques comunicantes

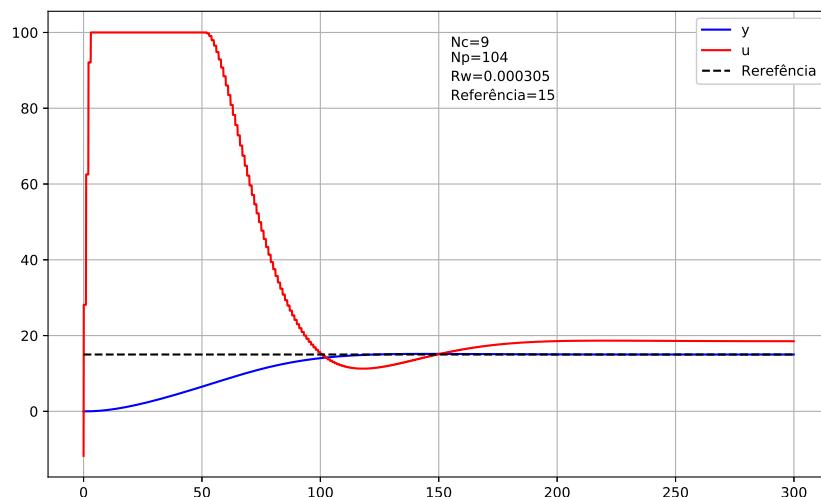


Figura 5.2 – Simulação do modelo dos tanques com controlador MPC

modelagem e pelo fato que o sistema real não é linear, sendo o modelo linearizado. No entanto a resposta continua satisfatória.

Como foi utilizado um observador, também pode-se perceber os efeitos do erro de modelagem. Os estado estimado, x_1 , embora apresente a mesma dinâmica, não apresenta o mesmo ganho que o estado real, h_1 . É interessante notar, no entanto, que esta diferença não atrapalha o controlador, que continua capaz de seguir a referência.

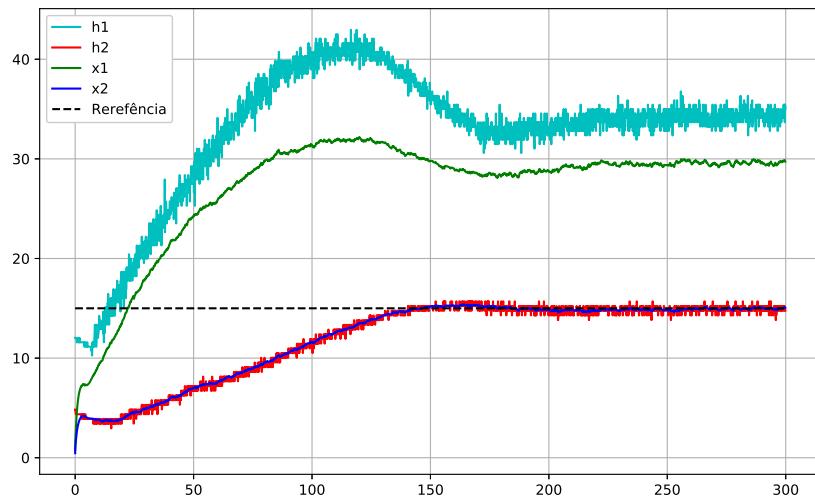


Figura 5.3 – MPC aplicado ao sistema real

5.1.2 Modelagem térmica e SPD

Os resultados obtidos por Barroso (2017) foram simulados e utilizou-se o modelo SPD apresentado com os observadores de Kalman e exponencial com fator de esquecimento. Foi possível ver nos estados a saída variando conforme a distância do atuador aumenta, como pode ser visto na Figura 5.4. Os observadores também estimam com precisão os estados do sistema simulado, conforme Figura 5.5.

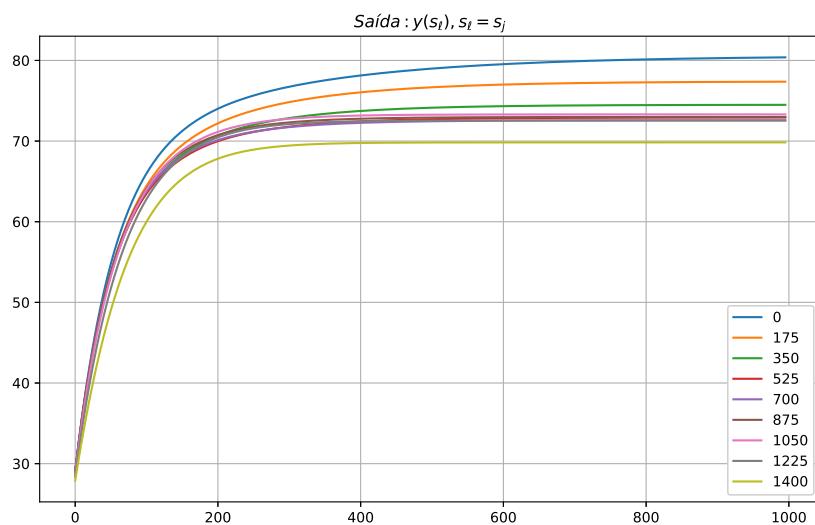


Figura 5.4 – SPD simulado

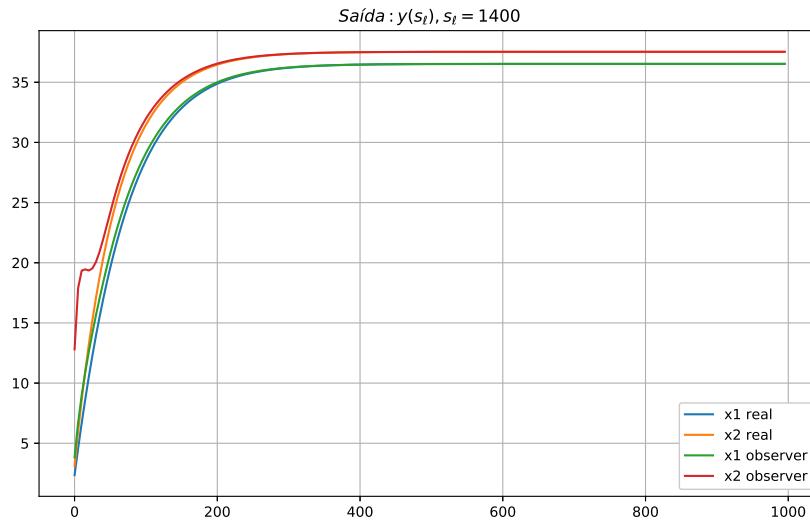


Figura 5.5 – SPD simulado com observador exponencial

5.2 Modificação do hardware e implantação da plataforma

Ao estudar o circuito de acionamento e sensoriamento percebeu-se que não há a necessidade de alterar o circuito. Devido a forma como o mesmo está montado, foi necessário apenas a reprogramação dos microcontroladores presentes para possibilitar a comunicação direta através de cabos USB. Após a reprogramação foi possível acionar o circuito de potência e ler os valores dos sensores.

A instalação do Raspberry não foi feita pois o mesmo não foi adquirido a tempo. No entanto, as modificações necessárias para a sua inserção foram executadas. Tais modificações compreendem principalmente mudanças no software *moirai* e já foram publicadas.

5.3 Calibração

O funcionamento do circuito eletrônico foi feito primeiramente acionando o sistema desenvolvido por Barroso (2017). Após verificar que este era capaz de acionar as resistências e ler valores de todos os sensores, foi feita a modificação dos códigos dos microcontroladores para permitir a comunicação com a plataforma e testou-se novamente com o novo código. Verificou-se a necessidade de recalibrar os sensores e reordená-los, identificando qual estava conectado a qual porta. Devido a forma como se dá o acionamento não é

necessária a calibração dos atuadores.

Não é possível fazer a calibração utilizando o próprio ar como fluido aquecido. Isso pois, independente do recipiente, serão formadas correntes de convecção que farão com que os sensores, por mais próximos que fiquem uns dos outros, façam leituras diferentes. Esse efeito sempre se mostra durante a validação da calibração.

Utilizou-se então um *Becker* de 1 L e água destilada. Os sensores foram submersos na água junto com um ebulidor. Caso o ebulidor seja de 220 V, pode-se ligá-lo nas fases da resistência da própria planta, desconectando a R_3 e conectando o ebulidor. Caso ele seja 110 V, pode-se ligar uma fase na planta e pegar o neutro na rede elétrica. Recomenda-se atenção às normas de segurança.

Um PI foi sintonizado empíricamente apenas para estabilizar e rejeitar distúrbios, visando manter a temperatura da água constante independente da temperatura ambiente. Foram feitas medições usando como referência as temperaturas de 80 °C à 110 °C, variando de 5 em 5, tomando como saída o sensor 2. Junto aos sensores havia um termômetro, que foi usado como temperatura referência.

Assim, a temperatura dos termômetros ficam fixas e o termômetro externo exibe a temperatura real da água. Com todos os valores, pode-se utilizar mínimos quadrados de forma a encontrar uma expressão que associa a temperatura medida pelo termômetro com a temperatura real do meio. Os valores usados para a calibração podem ser vistos na Tabela 5.1, onde T_1 e T_2 são os dois termopares externos, cuja média foi utilizada como referência.

Com esses dados, encontrou-se as equações de calibração (5.1) à (5.9), onde x é a leitura do sensor correspondente.

Tabela 5.1 – Valores utilizados na calibração

	110	105	100	95	90	85	80
S_2	109	105	100	94	89	84	80
S_3	109	105	100	95	90	85	80
S_4	109	105	100	95	90	85	80
S_5	109	105	100	95	90	85	80
S_6	109	104	99	94	89	84	79
S_7	110	105	100	95	90	85	80
S_8	109	104	99	94	89	84	79
S_9	110	105	100	95	90	85	80
S_{10}	109	104	99	94	90	84	80
T_1	90.5	88.0	84.3	80.2	76.3	72.1	68.0
T_2	90.6	87.9	84.5	80.6	76.3	72.2	68.1

$$S_2 = 0.7634x + 7.4859 \quad (5.1)$$

$$S_3 = 0.7792x + 5.7620 \quad (5.2)$$

$$S_4 = 0.7801x + 5.6355 \quad (5.3)$$

$$S_5 = 0.7736x + 6.2662 \quad (5.4)$$

$$S_6 = 0.7697x + 7.3317 \quad (5.5)$$

$$S_7 = 0.7660x + 7.0308 \quad (5.6)$$

$$S_8 = 0.7769x + 6.5083 \quad (5.7)$$

$$S_9 = 0.7664x + 6.9612 \quad (5.8)$$

$$S_{10} = 0.7818x + 5.9085 \quad (5.9)$$

5.4 Síntese e validação do observador

Seguindo a proposta do trabalho, foi necessário desenvolver um observador para o modelo desenvolvido por Barroso (2017). Em sua tese ele apresenta vários modelos de observadores. Dois foram selecionados para serem testados, o observador de Kalman e o observador com fator de esquecimento.

Ambos apresentaram performance parecida, mas o observador de Kalman, por ser também um filtro, retorna valores menos esparços. Porém, ele apresenta dois problemas

em sua estimativa: possui um erro de *offset* e não replica variações do estado medido no estado estimado. A saída com tais problemas evidenciados pode ser vista na Figura 5.6.

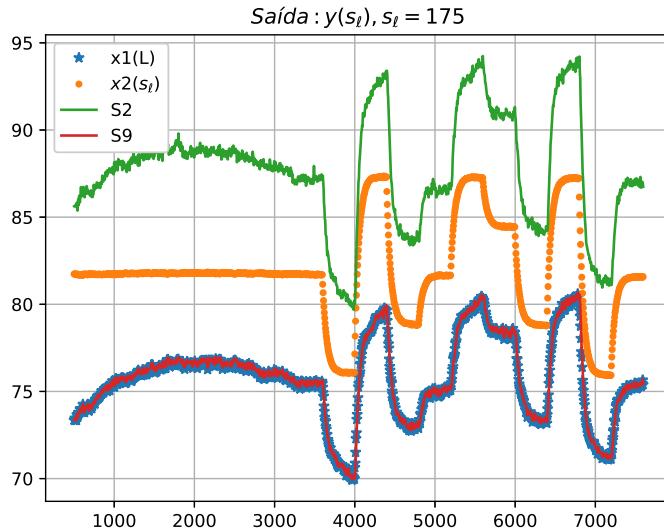


Figura 5.6 – Filtro de Kalman com ganho estático e erro de *offset* e de seguimento

Para resolver o problema do seguimento de variações no sinal medido, utilizou-se do seguinte recurso matemático: zerou-se a matriz Q , de covariâncias nos estados, o que faz com que ambos estados não mais sigam tal variação, conforme pode ser visto na Figura 5.7.

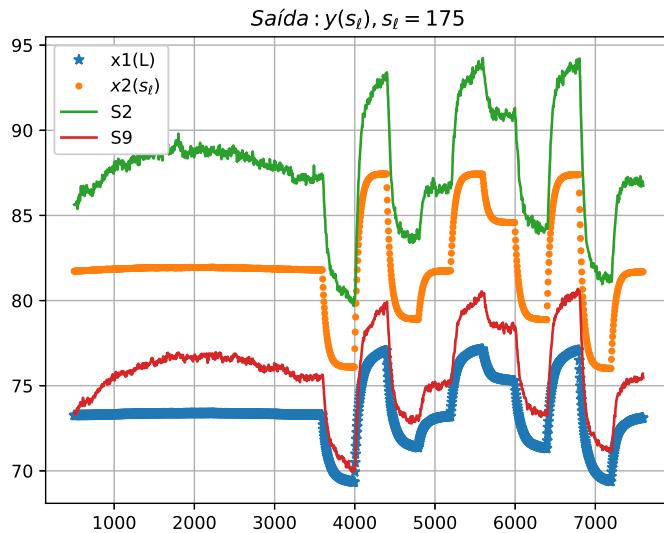


Figura 5.7 – Filtro de Kalman com matriz $Q = 0$

Supondo que a perda de calor durante o trecho é zero, a mesma variação que aparece no sensor S_L deveria aparecer no sensor S_ℓ . Assim, soma-se a ambos estados a diferença

entre a mediação e a estimativa de S_L . O resultado pode ser visto na Figura 5.8.

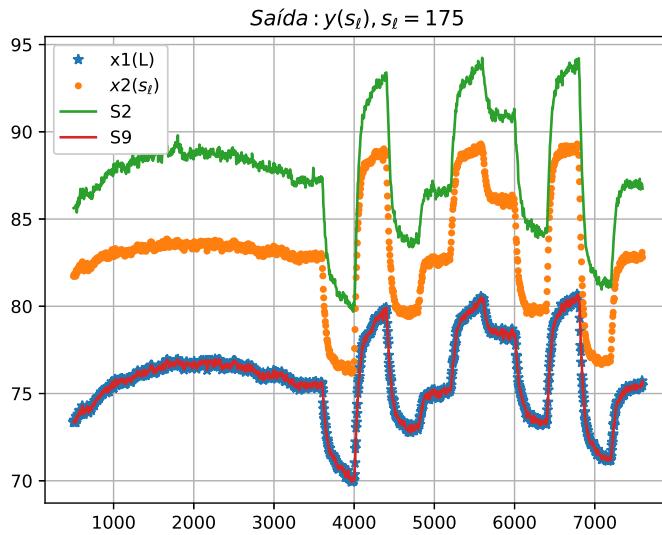


Figura 5.8 – Filtro de Kalman com offset

O efeito citado está atenuado, por causa do mesmo problema que gera o *offset*. Trata-se do fato de que o modelo não leva em conta a temperatura ambiente. Assim, variações na mesma afetam a saída do observador. A melhor solução para o problema seria o levantamento de um novo modelo que modele a temperatura ambiente como um distúrbio no sistema, mas isso seria um trabalho de nível de mestrado. Fez então um procedimento de forma a obter uma função que liga a temperatura ambiente ao sinal de controle de equilíbrio. A resposta utilizando essa função pode ser vista na Figura 5.9.

Para gerar essa função segue-se o seguinte procedimento: coloca-se o sistema para funcionar com um PI seguindo a referência de 80 °C graus e controlando a temperatura ambiente à 25 °C com um ar condicionado. Dessa forma os sensores apresentam uma medição constante e as variações da temperatura ambiente são refletidas no sinal de controle. Como o ar condicionado é capaz de manter a temperatura estável com um erro de aproximadamente 0,5 °C, pode-se considerar que ela está constante.

Coleta-se todas as medições após o sistema estar certamente em equilíbrio. Idealmente, repete-se o procedimento para várias temperaturas ambientes. Devido à dificuldade de controlar a temperatura em outros pontos, coletou-se apenas um outro ponto com o ar desligado, em um período do dia em que a temperatura ambiente se estabilizou em 28 °C.

Esses dois pontos foram utilizados para encontrar uma reta que passa por ambos, e essa função foi tomada como uma relação entre temperatura ambiente e temperatura de

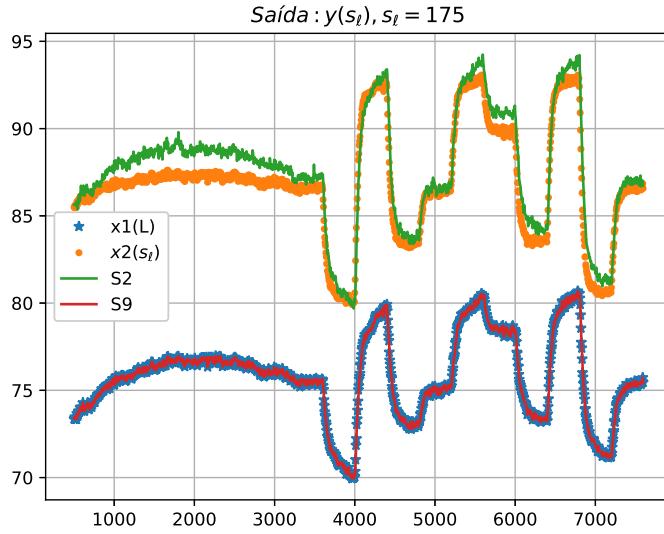


Figura 5.9 – Filtro de Kalman com todas as correções aplicadas

equilíbrio. A equação encontrada foi

$$u = -1.34T_{amb} + 110.97, \quad (5.10)$$

onde T_{amb} é a temperatura ambiente. Utilizando essa equação, o sinal de controle de equilíbrio é recalculado a cada amostragem, o que faz com que o observador calcule a temperatura dos sensores com menos erro.

No entanto, percebeu-se que essa relação não é linear, e que mais pontos, principalmente para as temperaturas mais altas, seriam necessários. Assim, o observador continua inserindo um offset nas temperaturas acima de 80 °C.

A Figura 5.10 mostra o controlador MPC atuando no sistema utilizando o observador com todas as correções. Percebe-se que o erro nas temperaturas acima de 80 °C aumenta conforme a temperatura aumenta, e que o sistema não conseguiu chegar aos 90 °C, apesar da resistência estar em sua potência máxima. Assim a solução utilizada pode funcionar se forem levantadas curvas para várias temperaturas ambiente e de referências desejadas, mas o desenvolvimento de um novo modelo que leve em conta a temperatura ambiente é a melhor solução.

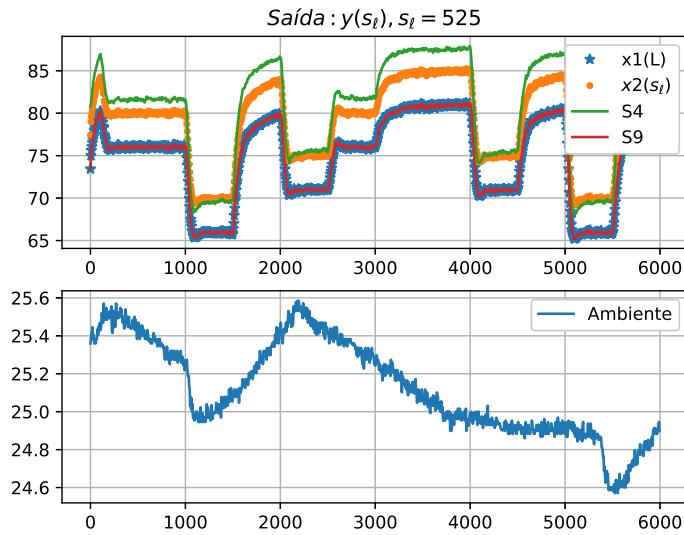


Figura 5.10 – MPC no Filtro de Kalman com todas as correções aplicadas

5.5 Escolha de N_p e N_c

Assim como em muitos métodos de controle moderno e robusto, não há uma relação direta entre os parâmetros N_p e N_c e resposta do sistema no tempo. Dessa forma, não é possível calcular esses parâmetros de forma a se obter uma resposta desejada, e eles devem ser ajustados empíricamente.

Há regras na literatura para obtenção de parâmetros razoáveis. Por exemplo, segundo Wang (2009), N_c deve ser de 4 a 10 vezes menor que N_p e N_p deve ser suficiente para cobrir ao menos 1τ . Um N_p maior que o números de amostras necessário para o regime permanente em malha aberta é desperdício de processamento. E $N_p \geq N_c$, pois as dimensões não seriam compatíveis caso contrário.

Utilizando essas regras pode-se verificar quem seriam os limites inferiores e superiores de N_p e N_c e utiliza-los para gerar todos as combinações possíveis. Em simulação, todas podem ser testadas, terem seus índices de desempenho calculados e comparados, de forma a escolher a combinação com o melhor índice de desempenho. Como os índices de desempenho penalizam características diferentes da resposta, como erro de estado estacionário ou duração de transitório, a escolha do índice pode ajudar a encontrar a melhor combinação para a característica desejada.

Foi escolhido o índice IAE, que penaliza tanto o transitório quanto o regime permanente de forma igual. Como o sistema tem sabidamente erro zero de seguimento de

referência do tipo degrau, esse erro penalizou apenas o transitório. Foram feitas combinações com N_p variando de 10 a 100 e N de 2 a 50. Note que agora fala-se de N e não N_c , pois está sendo utilizado no sistema as funções de Laguerre.

Todas as combinações foram simuladas e tiveram seus índices calculados. Foram salvos o melhor e pior caso. Não foi o caso desse experimento, mas vale ressaltar que algumas combinações podem desestabilizar o sistema, logo convém adicionar ao código de teste uma verificação de estabilidade da resposta. Na Figura 5.11 pode-se ver a resposta de todas as 840 combinações. A melhor resposta foi $N = 8$ e $N_p = 86$, que resultou em $IAE = 2254$ e a pior resposta foi $N = 5$ e $N_p = 100$, que resultou em $IAE = 2665$.

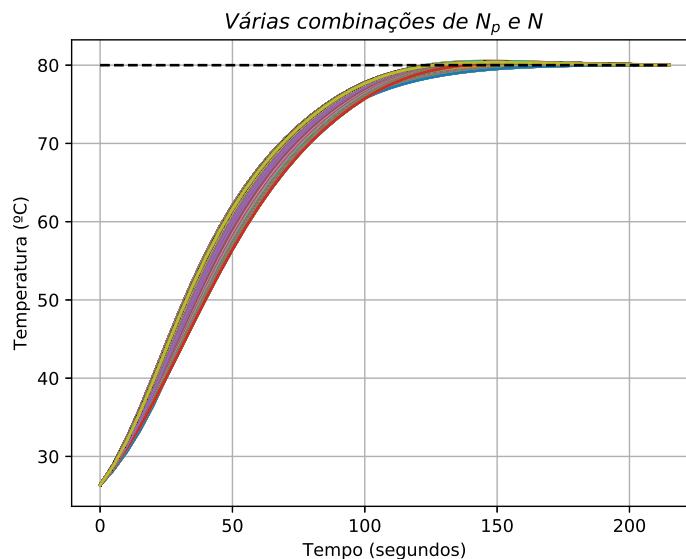


Figura 5.11 – Resposta das 840 combinações de N_p e N

A resposta da melhor combinação pode ser vista na Figura 5.12. A dinâmica dessa resposta será utilizada na sintonia dos controladores PID por síntese direta.

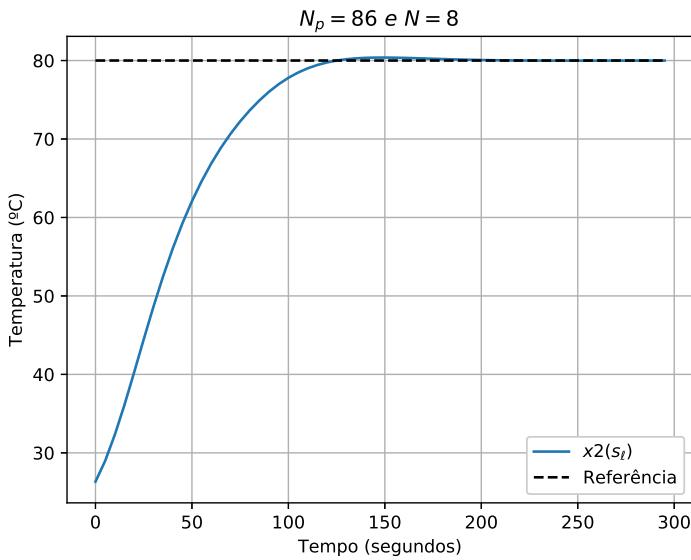


Figura 5.12 – Resposta do MPC com $N_p = 86$ e $N = 8$

5.6 PI por síntese direta

A síntese do controlador PID foi realizada utilizando a técnica descrita em Chen e Seborg (2002). Utilizou-se as fórmulas 16 e 17 do artigo, transcritas aqui nas Equações (5.11) e (5.12), tomando $\tau = 45$, $K = 0.36$ e $\tau_d = 33$. Os valores das constantes de tempo foram calculados como o tempo de acomodação sobre 6, pois ao utilizar $\frac{T_s}{4}$ a resposta obtida não chegava de fato ao tempo de acomodação desejado, sendo esse quase 100 segundos maior.

$$K_p = \frac{1}{K} \frac{\tau}{\tau_1 s} \quad (5.11)$$

$$T_i = \tau \quad (5.12)$$

O controlador encontrado foi um PI com $K_p = 3.73$ e $K_i = 0.08$. As respostas do controlador MPC e PI controlando a temperatura à 350 mm e 875 mm podem ser vistas na Figura 5.13.

Os índices de desempenho dos controladores são apresentados na Tabela 5.2.

Nos índices, um número menor representa melhor desempenho, assim, segundo o critério IAE, o MPC é melhor para o sensor à 875 mm e pior à 350 mm. Já de acordo com os critérios ITAE e $IA\Delta u$, o MPC é melhor em ambos os casos. O $IA\Delta y$ mostra um empate à 875 mm e melhor desempenho do PI à 350 mm, mas a diferença é pequena e pode-se considerar empate em ambos casos.

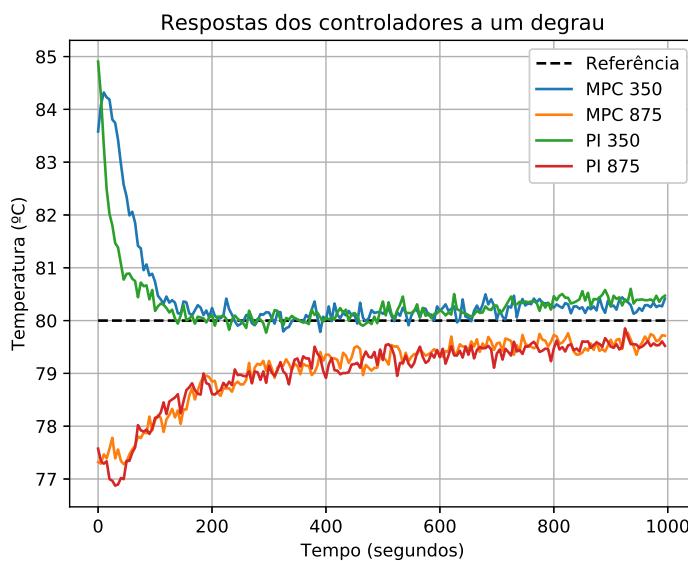


Figura 5.13 – Comparaçāo das respostas dos MPCs e PIIs

Tabela 5.2 – Índices de desempenho

	IAE	ITAE	$IA\Delta y$	$IA\Delta u$
MPC 350	1.32	1	1.09	1
PI 350	1	1.15	1	1.26
MPC 875	1	1	1	1
PI 875	1.05	1.09	1	1.01

5.7 Melhorias na plataforma

As principais modificações foram:

- implementação de um mecanismo de *backup* de todo o banco: o uso de múltiplos computadores na mesma planta pode ser facilitado com a opção de realizar backup de todo o banco. Isso também permite a liberação de espaço no sistema sem a perda de dados, já que pode-se guardar os dados antigos e apagar os testes, deixando a interface mais limpa;
- implementação de importação e exportação de configuração de hardware: também interessante quando várias pessoas utilizam o mesmo hardware. Mudanças, como novas calibrações, podem ser compartilhadas de forma simples e novos usuários podem começar a utilizar a planta rapidamente, apenas importando uma configuração já testada;

- seleção das variáveis a serem exibidas em gráficos: durante o desenvolvimento inicial não se pensou na possibilidade do usuário precisar salvar dezenas ou centenas de variáveis. Toda variável salva geraria um gráfico para o usuário, sempre. Durante os testes com casos reais, realizados por alunos de graduação e mestrado, observou-se a necessidade de limitar o número de gráficos gerados sem limitar o número de variáveis salvas. Isso foi feito colocando uma opção de selecionar quais gráficos serão exibidos;
- opção de clonar um teste ou controlador: durante os testes também percebeu-se a necessidade de duplicar um teste inteiro para modificar apenas um valor de variável, ou outra modificação pequena;
- melhoria da forma de apresentação de erros no terminal: os erros apresentados no terminal não eram explicativos, não mostrando, por exemplo, onde ocorreu o erro. Agora os erros trazem o escopo e a linha do erro, bem como a mensagem principal (erro mais relevante). Esse erro, na versão 1.0.11 do Lachesis, também é exibido para o usuário na interface, no componente Gráficos;
- melhoria de velocidade na recuperação de dados do banco: foram utilizadas funções disponíveis em versões mais recentes do MongoDB para melhorar a filtragem e transformação dos dados retornados. Esses eram feitos anteriormente em Python, com uma implementação mais lenta que aquela do banco de dados;
- inserção de um novo banco de dados: MySQL (necessário no Raspberry), sua necessidade veio da modificação anterior. Ao atualizar o banco de dados percebeu-se que as versões mais novas não mais suportam sistemas operacionais de 32bits, por isso foi necessário encontrar um banco de dados capaz de executar em um Raspberry Pi de 32bits. A seleção do MySQL se deu pela sua disponibilidade e atualização no Raspbian, sistema operacional do Raspberry e por sua implementação em linguagem C/C++. Outros bancos de dados não estruturados do tipo *document store* foram considerados, mas a falta de documentação, api pobre e/ou implementação em linguagem interpretada (mesmo que por máquina virtual) os fizeram menos interessantes;
- implementação de busca por drivers (ahio) em caminhos arbitrários: a biblioteca

ahio implementa alguns drivers padrão que estão presentes em seu código fonte. No entanto, faz-se necessário a implementação de drivers específicos, que não devem ser inseridos e distribuídos na biblioteca por não serem úteis em outros contextos, por exemplo, o driver do forno e os drivers utilizados pelos alunos Affonso Salomão e Bernardo Amin em seus TCCs. Para solucionar esse problema criou-se a possibilidade de procurar por drivers em caminhos informados pelo usuário, exportando-se funções para adicionar e remover caminhos do *PATH*. No aplicativo *moirai* foi criada a opção de se definir uma variável de ambiente denominada *AHIO_PATH* contendo caminhos separados por “:” (dois pontos), conforme padrão *POSIX*.

A lista completa pode ser vista nos *commits* de cada projeto, publicados desde 1º de março de 2018:

- Lachesis: <https://github.com/acristoffers/Lachesis/commits/master>
- moirai: <https://github.com/acristoffers/moirai/commits/master>
- ahio: <https://github.com/acristoffers/ahio/commits/master>

Considerações finais

Os controladores PI e MPC apresentaram respostas parecidas. No entanto, o controlador MPC obteve uma resposta até 27% melhor segundo o índice IVu, o que se traduz em economia de energia por menos mudanças no atuador. Já o ITAE mostra o MPC com uma vantagem de até 15% sobre o PI.

Estudos futuros poderiam utilizar outro sistema para melhor explorar os pontos chaves do MPC, que são a saturação de atuadores, sinal de saída e variação do sinal de controle, bem como o uso de múltiplas entradas e múltiplas saídas e de ordem elevadas sem modificação do *framework* do controlador. Esses casos não foram abordados pois o modelo de interesse, que é SISO de segunda ordem, foi linearizado longe da saturação do atuador.

A modelagem do sistema a parâmetros distribuídos via subsistemas interconectados possibilitou o controle da temperatura em um ponto onde não há sensor. Foi detectada a necessidade de melhorar o modelo, inserindo a temperatura ambiente como parâmetro de entrada, mas isso não foi impedimento para a realização do controle. A alternativa utilizada não se mostrou viável e recomenda-se que o modelo seja melhorado. Uma alternativa baseada em observadores também pode ser estudada.

Os observadores não funcionaram conforme esperado. Independente dos valores ajustáveis (R e Q) o estado estimado sempre converge para um valor e não mais varia junto com o estado medido, a não ser que a variação desse seja grande. O efeito é aquele de um filtro, mas ignorando a matriz de covariança. Assim, para trabalhos futuros, recomenda-se o estudo de desenvolvimento e implementação de observadores, visando, principalmente, resolver o problema encontrado no modelo.

A partir do trabalho aqui desenvolvido foi publicado um artigo na revista romena *Studies in Informatics and Control*, volume 27, número 3, de setembro de 2018 intitulado *Affordable Control Platform with MPC Application* (Plataforma de Controle de Baixo Custo com aplicação de MPC). O trabalho envolve duas das principais áreas desse trabalho, o controlador MPC e a plataforma de controle. O resumo do trabalho é apresentado, traduzido, a seguir.

Este artigo apresenta uma plataforma de controle desenvolvida para interfacear com vários hardwares, permitindo o design e a rápida implementação até mesmo de controladores avançados, tanto em sistemas acadêmicos quanto industriais. O código dos controladores é escrito na linguagem Python, de código aberto, facilitando a tradução de código normalmente escrito em software comercial. A plataforma proposta pode usar desde Arduinos até Computadores Lógicos Programáveis (PLCs). Além da pesquisa e testes em instalações industriais, a simplicidade da plataforma proposta permite seu uso também para fins educacionais e de treinamento. Portanto, a plataforma proposta pode ajudar os alunos a se concentrarem na análise de sistemas e na teoria de controle, em vez de problemas de interface de hardware, enquanto usam hardware de baixo custo. Desenvolvida em um esquema cliente-servidor, a plataforma pode ser executada em computadores acessíveis, aproveitando as ferramentas matemáticas e gráficas de alto nível disponíveis na linguagem Python, permitindo a rápida implementação de controladores avançados. O uso desta plataforma é ilustrado com a implementação de um controle preditivo modelo (MPC) de um controle de nível em um processo em escala de laboratório. Um PLC é usado para tomar as medidas de nível, para despachar sinais de controle e também para intertravar tarefas seguras. O controlador é executado em um computador Raspberry Pi que se comunica com o PLC por meio de um link Ethernet.

Referências

- BARROSO, N F. **Estratégia de monitoramento de sistemas distribuídos baseada em observadores do tipo Kalman.** 2017. Tese (Mestrado) – CEFET-MG.
- _____. **Instrumentação Virtual Aplicada à Automação de um Sistema Térmico para Experimentação Via Web.** 2015. TCC – CEFET-MG.
- BLACK, H. S. Stabilized Feedback Amplifiers. **Bell System Technical Journal**, v. 13, n. 1, p. 1–18, 1934. ISSN 15387305. DOI: [10.1002/j.1538-7305.1934.tb00652.x](https://doi.org/10.1002/j.1538-7305.1934.tb00652.x).
- BODE, Hendrik Wade. Feedback Amplifier Design. **Bell System Technical Journal**, v. 19, n. 1, p. 42, 1940.
- BRYSON, Arthur E.; HO, Yu-Chi; SIOURIS, George M. **Applied Optimal Control: Optimization, Estimation, and Control.** Revised. [S.l.]: Taylor & Francis, 1979. v. 9, p. 366–367. ISBN 9780891162285. DOI: [10.1109/TSMC.1979.4310229](https://doi.org/10.1109/TSMC.1979.4310229).
- CAIRANO, S Di. An industry perspective on MPC in large volumes applications: Potential Benefits and Open Challenges. In: PROC. 4th IFAC Nonlinear Model Predictive Control ... [S.l.: s.n.], 2012.
- CALDEIRA, André. **Techniques d'analyse de stabilité et synthèse de contrôle pour des systèmes hyperboliques.** Mar. 2017. Tese (Doutorado).
- CHEN, Dan; SEBORG, Dale E. PI/PID Controller Design Based on Direct Synthesis and Disturbance Rejection. American Chemical Society, 2002. DOI: [10.1021/IE010756M](https://doi.org/10.1021/IE010756M).
- CUTLER, C.R.; RAMAKER, B.L. Dynamic matrix control- A computer control algorithm. **Joint Automatic Control Conference**, v. 1, 1980.

- DORF, Richard C.; BISHOP, Robert H. **Modern Control Systems**. [S.l.]: Pearson, 2010. p. 1104. ISBN 9780136024583.
- KALMAN, R. E. A New Approach to Linear Filtering and Prediction Problems. **Journal of Basic Engineering**, American Society of Mechanical Engineers, v. 82, n. 1, p. 35, mar. 1960. ISSN 00219223. DOI: 10.1115/1.3662552.
- LIBERTY, S. **Modern control engineering**. [S.l.]: Pearson, 1972. v. 17, p. 419–419. ISBN 0136156738. DOI: 10.1109/TAC.1972.1100013. arXiv: 0605511 [cond-mat].
- LYAPUNOV, Aleksandr Mikhailovich. Problème Général de la Stabilité du Mouvement. **Annales de la Faculté des sciences de Toulouse : Mathématiques**, GAUTHIER-VILLARS, IMPRIMEUR-EDITEUR ; ED. PRIVAT, IMPRIMEUR-LIBRAIRE, v. 9, p. 203–474, 1892.
- MORARI, Manfred; LEE, Jay H. Model predictive control: past, present and future. **Computers I& Chemical Engineering**, v. 23, n. 4–5, p. 667–682, 1999. ISSN 0098-1354. DOI: 10.1016/S0098-1354(98)00301-9.
- PATWARDHAN, Sachin C. **A Gentle Introduction to Model Predictive Control (MPC) Formulations based on Discrete Linear State Space Models**. [S.l.: s.n.], 2014.
- SILVA, J V Valle. **Modelagem Matemática a parâmetros distribuídos e Controle em Malha Fechada de um sistema de aquecimento de ar**. [S.l.: s.n.], 2014.
- TICLEA, Alexandru; BESANÇON, Gildas. State and parameter estimation via discrete-time exponential forgetting factor observer. **IFAC Proceedings Volumes**, Elsevier, v. 42, n. 10, p. 1370–1374, jan. 2009. ISSN 1474-6670. DOI: 10.3182/20090706-3-FR-2004.00228.
- WANG, Liuping. **Model Predictive Control System Design and Implementation Using MATLAB**. 1. ed. [S.l.]: Springer-Verlag London, 2009. p. 403. (Advances in Industrial Control). ISBN 9781848823303. DOI: 10.1007/978-1-84882-331-0. arXiv: arXiv:1011.1669v3.
- ZHANG, Xi. **Fast MPC Solvers for Systems with Hard Real-Time Constraints**. [S.l.: s.n.], 2016. p. 86.